Springer Series in Statistics

Advisors:

P. Bickel, P. Diggle, S. Fienberg, U. Gather,
I. Olkin, S. Zeger

Springer Series in Statistics

Alho/Spencer: Statistical Demography and Forecasting
Andersen/Borgan/Gill/Keiding: Statistical Models Based on Counting Processes
Atkinson/Riani: Robust Diagnostic Regression Analysis
Atkinson/Riani/Ceriloi: Exploring Multivariate Data with the Forward Search
Berger: Statistical Decision Theory and Bayesian Analysis, 2nd edition
Berk: Statistical Learning from a Regression Perspective
Borg/Groenen: Modern Multidimensional Scaling: Theory and Applications, 2nd edition
Brockwell/Davis: Time Series: Theory and Methods, 2nd edition
Bucklew: Introduction to Rare Event Simulation
Cappé/Moulines/Rydén: Inference in Hidden Markov Models
Chan/Tong: Chaos: A Statistical Perspective
Chen/Shao/Ibrahim: Monte Carlo Methods in Bayesian Computation
Coles: An Introduction to Statistical Modeling of Extreme Values
Devroye/Lugosi: Combinatorial Methods in Density Estimation
Diggle/Ribeiro: Model-based Geostatistics
Dudoit/Van der Laan: Multiple Testing Procedures with Applications to Genomics
Efromovich: Nonparametric Curve Estimation: Methods, Theory, and Applications
Eggermont/LaRiccia: Maximum Penalized Likelihood Estimation, Volume I: Density Estimation
Fahrmeir/Tutz: Multivariate Statistical Modeling Based on Generalized Linear Models, 2nd edition
Fan/Yao: Nonlinear Time Series: Nonparametric and Parametric Methods
Ferraty/Vieu: Nonparametric Functional Data Analysis: Theory and Practice
Ferreira/Lee: Multiscale Modeling: A Bayesian Perspective
Fienberg/Hoaglin: Selected Papers of Frederick Mosteller
Frühwirth-Schnatter: Finite Mixture and Markov Switching Models
Ghosh/Ramamoorthi: Bayesian Nonparametrics
Glaz/Naus/Wallenstein: Scan Statistics
Good: Permutation Tests: Parametric and Bootstrap Tests of Hypotheses, 3rd edition
Gouriéroux: ARCH Models and Financial Applications
Gu: Smoothing Spline ANOVA Models
Györfi/Kohler/Krzyźak/Walk: A Distribution-Free Theory of Nonparametric Regression
Haberman: Advanced Statistics, Volume I: Description of Populations
Hall: The Bootstrap and Edgeworth Expansion
Härdle: Smoothing Techniques: With Implementation in S
Harrell: Regression Modeling Strategies: With Applications to Linear Models, Logistic Regression, and Survival Analysis
Hart: Nonparametric Smoothing and Lack-of-Fit Tests
Hastie/Tibshirani/Friedman: The Elements of Statistical Learning: Data Mining, Inference, and Prediction
Hedayat/Sloane/Stufken: Orthogonal Arrays: Theory and Applications
Heyde: Quasi-Likelihood and its Application: A General Approach to Optimal Parameter Estimation
Huet/Bouvier/Poursat/Jolivet: Statistical Tools for Nonlinear Regression: A Practical Guide with S-PLUS and R Examples, 2nd edition

(continued after index)

Richard A. Berk

Statistical Learning from a Regression Perspective

Springer

Richard A. Berk
Department of Criminology, School of Arts and Sciences
Department of Statistics, The Wharton School
The University of Pennsylvania
Philadelphia, PA
USA

ISBN: 978-0-387-77500-5 e-ISBN: 978-0-387-77501-2
DOI: 10.1007/978-0-387-77501-2

Library of Congress Control Number: 2008926886

Printed on acid-free paper

9 8 7 6 5 4 3 2 1

springer.com

In memory of Peter H. Rossi, a mentor, colleague, and friend.

“In God we trust. All others must have data.”
(W. Edwards Deming)

Preface

As I was writing my recent book on regression analysis (Berk, 2003), I was struck by how few alternatives to conventional regression there were. In the social sciences, for example, one either did causal modeling econometric style or largely gave up quantitative work. The life sciences did not seem quite so driven by causal modeling, but causal modeling was a popular tool. As I argued at length in my book, causal modeling as commonly undertaken is a loser.

There also seemed to be a more general problem. Across a range of scientific disciplines there was too often little interest in statistical tools emphasizing induction and description. With the primary goal of getting the "right" model and its associated p-values, the older and interesting tradition of exploratory data analysis had largely become an under-the-table activity; the approach was in fact commonly used, but rarely discussed in polite company. How could one be a real scientist, guided by "theory" and engaged in deductive model testing, while at the same time snooping around in the data to determine which models to test? In the battle for prestige, model testing had won.

Around the same time, I became aware of some new developments in applied mathematics, computer sciences, and statistics making data exploration a virtue. And with the virtue came a variety of new ideas and concepts, coupled with the very latest in statistical computing. These new approaches, variously identified as "data mining," "statistical learning," "machine learning," and other names, were being tried in a number of the natural and biomedical sciences, and the initial experience looked promising.

As I started to read more deeply, however, I was struck by how difficult it was to work across writings from such disparate disciplines. Even when the material was essentially the same, it was very difficult to tell if it was. Each discipline brought it own goals, concepts, naming conventions, and (maybe worst of all) notation to the table.

In the midst of trying to impose some of my own order on the material, I came upon *The Elements of Statistical Learning*, by Trevor Hastie, Robert

Tibshirani, and Jerome Friedman (Springer-Verlag, 2001). I saw in the book a heroic effort to integrate a very wide variety of data analysis tools. I learned from the book and was then able to approach more primary material within a useful framework.

This book is my attempt to integrate some of the same material and some new developments of the past six years. Its intended audience is practitioners in the social, biomedical, and ecological sciences. Applications to real data addressing real empirical questions are emphasized. Although considerable effort has gone into providing explanations of why the statistical procedures work the way they do, the required mathematical background is modest. A solid course or two in regression analysis and some familiarity with resampling procedures should suffice. A good benchmark for regression is Freedman's *Statistical Models: Theory and Practice* (2005). A good benchmark for resampling is Manly's *Randomization, Bootstrap, and Monte Carlo Methods in Biology* 1997. Matrix algebra and calculus are used only as languages of exposition, and only as needed. There are no proofs to be followed.

The procedures discussed are limited to those that can be viewed as a form of regression analysis. As explained more completely in the first chapter, this means concentrating on statistical tools for which the conditional distribution of a response variable is the defining interest and for which characterizing the relationships between predictors and the response is undertaken in a serious and accessible manner.

Regression analysis provides a unifying theme that will ease translations across disciplines. It will also increase the comfort level for many scientists and policy analysts for whom regression analysis is a key data analysis tool. At the same time, a regression framework will highlight how the approaches discussed can be seen as alternatives to conventional causal modeling.

Because the goal is to convey how these procedures can be (and are being) used in practice, the material requires relatively in-depth illustrations and rather detailed information on the context in which the data analysis is being undertaken. The book draws heavily, therefore, on datasets with which I am very familiar. The same point applies to the software used and described.

The regression framework comes at a price. A 2005 announcement for a conference on data mining sponsored by the Society for Industrial and Applied Mathematics (SIAM) listed the following topics: query/constraint-based data mining, trend and periodicity analysis, mining data streams, data reduction/preprocessing, feature extraction and selection, postprocessing, collaborative filtering/personalization, cost-based decision making, visual data mining, privacy-sensitive data mining, and lots more. Many of these topics cannot be considered a form of regression analysis. For example, procedures used for edge detection (e.g., determining the boundaries of different kinds of land use from remote sensing data) are basically a filtering process to remove noise from the signal.

Another class of problems makes no distinction between predictors and responses. The relevant techniques can be closely related, at least in spirit, to

procedures such as factor analysis and cluster analysis. One might explore, for example, the interaction patterns among children at school: who plays with whom. These too are not discussed.

Other topics can be considered regression analysis only as a formality. For example, a common data mining application in marketing is to extract from the purchasing behavior of individual shoppers patterns that can be used to forecast future purchases. But there are no predictors in the usual regression sense. The conditioning is on each individual shopper. The question is not what features of shoppers predict what they will purchase, but what a given shopper is likely to purchase.

Finally, there are a large number of procedures that focus on the conditional distribution of the response, much as with any regression analysis, but with little attention to how the predictors are related to the response (Horváth and Yamamoto, 2006; Camacho et al., 2006). Such procedures neglect a key feature of regression analysis, at least as discussed in this book, and are not considered. That said, there is no principled reason in many cases why the role of each predictor could not be better represented, and perhaps in the near future that shortcoming will be remedied.

In short, although using a regression framework implies a big-tent approach to the topics included, it is not an exhaustive tent. Many interesting and powerful tools are not discussed. Where appropriate, however, references to that material are provided.

I may have gone a bit overboard with the number of citations I provide. The relevant literatures are changing and growing rapidly. Today's breakthrough can be tomorrow's bust, and work that by current thinking is uninteresting can be the spark for dramatic advances in the future. At any given moment, it can be difficult to determine which is which. In response, I have attempted to provide a rich mix of background material, even at the risk of not being sufficiently selective. (And I have probably missed some useful papers nevertheless.)

In the material that follows, I have tried to use consistent notation. This has proved to be very difficult because of important differences in the conceptual traditions represented and the complexity of statistical tools discussed. For example, it is common to see the use of the expected value operator even when the data cannot be characterized as a collection of random variables and when the sole goal is description.

I draw where I can from the notation used in *The Elements of Statistical Learning* (Hastie et al., 2001). Thus, the symbol X is used for an input variable, or predictor in statistical parlance. When X is a set of inputs to be treated as a vector, each component is indexed by a subscript (e.g., X_j). Quantitative outputs, also called response variables, are represented by Y, and categorical outputs, another kind of response variable, are represented by G with K categories. Upper case letters are used to refer to variables in a general way, with details to follow as needed. Sometimes these variables are

treated as random variables, and sometimes not. I try to make that clear in context.

Observed values are shown in lower case, usually with a subscript. Thus x_i is the ith observed value for the variable X. Sometimes these observed values are nothing more than the data on hand. Sometimes they are realizations of random variables. Again, I try to make this clear in context.

Matrices are represented in bold uppercase. For example, in matrix form the usual set of p predictors, each with N observations, is an $N \times p$ matrix $\mathbf{X}$. The subscript i is generally used for observations and the subscript j for variables. Bold lowercase letters are used for vectors with N elements, commonly columns of $\mathbf{X}$. Other vectors are generally not represented in boldface fonts, but again, I try to make this clear in context.

If one treats Y as a random variable, its observed values y are either a random sample from a population or a realization of a stochastic process. The conditional means of the random variable Y for various configurations of $\mathbf{X}$-values are commonly referred to as "expected values," and are either the conditional means of Y for different configurations of $\mathbf{X}$-values in the population or for the stochastic process by which the data were generated. A common notation is $\mathrm{E}(\mathrm{Y}|\mathbf{X})$. The $\mathrm{E}(\mathrm{Y}|\mathbf{X})$ is also often called a "parameter." The conditional means computed from the data are often called "sample statistics," or in this case,"sample means." In the regression context, the sample means are commonly referred to as the fitted values, often written as $\hat{y}|\mathbf{X}$. Subscripting can follow as already described.

Unfortunately, after that it gets messier. First, I often have to decipher the intent in the notation used by others. No doubt I sometimes get it wrong. For example, it is often unclear if a computer algorithm is formally meant to be an estimator or a descriptor.

Second, there are some complications in representing nested realizations of the same variable (as in the bootstrap), or model output that is subject to several different chance processes. There is a practical limit to the number and types of bars, asterisks, hats, and tildes one can effectively use. I try to provide warnings (and apologies) when things get cluttered.

There are also some labeling issues. When I am referring to the general linear model (i.e., linear regression, analysis of variance, and analysis of covariance), I use the terms classical linear regression, or conventional linear regression. All regressions in which the functional forms are determined before the fitting process begins, I call parametric. All regressions in which the functional forms are determined as part of the fitting process, I call nonparametric. When there is some of both, I call the regressions semiparametric. Sometimes the lines among parametric, nonparametric, and semiparametric are fuzzy, but I try to make clear what I mean in context. Although these naming conventions are roughly consistent with much common practice, they are not universal.

All of the computing done for this book was undertaken in R. R is a programming language designed for statistical computing and graphics. It has

become a major vehicle for developmental work in statistics and is increasingly being used by practitioners. A key reason for relying on R for this book is that most of the newest developments in statistical learning and related fields can be found in R. Another reason is that it is free.

Readers familiar with S or S-plus will immediately feel at home; R is basically a "dialect" of S. For others, there are several excellent books providing a good introduction to data analysis using R. Dalgaard (2002), Crawley (2007), and Maindonald and Braun (2007) are all very accessible. Readers who are especially interested in graphics should consult Murrell (2006). The most useful R website can be found at http://www.r-project.org/.

The use of R raises the question of how much R-code to include. The R-code used to construct all of the applications in the book could be made available. However, detailed code is largely not shown. Many of the procedures used are somewhat in flux. Code that works one day may need some tweaking the next. As an alternative, the procedures discussed are identified as needed so that detailed information about how to proceed in R can be easily obtained from R help commands or supporting documentation. In addition, there is a web site where many examples, including the data, can be found (WEB ADDRESS TO BE ADDED). When the data used in this book are proprietary or otherwise not publicly available, similar data and appropriate R-code are substituted.

There are exercises at the end of each chapter. They are meant to be hands-on data analyses built around R. As such, they require some facility with R. However, the goals of each problem are reasonably clear so that other software and datasets can be used. Often the exercises can be usefully repeated with different datasets.

The book has been written so that later chapters depend substantially on earlier chapters. For example, because classification and regression trees (CART) can be an important component of boosting, it may be difficult to follow the discussion of boosting without having read the earlier chapter on CART. However, readers who already have a solid background in material covered earlier should have little trouble skipping ahead. The notation and terms used are reasonably standard or can be easily figured out. In addition, the final chapter can be read at almost any time. One reviewer suggested that much of the material could be usefully brought forward to Chapter 1.

Finally, there is the matter of tone. The past several decades have seen the development of a dizzying array of new statistical procedures, sometimes introduced with the hype of a big-budget movie. Advertising from major statistical software providers has typically made things worse. Although there have been genuine and useful advances, none of the techniques have ever lived up to their most optimistic billing. Widespread misuse has further increased the gap between promised performance and actual performance. In this book, therefore, the tone will be cautious, some might even say dark. I hope this will not discourage readers from engaging seriously with the material. The intent

is to provide a balanced discussion of the limitations as well as the strengths of the statistical learning procedures.

While working on this book, I was able to rely on support from several sources. Much of the work was funded by a grant from the National Science Foundation: SES-0437169, "Ensemble Methods for Data Analysis in the Behavioral, Social and Economic Sciences." The first draft was completed while I was on sabbatical at the Department of Earth, Atmosphere, and Oceans, at the Ecole Normale Supérieur in Paris. The second draft was completed after I moved from UCLA to the University of Pennsylvania. All three locations provided congenial working environments. Most important, I benefited enormously from discussions about statistical learning with colleagues at UCLA, Penn and elsewhere: Larry Brown, Andreas Buja, Jan de Leeuw, David Freedman, Mark Hansen, Andy Liaw, Greg Ridgeway, Bob Stine, Mikhail Traskin and Adi Wyner. Each is knowledgeable, smart and constructive. I also learned a great deal from several very helpful, anonymous reviews. Dick Koch was enormously helpful and patient when I had problems making TeXShop perform properly. Finally, I have benefited over the past several years from interacting with talented graduate students: Yan He, Weihua Huang, Brian Kriegler, and Jie Shen. Brian Kriegler deserves a special thanks for working through the exercises at the end of each chapter.

Certain datasets and analyses were funded as part of research projects undertaken for the California Policy Research Center, The Inter-America Tropical Tuna Commission, the National Institute of Justice, the County of Los Angeles, the California Department of Correction and Rehabilitation, the Los Angeles Sheriff's Department, and the Philadelphia Department of Adult Probation and Parole. Support from all of these sources is gratefully acknowledged.

Contents

1

Statistical Learning as a Regression Problem

1.1 Getting Started

As a first approximation, one can think of statistical learning as the "muscle car" version of Exploratory Data Analysis (EDA). Just as in EDA , the data are approached with relatively little prior information and examined in a highly inductive manner. Knowledge discovery can be a key goal. But thanks to the enormous developments in computing power and computer algorithms over the past two decades, it is possible to extract information that would have previouslybeen inaccessible. In addition, because statistical learning has evolved in a number of different disciplines, its goals and approaches are far more varied than conventional EDA.

In this book, the focus is on statistical learning procedures that can be understood within a regression framework. For a wide variety of applications, this will not pose a significant constraint and will greatly facilitate the exposition. The researchers in statistics, applied mathematics and computer science responsible for most statistical learning techniques often employ their own distinct jargon and have a penchant for attaching cute, but somewhat obscure, labels to their products: bagging, boosting, bundling, random forests, the lasso, and others. There is also widespread use of acronyms: CART, MARS, MART, LARS, and many more. A regression framework provides a convenient and instructive structure in which these procedures can be more easily understood.

After a brief discussion of how statisticians think about regression analysis, the chapter introduces a number of key concepts and raises broader issues that reappear in later chapters. It may be a little difficult for some readers to follow parts of the discussion, or its motivation, the first time around. However, later chapters will flow far better with some this preliminary material on the table, and readers are encouraged to return to this chapter as needed.

R.A. Berk, *Statistical Learning from a Regression Perspective*,
DOI: 10.1007/978-0-387-77501-2_1,

1.2 Setting the Regression Context

We begin with a brief consideration of what regression analysis is. A knee-jerk response in many academic disciplines and policy applications may be to equate regression analysis with causal modeling. This is too narrow and even misleading. Causal modeling is actually an interpretive framework that is imposed on the results of a regression analysis. An alternative knee-jerk response may be to equate regression analysis with the general linear model. At most, the general linear model can be seen as a special case of regression analysis.

Statisticians commonly define regression so that the goal is to understand "as far as possible with the available data how the conditional distribution of some response **y** varies across subpopulations determined by the possible values of the predictor or predictors" (Cook and Weisberg, 1999: 27). That is, interest centers on the distribution of the response variable Y conditioning on one or more predictors X.

This definition includes a wide variety of elementary procedures easily implemented in R. (See, for example, Maindonald and Braun, 2007: Chapter 2.) For example, consider Figures 1.1 and 1.2. The first shows the distribution of SAT scores for recent applicants to a major university, who self-identify as "Hispanic." The second shows the distribution of SAT scores for recent applicants to that same university, who self-identify as "Asian."

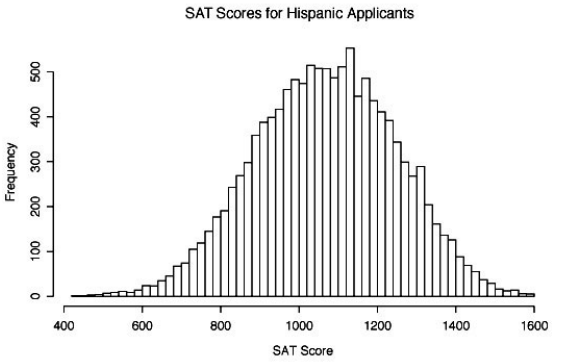


Fig. 1.1. Distribution of SAT scores for Hispanic applicants.

It is clear that the two distributions differ substantially. The Asian distribution is shifted to the right, leading to a distribution with a higher mean (1227 compared to 1072), a smaller standard deviation (170 compared to 180), and greater skewing. A comparative description of the two histograms alone constitutes a proper regression analysis. Using various summary statistics, some key features of the two displays are compared and contrasted (Berk, 2003: Chapter 1).

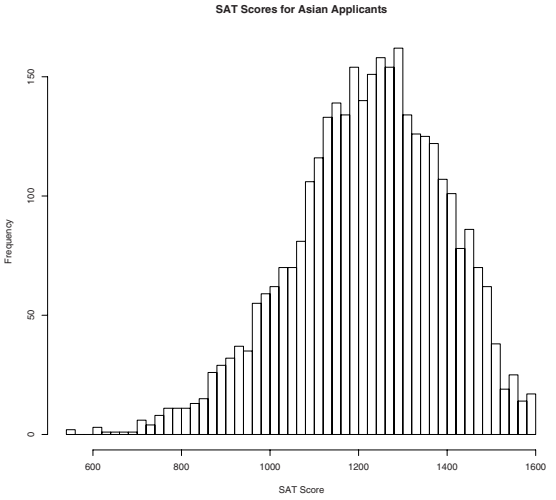


Fig. 1.2. Distribution of SAT scores for Asian applicants.

Should one be especially interested in a comparison of the means, one could proceed descriptively with a conventional least squares regression analysis as a special case. That is, for each observation i, one could let

$$\hat{y}_i = \beta_0 + \beta_1 x_i, \tag{1.1}$$

where the response variable y_i is each applicant's SAT score, x_i is an indicator variable coded "1" if the applicant is Asian and "0" if the applicant is Hispanic, β_0 is the mean SAT score for Hispanic applicants, β_1 is how much larger (or smaller) the mean SAT score for Asian applicants happens to be, and i is an index running from 1 to the number of Hispanic and Asian applicants, N. Here, $\beta_0 = 1072$ and $\beta_1 = (1227 - 1072) = 155$.

One can reverse the roles of the two variables and undertake another legitimate kind of regression analysis. Figure 1.3 shows a scatterplot with the SAT score on the horizontal axis and on the vertical axis an indicator variable coded "1" if the applicant self-identifies as Asian and "0" if the applicant self-identifies as Hispanic. The points in the plot have been jittered vertically to make the scatterplot easier to to read. Jittering adds a bit of noise to each observation, in this case for the Asian indicator variable.

Because the higher points in Figure 1.3 (around 1.0) are to the right of the lower points (around 0.0), the proportion of Asians increases moving from left to right. That is, the conditional proportion increases with SAT score. A least squares regression line overlaid on the scatterplot can quantify the association. It is of the same form as Equation 1.1 with the roles of Y and X exchanged. The slope, β_1, indicates that for each additional 100 SAT points, Asian representation, compared to Hispanics, increases about 8% on the av-

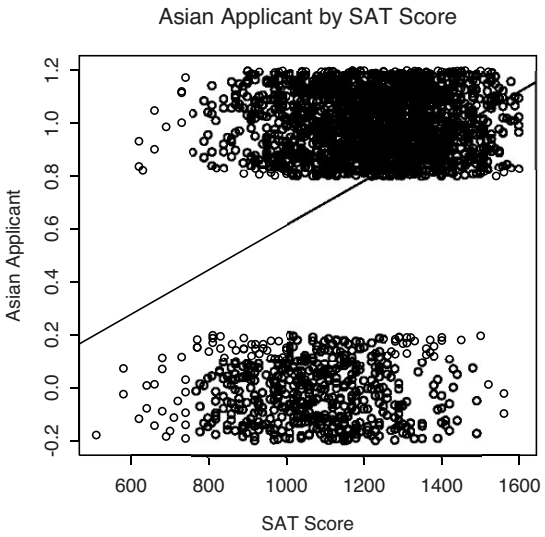


Fig. 1.3. Asian Applicants by SAT Score

erage. The intercept, β_0, is in this case a negative number, indicating that using a straight line may not be the best way to describe the relationship. An S-shaped function such as the logistic curve might do a better job. Moving to logistic regression would have still been a regression analysis.

There is no requirement that either variable be measured on an equal interval scale. Both variables can be categorical. Table 1.1 shows a cross-tabulation, using those same college applicants, for whether the applicant self-identifies as African-American and whether the applicant falls within a special admissions category of "athlete." The athlete designation usually places the applicant in a special pool that only includes other athletes.

It is readily apparent from the marginal distributions that athlete applicants and African-American applicants represent very small fractions of the total number of applicants (about .6% and 4.6%, respectively). One can also see from the within-row percentages that 2.1% of all African-American applicants are designated as athletes whereas around .5% of all other applicants are. This is a difference of 1.6%. Stated differently, African-American applicants are over four times more likely to be placed in the athlete applicant pool.

Table 1.1 is another example of a regression analysis. The proportion placed in the athlete applicant pool is computed conditional on whether the applicant self-identifies as African-American. And as before, one can arrive at the very same results with a least squares regression analysis. In Equation 1.1, the response y_i is an indicator variable coded "1" if the applica-

	Not an Athlete (%)	Athlete (%)	Row Percentage
Not Black	99.5	0.5	95.3
Black	97.9	2.1	4.6
Column Percentage	99.4	0.6	100=96,277

Table 1.1. Ethnicity by athlete designation.

tion is placed in the athlete pool and coded "0" otherwise. The explanatory variable x_i is an indicator variable coded "1" if the applicant is self-identifies as African-American and "0" if not. The value of β_0 is then .005, and $\beta_1 = .021 - .005 = .016$.

Figure 1.4 displays the most familiar kind of plot for regression analysis. The SAT score is plotted against an applicant's household income, with the regression line overlaid. Both variables are on an equal interval scale. The scatterplot was constructed for a random sample of 500 applicants to make the graph more legible.

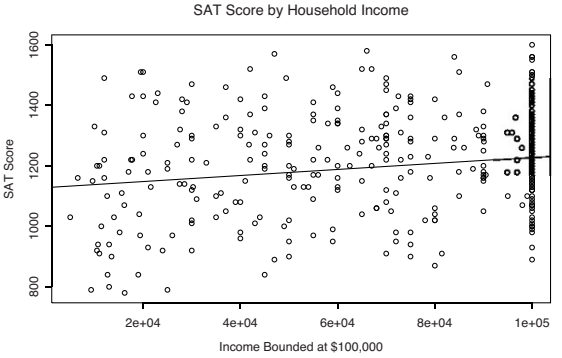


Fig. 1.4. SAT scores by family income.

As one moves from left to right, one can see how the conditional mean of the SAT score (y_i) changes with the household income (x_i). The value of β_0 is about 1200, which is the mean SAT score for households with no reported income. In this instance, it is not clear that the value of β_0 makes any substantive sense, but it is needed to locate the regression line. For each $1000 of income, the mean SAT score increases about 1 point. But note that families with more than $100,000 of income are treated as having no more than $100,000 of income (because of the way the application forms are filled out). Consequently, the slope would be perhaps a little steeper than had the full range of income figures been available.

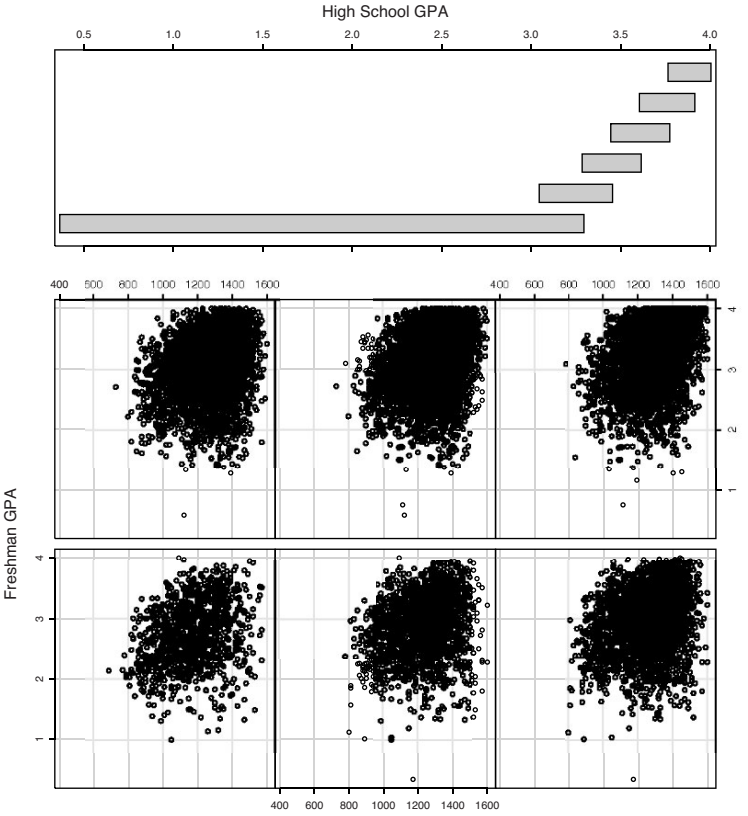


Fig. 1.5. Freshman GPA on SAT holding high school GPA constant.

Each of the four examples could easily have included more than one predictor. For example, Figure 1.5 shows a conditioning plot, sometimes called a "coplot" (Cleveland, 1993: 182–190). There are scatterplots of the Grade Point Average (GPA) of college freshmen, from that same university, against their SAT scores, holding constant their high school GPA. The scatterplots are read left to right starting with the bottom row. For the second row, one starts with the first plot on the left side. The conditioning subsets, defined by high school GPA, are shown in the top panel. The spans of the conditioning variable and amount of overlap between the subsets have been tuned to allow for a sufficient number of observations in each.

One can see that there is a positive association between SAT score and freshman GPA, within ranges of high school GPA. Holding the high school GPA approximately constant, the SAT score is related to performance in the first year of college. One can also see that the plots shift upward as one moves from the lower-left corner to the upper-right corner, indicating that the

high school GPA is also positively related to performance in the first year of college, holding the SAT score constant. Note, for instance, that the vertical slice of points at an SAT score of 1200 rises toward the upper boundary of the plots. Finally, there is apparently a ceiling effect of the 4.0 upper limit for freshman GPA in the top three graphs, implying that the relationship between SAT score and freshman GPA would probably be stronger if students who performed especially well could receive grades higher than 4.0. More precise statements of these sorts could be made by adding a second predictor to a regression equation of the form shown in Equation 1.1.

There are several broad lessons in these initial illustrations. First, regression analysis seeks to characterize conditional distributions. The response variable and the predictors can be categorical or quantitative variables. That's the long and the short of it.

Second, within that definition, one is free to choose whatever procedures seem to be the most useful. Graphs, for instance, are not automatically better or worse than numerical summaries, and a wide variety of each can be helpful, at least in principle. The choice depends on the nature of the information to be extracted from the data and the audience for the results. For example, graphs can be more effective than numerical summaries when broad patterns in the data are more important than a few precise values. If numerical summaries are desirable, there are no necessary restrictions on the functional forms used or on how the numbers are computed. Classical linear regression is just a special case. Regression analysis is a "big-tent" procedure.

Third, although analyses such as these may reflect cause-and-effect relationships or motivate a search for causal explanations, there is nothing in a regression analysis that requires inferences about cause and effect. And there is certainly no requirement that the regression analysis be formulated as a "causal model" in which the causal mechanisms by which the data were produced are explicitly represented (Berk, 2003: Chapter 1). The fact that regression equations are often advertised as causal models does not mean the two are the same.

Fourth, there is also nothing in regression analysis that requires statistical inference: formal tests of null hypotheses or confidence intervals. These can sometimes be very useful but go beyond the definition of a regression analysis (Berk, 2003: Chapter 1). Theyare an add-on. Moreover, even if the data are generated in a manner that can justify statistical inference, there are real questions about how to interpret the p-values that result after an extensive exploratory analysis. In general, the p-values will be too small, sometimes dramatically so. A bit more is said about this shortly in the context of "data snooping." In later chapters, a more detailed discussion is undertaken under the rubric of overfitting.

Finally, a regression analysis can serve a variety of purposes. Most directly, a regression analysis can be used to describe the relationships between variables. For example, Figure 1.5 addresses the nature of the association between freshman GPA and SAT score, holding high school GPA constant. Insofar as

a relationship can be found, it may also serve as the basis for useful forecasting. For instance, if SAT is an effective predictor of later performance in college, beyond what one might learn from a student's GPA in high school, an SAT score might be an important piece of information to use in an admission decision. And, although a regression analysis by itself is silent on cause and effect, it can under some circumstances be applied to characterize relationships taken to be causal because of additional information about how the data were generated. In perhaps the best situation, a regression analysis is applied to data from a randomized experiment in which one or more interventions are consciously manipulated, and the goal is to estimate a "contrast" with respect to the outcome for different treatment groups. For example, although a person's race cannot be manipulated, the race recorded in a home mortgage application can be. Then one can estimate the causal effect of a racial label on the interest rate offered, other things being equal. (Studies like this have been done.) It cannot be overemphasized that causal inference comes from knowledge about the experiment, not from the regression analysis. The regression analysis merely describes relationships that are already demonstrably causal.

1.3 The Transition to Statistical Learning

Statistical learning within a regression framework retains the focus on the conditional distribution of a response variable with respect to one or more predictors. Various features of that conditional distribution can be relevant, but the conditional mean will play a central role. How does the conditional mean of the response vary depending on the values of its predictors?

Where statistical learning can differ from conventional linear regression is in how that conditional relationship comes to be characterized. In conventional linear regression, functional forms linking the predictors to the response are determined before the fitting process begins. The same is true of the generalized linear model (e.g., logistic regression and Poisson regression) and conventional nonlinear regression. In that sense, all of these procedures can be called parametric.

In statistical learning, there is far less reliance on prior information when functional forms are determined to link predictors to the response. Although there will sometimes be constraints on the kinds of functions permitted, the functional forms are, by and large, arrived at inductively from the data. In that sense, statistical learning procedures can be called nonparametric.

Statistical learning is likely to shine when the functional forms are unknown and substantially nonlinear. Readers familiar with stepwise regression already have some appreciation for the look and feel of several statistical learning features. Readers familiar with smoothers can probably anticipate that smoothing a scatterplot can be a form of statistical learning. The use

of the word "learning" is a metaphor for the exploratory manner in which relationships between variables are determined.

1.3.1 Some Goals of Statistical Learning

As with all statistical procedures, statistical learning necessarily raises a number of "meta-issues." We need to consider these over the next several pages so that the key features of the context and underpinnings of statistical learning are familiar. Parts may seem a little abstract but they lay the foundation for the more nuts-and-bolts material that follow. Some of the material may benefit from rereading after later chapters have been read.

As with any statistical procedure, the goals of statistical learning depend fundamentally on how the data were generated. It is often useful to think about the data on hand as the product of a specific data-generation process, also sometimes called a data-generation mechanism . The data-generation process is a product of natural forces and the activities of researchers.

An example of data generated primarily by natural forces might be a time series of air quality measures in a particular metropolitan area. An example of data generated primarily by researchers might be a clinical trial for a new cancer treatment. An example of data generated by a rich mix of the two might be a probability sample of registered voters in which voting preferences are reported. The degree to which researchers intervene in a natural process determines how much of the data-generation process will be characterized by research protocols such as random assignment or probability sampling.

A conceptual distinction is often made between two kinds of data-generation processes. One kind conceives of a stochastic process with the observations on hand a realization of that real process. For example, suppose that there are $i = 1, 2, \ldots, N$ observations in a dataset. Nature generates each observed value of y_i using, say, of $y_i = 5 + 3x_i + 1.5z_i + \varepsilon_i$, where x_i and z_i are predictors, and the value ε_i behaves as if drawn from a single distribution with a mean of 0.0, independently of any other ε_j, and independently of x_i and z_i. The systematic part of the data-generation process is $5 + 3x_i + 1.5z_i$. This is usually treated as fixed. The stochastic part is ε_i. Chance is built in solely through ε_i so that y_i is a random variable. Sometimes the distribution from which ε_i is drawn is said to be of a particular form such as the normal. In more formal terms: $\varepsilon_i \sim NIID(0, \sigma^2)$.

Another kind of data-generation process assumes that there exists a population of potential observations. Suppose that in this population there are, again, three variables. There is a response variable y_i and two predictors x_i and z_i. All three variables in the population are fixed; there is no stochastic component. If one were able to observe all of the variables for all elements in the population over and over, their values would not change. For each possible configuration of values for x_i and z_i, there is a mean value for the response. Nature computes these means using, for instance, $5 + 3x_i + 1.5z_i$, where for

each mean, the index i is limited to those observations with the same values for x_i and z_i.

Commonly, there will be variation in the values of y_i around each conditional mean. The values of y_i around each conditional mean are unrelated to the values of the predictors, and are sometimes said to have a particular distribution, often the normal. A probability sample of size N is taken from the population. The three variables are now random variables. If a second probability sample were drawn, many (or even all) of the values for each of the variables would be different. In the sample, one can write, as before, $y_i = 5 + 3x_i + 1.5z_i + \varepsilon_i$, where $\varepsilon_i \sim NIID(0, \sigma^2)$. But now, ε_i results from the probability sampling, not from nature. It is common to treat the predictors as fixed, once they materialize in a given sample. In short, both kinds of data-generation processes can lead to the same formal expression of how y_i came to be.

Whether the data are a realization or a probability sample, an important goal can be to estimate from the data on hand how the two predictors are related to the response. For the stochastic process, that would imply trying to accurately represent the systematic component of y_i. For the probability sample, that would imply trying to accurately represent how the conditional means in the population are constructed from the predictors. Thus, both such enterprises are really the same. Indeed, effectively the same statistical tools can be used whether the data are treated as a realization of a stochastic process or a probability sample drawn from a population. Nevertheless, the two accounts can have different implications for the credibility of any subsequent analysis. Assumptions built into the data analysis need to be justified by a credible explanation of how the data were actually generated. In practice, therefore, that account will either be about a stochastic process or about what is going on in the population from which the data were sampled at random. It can also be important to verify what the sampling design was and whether it was implemented properly.

It is common to represent the data-generation process with a statistical model. A broad and popular class of data-generation processes can be written as $Y = f(X) + \varepsilon$, where Y is the response variable, X is a set of predictors, ε is a disturbance term, and $f(X)$ is some function mapping the predictors to the systematic part of Y. For a conventional linear model, $f(X)$ is a linear combination of the p predictors: $\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$. An alternative might be to exponentiate the linear combination of predictors, as is common in Poisson regression. In statistical learning, $f(X)$ is far more open-ended. However, for a set of N observations $i = 1, 2, \ldots, N$, each ε_i is often assumed to have the same convenient properties as it does for classical linear regression: $\varepsilon_i \sim NIID(0, \sigma^2)$.

A credible parametric regression model depends on specifying a number of details justified with reference to subject matter knowledge and past research. For example, what is it about how the data were generated that permits one to assume that the predictors enter the model in a particular manner? Although

statistical learning is often less assumption bound, it is never assumption free. There can be, for instance, the need to explain why the disturbances represented by ε_i have an expectation of zero, are independent of the predictors (or at least uncorrelated with them), and are independent of one another. These are very strong statements about how the data were generated. They require careful thought and justification, often beginning with a decision about whether to treat the data as a random realization, a random sample, or "just" a dataset.

Such questions are often difficult when $f(X)$ is known to take a particular parametric form. When $f(X)$ is to be largely determined by the data, the questions can be daunting. For example, how does one argue that ε_i is at least uncorrelated with the predictors when the transformations to be applied to these predictors are not yet known? And if any of the assumptions made about ε_i are substantially wrong, the results of the data analysis can be substantially wrong as well.

Much the same formulation can apply to categorical outcomes, so that one might write $G = f(X) + \varepsilon$. It is more common to write the expression for the conditional expectation of G and then to characterize the uncertainty separately. In the case of a binary response, for example, one could write $G = f(X) + \varepsilon$, with G coded as "1" or "0". Usually, however, one replaces G with the mean-value parameter (McCullagh and Nelder, 1989: 30), in this instance a probability, alters the $f(X)$ accordingly, and then indicates that the probability is a parameter in the binomial distribution by which the binary outcomes are generated. The binomial distribution is responsible for the uncertainty. But just as for quantitative outcomes, a strong subject matter case needs to be made that a particular formulation applies. Thus, what is the rationale for assuming that all observations with the same set of predictor values are subject to the exact same conditional probability? Why is there no heterogeneity in that conditional probability? Or, what reason is there to believe that, conditional on the predictor values, the binary events are independent of one another.

Taking the data-generation process into account is not usually by itself sufficient. It can also be important to consider what use will be made of the data analysis. One key dimension is what information from a data analysis will figure in the conclusions to be drawn. Sometimes interest centers primarily on the fitted values for the $f(X)$. There is no concern with representing how the predictors are related to the response. Building on an earlier example, the goal may be to make admissions decisions to a university based on information about the performance in their freshman year of previous successful applicants. Why some students do better than others does not matter because the admissions office is in no position to do anything about that.

Alternatively, the primary concern may be in learning how inputs are related to outputs. Organizations on campus that provide support services, such as tutoring for matriculating students, will want some guidance on where to intervene. Are students for whom English is not a primary language, for

instance, at greater risk for poor academic performance? In short, whether attention is directed toward the fitted values, the relationships between inputs and outputs, or both, will affect how the data analysis is done and assessments of its worth.

There is another use that should be briefly mentioned, but does not fall within a regression perspective. One might want to compute summary statistics for the response, conditional upon certain values of the predictors. For example, one might want an estimate of number of students for whom English is a second language, and who will not graduate. No comparisons will be made to other students, and whether any graduation problems stem from language difficulties or from other factors is not of immediate interest. There is no concern with how the response changes depending on the values of predictors. Such applications are not considered here.

Another important factor shaping a data analysis can be what kind of story is likely to be told. Just as in parametric regression, there can be four kinds of stories.

1. *A Causal Story*—A data-generation process is assumed and given a causal interpretation (Freedman, 2004). For any study unit i subject to this process, the values of any X_{ij} can be set (manipulated) independently of the values of any other predictor. A researcher or nature determines these values. Then some natural process maps X_i onto $f(X_i)$ and attaches a value of ε_i. The value of ε_i behaves as if drawn at random from some distribution, independently of the values of X_i. Once again, the absence of a linear correlation can often be sufficient. Other units are subjected to the same process with each value of ε_i drawn independently of one another. In addition, the values of X_i set for any given unit do not alter the response of any other unit. All this can be framed as the stochastic process responsible for the data on hand, or for the population from which a random sample is drawn.

   In this setting, the job of statistical learning is to recover the $f(X)$ nature uses. The residuals, defined as the arithmetic difference between the observed response values and the estimated $f(X)$, are used to characterize the distribution of the ε_i. These goals are the same as for all causal modeling. What statistical learning offers is powerful tools to help learn inductively what $f(X)$ may be. One can think of this as function estimation. For example, one might be interested in the function that turns a person's human capital into earnings, or volatile hydrocarbons into ozone.

2. *A Conditional Distribution Story*—The basic formulation is the same; there is an assumed model with many of its features to be informed by the data. However, no causal interpretation is given. The $f(X)$ is descriptive only. One is satisfied saying something such as $Y \sim N(f(X), \sigma^2)$, where $f(X)$ now represents the conditional mean or perhaps the condi-

tional expected value. To take a simple illustration, for bivariate linear regression one might write $y_i \sim N(\beta_0 + \beta_1 x_i, \sigma^2)$, with $f(X) = \beta_0 + \beta_1 x_i$ as the expected value of y_i.

Interest centers on the conditional distribution $Y|X$, but no claims are made that manipulating the value of X alters Y. So, there is no need to assume that one or more of the predictors can be individually manipulated, or that setting a predictor at a particular value for the ith case does not affect the response of the jth case. The key assumptions center on distribution of the ε_i. For example, one must persuasively argue that ε_i is at least uncorrelated with, and ideally independent of, the predictors. As in the causal story, this is a demanding task when the functions to be applied to X are not yet known.

Just as for the causal modeling story, statistical learning is used to characterize the $f(X)$, taken to be real and "out there." This too may be seen as function estimation. Thus, one might be interested in the function that associates a score on the mathematics Scholastic Aptitude Test with a score on the verbal Scholastic Aptitude Test, or the function that associates the number of predators with the number of prey.

3. *A Data Summary Story*—Although there can be a data-generation process in principle, it plays no direct role in the data analysis. Thus, there is no assumed model by which the data were generated. There are only the data. Statistical learning is then used as a data reduction tool. The aim is to provide through some data-derived $f(X)$ an accurate and accessible summary of how, in the data on hand, y_i is systematically related to a set of predictors $x_{ij}, i = 1, 2, \ldots, N; j = 1, 2, \ldots, p$. In the same spirit, there can be residuals just as in the two earlier stories, but these have no necessary correspondence to some ε_i; residuals are computed from the data whereas the ε_i is a feature of a hypothetical model.

   One may only be interested in the conditional distribution of the data being analyzed or one may also wish to generalize beyond the data to other similar settings. In some cases, interest ultimately may be in the $f(X)$, but its recovery is at least premature. For example, one might still be interested in the function that turns a person's human capital into earnings but have only proxy measures for human capital (e.g., years of education).

4. *A Forecasting Story*—Using the data on hand, one constructs a function with which to make forecasts. If there is some real $f(X)$ "out there" that can be exploited for forecasting, all the better. But there is no interest in causal effects, no interest in the conditional distribution $Y|X$, and no interest in data reduction unless they can improve forecasting skill. Statistical learning is a tool for developing a useful forecasting apparatus.

For example, one might be interested in forecasting water quality at local beaches from the previous week's precipitation.

The difference between the causal story and the data reduction story is sometimes a matter of degree. Consider a simple example. Suppose there is a single predictor, and the $f(X)$ is actually a parabola: $f(X) = \beta_0 + \beta_1 X + \beta_2 X^2$. For the causal story, the goal is to determine from the data that $f(X)$ takes this exact form, including accurate estimates of the values for its three parameters. For the data summary story, a data analyst might be satisfied learning that the $f(X)$ is smooth and convex, and the approximate value of X at which $f(X)$ is minimized.

Why would the researcher settle for less than the full causal story if $f(X)$ is causal? Perhaps the statistical learning procedure applied does not work well for these kinds of functions. Or perhaps the predictor is measured poorly, or is at best a proxy for the predictor needed. Indeed, the predictor used may be the wrong one altogether. Or perhaps the response is measured with so much noise that the $f(X)$ is effectively obscured.

A consideration of the four possible stories raises a topic to which we return many times: the appropriate loss function to be employed. Almost inevitably, the empirical correspondence between Y or G and the $\hat{f}(X)$ will be imperfect. The fitting enterprise, therefore, depends on how the disparities between the observed response variable values and the fitted response variable values are treated. Usual practice is to minimize a loss function with these disparities as inputs. In conventional least squares regression, for example, the loss function is quadratic.

In the pages ahead, a variety of loss functions are considered. Some include a penalty for fitted values that are unnecessarily complex. Some allow for asymmetric losses so that for classification exercises, false positives can be weighted differently from false negatives. For example, the costs of mistakenly concluding that a patient has cancer are likely to be very different from the costs of mistakenly concluding that a patient is cancer free. We show that the loss function one chooses can dramatically affect the fitted values, with important implications for the story to be told and how that story will be used.

1.3.2 Statistical Inference

In principle, statistical tests and confidence intervals can be important for each of the four stories. If there is a data-generation process to be characterized, it is common to proceed as if the data have been produced in a manner that introduces some randomness. As already mentioned, sometimes that randomness is a product of probability sampling undertaken by the researcher. Sometimes that randomness is taken to be an inherent part of how nature generated the data (sometimes called "model-based sampling"). In either case, if the data were generated again, they are almost certain to be different, at

least a bit, because of "chance." So, the role of chance needs to be addressed as part of the data analysis.

For example, the expression $Y = f(X) + \varepsilon$, ε is often seen as the source of the randomness, whether it be a product of actions undertaken by the researcher or by nature. Should parametric regression be applied to characterize the $f(X)$, statistical tests and confidence intervals can naturally follow. The same can hold for statistical learning.

Statistical inference has no role when description of the data on hand is the only goal or when the links to a population or stochastic process cannot be credibly articulated in a manner the p-values require. For example, testing hypotheses about how average earnings vary with seniority for a quota sample of clerical workers intercepted in a local shopping mall will likely produce uninterpretable p-values, even if it were possible to figure out the population to which inferences were being made. Absent random sampling, what would be required is a credible account of a natural data-generation process meeting the requisite assumptions. As already noted, such accounts are often very difficult to construct.

Statistical inference can be important for forecasting. But, the forecasts must be into a probability sample from the same population as the data on hand, or into a realization of the same stochastic process. Otherwise, the computed probabilities are likely to have no useful meaning. If, for example, the forecasts are made into a random sample from a different population, or into a convenience sample, the forecasts can differ from what actually transpires because the response is not related to the predictors in the same fashion it was in the data from which the forecasts were constructed. More than random error is involved. A 95% confidence interval, for example, will not cover the population value 95% of the time because each interval is offset by some amount of bias.

Some readers may wonder why there has been no discussion of random assignment as a way in which a chance process can affect data. Random assignment to a treatment group or to a control group within a randomized clinical trial, for instance, can be formulated within a probability sampling framework. But the inferential issues are somewhat different from random sampling in observational studies. Under random assignment, the uncertainty involves chance variation in estimated treatment effects because of the way in which a fixed group of study units is assigned to treatments and control conditions. There is usually no larger population to which inferences are formally being drawn. There is also commonly the need to construct a theoretical model of causal effects. In short, although many randomized experiments in principle can be analyzed using statistical learning procedures (e.g., within a dose-response framework), they are rarely needed and even more rarely used. Randomized experiments are not considered further in this book.

To summarize, statistical inference can in principle play a useful role across a wide range of statistical learning applications. In practice, however, we show that statistical inference is not particularly salient in statistical learning. Even

if the data are known to have been generated in a manner that might justify statistical inference, and even if all of the necessary prerequisites for statistical inference are present (e.g., all of the needed predictors), the way statistical learning is undertaken can make statistical inference inappropriate.

1.3.3 Some Initial Cautions

Expositions of statistical learning commonly assume that the goal is to tell a causal story or a conditional distribution story. There is some truth external to the data that statistical learning will help to reveal. One wants to know the $f(X)$ and if the $f(X)$ can be represented as parametric, the values of its parameters as well.

It is important to be clear from the start that no credible statistician would ever claim that even when all of the necessary predictors are present and perfectly measured, there are one or more statistical learning procedures that will exactly capture the $f(X)$. The data with which one works will necessarily be an imperfect reflection of the $f(X)$ because of the impact of ε; the values from the $f(X)$ and ε are thoroughly commingled as Y is generated. Because ε is unobservable, it cannot be removed from Y in order to obtain the $f(X)$. This is a fundamental problem inherent in all estimation.

Matters are further complicated by the need to learn from the data both the underlying functional forms and the values of key parameters. We show later that the need to learn about the functional forms can place very heavy demands on a dataset. Large samples are often necessary with the observations more densely packed where the nonlinear functions are changing more rapidly. We also show that the performance of all statistical learning procedures is significantly determined by tuning parameters for which only very broad guidelines are likely to exist. Craft lore rather than proved theorems can dominate practice.

Just as in parametric regression, researchers have to be satisfied with one of two possible fallback positions. The first entails desirable finite sample properties such as unbiasedness and efficiency. For reasons that become clear later, it is difficult for statistical learning procedures to satisfy these requirements. The second fallback position entails desirable asymptotic properties such as consistency. Of late there have been some successes for a number of statistical learning procedures (Breiman, 2004; Jiang, 2004; Lugosi and Vayatis, 2004; Efron et al., 2004; Zhang and Yu, 2005; Bickel et al., 2006; Bühlmann, 2006; Traskin, 2008), but as considered in subsequent chapters, these formal results often do not answer the questions that applied researchers would make a top priority. And there remains, as always, the matter of how best to make use of asymptotic results for the sample on hand.

In practice, moreover, real world studies rarely cooperate with what the theoretical statistical work requires. Perhaps most obviously, if the goal is to recover the $f(X)$, X must be known; each and every predictor must be identified. Then, all of the predictors must be in the dataset to be analyzed

and measured without error (even random measurement error). It is difficult to find examples in which this is even approximately true, and most applications are not even close.

If the goal is to recover the $f(X)$, often the best that one can do is to proceed with the understanding that one's results will be biased and inconsistent, sometimes substantially. Because both the direction and the magnitude of these difficulties are usually unknown—if they were known, they would not be difficulties—it can be difficult to determine what to make of the fitted values.

By default, therefore, real applications are usually about data reduction, and occasionally about forecasting. One important implication is that for these stories, a substantial portion of the theoretical justification for many statistical learning procedures provides indirect guidance at best. Consequently, rationales for particular statistical learning applications tend to rely on in-sample features of the fit, forecasting skill, and subject matter knowledge. This is a point to which we return many times in the pages ahead.

Finally, there is nothing in any of the four stories that requires statistical learning. One can apply parametric regression to the very same ends. Statistical learning earns its keep when, for causal and estimation stories, the $f(X)$ is not well understood but is likely to be substantially nonlinear and hence, complex. In the same spirit, statistical learning earns its keep for data reduction and forecasting when the systematic information in the data is best captured with a complex fit.

"Complexity" can be conceptualized in different ways, many of which are not easily represented within a statistical framework (Zellner et al., 2001). For example, is "simple" the opposite of "complex" or is "parsimonious" a better choice? And if parsimonious, how might one translate that into statistical concepts? As a practical matter, complexity is commonly represented by the degrees of freedom "used up" in the fitting process. A statistical learning procedure produces a more complex fit when more degrees of freedom are used up. In parametric regression, for instance, a larger number of regression coefficients implies that a larger number of degrees of freedom will be spent, and that a more complex rendering of the $f(X)$ will result. We show that this is a special case of how complexity is often measured in statistical learning. Indeed, relying solely on the degrees of freedom used up can be unsatisfying. Which is more complex: $\hat{y}_i = \hat{\beta} x_i$ or $\hat{y}_i = \hat{\beta} x_i^2$? Or, are they equally complex? We show that with some of the most recent and advanced statistical learning applications, complexity is even more difficult to conceptualize and measure.

1.3.4 A Cartoon Illustration

One can get a more grounded sense of the issues by comparing a hypothetical fit using linear regression to a hypothetical fit that might result from statistical learning. Suppose one is concerned about the number of misconduct incidents committed by prison inmates. The response variable is the number of such

incidents reported for each inmate during a one year period. The predictor is the nominal sentence length of the prison term each prisoner received.

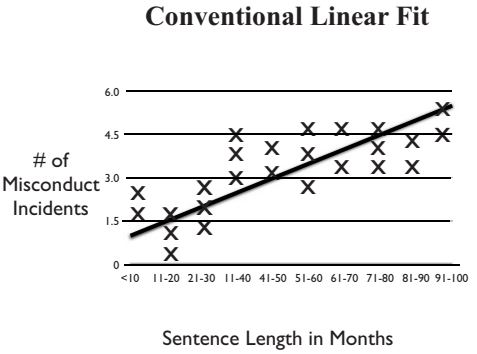


Fig. 1.6. Imposing a least squares line on the data.

Figure 1.6 is a hypothetical scatterplot with a least squares regression line overlaid. The regression line shows that in general, the relationship is positive. The number of misconduct incidents increases with sentence length. To the eye, the fit is quite good, and a positive relationship is hardly surprising. Longer prison sentences are generally associated with more serious crimes and the criminal histories of "habitual offenders." Both are thought to characterize inmates who would not "program" well. So, a researcher might well be satisfied with the results.

However, Figure 1.7 shows that if the data are allowed to play a larger role in determining the functional form, a somewhat different story emerges. The number of misconduct incidents decreases with sentences up to 20 months, increases rapidly with sentences from 20 to 40 months, and is almost flat thereafter. There is probably no simple explanation for this pattern.

The sentence lengths for which the relationship is nearly flat may represent older inmates subject to sentence length enhancements because of earlier convictions. It is well known that older inmates are much less likely to get into trouble in prison. The sentence lengths associated with the rapid increase in misconduct incidents may reflect the behavior of younger inmates, often gang members, convicted of serious crimes, but not yet subject to sentencing enhancements (i.e., "gang-bangers"). The sentence lengths associated with declines in misconduct could represent inmates with short sentences who wish to stay out of any trouble that would jeopardize their release. The relationship

is negative, perhaps because the risks of becoming involved in a misconduct incident increase with time behind bars. So, those inmates who are thinking ahead to their parole dates may be especially careful if their period of risk is longer.

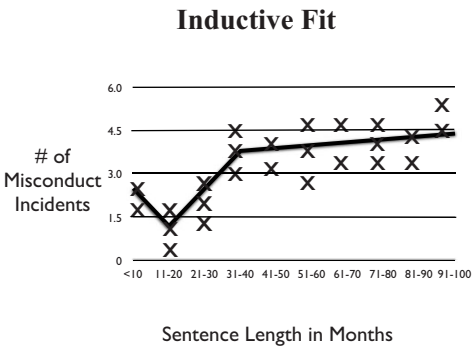


Fig. 1.7. Letting the data determine the functional form.

It is also possible that the inductive fit in Figure 1.7 is dominated by happenstance. Note that the nonlinear fit is substantially driven by three inmates with sentences between 11 and 20 months. If for those three observations, the number of misconduct incidents were a bit greater, or if those three observations were missing altogether, the original linear fit would be far more satisfactory.

A comparison between Figure 1.6 and Figure 1.7 raises an important issue to which we return many times. When the linear fit was imposed in Figure 1.6, the support for that line came from the full range of the available data. With the inductive fit, the support was highly local. For these data, the linear fit may not capture as well the relationship between sentence length and misconduct. But the linear fit is likely to be more stable under random variation in the data. The inductive fit may capture better the relationship between sentence length and misconduct. But the inductive fit may be relatively unstable under random variation in the data. In other words, the inductive fit may be too much a product of overfitting. Random variation is being interpreted as systematic variation. These and related points are more formally addressed in the pages ahead.

To complete the story, Figure 1.8 shows an inductive fit for a binary outcome, coded "1" for misconduct and "0" for no misconduct. Each "X" in

Figure 1.8 represents many data points that cannot be seen because of overprinting. The same issues arise. One can impose a functional form, such as the logistic or, as in Figure 1.8, allow the functional form to respond substantially to the data.

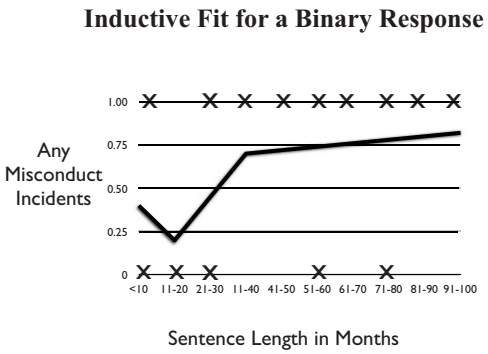


Fig. 1.8. Letting the data determine the functional form for a binary response.

1.3.5 A Taste of Things to Come

Figures 1.6 through 1.8 begin to raise a number of difficult questions that are addressed throughout the course of the book. Sometimes there some are good answers, but often the best answers are highly provisional. And often the answers come in the form of statistical procedures that at first seem somewhat curious. Consider the following sequences of operations.

1. Fit the data with some conventional procedure.
2. Compute the residuals as the difference between the fitted values and the actual values.
3. Compute a measure of fit from the residuals, such as the error sum of squares (also known as the residual sum of squares).
4. Apply the fitting procedure again, but weight the observations so that the cases for which the absolute value of the residuals is larger receive more weight, and the cases for which the absolute value of the residuals is smaller receive less weight.
5. Repeat the first four steps 1000 times.
6. Compute the final set of fitted values as a weighted average of the fitted values over the 1000 passes through the data, with the weights a function

of the error sum of squares computed in Step 3. Fitted values with a better overall fit are given more weight in the averaging. That is, each $\hat{y}_i$ is a weighted average of 1000 fitted values for each observation i, weighted so that the fitted values that perform better for a given pass are given more weight than those that perform worse.

7. Output the averaged fitted values, the predictor values, and some measure of fit between the fitted values of the response and the actual values of the response.

In this procedure, the averaged fitted values and the predictor values characterize the response function. The response function is a way to represent how the predictors are associated with the response. The overall measure of fit conveys how well the response function corresponds to the data. But a good fit has no necessary implications for whether the response function is "correct." It is not even clear what "correct" means for this procedure. No data-generation process has been proposed, let alone a causal model. There is also no apparent role for statistical inference. We begin and end with the data on hand.

The procedure just outlined has a lot in common with "boosting" (Schapire, 1999), a statistical learning procedure that is considered later. An interesting feature is that with each iteration, observations that are more difficult to fit are given more weight. We show, however, that boosting has more in common with conventional statistical procedures than might first appear. There is a loss function being minimized, just as in conventional parametric regression. What can be novel is the particular loss function being used and the manner in which a very flexible fitting function is constructed.

In many situations, boosting fits the data well, and often substantially better than procedures that make only one pass through the data. This can mean that it will forecast more accurately too. By these criteria, boosting is likely to outperform conventional linear regression even if the linear regression is implemented in a stepwise manner. The superior performance can become more apparent as the sought-after response function becomes more nonlinear.

Figure 1.9 provides an example of some boosting output. The response variable is the number of homeless individuals in a census tract in log units. The single predictor is the percentage of the land in a census tract used for residential purposes. Plotted are the fitted values.

One might well have expected a negative relationship overall, but had a linear regression model been imposed, the conclusions would have been quite misleading. There is a precipitous drop in the number of homeless as the percentage of residential land use in a tract varies from about 5% to about 15%. On either side, the relationship is essentially flat. The pattern looks like a neighborhood tipping effect, which might not have been anticipated. It is very unlikely that the precise location of the transition would have been anticipated. This is just the kind of relationship with which statistical learning procedures can excel, and conventional regression can stumble.

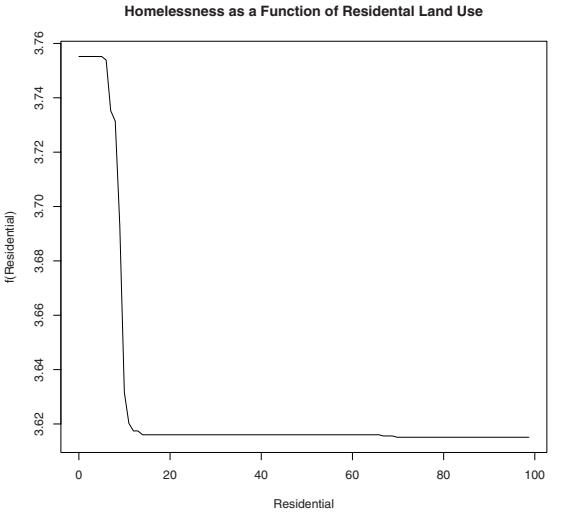


Fig. 1.9. Log of the number of homeless as a function of the percentage of land devoted to residential use.

Figure 1.9 was constructed from 100 passes through the data. A number of the other statistical learning procedures to be considered also make many passes through the data, but with very interesting twists and turns. For example, one can let each pass through the data be based on a random sample with replacement of the data on hand. One averages as before, but without any weighting. This is a first approximation of a procedure called "bagging" (Breiman, 1996).

The quick introduction to boosting and bagging no doubt looks very different from parametric regression. One puts one's faith in a computer algorithm and pushes the run key. No data-generation process need be specified and so there seems to be no need for a statistical model. For these reasons, Breiman (2001b) has called such methods "algorithmic." However, the break with conventional regression need not be that dramatic. Statistical learning, as the basis for the causal story or the conditional distribution story, relies on a specified data-generation process and a statistical model. It is the data summary story and the forecasting story that can be seen as "algorithmic" in the sense Breiman meant.

1.4 Some Initial Concepts and Definitions

Given the regression analysis framework, a wide variety of statistical learning procedures and approaches are examined. But, before going much farther down that road, a few definitions and concepts are necessary. They play a key

role in the chapters ahead and, at this point, benefit from a brief introduction. We return to this content many times, so nothing like mastery is required now. And that's a good thing because some readers will find the content challenging, at least at a first reading.

1.4.1 Overall Goals

The procedures we examine have been described in many different ways (Sutton and Barto, 1999; Christianini and Shawe-Taylor, 2000; Witten and Frank, 2000; Hand et al., 2001; Hastie et al., 2001; Breiman 2001b; Dasu and Johnson, 2003), and associated with them are a variety of names: statistical learning, machine learning, reinforcement learning, algorithmic modeling, and others. "Statistical learning" as used in the pages that follow, is based on the following notions.

There may or may not be some data-generation process "out there" whose features we wish to learn about from the data. Such a construct is outside of the data and can help set the goals of a data analysis: what kind of conclusions are to be drawn from the results of the data analysis? A proposed data-generation process can also help provide a rationale for one statistical learning procedure rather than another. But much of the hands-on job of applying statistical learning to data proceeds in the same manner whether or not a data-generation mechanism has been proposed.

The earlier definition of regression analysis applies. Thus, for a quantitative response variable the goal is to examine $Y|X$ for a response Y and a set of predictors X. If the response variable is categorical, the goal is to examine $G|X$ for a response G and a set of predictors X. X may be categorical, quantitative, or a mix of the two. Consistent with common regression practice, the observed values of X are usually treated as fixed.

Many different features of $Y|X$ can be examined, but the conditional mean, $\bar{Y}|X$, is usually a key concern. This is the feature of $Y|X$ that has to date received the most attention. $\bar{Y}|X$ is sometimes interpreted as an expected value. For $G|X$, the conditional proportion is usually of interest for each of its K categories. $\bar{G}|X$ is sometimes interpreted as a conditional probability. In either case, $G|X$ is typically linked to response categories, often called "classes." The goal is to assign cases to classes. Then, the task can be called "classification," and the procedure employed can be called a "classifier."

1.4.2 Loss Functions and Related Concepts

In real applications, any efforts to fit the values of the response variable with one or more functions of the predictors will almost always be less than perfect. Indeed, if the fit is perfect, it is likely that some serious mistake has been made. Nevertheless, it often makes good sense to try to fit the response values as well as possible. This implies that a fitting criterion needs to be defined in order to characterize how good the fit is.

Fitting criteria are commonly called loss functions, cost functions, or objective functions. We use those terms interchangeably. The most common loss function for quantitative response variables is squared error loss, also called quadratic loss: $[Y - \hat{f}(X)]^2$. Recall that the mean is the central tendency measure that minimizes the sum of the squared deviations around itself. The fitted values that minimize squared error loss are the conditional means for each configuration of x-values. That is, compute $\bar{y}|X = x$ for each $X = x$.

If one wants to treat the data as a random sample for a well-defined population or as a random realization from a well-defined stochastic process, it follows that these conditional means are unbiased estimates and have other desirable formal properties. By a "well-defined" population, one means that it is possible to determine which units are in the population and which are not. For example, "all living adults in the United States" is not well defined until "adult" and "in the United States" are defined. A stochastic process is "well-defined" when it is thoroughly and precisely described. This will usually require one or more mathematical expressions. Thus, "a set of coin flips" by itself is not a sufficient definition. One would need to specify a particular binomial process (if that is the intent).

If concern is with bias, one can think of the conditional means as the "gold standard" under squared error loss. In practice, however, the gold standard can have some undesirable side effects. In particular, there may be for any particular $X = x$ no values of the response variable, or so few that the computed mean has a very large amount of sampling error.

There are a number of potential fixes. One is to compute the response variable conditional means not for $X = x$, but rather for X close by x. For example, if the predictors are age and education, rather than computing the conditional mean of income for, say, 28-year olds with four years of work experience, one might compute the conditional mean of income for all individuals between 25 and 27 who have between three and five years of work experience. For unbiasedness to be maintained, the true mean for the first set of individuals must be the same as the true mean for the second (and larger) set of individuals. "True" denotes the conditional mean in the population or the mean associated with the stochastic process responsible for the data. The goal is still to estimate the true mean income for individuals who are 28 and have four years of work experience, but information from nearby ages and years of work experience is being used.

In practice, one will rarely know if such assumptions are correct, but there may be evidence that they are close enough; the conditional means for the two groups are not likely to differ by enough to matter. For example, from past studies one many know that incomes usually change slowly as age and seniority change. Consequently, the bias that would be produced is negligible. Nearest neighbor methods, discussed later, build on this approach.

Computing different conditional means for different ranges of predictor values is the same as assuming the $f(X)$ is a step function. Alternatively, one might assume that the $f(X)$ is linear or some other smooth function of X. In

effect, the data are pooled once again, but in a different manner so that more information is brought to bear on each conditional mean. Each conditional mean now can be computed using least squares regression, for example, with information from all of the observations in the dataset. For unbiasedness to be maintained, the true conditional means must fall on the assumed smooth function of the predictors. In this sense, the model for the conditional means is "right."

Squared error loss imposes a particular way of weighting the disparities between the response and the fitted values that needs to be considered carefully in the context of the data's properties and how the results of the data analysis will be used. Squaring makes large disparities especially influential in the fitting process, which can make the fitted values vulnerable to outliers (i.e., values that lie some distance from the mass of the data). In addition, the weights are symmetric; fitted values above the response are treated the same as fitted values below the response. Yet, symmetric weights are often inappropriate.

The tradition of resistant/robust estimation grew in part as a reaction to the problems caused by outliers under squared error loss. One option that we consider in a later chapter is linear loss: $|Y - \hat{f}(X)|$. Recall that the median is the central tendency measure that minimizes the sum of the absolute values of the deviations around itself. Thus, linear loss leads to computing the conditional median rather than the conditional mean. Then, the statistical issues that follow are much like those associated with the conditional mean.

The consequences of symmetric weighting are a very important issue to which we return later. We show that taking asymmetric costs into account can significantly change the fitted values. Decisions made from these fitted values can significantly change as well. Consider again, for example, the number of homeless in a census tract as the response variable, and predictors that are features of census tracts. Overestimating the number of homeless individuals in a census tract can have very different implications from underestimating the number of homeless individuals in a census tract. Yet, a symmetric loss function would assume that in the metric of costs their consequences are exactly the same.

Consider now a categorical response variable and a set of predictors. Just as for a quantitative response variable, one can proceed with a symmetric loss function. Suppose there are K distinct and mutually exclusive classes. Any misclassification—the fitted class is the wrong class—is given the same weight of 1.0. For example, the error of asserting that a high school student is a dropout when that student is not is given the same weight as asserting that a high school student is not a dropout when that student is. In both cases, the errors are given a value of 1.0. Correct classifications are given a value of 0.0. To minimize the sum of these errors over classes, it is clear that one can simply classify by the most common class. Because of the 0/1 coding of losses, using the sum of the squared errors gives the same result. In other terms that are used a lot later, the fitted class is determined by a "vote" in which the

class with the plurality wins. When there are two classes, classification is by majority vote. Such procedures can be placed in a Bayesian decision theory framework (e.g., Bishop, 2006: 38–46) and are often called "Bayes classifiers." The proportion misclassified by this approach is often called the "Bayes error rate."

For example, given $X = x$ and a response of dropout or no dropout, if 65% of the students are not dropouts, all of the students for which $X = x$ are assumed to have not dropped out. Then, 35% of the students for which $X = x$ are misclassified. Suppose now that there are three response categories: drop out, no dropout, and moved to another school. Also suppose that 45% of the students fall in the no dropout category and that this is the largest percentage of the three for $X = x$. Then, all students for whom $X = x$ are assumed to have not dropped out and 55% are misclassified. In both illustrations, not dropping out is the fitted class. A similar rationale can be applied to each subset of observations defined by each unique configuration of x-values. Then it is possible to sum misclassifications over these unique values to obtain an overall proportion of cases misclassified.

Just as for quantitative response variables, one may use this reasoning to obtain useful estimators for parameters of a population or of a stochastic process. For example, if the data are a random sample from a well-defined population, each conditional proportion computed from the sample is an unbiased estimate of its population conditional proportion. That is, the proportion computed for each class is an estimate of the population proportion within that class. The term "Bayes risk" is sometimes applied to the expected value of the classification error when, in just such circumstances, the response variable is a random variable. A useful and accessible discussion from a more purely Bayesian perspective can be found in Ripley (1996: Section 2.1). A complementary discussion from a computer science perspective can be found in Bishop (2006: Section 1.5).

Finally, treating all classification errors as generating the same costs is often inappropriate, especially when real decisions will be made based on the classifications. As before, symmetric loss functions can be misleading. For example, the costs to the student and the school of failing to identify a student as a potential dropout may be very different from the costs to the student and the school of incorrectly identifying a student as a potential dropout. If these costs can be usefully approximated, it only makes sense to take them into account before actions are taken. We show later that building in the costs of classification errors can dramatically alter the classifications themselves.

1.4.3 Linear Estimators

Within the context of a regression analysis, consider a dataset with N observations. There is a single predictor X and a single value of X, x_0. Generalizations to more than one predictor are provided in a later chapter. The fitted value for $\hat{y}_0$ at x_0 can be written as

$$\hat{y}_0 = \sum_{j=1}^{N} \mathbf{S}_{0j} y_j. \tag{1.2}$$

S is an N by N matrix of fixed weights and is sometimes called a "smoother matrix." The subscript 0 denotes the row corresponding to the case whose fitted value of y is to be constructed. The subscript j denotes the column in which the weight is found. In other words, the fitted value $\hat{y}_0$ at x_0 is a linear combination of all N values of y_i, with the weights determined by $\mathbf{S}_{0j}$. In many applications, the weights decline with the distance from x_0. Sometimes the declines are abrupt, as in a step function. In practice, therefore, a substantial number of the values in $\mathbf{S}_{0j}$ can be zero.

If formal estimation of a conditional mean of the population is the goal, one has a linear estimator $\bar{y}|x$. It is a linear estimator because with **S** fixed, each value of y_i is multiplied by a constant before the y_i are added together; $\hat{y}_0$ is a linear combination of the y_i. Linear estimators play a central role in all of the chapters ahead. Linearity can make it easier to determine the formal properties of an estimator, and linear estimators and are often easier to understand. But one must be clear that even if an estimator is linear, the relationship between $\hat{y}$ and x can still be highly nonlinear, as we soon show.

$\mathbf{S}_{0j}$ has much in common with the hat matrix from conventional linear regression analysis. Recall that

$$\hat{\mathbf{y}} = \mathbf{X}\boldsymbol{\beta} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} = \mathbf{H}\mathbf{y}. \tag{1.3}$$

The hat matrix **H** transforms the y_i in a linear fashion into $\hat{y}_i$. $\mathbf{S}_{0j}$ performs the same function, but can be constructed using more general procedures.

Consider the following cartoon illustration in matrix format. There are five observations constituting a time series. The goal is to compute a moving average of three observations going from the first observation to the last. In this case, the middle value is given twice the weight of values on either side. Endpoints are often a complication in such circumstances and here, the first and last observations are simply taken as is.

$$\begin{pmatrix} 1.0 & 0 & 0 & 0 & 0 \\ .25 & .50 & .25 & 0 & 0 \\ 0 & .25 & .50 & .25 & 0 \\ 0 & 0 & .25 & .50 & .25 \\ 0 & 0 & 0 & 0 & 1.0 \end{pmatrix} \begin{pmatrix} 3.0 \\ 5.0 \\ 6.0 \\ 9.0 \\ 10.0 \end{pmatrix} = \begin{pmatrix} 3.00 \\ 4.75 \\ 6.50 \\ 8.50 \\ 10.00 \end{pmatrix}. \tag{1.4}$$

The leftmost matrix is **S**. It is post multiplied by the vector **y** to yield the fitted values $\hat{\mathbf{y}}$. But from where do the values in $\mathbf{S}_{0j}$ come? If there are predictors, it only makes sense to try to use them. Consequently, $\mathbf{S}_{0j}$ is usually constructed from X.

1.4.4 Degrees of Freedom

Recall that, loosely speaking, the degrees of freedom associated with an estimate is the number of observations that are free to vary, given how the estimate is computed. Consider a variable with N observations. In the case of the mean, if one knows the values of $N - 1$ of those observations, and one knows the value of the mean, the value of the remaining observation can be easily obtained. Given the mean, $N - 1$ observations are free to vary. The remaining observation is not. So, there are $N - 1$ degrees of freedom associated with the estimator of the mean.

This sort of reasoning carries over to many common statistics including those associated with parametric regression analysis. The number of degrees of freedom "used up" when the fitted values are computed is the number of regression parameters whose values need to be obtained (i.e., the intercept plus the regression coefficients). The degrees of freedom remaining, often called the "residual degrees of freedom," is the number of observations minus the number of these parameters. One of the interesting properties of the hat matrix is that the sum of its main diagonal elements (i.e., the trace) equals the number of regression parameters estimated. This is of little practical use with parametric regression because one can arrive at the same number by simply counting all of the regression coefficients and the intercept. However, the similarities between the $\mathbf{H}$ and $\mathbf{S}$ (Hastie et al., 2001: 129–130) mean that the trace of $\mathbf{S}$ can be interpreted as the degrees of freedom used up. Its value is sometimes called the "effective degrees of freedom" and can roughly be interpreted as the "equivalent number of parameters" (Ruppert et al., 2003: Section 3.13). That is, the trace of $\mathbf{S}$ can be thought of as capturing how much less the data are free to vary given the calculations represented in $\mathbf{S}$. The residual degrees of freedom can then be computed by subtraction (see also Green and Silverman, 1994: Section 3.3.4).

There are other definitions of the degrees of freedom associated with a smoother matrix. In particular, Ruppert and his colleagues (2003: Section 3.14) favor

$$df_{\mathbf{S}} = 2\text{tr}(\mathbf{S}) - \text{tr}(\mathbf{S}\mathbf{S}^T). \tag{1.5}$$

In practice, the two definitions of the smoother degrees of freedom will not often vary by a great deal, but whether the two definitions lead to different conclusions depends in part on how they are used. If used to compute an estimate of the residual variance, their difference can sometimes matter. If used to characterize the complexity of the fitting function, their differences are usually less important because one smoother is compared to another applying the same yardstick. The latter application is far more salient in subsequent discussions.

Beyond its relative simplicity, there seem to be interpretive reasons for favoring the first definition (Hastie et al., 2001: 130–133). Consequently, we use the trace of $\mathbf{S}$ as the smoother degrees of freedom. We show that the larger the value of the effective degrees of freedom, the more flexible is the fitting

function and the more complex the fit. We also show that the effective degrees of freedom does not have to be an integer.

1.4.5 Model Evaluation

Just as in any fitting exercise, there needs to be a way to evaluate the quality of the fit. This evaluation is best done combining quantitative information obtained during the data analysis with subject matter expertise and policy concerns. A model that fails in subject matter or policy terms is of little use no matter how well it scores on statistical criteria. This might mean, for example, choosing a smoother fit than might be favored by a some statistical measure if the substantive implications are more easily understood and more consistent with existing subject matter knowledge.

But what kind of quantitative measure should be used? For the conventional linear regression, it is common to work with the mean squared error, $\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$, or some standardized version such the R^2. The R^2 is usually interpreted as the proportion of the variance of the response that is accounted for by the predictors. So, larger values of R^2 can be considered better than smaller values of R^2, and it follows that fitted values with a larger R^2 can be considered better than fitted values with a smaller.

The mean squared error and the R^2 are "resubstitution" fit statistics because the data used to evaluate the model is exactly the same as the data used to build the model. Such measures can convey unjustified optimism about the quality of the fit. In an effort to minimize the error sum of squares, the fitted values will respond as best they can to the data on hand, some features of which may be idiosyncratic. Then, the results will not generalize well to new samples from the same population. The unjustified optimism is exacerbated when the number of regression coefficients being estimated is large relative to the sample size because the fitting function gains flexibility relative to the amount of data. "Overfitting" is sometimes the term used to describe how unjustified optimism can be produced.

In response, a measure of fit can be used that attempts to adjust for the overfitting. A simple alternative to R^2 is R^2 adjusted for degrees of freedom:

$$\text{Adj}R^2 = 1 - \left[(1 - R^2)\left(\frac{N-1}{N-p-1}\right)\right], \tag{1.6}$$

where N is the of number observations, and p is the number of parameters whose values are determined by the data. For a given number of observations, increasing the number of unknown parameters reduces the measure of fit. By "unknown," one means parameters whose values are to be determined by the data. Just as with its unadjusted cousin, bigger is better.

Taking the degrees of freedom into account leads to a conceptual improvement over the unadjusted R^2 because an effort is made to discount fit quality resulting solely from the complexity of the fitting function. However, the adjusted R^2 lacks much formal justification and is not easily generalized beyond

a least squares context. The common use of terms such as "pseudo R^2" in such settings is telling.

The Akaike Information Criterion (AIC) and the Bayesian Information Criterion (BIC) are alternatives to the adjusted R^2. They are based on far more rigorous statistical theory, which can be an important consideration. The AIC represents the relative amount of information lost (using the Kullbeck–Leibler information) when a given model, whose parameters are estimated from the data, is compared to the unknown but true processes that generated the data (Akaike 1973). Thus, the baseline is found in a conceptual entity beyond the data themselves. The smaller the AIC the better the approximation to the truth. The BIC is based on the Bayesian posterior probability of a given model (Schwartz, 1978; Raftery, 1995: Sections 4.1–4.2) compared to the "null model" with no predictors. There is again the construct of a true model serving as a target. A larger posterior probability implies that the model is more credible and that one has a better approximation of the truth. The BIC is smaller when the posterior probability is larger.

The AIC and the BIC can be properly applied to a much larger set of fitting procedures than the various kinds of R^2s. But as with the adjusted R^2, the AIC and BIC can be seen as altering the model's measure of fit by imposing a penalty for complexity. The penalty for the AIC and BIC increases with the number of unknown parameters in the model and decreases with the sample size. In other words, the penalty is larger when the number of parameters increases relative to the number of observations.

The AIC can be written in a number of ways, but one common expression is

$$\text{AIC} = \log\left[\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2\right] + \frac{2p}{N}. \tag{1.7}$$

Likewise, the BIC can be written in a number of ways. A common expression is

$$\text{BIC} = \log\left[\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2\right] + \frac{[\log(N)]p}{N}. \tag{1.8}$$

The BIC imposes a heavier penalty for the number of regression parameters. But neither is easy to interpret without some baseline or other means of comparison. We return to this issue shortly.

The AIC and BIC can be generalized so that in principle they are useful fit statistics for statistical learning procedures. In particular, it is common to replace p with $\text{tr}(\mathbf{S})$. However, both measures can prove to be inadequate when it is not apparent how to quantify the effective degrees of freedom.

For example, statistical learning procedures are often applied several times to the data with one or more tuning parameters varied. The AIC may be computed for each. But each AIC is ignorant about the information obtained from prior fitting attempts and how many degrees of freedom were expended in the process. Matters are even more complicated if some of the variables

are transformed or recoded after examining descriptive statistics before the fitting begins. Often, the effective degrees of freedom used in the AIC and BIC will be too few. Some unjustified optimism remains.

As an alternative, it can be useful to think about the various measures of fit as efforts to characterize how well the fitting function forecasts. Then, one popular definition of fit is the expected prediction error (also called "expected forecasting error," or "expected generalization error")

$$\text{PRE} = E[(Y - f(X))^2]. \tag{1.9}$$

PRE is the mean squared error in the population from which the data were sampled or over limitless independent realizations of the stochastic process that generated the data. An alternative definition is the expectation of the sum of the absolute values of $(Y - f(X))$. How does one go about estimating such quantities?

An excellent option is to work with two (or more) random samples from the same population. One sample is treated as the "training sample" and the other sample is treated as the "test sample." A fitting function built from the training sample is applied to the test sample, and a measure of prediction error computed. That is, data from the test sample are used with the fitting function from the training sample to produce fitted values. These are paired with the observed values in the test sample when, for example, the mean squared error is calculated.

A lot also can be learned by unpacking the overall measure of test sample prediction error. It can be instructive to learn which observations are being underestimated and which are being overestimated, and by how much. For example, if the response variable is the number of homeless individuals in a census tract, it would be important to know if the fitted values tend to substantially underestimate homeless counts in census tracts where a large number of homeless individuals are likely to be found. Or the problems could be spatial; predictions for census tracts in one part of a metropolitan area may in general be less accurate. Such information can provide a more sensitive evaluation of the fitted values and sometimes suggest ways the fitting function might be improved.

Often there is no test sample. Under these circumstances, there are interesting alternatives that nevertheless draw directly on the idea of a training sample and a test sample. "Drop-one" cross-validation is a popular example. Drop-one cross-validation is also called "leave-one-out" cross-validation, "jackknife" cross-validaton, and "N-fold" cross-validation.

Imagine a statistical learning procedure that is applied N times to the data. Each observation in turn is dropped from the dataset and its fitted value computed. That is, each fitting is based on $N-1$ observations, from which the fitted value of the dropped observation can be computed. The mean squared error computed from the dropped values and their corresponding fitted values is a cross-validation measure of fit and an estimate of prediction error.

More formally,

$$CV = \sum_{i=1}^{N} [y_i - \hat{f}_i^{-i}(\mathbf{X}_i)]^2, \tag{1.10}$$

where the superscript $-i$ signifies that observation i has been dropped. One nice feature of cross-validation is that the effective degrees of freedom does not enter explicitly into the calculations. Another nice feature is that it has a PRE interpretation.

The Generalized Cross-Validation statistic (GCV) can be a handy cross-validation approximation. It can be applied to the existing data as a whole and is easily computed as

$$GCV = \frac{1}{N} \sum_{i=1}^{N} \left[\frac{y_i - \hat{f}_i(\mathbf{X}_i)}{1 - \text{tr}(\mathbf{S})/N} \right]^2. \tag{1.11}$$

The GCV requires the user to decide which definition of the effective degrees of freedom is to be used. As shown, the trace of the smoother matrix is the usual choice.

Another approach in the same spirit exploits a bootstrap procedure (Efron, 1983; Efron and Tibshirani, 1993: Section 17.7). In its simplest form, one takes B samples of size N, with replacement, from the data. Each random sample serves as a training sample so that for each, a fitting function is constructed. The original (unsampled) data serve as a test sample. One can then compute a mean square error B times using the fitting function from each of the training samples and the data from the test sample. Averaging over the B samples provides an estimate of prediction error.

The simple bootstrap estimate of prediction error is not entirely satisfactory. Each bootstrap sample is drawn from the original sample; the original sample is not really a pure test sample. The overlap leads to estimates of prediction error that are too small.

A better approach is to borrow some ideas from cross-validation. For any given bootstrap of size N, about a third of the observations will by chance not be selected. These observations can serve as a test sample when estimates of prediction error are computed. With a sufficient number of bootstrap samples, any given observation will likely fall in a test sample several times. The average mean squared error for each observation in its test samples provides an observation-specific estimate of prediction error. Averaging these over the N observations leads to an overall estimate of prediction error based on the "drop-one" bootstrap.

However, because each bootstrap sample will on the average contain only about two-thirds of the unique observations of the original sample, the data used to construct the fitting function will be more sparse than the full dataset. If $f(X)$ is complex, some of its features will be missed. Bias results. Consequently, the estimate of prediction error will be inflated. A correction can be introduced that leads to the following expression for the estimated prediction error.

$$\widehat{\text{PRE}} = .368(\text{MSE}_R) + .632(\text{MSE}_B), \tag{1.12}$$

where MSE_R is the resubstitution mean squared error from the original sample, and MSE_B is the mean squared error from the drop-one bootstrap. Further improvements are possible (Hastie et al., 2001: 219–220).

Finally it is also possible, with small modifications, to treat the AIC and BIC as estimates of prediction error. Thus,

$$\text{AIC} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 + \frac{2p\sigma^2}{N}. \tag{1.13}$$

In this form, the AIC is known as the C_p statistic and can provide an unbiased estimate of the prediction error (Efron and Tibshirani, 1993: 242). The BIC is now computed as

$$\text{BIC} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 + \frac{[\log(N)]\sigma^2 p}{N}, \tag{1.14}$$

and can provide a consistent estimate of the prediction error (Efron and Tibshirani, 1993: 242).

In both equations, p may be replaced by $\text{tr}(\mathbf{S})$, and an estimate of σ^2 is required. Estimates of σ^2 are usually constructed from the observed values of the response and the $\hat{f}(X)$. One proceeds as if the $\hat{f}(X)$ has negligible bias and the disturbances behave as the model $Y = F(X) + \varepsilon$ requires. Moreover, the model is specified independently of the data being analyzed; there can be no data snooping (Luo et al., 2006: 165–167). If these requirements are substantially violated, $\hat{\sigma}^2$ can be substantially biased so that the estimates of prediction error will be substantially biased as well.

These implications of data snooping can generalize. All of the methods to properly represent prediction error can provide overly optimistic results if the impact of data snooping is not taken into account. For example, if drop-one methods are employed with data already altered using information from earlier models, there is no way for the procedures used to estimate prediction error to know about the earlier snooping. Falsely small estimates can result. Or as we show later, the summary statistics just described can be used to tune a model. This can be a useful way to undertake the inductive development of a model. But, it is very easy to slip into "overtuning." For example, neither the AIC or BIC can know about models previously considered and rejected. Likewise, cross-validation depends on random splits of the data into training and test samples whose independence, such as it is, can be rapidly compromised when the data are used repeatedly in the same modeling enterprise. Falsely optimistic measures of model performance can result.

In summary, there are many choices available for obtaining more honest measures of fit. The resampling methods yield estimates having an intuitive appeal, but can be computationally taxing, especially in many statistical learning applications. The adjustments to the resubstitution mean squared

error are far easier to compute, but require a credible value for the effective degrees of freedom and for some, a credible estimate of σ^2. There also can be important differences in performance for given samples. Finally, there remain a number of unresolved issues even for the linear estimators emphasized here; nonlinear estimators are more demanding still (Efron, 2004). Therefore, there seems to be no definitive guidance on which measures to use and in practice, those that can be computed easily seem to dominate. The best course, when feasible, is to have a training sample and a true test sample. A key asset is that one can use as test data the dataset as it was before any data snooping was undertaken. A more honest assessment of overfitting can result, especially if the fit measures are not overused.

1.4.6 Model Selection

It is a relatively small step from model evaluation to model selection. A model that performs better is chosen over a model that performs worse. In parametric regression, model selection usually means deciding by some quantitative yardstick which predictors, and transformation thereof, belong in the model. One way to think about this process is that among all of the available regressors and/or their transformations that could be included, some have their regression coefficients set to zero. Regressors with such regression coefficients are, by definition, excluded from the model.

With parametric regression, the model selection process can be undertaken in at least three ways. First, hypothesis tests may be used when the candidate models are nested within a single, all-encompassing model (Cook and Weisberg, 1999: 266–272). The usual null hypothesis is defined by one or more constraints on the regression coefficients. The most common constraints require that a subset of the regression coefficients in the all-encompassing model be equal to zero. The smaller model is assumed to be the correct model unless the null hypothesis is rejected. Likelihood ratio and f-tests are popular.

Second, one may apply various regression diagnostics from which model flaws can sometimes be identified. After that, remedies can be sought. For example, plots of a model's residuals against its fitted values can reveal nonlinearities in the data that have been overlooked. Likewise, added variable plots (Cook and Weisberg, 1999: Section 10.5) can point to possible omitted variables and/or needed transformations.

Third, one may compare various candidate models by some goodness-of-fit measure, and choose the one that fits the data best. As just discussed, this is best done discounting fit quality for the number of unknown regression parameters. The AIC and BIC are common choices.

The AIC is one of a class of "asymptotically efficient" model selection tools (Hurvich and Tsai, 1989) that tries to find the model that loses the least information relative to how the data were actually generated. No claims are necessarily made, however, that the model selected is "correct" or even that

a correct model is in the set of models examined. The hope is that the model selected is a useful approximation of the "truth."

The BIC is one of a class of consistent model selection tools that in principle will find the correct predictors, if they are included within the set of models examined, by determining the proper dimension of the regressor matrix. It is this dimension that is consistently estimated. Suppose there is a set of predictors in the data set placed in any arbitrary order. Some of the predictors are unrelated to the response. Starting at the top of the regressor ordering, the BIC estimates how many of these predictors should be included. If there are, say, 15 ordered predictors, but none of the predictors after the first 9 are related to the response, the selected models should contain the first 9 regressors. Some of these may not be related to the response. But in principle, all of the predictors that belong in the true model are selected and the false positives make little difference because they are not associated with the response.

If the data analyst believes that the true model is within the set to be examined, some argue that the BIC can be a better model selection tool. If the data analyst does not believe that the true model is within the set to be examined, some argue that the AIC can be a better model selection tool. However, there are other versions of the AIC which are arguably superior to the BIC when the correct model is among those that could be constructed from the dataset (Simonoff and Tsai, 1999). At this point, there seems to be no clear consensus on which selection criterion is best. For the kinds of procedures considered in this book, the idea that there is a correct model to be found in the data seems somewhat anachronistic. Other approaches to model selection are considered in later chapters.

For all three approaches to model selection, model parsimony is also important. Simpler models are preferred, other things equal, and sometimes when they are not equal. Simpler models can be more stable to modest changes in the model, otherwise inconsequential differences in how the variables are measured, and random perturbations of the data, such as occur under random sampling. Simpler models may also be more easily interpreted. This can mean, for example, that a model selected using the AIC may not be simple enough despite adjustments for the effective number of parameters.

Sometimes model selection procedures are usefully automated. The canonical illustration is conventional stepwise regression. In the forward selection case, among all the candidate variables, the one with the largest correlation with the response is included in the model first. The slope and intercept are estimated. Among all of the remaining candidate regressors, the one with the largest partial correlation (conditioning on the regressor already in the model) is included. The values of the two slopes and intercept are then computed. This process is continued until no new candidate regressor improves the model fit sufficiently. Other criteria can be used to decide which variables to include, such as hypothesis tests or a measure of fit as with the ones just discussed. A key point is that after each predictor is introduced into the model, the values

of all of the model's parameters are recomputed. This is different from stagewise regression, discussed in later chapters, in which the values of regression coefficients computed in one step are not recomputed when a new predictor is added to the model.

Additional examples of automated model selection include backward selection stepwise regression in which less important predictors are eliminated one by one from the all-encompassing model, and all-subsets regression in which all possible submodels are compared. Among the risks of automation are that subject matter information is neglected so that the resulting models are not informative.

Both stepwise regression and stagewise regression are examples of "greedy algorithms," which figure significantly in later chapters. They are greedy because at each step or stage the single best predictor is selected but then not reconsidered later. As a result, overall optimization can be sacrificed to the local optimization undertaken at each step or stage; the long term can be jeopardized by short-term thinking. Yet, greedy algorithms are often practical and effective. The alternative of searching over all possible models can become computationally taxing, or even intractable, if there are a large number of predictors.

Whether automated or not, all model selection procedures can risk serious overfitting. For example, t-tests applied to later models do not take into account the t-tests applied to earlier models. One result can be spuriously small p-values. Then, predictors may be retained when they should be removed; a null hypothesis is falsely rejected. An unnecessarily complicated model can result, which may not generalize well. The same concerns apply to the various measures of fit just discussed, especially when they are used repeatedly. At a deeper level, all statistical inference can be seriously jeopardized when the same data are used to inductively build a model and then to estimate the model's final set of parameters. More is said about this in the next chapter.

Despite the many unresolved problems with model selection for parametric regression, we soon show that, broadly speaking, the same three model selection strategies are commonly applied in statistical learning. There are also many opportunities for automation. At the same time, however, model selection needs to be placed in a broader context.

With statistical learning, models may be characterized not just by constraints placed directly on individual regression coefficients one by one, but by constraints placed on the objective function being minimized. For example, penalties for model complexity, in much the same spirit as those used in the AIC and BIC, can be imposed when a least squares fitting criterion is applied. One goal can be to reduce the risks of overfitting in the model itself, not just alter the measure of fit. This is a theme that surfaces a number of times in later chapters. The resulting model can have regression coefficients that are generally smaller in absolute value (but not necessarily zero) than they would have been had the complexity penalty not been imposed.

If an important goal of a statistical learning procedure is to arrive at a model in which subject matter conclusions depend on the particular regressors included, the model-building process should be able to force unnecessary regression coefficients to be zero. Then, model selection becomes regressor selection. If instead, interest centers more on the fitted values, there is no requirement that in the model building any regression coefficients should be zero. For example, a useful balance between bias and variance may require altering each of the regression coefficients a bit without trying to force any of them to zero. There is no intent to weed out any regressors. In short, what makes one model better than another depends in part on which model outputs are more important.

1.4.7 Basis Functions

Basis functions play a key role in all of the statistical learning procedures discussed. Basis functions are transformations of predictors that can allow for a more flexible fitting function by increasing the dimensionality of the regressor matrix. A set of p predictors becomes a set of predictors greater than p. This can allow the fitted values to be more responsive to the data.

Consider first the case when there is but a single predictor. X contains two columns, one column with the values of that single predictor and one column solely of 1s for the intercept. The $N \times 2$ matrix is sometimes called the "basis" of a bivariate regression model. This basis can be expanded if one allows transformations of X. A very powerful and flexible set of transformations can be written as

$$f(X) = \sum_{m=1}^{M} \beta_m h_m(X). \tag{1.15}$$

There are M transformations of X, which can include the untransformed predictor and a column of 1s (allowing for a y-intercept). β_m is the weight given to the mth transformation, and $h_m(X)$ is the mth transformation of X. Consequently, $f(X)$ is a linear combination of transformed values of X. The right-hand side is sometimes called a "linear basis expansion" of X.

One common transformation employs polynomial terms such as 1, x, x^2, x^3. Then, Equation 1.15 takes the form

$$f(X) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3. \tag{1.16}$$

When least squares is applied, a conventional hat matrix follows from which fitted values may be constructed.

Another popular option is to construct a set of indicator variables. For example, one might have predictor z, transformed in the following manner.

$$f(Z) = \beta_0 + \beta_1(I[z > 5]) + \beta_2(I[z > 8|z > 5]) + \beta_3(I[z < 2]). \tag{1.17}$$

As before, fitting by least squares leads to a conventional hat matrix from which the fitted values may be constructed.

Equation 1.15 can be generalized so that $p > 1$ predictors may be included:

$$f(X) = \sum_{j=1}^{p} \sum_{m=1}^{M_j} \beta_{jm} h_{jm}(X). \tag{1.18}$$

Each predictor has its own set of transformations. Then, all of the transformations for all predictors, each with its own weight β_{jm}, are combined in a linear fashion. For example, one could combine Equations 1.16 and 1.17 with both X and Z as predictors.

Why use the additive formulation when there is more than one predictor? With each additional predictor, the number of observations needed can increase enormously; the volume to be filled with data goes up as a function of the power of the number of predictor dimensions. This is what lies behind the "curse of dimensionality." One important result can be data that are too sparse for the intended analysis. In addition, there can be very taxing computational demands. So, it is often necessary to restrict the class of functions of X examined. One hopes that the response variable will be fitted sufficiently well by a model that is less flexible than what one ideally might like. We show one manner in which this plays out when smoothers are discussed in the next chapter.

Equation 1.18 has the additional benefit of retaining some of the same look and feel as conventional linear regression. This can lead to simplifications in the underlying mathematics, more effective computer algorithms, and more transparent interpretations. We show soon that Equation 1.18 leads to surprisingly flexible and effective fitting procedures in part because many complex functions can be well approximated by low-order polynomials and other relatively simple transformations.

But if Equation 1.18 is essentially multiple regression, where is the statistical learning? The answer will at this point probably seem a bit perplexing. The parametric structure of each basis function by itself can lead to a nonparametric fitting function when all of the pieces are used at once. And how these pieces are used can be substantially determined in an inductive manner by the data. It is a bit like constructing a highly nonlinear function from a large number of very small line segments connected end to end.

An Illustration

Consider, for example, how Equation 1.16 might be used within the following algorithm for a single response variable and a single predictor.

1. Select a target value of y, y_0, with its associated x_0.
2. Find the 20% of the observations whose values on x place them nearest to x_0.

3. Estimate for that subset of observations a cubic regression equation, weighting each of these observations so their weights decline linearly with distance from x_0.
4. From the results, construct the predicted value of y_0, $\hat{y}_0$.
5. Repeat Steps 1–4 for each unique value of x.

The result is a set of fitted values that can be called a "smoothed" version of y. The procedure goes under various names such as "locally weighted regression" (Cleveland, 1979). As we show later, there are often better ways to do locally weighted regression, but locally weighted regression can be written within a basis function framework.

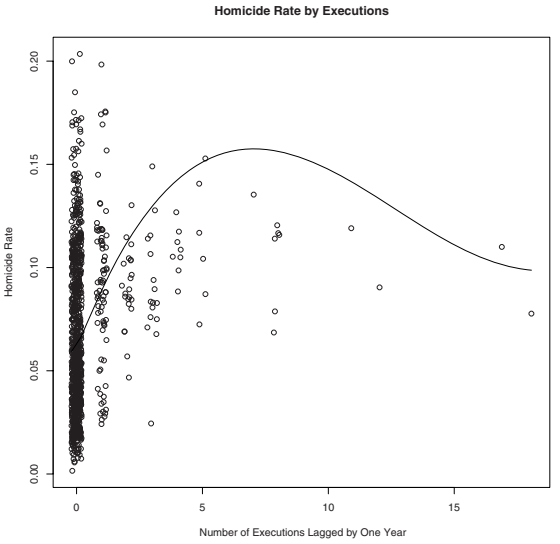


Fig. 1.10. The homicide rate per 1,000 as a function of the number of executions

Figure 1.10 provides an illustration. The data include the set of all 50 states each year from 1978 to 1998, for a total of 1000 observations. Each year, the homicide rate and the number of executions for capital crimes are recorded. Data such as these have been central in a recent debate about the deterrent value of the death penalty (Berk, 2005a).

In Figure 1.10, executions lagged by one year is on the horizontal axis, and the homicide rate per 1000 people is on the vertical axis. There are 1000 observations: 50 states times 20 years. Consequently, an observation is a state-year. To make the scatterplot more comprehensible, the number of executions has been jittered. But in most years, most states execute no one. Over 80% of the observations have zero executions. A very few states in a very few years execute more than five individuals. Years in which more than five individuals

in a state are executed represent about 1% of the data (i.e., 11 observations out of 1000).

The fitted values for a locally weighted regression are overlaid, constructed from the original (unjittered) data. One can see that for five executions or less, the relationship between the number of executions and the homicide rate one year later is positive. More executions are followed one year later by more homicides. Thus, there is a positive relationship for 99% of the data. When a given state in a given year executes six or more individuals, the relationship turns negative. With more executions, there are fewer homicides one year later. But one can see that there are almost no data supporting this relationship and in fact, a proper confidence interval around that portion of the curve would show that the true curve could easily be flat and even positive. In short, for 99% of the data the relationship is positive and for the atypical 1%, one really cannot tell. (For more details, see Berk, 2005a.)

Figure 1.10 represents a descriptive exercise. No data-generation mechanism was proposed, let alone a causal model. The goal was to provide an instructive visual summary of how the homicide rate is related to the number of executions one year earlier. A key point is that no functional form was imposed on the response function despite the application of parametric cubic regression in many local regions of the data. Had a single, parametric, linear relationship been imposed on the data, a misleading negative slope would have materialized. The few observations on the far right side of the plot are highly influential.

As an alternative fitting approach, consider a step function. For Equation 1.17, the choices of where to segment z can also be determined empirically. We consider in some detail later how this is done, but the basic idea is to choose each break point so that the fit is improved the greatest amount possible. For example, an initial break point at $z = 5$ might be found by trying all possible break points and choosing the one that reduced the error sum of squares the most. Then, within the two subsets of observations ($z > 5$ and $z \leq 5$), the next two break points would be independently chosen with the same goal in mind: to reduce the error sum of squares the largest amount possible. Classification and Regression Trees (Breiman et al., 1984) is based on this general idea and figure significantly in material presented in later chapters. Classification and regression trees can also be written as a set of basis functions that produce a nonparametric fit.

What would be the result if the indicator variable approach were applied to the data in Figure 1.10? The result is a single break point at $z = 1$. In effect, a split is made between no executions and one or more executions. The conditional mean when there are no executions is .066. The conditional mean when there is one execution or more is .094. When there is one or more executions, the homicide rate per 1000 people is about 50% greater the following year. An overlay of these fitted values on Figure 1.10 would reveal a step function. There would be a horizontal line between 0 and 1 at a value of .066, and another horizontal line between 1 and the 18 (the largest value

for the number of executions) at a value of .094. A vertical line at 1 would connect the two.

One important implication of the step function is that the fit is not improved a meaningful amount by segmenting the data any further. It is not useful to distinguish between, say, six executions and ten executions. This underscores the earlier point that data beyond five executions are far too sparse to be of much use.

1.5 Some Common Themes

Although the variety of procedures that are discussed later can differ greatly in look and feel, and although they often come from rather disparate intellectual traditions, there are several themes running through the material. These are usefully flagged before getting into lots of details.

- *Goals*—The goals of the procedures discussed are usually some mix of description, classification, and forecasting. Sometimes, the intent is to characterize a data-generation mechanism. But the techniques used in statistical learning are not usually considered causal models, and causal inference is rarely a goal. Likewise, statistical inference is not usually considered a key activity, sometimes because the data do not justify it, sometimes because a data generation mechanism has not been clearly articulated, and sometimes because credible procedures for statistical inference have not be developed.
- *Forecasting Skill*—In much of the statistical learning literature on which we rely, the true test of a procedure is not how well it fits the data on hand, but how well it forecasts. Forecasting skill is the gold standard. We show that although forecasting skill is widely accepted as the key performance criterion, how forecasting skill is defined and operationalized can vary substantially.
- *The Bias–Variance Tradeoff*—The bias-variance tradeoff is very visible for all of the procedures discussed. The basic point is that more flexible fitting functions will usually fit the data better but will typically generate less stable results. A better fit can imply less bias but more variance. Insofar as a true mean function exists, the fitted values have less systematic error. However, the fitted values for a new random sample from the same population will differ more from the original set of fitted values. Conversely, less flexible fitting functions will usually have more systematic error, but typically generate more stable results. A worse fit can imply more bias but less variance. Insofar as a true mean function exists, the fitted values have more systematic error. However, the fitted values for a new random sample from the same population will differ less from the original set of fitted values. A key goal in statistical learning can be to strike a useful balance between the bias and the variance or better still, find a way around

it. We also show that different statistical learning procedures can address the bias-variance tradeoff in different ways.

- *Loss Functions*—All of the statistical learning procedures discussed try to fit the data taking the disparities between the fitted values and the actual values into account. These disparities are "losses" that need to be weighted, aggregated, and minimized in some sensible fashion as the fitted values are computed. There are very important differences in how the disparities are handled from one statistical learning procedure to another.
- *Overfitting*—A major difficulty in statistical learning is overfitting. Very flexible fitting procedures will tend to respond to idiosyncratic features of the data, producing results that do not generalize well to new data. The results tend to be dataset-specific. The results can elicit very creative subject matter interpretations that, unfortunately, are stories about the noise not the signal.
- *Tuning*—For all of the statistical learning procedures examined, there are choices to be made about "tuning parameters." These are not population parameters of subject matter or statistical interest. They are parameters, much like dials on a machine, that determine how a procedure functions. There is often little statistical theory to guide the selection of tuning parameter values. Consequently, data analysts will proceed by craft lore or trial and error. There is nothing inherently wrong with such practices, but there will commonly be a strong temptation to try a large number of different tuning parameter settings. This can lead to "overtuning" and hence, overfitting. Sometimes the tuning is done using one or more measures of model performance, such as goodness-of-fit. Here, too, overuse can lead to overfitting even when the fit statistic has been designed to counter certain causes of overfitting.
- *Interpretability*—Within a regression framework, results that are difficult to interpret in subject matter terms, no matter how good the fit, are often of little use. This will sometimes lead to another kind of tradeoff. Fitting functions that perform very well by various technical criteria may stumble when the time comes to understand what the results mean. Important features of the data may be lost. It will sometimes be useful, therefore, to relax the technical performance criteria a bit in order to get results that make sense.
- *Differences That Make No Difference*—In almost every issue of journals that publish work on statistical learning and related procedures, there will be articles offering some new wrinkle on existing techniques, or even new procedures, often with strong claims about superior performance compared to some number of other approaches. Such claims are often data-specific but even if broadly true, rarely translate into important implications for practice. Often the claims of improved performance are small by any standard. Some claims of improved performance are unimportant for the subject matter problem being tackled. But even when the improvements seem

to be legitimately substantial, they often address secondary concerns. In short, the newest is not necessarily the best.

- *Rapid Development*—The concepts, understandings, and tools that fall under the rubric of statistical learning are evolving rapidly. It is very difficult to keep up with the field, and today's breakthrough can be tomorrow's bust. Moreover, the pace of technical development has to date vastly outstripped the pace at which hands-on experience with real data has accumulated. As a result, it is very difficult to provide grounded advice to data analysts working on real scientific and policy problems. There is so far relatively little data analysis lore for many of the newer tools. In addition, most popular software packages are several years behind the curve. Researchers who want to use the most recent advances either have to work in the software environment where the tools are being developed (e.g., R or Matlab), or in special-purpose proprietary packages such as the one available from Salford Systems (http://www.salford-systems.com/).
- *Data Quality Really Matters*—Just as in any form of regression analysis, good data are a necessary prerequisite. If there are no useful predictors, if the data are sparse, if key variables are highly skewed or unbalanced, or if the key variables are poorly measured, it is very unlikely that the choice of one among several statistical learning procedures will be very important. The problems are bigger than that. It is rare indeed when even the most sophisticated and powerful statistical learning procedures can overcome the liabilities of bad data.

1.6 Summary and Conclusions

The statistical learning emphasized in this book is treated as a form of regression analysis, broadly defined, with no necessary commitment a priori to any particular functional relationship between predictors and the response. The relationships between the predictors and the response are substantially determined from the data. The stance taken is within the spirit of procedures such as stepwise regression, but beyond allowing the data to determine which predictors are useful, the data are allowed to help determine what predictor functions are most appropriate. In practice, this means subcontracting a large part of the data analysis to one or more computer algorithms.

What role can subject matter "theory" have? Subject matter theory can be very important in

1. Framing the empirical questions to be addressed
2. Defining a data-generation mechanism
3. Designing and implementing the data collection
4. Determining which variables in the dataset are to be inputs and which are to be outputs
5. Settling on the values of tuning parameters

6. Deciding which results make sense

But none of these activities is necessarily formal or deductive, and they leave lots of room for interpretation. If the truth be told, subject matter theory plays much the same role in statistical learning as it does in most conventional analyses. But in statistical learning, there is often far less posturing.

At least as important as subject matter theory is information on how the statistical learning results are to be used. Central in these discussions is the concept of a loss function, which determines the costs of inaccurate fitted values. There is really no way to avoid a consideration of loss functions for any statistical learning approach (and for most other statistical procedures as well). A loss function typically is assumed, even if it is not explicitly acknowledged by the data analyst. And the loss function employed can have an enormous impact on the results. Subject matter expertise is necessarily brought to bear when loss functions are selected, and the practical applied decisions to be made also can play a key role.

Finally, the nature of exploratory data analysis needs to be briefly revisited to put both "data snooping" and statistical learning in their proper places. Data snooping is under-the-table exploratory data analysis. The data are studied at great length, various transformations are undertaken, a large number of statistical procedures are applied and then, only the best results are reported. This is a common ruse in most sciences that creates technical problems (e.g., invalidating statistical tests) and misleading results. An exploratory data analysis is presented as if it were a confirmatory data analysis.

A form of data snooping can also be undertaken with more apparent legitimacy when systematic model selection procedures are employed. If data used for model selection are also used to construct and evaluate the chosen model, data snooping is again at work. The procedures used may be forthrightly described, but as a formal matter, statistical inference can be undermined. The usual regression statistical inference is undertaken conditional upon a model known before the data are examined. There is a small but instructive literature showing that the unconditional distribution of the post model-selection estimator cannot be arrived at with sufficiently useful accuracy, even asymptotically (Freedman et al., 1988; Danilov and Magnus, 2004; Leeb and Pötscher, 2006). So, the usual conditional tests do not address the question that data analysts typically want to answer, and the unconditional tests can be very problematic. We return to this later. For now, the point is that model selection procedures share a lot with conventional exploratory data analysis.

Statistical learning is usually exploratory as well. But at least in principle, the exploration is undertaken by a set of very explicit rules represented in the algorithms employed. No one is hiding the ball. Equally important, we show later that great pains are taken to avoid the seductions of overfitting. There is often a training dataset to which the procedures are applied and a test dataset to determine whether the results are too good to be true. When there are no test data, there can be clever resampling techniques or helpful adjustments

to measures of fit that can sometimes achieve much the same result. Rather than reporting only the best results, a conscious effort is made to report only the honest results.

Despite these good intentions, statistical learning practice can be subverted by data snooping. In particular, a procedure that by itself does not overfit can be applied to the data many times. After each pass through the data, the results are examined, and the statistical learning algorithm is tuned in the hope of producing better results. This sequence of fitting attempts can lead to overfitting despite protections against overfitting built into the algorithm when it is applied just once. The difficulties are compounded when the tuning process goes unreported. Sensitivity to overfitting is an important strength of statistical learning. But that sensitivity does not confer immunity.

Exercises

The purpose of these exercises is to provide a bit of practice doing regression analyses by examining conditional distributions without the aid of conventional linear regression. You will see that regression analysis does not require a parametric model.

Problem Set 1

Load the R dataset "airquality" using data(airquality). Learn about the data set using help(airquality). Attach the dataset "airquality" using attach(airquality). If you do not have access to R, or choose to work with other software, exercises in the same spirit can be easily undertaken. Likewise, exercises in the same spirit can be easily undertaken with other data sets.

1. Using pairs(), construct of a scatterplot matrix for all of the variables except for "Month" and "Day." Describe the relationships between each pair of variables.

2.
3. boxplot Using boxplot(), construct side-by-side boxplots for ozone concentrations against month and ozone concentrations against day. Does the ozone distribution vary by month of the year and day of the month? In what ways?

4. What would one have to assume to use month or day of the month in scatterplots?

5. Construct a three-dimensional scatterplot with ozone concentrations as the response and temperature and wind speed as predictors. (Maybe use cloud() from the lattice package.) What patterns can you make out? It

is difficult to see much. Other kinds of plots for three variables are often more useful.

6. Construct a conditioning plot using coplot() with ozone concentrations as the response, temperature as a predictor, and wind speed as a conditioning variable. How does the conditioning plot attempt to hold wind speed constant?

7. Consider all the conditioning scatterplots. What common patterns do you see? What does this tell you about how ozone concentrations are related to temperature with wind speed held constant?

8. How do the patterns differ across the conditioning scatter plots? What does that tell you about how wind is related to ozone concentration holding temperature constant? What does that tell you about how the relationship between ozone concentrations and temperature can differ for different wind speeds?

9. Construct an indicator variable for missing data. Using table() or xtab(), cross-tabulate the indicator against month. What do you learn about the pattern of missing data? How might your earlier analyses using the conditioning plot be affected?

10. Write out the parametric regression model that seems to be most consistent with what you have learned from the conditioning plot. Try to justify all of the assumptions you are imposing.

11. Implement your regression model in R using lm() and examine the results. How do your conclusions about the correlates of ozone concentrations learned from the regression model compare to the conclusions about the correlates of ozone concentrations learned from the conditioning plot?

Problem Set 2

The purpose of this exercise is to give you some understanding about how the complexity of a fitting function affects the results of a regression analysis and whether popular measures of fit compensate sensibly.

1. Construct the data as follows. For your predictor: x = rep(1:20, times = 10). This will give you 200 observations with values 1 through 20. For your response: y = rnorm(200). This will give you 200 random draws from the standard normal distribution.

2. Plot the response against the predictor and describe what you see.

3. Apply a bivariate regression using lm() and then glm(). Describe what overall conclusions you draw from the two sets of output. (The fit should be the same but the output from the two procedures are a bit different.)

4. Repeat the two bivariate regressions with the predictor as a factor. Use the same R code as before but use as.factor(x) instead of x.

5. How do the two sets of output differ from the previous sets? Focus on the overall measures of fit. Do the adjustments for the degrees of freedom used up seem to be effective in this case?

Problem Set 3

This purpose of this exercise is to explore how degrees of freedom used up across models can affect results and whether popular measures of fit compensate sensibly.

Construct a dataset as follows. Using R, construct 50 variables as independent random draws from a standardized normal distribution. Each predictor should have 100 observations. One easy way to do this is drawing 50 times 100 values using rnorm() and then formatting the result into a 50 by 100 matrix with matrix(). It will turn out to be helpful if that matrix is made into a data frame using data.frame(). Then, attach the data frame using the attach() command.

Now run a linear regression using that data frame. Apply lm() and assign the output to some name so that you can retrieve it later. To keep things simple, the only argument should be the data frame name. The first column will automatically be chosen as the response variable. Do not look at the output (yet).

Now apply a stepwise regression to the data. There are several stepwise regression procedures in R, but stepAIC() in the MASS library is one good one. Just feed the saved output object from lm() into stepAIC(). Again, save the output by assigning it to some name. For this exercise, accept the default settings. Do not look at the output (yet).

1. What do you expect the output from the first regression analysis (not the stepwise regresson) to look like: the regression coefficients, the t-tests, the R^2, the adjusted R^2, and the F-test?

2. Now look at the output from the first regression. How does the actual output compare with your expectations?

3. What do you conclude about how well linear regression produces results consistent with how you know the data were generated?

4. Now examine the output from the stepwise regression: the regression coefficients, the t-tests, the R^2, the adjusted R^2, and the F-test. How do these compare to the output from your initial regression analysis?

5. From that comparison, what are the possible implications for model selection and overfitting?

6. How would these implications change if the ratio of the number of observations to the number of predictors were much larger (e.g., 10 predictors with 100 observations) or much smaller (e.g., 90 predictors with 100 observations)? Try it. What happens? How do the comparisons between the conventional regression results and the stepwise regression results change depending on the ratio of the number of observations to the number of predictors?

7. What are some lessons for model selection and overfitting?

2

Regression Splines and Regression Smoothers

2.1 Introduction

This chapter launches a more detailed examination of statistical learning within a regression framework. Once again, the focus is on conditional distributions. But now, the mean function for a response variable is central. How does the mean vary with different predictor values? The intent is to begin with procedures that have much the same look and feel as conventional linear regression and gradually move toward procedures that do not.

2.2 Regression Splines

A "spline" is a thin strip of wood that can be easily bent to follow a curved line (Green and Silverman, 1994: 4). Historically, it was used in drafting for drawing smooth curves. Regression splines, a statistical translation of this idea, are a way to represent non-linear, but unknown, mean functions.

Regression splines are not used a great deal in empirical work. As we show, there are usually better ways to proceed. Nevertheless, it is important to consider them, at least briefly. They provide an instructive transition between conventional parametric regression and the kinds of smoothers commonly seen in statistical learning.

2.2.1 Applying a Piecewise Linear Basis

For a piecewise linear basis, the goal is to fit the data with a broken line (or hyperplane) such that at each break point the left-hand edge meets the right-hand edge. When there is a single predictor, for instance, the fit is a set of straight line segments, connected end to end, sometimes called "piecewise linear." Figure 2.1 is a simple illustration using three straight lines joined end to end. There is a response variable represented by y and a predictor represented by x. For now, only the fitted values are shown.

R.A. Berk, *Statistical Learning from a Regression Perspective*,
DOI: 10.1007/978-0-387-77501-2_2,

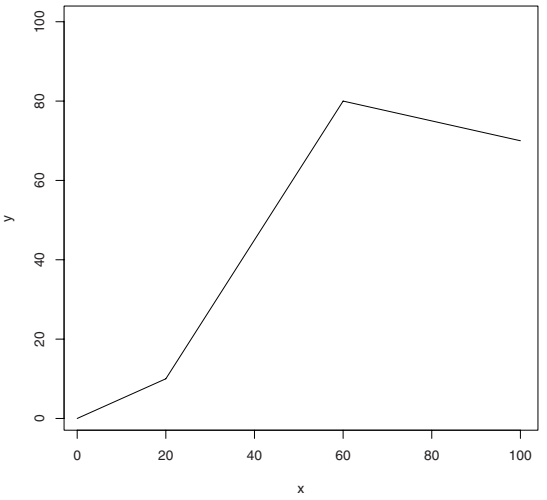


Fig. 2.1. An illustration of piecewise linear function with two knots.

Constructing such a function for the conditional means is straightforward in principle. First, one decides where the break points on x will be. If there is a single predictor, as in this illustration, the break points might be chosen after examining a scatter plot of y on x. When possible, subject matter expertise should also be used to help determine the break points. For example, x might be years, and then the break points might be determined by specific historical events. Thus, y might be a measure of a river's biodiversity, and x might be time in months, with one breakpoint representing the removal of a major dam and another breakpoint representing a toxic chemical spill. Let the break points here be defined at $x = a$ and $x = b$ (with $b > a$). In Figure 2.1, $a = 20$ and $b = 60$. Such break points are often called "knots."

The second step is to define two indicator variables to represent the break points. Here, the first (I_a) is equal to 1 if x is greater 20 and equal to 0 otherwise. The second (I_b) is equal to 1 if x is greater than 60 and equal to 0 otherwise. We let x_a be the value of x at the first break point, and x_b be the value of x at the second break point.

The third step is to define the mean function. Because at this point description is the primary goal, the conditional mean of y is represented by $\bar{y}|x$ rather than by $E(y|x)$. The latter implies that Y is a random variable. For now, it does not matter whether Y is a random variable. Then,

$$\bar{y}|x = \beta_0 + \beta_1 x + \beta_2(x - x_a)I_a + \beta_3(x - x_b)I_b. \tag{2.1}$$

Looking back at Equation 1.15, it is apparent that there are four transformations of X, $h_m(X)$s, in which the first function of x is a constant.

The mean function for x less than a is

$$\bar{y}|x = \beta_0 + \beta_1 x. \tag{2.2}$$

In Figure 2.1, β_0 is zero, and β_1 is positive.

For values of x greater than a but smaller than b, the mean function becomes

$$\bar{y}|x = (\beta_0 - \beta_2 x_a) + (\beta_1 + \beta_2)x. \tag{2.3}$$

For a positive β_1 and β_2, the line beyond $x = a$ is steeper because the slope is $(\beta_1 + \beta_2)$. The intercept is lower because of the second term in $(\beta_0 - \beta_2 x_a)$. This too is consistent with Figure 2.1. If β_2 is negative, the reverse would apply.

For values of x greater than b, the mean function becomes,

$$\bar{y}|x = (\beta_0 - \beta_2 x_a - \beta_3 x_b) + (\beta_1 + \beta_2 + \beta_3)x. \tag{2.4}$$

For these values of x, the slope is altered by adding β_3 to the slope of the previous line segment. The intercept is altered by subtracting $\beta_3 x_b$. The sign and magnitude of β_3 determine if the slope of the new line segment is positive or negative and how steep it is. The intercept will shift accordingly. In Figure 2.1, β_3 is negative and large enough to make the slope negative. The intercept is increased substantially.

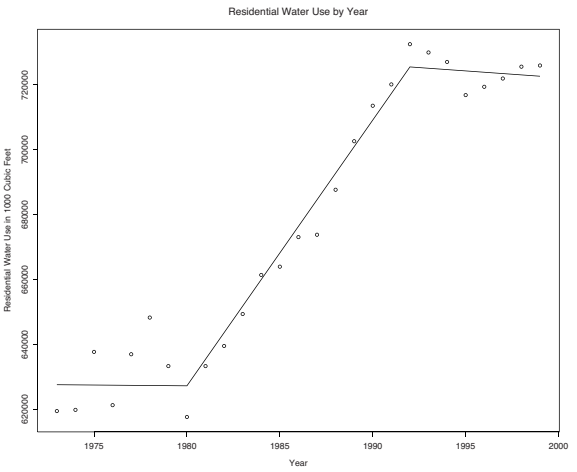


Fig. 2.2. A piecewise linear basis applied to water use by year.

Figure 2.2 shows a three-piece linear regression spline applied to water use data from Tokyo over a period of 27 years. Residential water use in 1000s of

cubic feet is on the vertical axis. Year is on the horizontal axis. The locations of the break points were chosen after inspecting the scatterplot, with some reliance on subject matter expertise about residential water use in Japan.

It is clear that water use was flat until about 1980, then increased linearly until about 1996, and then flattened out again. The first break point may correspond to a transition toward much faster economic and population growth. The second break point may correspond to the introduction of more water-efficient technology. But why the transitions are so sharp is mysterious. One possibility is that the break points correspond in part to changes in how the water use data were collected or reported.

It is perhaps most common to see regression splines fit to data in which time is used as the sole predictor. The end-to-end connections between line segments lead naturally to processes that unfold over time. The line segment on the right side of a knot begins where the line segment on the left side of the knot ends. But there is nothing about linear regression splines requiring that time be a predictor. For example, the response could be crop production per acre and the sole predictor could be the amount of phosphorus fertilizer applied to the soil. Crop production might increase in approximately a linear fashion until an excess of phosphorus caused other kinds of nutritional difficulties. At that point, crop yields might decline in roughly a linear manner.

Fitting line segments to data provides an example of "smoothing" a scatterplot, or applying a "smoother." The line segments are used in place of the data to characterize how x and y are related. The intent is to highlight key features of any association while removing unimportant details. This can often be accomplished by constructing fitted values in a manner that makes them more homogeneous than the set of conditional means of y computed for each unique value of x.

Imagine a scatterplot in which the number of observations was large enough so that for each value of x there were at least several values of y. One could compute the mean of y for each x-value. If one then drew a straight line between each of the adjacent conditional means, the resulting smoother would be an interpolation of the conditional means and as rough as possible. At the other extreme, imposing a single linear fit on all of the means at once would produce the smoothest smoother possible. Figure 2.2 falls somewhere in between. How to think about the degree of smoothness more formally is addressed later.

For a piecewise linear basis, one can simply compute functions such as Equation 2.1 with ordinary least squares. With the regression coefficients in hand, fitted values are easily constructed. Indeed, many software packages compute and store fitted values on a routine basis. Also widely available are procedures to construct the matrix of regressors, although it is not hard to do so one term at a time using common transformation capabilities. For example, the library *spline* has a procedure bs() that constructs a B-spline basis (discussed later) that can be easily used to represent the predictor matrix for piecewise linear regression.

In contrast to most applications of conventional linear regression, there would typically be little interest in the regression coefficients themselves; they are but a means to an end. The point of the exercise is to superimpose the fitted values on a scatterplot so that the relationship between y and x can be more effectively visualized. As we show later, and as was briefly anticipated in the last chapter, model selection will not necessarily be the same as regressor selection.

2.2.2 Polynomial Regression Splines

Smoothing a scatterplot using a piecewise linear basis has the great advantage of simplicity in concept and implementation. And by increasing the number of break points, very complicated relationships can be approximated. However, in most applications there are good reasons to believe that the underlying relationship is much smoother than can be easily represented with a set of straight line segments.

Greater continuity can be achieved by using polynomials in x for each segment. Cubic functions of x are a popular choice because they strike a nice balance between flexibility and complexity. When used to construct regression splines, the fit is sometimes called "piecewise cubic." The cubic polynomial serves as a "truncated power series basis" in x.

Unfortunately, simply joining polynomial segments end to end is unlikely to result in a visually appealing fit where the polynomial segments meet. The slopes of the two lines will often appear to change abruptly even when that is inconsistent with the data. Far better visual continuity usually can be achieved by constraining the first and second derivatives on either side of each break point to be the same.

Putting this all together, one can generalize the piecewise linear approach and impose the continuity requirements. Suppose there are K interior break points, usually called "interior knots." These are located at $\xi_1 < \cdots < \xi_K$ with two boundary knots added at ξ_0 and ξ_{K+1}. Then, one can use piecewise cubic polynomials in the following regression formulation,

$$\bar{y}|x = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \sum_{j=1}^{K} \theta_j (x - x_j)_+^3, \tag{2.5}$$

where the "+" indicates the positive values from the expression inside the parentheses, and there are $K + 4$ parameters whose values need to be computed. This leads to a conventional regression formulation with a matrix of predictor terms having $K + 4$ columns and N rows. Each row would have the corresponding values of the piecewise cubic polynomial function evaluated at the single value of x for that case. There is still only a single predictor, but now there are $K + 4$ basis functions.

The output for the far-right term in Equation 2.5 may not be apparent at first. Suppose the values of the predictor are arranged in order from low

to high. For example, $x = [1, 2, 4, 5, 7, 8]$. Suppose also that x_j is located at an x-value of 4. Then, $(x - x_j)_+^3 = [0, 0, 0, 1, 27, 64]$. The knot-value of 4 is subtracted from each value of x, the negative numbers set to 0, and the others cubed. All that changes from knot to knot is the value of x_j that is subtracted. There are K such knots and K such terms in the regression model.

Figure 2.3 shows the water use data again, but with a piecewise cubic polynomial overlaid that imposes the two continuity constraints. The fit looks quite good to the eye and captures about 95% of the variance in water use. But, in all fairness, the scatterplot did not present a great challenge. The point is to compare Figure 2.2 to Figure 2.3 and note the visual difference. The linear piecewise fit also accounted for about 95% of the variance. Which plot would be more instructive in practice would depend on the use to be made of the fitted values and on prior information about what a sensible $f(X)$ might be. The regression coefficients ranged widely and, as to be expected, did not by themselves add any useful information. The story was primarily in the fitted values.

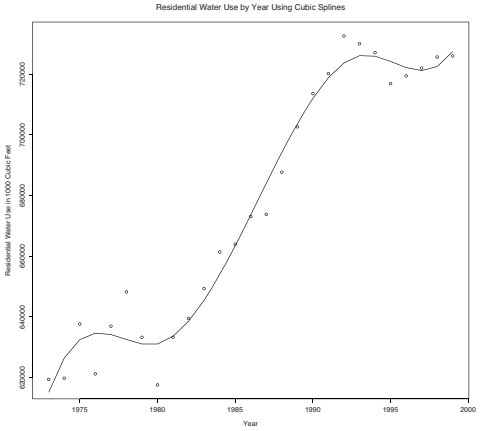


Fig. 2.3. A piecewise cubic polynomial applied to water use by year.

2.2.3 Natural Cubic Splines

Fitted values for piecewise cubic polynomials near the boundaries of x can be unstable because they fall at the ends of polynomial line segments where there are no continuity constraints, and where there may be little data. By "unstable" one means that a very few observations, which could vary over random samples from the same population, produce substantially different fitted values near the boundaries of x. As a result, the plot of the fitted values near the boundaries could look rather different from sample to sample.

Sometimes, constraints for behavior at the boundaries are added to increase stability. One common constraint imposes linearity on the fitted values beyond the boundaries of x. This introduces a bit of bias because it is very unlikely that if data beyond the current boundaries were available, their relationship with the response would be linear. However, the added stability is often worth it. When these constraints are added, the result is a "natural cubic spline."

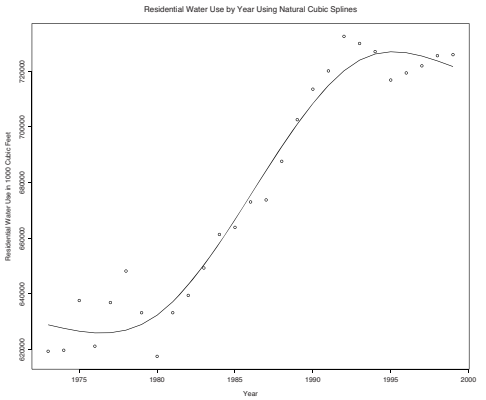


Fig. 2.4. Natural cubic regression splines applied to water use by year.

Figure 2.4 shows again a plot of the water use data on year, but now with a smoother constructed from natural cubic splines. One can see that the fitted values near the boundaries of x are somewhat different from the fitted values near the boundaries of x in Figure 2.3. The fitted values in Figure 2.4 are smoother, which is the desired result. There is one less bend near both boundaries. More generally, how one can formulate the boundary constraints is discussed in Hastie et al. (2001: Section 5.2.1).

The option of including extra constraints to help stabilize the fit provides an example of the bias–variance tradeoff. This is a topic to which we return many times in the pages ahead. For now, an informal overview for natural cubic splines may be useful.

The bias–variance tradeoff addresses some important properties of fitted values when the data are a random sample from a population or a realization of a stochastic process. Bias and variance refer to what can happen over a limitless number of hypothetical, independent, random samples or realizations; the context is the usual frequentist thought experiment. Therefore, the bias–variance tradeoff is only directly relevant when statistical inference is on the table and does not formally provide much insight when summary statistics are being used solely for description.

When constraints are imposed on a fitting process to make the fitted values less variable, bias in the fitted values can be introduced. The means of the fitted values over many samples or realizations will often be farther from the true conditional means of the response variable, which are the values one wants to estimate. However, in repeated independent random samples, or independent realizations of the data, the fitted values will vary less. When the fit is smoother, each fitted value is constructed, in effect, from a larger number of y-values. This increases stability in the same way that larger samples in general provide estimates with a smaller variance. Conversely, but using the same reasoning, a rougher fit can imply less bias but more variance over repeated samples or realizations. A tradeoff naturally follows.

Ideally, just the right amount of bias can be combined with just the right amount of variance so that over repeated random samples or realizations, the fitted values would be on the average as close to true response variable values as possible. "Close" can be operationalized in several ways, but it is often desirable to work with a test sample and then try to minimize the mean of the squared deviations between the fitted values and the observed values of the response variable (i.e., the mean squared error in a test sample).

For piecewise cubic polynomials and natural cubic splines, the degree of smoothness is primarily a function of the number of interior knots. In practice, the smaller the number of knots, the smoother are the fitted values. A smaller number of knots means that there are more constraints on the pattern of fitted values because there are fewer end-to-end, cubic line segments used in the fitting process. Consequently, less provision is made for potential twists and turns.

But placement matters too. Ideally, knots should be located where it is thought that the $f(X)$ is changing most rapidly. In some cases, inspection of the data, coupled with subject matter knowledge, can be used to determine the number and placement of knots. The water use data just considered were analyzed in this manner.

Alternatively, the number and placement of knots can be approached as a model selection problem. Any of the fit statistics discussed in the last chapter, such as the GCV, can be used to determine the number of knots, given a set of candidate locations. The number of knots translates into a penalty for the number of regression parameters whose values are being estimated from the data. The penalty increases with the number of knots, just as the penalty would normally increase with the number of regression parameters whose values were not known a priori. Then, the goal is to choose the knot number that minimizes the fit statistic. Knot selection is essentially regressor selection. In other words, a set of potential knots is specified, and fit statistics are used to determine which knots are really needed.

The fit statistics are largely silent on where to place the knots. Two models with the same number of knots can produce very different fitted values if the placement of the knots substantially differs. Two models with a very different number of knots may fit the data about the same, depending on

where the knots are placed. Moreover, absent subject matter information, knot placement has been long known to be a difficult technical problem, especially when there is more than one predictor (de Boors, 2001). The fitted values are related to where the knots are placed in a very complicated manner. Fortunately, methods discussed later sidestep the knot location problem.

Even if a good case for candidate knot locations can be made, one must be careful about taking any of the fit measures too literally. First, there will often be several models with rather similar values, whatever the kind of fit statistic used. Then, selecting a single model as "best" using the fit measure alone may amplify a small numerical superiority into a large difference in the results, especially if the goal is to interpret how the predictors are related to the response. Some jokingly call this "specious specificity." Second, the same issues can arise when comparing the models selected by the different kinds of fit statistics. The impact of very small differences in fit can lead to very large difference in the results. Third, one must be a very careful to not let small differences in the fit statistics automatically trump subject matter knowledge. The risk is arriving at a model that may be difficult to interpret, or effectively worthless.

In summary, for regression splines of the sort just discussed, there is no straightforward way to arrive at the best tradeoff between the bias and the variance because there is no straightforward way to determine knot location. A key implication is that it is very difficult to arrive at a model that is demonstrably the best. Fortunately, there are other approaches to smoothing that are more promising.

2.2.4 *B*-Splines

In practice, data analyses using piecewise cubic polynominals and natural cubic splines are rarely constructed directly from polynomials of x. They are commonly constructed using a B-spline basis, largely because of computational convenience. A serious discussion of B-splines would take us far afield and accessible summaries can be found in Gifi (1990) and Hastie et al. (2001). Nevertheless several observations are worth making.

The goal is to construct transformations of x that allow for a cubic piecewise fit but that have nice numerical properties and are easy to manipulate. B-splines do well by all three criteria. They are computed in a recursive manner from very simple functions to more complex ones, and consistent with the approach to basis functions taken here, can be represented as a linear basis expansion.

For a series of knots, which usually include several beyond the upper and lower boundaries of x, indicator variables are defined for each region marked off by the knots. If a value of x falls within a given region, the indicator variable for that region is coded 1, and coded 0 otherwise. For example, if there is a knot at an x-value of 2 and the next knot at an x-value of 3, the x-values between them form a region with its own indicator variable coded 1

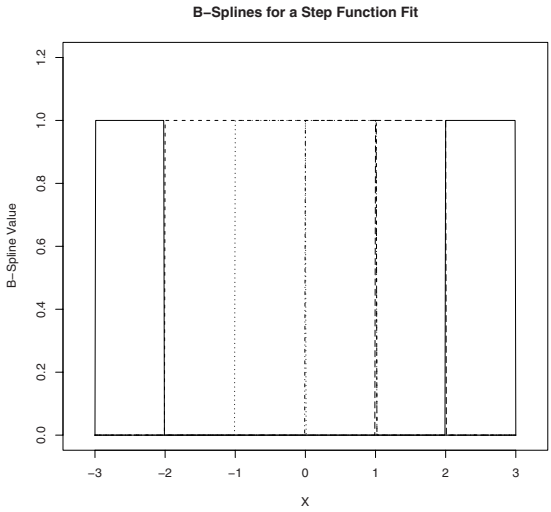


Fig. 2.5. Degree zero B-splines.

if the value of x falls in that region (e.g., $x = 2.3$), and coded 0 otherwise. The result is a set of indicator variables, with values of 1 or 0, for each region. These indicator variables define a set of degree zero B-splines.

Figure 2.5 is an illustration with interior knots at -2, -1, 0, 1, and 2. Using the indicator variables as regressors will produce a step function when y is regressed on x; they are the basis for a step function fit. The steps will be located at the knots.

Next a transformation can be applied to the degree zero B-splines. (See Hastie et al., 2001: 160–163). The result is a set of degree one B-splines. Figure 2.6 shows the set of degree one B-splines derived from the indicator variables shown in Figure 2.5. The triangular shape is characteristic of degree one B-splines, and implies that the values for each spline are no longer just 0 or 1, but proportions in between as well.

Degree one B-splines are the basis for linear piecewise fits. Here, the regressor matrix includes eight columns whose values appear in Figure 2.6. The content of each column is the B-spline values for each value of x. Regressing a response on that matrix will produce a linear piecewise fit with knots at -2, -1, 0, 1, and 2.

A transformation of the same form can now be applied to the degree one B-splines. This leads to a set of degree two B-splines that are the basis for a quadratic piecewise fit. For this illustration, there is now a matrix with nine columns that can serve as a regressor matrix. The set of such B-splines is shown in Figure 2.7 and as before, the shapes are characteristic.

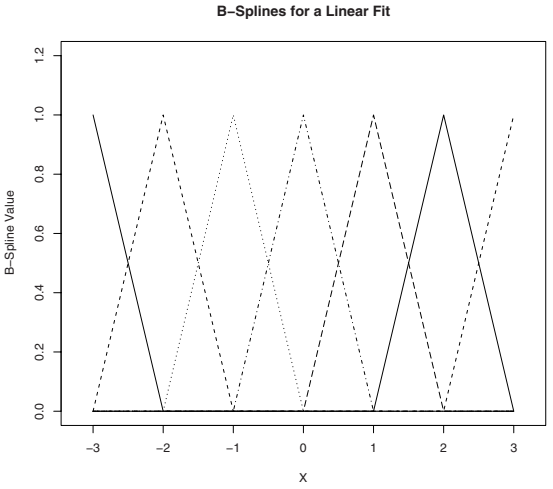


Fig. 2.6. Degree one B-splines.

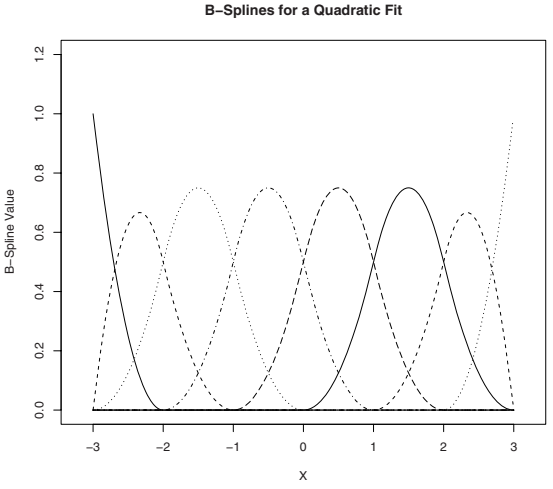


Fig. 2.7. Degree two B-splines.

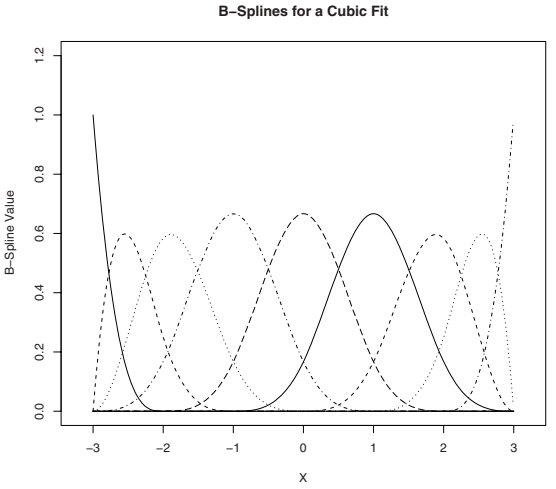


Fig. 2.8. Degree three B-splines.

The same kind of transformation can then be applied to the degree two B-splines. The result is a set of degree three B-splines that are the basis for a cubic piecewise fit. Figure 2.8 shows the set of degree three splines, whose shapes are, once again, characteristic. They can be used to construct a regressor matrix with nine columns.

All splines are a linear combinations of B-splines; B-splines are a basis for the space of all splines. They are also a well-conditioned basis because they are fairly close to orthogonal, and they can be computed in a stable and efficient manner. For our purposes, the main point is that B-splines are a computational device used to construct cubic piecewise fitted values. When such smoothers are employed, B-splines are doing the work behind the scenes.

2.3 Penalized Smoothing

The placement of knots, the number of knots, and the degree of the polynomial are subject to manipulation by a data analyst. All three can be used to construct a highly flexible fitting function that will track the data well. Because a good fit is typically considered desirable, there is sufficient reason in practice to worry about overfitting. The pull toward constructing a good fit can be very strong.

The fit statistics considered earlier can provide some protection against overfitting. They can help compensate for the amount of flexibility built into a given fitting function. However, they function indirectly. They are applied

after a model has been constructed to obtain a more honest measure of fit quality that can sometimes inform future fitting attempts.

A useful alternative is to alter the fitting process itself so that potential overfitting of a given model comes at a price. In particular, a penalty can be introduced into the loss function to be optimized that imposes increasing losses with increasing flexibility, regardless of how well the model is otherwise doing. In part because this approach has wide applicability, it is worth our attention now. Penalized fitting procedures figure significantly in this and later chapters.

2.3.1 Shrinkage

All of the procedures discussed in this chapter can be formulated as a conventional regression analysis. The procedures vary in the regressor matrix employed and how that matrix is determined. Whatever the regressor matrix used, there will be a set of regression coefficients. The larger the absolute value of these coefficients, other things being equal, the more the fitted values can vary.

To get some feel for this, consider a conventional regression analysis with an indicator variable as the sole regressor. If its regression coefficient equals zero, the fitted values will be a straight line, parallel to the x-axis, located at the unconditional mean of the response. As the regression coefficient increases in absolute value, the resulting step function will have a step of increasing size. The fit becomes more rough. More generally, the potential for rougher fit is greater with larger regression coefficients. Insofar as the roughness results from fitting idiosyncratic features of the data, there is overfitting. There are situations, therefore, in which it can be useful to control how large the regression coefficients are allowed to become.

A number of proposals have been offered for how to control the magnitude of regression coefficients. (See Ruppert et al., 2003: Section 3.5 for a very accessible discussion. Two popular suggestions are

1. Constrain the sum of the absolute values of the regression coefficients to be less than some constant C (sometimes called an L_1–penalty).
2. Constrain the sum of the squared regression coefficients to be less than some constant C (sometimes called an L_2–penalty).

The smaller the value of C is, the smaller the sum. The smaller the sum, the smaller is the typical magnitude of the regression coefficients. In part because the units in which the regressors are measured will affect how much each regressor contributes to the sum, it can make good sense to work with standardized regressors. The intercept does not figure in either constraint and is usually addressed separately.

Both constraints lead to "shrinkage methods." The regression coefficients can be "shrunk" toward zero, making the fitted values more homogeneous.

The goal is to introduce a small amount of bias into the computed regression coefficients in trade for a substantial reduction in their variance. There may also be subject matter reasons for preferring a smoother set of fitted values. Subject matter theory and/or past research may suggest that the response is a relatively smooth function of the predictors.

Shrinkage methods can be applied with the usual regressor matrix or with regressor matrices of the sorts we have considered in this chapter. With our focus on statistical learning, the latter is emphasized shortly. We start, however, within a conventional multiple regression framework and p predictors. We show that there can be two somewhat different goals: to construct more stable fitted values and to determine which regressors can be included as predictors. Shrinkage methods can be viewed as a form of "regularization," which figures significantly in later chapters.

One also can recast some measures of fit discussed in the last chapter within a shrinkage framework. The total number of regression coefficients to be estimated can serve as a constraint and is sometimes called an L_0–penalty. Maximizing the adjusted R^2, for example, can be seen as maximizing the usual error sum of squares subject to a penalty for the number of regression coefficients in the model (Fan and Li, 2006).

Ridge Regression

Suppose one adopts the constraint that the sum of the p squared regression coefficients is less than C. The L_2 constraint leads directly to ridge regression. The task is to obtain values for the regression coefficients so that

$$\hat{\boldsymbol{\beta}} = \min_{\beta} \left[\sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right]. \tag{2.6}$$

In Equation 2.6, the usual expression for the error sum of squares has a new component. That component is the sum of the squared regression coefficients multiplied by a constant λ. When Equation 2.6 is minimized in order to obtain $\hat{\boldsymbol{\beta}}$, the sizes of the squared regression coefficients are taken into account.

For a given value of λ, the larger the $\sum_{j=1}^{p} \beta_j^2$ is, the larger the increment to the error sum of squares. The $\sum_{j=1}^{p} \beta_j^2$ can be thought of as the penalty function. For a given value of $\sum_{j=1}^{p} \beta_j^2$, the larger the value of λ is, the larger the increment to the error sum of squares; λ determines how much weight is given to the penalty. In short, $\sum_{j=1}^{p} \beta_j^2$ is what is being constrained, and λ imposes the constraint. C is inversely related to λ. The smaller the value of C, the larger is the value of λ.

It follows that the ridge regression estimator is

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}, \tag{2.7}$$

where $\mathbf{I}$ is a $p \times p$ identity matrix. The column of 1s for the intercept is dropped from $\mathbf{X}$.

In Equation 2.7, λ plays same role as in Equation 2.6, but can now be seen as a tuning parameter. It is not an estimate of some feature of a population or a stochastic process. Its role is to help provide an appropriate fit to the data and can be altered directly by the data analyst. As such, it has a different status from the regression coefficients, whose values are determined through the minimization process itself, conditional upon the value of λ.

The value of λ is added to the main diagonal of the cross-product matrix $\mathbf{X}^T\mathbf{X}$, which determines how much the estimated regression coefficients are "shrunk" toward zero (and hence, each other). A λ of zero produces the usual least squares result. As λ increases in size, the least squares regression coefficients approach zero, and the fitted values are smoother. In effect, the variances of the predictors are being increased with no change in the covariances between predictors or with the response variable. This is easy to appreciate in the case of a single predictor. For a single predictor, the regression coefficient is the covariance of the predictor with the response divided by the variance of the predictor. So, if the covariance is unchanged and the variance is increased, the absolute value of the regression coefficient is smaller.

The results are not invariant to the scales used for the predictors; the regression coefficients obtained will differ in a complicated manner depending on the units in which the predictors are measured. It is common, therefore, to standardize the predictors before the estimation begins. However, standardization is just a convention and does not solve the problem of the results being scale-dependent. Knowing how much the average response changes in standard deviation units for a one standard deviation change in a predictor conveys little unless one also knows the size of the two standard deviations. And those standard deviations are scale-dependent.

A key issue is how the value of λ is chosen. One option is trial and error. Different values of λ are tried until the desirable amount of smoothness is achieved. Alternatively, the value of λ is selected by some measure of prediction error such as the cross-validation statistic. The value of λ is chosen to maximize prediction accuracy. Both methods can lead to overfitting insofar as many different models are applied to the training data.

What one makes of output from a ridge regression depends substantially on the usual issues. If estimation is an important goal, one must be able to credibly argue that for each configuration of x-values, one can treat the data on hand as a random sample or realization, as discussed earlier. Then, one must meet the usual regression assumptions. If, for example, there are omitted predictors, whether the resulting biases are likely to be large enough to matter in practice would need to be addressed on a case-by-case basis.

However, ridge regression introduces some additional complications. The estimates of the regression coefficients and hence, the fitted values, are biased by design. If hypothesis tests are undertaken and conventional regression output used, the reported p-values are no longer accurate. And if conventional confidence intervals are constructed, they do not have their usual coverage. The regression estimates are necessarily offset by a systematic but unknown

amount. We return to this matter a little later after other shrinkage procedures are discussed.

Ridge regression was first developed to address the instability of estimated regression coefficients when regressors are highly correlated. It is now appreciated that the applications are broader. Here, we are interested at least as much in description as in estimation, and ridge regression provides one means to alter the smoothness of the fitted values. Also, shrinkage methods can be given a Bayesian interpretation in which the regression coefficients are shrunk toward a prior joint distribution of the regression coefficients. Some researchers find this instructive.

The Lasso

Suppose that one now adopts the constraint that the sum of the absolute values of the regression coefficients is less than some constant. The L_1 constraint leads to a regression procedure known as the lasso (Tibshirani, 1996) whose estimated regression coefficients are defined by

$$\hat{\beta} = \min_{\beta} \left[\sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right]. \tag{2.8}$$

Unlike the ridge penalty, the lasso penalty leads to a nonlinear estimator, and a quadratic programming solution is needed. As before, the value of λ is a tuning parameter, determined empirically, usually through some measure of prediction error. Just as with ridge regression, a λ of zero yields the usual least squares results. As the value of λ increases, the regression coefficients are shrunk toward zero.

Hastie and his colleagues (2001: Section 3.4.5) place ridge regression and the lasso in a larger context in order to compare them to each other and to other procedures. A major interest is the patterns of shrinkage as the λ changes. Ridge regression tends to shrink the coefficients so that they all reach zero together as λ gets large. The lasso shrinks the coefficients so that some reach zero well before others as λ gets large. Thus, the lasso performs in a manner that has some important commonalities with model selection procedures used to choose a subset of regressors. Rosset and Zhu (2007) consider the path that the regression coefficients take as the value of λ changes, place the lasso in a class of regularization processes in which the solution path is piecewise linear, and then develop a robust version of the lasso. Wang et al. (2007) combine quantile regression with the lasso to derive another robust variable selection approach. We show later that the lasso has some interesting connections to boosting. In short, the lasso is more than a regularization procedure. It can help to provide useful insights about a wide variety of statistical tools.

Of late, there has been a lot of interest in the theoretical properties of the lasso and related procedures (Fan and Li, 2006; Meinshausen and Bühlmann,

2006). In particular, assuming that one has a set of predictors that includes all of those that belong in the model, as well as some number of irrelevant predictors, a key question is whether the lasso selects the correct predictors, at least as the sample size increases without limit. At this point, the answer seems to be sometimes it does and sometimes it does not. In particular, certain patterns of moderate to high correlations between predictors can lead to inappropriate predictors being selected along with the correct ones. Moreover, it can be very difficult to know with a real dataset whether or the problematic relationships between the predictors exist. In part as a response, Zou (2006) has proposed the adaptive lasso, which in theory is an improvement. The wrinkle is to employ "cleverly chosen" weights for the regression coefficients in the L_1 penalty function (Zou, 2006: section 3.1). The weights, in turn, are determined by another tuning parameter (in addition to λ). Finally, concerns have been raised about how well the lasso performs when there are heavy-tailed disturbance distributions or outliers. One response is to combine the lasso with quantile regression so that larger residuals are given relatively less weight in the fitting process (Wang et al., 2007).

In practice, the overriding problem with the lasso is the usual one: the underlying regression formulation has to be effectively correct. The data were in fact generated by a process represented with sufficient accuracy by a particular linear regression model. It is just that one has a dataset that includes not only the correct regressors but some incorrect ones, and the data analyst does not know which is which. The proper kind of shrinkage will reveal which regressors belong in the model. Alternatively, one has precisely the correct predictors in the dataset, but better performing estimates might be obtained through regularization. In either case, however, statistical inference for the lasso suffers from the same complications as ridge regression. Conventional expressions for confidence intervals and hypothesis tests do not apply.

The Elastic Net

If an important goal of a regression data analysis is to reduce the complexity of the model, the lasso has some advantages over ridge regression. But the lasso can also run into problems (Zou and Hastie, 2005). For example, when the number of predictors is larger than the number of observations (which is common with microarray data), the number of predictors selected cannot exceed the number of observations. In addition, there are the problems already noted with the selection of some inappropriate predictors.

In response to these difficulties, Zou and Hastie (2005) combine the penalties from ridge regression and the lasso. The result, called the elastic net, is

$$\hat{\beta} = \min_{\beta} \left[\sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda_1 \sum_{j=1}^{p} |\beta_j| + \lambda_2 \sum_{j=1}^{p} \beta_j^2 \right]. \qquad (2.9)$$

Minimization of Equation 2.9 produces what Zou and Hastie (2005) call "naive" coefficients that need to be adjusted further. The adjustment is simple, and some initial applications and simulations suggest that the elastic net can improve on the lasso. Of course, the regression model still needs to be credible, and even if it is, conventional expressions for confidence intervals and hypothesis tests are inappropriate.

The Dantzig Selector

The Dantzig Selector is another shrinkage estimator that can be used for variable selection (Candes and Tao, 2007). It seems to perform especially well when the number of predictors is large relative to the number of observations and even allows for the number of predictors to be larger than the number of observations. One must assume that the true set of regression coefficients is "sufficiently sparse" so that a substantial number of predictors actually have regression coefficients of zero. In effect, this guarantees identifiability. If in practice the linear regression model specified satisfies all of the usual assumptions, save for including a relatively large number of unnecessary predictors, the Dantzig Selector can find the predictors with regression coefficients equal to zero.

The Dantzig Selector has the following formulation.

$$\hat{\beta} = \min_{\beta} \sum_{j=1}^{p} |\beta_j| \text{ subject to } \sum_{i=1}^{n} |x_{ij} r_i| < \lambda \tag{2.10}$$

for $j = 1, 2, \ldots, p$, where the predictors have been standardized to z-scores, r_i is the usual regression residual, λ is a tuning parameter, and p is the number of predictors.

The Dantzig Selector, like the lasso, uses the sum of the absolute values of the regression coefficients as an argument. Minimizing the sum of the absolute values of the regression coefficients can produce regression coefficients that are exactly zero, and therefore, the associated predictors are removed from the analysis. But the key idea is that $\sum_{i=1}^{n} |x_{ij} r_i|$ captures any association between the residuals and each predictor in turn. When for each predictor $\sum_{i=1}^{n} |x_{ij} r_i| = 0$, one has the usual least squares solution in which by construction, the predictors are unrelated to the residuals. With $\sum_{i=1}^{n} |x_{ij} r_i| > 0$, bias is introduced because one or more predictors is associated with the residuals. By setting the value of λ, one can introduce varying degrees of association between each predictor and the residuals, and varying degrees of bias in the estimated regression parameters.

Work by Gareth and Radchenko (2007) extends applications of the Dantzig Selector to the entire generalized linear model. It may also be a useful tool when applied to functional linear regression (Gareth and Zhu, 2007). An important insight is that the Dantzig Selector can be formulated within a maximum likelihood framework such that the tuning parameter allows the partial

derivatives of the likelihood function with respect to the regression coefficients to be nonzero. Consequently, the solution is moved away from the maximum likelihood result. As before, some bias is introduced that can shrink the appropriate regression coefficients to zero.

To date, hands-on experience with the Dantzig Selector is very limited and it is not clear how the Dantzig Selector performs compared to obvious competitors such as the lasso (Efron et al., 2007; Meinshausen, 2007) In addition, potential insights into statistical learning have yet to be well explored (Cai and Lv, 2007). However, the ideas built into the Dantzig Selector are provocative, and it may have a bright future.

Regularization and Derivative Expectation Operator: Rodeo

Rodeo is perhaps the most recent entry into the shrinkage sweepstakes (Lafferty and Wasserman, 2008). It is related to adaptive smoothing, which is discussed later in this chapter, as a result, and to the lasso. The goal is to apply shrinkage to nonparametric regression, also discussed shortly, so that irrelevant predictors can be identified and removed. Rodeo assumes, as before, that one has in the data all of the correct regressors and some additional ones. The irrelevant predictors make the full set of predictors "sparse."

It is difficult to be very specific before nonparametric regression is more fully discussed, but the basic approach can be easily described. Suppose there is a single predictor. The degree of smoothness for the computed $f(X)$ is varied starting with a very smooth $f(X)$ and gradually making it more rough. If on the average the fitted values are much the same regardless of the degree of smoothing, that predictor is not meaningfully related to the response. Smooth, rough or in between, the $f(X)$ does not change significantly. Conversely, if on the average the fitted values vary substantially as the degree of smoothness changes, the predictor is meaningfully related to the response. The degree of smoothness matters for the $f(X)$.

Now imagine having p predictors. If changing the degree of smoothing has little impact on the average fitted value for a given predictor, one can conclude that that predictor is not relevant. If changing the degree of smoothing has a large impact on the average fitted value for a given predictor, one can conclude that that predictor is relevant.

As a practical matter, rodeo begins with a very smooth version of the $f(X)$. Gradually, the $f(X)$ is made less smooth for each predictor. For any predictor and given amount of smoothness, there is an aggregate derivative over observations representing how much the $f(X)$ changes with infinitesimal changes in the amount of smoothing. When for any predictor the derivative is smaller than some threshold for that predictor, the predictor is deleted from the model. Ideally, the irrelevant predictors are deleted first leaving behind the relevant predictors.

It is far too early to know how effective rodeo will be in practice. More important for now is its conceptual structure. All of the shrinkage proce-

dures considered thus far place constraints on regression coefficients, which as derivatives represent how the average response changes as a function of an infinitesimal change in predictor values. Rodeo places constraints on how much the average response changes with infinitesimal changes in the amount of smoothing. More generally, rodeo addresses shrinkage for nonparametric regression. This provides a useful transition to smoothing splines, which are addressed shortly.

2.3.2 Shrinkage and Statistical Inference

If the data used in a shrinkage procedure have been generated by random sampling or by a known stochastic process, statistical inference may be called for. As mentioned earlier, however, shrinkage estimates present special problems. Even if the regression model meets the requisite assumptions, shrinkage introduces bias by design. If the regression estimates are biased, conventional confidence intervals will not have their advertised coverage. For example, the 95% confidence interval for a particular regressioncoefficient will not contain the true value for that regression coefficient 95% of the time. The estimate is offset by some unknown amount so that the actual coverage will be less than 95%. Similar problems exist for hypothesis tests. The disparity between the null hypothesis and the sample statistic will be either too large or too small because of the offset caused by the bias. As a result, the computed p-values will be too large or too small as well.

Recall that the traditional goal of shrinkage is to construct sample estimates as close as possible to their population counterparts by the mean squared error criterion. The uncertainty estimates, therefore, risk confounding the variance with the bias. This can mean that a sensible confidence interval needs to take both into account if the usual coverage is to be represented. Likewise, sensible tests need to produce p-values that respond to both.

Because the nature of the bias is unknown, there is no easy fix. All one can know for sure is that the conventional procedures by which one constructs confidence intervals or performs hypothesis tests will be incorrect and that statistical inference reported by the regression software is likely to be incorrect as well.

In some settings, it can be prudent to reduce aspirations. One can focus on the variance alone. If the question solely is how much instability there is in the estimates of the regression coefficients or the fitted values, the bias is not longer formally relevant (Buja and Rolke, 2007). It follows that bootstrap samples of the observations (i.e., of Y and of X) can be used, much as with the simple percentile method, to construct useful intervals, which are in theory covering properly the values of the population parameters shifted up or down by the shrinkage. The target is no longer the "truth."

There also seems to be the prospect of useful alternative procedures based on Stein estimators and empirical Bayes methods (Carlin and Louis, 1996). The basic idea is that if one computes the conditional mean for a small region

defined by predictor values, that estimate will likely be reasonably unbiased with respect to the true conditional mean in that region. The difference between that value and the average of the shrunken fitted values in the region can provide a useful approximation of the direction and size of the bias. That approximation can then be combined with an estimate of the variance to construct improved confidence intervals and tests (Brown et al., 2005). To date, this approach has only been developed for single predictors, but extensions to multiple predictors seem possible.

In summary, statistical inference for shrinkage estimates is largely an unsolved problem. At the very least, there seems to be no consensus on how best to undertake statistical inference on shrinkage estimates when there is a need to consider the impact of the bias. Similar issues can arise for a number of the procedures considered in later chapters.

2.3.3 Shrinkage: So What?

When shrinkage is applied to conventional regression estimates there can be, as noted earlier, two goals. First, one might be interested in model selection. The lasso and the elastic net can provide useful alternatives to conventional model selection procedures, such as nested statistical tests, if their assumptions are approximately met. Shrinkage is used to select the regressors and then a conventional regression equation is estimated. However, problems discussed earlier about postmodel selection statistical inference remain, and there is never any guarantee, regardless of the method, that the model selected will make scientific or policy sense. There is no necessary correspondence between the statistical criteria and good science or good policy. The models that result should be seen as highly provisional.

Second, one might be interested in striking a good balance between the bias and the variance; the problem is not model selection in the usual sense. Then, whether ridge regression, the lasso, or the elastic net (or some other penalty formulation) should be strongly preferred is less clear. A lot depends on the properties of the data on hand (Zou and Hastie, 2005).

In short, shrinkage procedures at this point look to be primarily niche players in routine data analysis. They have some promise for model selection and for addressing the bias–variance tradeoff in conventional regression. The major reason why shrinkage has been discussed here is that imposing penalties on the fitting process to smooth the fitted values is more generally useful. In addition, the issues that shrinkage raises, and the concepts shrinkage introduces, play an important role in more advanced smoothers, and in procedures considered in later chapters. There are also some interesting applications that are beyond the scope of this book. For example, Zou, Hastie, and Tibshirani (2006) apply the elastic net to principal components analysis.

2.4 Smoothing Splines

For the spline-based procedures considered thus far, the number and location of knots had to be determined a priori or in the case of the number of knots, by some measure of fit. We now consider an alternative that does not require a priori knots. A key feature of this approach is to effectively saturate the predictor space with knots and then protect against overfitting by constraining the impact the knots can have on the fitted values. The influence that knots can have can be diluted; the initial number of knots does not have to change but the impact of some can be shrunk to zero. The key is a somewhat different kind of penalty for undue complexity.

We begin by requiring that for our single predictor and response variable, there is a function $f(X)$ with two derivatives over its entire surface. This is a common assumption in the statistical learning literature and in practice does not seem to be particularly restrictive. The goal is to minimize a "penalized" error sum of squares of the form

$$\text{RSS}(f, \lambda) = \sum_{i=1}^{N} [y_i - f(x_i)]^2 + \lambda \int [f''(t)]^2 dt, \tag{2.11}$$

where λ is, as before, a tuning parameter. The first term on the righ-hand side captures how close the fitted values are to the actual values of y. It is just the usual error sum of squares. The second imposes a cost for the complexity of the fit. The integral quantifies the roughness penalty, and λ determines the weight given to that penalty in the fitting process. At one extreme, as λ increases without limit, the fitted values approach the least squares line. Because no second derivatives are allowed, the fitted values are as smooth as they can be. At the other extreme, as λ decreases toward zero, the fitted values approach an interpolation of the values of the response variable.

Equation 2.11 addresses the bias–variance tradeoff head-on. When λ is larger, the fitted values are smoother, with the likely consequence of more bias and less variance. When λ is smaller, the fitted values are rougher with the likely consequence of less bias and more variance. Thus, the value of λ can be used in place of the number of knots to tune the bias–variance tradeoff.

For a given value of λ, Equation 2.11 can be minimized. Hastie et al. (2001: Section 5.4) explain that a unique solution results, based on a set of natural cubic splines with N knots. This assumes that there are N distinct values of x. There will be fewer knots if there are less than N distinct values of x.

It follows that

$$f(x) = \sum_{j=1}^{N} N_j(x)\theta_j, \tag{2.12}$$

where θ_j is a set of weights, $N_j(x)$ is an N-dimensional set of basis functions for the natural cubic splines being used, and j stands for the number of knots, of which there can be a maximum of N.

Consider the following toy example, in which x takes on values 0 to 1 in steps of .20. In this case, $j = 6$, and Equation 2.12, written as $f(x) = \mathbf{N}\theta$, takes the form of

$$f(x) = \begin{pmatrix} -.267 & 0 & 0 & -.214 & .652 & -.429 \\ .591 & .167 & 0 & -.061 & .182 & -.121 \\ .158 & .667 & .167 & -.006 & .019 & -.012 \\ 0 & .167 & 0.667 & .155 & .036 & -.024 \\ 0 & 0 & .167 & .596 & .214 & .024 \\ 0 & 0 & 0 & -.143 & .429 & .714 \end{pmatrix} \begin{pmatrix} \theta_1 \\ \theta_2 \\ \theta_2 \\ \theta_4 \\ \theta_5 \\ \theta_6 \end{pmatrix}. \tag{2.13}$$

Equation 2.11 can be rewritten using a natural cubic spline basis and then the solution becomes

$$\hat{\theta} = (\mathbf{N}^T\mathbf{N} + \lambda\boldsymbol{\Omega}_N)^{-1}\mathbf{N}^T\mathbf{y}, \tag{2.14}$$

with $[\boldsymbol{\Omega}_N]_{ij} = \int N_j''(t)N_k''(t)dt$, where the second derivatives are for the function that transforms x into its natural cubic spline basis. $[\boldsymbol{\Omega}_N]$ has larger values where the predictor is rougher, and given the linear estimator, this is where the fitted values are rougher as well. The penalty is the same as in Equation 2.11.

Equation 2.14 can be seen as a generalized form of ridge regression. With ridge regression, for instance, $[\boldsymbol{\Omega}_N]$ is an identity matrix. In practice, N is replaced by a basis of B-splines that is used to compute the natural cubic splines.

The requirement of N knots may seem odd because it appears to imply that N degrees of freedom are used up. However, for values of λ greater than zero, the resulting smoother is shrunk toward a linear fit. In other words, whenever the penalty for complexity comes into play, it makes the fitted values more smooth, and in so doing, reduces the number of degrees of freedom actually used up. Larger values of λ mean that fewer degrees of freedom are lost.

As with the number of knots, the value of λ can be determined a priori or through model selection procedures. One common approach is based on N-fold (drop-one) cross-validation, briefly discussed in the last chapter. The value of λ is chosen so that

$$\mathrm{CV}(\hat{f}_\lambda) = \sum_{i=1}^{N}[y_i - \hat{f}_i^{(-i)}(x_i)]^2 \tag{2.15}$$

is as small as possible. Recall that $\hat{f}_i^{(-i)}(x_i)$ is the fitted value with case i removed. Using the CV to select λ is one automated way to find a promising balance between the bias and the variance in the fitted values. However, all of the earlier caveats apply.

2.4.1 An Illustration

To help fix all these ideas, we turn to an application of smoothing splines. Figure 2.9 shows a smoothed scatterplot based on equations 2.11 and 2.15. The data come from seven Japanese cities from 1973 through 1999. The response variable is residential water use in 1000s of cubic feet. The predictor is population size. The standard thinking about water consumption is that it increases linearly with population.

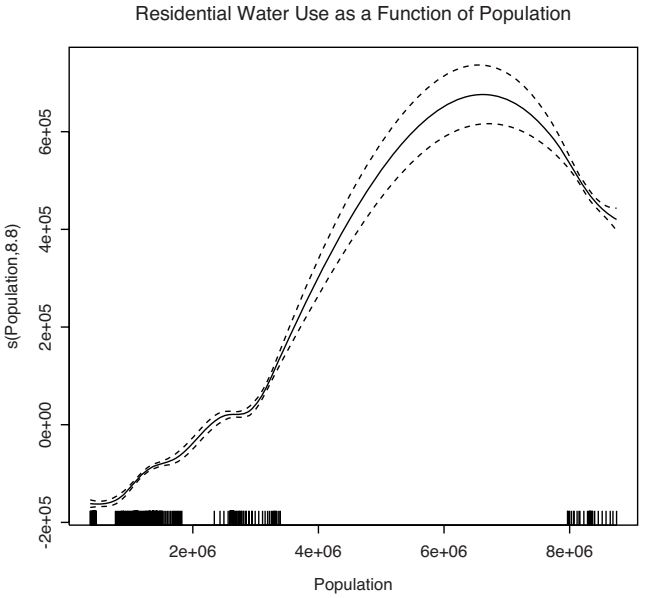


Fig. 2.9. An application of penalized regression splines.

In Figure 2.9, population is on the horizontal axis, and the fitted values are on the vertical axis. The smoother is represented by the solid line, and a point-by-point 95% confidence interval by the broken line, assuming that estimation is at least in principle justified. If $\mathbf{S}$ is the smoother matrix, the covariance of $\hat{f}(x) = \mathbf{S}\mathbf{S}^T\sigma^2$. With σ^2 estimated by the error sum of squares divided by $N - tr(\mathbf{S})$, the main diagonal of $\text{cov}[\hat{f}(x)]$ contains point-by-point estimates of the error variance. Then, with Gaussian errors and negligible bias, plus or minus twice the square root of the variances can be viewed as a point-by-point 95% confidence interval. (Hastie and Tibshirani, 1990: Section 3.8). We consider statistical inference for smoothers in more depth shortly.

The rug plot at the bottom of the plot shows where the population data tend to be located. One implication is that there are no data over the range where the curve starts to bend downward. As one would expect, the confidence

interval widens substantially around the large bend in the fitted values because there are very little data providing support. Although this makes good sense, we consider below why it would be risky to treat the band plotted in Figure 2.9 as a 95% confidence interval.

Figure 2.9 shows a positive relationship for the smaller population centers that is approximately linear, and a negative relationship for larger population centers that is also approximately linear. The latter results from factors in the biggest cities, such as affluence and the use of water-efficient technology, that are not considered when population is the sole predictor (Berk and Rothenberg, 2004).

Figure 2.9 was constructed with the gam() procedure in the *mgcv* library. The symbol "s" in the label on the vertical axis means that a smoother has been applied. In this case, the smoother is based on penalized regression splines of the sort just discussed with the value of λ determined by the GCV statistic. The "8.8" in the label is the effective degrees of freedom (or the equivalent number of parameters) used up by the smoother. Clearly, 8.8 is a lot smaller than the number of observations, but some distance from 1.0. The result is a rather smooth function that is substantially nonlinear. One degree of freedom would have been used up had a linear smooth materialized. With a greater effective degrees of freedom, the fitted values are less smooth.

2.5 Locally Weighted Regression as a Smoother

2.5.1 Nearest Neighbor Methods

Thus far, the discussion of smoothing has been built upon a foundation of conventional linear regression. Another approach to smoothing is from the perspective of nearest neighbor methods. Consider Figure 2.10 in which the shaded ellipse represents a scatter of points for values for x and y.

There is a target value of x, labeled x_0, for which a conditional mean $\bar{y}_0$ is to be computed. There may be only one such value of x or a relatively small number of such values. As a result, a conditional mean computed from those values alone risks being very unstable. One possible solution is to compute $\bar{y}_0$ from observations with values of x close to x_0. The rectangle overlaid on the scatterplot illustrates a region of "nearest neighbors" that might be used. Insofar as the conditional means for x are not changing systematically within that region, a useful value for $\bar{y}_0$ can be obtained. If that conditional mean is to be used as an estimate, it will be unbiased and likely be more stable than the conditional mean estimated only for the observations with $x = x_0$. In practice, however, some bias is often introduced. As before, the hope is that the increase in the bias is small compared to the decrease in the variance.

A key issue is how the nearest neighbors are defined. One option is to take the k closest observations using the metric of x. For example, if x is age, x_0 is 24 years old, and k is 10, the ten closest x-values may range from 23

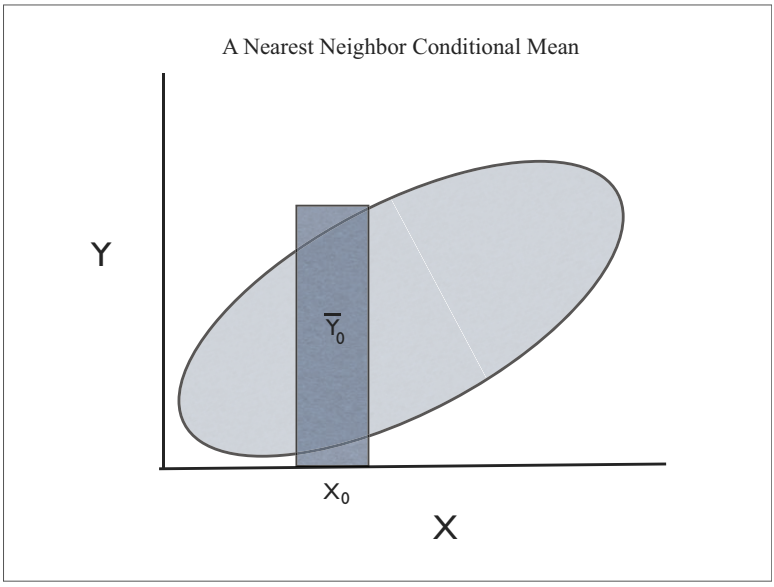


Fig. 2.10. A conditional mean for a target value of X.

to 27 years of age. Another option is take the some fixed fraction f of the observations that are closest to x_0. For example, if the closest 25% of the observations were taken, k might turn out to be 31, and the age-values might range between 21 and 29. Yet another option is to vary either k or f depending on the variability in y within a neighborhood. If there is more heterogeneity that is likely to be noise, larger values of k or f can be desirable to improve stability. Note that for any of these approaches, the neighborhoods will likely overlap. For another target value near x_0, some near neighbors will likely be in both neighborhoods. There also is no requirement that the neighborhood be symmetric around x_0.

Suppose now that for each unique value of x a nearest neighbor conditional mean for y is computed using one of the approaches just summarized. Figure 2.11 shows a set of such means connected by straight lines. The pattern provides a visualization of how the means of y vary with x. As such, the nearest neighbor methods can been seen as a smoother.

Figure 2.11 will change depending on the size of the neighborhood. Larger neighborhoods will tend to make the smoothed values less variable. If the smoothed values are to be treated as estimates, they will likely be more biased and more stable. Smaller neighborhoods will tend to make the smoothed values more variable. If the smoothed values are to be treated as estimates, they will likely be less biased and less stable.

Nearest neighbor methods can be very effective in practice and have been elaborated in many ways. There can be more than one predictor, for example,

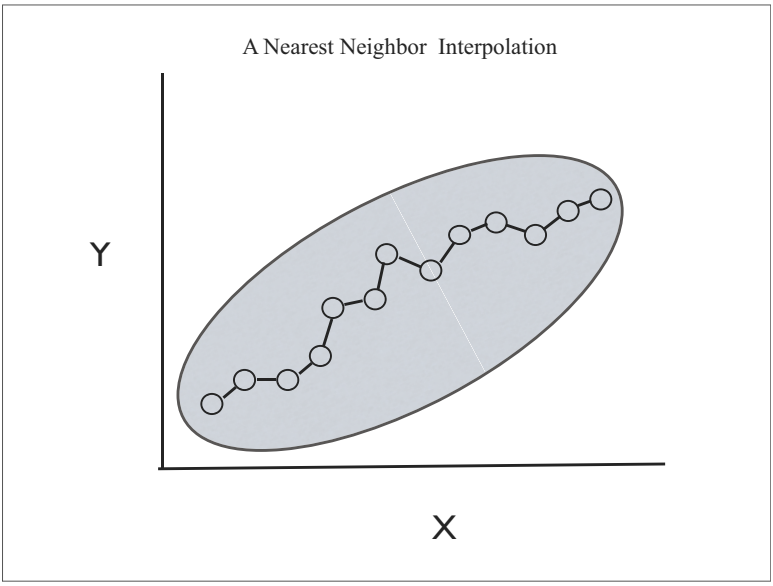


Fig. 2.11. Interpolated conditional means.

which raises some difficult issues about how to best define the neighborhood (e.g., Hastie and Tibshirani, 1996). This is a matter to which we return.

For our purposes, perhaps the major weakness of nearest neighbor methods is they are not derived as a way to represent how Y is related to X; they are not explicitly linked to some $f(X)$. One consequence is that when there are more than two predictors, there is little guidance on how to represent the manner in which the predictors are related to the response.

Nevertheless, nearest neighbor methods introduce some very important issues and procedures that figure significantly in this and later chapters. Indeed, the line between nearest neighbor methods and a number of other techniques can be pretty fuzzy. Readers interested in learning more about nearest neighbor methods should consult Ripley (1996) and Shakhnarovich (2006).

What if within each neighborhood the conditional means of y vary systematically? At the very least, there is information being ignored that could improve the estimate of $\bar{y}_0$. Just as in conventional linear regression, if y is related to x in a systematic fashion, there can be less variation in the regression residuals than around the neighborhood mean of y. More stable estimates can follow. The idea of applying linear regression within each neighborhood leads directly to a smoothing procedure known as locally weighted regression.

2.5.2 Locally Weighted Regression

Although spline smoothers are widely used, lowess (Cleveland, 1979) is a useful alternative that was developed before penalized regression smoothers. It is

comparatively easy to understand and remains a very handy tool. Lowess also has a more "algorithmic" feel than penalized regression smoothers and is, therefore, a useful didactic device for the material that follows. Lowess stands for "Locally Weighted Scatterplot Smoothing," although there seem to be a number of translations of "lowess."

We stick with the one predictor case a bit longer. For any given value of the predictor x_0, a polynomial regression is constructed only from observations with x-values that are nearest neighbors of x_0. Among these, observations with x-values closer to x_0 are weighted more heavily. Then, $\hat{y}_0$ is computed from the fitted regression and used as the smoothed value of the response y at x_0. The process is repeated for all other values of x. Although the polynomial is often of degree one (linear), quadratic and cubic polynomials are also used. It is not clear that much is gained in practice using the quadratic or cubic form. In some implementations, one can also employ a degree zero polynomial, in which case no regression is computed, and the conditional mean of y in the neighborhood is used as $\hat{y}_0$. This is the nearest neighbor approach discussed above except for the use of distance weighting.

The precise weight given to each observation depends on the weighting function employed. The normal distribution is one option. That is, the weights form a bell-shaped curve centered on x_0 that declines with distance from x_0. The tricube is another option. Differences between x_0 and each value of x in the window are divided by the length of the window along x. This standardizes the differences. Then the differences are transformed as $(1 - |z|^3)^3$, where z is the standardized difference. Values of x outside the window are given weights of 0.0. As an empirical matter, most of the common weighting functions give about the same results.

As discussed for nearest neighbor methods, the amount of smoothing depends on the proportion of the total number of observations used when each local regression line is constructed. Proportions between .25 and .75 are common. The proportion has been given various names in the smoothing literature; "window" or "span" or "bandwidth" are all used. The larger the proportion of observations included, the smoother are the fitted values. The bandwidth plays the same role as the number of knots in regression splines or λ in smoothing splines. Some software also permits the bandwidth to be chosen in the units of the regressor. For example, if the predictor is population size, the span might be defined as 10,000 people wide.

More formally, each local regression at each x_0 is constructed by minimizing the weighted sum of squares with respect to the intercept and slope for the $M \leq N$ observations included in the window. Thus,

$$\text{RSS}^*(\boldsymbol{\beta}) = (\mathbf{y}^* - \mathbf{X}^*\boldsymbol{\beta})^T \mathbf{W}^* (\mathbf{y}^* - \mathbf{X}^*\boldsymbol{\beta}). \tag{2.16}$$

The asterisk indicates that only the observations in the window are included. The regressor matrix $\mathbf{X}^*$ can contain polynomial terms for the predictor should that be desired. $\mathbf{W}^*$ is a diagonal matrix conforming to $\mathbf{X}^*$, with

diagonal elements w_i^*, which are a function of distance from x_0. This is where the weighting-by-distance gets done. The algorithm then operates as follows.

1. Choose the smoothing parameter such as bandwidth, f, which is a proportion between 0 and 1.
2. Choose a point x_0 and from that the $(f \times N = M)$ nearest points on x.
3. For these M nearest neighbor points, compute a weighted least squares regression line for y on x.
4. Construct the fitted value $\hat{y}_0$ for that single x_0.
5. Repeat Steps 2 through 4 for each value of x. Near the boundary values of x, constraints are sometimes imposed much like those imposed on cubic splines and for the same reasons.
6. Connect these $\hat{y}$s with a line.

There is also a robust version of lowess. After the entire fitting process is completed, residuals are computed in the usual way. Weights are constructed from these residuals. Larger residuals are given smaller weights and smaller residuals larger weights. Using these weights, the fitting process is repeated. This, in turn, can be iterated until the fitted values do not change much (Cleveland, 1979) or until some predetermined number of iterations is reached (e.g., three). The basic idea is to make observations with very large residuals less important in the fitting.

Whether the "robustification" of lowess is useful will be application-specific and depend heavily on the window size chosen. Larger windows will tend to smooth the impact of outlier residuals. Equally important, because the scatterplot being smoothed is easily plotted and examined, it is usually easy to spot the possible impact of outlier residuals and if necessary, take them into account when the results are reported. In short, there is no automatic need for the robust version of lowess when there seem to be a few values of the response that perhaps distort the fit.

An Illustration

Figure 2.12 shows for a set of Japanese cities over 21 years a (nonrobust) lowess smooth of residential water consumption on the average price of water. Economic theory says the slope should be negative, other things being equal. It is difficult to make much of Figure 2.12. The window is set at .10 (10% of the data) so the fitted values are highly variable.

In Figure 2.13, the span is increased to .50 (50% of the data). Clearly, the result is a much smoother fit. In Figure 2.14, the span is still .50, but the fitting is based on an M-estimator (to "robustify" the fitted values), not conventional least squares. The change of the fitting function makes little difference in this example, and that seems to be a common outcome.

Figure 2.15 uses a span of .90 (90% of the data) and returns to the Gaussian weighting function. Clearly, this produces by far the smoothest fit. But which

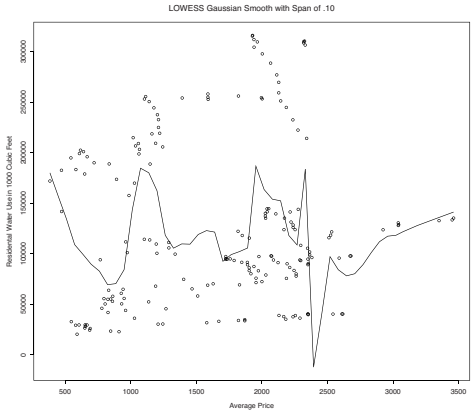


Fig. 2.12. Lowess Gaussian smooth of water consumption on average price: span = .10.

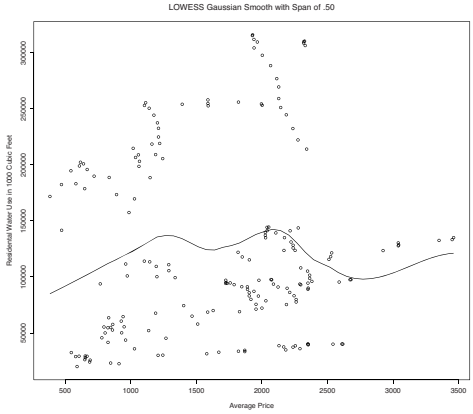


Fig. 2.13. Lowess Gaussian smooth of water consumption on average price: span = .50.

fit is best? The answer depends heavily on subject matter knowledge. In this case, one would anticipate a rather smooth, monotonically declining curve. All of the fitted values but the final set seem unduly variable and inconsistent with the way consumers should respond to price. Figure 2.15 is, therefore, probably the most informative.

However, the smoothed values are quite flat, with a slight upward trend followed by a slight downward trend. When water is relatively cheap, higher prices lead to more water consumption. When water is relatively expensive, higher prices lead to less water consumption. It is difficult to think of an explanation consistent with economic theory and more likely the positive seg-

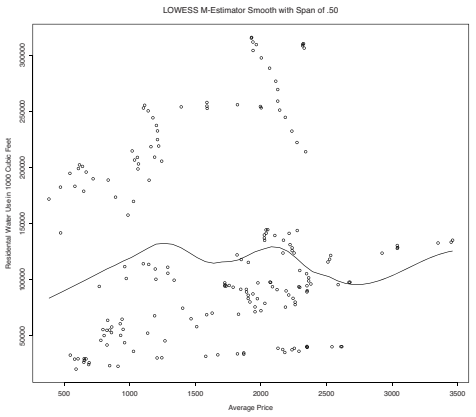


Fig. 2.14. Lowess M-estimator smooth of water consumption on average price: span = .50.

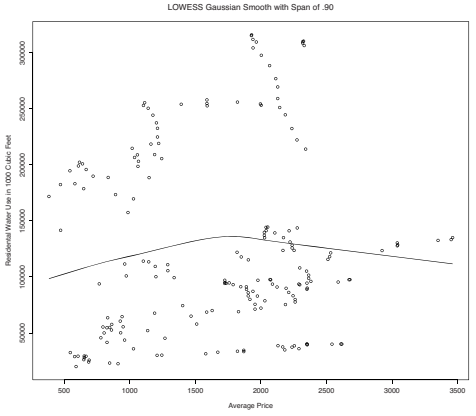


Fig. 2.15. Lowess Gaussian smooth of water consumption on average price: span = .90.

ment (at least) of the curve is an artifact caused by omitted predictors such as income. (For further discussion see Berk and Rothman, 2004.)

It may be important to underscore that even though the smoothed values in Figures 2.12 through 2.15 do not represent causal models, any interpretations resting on cause-and-effect claims need to consider many of the same issues that arise in conventional causal modeling. Omitted variables are surely one key concern. If the goal is description alone, then it is not even clear what an "omitted variable" is. The statistical definition requires that for a potential predictor to be an "omitted variable," it must be correlated with the response variable and any predictors already included in the analysis. But it is difficult to attach much import to the word "omitted" except in a causal context.

Perhaps the strongest statement that could be made is that the description is not complete.

In any case, the main message here is not meant to be substantive. The main message is that the bandwidth specified for lowess can make a big difference. The choice of bandwidth really matters. Because of this, there have been, in much the same spirit as the choice of λ in penalized regression, many attempts to automate and rationalize the selection of bandwidth size. For example, the generalized cross-validation statistic can be used to select the bandwidth (Loader, 2004: Section 4).

Such procedures can work well as a place to start. But once again, automation takes no notice of subject matter knowledge, and more useful visualizations are often produced when the choice of bandwidth is informed, at least in part, by information brought to the analysis from outside the data. It is doubtful that an automated procedure would have selected Figure 2.15. More likely, something close to Figure 2.13 would have been chosen. There is also the risk of overfitting, especially if a large number of bandwidths is tried.

2.6 Smoothers for Multiple Predictors

The last set of figures is only the most recent example in which the limitations of a single predictor were apparent. Many more things could be related to water consumption than price alone. The time has come to consider smoothers when there is more than one predictor.

In principle, it is a simple matter to include many predictors and then smooth a multidimensional space. However, there are three significant complications in practice. The first problem is the "curse of dimensionality." As the number of predictors increases, the space the data need to populate increases as a power function. Consequently, the demand for data increases very rapidly, and one risks data that are far too sparse to produce a meaningful fit. There are too few observations, or those observations are not spread around sufficiently to provide the support needed. One must, in effect, extrapolate into regions where there is little or no information. To be sensible, such extrapolations would depend on knowing the $f(X)$ quite well. But it is precisely because the $f(X)$ is unknown that smoothing is undertaken to begin with.

The second problem is that there are often conceptual complications associated with multiple predictors. In the case of lowess, for example, how is the neighborhood near x_0 to be defined (Fan and Gijbels, 1996: 299-300)? One option is to use Euclidian distance. But then the neighborhood will depend on the units in which predictors happen to be measured. The common practice of transforming the variables into standard deviation units does not really seem to solve the problem, especially when coupled with the need to weight observations by proximity to x_0.

Consider the case of two predictors. Suppose the standard deviation for one predictor is five years of age, and the standard deviation for the other predictor is two years of education. Now suppose one observation falls at x_0's value of education, but is five years of age higher than x_0. Suppose another observation falls at x_0's value for age, but is two years higher in eduction than x_0. Both are one standard deviation unit away from x_0 in Euclidian distance. But do we really want to say they are equally close?

Another approach to neighborhood definition is to use the same span for both predictors, but apply it separately in each direction. Why this is a better definition of a neighborhood is not clear. And one must still define a distance metric by which the observation in the neighborhood will be weighted.

Yet another alternative is to define a neighborhood by the importance of each dimension of the predictor space or a transformed predictor space. Where in that space the response is changing more rapidly, the neighborhood should be smaller. That way, significant variation in the fitted values is not smoothed away. We show later in this chapter that locally adaptive smoothers take a related approach. We learn that significant computation problems can follow.

The third problem is that gaining meaningful access to the results is no longer straightforward. When there are more than two predictors, one can no longer graph the fitted surface in the usual way. How does one make sense of a surface in more than three dimensions?

2.6.1 Smoothing in Two Dimensions

With only two predictors, there are some fairly straightforward extensions of conventional smoothers that can be instructive, even in the face of the three problems just discussed. For example, with smoothing splines, the penalized sum of squares in Equation 2.11 can be generalized. The solution is a set of "thin plate splines," and the results can be plotted. Thin plate splines are a two-dimensional generalization of the one-dimension cubic splines considered earlier. More specifically, Equation 2.11 can be generalized as

$$\min_f \sum_{i=1}^{N} \{y_i - f(x_i)\}^2 + \lambda J[f], \tag{2.17}$$

where J is an appropriate penalty functional of f. For the two-dimensional case,

$$J[f] = \int\int_{\Re^2} \left[\left(\frac{\partial^2 f(x)}{\partial x_1^2} \right)^2 + 2 \left(\frac{\partial^2 f(x)}{\partial x_1 x_2} \right)^2 + \left(\frac{\partial^2 f(x)}{\partial x_2^2} \right)^2 \right] dx_1 dx_2. \tag{2.18}$$

Equation 2.18 captures the roughness of the fitted values in a two-dimensional predictor space. The fitted values are rougher when the two second derivatives are larger. As before, the weight of this penalty is determined by the value of λ.

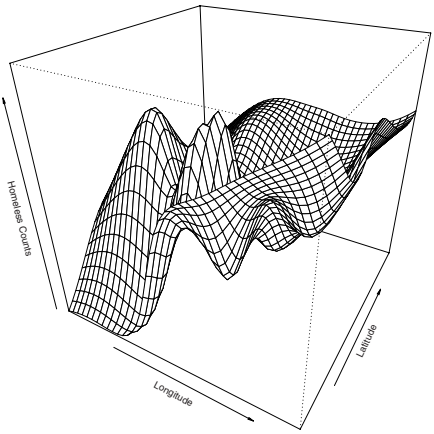


Fig. 2.16. Perspective plot of smoothed values of homelessness constructed from smoothing splines.

An Illustration

Figure 2.16 shows for a particular urban area a two predictor smooth of the homeless counts for census tracts by longitude and latitude. The value of λ was determined by the generalized cross-validation statistic. One can see that homelessness varies substantially by census tract. For example, the peak toward the middle of the plot is the downtown skid row area. The immediately surrounding areas have relatively low numbers of homeless individuals.

Figure 2.17 repeats the analysis with a two-predictor lowess smoother. The extension of lowess from one predictor to two proceeds as one would expect. A neighborhood and the within-neighborhood weighting are defined by Euclidian distance. Each neighborhood is now a solid rather than a plane so the local regression has two predictors rather than one. In this application, both predictors are in the same units, which makes the use of Euclidian distance far less controversial.

There is again a concentration of homeless in the skid row area, but now the spike is far more pronounced. It is difficult to determine precisely why the two plots differ. One possible explanation involves the manner in which the degree of smoothing is determined. For Figure 2.16, the value of λ was computed as part of the fitting algorithm. For Figure 2.17, the size of the span was determined in part by subject matter knowledge that made some results more credible than others. It is hard to know if the amount of smoothing in

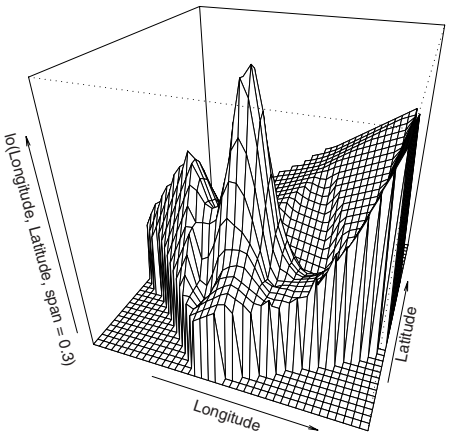


Fig. 2.17. Perspective plot of smoothed values of homelessness constructed from lowess.

the two figures is the same, but plots with somewhat different span values never eliminated the clear difference in the skid row effect.

The two smoothing procedures also differ substantially in their internal machinery. Smoothing splines, as do natural cubic splines, place a premium on fitted values that possess great continuity. Lowess does not build in such continuity so that sharp changes in direction can appear, especially when the span is small. Therefore, it would not be surprising to find that Figure 2.17 has a more jagged appearance. In short, a likely reason for the lower peak in homelessness for skid row in Figure 2.16 is that the sharp spike is rounded off.

Is there a way to choose between Figure 2.16 and Figure 2.17? The spatial patterns of homelessness show sharp differences over a distance of just a few blocks. Skid row, for example, is only two blocks from a cluster of modern, high rise office buildings where the number of homeless on the streets is very low. As a result, it is misleading to round off most of the transitions between areas with many homeless individuals and areas with few. Reality is closer to a two-dimensional step function. On these grounds, Figure 2.17 is probably a more accurate (if less elegant) visualization.

With more than two predictors, one generally needs another strategy. The data are often too sparse, and visualization is a major obstacle. The generalized additive model is one popular approach that meshes well with an

emphasis on regression and the use of a linear combination of basis functions.

2.6.2 The Generalized Additive Model

The Generalized Additive Model (GAM) is superficially an easy extension of the Generalized Linear Model (GLM). GAM tries to defeat the curse of dimensionality by assuming that the conditional mean of the response is a linear combination of functions of each predictor. Thus, the mean function for the generalized additive model with p predictors can be written as

$$\bar{Y}|X = \alpha + \sum_{j=1}^{p} f_j(X_j), \tag{2.19}$$

where α is a conventional intercept term.

In the same manner as the generalized linear model, the generalized additive model permits several different "link functions" and disturbance distributions. For example, with a binary response, the link function can be the log of the odds (the "logit") of the response, and the disturbance distribution can be logistic. This is analogous to logistic regression within the generalized linear model. But, there are no regression coefficients associated with the predictors. Regression coefficients would just scale up or scale down the functions of predictors, and so they are unnecessary. Whatever impact they would have is absorbed in the function itself. The role of the regression coefficients cannot be distinguished from the role of the transformation and therefore, the regression coefficients are not identified.

Each predictor can have its own functional relationship to the response. These functions can be estimated using single-predictor smoothers of the sort addressed above. Hence, the term nonparametric is usually applied despite the a priori commitment to an additive formulation. Alternatively, all of the functions may be specified in advance with the usual linear model as a special case. All of the common regression options are available, including the wide range of transformations one sees in practice: logs, polynomials, roots, product variables (for interaction effects), and indicator variables. As a result, GAM can be parametric as well and in this form is really no different from the generalized linear model. The parametric and nonparametric forms can be mixed so that some of the functions are derived empirically from the data, and some are specified in advance. Then the model is often called semiparametric.

With the additive form, one can use for GAM the same conception of "holding constant" that applies to conventional linear regression. The relationship between a given predictor and the response is constructed with (1) the linear dependence between the response and all other predictors removed, and (2) with the linear dependence between the given predictor and all other predictors removed. It is important to recall that the linear dependence removed is between the variables in whatever their transformed states happen to

be. Thus, there is no requirement of linear relationships between the variables in their original units.

More formally, consider for now the case of predictors x and z. Let

$$(\bar{y}|x, z) = \alpha + f_1(x) + f_2(z). \tag{2.20}$$

For now, assume that the $f_1(x)$ and $f_2(z)$ have been determined. For notational convenience $f_1(x)$ is denoted by x^*, and $f_2(z)$ is denoted by z^*. We focus first on $f_1(x)$.

Suppose that we estimate the regression parameters (i.e., intercepts and slopes) of the following two equations,

$$(\bar{y}|z^*) = \beta_0 + \alpha_1 z^*, \tag{2.21}$$

$$(\bar{x}^*|z^*) = \gamma_0 + \gamma_1 z^*. \tag{2.22}$$

For each, we compute the residuals $e_{y|z^*}$ and $e_{x^*|z^*}$. Finally, we estimate δ and $f_3(e_{x^*|z^*})$ in

$$(\bar{e}_{y|z^*}|e_{x^*|z^*}) = \delta + f_3(e_{x^*|z^*}). \tag{2.23}$$

The function $f_3(e_{x^*|z^*})$ should be identical to the function $f_1(x)$. A similar logic applies to $f_2(z)$. In other words, with the two functions determined, the usual covariance adjustments apply. "Holding constant" means to residualize precisely as Equations 2.20 through 2.23 specify. This is exactly the same logic that lies beneath added variable plots, sometimes called "partial plots" (Cook and Weisberg, 1999: Section 10.5).

But what if the transformations of all of the predictors are not known in advance? What if at least one of the functions (and usually several) is to be constructed empirically from the data? How does one estimate the function when the function needs to take the covariance adjustments into account? And one cannot apply the covariance adjustments unless the functions are known. To solve this problem, we turn to a computational algorithm called "backfitting."

A GAM Fitting Algorithm

The backfitting algorithm is a common way to estimate the functions and coefficients in Equation 2.19 (Hastie and Tibshirani, 1990: Section 4.4). The basic idea is to cycle through one function at a time in the following manner.

1. Initialize with $\alpha = \bar{y}, f_j = f_j^0, j = 1, \ldots, p$. Each predictor is given an initial functional relationship to the response such as a linear one. The intercept is given an initial value of the mean of y.
2. Repeat for $j = 1, \ldots, p, 1, \ldots, p, \ldots,$

$$f_k = S_j(y - \alpha - \sum_{j \neq k} f_j(x_k)). \tag{2.24}$$

A single predictor j is selected. Fitted values are constructed using all of the other predictors. These fitted values are subtracted from the response. A smoother S_j is applied to the resulting "residuals," taken to be a function of the single excluded predictor. The smoother updates the function for that predictor. Each of the other predictors is, in turn, subjected to the same process.

3. Continue Step 2 until the individual functions do not change.

Within any backfitting algorithm, a wide variety of smoothers can be applied and in the past have been. For example, both lowess and penalized regression splines have been available in R. Some procedures also permit the use of functions of two predictors at a time, so that the smoothed values represent a surface rather than a line, just as in Figures 2.16 and 2.17. That is, one can work with a linear combination of bivariate smoothed values.

In recent work (Wood, 2000, 2003, 2004), a somewhat different algorithm has been developed. The basic idea is to represent each of the functions to be determined empirically by a set of B-splines so that there is a single matrix of regressors for all of the unknown functions. This can then be combined with a regressor matrix for any terms whose functions are taken to be known a priori. The result is a multivariate generalization of penalized regression splines considered earlier when Equation 2.11 was discussed. Claims have been made that this approach has several advantages including more stable estimates, direct links to penalized fitting, and more straightforward extensions to conventional statistical inference. Whether any of these advantages would make much difference in practice is still to be determined.

The procedure gam() in R from the *mgcv* library is now implemented using this new algorithm. There are two GAM procedures in R, both called gam(). GAM using the traditional backfitting algorithm can be found in the R library *gam*.

An Illustration

We return now to the data used earlier on the possible deterrence impact of the death penalty. Recall that the data are a pooled cross-section time series of 50 states over 20 years. As before, the homicide rate is the response and the number of executions lagged by one year is a predictor. To control for the average differences between states, the homicide rate in each state, just before the beginning of each time series (1977), is used as a control variable. The multivariate penalized regression smoother just described is employed with the size of the penalty for each (λ) determined by the generalized cross-validation statistic.

The fit is excellent. About 90% of the variance is accounted for. Nearly all of this can be attributed to the values of the homicide rate when used as a control. Figure 2.18 shows the fitted values as a function of each predictor. If one ignores the very few values for the homicide rate that represent a very

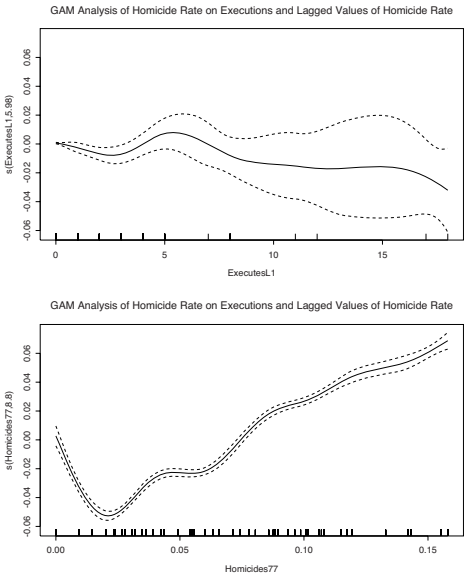


Fig. 2.18. GAM homicide results for executions controlling for the homicide rate in 1977.

few states over a very few years, the homicide rate over time is strongly and positively related to the homicide rate just before the beginning of the time series (i.e., Homicides77). Whatever the social processes in states that caused variation in the homicide rate in 1977, those same processes appear to persist over the next 20 years. The negative slope at the far left of the curve is much more difficult to understand and would need to be examined further. The likely explanation is the role of one or more predictors not included in the analysis.

The relationship between the number of executions lagged by one year (i.e., ExecutesL1) and the homicide rate is not strong overall. When there are five or fewer executions, which reflects 99% of the data, the relationship starts out being slightly negative and then turns more strongly positive. Net, the relationship is positive: more executions, more homicides a year later. The relationship when there are more than five executions, which reflects 1% of the data, is moderately negative. However, in part because there are so few observations, the nominal 95% point-by-point confidence interval (more on that shortly) is very wide and encloses a region that would easily allow for a flat or even positive slope. In short, there is no evidence whatsoever for deterrence for most of the states in most of the years, and evidence in favor of deterrence for the few outliers is not much stronger. (For more details see Berk, 2005a.)

One might wonder why there was no discussion of any regression coefficients. Recall that there are none. The fitted values for each predictor capture the average change in the response variable Y for small changes in a predictor. Because the fitted values have a nonlinear relationship with the response, there is not a single slope. In a sense, the graphs are the "slope." Or more formally, the derivative at any value of a predictor is the slope at that point.

Finally, both graphs in Figure 2.18 plot the fitted values centered on zero. This follows from the residualizing process described earlier. Recall that when there is an intercept, residuals have a mean of zero. Note also that the vertical axes have the same scales. This facilitates making comparisons between the response functions for different predictors.

2.7 Smoothers with Categorical Variables

As discussed in Chapter 1, smoothers can be used with categorical variables. When a predictor is categorical, however, there is really nothing to smooth. A binary predictor can take on only two values. The smoother is then just a straight line connecting the two conditional means of the response. For a predictor with more than two categories, there is no way to order the categories along the predictor axis. Any imposed order would imply assigning numbers to the categories. How the numbers were assigned could make an enormous difference in the resulting fitting values, and these assigned numbers would necessarily be arbitrary.

When the response is categorical and binary, smoothing can be a very useful procedure. All of the earlier benefits apply. In addition, because it is very difficult to see much in a scatterplot with a categorical response, a smoother may be the only way to gain some visual leverage on what may be going on.

2.7.1 An Illustration

We return now to the admissions data from a large public university. We apply GAM with admitted or not as the response. The predictors for each applicant are (1) mathematics SAT score, (2) verbal SAT score, (3) grade point average in high school, and (4) self-identified ethnic background. Figures 2.19 through 2.21 show the plots for the first three predictors. The residualized data are also shown. In each case, the vertical axis is in logits, just as it would be with logistic regression.

Figure 2.19 shows that beginning with SAT math scores of about 600 or higher, SAT math scores are positively, but modestly, related to the log-odds of admission. For lower math scores, there seems to be no relationship with the log-odds of admission.

Figure 2.20 shows that beginning with SAT verbal scores of about 450, SAT verbal scores are positively, but modestly, related to the log-odds of admission.

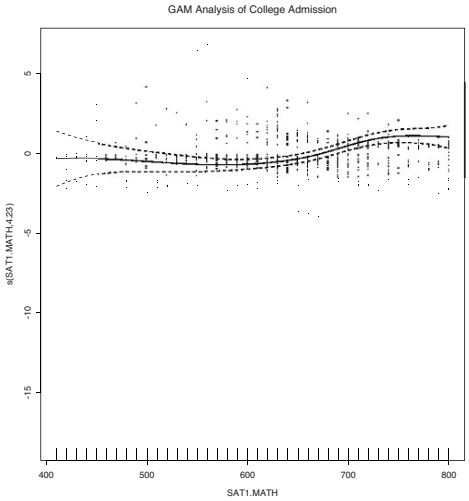


Fig. 2.19. Admission as a function of SAT math score.

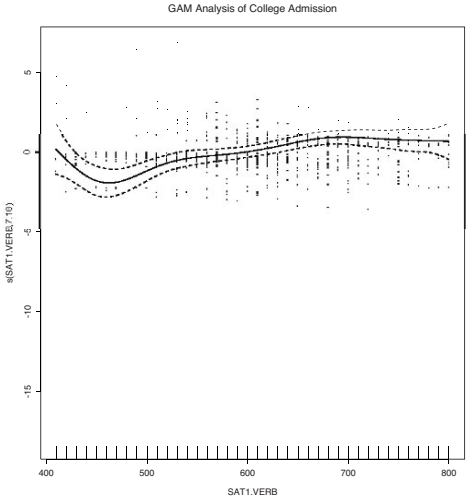


Fig. 2.20. Admisson as a function of SAT verbal score.

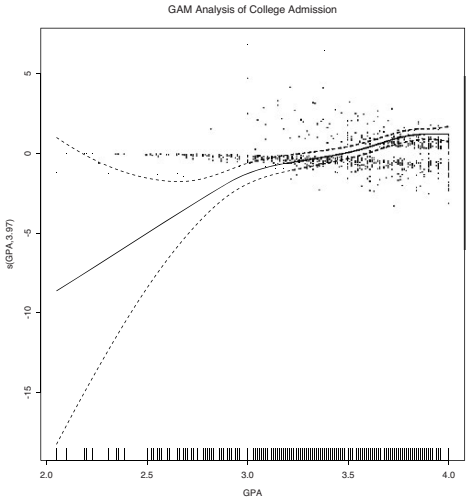


Fig. 2.21. Admission as a function of high school GPA.

The relationship seems to flatten out around a score of about 700 and even turn slightly negative. The negative relationship for SAT verbal scores below 450 is difficult to understand, but the data in that region are very sparse.

Figure 2.21 shows that over its full range, high school GPA is positively and strongly related to the log-odds of admission. It seems that the admissions process is weighting high school GPA far more heavily than SAT scores.

There are no plots for the categorical variable ethnicity. But for GAM, conventional regression coefficients are provided for all predictors whose functional forms are determined a priori. And that includes categorical variables. In this case, the regression coefficients reveal that holding constant SAT scores and high school GPA, the odds that an Anglo or Asian student will be admitted are substantially lower than for Hispanic and African-American students.

In short, there is certainly no lockstep relationship between earlier academic performance, as measured by SAT scores and high school GPA, and admission. Other factors are taken into account. This means that the apparent impact of ethnicity needs to be unpacked. Are there other predictors that would eliminate ethnicity as a useful explanatory variable? And if not, one cannot know without further study how an applicant's ethnicity comes into play. Is it directly used in the admission decision and/or is the impact really explained by characteristics of the applicant that are associated with ethnicity but not part of the official record?

As noted earlier, if the point is to explain why response functions come out as they do, causal thinking is often unavoidable. But there is nothing in any of the results that conveys what would happen if, for example, a given applicant's reported SAT scores were altered. To learn that, one would have to

actually alter the scores used in the admissions process. Such an experiment could perhaps be done if humansubject concerns could be overcome. What the analysis indicates thus far is that SAT scores, high school GPA, and ethnicity would probably need to be among the factors manipulated. And there would surely be others. The analysis also implies that if the purpose was to forecast admissions, SAT scores, high school GPA, and ethnicity might well provide considerable forecasting skill.

The statistical message is much the same as before. Allowing the data to determine how predictors are related to the response can be very instructive, even when (or perhaps especially when) the response variable is categorical. A natural question, however, is how restrictive the GAM's additive form is in practice. Experience to date suggests that the additive restriction is often not a serious obstacle. For instance, if there is an interest in interaction effects, these can be represented by a two-dimensional smoother (for two-way interactions) or by including product variables. This comes up often with spatial data, for example, where location is measured by variables such as longitude and latitude. If the response surface is substantially torqued, the additive terms are insufficient. One needs either a two-dimensional smoother or a product of the two spatial dimensions as another term in the model.

2.8 Locally Adaptive Smoothers

Under some circumstances, regression splines and regression smoothers can stumble when relationships with the response have sharp inflection points or steps. If all of the sharp inflection points or steps are about the same size, smoothing parameters can be set to either remove them all or to show them all. But when they are substantially different sizes, the risk is that some will be removed and some will not, even if all are equally informative.

One potential solution is to allow the smoothing parameters to vary locally so that they can adapt to sharp changes in the response function. For example, the bandwidth can be made smaller where the mean function seems to be changing most rapidly. This is very hard to do by hand without an impractical amount of trial and error. But there are a number of "locally adaptive" procedures that can automate the process. (Fan and Gijbels, 2006; Loader, 1999).

Figure 2.22 shows an example based on simulated data taken from Loader's instructive book (Loader, 1999). The simulated data have several telling features. First, the signal-to-noise ratio is very high. Second, the apparent pattern is far more variable in some regions than others. Third, the number of observations is relatively large. Finally, the mean function is extremely nonlinear. Taken together, these four features make it relatively easy to see what the mean function looks like without the aid of any smoother whatsoever.

The adaptive smoother that is overlaid clearly performs very well. A smoother with a single smoothing parameter would likely wash out the cycles

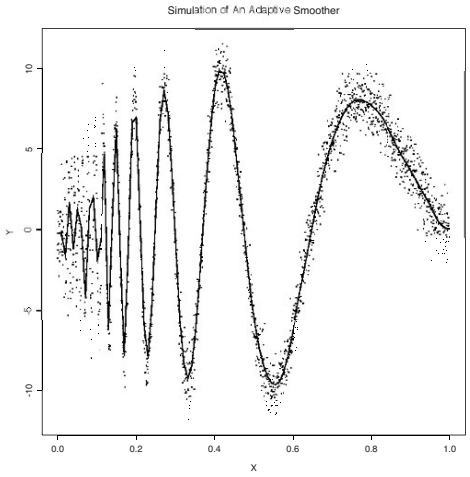


Fig. 2.22. Loader's illustration of a locally adaptive smoother.

at the far left while retaining those in the middle and far right. Important information would be lost. But how often will real data have the four properties that characterize these data? Many disciplines such as engineering have data in which such features may be common. But in the social and life sciences, these kinds of data are rare.

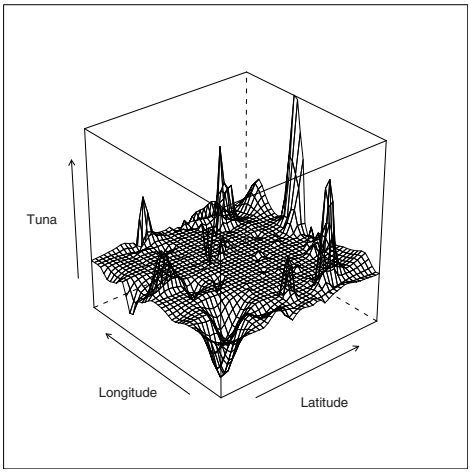


Fig. 2.23. Adaptive smoothed values of tuna caught in the southeastern Pacific.

Figure 2.23 presents some real data. The response is the number of tuna netted at various locations in the south eastern Pacific Ocean. The predictors

are longitude and latitude. Because of the flow of ocean currents and the clustering of smaller fish that tuna eat, the good fishing grounds have relatively sharp, nonlinear boundaries. Moreover the data are quite good. Because of international concern about the risks to dolphin when nets are used to catch tuna, there are official observers on tuna boats to record the size and locations of all catches. In short, the data themsleves are probably not seriously distorted by measurement error.

Once again the adaptive smoother does a good job. The productive fishing grounds are dramatically shown with some being far better than others. The smoothing process does not seem to be sacrificing the smaller spikes.

When the data are of just the sort required, adaptive smoothers can be very effective. Using them when they are not needed may not cause any scientific harm because adaptive procedures will, in effect, revert to a single smoothing parameter approach. But there can be significant computational costs. Existing software can seriously challenge the capacity of desktop computers, will usually require that several tuning parameters be specified, and are typically limited to no more than two predictors.

2.9 The Role of Statistical Inference

Many of the smoothers we have considered in this chapter rest upon a set of regression equations constructed for partitions of the data. The partitions are defined as functions of predictor values. Then, for any given partition, the fitted value is determined by a conventional parametric regression equation (sometimes with weights). Alternatively, the smoother results from a regression equation with a penalty attached for complexity. In either case, it might seem that conventional expressions for the standard error of fitted values would follow as usual. So, let's pursue that for a bit.

2.9.1 Some Apparent Prerequisites

A key issue that must be addressed before statistical inference with smoothers is undertaken is whether estimation itself is a reasonable activity. There are three scenarios.

1. There is an assumed $f(X)$, and the data are a random sample from a well-defined population or a random realization from a well-defined stochastic process. Estimation is at least on the table accompanied by assessments of uncertainty.
2. No $f(X)$ is assumed, but a goal is to arrive at a best guess of the values of a set of conditional means or proportions in a population or as features of a stochastic process. Estimation is again on the table along with assessment of uncertainty.

3. The sole goal is description of the data on hand. Estimation is taken off the table and with it, assessments of uncertainty.

We begin with the first scenario. We show that superficially all looks well. We also show, as the saying goes, that looks can be deceiving.

2.9.2 Confidence Intervals

As before we let

$$Y = f(X) + \varepsilon, \tag{2.25}$$

but where $\varepsilon \sim \text{NIID}(0, \sigma^2)$. One estimates $f(X)$ with $\hat{f}(X)$, which for the smoothers we have considered is $\mathbf{Sy}$. Then

$$\text{cov}(\hat{\text{f}}(\text{X})) = \mathbf{S}\,\text{cov}(\mathbf{y})\mathbf{S}^{\text{T}} = \mathbf{S}\mathbf{S}^{\text{T}}\sigma^2. \tag{2.26}$$

The square root of the diagonal elements of $\mathbf{S}\mathbf{S}^T\sigma^2$ are the standard errors for each fitted value. To make it operational, one needs $\hat{\sigma}^2$.

The error sum of squares can be computed as the sum of the squared differences between the fitted values and the observed values. The denominator is where there can be complications: what is one to use as the degrees of freedom lost to the fitting function? One popular definition, noted earlier, is the trace of the smoother matrix $\mathbf{S}$, which is related to the number of basis functions and to the number of parameters in the model (Hastie et al., 2001: 130). This definition is intuitively pleasing, broadly applicable to a variety of smoothers, and works well with hypothesis tests (considered shortly).

The larger the trace, the less smooth are the fitted values. This is because more relative weight is given to the values of the response variable actually being fitted and less relative weight is given to other (usually nearby) values of the response. Consider again the toy example from Chapter 1. Because the rows sum to 1.0, making the elements in the main diagonal larger makes the weights off the main diagonal smaller. The result is that the weighted average more heavily counts the value of the response being smoothed. The fitted values are relatively less smooth.

$$\begin{pmatrix} 1.0 & 0 & 0 & 0 & 0 \\ .25 & .50 & .25 & 0 & 0 \\ 0 & .25 & .50 & .25 & 0 \\ 0 & 0 & .25 & .50 & .25 \\ 0 & 0 & 0 & 0 & 1.0 \end{pmatrix} \begin{pmatrix} 3.0 \\ 5.0 \\ 6.0 \\ 9.0 \\ 10.0 \end{pmatrix} = \begin{pmatrix} 3.00 \\ 4.75 \\ 6.50 \\ 8.50 \\ 10.00 \end{pmatrix}. \tag{2.27}$$

With the effective degrees of freedom defined, $\hat{\sigma}^2$ is computed by dividing the error sum of squares by $N - \text{trace}(\mathbf{S})$. The denominator, in the same spirit as the usual regression estimate of $\hat{\sigma}^2$, represents the degrees of freedom "left over" by the model. Then $\hat{\sigma}^2$ is used in place of σ^2, making Equation 2.26 operational. Adding ± 1.96 times the square root of the diagonal to the fitted

value results leads to what looks like a 95% confidence interval at that point. All other points are treated in a similar fashion.

Because σ^2 is assumed to be constant, and because $N - \text{trace}(\mathbf{S})$ is a constant for any given dataset and model, the size of the standard error depends substantially upon the diagonal elements of $\mathbf{SS}^T$. These, in turn, depend on the diagonal elements in $\mathbf{S}$. So, larger estimated standard errors for a given dataset are found in regions where the fitted values are less smooth. And it is here that the fitted values may be a less effective stand-in for the values of the response variable. Taken at face value, this would seem to make sense.

However, there are several problems. First, the value of λ (or the analogous tuning parameter) is assumed to be known. When it needs to be determined from the data, there is an additional source of uncertainty that is not taken into account. Second, trying different possible values for λ is a form of data snooping and will often lead to estimates of uncertainty that are too optimistic. Third, unless the data were generated by probability sampling, the usual confidence intervals depend on model-based sampling, here, centered on how the values of ε are supposed to be generated (Thompson, 2002; Berk, 2003). Constructing a plausible story is usually very difficult, especially when the fitting function is to be inductively determined. Finally, the smoother tuned by λ is assumed to provide unbiased estimates of the true conditional means. In practice, this is very unlikely to be true. In particular, it will often be desirable to introduce bias to reduce the variance. And if there is bias, a 95% confidence interval will not cover the true value 95% of the time. The interval will be shifted higher or lower by the unknown value of the bias.

Recalling our earlier discussion of statistical inference for shrinkage estimators, one response can be to settle for estimates of the instability of the fitted values; the impact of the bias is ignored. Then to address instability, a bootstrap resampling of cases can lead to helpful results (Buja and Rolke, 2007) for such instability. But any intervals constructed in this manner are unlikely to be defensible as true confidence intervals.

If potential bias is to be addressed as well, there are some recent advances that have promise (Goldenshluger and Tsybakov, 2001; Zhang, 2005; Brown et al., 2005). Just as in the shrinkage case briefly addressed earlier, one can often obtain reasonably unbiased estimates of the true conditional means using the estimated conditional means for small regions of the predictor space. The disparity between those estimated conditional means and the conditional means produced by the smoother can provide important information on the direction and size of the bias in each region. When this information is combined with estimates of the variance, approximately correct confidence intervals can follow. There is not yet much formal mathematics behind these approaches and it is not clear at this point how well the procedures will perform in practice. There is also the current limitation to a single predictor.

The prospects might seem somewhat brighter under the second scenario: there is no $f(X)$, but there are population or stochastic process parameters, and the data were generated in a manner allowing for statistical inference, such

as random sampling. With no $f(X)$ to steer the analysis, interest centers only on a set of conditional means. If one treats the predictor values as fixed, this may seem like business as usual. However, the smoothing will likely introduce bias in the fitted values and the same problems surface. Confidence intervals risk being seriously misleading. In short, we are back to the previous scenario.

In practice, the third scenario is likely to be the operational one. There may be no credible $f(X)$, no population or stochastic process, or the data may be of insufficient quality (e.g., key predictors are missing). Then, the goal is likely to be description, and estimation is inappropriate. Whether any of these obstacles are recognized is often for subjectmatter experts to determine.

2.9.3 Statistical Tests

The statistical tests associated with conventional parametric regression have a structure that can be ported to the smoothers we have been considering. Consider the usual F-test used with the conventional linear regression model. Recall that the F-ratio is constructed in part from the error sum of squares under the null hypothesis and the error sum of squares from the alternative hypothesis, with their difference adjusted for the difference degrees of freedom. The ratio is meant to capture how much worse the fit becomes under the null hypothesis. The same kind of formulation can be applied with regression splines and regression smoothers.

Assume that Equation 2.25 holds. Then, drawing on Loader's discussion (2004: 17–18) — see also Hastie and Tibshirani (1990: 65–67) — suppose before looking at the data one decides that the null hypothesis is a conventional linear fit, and the alternative hypothesis is any smoother-based fit. Is the fit produced by the smoother "statistically significant" compared to the linear fit? This may not be a very interesting or instructive comparison, however, it comports well with conventional regression practice.

Let $\mathbf{H}$ be the hat matrix for a linear regression fit of the data, and $\mathbf{S}$ be the smoother matrix for some alternative fit. From this, one can construct the usual sort of test statistic as follows,

$$F = \frac{(\mathbf{Sy} - \mathbf{Hy})^2/df}{\hat{\sigma}^2}, \tag{2.28}$$

where $df = \text{trace}((\mathbf{S}-\mathbf{H})^T(\mathbf{S}-\mathbf{H}))$, and $\hat{\sigma}^2$ is usually estimated from the larger model. Loader points out that the F-ratio in Equation 2.28 does not quite have an F distribution, although there are ways to make the approximation better. Such tests are approximate and insofar as the assumed normality is incorrect, the test may not live up to its billing. Alternatively, the bootstrap can be applied. The idea is to work with the residuals in much the same manner as done for parametric regression (Efron and Tibshirani, 1993: 111–112).

1. Apply a smoother to the data.

2. Compute the residuals as the differences between the fitted values and observed values of the response variable (i.e., $\mathbf{y} - \hat{\mathbf{y}}$).
3. Draw with replacement a random sample of residuals the same size as the number of observations in the data.
4. Construct new values for the response by adding to each fitted value from Step 1, a sampled value from the residuals.
5. Compute the F-statistic of interest as in Equation 2.28. This will mean applying the null model and the alternative model to the reconstituted data.
6. Repeat Steps 2–5 a large number of times (e.g., 1000).
7. The histogram of the F-statistics provides an estimate of the true distribution of the F ratio.

But in the end, all such tests must be treated with caution. All of the concerns noted about confidence intervals apply. In particular, the smoothing process will typically lead to bias. If there is bias in the fitted values, the p-values computed will capture not just the variance but the bias. The distance between the null hypothesis and the estimated fitted values will be too large or too small, depending on the nature of the bias, and the p-values will be either too large or too small as well.

2.9.4 Can Asymptotics Help?

The asymptotics for the smoothers we have considered require that the number of observations must increase without limit and the number of unique values of the predictors (i.e., "design points") must increase without limit. That is, in order to obtain consistent estimates of the conditional means, both these conditions must apply. The number of design points must increase without limit so there are no "holes" in the fitted values. If there are holes, some form of interpolation or averaging is necessary, which means that the true conditional means in that hole will probably not be accurately represented.

In some very large datasets with relatively few predictors, these requirements may be approximately met. But for many datasets, the approximation to the requisite thought experiment is poor so that it is very difficult to rely on the asymptotic results. Equally important, if any smoothing is undertaken, there is the risk of nonnegligible bias that remains even asymptotically. To take an extreme case, if a linear fit is forced on a nonlinear $f(X)$, increasing the sample size does not overcome the bias introduced.

In short, even if estimation is a worthy goal, the associated statistical inference can be highly problematic. If one cares only about the stability of the fitted values, resampling procedures can be instructive. But if one cares about taking the bias into account, it is currently not clear how best to proceed. The good news is that statistical inference for smoothers is being addressed by some very talented statisticians. There may be some useful procedures soon.

2.10 Software Issues

All of the computing done is this chapter was implemented in R. Within R, the following smoothing and regression procedures were used.

1. Linear Regression: lm()—a very flexible and very powerful procedure for implementing the general linear model.
2. Generalized Linear Model: glm()—a very flexible and very powerful procedure for implementing the generalized linear model. Its structure is much like lm().
3. Scatterplot Smoothing: scatter.smooth()—a very flexibile and rich implementation of a two-dimensional lowess smoother of a scatter plot. The output is the scatter plot with the fitted values overlaid.
4. Local Adaptive Smoothing: locfit()—a generalization of lowess to allow for up to two predictors with local adaptation for bandwidth. The code is powerful and sophisticated, but the documentation is spotty. It can be found in the R library *locfit*.
5. Generalized Additive Model: gam()—it comes in two implementations. One can be found in the library (gam) and uses the backfitting algorithm. A second implementation uses penalized regression and can be found in the R library *mgcv*. They perform broadly the same, but differ a bit in the options offered to users.
6. Spline Basis Construction: bs(), ns()—two procedures that are used to construct b-spline bases for smoothers, bs() for B-splines and ns() for natural cubic splines. These are automatically called by some smoothing procedures or can be used as an intermediate step for more hand-tailored smoothing. They can be found in the R library *splines*.
7. Two-Dimensional Plotting: plot()—this can be used as a standalone or when paired with an R object produced by procedures such as lm(), locfit(), or gam().
8. Three-dimensional plotting: contour(), persp() for contour plotting and perspective plotting, respectively—both are slick and powerful, but a bit tricky to use. There is a need to construct the plotting grid before points and any fitted values are overlaid. Alternatively one can work with the graphing procedures in the library *lattice*. For example, wireframe() is a very elegant improvement over contour().

Perhaps the major operational problem for smoothing is sparse data. In the simplest case, there may be only a few distinct values for a predictor so that there is really nothing to smooth. For example, if a predictor only has observations at three of its values, there is not much that can be done. The choice is between no smoothing at all (i.e., just connecting the three conditional means of the response), or a single straight line. There is no clear lower limit to the number of predictor values for which there must be data, but smoothing when there are fewer than about ten values is not likely to be instructive. There can be few unique values for a predictor either because the

data are lumpy or because of how the predictor is defined and measured. For example, the number of children in a household for a given sample may have only five distinct integer values.

In addition, the curse of dimensionality can rapidly turn an adequate dataset into an inadequate one. The data may become far too thin overall, so that the large variance associated with the fitted values will negate any possibility of seeing what the mean function is likely to be. Or, important partitions of the data may suffer from the same problem. Most frustrating of all, some procedures will abort with sparse data, sometimes taking down the statistical procedures being used and even the entire operating system.

Many of the smoothing procedures have tuning parameters that can be used for taking relatively large bites of the data. For data that are potentially sparse, it will often be helpful to begin an analysis with large bites so that within each window there are a sufficient number of observations. If the fitted values seem stable, smaller bites may be tried.

A good sense of how stable the fitted values are can sometimes be obtained from a point-by-point confidence interval, as long as one does not take the attached probability very seriously. As noted earlier, bias will offset the intervals so that the coverage is unknown. But, if the point-by-point intervals are so large that the fitted values could plausibly range very widely, the fitted values do not provide a useful fix on the mean function. This is very important to take into account when the fitted values are interpreted.

2.11 Summary and Conclusions

Regression splines and regression smoothers can be very useful tools for describing relationships between a response variable and one or more predictors. As long as one is content to "merely" describe, these methods are consistent with the goals of an exploratory data analysis.

Experience suggests that for most datasets, it does not make a great difference which brand of smoother one uses. The dominant factor is usually bandwidth or other parameters that determine the bias–variance tradeoff. Likewise, all of the measures of fit that take model complexity into account lead to largely the same substantive results, especially when data are noisy.

There are also several important caveats that need to be kept in mind. First, as with any regression analysis, there is no necessary connection between the computer output and how the data were generated. There is, therefore, no necessary connection to causal inference. Although the output can be very helpful when considering matters of cause and effect, regression splines and regression smoothers are usually not meant to represent how manipulating one or more predictors will change the response.

Second, statistical inference should be approached with great care. Smoothers are often meant to be exploratory and as such can easily jeopardize formal tests and confidence intervals. Moreover, they typically introduce bias into

the fitted values with the goal of reducing their variance. It is also important to look beneath the computer output and understand how the statistical inference was undertaken.

Third, overfitting can be a serious problem. The results from the data examined may not generalize well to other random samples from the same population. We consider overfitting in depth in later chapters. For now, *caveat emptor*.

Finally, for a wide range of problems, there are statistical learning techniques that arguably perform better than the procedures discussed in this chapter. They can fit the data better, are less subject to overfitting, and permit a wider range of information to be brought to bear. One price, however, is that the links to conventional regression analysis become far more tenuous. In the next chapter, we start down this path.

Exercises

Problem Set 1: Smoothers with a Single Predictor

1. Load the dataset called airquality using the command data(airquality). Attach the data with the command attach(airquality). Use gam() from the *gam* library with Ozone as the response and Temp as the sole predictor. Estimate the following three models assigning the output of each to its own name (e.g., output1 for the first model).

   ```
   gam(Ozone ~ Temp)
   gam(Ozone ~ as.factor(Temp) )
   gam(Ozone ~ s(Temp) )
   ```

   The first model is the smoothest model possible. Why is that? The second model is the roughest model possible. Why is that? The third model is a compromise between the two in which the degree of smoothing is determined by the GCV statistic. (See the gam() documentation followed by the smoothing spline documentation.)

   For each model, examine the numerical output and plot the fitted values against the predictor. For example, if the results of the first model are assigned to the name "output1," use plot.gam (output1, residuals=TRUE).

   Which model has the best fit judging by the residual deviance? Which model has the best fit judging by the AIC? Why might the choice of the best model differ depending on which measure of fit is used? Which model seems to be most useful judging by the plots? Why is that?

2. Overlay a lowess smooth on a scatterplot with the variable Ozone on the vertical axis and the variable Temp on the horizontal axis. Vary three tuning parameters: span: .25, .50, .75; degree: 0, 1, 2; family as Gaussian or symmetric. Describe what happens to the fitted values as each tuning parameter is varied. Which tuning parameter seems to matter most?

3. The relationship between temperature and ozone concentrations should be positive and monotonic. From the question above, select a single set of tuning parameter values that produces a fit you like best. Explain why you like that fit best. If there are several sets of fitted values you like about equally, explain what it is about these fitted values that you like.

4. For the overlay of the fitted values you like best (or select a set from among those you like best) describe how temperature is related to ozone concentrations.

5. One can address the stability of the fitted values using the bootstrap percentile method. Load the library *simpleboot*. The procedure first requires that you run lowess and then you apply the bootstrap. For example: assign loess(Ozone $\sim$ Temp) to a name such as "smooth." Then assign loess.boot (smooth) to a name such as"bo." Finally use plot(bo). The point-by-point interval is constructed by taking the standard deviations of the fitted values for each point over bootstrap samples, multiplying each by two, and adding that product to the fitted values at each point and subtracting that product from the fitted values at each point.

   For what values of temperature does the instability appear to be about the largest? For what values of temperature does the instability appear to be the smallest? What in the data accounts for these differences?

Problem Set 2: Smoothers with Two Predictors

1. From the library *assist* load the dataset TXtemp. Load the library *gam*. With mmtemp as the response and longitude and latitude as the predictors, apply gam(). Construct the fitted values using the sum of a 1-D lowess smooth of longitude and a 1-D smooth of latitude. Try several different values for the degrees of freedom of each. Try different values for the degree of the polynomial. You can learn how to vary these tuning parameters with help(gam) and help(lo). Use the summary() command to examine the output and the plot.gam() to plot the two partial response functions. To get both plots on the same page use par(mfrow=c(1,1)). How are longitude and latitude related to temperature?

2. Repeat the analysis in 1, but now construct the fitted values using a single 2-D smoother of longitude and latitude together. Again, try several different values for the span and degree of the polynomial. Examine the tabular output with summary() and the plot using plot.gam(). How do these results compare to those using two 1-D predictor smooths?

Problem Set 3: Smoothers with More Than Two Predictors

1. Now build an additive model for mmtemp with the predictors longitude, latitude, year, and month. Use a lowess smooth for each. Try different spans and polynomial degrees. Again use the summary() and plot.gam() command. To get all four graphs on the same page use par(mfrow=c(2,2)). How is temperature related to each of the four predictors?

2. Repeat the analysis done for 1, but with penalized smoothing splines. The operator in front of each predictor is now s and not lo. Read the help documentation for gam(), and s(). How is temperature related to each of the four predictors? How do the conclusions from 1 compare with the conclusions drawn here? Why?

Problem Set 4: Smoothers with a Binary Response Variable

1. From the *car* library, load the dataset Mroz. Using the glm(), regress labor force participation on age, income, and the log of wages. From the library *gam*, use gam() to repeat the analysis, smoothing each of the predictors. Note that labor force participation is a binary variable. Compare and contrast your conclusions from the two sets of results. Which procedure seems more appropriate here? Why?

3

Classification and Regression Trees (CART)

3.1 Introduction

Suppose one had a single quantitative response variable and several predictors. There is interest in $\bar{y}|x$. The task is to find the single best predictor. To do this, two kinds of searches are undertaken. First, for each predictor, all possible splits of the predictor values are considered. For example, if the predictor is age, and there are age-values of 21 through 24, all possible splits maintaining order would be 21 versus 22-24, 21-22 versus 23-24, and 21-23 versus 24. If the predictor is marital status with categories never married, married, and divorced, all possible splits would be never married versus married and divorced, married versus never married and divorced, and divorced versus never married and married. For categorical variables, there is no order to maintain.

For each predictor, the best split is selected. The baseline is the sum of squares of the response variable. For each split of a given predictor, a sum of squares is computed within each of the two splits and added. Their sum will be equal to or less than the original sum of squares for the response variable. The "best" split for each predictor is defined as the split that reduces the sum of squares the most.

Second, with the best split of each predictor determined, the best split overall is determined. The same sum of squares criterion is used along with the results from the previous step. By selecting the best split overall, the best predictor by this sum of squares criterion is implicitly chosen.

With the two-step search completed, the winning split is used to subset the data. In other words, the best split for the best predictor defines two subsets. For example, if the best split were to be 21-22 versus 23-24 years of age, all individuals 21-22 would form one subset and all individuals 23-24 would form the other subset.

There are now two partitions of the original data, defined by best split within and between the predictors. Next, the same two-step procedure is applied to each partition separately; the best split within and between predictors

R.A. Berk, *Statistical Learning from a Regression Perspective*,
DOI: 10.1007/978-0-387-77501-2_3,

for each subset is found. This leads to four partitions of the data, and once again, the two-step search procedure is undertaken separately for each. The process can continue until there is no meaningful reduction in the error sum of squares.

As we show shortly, the result is a recursive partitioning of the data that can be represented within a basis function framework. The basis functions are indicator variables defined by the best splits. With these determined, a regression of the response on the basis functions yields regression coefficients and fit statistics as usual. In practice, there is no need to translate the partitioning into a regression model; the partitioning results stand on their own as a regression analysis. But if one wishes, the recursive partitioning can be seen as a special form of stepwise regression.

The two-step search procedure is easily generalized so that the response variable can be categorical, and in probably its most visible implementation, the recursive partitioning is called Classification and Regression Trees (CART). CART has been in use for about 20 years (Breiman et al., 1984) and remains a popular data analysis tool. In this chapter, CART is examined in considerable depth, not just because it can be of practical value, but because it raises a number of important, broader issues. It also can be a foundation for statistical learning discussed in three subsequent chapters.

The focus is on CART as it has traditionally been implemented. Although there are some recent refinements of CART (Chipman et al., 1998; Loh, 2002; Su et al., 2004), they are peripheral to the aims of this chapter. There are also CART-like procedures such as C5.0 (Quinlan, 1993) with roots in computer science. A discussion of these procedures would take us some distance from the statistical traditions emphasized here, although we later consider a paper by Hothorn and his colleagues (2006) that is somewhat more than a refinement of CART.

Chapter 2 was devoted almost entirely to quantitative response variables. Equal time and more is now given to categorical, and especially binary, response variables. As noted earlier, procedures that assign observations to classes are sometimes called "classifiers." When CART is used with categorical response variables, it is an example of a classifier.

Categorical response variables introduce a number of significant complications that either do not apply to quantitative response variables, or apply only at a much higher level of abstraction. We now need to get this material on the table, in part because it is important for classifiers in addition to CART. We also emphasize the differences among description, estimation, and forecasting. In CART, these are not just differences in how the tools are used, but go to the nuts and bolts of how the procedure performs.

This is a long and somewhat tedious chapter. An effort has been made to include only the material that is really needed. But that's a lot, and it is probably necessary to slog through it all.

3.2 An Overview of Recursive Partitioning with CART

As already noted, classification and regression trees works by a recursive partitioning of the data. Recursive partitioning is a stagewise process that sequentially breaks the data up into smaller and smaller pieces. It is "stagewise," not "stepwise," because earlier stages are not revisited after the results from later stages are known.

Recursive partitioning can be formulated within the basis function framework discussed in Chapter 2. Recall that

$$f(X) = \sum_{j=1}^{p} \sum_{m=1}^{M_j} \beta_{jm} h_{jm}(X). \tag{3.1}$$

Each of the p predictors has its own set of transformations, and all of the transformations for all predictors, each with its own weight β_{jm}, are combined in a linear fashion. Recall also that indicator variables were included as possible transformations. This is a key feature of CART.

To see the relevance of Equation 3.1 for CART, it is necessary to appreciate how CART implements recursive partitioning. The goal of CART's recursive partitioning is to exploit information contained within a set of predictors to create subsets of the data. Each subset is constructed so that the values of the response variable in each are as similar as possible. The process proceeds one partition at a time so that once a partition is constructed, it is not reconsidered when later partitions are defined.

Figure 3.1 is a three-dimensional scatterplot. There is a binary outcome G coded "A" or "B," and predictors x and z. Figure 3.1 is meant to illustrate a simple classification problem as it might be attacked by CART.

The single vertical line at, say, $z = 3$ produces the first partition. The double horizontal line at $x = 6$ produces the second partition. The triple horizontal line at $x = -4$ produces the third partition. CART constructs partitions with a series of straight-line boundaries perpendicular to the axis of the predictor being used.

In this simple illustration, the upper-left partition and the lower-right partition are fully homogeneous. This is good. There remains considerable heterogeneity in the other two partitions and in principle, their partitioning could continue. Figure 3.1 reveals that cases with $z \leq 3$ and $x > 6$ are always "A." Cases with $z > 3$ and $x \leq -4$ are always "B." Thus, we are on our way to describing distribution of the As and Bs conditional upon x and z. The regression framework still applies.

3.2.1 Tree Diagrams

CART output is often shown as an inverted tree. Figure 3.2 is a simple illustration. The full dataset is contained in the root node. The data are then broken into two mutually exclusive pieces. Cases with $x > c_1$ go to the right,

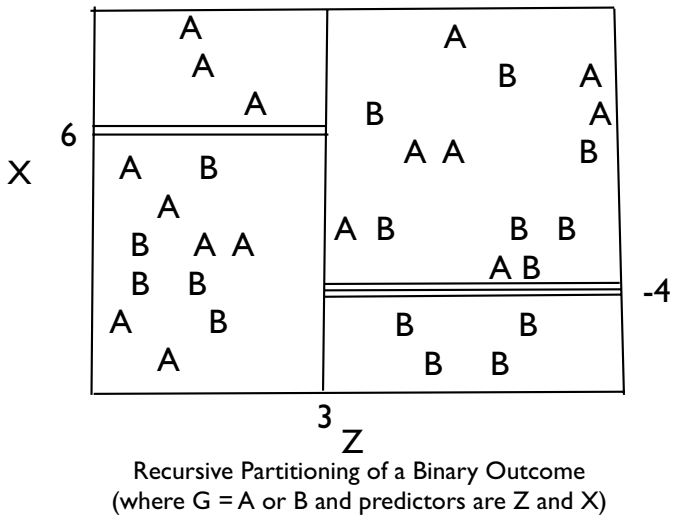


Fig. 3.1. Recursive Partitioning Logic in CART

and cases with $x \leq c_1$ go to the left. The latter are then in terminal node 1, which is not subject to any more subsetting. The former are then in an internal node that can be usefully subdivided further. The cases in the internal node are then partitioned again. Observations with $z > c_2$ go to the right and into terminal node 3. Observations with $z \leq c_2$ go to the left and into terminal node 2.

In this case, all splits beyond the initial split of the root node imply, in regression language, interaction effects. The split imposed at the internal node, for instance, only applies to observations with x-values that are greater than c_1. The impact of z depends on a value of x, which is an interaction effect.

When there is no natural order to a predictor's values, the partitioning criterion selected is usually represented by the name of the variable along with the values that go to the right (or left, depending on the software) side. For example, if ethnicity is a predictor and there are five ethnicities represented by the letters a though e, the software might represent the partitioning criterion for a given split as *ethnicity=ade*. All cases belonging to ethnic groups a,d, and e are being placed in the right-hand partition.

Splits after the initial split do not have to represent interaction effects. If an immediately subsequent partitioning of the data uses the same predictor (with a different breakpoint), the result is an additional step in the step function for that predictor. A more complicated nonlinear function results, but not an

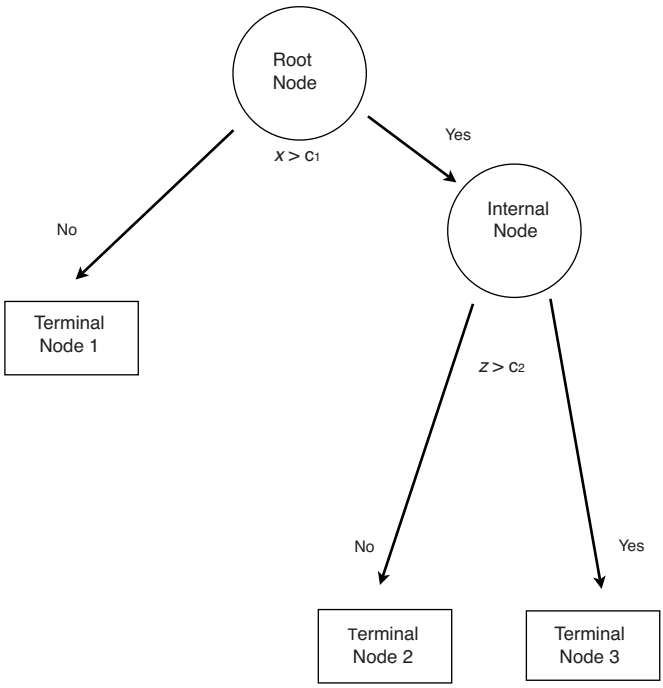


Fig. 3.2. CART tree structure.

interaction effect. In practice, however, most partitions of the data represent interaction effects.

It is easy to show that results such as those shown in Figure 3.2 can be written within the basis function framework of Equation 3.1. One just represents all of the terminal nodes with indicator variables, each of which is a function of one or more predictors (including the constant term). Thus,

$$f(X, Z) = \beta_0 + \beta_1[(I(x \leq c_1)] + \beta_2[I(x > c_1 \ \& \ z \leq c_2)] + \beta_3[I(x > c_1 \ \& \ z > c_2)]. \quad (3.2)$$

One can see again the importance of interaction effects whenever two or more predictors are needed to construct the indicator variable. Interaction effects need to be kept in mind when CART tree diagrams are interpreted.

3.2.2 Classification and Forecasting with CART

There is far more to the output from CART than a tree diagram. Within each of the terminal nodes, the proportion of "successes" and proportion of "failures" are calculated. These conditional proportions can be of significant descriptive interest. For example, if the proportion of successes in terminal node 3 is .70, one can say for cases with $x > c_1$ and $z > c_2$ that the proportion of successes is .70. Analogous statements can be made about the other terminal nodes. Ideally, these proportions will vary substantially, implying that the partitioning is making important distinctions between different kinds of cases. If you know for any given case the value of x and the value of z, it really matters for the proportion of successes.

In addition, the proportions can be used to attach labels to observations. If the majority of observations in a partition are As, all of the observations in that partition might be assigned to class A. If the majority of observations in a partition are Bs, all of the observations in that partition might be assigned to class B. These labels convey what is most typical in a partition and if the observations need to be organized into categories, provide a ready way to determine which observations belong where. When CART is used in this manner, it is being used explicitly as a classifier.

Often, the assigned classes can also be used for forecasting. Suppose one knows that observations with certain values for predictors fall in a particular partition, and that the majority of observations in that partition are, say, of category A. Then, new observations that would fall in that partition, but for which the response is unknown, might be predicted to be A as well.

3.2.3 Confusion Tables

At least as important as the tree diagram is a classification table that cross-tabulates the observed classes and the classes that CART assigns. When the observed classes and the assigned classes come from the data used to build the tree, the table can be used to understand how skillful CART has been in fitting the data. When the observed classes and the assigned classes come from data not used to build the tree, the table can be used to understand how skillful CART has been in forecasting. In either case, the cross-tabulation is often called a "confusion table." We consider confusion tables many times in the pages ahead, but a few details are important to introduce now. The confusion

	Failure Predicted	Success Predicted	Model Error
Failure	a	b	$b/(a+b)$
Success	c	d	$c/(c+d)$
Use Error	$c/(a+c)$	$b/(b+d)$	Overall Error $= \frac{(b+c)}{(a+b+c+d)}$

Table 3.1. A confusion table.

tables are structured to contain a bit more information than is customarily done.

Table 3.1 shows an idealized confusion table. There are two classes for the response variable: success and failure. The letters in the cells of the table are cell counts. For example, the letter "a" is the number of observations falling in the upper-left cell. All of the observations in that cell are characterized by an observed "failure" and a predicted "failure." If the observations are from the data used to build the tree,"predicted" means "assigned." If the observations are from data not used to build the tree, "predicted" means "forecasted." The difference between fitting and forecasting is critical in the next several chapters.

There are generally four assessments that are made from confusion tables.

1. The overall proportion of cases incorrectly classified is an initial way to assess the quality of the fit. The overall proportion of cases incorrectly forecasted is an initial way to assess forecasting skill. Both are simply the number of observations in the off-diagonal divided by the total number of observations. If all of the observations fall on the main diagonal, CART has, by this measure, performed perfectly; none of the observations are either classified or forecasted incorrectly. When no cases fall in the main diagonal, CART is a total failure. All of the observations are either classified or forecasted incorrectly.

   Clearly, a low proportion for this "overall error" is desirable, but how good is good depends on the baseline of classification or forecasting skill when no predictors are used. The real issue is how much better one does once the information in the predictors is exploited. A lot more to is said about this shortly.

2. The overall error neglects that it will often be more important to be accurate for one of the response variable classes than for another. For example, it may be more important to correctly diagnose a fatal illness than to correctly diagnose good health. This is where the row proportions shown in the far right-hand column become critical. For each actual class, the row proportion is the number of observations incorrectly classified or forecasted divided by the total of observations of that class. Each row proportion characterizes errors made by the statistical procedure or model.

When the true class is known, how common is it for the procedure to fail to identify it?

The two kinds of model failures are sometimes called "false positives" and "false negatives." Successes incorrectly called failures are false negatives. Failures incorrectly called successes are false positives. The row proportions that represent the relative frequency of model-generated false negatives and false positives should, ideally, be small. Just as for overall error, the goal is to do better using the information contained in the predictors than could be done ignoring that information. But, the exercise now is done for each row separately. It is common for the model to perform better for one outcome than the other.

3. The column proportions address a somewhat different question. They are the proportion of times when a particular class is assigned or forecasted that the assignment or forecast will be wrong. Whereas the row proportions help evaluate how well CART has performed, the column proportions help evaluate how useful the CART results are likely to be if put to work. The row proportions condition on the true class. The column proportions condition on the class assigned or forecasted. The latter, therefore, convey what would happen if a practitioner used the CART results to classify or forecast. One conditions on either predicted success or on predicted failure from which two different estimates of errors in use can be obtained. Just as for model errors, it is common for the errors in use to differ depending on the outcome. The goal is much the same as for model error: for each column, to be wrong a smaller fraction than if the predictors were ignored.

4. The lower-left cell and the upper-right cell contain, respectively, false negatives and false positives. The ratio of the number of false negatives to the number of false positives shows how the results are trading one kind of error for the other. For example, if b is 5 times larger than c, there are five false positives for every false negative. This means that CART is in this instance treating false negatives as five times more important than false positives; one false negative is "worth" five false positives. Ratios such as this play a key role in our discussion later of how to place costs on false negatives and false positives.

In summary, confusion tables are a critical diagnostic tool. We rely on them in this chapter and all subsequent ones. They also raise some important issues that are salient in the pages ahead.

3.2.4 CART as an Adaptive Nearest Neighbor Method

It can be instructive to think about CART within an adaptive nearest neighbor framework. The partitions shown in Figure 3.1 can be viewed as neighbor-

hoods defined by nearest neighbors. But unlike conventional nearest neighbor methods, CART arrives at those neighborhoods adaptively.

Consider, for example, terminal node 3 in Figure 3.2. Within that node are all observations whose values of x are greater than c_1, and whose values of z are greater than c_2. For these observations, a conditional mean or proportion can be computed. In other words, the nearest neighbors for either of these summary statistics are defined as all cases for which $x > c_1$ and $z > c_2$. Each of the observations for which this condition holds can be used to arrive at a single numerical summary for the response variable.

The neighborhood represented by the terminal nodes is adaptive in three senses. First, information from the response variable is used to determine the neighborhood. Some measure of fit is exploited. Recall that nearest neighbor methods that are not adaptive define the nearest neighbors by their similarity on predictor values alone. Second, because a large number of predictors and break points are examined, a large number of potential neighborhoods are evaluated before an actual neighborhood is defined. Third, the terminal node neighborhoods that result can be defined by different sets of predictors and different sets of cutpoints. Both are determined inductively by the CART algorithm. For example, a given predictor can help define one terminal node, but not another. Even when a given predictor is used to define more than one terminal node, it may enter at a different stage of the partitioning and use a different break point.

The terminal node neighborhoods are constructed sequentially by where in the predictor space some step function for the response is changing most rapidly. This follows from the desire to make the two resulting subsets as homogeneous as possible. Then, because for each split the single best predictor is chosen, each terminal node, and its implied neighborhood, can be defined using a subset of predictors. That is, one need not define nearest neighbors using the entire predictor space. This is in contrast to the multivariate lowess smoother discussed in the last chapter.

But making the subsets as homogeneous as possible does not usually lead to terminal nodes that are completely homogeneous. We show later that to make the terminal nodes homogeneous, very large trees can result with very few cases in each terminal node. Such trees can be very unstable. Thus in the case of binary outcomes, for example, there will usually be a mix of 1s and 0s. One clear consequence is classification error. Whatever the label that is attached to each terminal node, it will be the incorrect label for some observations.

A second consequence is more subtle. Suppose the goal is to estimate the proportion of 1s for all observations with the same set of x-values; one is interested in $\bar{y}|x_0$, where x_0 represents the given set of x-values (e.g., Asian, female, high school students with family incomes of more than $100,000). Unless the terminal node in which all such cases land contains only those observations, there will be other observations with different sets of x-values. When a proportion of 1s is calculated, all of the observations in the node will

be used. Unless all of the proportions of 1s for each of these different sets of x-values are the same as for cases with x_0, the estimated proportion for the observations with x_0 will be biased. For example, Asian and Anglo female high school students with family incomes of more than $100,000 may be pooled in a given terminal node. If the two groups have different proportions of 1s for the response variable, a biased estimate of the proportion of 1s will follow. More is said about this later.

In summary, although smoothers, adaptive nearest neighbor methods, and CART come from very different traditions, they have important similarities. We show additional and helpful connections to other statistical learning procedures in subsequent chapters.

3.2.5 What CART Needs to Do

With the overview of CART completed, we can begin a more detailed discussion. This discussion can be put into a useful context by considering the technical problems CART must solve. There are six such problems.

1. A criterion for the subsetting is required. By what criteria will the partitions be determined?
2. A criterion for the variable selection is required. At each stage, how will the variable used to define the new partition be selected?
3. A way is needed to consider how "good" the tree is. Regressionlike fit measures can be useful, but for classification problems, there will be classification errors with which to contend. Can a tree that makes lots of mistakes in classifying or forecasting cases be "good?"
4. A way is needed to influence the size of the tree so that only useful terminal nodes are constructed. We show that this is related to the bias–variance tradeoff.
5. A way is needed to protect against overfitting. CART is another example of high-powered exploratory data analysis. How can the generalizability of the results be strengthened?
6. Ways are needed to interpret and communicate the results. Tree diagrams are a start, but by themselves neglect some important features of CART results.

The specific solutions to these problems depend in part on whether the response is categorical or quantitative: whether a classification tree or a regression tree, respectively, is desired. Here, we continue with the emphasis on classification problems for categorical response variables and address each of the six problems along the way.

Software can matter too. There are several popular implementations of CART as originally formulated by Breiman and his colleagues (1984). These differ largely in details, but sometimes more fundamental differences arise, especially as new approaches to recursive partitioning are developed. Consistent

with the use of R for all computing in this book, CART implementations in R are the focus. In addition, the focus is on the traditional CART approach because this is the structure on which more recent statistical learning procedures most commonly build.

3.3 Splitting a Node

The first problem that CART needs to solve is how to split each node using information contained in the set of predictors. For an equal interval predictor with m distinct values, there are $m - 1$ splits that maintain the existing ordering of values. So, $m - 1$ splits on that variable need to be evaluated. For example, if there are 50 distinct high school GPA scores possible, there are 49 possible splits that maintain the existing order. However, there are often algorithmic shortcuts that can capitalize, for instance, on ordering the splits by the size of the conditional mean or proportion. The same logic holds for ordinal predictors.

Order does not matter for categorical predictors. Consequently, a categorical variable with k categories has $(2^{k-1} - 1)$ possible splits. For example, if there are five ethnic categories, there are 15 possible splits. Hence, although there are sometimes shortcuts here too, the computational burdens are generally much heavier for categorical variables. There are no restrictions on how a categorical predictor is split.

Starting at the root node, CART evaluates all possible splits of all predictor variables and picks the "best" single split overall. The best split of the variable selected is better than the best split of any other predictor. The data are then partitioned according to that best split. The same process is applied to all subsequent nodes until all cases have been placed in a terminal node. Because the final partitions do not overlap, each case can only be in one terminal node. But how is "best" to be defined? It is common to focus on the "impurity" of a node. The goal is to have as little impurity overall as possible. Consequently, the "best" split is the one that reduces impurity the most. To help simplify the exposition that follows, assume a binary response variable coded 1 or 0. The term "success" for now is used to refer to outcomes coded "1" and "failure" to refer to outcomes coded "0."

Many formal expositions of CART assume the data are a random sample from a well-defined population. Then one can consider, for example, the proportion of times in a limitless number of independent samples that a success or failure would occur. Proportions computed from the sample data can be used as an estimate of these probabilities. When the data are a population, the thought experiment of a limitless number of independent samples no longer applies, and, therefore, there are no probabilities to estimate. The proportions stand on their own as descriptive statistics.

If the data are a nonprobability sample, there is the option of invoking model-based sampling, common in conventional regression (Thompson, 2002:

section 8.3). However, model-based sampling must be used very cautiously. There is no a priori model to determine how uncertainty is introduced. If there is any model at all, it is formulated inductively from the data. Consequently, all of the issues surrounding statistical inference with smoothers resurface. It can be better in practice to let the proportions computed from a nonprobability sample stand on their own as summaries of the data, but not as estimates of anything.

The exposition that follows makes an effort to consistently distinguish between proportions and probabilities. If the goal is description, a focus on proportions is sufficient. If the goal is estimation, then estimation of probabilities necessarily enters.

Suppose for now that the data are a random sample from a well-defined population, and the concept of a probability applies. Consider a given node, designated as node A. The "impurity" of node A is taken to be a nonnegative function of the probability that $y = 1$, written as $p(y = 1|A)$. If A is a terminal node, ideally it should be composed of cases that are all equal to 1 or all equal to 0. Then $p(y = 1|A)$ would be estimated as 1.0 or 0.0. Intuitively, impurity is the smallest it can be. In contrast, if half the cases are equal to 1 and half the cases are equal to 0, the estimated probability is equal to .50. A is the most impure it can be because a given case is as likely to be a 1 as it is a 0.

One can more formally build on these intuitions. Let the impurity of node A be

$$I(A) = \phi[p(y = 1|A)], \tag{3.3}$$

with $\phi \geq 0$, $\phi(p) = \phi(1-p)$, and $\phi(0) = \phi(1) < \phi(p)$. In other words, impurity is nonnegative, and symmetrical with a minimum when A contains all 0s or all 1s, and a maximum when A contains half of each. Note that the use of I in Equation 3.3 for impurity should not be confused with the use of I to represent an indicator variable. The different meanings should be clear in context.

There remains a need to define ϕ. Three definitions have been used in the past: Bayes error, the cross-entropy function, and the Gini index. In order they are:

$$\phi(p) = \min(p, 1 - p); \tag{3.4}$$

$$\phi(p) = -p\ \log(p) - (1 - p)\ \log(1 - p); \tag{3.5}$$

and

$$\phi(p) = p\ (1 - p). \tag{3.6}$$

All three functions for impurity are concave, having minimums at $p = 0$ and $p = 1$ and a maximum at $p = .5$. Entropy and the Gini index are the most commonly used, and generally give very similar results except when there are more than two response categories. Then, there is some reason to favor the Gini index (Breiman et al. 1984: 111). The Gini index is more likely to partition the data so that there is one relatively homogeneous node having

relatively few cases. The other nodes are then relatively heterogeneous and have relatively more cases. For most data analyses, this is a desirable result. Entropy tends to partition the data so that all of the nodes for a given split are about equal in size and homogeneity. This is generally less desirable. But the choice between the two impurity functions can depend on the costs associated with classification errors, which is a topics addressed shortly. indexGini index

One might legitimately wonder why CART does not directly minimize classification error. Direct minimization of overall classification error is discussed in some detail by Breiman and his colleagues (1984: Section 4.1). One serious problem is that there can be several splits for a given stage minimizing classification error. A more subtle problem is that minimizing classification error at each stage has a tendency, like entropy, to produce a tree structure that is often more difficult to interpret. For now, we focus on node impurity as just defined. However, direct minimization of classification error resurfaces as a useful goal when boosting is considered in Chapter 6.

Building on Zhang and Singer (1999; Chapters 2 and 4), a simple example may help to make the discussion of impurity more concrete. For any internal node, we focus on a potential left "daughter" node A_L, and a right "daughter" node A_R. We wish to evaluate the usefulness of a potential partitioning of the data. Table 3.2 provides the information needed. We continue to work with probabilities, although the same practical lessons follow using proportions instead. And with no important loss of generality, the illustration uses entropy as the way impurity is represented.

As before, we let $y = 1$ if there is a success and 0 otherwise. Because the data are a random sample, estimation is a legitimate enterprise; we are not limited to description alone. The estimate of $p(y = 1|A_L)$ is given by $n_{12}/n_{1.}$. Similarly, the estimate $p(y = 1|A_R)$ is given by $n_{22}/n_{2.}$.

	Failure	Success	Total
Left Node: $x \leq c$	n_{11}	n_{12}	$n_{1.}$
Right Node: $x > c$	n_{21}	n_{22}	$n_{2.}$
	$n_{.1}$	$n_{.2}$	$n_{..}$

Table 3.2. Information used to determine the usefulness of a potential split.

It follows that "entropy impurity" for the left daughter is

$$I(A_L) = -\frac{n_{11}}{n_{1.}}\log(\frac{\mathrm{n_{11}}}{\mathrm{n_{1.}}}) - \frac{\mathrm{n_{12}}}{\mathrm{n_{1.}}}\log(\frac{\mathrm{n_{12}}}{\mathrm{n_{1.}}}). \tag{3.7}$$

"Entropy impurity" for the right daughter is

$$I(A_R) = -\frac{n_{21}}{n_{2.}}\log(\frac{\mathrm{n_{21}}}{\mathrm{n_{2.}}}) - \frac{\mathrm{n_{22}}}{\mathrm{n_{2.}}}\log(\frac{\mathrm{n_{22}}}{\mathrm{n_{2.}}}). \tag{3.8}$$

Imagine that for the left daughter there are 300 observations with 100 successes and 200 failures. It follows that the impurity is $-.67(-.40) - .33(-1.11) = .27 + .37 = .64$. Imagine now that for the right daughter there are 100 observations with 45 successes and 55 failures. It follows that this impurity is $-.55(-.60) - .45(-.80) = .33 + .36 = .69$.

To put these numbers in context, it helps to consider the smallest and largest possible values for the impurity. The greatest impurity one could obtain would be for 50% successes and 50% for failures. The computed value for that level of impurity would be .693. For proportions of 1.0 or 0.0, the value of entropy impurity is necessarily 0. In short, the minimum value is 0, and the maximum is a little more than .69. The closer one gets to 50-50, where the impurity is the greatest, the closer one gets to .693. The impurity numbers computed are rather close to this upper bound and reflect, therefore, substantial heterogeneity found in both daughter nodes. It is likely that this split would not be considered to be a very good one.

Once all possible splits across all possible variables are evaluated in this manner, a decision is made about which split to use. The impact of a split is not just a function of the impurity of a node, however. The importance of each node must also be taken into account. It stands to reason that a node in which few cases are likely to fall should be less important than a node in which many cases are likely to fall. In the big picture, the former probably will not matter much, but the latter probably will.

We define the improvement resulting from a split as the impurity of the parent node minus the weighted left and right daughter impurities. If this is a large number, entropy impurity is reduced substantially.

More formally, the benefits of the split s for node A,

$$\Delta I(s, A) = I(A) - p(A_L)I(A_L) - p(A_R)I(A_R), \tag{3.9}$$

where $I(A)$ is the value of the parent impurity, $p(A_R)$ is the probability of a case falling in the right daughter node, $p(A_L)$ is the probability of a case falling in the left daughter node, and the rest is defined as before. The two probabilities can be estimated from the information such as provided in Table 3.2; they are just the marginal proportions $n_{1.}/n_{..}$ and $n_{2.}/n_{..}$.

$\Delta I(s, A)$ is essentially the reduction in the deviance and thus, there is a clear link to the generalized linear model that can prove useful when different fitting procedures are compared. CART finds the best $\Delta I(s, A)$ for each variable. The variable and split with the largest value are then chosen to define the new partition. The same approach is applied to all subsequent nodes.

It makes no difference to the CART algorithm whether the proportions computed are taken at face value as summary statistics, or as estimates of probabilities. The partitions that result are the same. What can differ is whether a given dataset or a random sample is being analyzed. This is up to the user.

The CART algorithm can keep partitioning until there is one case in each node. There is then no impurity whatsoever. Such a tree is called "saturated."

However, well before a tree is saturated, there will usually be far too many terminal nodes to interpret, and the number of cases in each will be quite small. The very small node sizes lead to very unstable results. Small changes in the data can produce trees with rather different structures and interpretations. One option is to prohibit CART from constructing any terminal nodes with sample sizes smaller than some specified value. A second option is considered shortly. And we show in later chapters that there can be ways to work usefully with saturated trees, as long as there is a very large number of them.

3.4 More on Classification

For some applications, the data analysis can stop once all of the cases are assigned to a terminal node. The partitions and the proportions of successes in each are all that matter. For example, very much within a regression framework, it may be of interest to learn which characteristics of students are associated with the estimated probability of dropping out of school. How much higher might the probability be for students whose parents did not graduate from high school themselves, compared to the probability for students whose parents did graduate (Thompson, 2002: Section 8.3)?

But often there is an important additional step. That step is classification. Using the distribution of cases in a given node, the user wants to call all cases in that node the same thing. For example, if the students in a particular terminal node have an estimated probability greater than .50 of dropping out of school, all students in that node might be labeled as high risk, and then be offered special remedial services. The data partitions constructed by CART are now fixed. Classification takes the partitions as given and applies a rule by which all the observations within a given terminal node are assigned to a single class.

Classification raises a number of new issues that revolve around the consequences of classification errors. What happens to students who are really at high risk for dropping out of school but who are not identified as such? What happens to students who are not at high risk for dropping out of school, but who are labeled as high risk? To consider such questions, we need to be a bit more clear on what fitted values are in CART.

3.4.1 Fitted Values and Related Terms

We need to broaden the discussion just a bit to consider some ways in which CART is related some other techniques and to clarify some terms that can be used in more than one way. In particular, the term "fitted values" can have several different meanings.

CART is a method to construct, using a set of predictors, a set of conditional distributions. Interest commonly centers on some measure of location for those conditional distributions. For classification problems, the conditional

proportion is usually the measure. We show later that for regression problems, the measure is usually the conditional mean. And we have already discussed how, using basis functions, explicit links to parametric regression can be made. It follows that most of the issues raised by parametric regression, and most of concepts associated with parametric regression, carry over.

But there are also some ways in which CART's links to parametric regression can be a bit confusing. To begin, one must be clear about the distinction between the value of the response variable associated with each case, and the value of the response variable that CART can assign. The former comes directly from the data themselves and is unrelated to whatever statistical procedures are applied. The latter is an output of CART. One can think of the values assigned to observations by CART as fitted values, much as in conventional regression analysis.

Quantitative Fitted Values

For classification problems, there are two kinds of fitted values. First, each terminal node can be characterized by the proportion of cases for each of the classes. The CART algorithm determines which observations go to which terminal nodes and stops. Within each node, the proportion of observations from each of the classes is then computed. Once these are computed, they can be used to characterize all the observations in a given terminal node.

For example, if the response variable is binary, the proportion of "successes" in a given terminal node can be assigned to each observation in that terminal node. If that proportion is .25, for instance, one can say that for all the cases in that node, the proportion of successes is .25. This is an illustration of description.

Sometimes it may be possible to treat the data either as a random sample from a well-defined population or as a realization of a well-defined stochastic process. Then, the computed proportions for each terminal node can be viewed as estimates of population values or as estimates of parameters associated with the stochastic process. The proportions may then be interpreted as probabilities. The proportion of successes of, say, .25 becomes an estimate of some population value or of some parameter defining the stochastic processes. When it is then assigned to each case in a given terminal node, it may be interpreted as an estimate of the probability of a success for that case.

Qualitative Fitted Values

The second kind of fitted value requires CART to take an additional step: a class must be assigned to each terminal node. As described above, one way this may be accomplished is by a majority vote within each terminal node (or plurality if there are more than two response categories). Then the class assigned to a terminal node is assigned to each observation in that terminal node. The assigned class can be represented by any set of distinct characters,

but for binary response variables, values such as "0" and "1" are common and handy. If, for instance, a given terminal node is assigned a class of "1", all observations in the node are assigned a class of "1."

Much as with the proportions assigned to observations, the classes assigned to observations may be treated as descriptions of the data on hand and if justified, as estimates as well. The class assigned is the estimated class. Even when there is no forecasting involved, the estimated class is often called the "predicted" class.

To summarize, there are in CART two kinds of fitted values used for description: the classes assigned to each terminal node and the proportions of observations that fall into each class. There are also two kinds of fitted values used for estimation: the estimated class and the estimated probability of one class versus another. These distinctions are easy enough to remember and are needed when more advanced procedures are introduced in the next chapter. There are several new ways to think about fitted values and several new kinds as well.

3.4.2 An Example

A key issue for prison administrators is understanding which inmates are likely to place themselves and others in harm's way. Use of narcotics, assaults on prison guards, and homicides are examples. Although such events are relatively rare, they carry very serious consequences. It follows that it would be very useful if such conduct could be anticipated. Then, for the high-risk inmates, preventive measures might be taken. For example, inmates from rival street gangs might be housed in different prisons. A prerequisite, however, is a way to find useful predictors of misconduct in prison.

Using data from the administrative records of a large state prison system, Figure 3.3 shows a classification tree for which inmates engage in some form of reportable misconduct while in prison. A minimum node sample size of 200 was imposed to stabilize the results and for this initial CART example, to keep the diagram very simple. The two predictor variables in Figure 3.3, selected by CART from a larger set of 12 predictors, are defined as follows.

1. term: Nominal sentence length in years. (The nominal sentence is the sentence given by the trial judge. Inmates are often released before their nominal sentence is fully served.)
2. agerec: Age at arrival at the prison reception center in years with a = 16–20, b = 21–26, c = 27–35, and d = 36 or older.

Terminal nodes are labeled "0" if the majority do not engage in misconduct and "1" if the majority do. The numbers below each terminal node show the distribution of no misconduct to misconduct. Thus, for the root node, which contains all of the data before any partitions are constructed, there are 3807 cases with no reported misconduct and 999 cases with reported misconduct.

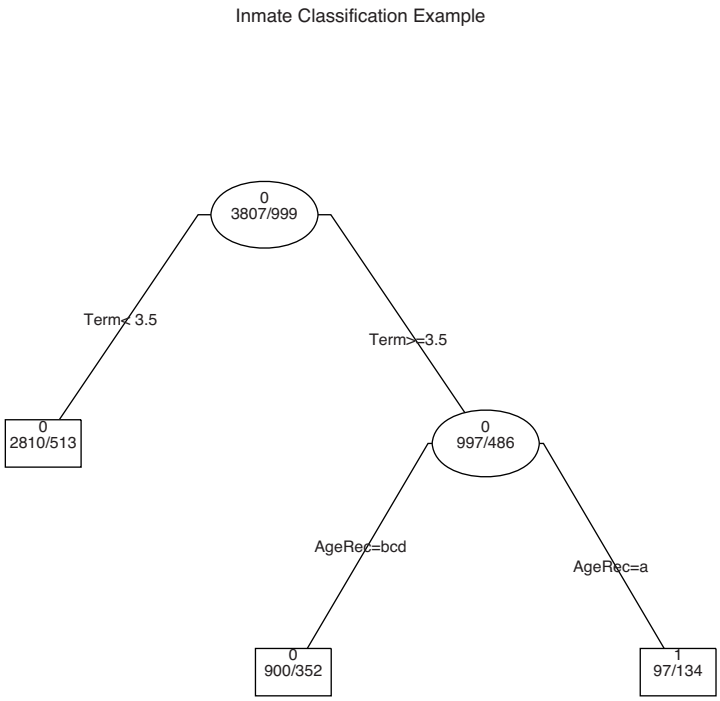


Fig. 3.3. Recursive partitioning of prison data.

Over the 18 months in which the data were collected, about 22% of the inmates had at least one reported misconduct incident.

Figure 3.3 shows a useful level of skill even using just 2 of the 12 predictors. For the far-right terminal node, there are 97 cases with no misconduct and 134 cases with misconduct. In this partition of the data, a little over 58% of the inmates have at least one reported misconduct incident. As a descriptive matter, misconduct for these inmates is about 2.5 times more common than for all inmates on the average and much higher than for any of the other terminal nodes.

All of the cases in the far-right terminal node are inmates who are serving terms of more than 3.5 years, and who arrived at the prison reception center

under the age of 21. They are very young offenders sentenced to long prison terms. It is rather difficult to receive a long prison term at so young an age. Note also that there is an interaction effect between term length and age at reception. Age at reception only has its effect for inmates who are serving terms of 3.5 years or longer.

Generally, crimes committed before the age of 18 do not count when at sentencing an offender's criminal record is considered. A long sentence usually requires several felonies or one very serious felony. So, the inmates in the right terminal node have either been very active or have engaged in very serious crimes.

That such inmates are also difficult in prison would surely be no surprise to criminologists, but Figure 3.3 contains useful information for prison administrators. Very young inmates serving long prison terms may need to be handled somewhat differently from other inmates. One might place them, for instance, in prisons where the day-to-day supervision is more intrusive.

A conventional analysis using logistic regression would likely not have performed as well by comparison. Consider what form a comparable logistic regression would have to take. The far-right terminal node would be a double interaction effect represented by a product variable constructed from two terms. The node just to its left would as well. The far left terminal node is the only main effect. It is unlikely that a researcher would have specified such a model for a logistic regression a priori. When the more likely all-main-effects model was applied to these data, the fit was dramatically worse and led to somewhat different conclusions.

But what if prison administrators want to identify a subset of inmates who are disproportionately likely to engage in misconduct? Then, the classification step is required. For the moment, we apply a simple rule: if the majority of inmates in a given node have reported incidents of prison misconduct, all inmates in that node will be classified as high risk for such behavior. This is consistent with the labels of "1" or "0" in Figure 3.3, and the resulting classifications could lead prison administrators to treat differently the inmates labeled as high risk.

Implicit in the desire to identify high-risk inmates using a set of predictors is to treat the data on hand as a random realization from whatever the stochastic process is that delivers convicted felons to prison. Then, a CART analysis of the data on hand may be used to construct estimates of the likely class for new inmates as they come in the front door and of the probability of misconduct as well. Descriptors are being treated as estimates and estimates can then be used as forecasts.

More details on how this might be done are considered later. But the basic idea is to use the tree diagram. New observations for which the response is unknown would be assigned to a terminal node based on their particular predictor values (e.g., term greater than 3.5 years and age under 21). For each observation, the class previously assigned to each terminal node would be used as the forecasted class for that observation. And for each observation,

the proportions previously used to describe each terminal node would be used as the forecasted misconduct probability for that observation.

But for such forecasts to be fully useful, a way is needed to build in the consequences of the forecasting process. And in fact, CART is making some default decisions about those consequences, which a user of the forecasts may not like. In this example, CART is treating the costs of failing to properly identify high-risk inmates the same as the costs of falsely identifying high-risk inmates. This equivalence may be undesirable, so the costs of misclassifications need further discussion. We turn to that now.

3.5 Classification Errors and Costs

Up to this point, we have proceeded with CART classifying by majority vote. For each terminal node, CART counts the number of cases in one class and the number of cases in the other class, and classifies all cases as the class with the majority of votes. When there are more than two categories, classification is by the category with the greatest number of votes, which then can be just a plurality.

Going back to Figure 3.3, and the terminal node in the lower right hand corner as an illustration, there are 97 votes (i.e., cases) for no misconduct and 134 votes (i.e., cases) for misconduct, so all cases in that node are classified as "misconduct." That implies that 97 cases are misclassified. They are classified as inmates who engaged in misconduct when they actually had not.

The other two terminal nodes in Figure 3.3 would be approached in the same way. In each of these terminal nodes, a majority vote would produce a no misconduct classification. Then, each of the cases for which there actually was misconduct would be misclassified. So, there are 352 classification errors for the terminal node in the middle and 513 classification errors for the far-left terminal node.

Whether Figure 3.3 is satisfactory from a user's point of view, however, depends on more than the number of classification errors. It depends on how the classifications will be used. Looking at the far-right terminal node again, there are 97 "false positives." If the cost of false positives is very high, the results in that node may be unsatisfactory.

Suppose the CART analysis were used for forecasting and that for new inmates classified as high risk, special housing arrangements were desirable. But such housing, which typically requires closer levels of supervision, can be very costly. Moreover, there may be a relatively small number of beds within the prison system that would be appropriate. Yet, the far terminal node in Figure 3.3 implies that for every ten inmates who really might need the special housing, there would be about seven who probably do not. Perhaps it would be better, therefore, if the threshold by which high risk inmates were classified was higher. For instance, rather than a majority vote, a two-thirds vote might be required.

Consider now either one of the other two terminal nodes. All of the new inmates falling in the middle node would not be candidates for special housing. Yet, for every ten inmates appropriately classified as low risk, there would be about four inmates who really were not. Should one of them attack a guard or another inmate, the costs could be very high. Perhaps it would be better if the threshold for high-risk inmates were lower. For instance, rather than a majority vote, a one-third vote might be required.

Thus, there would seem to be conflicting voting rules and no apparent way to reconcile them. Should the classification assigned to new inmates be determined by one-third vote, a majority vote, a two-thirds vote, or something else? And there would seem to be no solution to this problem unless costs are somehow factored in.

One could also have forecasted the probability of misconduct. But that would have implied a different kind of forecasting enterprise. At the level of the individual inmate, the truth that the world could ultimately provide is whether for that inmate there was or was not a reported incident of misconduct. So, that is what needs to be forecasted. And that is the outcome for which the costs are forecasting errors can properly be addressed. There is no observed value of the response at the level of the individual inmate that would make sense coded as a proportion. Consequently, there is no truth to which a forecasted probability can be compared.

In contrast, if the goal were to forecast the proportion of inmates in a particular group (e.g., a particular cell block) who would have a reported incident of misconduct, then the world could in principle generate a true proportion, which might be of interest to forecast. But at the level of the group, determining the conditional proportion is no longer a classification problem, but a regression problem. We consider regression trees later. Suffice it for now to say that the way costs are taken into account in regression trees is quite different.

3.5.1 Default Costs in CART

Without any apparent consideration of costs, CART can make classification decisions about the misconduct of inmates. But in fact, costs are factored in. Table 3.4 shows some results.

	Predict No Misconduct	Predict Misconduct	Model Error
No Misconduct	3710	97	.03
Misconduct	865	134	.88
Use Error	.19	.42	Overall Error = .20

Table 3.3. CART confusion table for forecasting inmate misconduct

As noted earlier, tables of the form of Table 3.4 are sometimes called confusion tables. They can summarize in a particular way, the classification skill (or as we see later, forecasting skill) of a particular classifier. Here, that classifier has been constructed by CART. The table is a cross-tabulation of the actual outcome against the outcome classified. There is a row for each of the two actual outcomes. Each column is for the outcome classified. Correct classifications are in the main diagonal. Misclassifications are in the off-diagonal elements. Thus, we learn that 962 out of 4806 cases (i.e., .20) were incorrectly classified. But, how good this is depends on the baseline.

Had no predictors been used, classification could have been done from the marginal distribution alone. On the one hand, all cases could have been classified as having no reported misconduct. Then, about .22 (999/4806) of the cases would have been incorrectly classified. All cases could have been classified as having reported misconduct. Then, about .78 (3807/4806) of the cases would have been classified incorrectly. Clearly, classifying all cases as if no misconduct had occurred leads to a far lower error proportion and using no predictors, is as good as one can do. Then, it seems that CART is not reducing misclassification all that much (.22 to .20).

However, the overall fit ignores how well CART does when the two response variable categories separated. In this case, the absence of misconduct can be classified with near perfection. In contrast, instances of misconduct are misclassified about 88% of the time. Is this a desirable balance?

The columns in Table 3.4 are also instructive. If the class of no misconduct is assigned, it is wrong for about .19 of the observations. If the class of misconduct is assigned, it is wrong for about .42 of the observations. So, mistakes are relatively more common when misconduct is assigned. Is this desirable?

For both the row and column proportions, a lot depends on the off-diagonal cells. In the process of minimizing the heterogeneity for each partition of the data, CART arrives at a result with 97 instances in which a case is classified as having reported misconduct when that is false. There are also 865 instances in which a case is classified as having no reported misconduct when that is false. Given the concern about inmate misconduct, we call the former false positives and the later false negatives.

From Table 3.4, one can see that there are 8.9 false negatives for every false positive (865/97). The default CART solution for these data trades nearly 9 false negatives for 1 false positive. So, for every inmate who might be incorrectly placed in a high-security setting, for example, there are nearly 9 inmates who might be incorrectly placed in a low-security setting. This implies that whatever the actual costs of false negatives and false positives, false positives are being treated as if they were about 9 times more costly than false negatives.

Several important lessons follow. First, CART (and every other classification procedure for that matter) must introduce costs when a classification decision is made. There is no way to circumvent this. Second, even if the data analyst never considers costs, they are built in. To not consider costs is to

leave the cost determination to the classification algorithm. Third, the way cases are classified or forecasted will vary depending on the costs introduced. As a result, the entire confusion table can change dramatically. The costs used can make a very big difference. Finally, the costs that matter are the costs of classification errors. And as the discussion of Table 3.4 illustrates, it is the ratio of those costs that is critical.

If costs are so important, there is a need to understand how they are incorporated in CART. And this will set the stage for the data analyst to introduce costs explicitly into the analysis. A key step is to appreciate the role of the "prior probabilities" associated with the categorical response variable.

3.5.2 Prior Probabilities and Costs

The marginal distribution of any categorical response variable will have a proportion of the observations in each response category. In our prison example, .22 of the inmates had a reported incident of misconduct, and .88 of the inmates did not. However, before looking at the data, one might already hold strong beliefs from past research or other information about what those marginal proportions should be. For example, the design through which the data were collected may have over sampled inmates reported for misconduct in order to have a sufficient number of them in the study. But for many uses of the results, it would make sense to weight the observations back to the actual proportion of inmates who engage in misconduct. These actual proportions can sometimes be conceptualized as the "prior probabilities" associated with the response variable. The word "prior" comes from Bayesian statistical traditions in which the "prior" refers to the beliefs of the data analyst, before the data are examined, about the probability density or distribution of some parameter.

There has been some recent work within Bayesian traditions that allows for a "pinball prior" for tree size and some features of tree shape (Wu et al., 2007). That is, key features of the tree itself are given a prior probability distribution. The ideas advanced are truly interesting, but in practice it is not clear whether there would be information available to make the pinball prior more than a convenient fiction, and there is almost no experience with this approach to date. It will be some time before we learn whether there is much payoff for real data analysis. Consequently, when the term "prior" is used from here forward, reference is being made, unless otherwise stated, to the prior distribution of the response variable only.

Before we get any farther, we need some additional notation. This notation and the surrounding discussion draws heavily on Therneau and Atkinson (1997). Suppose there are N observations, C classes for the response variable, and K terminal nodes. We define π_i for $i = 1, 2, \ldots,$ as the prior probability of being in class i. For the binary cases, i would equal 1 or 2. As just noted, these marginal probabilities are sometimes called "prior probabilities."

$L(i, j)$ is the loss matrix for incorrectly classifying a case that is really an i as a j. It is this matrix that captures the costs of classification errors. For a binary outcome, the matrix is 2 by 2, where the cost for correct classification is, with no loss of generality, taken to be zero.

We let A be some node in the tree and $\tau(x)$ be the true class for an observation x, where x represents the vector of predictor variable values for that observation. We also let $\tau(A)$ be the class assigned to node A if node A is a terminal node. Finally, N_i and N_A are the number of observations in the sample that are in class i and in node A, respectively, with N_{iA} the number of observations of class i in node A. The following relationships then hold.

1. $P(A)$ is the probability of cases appearing in node A, which is equivalent to $\sum_{i=1}^{C} \pi_i P[x \in A|\tau(x) = i]$. It can be estimated by $\sum_{i=1}^{C} \pi_i(N_{iA}/N_i)$. Note that the prior probabilities figure directly in these calculations and, as a result, can affect the tree structure.
2. Then, $p(i|A)$ is the probability of class i given that a case is in node A, or $P[\tau(x) = i|x \in A]$. It can be estimated by the number of cases of class i in node A, divided by the total number of cases in that node. It is instructive that this probability equals $\pi_i P[x \in A|\tau(x) = i]/P[x \in A]$, which can also be estimated by $\pi_i(N_{iA}/N_i)/\sum_{i=1}^{C} \pi_i(N_{iA}/N_i)$. The priors can make a real difference because the probability of a case with true class i landing in A depends in part on the probability that a case is truly of class i to begin with.
3. $R(A)$ is the "risk" associated with node A, where $\sum_{i=1}^{C} p(i|A)L(i, \tau(A))$, and where $\tau(A)$ is chosen to minimize risk. In other words, the risk associated with node A is for the binary case the probability of a case of type "1" falling in that node times costs that follow, plus the probability of a case of type "2" falling in that node times costs that follow. Risk is, therefore, a function of both the probabilities and the costs. Because the probabilities depend on the prior probabilities, the prior probabilities affect risks.
4. $R(T)$ is the risk of the entire tree T, which equals $\sum_{j=1}^{K} P(A_j)R(A_j)$, where A_j are the terminal nodes of the tree. We are now just adding the total risk associated with each node, weighting by the probability of cases falling in that node. This can also be called the "expected cost" of the entire tree.

And now the punch line. If $L(i, j) = 1$ for all $i \neq j$, and the prior probabilities π_i are taken to be the observed class proportions in the sample, then $p(i|A) = N_{iA}/N_A$, and $R(T)$ is the proportion misclassified. The same applies to the $R(A)$, the risk associated with any particular terminal node. Replacing $L(i, j) = 1$ with $L(i, j) = m$, where m is some constant, just scales up or down the risk by some arbitrary amount and makes no difference to the CART algorithm.

Therefore, if we just let the data determine everything, it is the same as (a) making the costs of all classification errors represented in the loss function

the same and (b) taking the empirical distribution as the appropriate prior distribution. The partitions that follow depend on two conditions. Classification by a majority vote of the cases in each terminal node also depends on these conditions. If either of these conditions is different, it can easily lead to different partitions and different classifications.

We are now ready to revisit the tree diagram in Figure 3.3 and the numbers in Table 3.2. These are the tree and classifications that result when the researcher does not consciously introduce the relative costs of false negatives and false positives. Figure 3.3 assumes equal costs for all classification errors in the loss function and the empirical distribution of the response variable as the prior distribution.

But what does one do if as in Table 3.2 the balance of false negatives to false positives is unsatisfactory? It would seem that the easiest thing to do would be to alter the costs in the loss matrix. That way, one might be able to produce a more acceptable ratio of false negatives to false positives.

However, recall that the risk associated with a node is scaled by the product of the prior probabilities and the entries in the loss matrix. To see the consequences of this, suppose there exist a $\tilde{\pi}$ and an $\tilde{L}$ so that

$$\tilde{\pi}_i \tilde{L}(i,j) = \pi_i L(i,j). \tag{3.10}$$

Then the risk associated with that node are unchanged, and it does not matter what the particular values of $\tilde{\pi}$ and $\tilde{L}$ happen to be as long as the equality holds. This opens the door for lots of possibilities. If one just thinks of the right-hand side as the weight given to the classification errors for class i in a given node, and if more or less weight is desired, one can alter either the priors or the costs or both. In practice, it is less work to alter one of them, and the choice can depend on how the software is written. In the binary response case, if one wanted to alter the weights by altering just the prior distribution to π_i^*, one would use

$$\tilde{\pi}_i^* = \frac{\pi_i L_i^*}{\pi_i L_i^* + \pi_j L_j^*}. \tag{3.11}$$

The index i would take on one value for the no misconduct class (e.g., 1) and another value for the misconduct class (e.g., 2). The values of π_i would be the probabilities associated with the empirical prior distribution. The values of L_i^* would be the new costs. Note that because of the normalization, all that matters in the loss matrix is relative costs. Thus, one just has to know, for example, that the cost of one kind of classification error is three times the cost of another kind of classification error, not their actual costs.

Let's try an example so that the reasoning is clear. Suppose for the prison data one were to let the data determine everything. Then, the empirical prior distribution is about .8 for no misconduct and about .2 for misconduct. The cost of a false negative or a false positive is 1.0.

Suppose we now wanted the cost of a false negative to be twice the cost of a false positive: the 1 to 1 ratio would be 1 to 2. For no misconduct, we let $\pi_1 \times 1.0 = .80 \times 1.0 = .80$. For misconduct, we let $\pi_2 \times 1.0 = .20 \times 2.0 = .40$.

But then we need to normalize these values so that as probabilities they sum to 1.0. Normalizing π_1^*, we compute $(4/5)/(4/5+2/5) = .67$. Normalizing π_2^*, we compute $(2/5)/(4/5 + 2/5) = .33$. So a 1 to 2 cost ratio for a false positive to a false negative can be obtained using for the prior distribution .67 and .33. There is no need to change the values in $L(i, j)$, which in effect, still have diagonal cost elements $L(i \neq j) = 1$. Finally, you can get the same results by using Equation 3.11 with $\pi_1 L_1^* = .80 \times 1.0$ and $\pi_2 L_2^* = .20 \times 2.0$.

It would also be handy if analogous procedures were available for categorical response variables with more than two response categories. However, with more than two response categories, there is likely to be more information in the loss matrix than can be properly captured by a prior distribution. More specifically, for any given observation, the cost of all misclassifications must be the same for there to be a prior distribution that can properly represent the costs of classification errors.

For example, suppose there were three inmate misconduct categories: no misconduct, rule violations, and activities that would be crimes if committed outside of prison. Then, if an inmate actually had no incidents of reported misconduct, the costs of incorrectly placing him or her in either the rule violation category or the crime category would have to be the same. In practice, it is rare that such constraints on the loss matrix would be appropriate. As a result, relative costs would have to be introduced directly using the loss matrix. This presents no problems when the software permits such input. But it is common for there to be no allowance for a loss matrix, especially for the more sophisticated forms of statistical learning to be considered in the next two chapters. Fortunately, there are then other options, many of which are rather clever. We consider these later.

To summarize, when the CART solution is determined solely by the data, the prior distribution is empirically determined, and the costs in the loss matrix of all classification errors are the same. Costs are being assigned even if the data analyst makes no conscious decision about them. Should the balance of false negatives to false positives that results be unsatisfactory, that balance can be changed. Either the costs in the loss matrix can be directly altered, leaving the prior distribution to be empirically determined, or the prior distribution can be altered leaving the default costs untouched. Much of the software currently available makes it easier to change the prior in the binary response case. When there are more than two response categories, it will usually be easier in practice to change the costs in the loss matrix directly.

3.6 Pruning

With the discussion of costs behind us, we can now return to the problem of overly complex trees and what can be done. Recall that setting a minimum

sample size for each terminal node is one strategy. A second strategy to constrain the size of the tree is called "pruning." The pruning process removes undesirable branches by combining nodes that do not reduce heterogeneity sufficiently for the extra complexity added. The process starts at the terminal nodes and works back up the tree until all of the remaining nodes are satisfactory.

Of late, pruning has not gotten a lot of attention. The problem that pruning addresses is very real. But, as CART has become superceded, pruning has become less salient. Consequently, the discussion of pruning is relatively short. The main objective is to highlight some important issues raised in the previous chapter that figure significantly in the pages ahead.

For a tree T, recall that the overall risk is

$$R(T) = \sum_{j=1}^{K} P(A_j)R(A_j). \tag{3.12}$$

This is the sum over all terminal nodes of the risk associated with each node times the probability of cases falling in that node. It might seem that a reasonable pruning strategy would be to simply minimize Equation 3.12. What could be better than that? Unfortunately, that would leave a saturated tree untouched. CART would construct enough terminal nodes so that all were homogeneous, even if that meant one node for each observation. With all terminal nodes homogeneous, the risk associated with each would be zero. The result would be unstable nodes, serious overfitting of the data, and far too much detail to usefully interpret.

The solution is much like what was seen in the previous chapter. A penalty is introduced for complexity, and we are back into the bias–variance tradeoff. With larger trees, there will be fewer classification errors, implying less bias. But larger trees will have terminal nodes with fewer cases in each, which implies greater instability and hence, greater variance. The trick is to find a sensible balance.

To take complexity into account in CART, a popular solution is to define an objective function, called "cost complexity," for pruning that includes an explicit penalty for complexity. The penalty is not based on the number of parameters, as in conventional regression, or on the effective degrees of freedom used, as in smoothing. For CART, the penalty is a function of the number of terminal nodes. More precisely, we try to minimize

$$R_\alpha(T) = R(T) + \alpha|\tilde{T}|. \tag{3.13}$$

R_α has two parts, the total costs of the classification errors for the tree as a whole, and a penalty for complexity. For the latter, $\alpha \geq 0$ is the complexity parameter playing much the same role as λ in regression smoothers. In place of the effective degrees of freedom, $|\tilde{T}|$ is the number of terminal nodes in tree T.

The value of α quantifies the penalty for each additional terminal node. The larger the value of α, the heavier is the penalty for complexity. When $\alpha = 0$, there is no penalty and a saturated tree results. So, α is the means by which the size of the tree can be determined.

Breiman et al. (1984: Section 3.3) prove that for any value of the complexity parameter α, there is a unique smallest subtree of T that minimizes cost complexity. Thus, there cannot be two subtrees of the same size with the same cost complexity. Given α, there is a unique solution.

In many CART implementations, there are ways the software can select a reasonable value for α or for parameters that play the same role (Zhang and Singer, 1999: Section 4.2.3). These defaults are often a good place to start, but will commonly lead to results that are unsatisfactory. The tree selected may make a tradeoff between the variance and the bias that is undesirable for the particular analysis being undertaken. For example, there may be far too much detail to be usefully interpreted. Moreover, overfitting or measurement error can produce trees that make very little subject matter sense.

Alternatively, one can specify by trial and error a value of α that leads to terminal nodes, each with a sufficient number of cases and that can be sensibly interpreted. Interpretation will depend on both the number of terminal nodes and the kinds of cases that fall in each, so a substantial number of different tree models may need to be examined.

In practice, whether one determines tree complexity by using α (or some other complexity parameter), or an explicit argument to the CART procedure determining the minimum terminal node sample size, seems to make little difference. The goal is to construct a useful classification tree. How exactly that is accomplished is less important as long as the steps undertaken and the various results evaluated are recorded so that the work can be replicated.

The major risk from examining a larger number of tree models leads to overfitting. Overfitting is already a potential problem in CART and drawing on information from many trees can only make matters worse. We turn to overfitting and useful responses to it in the next chapter. Suffice it to say, it is always good to have a training dataset and a test dataset.

3.6.1 Impurity Versus $R_\alpha(T)$

At this point, one might wonder why CART does not use Equation 3.13 from the start when a tree is built, instead of some measure of node impurity. $R_\alpha(T)$ would seem to have built in all of the end-user needs very directly.

The rationale for not using a function of classification errors as a fitting criterion is discussed in Breiman et al. (1984: Section 4.1). As a technical matter, there can be at any given node, no single best split. But perhaps a more important reason is that less satisfactory trees can result. Consider two splits. For the first, there are two nodes that are about equally heterogeneous. For the second, one node is far more heterogeneous than the other. Suppose the two splits reduce impurity about the same. Yet, minimizing some function

of classification errors could lead to the first split being chosen even though the second split was preferable. For the second split, the less heterogeneous node might serve as a terminal node, or might readily lead to one. The more heterogeneous node would be more subject to further partitioning. For the first split, both nodes would likely be partitioned substantially further. In general, therefore, more complicated tree structures will follow.

There can be good subject matter reasons as well. Thinking back to the prison example, finding a single node that was filled almost completely with misconduct cases would be a very useful result, even if the other terminal nodes were quite heterogeneous. In contrast, having all of the terminal nodes with roughly the same proportions of misconduct and no misconduct cases, would not be very useful. Using node impurity as a splitting criterion will largely prevent this kind of problem.

3.7 Missing Data

Missing data are a problem for all statistical analyses, and CART is no exception. Unfortunately, missing data are all too common and they create, broadly stated, the same kind of difficulties they create for conventional linear regression. There is the loss of statistical power with the reduction in sample size and real likelihood of bias insofar as the observations lost are not effectively a random sample of the total.

There is one and only one ironclad solution to missing data regardless of the form of data analysis: don't have any. The message is that it pays to invest heavily in the data collection so that missing data do not materialize or are very rare. There are alternatives to be sure, but all are risky.

A general discussion of missing data is beyond the scope of this book, and excellent treatments are easily found (Little and Rubin, 2002). But it is important to consider how missing data can affect CART and what some of the more common responses are.

If the data are really missing "completely at random," the only loss is statistical power. By "missing completely at random" one means that the mechanism by which the data are lost is equivalent to simple random sampling. And if the number of cases lost is not large, the reduction in power is not likely to matter much. It cannot be overemphasized, however, that the burden is on the researcher to make a convincing argument that the data are missing completely at random. Proceeding simply "as if" this were true is a ruse. The results are then conditional upon the missing completely at random assumption, and may be of little interest unless the credibility of that assumption can be determined.

A fallback position is that the data are missing "conditionally at random." One can subset the data based on the values of observable variables so that for each such subset, the data are missing completely at random. By "missing conditionally at random," one means that the mechanism by which the data

are lost is equivalent to stratified random sampling. If this assumption is correct, at least to a reasonable approximation, the analysis can be conducted separately for each of the subsets and the results pooled. But again, assumptions made about the manner in which the data are missing must be argued convincingly.

If either of these assumptions can be justified, it will sometimes useful to impute the values of the missing data. It is rarely sensible to impute missing values for the response variable. One would usually exploit information in the predictors and in so doing, the relationship between the response and the predictors can be fundamentally altered; one builds in a new relationship between Y and X. But sometimes it can be helpful to impute missing data for predictors.

For example, suppose age is a predictor. Then, one might compute the mean value of age over all of the data available and use that value of age when age is missing. If age is related to other predictors, one can use conditional means for the missing data (i.e. conditioning on the values of those predictors) as the imputed values for the missing data. For example, if age is related to education, one can impute the value of age based on whether a person has a college degree. There would be one imputed value for age for those with no college degree and another imputed value for age for those with a college degree.

The key problem with such imputation procedures is when the data are ultimately analyzed, using the real data and the imputed data, the statistical procedures cannot tell which is which and necessarily treat all of the observations alike. At the very least, therefore, it is likely that estimates of the uncertainty in the results will be wrong.

In particular, the imputed values come with sampling error, and this source of uncertainty will be overlooked. Another difficulty is that the imputed values, which are just fitted values, will have less variability than the original variable itself. Consequently, the data analyzed will be too homogeneous. Together, these two features of imputed values can seriously undermine statistical inference.

There are a number of very interesting procedures that attempt to get the statistical inference right when some of the data are imputed. They need not trouble us here. We will focus on how missing data can be addressed within CART.

3.7.1 Missing Data with CART

If data are missing for the response variable, the only viable strategy is "listwise deletion." Observations with missing data on the response variable are dropped totally from the analysis. If the data are missing completely at random, the main loss is statistical power. If not, bias of unknown size and direction can be introduced.

When the data are missing for one or more predictors, there are more options. Listwise deletion remains a possible choice, especially if there are not a lot of missing data (e.g., less than 5% of the total number of observations) Listwise deletion is not fancy, but it is also easy to implement and understand.

A second option is to impute the data outside of CART itself. To take a simple illustration, one might employ conventional regression in which for the complete data a predictor with the missing data is regressed on other predictors with which it is likely to be related. The resulting regression equation can then be used to impute what the missing values might be.

For example, suppose that for employed individuals there are some missing data for income. But income is strongly related to education, age, and occupation. For the observations with no missing data, income is regressed on education, age, and occupation. Then, for the observations that have missing income data, the values for the three predictors are inserted into the estimated regression equation. Predicted values are computed, which are used to fill in the holes in the income data.

However, even if reasonably unbiased estimates can be constructed, this strategy ignores the reduced variability of the predicted values and treats the imputed values as fixed. One response is to impute several values for each observation drawing at random, in effect, from the conditional distributions implied by the regression equation. It is then possible to get a better handle on the uncertainty associated with the imputed values. Little and Rubin (2002) is the canonical reference. An application to CART can be found in a PhD dissertation by He (2006).

A third option is to address the missing data problems for predictors within CART itself. There are a number of ways this might be done. We consider here one of the better approaches, and the one available with rpart() in R.

The first place where missing data will matter is when a split is chosen. Recall that

$$\Delta I(s, A) = I(A) - p(A_L)I(A_L) - p(A_R)I(A_R), \tag{3.14}$$

where $I(A)$ is the value of the parent impurity, $p(A_R)$ is the probability of a case falling in the right daughter node, $p(A_L)$ is the probability of a case falling in the left daughter node, $I(A_R)$ is the impurity of the right daughter node, and $I(A_L)$ is the impurity of the left daughter node. CART tries to find the predictor and the split for which $\Delta I(s, A)$ is as large as possible.

Consider the leading term on the right-hand side. One can calculate its value without any predictors and so, there are no missing values to worry about. However, to construct the two daughter nodes, predictors are required. Each predictor is evaluated as usual, but using only the predictor values that are not missing. That is, $I(A_R)$ and $I(A_L)$ are computed for each optimal split for each predictor using only the data available. The associated probabilities $p(A_R)$ and $p(A_L)$ are re-estimated for each predictor based on the data actually present. This approach is undertaken with the equivalent of pairwise

deletion; calculations are undertaken with the complete data available for that operation alone.

But determining the split is only half the story. Now, observations have to be assigned to one of the two daughter nodes. How can this be done if the predictor values needed are missing? CART employs a sort of "CART-lite" to impute those missing values by exploiting "surrogate variables."

Suppose there are ten other predictors $x_1 - x_{10}$ that are to be included in the CART analysis, and suppose there are missing observations for x_1 only, which happens to be the predictor chosen to define the split. The split necessarily defines two categories for x_1.

The predictor x_1 now becomes a binary response variable with the two classes determined by the split. CART is applied with x_1 as the response and $x_2 - x_{10}$ as potential splitting variables. Only one partitioning is allowed; a full tree is not constructed. The nine predictors are then ranked by the proportion of cases in x_1 that are misclassified. Predictors that do not do substantially better than the marginal distribution of x_1 are dropped from further consideration.

The variable with the lowest classification error for x_1 is used in place of x_1 to assign cases to one of the two daughter nodes when the observations on x_1 are missing. That is, the predicted classes for x_1 are used when the actual classes for x_1 are missing. If there are missing data for the highest ranked predictor of x_1, the second highest predictor is used instead. If there are missing data for the second highest ranked predictor of x_1, the third highest ranked predictor is used instead, and so on. If each of the variables $x_2 - x_{10}$ have missing data, the marginal distribution of the x_1 split is used. For example, if the split is defined so that $x_1 < c$ sends observations to the left and $x_1 \geq c$ sends cases to the right, cases with data missing on x_1, which have no surrogate to use instead, are placed along with the majority of cases.

This is a reasonable, but ad hoc, response to missing data. One can think of alternatives that might perform better. But the greatest risk is that if there are lots of missing data and the surrogate variables are used, the correspondence between the results and the data, had they been complete, can become very tenuous. In practice, the data will rarely be missing "completely at random" or even "conditionally at random." Then, if too many observations are manufactured, rather than collected, a new kind of generalization error will be introduced. The irony is that imputation can fail just when it is needed the most.

Perhaps the best advice is to avoid the use of surrogate variables. The temptations for misuse are great, and there is no clear missing data threshold beyond which imputation is likely to produce misleading results. Imputation of the missing values for the predictors will usually be a software option, not a requirement. (But check what the default is.)

Alternatively, one should at least look carefully at the results with and without using surrogates. Results that are substantially different need to be reported to whomever is going to use the findings. There may then be a way to

choose on subject matter grounds which results are more sensible. Sometimes neither set will be sensible, which takes us back to where we began. Great efforts should be made to avoid missing data.

There is one situation, however, in which using surrogate variables is probably necessary. As becomes more clear in the pages ahead, a number of statistical difficulties can follow when the response variable is highly skewed. The danger with missing data is that the skewing can be made worse. One may then have little choice but to impute the missing data.

3.8 Statistical Inference with CART

An initial question for statistical inference is what features of a CART model might be of interest. To date, attention has centered on an overall assessment of the CART model, and by implication, some measure of fit quality. The enterprise is model selection.

But just as with smoothers, a key issue that must be addressed before statistical inference with CART is considered is whether estimation is a reasonable activity to begin with. As before, there are three scenarios.

1. There is assumed to be a $f(X)$, and the data are a random sample from a well-defined population or a random realization from a well-defined stochastic process. Estimation is worth a good hard look and so are ways to represent uncertainty.
2. There is no $f(X)$ assumed, but the intent is to construct a best guess of the values of a set of conditional proportions in a population or as features of a stochastic process. Estimation is again in play, at least in principle. Ways to represent uncertainty are as well.
3. The sole goal is description of the data on hand. Estimation is not relevant even in principle.

We begin with the first case: there is a $f(X)$ and the data were generated in a manner required for statistical inference. For this first case, statistical inference can be problematic in CART. Perhaps the most obvious reason is the need to assume negligible bias in the results. Mentioned earlier in this chapter was the likely bias in the estimated proportions about which more is said later. More important in practice is the absence of some key predictors and/or substantial measurement error in those that are available.

There are more subtle difficulties as well. For binary response variables, it might seem natural to use the deviance (or a good approximation) as a measure of fit quality and then compare two CART models using a likelihood ratio test. Recall, two models are required, the smaller one nested within the larger one. The smaller model is the model under the null hypothesis. For both, the deviance is computed. Then, the difference in the two deviances has a χ^2 distribution with degrees of freedom equal to the degrees of freedom for the smaller model subtracted from the degrees of freedom for the larger

model. When quantitative response variables for CART are discussed below, an F-test is the natural procedure. Although one can compute the deviance for both models, it would be rare to find one CART model nested within another. Therefore the very logic of the comparison is undermined, and it would also be possible to have two models with different deviances that used the same degrees of freedom.

But what are the degrees of freedom for a CART model? For a parametric regression with p regression coefficients, $p + 1$ is the number of degrees of freedom used up. The degrees of freedom remaining is $N - (p+1)$. For CART, sometimes the number of terminal nodes plus one is assumed to play the same role as $p + 1$ for a parametric regression. This is because the CART results can be re-expressed as a regression model with an indicator variable for each terminal node. However, using the number of terminal nodes to arrive at the degrees of freedom lost fails to take into account all of the searching done as the tree is grown. Many more degrees of freedom are actually used up. Moreover, it is not clear how to make appropriate adjustments, although some simulation results suggest the true degrees of freedom used up is between 5 and 10 degrees of freedom per split of the data (Hastie et al., 2001: 297). An additional complication is that often many trees are examined as the output is "tuned" to the particular needs of the analyst. Whatever the number of degrees of freedom lost when the tree is grown, they will not include the degrees of freedom lost growing prior trees.

Confidence intervals suffer from similar difficulties. The negligible-bias assumption remains an important hurdle, and one needs a value for the degrees of freedom in order to estimate of any standard errors. Thus, even obtaining an appropriate estimate of the point-by-point standard error of the fitted values is tricky.

The degrees of freedom problems spill over into the fit statistics that can be used instead of tests for model evaluation and selection. Among the most common goodness-of-fit measures used with CART are the AIC and the BIC. For a binary response variable,

$$\text{AIC} = D + 2p; \tag{3.15}$$

and

$$\text{BIC} = D + \log(n)\, p, \tag{3.16}$$

where D is the deviance, n is the sample size, and p is the degrees of freedom used up in the calculations. Unfortunately, we are once again stuck with the problem of finding a credible value for p. The prospects are really no better under the second scenario when no $f(X)$ is assumed but there is interest in one or more several conditional proportions of corresponding terminal nodes. The nodes are just subsets of the data defined by the fixed values of predictors. But, there remains the problem of how one defines the degrees of freedom and the likely bias in the estimated proportion noted earlier.

Recently, Hothorn and his colleagues (2006) have suggested another approach to tree construction that builds on and then provides some hypothesis tests. A number of statisticians have observed that other things being equal, predictors with more possible breaks have a greater chance of being selected as a partitioning variable. As a result, there is bias built into the tree structure and implicitly, the results summarized in the terminal nodes. Hothorn and his colleagues suggest a tree-building procedure that has much the same look and feel of conventional stepwise regression.

1. With a null hypothesis that each predictor is unrelated to the response, conduct a global hypothesis test.
2. If the test is not rejected, stop.
3. If the test is rejected, select as the splitting variable the predictor having the strongest relationship with the response.
4. Choose the best split using the selected predictor.
5. Repeat Steps 1–4 until no further splits are indicated.

All of the tests are based on permutation distributions in which the response is shuffled. Each predictor is subjected to a permutation test under the null hypothesis of no association. An overall p-value is also computed adjusting for multiple tests (e.g., a Bonferroni correction). If the global null hypothesis is rejected, the predictor with the smallest p-value is chosen. Then, the split can be determined as usual. The same process is applied for each subsequent partitioning of the data.

There is excellent software in R implementing these procedures. Because recursive partitioning can be an intermediate step in other statistical learning procedures, there are also interesting extensions built into the software. The procedure can be found in the library *party*.

Hothorn and his colleagues argue that selecting predictors in this fashion leads to an unbiased recursive partitioning of the data. However, some caution is warranted. First, this approach assumes, as before, that there is a true $f(X)$ one is trying to estimate with $\hat{f}(X)$, that all of the predictors in X are in the dataset, and that all are well measured.

Second, the tests also take the predictors as fixed. Sampling variation comes only from the response variable. The motivating thought experiment envisions all possible assignments of the response variable values to existing cases within a given marginal distribution of the response variable. There is, therefore, an issue of how well the permutation thought experiment corresponds to the manner in which the data were actually generated and whether one's inferences are really to be limited only to the x-values that were realized.

Third, the results depend on the sample size; other things being equal, larger samples will lead to larger trees. Larger trees will usually perform quite differently from smaller trees, as we have seen. And one does not normally seek to determine the correct model conditional on the sample size. That is, it is at least unconventional to proceed as if there were many correct models, one for each sample size.

Fourth, there will still be bias in the estimates coming from terminal nodes insofar as there is remaining heterogeneity in predictor values associated with the response variable. This too was addressed earlier.

But more generally, for a very large number of applications, these sorts of concerns are moot. The third motivating scenario is likely to be the operational one in practice. There is no $f(X)$, or no population or stochastic process, or no appropriate data-generation mechanism, or the data may be of insufficient quality (e.g., key predictors are missing). Then, the goal is far more likely to be description than estimation. There are only, then, descriptions that are more or less useful. Furthermore, there is no requirement that a single description be chosen. Different descriptions may highlight different, but instructive, features of the data. Then, it can be appropriate to report more than one set of results.

3.9 Classification Versus Forecasting

In most of the discussion of CART so far, and all of the examples, the emphasis has been on fitting the data on hand. With categorical response variables, this has been a classification exercise. A key objective of the analysis is to minimize some aggregate measure of fit that depends substantially on classification errors and their costs. References to forecasting have typically been indirect and/or brief.

As a technical matter, however, the step to true forecasting is relatively easy. One applies the results from the data analyzed to new data not used in the fitting process. A key difference is that for the data used to build the tree, both the predictor values and response values are known. For the data to be used in the forecasting exercise, only the predictor values are known. The key assumption is that the relationships between the predictors and the response for the data analyzed would be the same for the new data, within chance error, were the values of the response variable known.

Sometimes forecasts into the future are desired. Sometimes forecasts into the past as desired (often called "backcasting"). And for some forecasts, time plays no role. A CART analysis undertaken on inmates in one prison, for example, may be used to forecast the misconduct of inmates in another prison. And to confuse things a bit more, we saw in the last section that when missing data are imputed, the enterprise looks like forecasting. In each of these applications, the key idea is that some or all of the values of a variable are unknown, and there is a need for some "best-guess" values. However, although there are some important parallels with forecasting, imputation does not involve a second dataset.

In CART, the classes assigned to terminal nodes are used as the best-guess values. In CART-speak, one "drops" new cases "down" the classification tree. Each case will "land" in one (and only one) terminal node. The earlier classification of each terminal node determines the prediction for all of the

new cases when they arrive. Thus, if a given terminal node is classified as class "1," all of the observations that come to rest in that terminal node are classified as a "1," and the class represented by that "1" is the forecasted class (e.g., misconduct).

But how is the quality of those forecasts determined if the response is not yet known? There are four steps.

1. Build a classification tree as usual using a training dataset taking the costs of classification errors into account.
2. Forecast using a test dataset in which the outcomes are known.
3. Determine forecasting skill from an analysis of the forecasting errors, perhaps using the percentage of cases incorrectly forecast conditioning on the truth. Usually, this is presented in the form of a confusion table.
4. Assume that the forecasting skill demonstrated with the test data applies to new data for which forecasts are desired. This assumption can be supported if the new data are a random sample from the same population as the training and test data.

Forecasting is an important application for CART and all of the procedures discussed in the pages ahead. Moreover, the difference between classification and forecasting, often confused when the term "prediction" is used for both, figures significantly in the next chapter. We show that forecasting errors, rather than classification errors, are better tools for refining algorithmic models.

3.10 Varying the Prior, Costs, and the Complexity Penalty

Figure 3.4 shows again the tree diagram for the CART analysis of inmate misconduct. The tree diagram has been simplified a bit anticipating the need to show more complicated structures shortly. Recall that the empirical distribution of the response variable was used as the prior distribution, and the costs were assumed to be the same for false negatives and false positives. For the earlier figure, the number of terminal nodes was constrained by explicitly setting the minimum terminal node sample size. Now, for reasons that are apparent shortly, we get to the very same place by setting the value of the penalty for complexity instead.

Recall also that a confusion table was presented as part of the earlier analysis. It is now reproduced as Table 3.4. One questionable feature of the table was that there were about eight false negatives for each false positive, implying that false positives were far more costly than false negatives. At that time, there was no justification given for these or any other ratio of relative costs.

Conversations with prison officials indicated that from their point of view, false negatives were much worse than false positives. Failing to anticipate

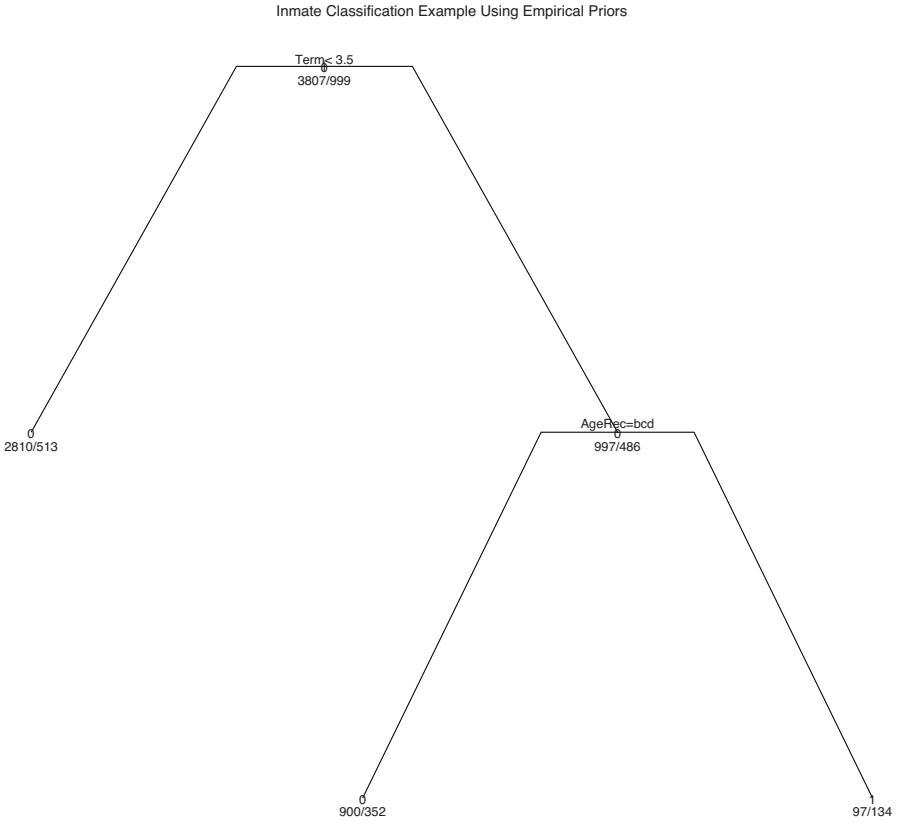


Fig. 3.4. Inmate misconduct example with empirical priors.

	Predict No Misconduct	Predict Misconduct	Model Error
No Misconduct	3710	97	.03
Misconduct	865	134	.88
Use Error	.19	.42	Overall Error = .20

Table 3.4. CART confusion table for forecasting inmate misconduct.

inmate misconduct, which could involve fatal violence, was of far greater concern than incorrectly labeling an inmate as high risk. When pushed, a prison official said that the cost of a false negative was at least five times greater than the cost of a false positive. Hence, the earlier analysis got things upside down.

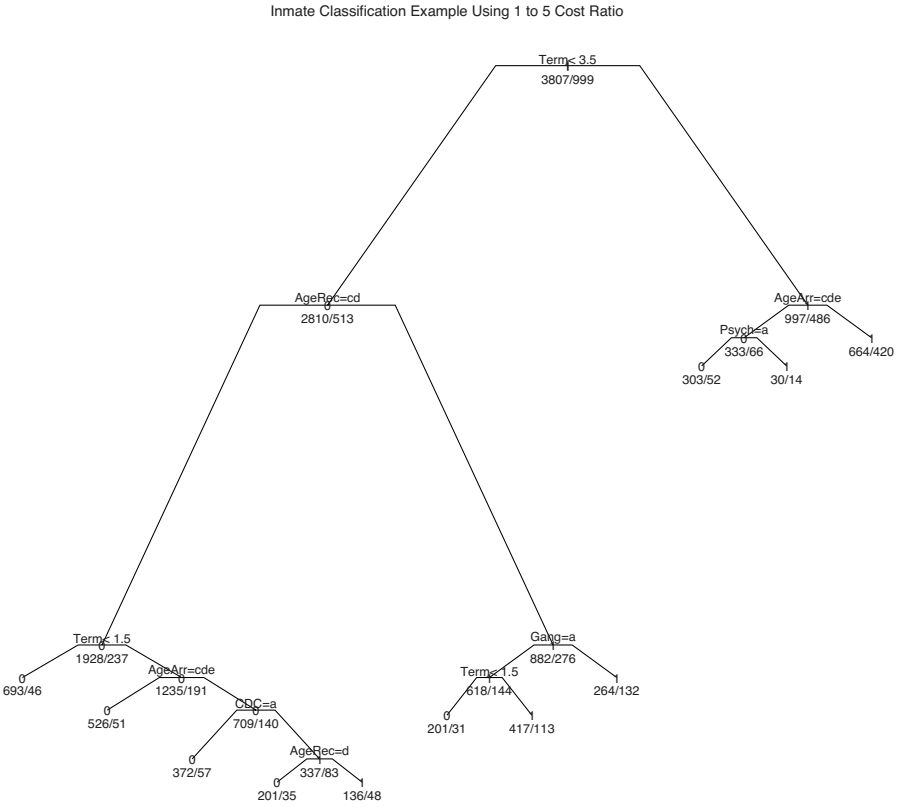


Fig. 3.5. Inmate misconduct example with one to five cost ratio.

Figure 3.5 shows the tree diagram that results when the same value for the complexity penalty is used, but the empirical prior is replaced by a new prior representing the one to five cost ratio for false positives to false neg-

atives elicited from prison officials. The new prior was calculated using the procedures described earlier.

Clearly, the results have changed substantially. Increasing dramatically the costs of false negatives relative to false positives (and it is only the relative costs that matter) leads to a very different result. There are many more terminal nodes reflecting the impact of several more variables. These now include:

1. Gang activity (Gang) with a = gang activity and b = no gang activity.
2. Age at first arrest (AgeArr) with a = 0–17, b = 18–21, c = 22–29, d = 30–35, and e = older than 35.
3. Age at arrival at the reception center (AgeRec) with a = 16–20, b = 21–26, c = 27–35, and d = 36 older than 35.
4. Mental illness (Psych) with a = ill and b = not ill.
5. Previously served time under the state's Department of Corrections (CDC) with a = served time and b = did not serve time.
6. Sentence length (Term) in years.

A larger number of predictors were included in the analysis. Figure 3.5 shows the predictors that CART selected.

We do not interpret Figure 3.5. It is quite complicated and would take us far afield. We consider a more simple CART analysis shortly. For now, the main point is that from Figure 3.5 one can see that the labeling of a terminal node with a "1" or a "0" is not the result of a majority vote of the cases in each node. The vote now takes costs into account captured by the altered prior distribution of the response. This leads directly to Table 3.5.

First, examine the off-diagonal cells. The balance of false negatives to false positives has been reversed. The ratio of false negatives to false positives now reflects approximately the one to five cost ratio of false negatives to false positives. The ratio is not exact because the cost ratio is but one input into the CART algorithm. CART is trying to respond to several features of the data and the analysis requested.

Second, the more desirable balance of false negatives to false positives seems to come with a price. Overall, there is an increase in the number of classification errors. The sum of the off-diagonal cells divided by the total number of cases is about .20 for Table 3.4 and about .37 for Table 3.5. The number of cases incorrectly classified has increased by 17%. So, by that yardstick we are doing substantially worse. This is a general result. Introducing any cost ratios other than one to one will increase the overall proportion of cases misclassified.

But the yardstick of correct classifications can be misleading. Once one abandons the assumption that the costs of false negatives and false positives are the same, the proportion of cases misclassified is not by itself responsive to the way the analysis has been undertaken. Classification errors are now being weighted to take differences in costs into account, but these weights are ignored when the proportion of cases misclassified is computed. All classification errors are being treated the same, even though they are not.

Third, a far more instructive way to consider how well CART has fit the data is to look at the classification errors conditional upon the actual outcome. This means focusing on the rows in Tables 3.4 and 3.5. Within a row, there is only one kind of classification error (false positives or false negatives), so the weighting problem disappears. One can then see that the proportion of misconduct cases misclassified has dropped from .88 to .27. At the same time, the proportion of no misconduct cases has increased from .03 to .40. A tradeoff between false negatives and false positives is generally to be expected and is plainly seen here.

A analogous tradeoff can be seen in the column proportions. When a case is assigned the no misconduct label, that label is now wrong for .11 of the observations. With equal costs, that proportion is .19. When a case is assigned the misconduct label, the proportion of cases labeled in error increases from .42 to .67.

To get some practical sense of what these changes mean, suppose 100 inmates are classified either as misconduct cases or no misconduct cases. For inmates given a no misconduct label, the number of inmates labeled in error drops from 19 to 11 when the equal costs assumption is replaced by the one to five costs assumption. Clearly, this would be a desirable result for prison administrators.

For inmates given a misconduct label, the number of inmates labeled in error increases from 42 to 67 when the equal costs assumption is replaced by the one to five costs assumption. For both of the assumptions about costs, far more errors are made when inmates are assigned to the misconduct class. And the one to five cost ratio makes things worse.

But, the increase in false positives is precisely the result mandated by the one to five cost ratio provided by prison administrators. The first analysis had too many false negatives relative to false positives. The second analysis was motivated by a need to correct this perceived imbalance. When the one to five cost ratio was determined, prison administrators were making a conscious decision to live with a greater number of false positives.

	Predict No Misconduct	Predict Misconduct	Model Error
No Misconduct	2296	1511	.40
Misconduct	272	727	.27
Use Error	.11	.67	Overall Error = .37

Table 3.5. CART confusion table for forecasting inmate misconduct using a one to five cost ratio.

The key point is this: although it is always desirable to have a small number of false negatives and false positives, a decision-maker may be better off with increases in either if relative costs of false negatives to false positives are more

accurately represented. More errors can actually lead to better decisions when costs are taken into account.

Figure 3.6 represents a third analysis of the same data. The one to five cost ratio is maintained, but a larger penalty is given for complexity. The intent is to simplify the tree so that only the most important and meaningful terminal nodes are included. At the same time, there is certainly nothing definitive about Figure 3.6. Another data analyst might quite properly construct a tree that was either more or less complicated.

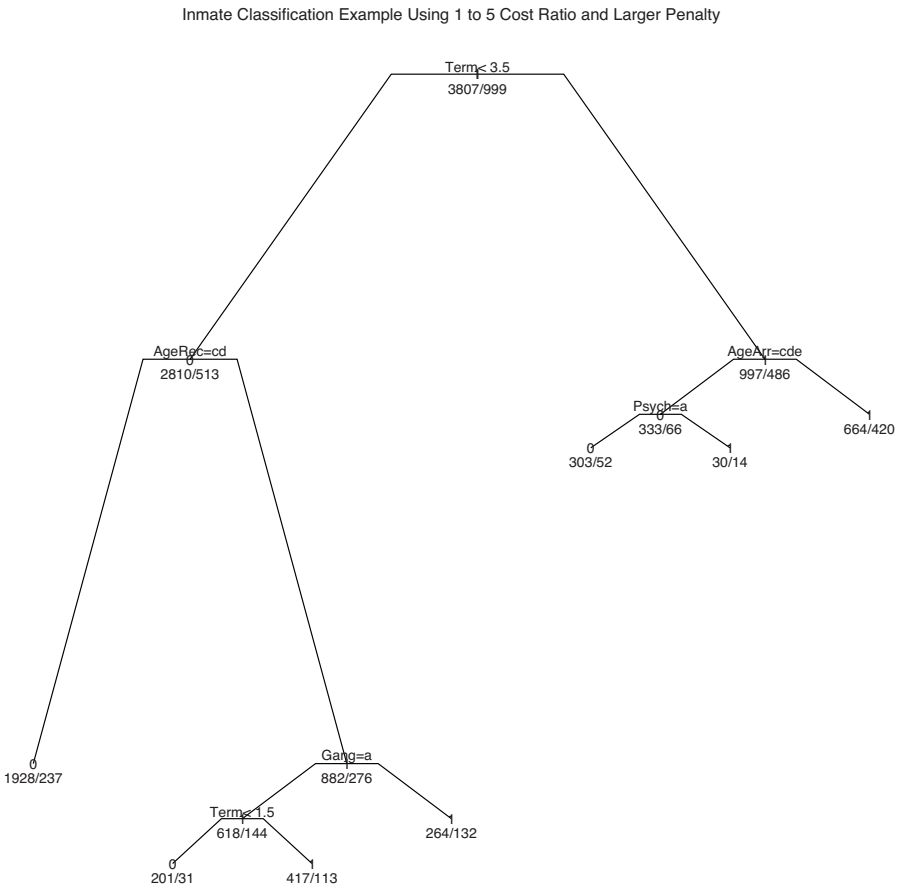


Fig. 3.6. Inmate misconduct example with one to five cost ratio and larger penalty.

Despite the use of weighted votes, the branches in the tree are interpreted exactly as before. We learn for instance, that individuals with sentences equal to 3.5 years or longer and under the age at 18 when first arrested are classified as prone to misconduct. And these inmates pose the highest risk. We also learn that individuals with sentences less than 3.5 years who are 27 years or older when they arrive at the prison reception center are not classified as prone to misconduct and pose the lowest risk. If one is interested in the proportion of inmates in each terminal node who were actually reported to have engaged in misconduct, that is available in the CART output, at least in the R implementation rpart().

The other combinations fall in between. Thus, (working down the tree) individuals who have been sentenced to less than 3.5 years, who are younger than 27 years, who have a history of gang activity, and who are actually serving very short sentences of less than 18 months, are not classified as prone to misconduct. Having a very short sentence seems to trump youth and gang activity, which are normally useful predictors of misconduct in prison. Note that if all of these factors are the same except that the sentence is between 18 months and 3.5 years, the inmate is classified as prone to misconduct.

The confusion table is not presented. Overall, there is an increase in the errors, as one would expect from a less complex classification tree. However, the smaller tree may be more stable, a topic to which we return later. Because the one to five cost ratio is maintained, the various tradeoffs associated with false negatives and false positives are essentially unchanged.

3.11 An Example with Three Response Categories

There is no formal problem in extending CART to three or more response variable categories. We return to the prison data once again and use the same predictors as before. But this time, there are three categories to the response: no misconduct, minor misconduct, and serious misconduct. These are coded as "0," "1," and "2," respectively. About 78% of the cases have no reported misconduct, about 20% have minor reported misconduct, and about 2% have serious reported misconduct. As required, these are mutually exclusive and exhaustive categories. The 2% represents very rare cases, which creates some special difficulties we address in more depth later. For now we ignore the problem.

The three response classes can be viewed as qualitatively different. In particular, most minor infractions are violations of prison rules and would, by and large, not be considered crimes if committed outside of prison. Most of the serious misconduct represents acts that would be felonies anywhere: drug trafficking, sexual assault, robbery, homicide, and the like. On the other hand, if one thinks of prison misconduct as arrayed on some scale of seriousness, the three classes are perhaps ordered. However, just as in conventional multinomial regression, CART take no account of such ordering. Whatever

information is contained in the ranks is ignored. In short, the three classes of misconduct are treated by CART as unordered categories whatever the truth may be.

The first job in the data analysis is to specify a loss matrix. Recall that when there are more than two response categories, costs cannot usually be captured in a prior distribution. So, the empirical distribution is taken as the prior distribution and costs are introduced directly with the loss matrix.

Table 3.6 shows a loss matrix containing costs roughly consistent with information provided by corrections officials. They were especially concerned about inmates who were a serious safety threat but not classified as such. In particular, if an inmate actually had an incident of reported serious misconduct (e.g., assault on a guard) and was classified as having no misconduct at all, the cost assigned is 20. If an inmate actually had an incident of reported minor misconduct (e.g., failure to report to a job assignment) and was classified as having no misconduct at all, the assigned cost is 10. The former error has twice the cost of the latter error, which in turn is substantially higher than any of the other costs. The costs of incorrectly classifying an inmate as misconduct free are relatively small.

	None Predicted	Minor Predicted	Serious Predicted
No Misconduct	0	2	5
Minor Misconduct	10	0	3
Serious Misconduct	20	10	0

Table 3.6. Costs of classification errors for three misconduct categories.

Figure 3.7 shows the resulting classification tree. It can be read just as when there were two response categories, but now the numbers associated with each node are the counts (left to right) for no misconduct, minor misconduct, and serious misconduct. When the unequal costs are used, the classification assigned to each terminal node is not by a plurality of votes. Some votes, in effect, have more weight than others.

In this instance, the very high relative costs for misclassifying serious incidents of inmate misconduct dominate the results. Of the six terminal nodes, four are classified as a "2," the coded value for serious misconduct. The two other terminal nodes are all classified as a "0," the coded value for no misconduct. And none of the terminal nodes are classified as a "1," the value for minor misconduct. The loss matrix led to only two kinds of classification: no misconduct and serious misconduct.

It is important to stress that these results are neither "right" nor "wrong." Their usefulness would depend in part on whether in retrospect, prison officials liked the values in the loss matrix that in this instance produced just two classifications: good inmates and bad inmates. Had the costs of failing

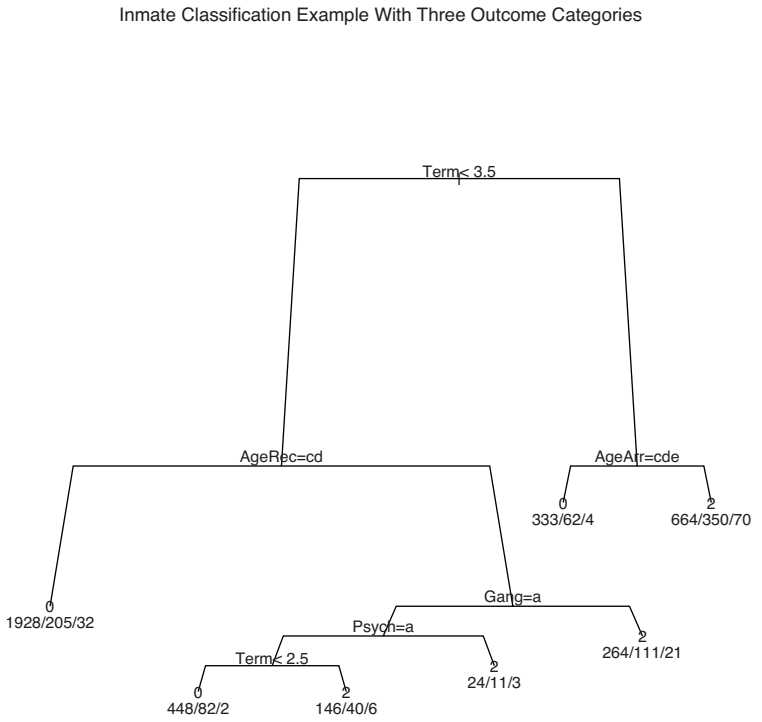


Fig. 3.7. Inmate misconduct with three response categories.

to properly classify the serious misconduct inmates been reduced relative to the other costs of misclassification, a very different set of results would have materialized, including the appearance of all three classification categories for the terminal nodes.

The roles that the predictors play are also shown, but sometimes imply complicated and even counter intuitive interpretations. One reason may be that distinctions are now being made among three kinds of inmate behavior, not two. Moreover, what some may think is an ordered set of classes is being treated as categorical only. Different patterns of association can result. Another possibility is that some of the breaks represent unstable distinctions that should not be taken seriously, a point discussed in more depth shortly.

If corrections officials were directly involved in discussions of how such results might be used, it is likely that several different patterns of costs and several different values for the complexity penalty would be applied. Trees requiring complicated and counter-intuitive interpretations might then be eliminated. For example, even if the costs in Table 3.6 were maintained, a tree pruned back to the first two breaks might well be preferable.

	Predict None	Predict Minor	Predict Serious	Model Error
No Misconduct	2709	0	1098	.29
Minor Misconduct	349	0	512	1.0
Serious Misconduct	38	0	100	.28
Use Error	.12	Undefined	.94	Overall Error = .42

Table 3.7. CART confusion table for forecasting three categories of inmate misconduct.

Another factor that would have to be taken into account is the confusion table. Table 3.7 shows the confusion table from Figure 3.7. The error percentages are computed combining the two sources of error in each row or column. Then row and column proportions are computed as before. Thus, Table 3.7 is interpreted in virtually the same manner as the binary outcome case.

Corrections officials would have to examine each row and column in Table 3.7 and decide if the numbers for the different kinds of false negatives and false positives were reasonable for their purposes. To take the most extreme example, no inmates are assigned to the minor misconduct class. It follows that all inmates who had engaged in minor misconduct are misclassified as being misconduct free or having engaged in serious misconduct. This would likely be unacceptable because in fact, minor misconduct is relatively common, can be quite disruptive, and can be a precursor to more serious incidents. In response, prison officials might favor altering the relative costs of classification errors for minor misconduct. For example, the cost of classifying cases of minor misconduct as if there were no misconduct might be increased.

There are perhaps two main messages from the analysis just summarized. First, there are in principle no logical or computational obstacles moving from two response categories to three or more response categories. But there can be problems from sparse data that will undermine a wide variety of statistical procedures, including CART. For example, some of the row proportions may be computed from very few observations. Important instabilities can result.

Second, moving from two response variable categories to three response variable categories complicates matters significantly. The CART algorithm is being implemented exactly as before. But, the loss matrix has more elements, the confusion table has more entries, and the tree diagram can be difficult to interpret. Compared to the binary case, these changes can be dramatic. For example, going from two to three response categories doubles the number of

elements in the loss matrix that need to be specified and doubles the kinds of classification error that need to be evaluated. Trial-and-error tuning that is likely to follow will usually require juggling many different parts of the output, which will be a challenge to data analysts and practitioners alike.

3.12 CART with Highly Skewed Response Distributions

Response variables that are highly skewed (often called "unbalanced") can create serious problems for any form of regression analysis. One problem can be sparse data, which can lead to unstable results or even an inability of the software to provide any results at all. Another problem is that the rare observations may have a disproportionate impact on the findings. Generalizations to the mass of the data can then be problematic. Yet another problem is that with highly skewed response variables, it can be very difficult to find predictors that are able to improve the overall fit. Thus, if the response is binary, and the marginal distribution is 95% 0s and 5% 1s, very accurate classifications can be determined from this information alone. If the 0 category is always assigned, the classifications will be correct 95% of the time without using any predictors whatsoever. It is difficult to do better than this.

There are hints in the material presented earlier of how these sorts of problems may sometimes be addressed. For example, if a prior distribution places heavy weight on the misclassification of rare cases, it is a bit like saying there are many more rare cases to be considered than the data indicate. And in fact, CART will behave as if this were so. We explore this matter in depth in the chapters to come and find that there are several other promising options. In the meantime, the message is that when the response variable is highly skewed, one must examine all CART output very carefully.

3.13 Some Cautions in Interpreting CART Results

Just as for any data analysis procedure, the output from CART always demands scrutiny before substantive conclusions are reached. There are commonly three kinds of potential problems: inappropriate response functions, unstable tree structures, and unstable classifications. All can produce results, which if taken at face value, risk serious interpretive errors.

3.13.1 Model Bias

If the goal of a CART analysis is to determine the $f(x)$, whether the function is part of a causal model or a feature of a conditional distribution, there is no guarantee that in a given sample CART will even come close. As noted more broadly in Chapter 1, there are no formal mathematical results indicating that CART will find the correct function from a given sample, even if all of

the requisite predictors are provided and even if these predictors are very well measured.

Indeed, there is good evidence, as noted earlier, that CART will tend to select predictors in a manner that introduces bias (Hothorn et al., 2006). Other things being equal, predictors with a larger number of distinct x-values are favored over predictors with a smaller number of distinct x-values when all possible splits are evaluated. In addition, the use of surrogate variables in response to missing data will further bias the selection of predictors.

Of at least equal concern is the matter of functional form. If $f(X)$ is smooth, each CART basis function, necessarily relying on indicator variables, will be incorrect. The hope is that the indicator variables will provide a useful $f(X)$ approximation. Typically, it is difficult to determine if the approximation is good, but as a formal matter, some bias is likely.

If the goal is to obtain estimates of the response conditional probabilities and classifications, there are related difficulties. If the functional form estimated is substantially in error, it likely the terminal node proportions and the classifications that follow will be substantially biased. And as discussed earlier, unless the terminal nodes are homogeneous, probability and classification estimates for individual cases will likely be biased.

In summary, even under the best of circumstances, unless the $f(X)$ is a step function, there will be biases in any CART estimates. The practical question is how serious the biases are likely to be. This is one important reason why CART has been largely superseded by procedures considered in later chapters.

3.13.2 Model Variance

CART partitions the data into more and more homogeneous subsets and then assigns a class label to each terminal node. The class that CART assigns to each case depends on the terminal node where that case comes to rest and the class label assigned to that node. There are several ways in which these steps can lead to unstable results.

The most obvious cause of instability is a small number of observations in any node. Then, a very few observations can send CART down one kind of branching structure rather than another. There can be a tipping effect in which a split relatively high up in the tree structure that depends on a few data points has cascading impacts on later splits. If those few observations are removed from the data, or if they are replaced with different observations, a different tree structure with different terminal nodes can follow.

Why might the new observations be different? They are likely to be different if there is a second random sample from the same population; random sampling error can make them different. They are also likely to be different if the predictor with which the split was constructed was measured with random error. This implies that a new realization of the data created by a new round of measurements can easily produce very different results. For example,

if SAT score is a predictor, the scores from the test taken in the junior year of high school will likely differ, at least a bit, from the scores taken in the senior year of high school at least because of noise alone.

Fortunately, small samples in particular nodes are easily spotted, and there are several good remedies within the usual CART software. One can, for example, increase the minimum number of observations in all nodes or increase the value of the complexity penalty. Larger node sizes will result, and the stability of the output will tend to increase.

A less apparent source of instability derives from terminal nodes that remain relatively heterogeneous. If the split in a given terminal node is near the 50–50 threshold, the movement of just a few cases across terminal nodes can change the classes assigned to each terminal node and dramatically alter the classes assigned to observations within them. Note that this is not a problem caused by a small number of observations in terminal nodes, although if there are few observations in terminal nodes, the instability caused by near-even proportions is made worse.

Instability resulting from heterogeneous terminal nodes is relatively easy to spot when the empirical distribution is used for the prior distribution and when the costs of false negatives and false positives are taken to be the same. For each terminal node, the numbers of observations in each response class can be easily inspected and compared. But when the classifications assigned to terminal nodes depend on more than the within-node counts, it is necessary to dig deeply in CART output to determine what is going on.

Even if one is able to conclude that heterogeneous terminal nodes are problematic, there is often little to be done. The cause lies in weak predictors that are unable to partition the data so that relatively homogeneous subsets result. And under such conditions, it may not be wise to take the tree structure or fitted values very seriously.

A related problem affects how a splitting variable is chosen. That variable may be the predictor from the preceding split, or another predictor. If the same variable, one has a step function representing nonlinear features of the relationship between that predictor and the response variable. If a new variable, one has an interaction effect representing a different kind of step nonlinearity. One predictor's relationship with the response variable depends on the value of the other predictor. The subject matter interpretations can also be very different. Yet, the choice between the two may be precarious, especially when predictors are highly correlated with each other.

Figure 3.8 shows possible partitions of the data when predictors x and z are highly correlated. In this illustration, CART constructs an initial partition with the solid vertical line. To the left of that line, B-values dominate. To the right of the line, A-values dominate. Both partitions are, therefore, more homogeneous than the dataset as a whole. The partitioning is a success.

But what should the next partition be? Consider a partition on the right side of Figure 3.8. There is clearly a cluster of all A-values in the upper right-

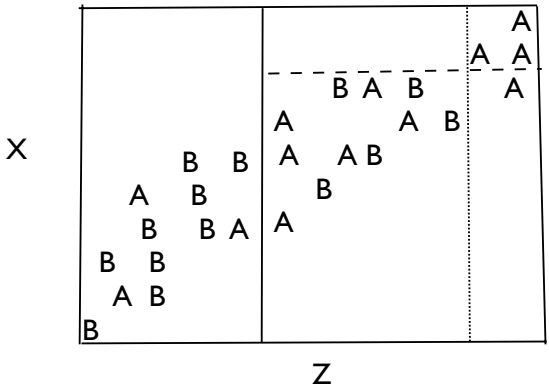


Fig. 3.8. High instability in CART.

hand corner. But because the two predictors are highly correlated, there are two ways to isolate them, which lead to very similar reductions in impurity.

The horizontal dashed line constructs a partition based on x. That partition isolates three of the four target values and implies an interaction effect. Given the previous split on z, the next split should be on x. An interaction effect results.

The other partitioning is accomplished using the vertical dotted line. This one uses z, and may be slightly preferred because now all four of the target values are isolated. As a result, a second step is introduced in the step function through which z is related to the response.

Both splits address nonlinearities in how the response is related to predictors. But one solution is a more complicated step function, and the other is an interaction effect. The choice between the two in this illustration depends on a single observation. With CART, there is always the possibility of instabilities of this sort. But just as in linear regression, potential instabilities are more likely to materialize if the predictors are highly correlated. Two or more predictors can more readily compete for the same partition of the data.

Figure 3.9 illustrates how a lower correlation between the two predictors can help. With a less clustered scatterplot, it is more difficult to find a small number of observations that both x and z can isolate. Notice that either of the competing partitions in Figure 3.8 now include many observations that fall into one partition but not the other. The two predictors are not competing

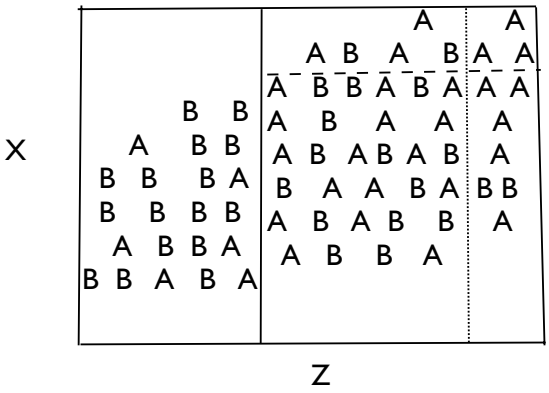


Fig. 3.9. Low instability in CART.

any longer for nearly the same set of observations. The choice between the two is now less likely to be made on a stray observation or two.

An unstable tree structure does not automatically mean that the assigned classes are unstable as well. In the case of two highly correlated predictors, for example, one can obtain rather different tree structures depending on which variable is chosen. And the different tree structure can certainly lead to different interpretations of how the response is related to the predictor. However, the classes assigned to observations, once they come to rest in a terminal node, may be much the same. A very different tree structure does not necessarily imply very different classifications. There may be two or more ways to get essentially the same classification results.

More generally, if the goal is accurate classification with little concern about describing how inputs are related to outputs, an unstable tree structure may not matter. What matters is whether under an alternative tree structure, observations tend to land in terminal nodes that classify those observations in the same way. It is not important whether there are more terminal nodes or fewer. The path taken to those nodes is not important either. And from that perspective, which highly correlated predictors are actually used to define the splits is formally irrelevant. The goal is stable classification.

In summary, just as in parametric regression, tree instability can result from small samples, weak predictors, or highly correlated predictors. Tree structure and/or the classes assigned to observations can be adversely affected. However, because of CART's stagewise structure, the particular tree that

results can be far more fragile than the output from a parametric regression. A single tweak near the top of the tree can fundamentally alter all that follows.

Two recommendations follow. First, there is (once again) no substitute for careful examination of the classification tree to make sure the tree makes sense. Splits that violate common sense, well-accepted theory, or past research may need to be discounted and may even call the entire tree into question.

Second, it can be very useful to get some empirical sense of how stable the results really are. As shown in the next chapter, a good strategy is to construct several bootstrap samples of the data and apply CART to each. If the tree structures are substantially different, interpreting the results from any one tree is risky. It is also important to compare the classifications from each of the trees. Substantial differences imply that the classifications assigned to observations cannot be relied upon.

A good place to start an examination of the classes assigned is with confusion tables. One question is whether the confusion tables for the different trees are alike. Another question is whether the same classes were assigned to the same cases over bootstrap samples; how consistently were the observations classified. One can compare the assigned classes for cases selected in two or more of the bootstrap samples. This idea is closely related to the "margin" an observation has and is discussed at some length in later chapters.

A related approach is to use the proportion of cases of a particular class in each terminal node. Sometimes these proportions can be considered estimates of the probability that an observation in a given terminal node is a member of the assigned class for that node. Then, for cases that appear in any pair of bootstrap samples, a scatterplot can be constructed using the terminal node proportions from two different trees. Substantial departures from the 45-degree line indicate meaningfully different assigned probabilities.

A more compelling approach is to determine if the different CART models classify test data in the same manner. When no test sample is available, there are creative approaches that depend on resampling, in much the same spirit as cross-validation. How such data may be obtained and exploited is a very important topic that is addressed in later chapters.

If one concludes that CART results are unstable, it can sometimes be helpful to apply CART to the original data again after setting the tuning parameters to produce more stable results. Once again, increasing minimum sample sizes for all nodes or increasing the value of the complexity penalty can be helpful. In the next chapter we consider other alternatives.

3.14 Regression Trees

The emphasis in this chapter has been on categorical response variables. Classification was the goal. The reasons were both pedagogical and practical. By concentrating on categorical response variables, the full range of fundamental issues surrounding CART are raised.

But CART is certainly not limited to categorical response variables. Quantitative response variables are also fair game. And with the discussion of categorical response variables largely behind us, a transition to quantitative response variables is relatively straightforward. It is possible to be brief.

A key difference with regression trees is the splitting criterion employed. For the conventional regression case, the impurity of a node is represented by the within-node sum of squares for the response:

$$i(\tau) = \sum (y_i - \bar{y}(\tau))^2, \tag{3.17}$$

where the summation is over all cases in node τ, and $\bar{y}(\tau)$ is the mean of those cases. Then, much as before, the split s is chosen to maximize

$$\Delta(s, \tau) = i(\tau) - i(\tau_L) - i(\tau_R). \tag{3.18}$$

No cost weights can be used because there is no reasonable way to consider false positives and false negatives without a categorical response variable. Then, to get the impurity for the entire tree, one sums over all terminal nodes to arrive at $R(T)$. Regression trees can be pruned, but usually with a penalty based on the AIC or some other statistic that takes the number of estimated parameters into account. Overall fit quality is then based on a summary statistic, such the root mean squared error or a measure that adjusts the degrees of freedom, much as in conventional parametric regression.

There is no classification as such. Each observation is placed in a terminal node and is then assigned the mean of that node. The assigned mean indicates how a case is "classified." The collection of means for all of the terminal nodes are, therefore, fitted values analogous to the fitted values from conventional parametric regression. They represent how the numerical response is related to the predictors.

Just as in parametric regression, it is fitted values that are typically used in forecasting. With these conditional means in hand, each new observation for which the outcome is unknown is placed in a terminal node, depending on its predictor values. The conditional mean of that node is generally taken as the "best guess" of what the value of the response variable should be.

All of the earlier concerns about CART still apply, save for those linked to the classes assigned. Potential bias and instability remain serious problems. And possible remedies are also effectively the same.

3.14.1 An Illustration

Key output of a regression tree is usually presented as a tree diagram, much as in the categorical response variable case. And as before, interaction effects can dominate. There are often easy extensions to the generalized linear model so that a response measured in counts, for example, can be used.

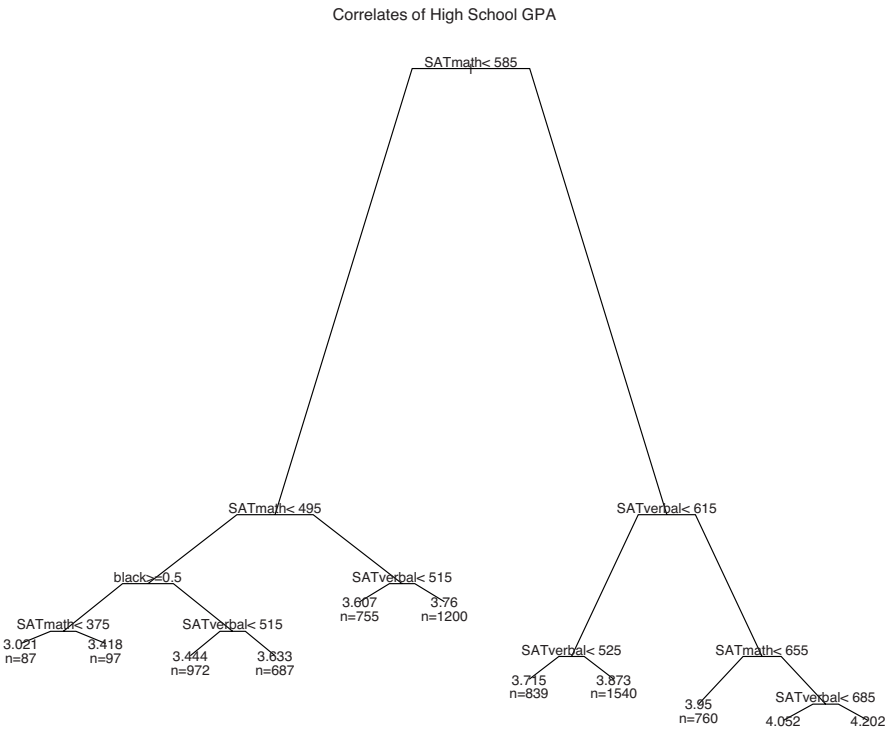


Fig. 3.10. Predictors of grade point average in high school.

Figure 3.10 shows a regression tree for correlates of high school grade point average for applicants to the large public university discussed earlier. Predictors include verbal SAT scores, mathematics SAT scores, and ethnicity. Grade point average can be as high as 5.0 because the scoring gives performance in advanced placement classes extra weight. The mean GPA in each node is shown along with the number of cases. The mean GPA for each terminal node is the fitted value assigned to all observations in that node.

The CART tree indicates that high school GPA is a nonlinear function of performance on the SATs. Ethnicity also seems to matter a bit. Only about 30% of the variance in high school GPA is accounted for with the available predictors. Clearly, GPA is measuring a lot in addition to whatever the SATs

measure, at least among applicants to this university. This is important because it implies that the high school GPA by itself contains potentially important information that cannot be obtained from the SATs.

The means in the terminal nodes range from a high of 4.202 to a low of 3.021. Interpretation follows directly from the tree. At each break point, the cases that meet the split criterion go to the left daughter node. For illustrative purposes, an indicator variable is constructed for each ethnic group. Because these are coded so that the value "1" means the presence of a given ethnicity and "0" the absence, splits are characterized by the halfway point of .5.

Just as in the classification case, many of the break points after the first can represent interaction effects For example, applicants in the node with an average GPA of 3.607 score below 515 on the verbal SAT and between 585 and 495 on the mathematics SAT. Applicants in the node with the highest average GPA score above 685 on the verbal SAT and 655 on the mathematics SAT. The race effect found for applications with middling verbal and math SATs is small and makes little intuitive sense. It may be a chance artifact.

3.14.2 Some Extensions

Save for some details, regression trees can be interpreted much as are classification trees. CART output can be represented in part by a tree diagram that shows how the predictors are related to the response. There are clear links to conventional linear regression as well, with conditional means as fitted values, variation around the fitted values as a product of within-node residuals, and familiar measures of fit.

For regression trees, costs are addressed with the fitting function itself, which in this case is the error sum of squares. Just as with conventional regression, therefore, the fitting assumes a quadratic loss function. Overestimates of the conditional mean are treated the same as underestimates of the conditional mean, and large residuals are given especially heavy weight (thanks to squaring).

We have already seen that when CART is used for classification, symmetric costs are often inappropriate. Yet they are required when the response variable is quantitative. For example, if an admissions officer is trying to forecast how well a student will do in college, a prediction that pegs the student's freshman GPA one full point too high has the same costs as a prediction that pegs the student's freshman GPA one full point too low. One consequence of the first error may be that the student actually struggles in college and then flunks out. Another consequence is that a student who might have done much better was not admitted instead. One consequence of the second error is not admitting a student who might have done very well in college. Another consequence is that a weaker student may have been admitted instead. None of these considerations can be easily introduced in the fitting process when the response variable is quantitative.

The heavy weight given to the largest residuals means that the fitting process can be greatly affected by influential observations, just as in conventional regression. A residual equal to 2.0 is only one point farther away from the conditional mean than a residual of 1.0, but the squaring weights the two observations four to one when the fitting is undertaken. One implication is that a few very atypical observations that are away from the mass of the data may dominate the results, which may then not characterize the bulk of the data appropriately. So, the results may not be an accurate description of the data on hand. A second implication is that the results can be very unstable with respect to random samples of the data. The story can change depending on whether the random sample analyzed happens to include the outliers. Generalization may be seriously compromised.

There are in principle all of the usual ways to "robustify" the CART fitting process. In particular, one can use a linear loss function instead of a quadratic, which implies fitting conditional medians rather than conditional means. Medians will not be affected by outliers and the linear loss function weights larger residuals less heavily than the quadratic loss function. An easier but less elegant fix is to fit a trimmed response so that, for instance, the largest 5% and the smallest 5% of the values for the response variable are dropped before the analysis begins. Some implementations of CART allow for these and other options. When they are available, their usefulness should be carefully assessed in the context of the data to be analyzed and the empirical questions being asked. At the moment, rpart() in R does not allow for linear loss.

Even if concerns about a few influential observations are not significant, it may be appropriate to abandon quadratic loss on other grounds. If there is interest in the conditional medians, linear loss follows naturally. If the response is a count, a Poisson formulation may be appropriate. Then the Poisson deviance is used as the splitting criterion. In R, rpart() currently allows for the Poisson. A rather different set of problems is generated because of CART's well-known proclivity to overfit and favor predictors with many possible splits. The former can lead to generalization error and the latter can lead to bias in the tree structure. As noted earlier, Hothorn et al. (2006) have proposed some novel means to address both problems. There are some important questions about how well their approach will work in practice, but the requisite software is available in R (in the library *party*) and is certainly worth trying.

3.14.3 Multivariate Adaptive Regression Splines (MARS)

Multivariate Adaptive Regression Splines (MARS) can be viewed as another kind of smoother, in the traditions of the last chapter, or as a twist on classification and regression trees. For ease of exposition, this discussion builds on CART, and it is brief. An excellent and more extensive examination of MARS can be found in Hastie et al. (2001: Section 9.4).

A key difference between CART and MARS is in the nature of the basis functions used. The MARS formulation is the broadly familiar

$$f(X) = \beta_0 + \sum_{m=1}^{M} \beta_m h_m(X), \tag{3.19}$$

where as before, there are M weighted basis functions $h_m(X)$. Likewise, the basis functions are still determined for each split by searching over all predictors and thresholds on those predictors before the best partition is selected. But in CART, the result is a step function. In MARS, the result is a V-shaped function composed of two linear splines, with its point at the threshold value. The two spline functions are mirror images of each other; hence the V-shape. Hastie et al. (2001: 283) call the two splines a "reflected pair."

Another important difference is that nodes may be split more than once. To take a simple example, the root node is initially split in two as usual. But at a later step, the root node may be revisited and split again through a new product variable. This capacity exists for all internal nodes as well.

In short, a MARS model takes the form of Equation 3.19, where the basis functions can be reflected pairs or the product of reflected pairs. Thus, MARS can fit increasingly higher-order interaction terms, just as CART does, but these are the product of linear splines not the product of indicator variables.

Estimation is done by least squares. There are often various tuning parameters that can help determine, for instance, how complex a model is permitted. The output of MARS can include the equation actually estimated, and an ANOVA-type partitioning of the explained variance to represent predictor "importance." A lot more is said about variable "importance" in the next several chapters.

MARS can be extended to classification tasks, where logistic regression replaces linear regression as the primary engine. For both regression problems or classification problems, MARS will sometimes perform better than CART. Its main comparative advantage is the ability to better capture additive models. At the same time, MARS suffers from many of the same weaknesses as CART.

One must be very clear that when MARS is used to describe how predictors are linked to a response, MARS is an exploratory tool. There is no pretense of producing a causal model despite the fact that Equation 3.19 can have much the look and feel of a conventional regression equation. The weights represented by the β_m are regression coefficients to be sure, but they have no necessary causal interpretation. MARS is in the same tradition as all of the procedures we have been considering.

MARS has its advocates, but it does not seem to have the same popularity as CART. One reason may be that MARS is not available in the more popular, large software packages such as STATA or SPSS, nor in free computing environments such as R. A license for MARS must be obtained from Salford Systems. And like CART, MARS seems to have been superseded by newer procedures (Friedman, 1991) that are discussed in the pages ahead.

3.15 Software Issues

All of the computing done is this book was implemented in R. Within R, the best CART implementation arguably is in the procedure rpart() in the R library *rpart*. It is a very powerful and flexible implementation, especially given all of the other capabilities available in R. CART is also available in a number of conventional statistical packages (e.g., SPSS), although the algorithms employed, the options provided, and user interfaces can vary dramatically. For those readers who have not tried to use CART, it may be helpful to briefly consider what the required inputs are likely to be.

1. The response variable needs to be specified, usually with information about whether it is to be treated as quantitative or categorical. CART software can sometimes get confused if the response is binary and represented by an indicator variable or some other numeric values. The safest way to proceed in rpart(), for example, is to explicitly label binary outcomes as a categorical variable (i.e., a factor).
2. The predictor variables, which can be quantitative, categorical, or both, need to be specified. If categorical, it may be important to indicate that explicitly. Otherwise, numerical values, really meant to be just category labels, may be treated as an equal interval scale. Among the issues that usually need to be thought through is whether any product variable for interaction effects should be constructed and entered as predictors or whether it is preferable to let CART construct interaction terms as needed.
3. The method, which usually means either a regression tree or a classification tree, needs to be determined. There are sometimes other options available for count data or survival data. Although some programs can determine the proper method from the nature of the response variable specified, it is usually a good idea to specify the method explicitly in a separate argument. That way, the user knows for certain what method is being used. Moreover, sometimes a CART algorithm will make the wrong choice.
4. The fitting function to be minimized needs to be specified. Sometimes this is determined when the method of analysis is selected and sometimes not. In rpart() with a categorical response variable, one has to specify whether the Gini index or the entropy is to be used.
5. The costs of false negatives and false positives need to be determined, sometimes with an appropriate prior distribution, a loss matrix, or some other means. It is also very important to understand what the defaults on these costs are.
6. A penalty for complexity is often required. It may be the value of α or some transformation of it. In rpart(), for example, the relevant parameter is *cp*, a standardization of α. In the regression case, the value of *cp* is α divided by the error sum of squares in the root node. In the classification case, α is divided by the cost complexity $R(T_0)$ of the root node.

The standardization makes the *cp*-value a fraction and allows for better comparisons across the results from different response variables.

7. Other tuning parameters are often available, such as the maximum number of splits and the minimum sample size for terminal nodes. It is very important to learn what the default values of the tuning parameters are. Although they are usually set to reasonable values, one should not assume that they are.
8. Often one can request predicted values either for the training data or for test data, and specify what form they should take. For classification trees, one usually has a choice between the predicted class or the predicted probability of membership in a class.
9. There are usually a number of output options for different kinds of graphics and summary tables. Which mix of outputs is appropriate depends on the data analysis task. For example, a richer mix of tabular outputs will often be needed the first time a training dataset is analyzed or if there are suspicions about the quality of the data. Output issues also arise for graphical output. For example, getting tree diagrams into a form that is aesthetically pleasing and easy to read can take some doing.
10. It is also likely that any CART software will have its fair share of quirks. With rpart(), for example, priors are entered as a list of the form c(.30,.70). But which element of the list is for which category? Perhaps the arguments should be entered as c(.70,.30). In rpart(), the proper sequence is determined by numerical or alphabetical order. For example, if the response is coded "1" or "2", c(.30,.70) assumes a prior distribution in which 30% of the cases are 1s and 70% of the cases are 2s. If the response is coded "yes" or "no," c(.30,.70) assumes a prior distribution in which 30% of the cases are no and 70% of the cases are for yes.
11. Programs will differ in how well they check for errors and how clearly they communicate with the user when problems are discovered. Continuing with the example of specifying priors, rpart() checks to see if they add to 1.0 and if not, tells the user very clearly. It is far less apparent from CART rpart() output if the loss matrix is constructed incorrectly.

3.16 Summary and Conclusions

CART can sometimes be an effective statistical learning tool. It is relatively easy to use, builds directly on familiar regression procedures, does not demand great computing power, and generates output that can be presented in an accessible manner. CART also provides a useful way to introduce the costs of classification errors. However, CART also has some important limitations.

First, if there is a true $f(X)$ outside of the data on hand, there is no reason to believe that CART's $\hat{f}(X)$ will provide an unbiased estimate. Despite the flexible ways in which CART can respond to data, substantial bias is a real possibility. As noted many times already, if one is interested in the $f(X)$,

one needs X. And X must be well measured. But even if these demanding prerequisites are met, the CART algorithm introduces some important constraints. The step functions used can badly misrepresent smooth functions of X, and the conditional proportions or means in terminal nodes will usually be determined in part by nearest neighbors that do not necessarily have the same population value for the response variable. One's best hope is that the flexibility CART provides will produce a $\hat{f}(X)$ that is less biased than an alternative derived from a parametric procedure.

Second, the splitting decisions can be very unstable. A few observations can in some situations dramatically affect which variables are selected and the precise values used for partitioning the data. Then, all subsequent partitions can be affected. This instability is closely related to overfitting, which can substantially limit the generalizability of the results. The findings from the data examined may not generalize well to other random samples from the same population (let alone to data from other populations). The problem of overfitting is addressed head-on in the next chapter.

Third, even moderately elaborate tree diagrams will seriously tax substantive understanding. The problem is not just complexity. CART is trying in a single-minded manner to use associations in the data to maximize the homogeneity of its data partitions. How those associations come to be represented may have nothing remotely to do with subject matter understandings or how subject matter experts think about those associations.

For example, well-accepted theory and past empirical research may argue strongly for a relatively smooth nonlinear relationship between a predictor and a response. CART may represent the relationship as a complicated step function. Even more confusing, the nonlinearities may be picked up in interaction effects with other predictors, thanks to associations in the data among all of the predictors and between the predictors and the response variable. The splitting process imposes no subject matter constraints on how the nonlinearities are captured. The subsetting can unfold within a single predictor (i.e., a step function) or within two more more predictors (i.e., interaction effects) without regard for what can be sensibly interpreted.

More troubling, if there is no well-accepted theory or strong empirical research to guide interpretation, the data analyst can be at the mercy of the CART algorithm. Thus, interaction effects empirically revealed may force interpretations that are essentially artifacts of how the subsetting was done. In this case, researchers will not just be confused, but risk being led astray.

Fourth, the exploratory nature of CART and the substantial likelihood of bias mitigates against the sensible use of statistical inference. Thus, the role of random sampling error can be very difficult to integrate into any subject matter conclusions. Assessments of the stability of the results can be very helpful, but these are somewhat different from the familiar confidence intervals and hypothesis test p-values.

In summary, it is important to distinguish between conditional means or proportions and the tree structure. Using the tree structure to interpret how

inputs are related to outputs is often a bad idea. Or put more constructively, there needs to be strong subject matter information to protect against misleading interpretations. If interest centers on the conditional means or proportions alone, however, one can sometimes be more hopeful. Trees that differ because of instability will often produce similar sets of fitted values. Thus if CART is used solely as a classifier, the results may be helpful. But if there is a need to explain why cases are classified as they are, the instability may be debilitating.

Fortunately, there is more in the bag of statistical learning tricks than CART. As we soon show, there is really no need to risk CART's significant limitations. One can do better and in the next chapter, we begin to consider how.

Exercises

Problem Set 1

The purpose of this exercise is to provide an initial sense of how CART compares to conventional linear regression.

1. To begin, construct a regression dataset with known properties:

```
x1=rnorm(300)
x2=rnrom(300)
error=2*rnorm(300)
y1=1+(2*x1)+(3*x2)+error
```

   Apply conventional linear regression using lm(). Then apply rpart(), and print the tree using text(). Compare the regression output to the way in which the data were actually generated. Compare the tree diagram to the way in which the data were actually generated. Compare how well linear regression and CART fit the data. (This may take a little doing depending on what summary measures of fit rpart() provides. One easy option is to construct the fitted values with predict() and then regress the fitted values on the observed values to get fit measure comparable to those from the linear regression analysis.) What do you conclude about the relative merits of linear regression and CART when the $f(X)$ is actually linear and additive?

2. Now, redefine the two predictors as binary factors and reconstruct the response variable.

```
x11=(x1 > 0)
x22=(x2 > 0)
y=1+(2*x11)+(3*x22)+error
```

Proceed as before comparing linear regression to CART. How do they compare? What do you conclude about the relative merits of linear regression and CART when the $f(X)$ is actually a step function and additive?

3. Under what circumstances is CART likely to perform better than linear regression? Consider separately the matter of how well the fitted values correspond to the observed values and the interpretation of how the predictors are related to the response.

Problem Set 2

The goal of the following exercises is to give you some hands-on experience with CART in comparison to some of the procedures covered in earlier chapters. An initial hurdle is getting R to do what you want. Make generous use of help(). Also, I have provided a number of hints along the way. However, I have tried to guide you to results in the least complicated way possible and as a consequence, some of the more subtle features of CART are not explored. Feel free to play with these in addition. You can't break anything.

Load the data set called "frogs" from the DAAG library. The data are from a study of ecological factors that may affect the presence of certain frog populations. The binary response variable is pres.abs. Use the help command to learn about the data. For ease of interpretation, limit yourself to the following predictors: altitude, distance, NoOfPools, NoOfSites, avrain, meanmin and meanmax.

1. Use logistic regression from glm() to consider how the predictors are related to whether frogs are present. Which predictors seem to matter? Do their signs make sense?

2. Using the procedure stepAIC() from the MASS library with the default for stepwise direction, find the model that minimizes the AIC. Which predictors remain? Do their signs make sense?

3. Using table() or xtabs(), construct a confusion table for the model arrived at by the stepwise procedure. The observed class is pres.abs. You will need to assign class labels to cases to get the "predicted" class. The procedure glm() stores under the name "fitted.values" the estimated conditional probabilities of the presence of frogs. If the probability is greater than .5, assign a "1" to that case. If the probability is equal to or less than .5, assign a "0" to that case. Now cross-tabulate the true class by the assigned class. What fraction of the cases is classified incorrectly? Is classification more accurate for the true presence of frogs or the true absence of frogs? What is a rationale for using .5 as the threshold for class assignment?

4. Using your best model from the stepwise procedure, apply the generalized additive model. Use smoothers for all of the predictors. Look at the

numerical output and the smoothed plots. How do the results compare to those from logistic regression?

5. Construct a confusion table for the model arrived at through GAM. Once again, the observed class is pres.abs. Use the same logic as applied previously to determine the assigned class. What fraction of the cases is classified incorrectly? Is classification more accurate for the true presence of frogs or the true absence of frogs? How do these results compare to the GLM results?

6. Going back to using all of the predictors you began with, apply CART to the frog data via the procedure rpart() in the library *rpart*. For now, accept all of the default settings. But it is usually a good idea to specify the method (here, method= "class") rather than let rpart() try to figure it out from your response variable. Use the print() command to see some key numerical output. Try to figure out what each piece of information means. Use the plot() and text() commands to construct a tree diagram. What predictors does CART select as important? How do they compare with your results from GLM and GAM? How do the interpretations of the results compare?

7. Use predict() to assign class labels to cases. You will need to use the help command for predict.rpart() to figure out how to do this. Then construct a confusion table for the assigned class and the observed class. What fraction of the cases is classified incorrectly? Is classification more accurate for the true presence of frogs or the true absence of frogs? How do these results compare to the GLM and GAM results? If the three differ substantially, explain why you think this has happened. Alternatively, if the three are much the same explain why you think this has happened.

8. Run the CART analysis again with different priors. Take a close look at the information available for rpart() using the help command. For example, for a perfectly balanced prior in rpart() you would include parms=list(prior= c(.50.50)). Try a prior of .5 for presence and then a prior of .30 for presence. (For this rpart() parameter, the prior probability of 0 comes first and the prior probability of 1 comes second.) What happens to the amount of classification error overall compared to the default? What happens to the ratio of false negatives to false positives? (To understand better what is going on look again at Section 3.5.2.)

9. Using Equation 3.11 set the prior so that false negatives are ten times more costly than false positives (with pres.abs = 1 called a "positive" and pres.abs = 0 called a "negative"). Apply CART. Study the output from print(), the tree diagram using plot() and text(), and the confusion table. What has changed enough to affect your interpretations of the results? What has not changed enough to affect your interpretations of the results?

10. Construct two random samples with replacement of the same size as the dataset. Use the sample() command to select at random the rows of data you need and use those values to define a new sample with R's indexing capability, x[r,c]. For the two new samples, apply CART with the default parameters. Construct a tree diagram for each. How do the two trees compare to each other and to your earlier results with default settings? What does this tell you about how stable your CART results are and about potential problems with overfitting.

11. Repeat what you have just done, but now set the minimim terminal node size to 50. You will need the argument control=rpart.control (minbucket=50)) in your call to rpart(). How do the three trees compare now? What are the implications for overfitting in CART?

Problem Set 3

Here is another opportunity to become familiar with CART, but this time with a quantitative response variable. From the library *car*, load the data set "Freedman." The dataset contains for 100 American cities the crime rate, population size, population density, and percent nonwhite of the population. The goal is to see what is associated with the crime rate.

1. Using the generalized additive model (GAM) from the library *gam*, regress the crime rate on the smoothed values of the three predictors. Examine the numerical output and the plots. Describe how the crime rate is related to the three predictors.

2. Repeat the analysis using rpart() and the default settings. Describe how the crime rate is related to the three predictors. How do the conclusions differ from those using the generalized additive model?

3. Plot the fitted values from the GAM analysis against the fitted values from the CART analysis. The fitted values for gam() are stored automatically. You will need to construct the fitted values for CART using predict(). What would the plot look like if the two sets of fitted values corresponded perfectly? What do you see instead? What does the scatterplot tell you about how the two sets of fitted values are related?

4. Overlay on the scatterplot the least squares line for the two sets of fitted values using abline(). If that regression line had a slope of 1.0 and an intercept of 0.0, what would that indicate about the relationship between the two sets of fitted values? What does that overlaid regression line indicate about how the two sets of fitted values are related?

5. Using scatter.smooth(), apply a lowess smoother to the scatterplot of the two sets of fitted values. Try several different spans. What do you conclude about the functional form of the relationship between the two sets of fitted values?

6. For the GAM results and the CART results, use cor() to compute separately the correlations between the fitted values and the observed values for the crime rate. What procedure has fitted values that are more highly correlated with the crime rate? Can you use this to determine which modeling approach fits the data better? If yes, explain why. If no, explain why.

4

Bagging

4.1 Introduction

In this chapter, we make a major transition. We have thus far focused on statistical learning procedures that produce a single set of results: regression coefficients, measures of fit, residuals, classifications, and others. Thus, there is but one regression equation, one set of smoothed values, or one classsification tree. Most statistical procedures operate in a similar fashion.

The discussion now shifts to statistical learning building on many sets of outputs that are aggregated to produce results. Such algorithms make a number of passes over the data. On each pass, inputs are linked to outputs just as before. But the ultimate results of interest are the collection of all the results from all passes over the data.

Bayesian model averaging may be a familiar illustration from another statistical tradition (Madigan et al., 1996; Hoeting et al., 1999). In Bayesian model averaging, there is an assumed $f(X)$; there is a "true model." A number of potentially true models, differing in the predictors selected, are evaluated. The model output is then averaged with weights determined by model uncertainty. Output from models with greater uncertainty are given less weight. From a statistical learning perspective, Bayesian model averaging has a number of problems, including the dependence that is necessarily built in across model results (Xu and Golay, 2006). We address shortly how statistical learning procedures relying on multiple results proceed rather differently.

Aggregate results can have several important benefits. Averaging over a collection of fitted values can help compensate for overfitting. That is, the averaging tends to cancel out results shaped by idiosyncratic features of the data. One can then obtain more stable fitted values and more honest assessments of how good the fit really is. Second, a large number of fitting attempts can produce very flexible fitting functions able to respond to systematic, but highly localized, features of the data. In effect, there can be a very large number of basis functions and the prospect of reducing bias in the fitted values. Third, putting the averaging and the flexible fitting functions together has the

R.A. Berk, *Statistical Learning from a Regression Perspective*,
DOI: 10.1007/978-0-387-77501-2_4,

potential to break the bias–variance tradeoff. Sometimes you can have your cake and eat it too.

In this chapter, we focus on bagging, which capitalizes on the averaging process. Averaging can reduce the variance. There are also some implications for bias. Later chapters consider statistical learning procedures that in different ways address more directly bias in the fitted values as well as the variance.

We emphasize categorical response variables. We are again concentrating on classifiers. The rationale is largely the same: the exposition is more effective and the step to quantitative predictors is easy to make. We begin with a return to the problem of overfitting. Although overfitting has been discussed several times in earlier chapters, it needs to be linked more directly to CART to help set the stage for our exposition of bagging.

4.2 Overfitting and Cross-Validation

A long-standing problem in the philosophy of science is whether the credibility of scientific conclusions is greater if the conclusions are evaluated through their forecasting skill or their consistency with the data on hand. That is, what weight should be given to an accurate forecast compared to a good fit? The answer is not straightforward, but in the end, accurate forecasts are likely to be more convincing. And one of the reasons is that forecasts are not vulnerable to overfitting, whether from intentional "fudging" or overzealous data exploration (Lipton, 2005).

Any attempt to summarize patterns in a dataset risks overfitting. All fitting procedures adapt to the data on hand so that even if the results are applied to a new sample from the same population, fit quality will likely decline. Hence, generalization can be somewhat risky. And to the degree that a fitting procedure is highly flexible, overfitting can be exacerbated. There is a greater opportunity to fit idiosyncratic features of the data. For example, Hastie et al. (2001: 200–203) show in a slightly different context that the unjustified "optimism increases linearly with the number of inputs or basis functions ..., but decreases as the training sample size increases." In other words, it can be highly desirable to have few parameters to be estimated and many observations with which to construct the estimates.

Consider CART as a key illustration. The basis function formulation can be instructively introduced at three points in the fitting process. First, for any given predictor being examined for its best split, overfitting will increase with the number of splits possible. In effect, a greater number of basis functions are being screened (where a given split leads to a basis function). Second, for each split, CART evaluates all possible predictors. An optimal spilt is chosen over all possible splits of all possible predictors. This defines the optimal basis function for that stage. Hence within each stage, overfitting increases as the number of candidate predictors increases. Third, for each new stage, a new optimal basis function is chosen and applied. Consequently, overfitting

increases with the number of stages, which for CART means the number of optimal basis functions, typically represented by the number of nodes in the tree.

The overfitting in CART can be misleading in a number of ways. Measures meant to reflect how well the model fits the data are likely to be too optimistic. Thus, for example, the number of classification errors may be too small. In addition, the model itself may have a structure that will not generalize well. For example, one or more predictors may be included in a tree that really do not belong. Finally, should statistical inference be introduced, standard errors can be too small. Overly narrow confidence intervals and falsely powerful tests follow.

Ideally, one would have two random samples from the same population: a training dataset and a test dataset. A tree would be built from the training data, and some measure of fit would be obtained. A simple measure might be the fraction of cases classified correctly. A more complicated measure might take the costs of false negatives and false positives into account. Then with the tree structure in place, cases from the test data would be "dropped down" the tree, and the fit computed again. It is almost certain that the fit would degrade, with how much being a measure of overfitting. The fit measure from the test data would be a better indicator of how accurate the classification process really is.

Often there is only a single dataset. Enter cross-validation. The data are split up into several randomly chosen, nonoverlapping, partitions of about the same size. That is, one samples without replacement. Ten such subsets are common. CART is applied to the data from nine of the partitions, and the results are evaluated with the remaining partition. So, if there are 1000 observations, one would build the tree on 900 randomly selected observations and evaluate the tree using the other 100 observations.

With ten partitions, the building and testing sequence could be undertaken ten times, each time with nine partitions as the training data and one partition as the test data. Each of the ten partitions would be part of the training data for nine of the ten analyses, and would serve as the test data for one of the ten analyses. From each of the ten test partitions, a measure of fit would be computed. An instructive measure of fit would be the average fit value over the ten splits. Relying on the test partitions reduces overfitting. Taken one at a time, the small test partitions can be vulnerable to sampling error. The averaging process tends to cancel out some chance variation. There is nothing magic about using ten random partitions of the data. When there are very few partitions, each training dataset will have far fewer observations than the entire sample. Insofar as the CART results are sample size dependent, substantial bias can be introduced. For example, with a smaller training sample, a less complex tree might result. However, when there are a great many partitions, largely the same data are used over and over to construct the tree, and the test datasets have very few observations. Then, the fit measure computed

from the test data can have high variance (Hastie et al., 2001: Section 7.10). Using five to ten splits seems to be a good compromise in practice.

Cross-validation is available in many implementations of CART and is discussed in the seminal book by Breiman and his colleagues (1984: Section 11.5). Often the number of splits of the data can be specified. When the number of splits is the same as the number of observations in the original sample, the process is sometimes called "leave-one-out" cross-validation. We discussed this in Chapter 1 when model evaluation was first addressed. As noted then, extensions on this basic idea using bootstrap samples are available (Efron and Tibshirani, 1993: Chapter 17).

Unfortunately, cross-validiation neglects the extracted pattern of associations between the inputs and the outputs, which may, because of overfitting, be very misleading. Although one may obtain a more honest measure of overall performance, the structure of the associations revealed by the analysis is not addressed. One may be stuck with a tree that makes little substantive sense or will not generalize well. But in the use of subsamples of the data and in averaging over subsamples, there is a very powerful idea. Bagging exploits that idea to address overfitting in a more fundamental manner.

4.3 Bagging as an Algorithm

The notion of combining fitted values from a number of fitting attempts has been suggested by several authors (LeBlanc and Tibshirani, 1996; Mojirsheibani, 1997; 1999) In an important sense, the whole becomes more than the sum of its parts. "Bagging," which stands for "Bootstrap Aggregation," is perhaps the earliest procedure to exploit a combination of fitted values based on random samples of the data (Breiman, 1996). Bagging may be best understood initially as nothing more than an algorithm.

Consider the following steps in a fitting algorithm with a dataset having N observations and a binary response variable.

1. Take a random sample of size N with replacement from the data.
2. Construct a classification tree as usual but do not prune.
3. Assign a class to each terminal node, and store the class attached to each case coupled with the predictor values for each observation.
4. Repeat Steps 1-3 a large number of times.
5. For each observation in the dataset, count the number of times over trees that it is classified in one category and the number of times over trees it is classified in the other category
6. Assign each observation to a final category by a majority vote over the set of trees. Thus, if 51% of the time over a large number of trees a given observation is classified as a "1," that becomes its classification.
7. Construct the confusion table from these class assignments.

Although there remain some important variations and details to consider, these are the key steps to produce "bagged" classification trees. The idea of classifying by averaging over the results from a large number of bootstrap samples generalizes easily to a wide variety of classifiers beyond CART. Later we show that bagging can be usefully applied for quantitative responses as well.

4.3.1 Margins

Bagging introduces some new concepts that need to be addressed, not just to deepen the understanding of bagging, but for some other procedures considered in later chapters. One of these concepts is the "margin."

Operationally, the difference between the proportion of times a case is correctly classified and the proportion of times it is incorrectly classified is sometimes called the "margin" for that case. If, over all trees, an observation is correctly classified 75% of the time and incorrectly classified 25% of the time, the margin is $.75 - .25 = .50$. Large margins are desirable because a more stable classification is implied. In a large number of random samples of the data, the class assigned to that observation is far more likely than not to be the same. Ideally, there should be large margins for all of the observations. This bodes well for generalization to new data. A more formal and extensive treatment of the concept of the "margin" is provided in the next chapter.

Recall the discussion in the previous chapter on instability in CART fitted values. Overfitting in CART tends to be more serious when for the terminal nodes the proportions of observations in each of the response variable classes tend to be similar. If the split is .51 versus 49, for instance, the movement of a little more than one percent of the cases from one class to another could change the class assigned to that node. Then, all of the cases that were correctly classified are now misclassified, and all of the cases incorrectly classified are now correctly classified. Were another sample taken, the initial node class might be reassigned, and the pattern of classification errors would change again. It follows that for any given observation in this terminal node, the margin is likely to be very small or even negative. Such observations will not be classified in a reliable manner. One might say that the vote over trees is too close to call.

Conversely, if the proportions within each terminal node are quite different, it would take the movement of relatively many cases to change the classes assigned. The bagged margins for observations across trees are likely to be larger and the classifications more stable. More reliable classifications result. One might say that the vote is a landslide.

4.3.2 Out-Of-Bag Observations

In the steps just described, the tree is built and then the data used to build the tree are used again to compute the classification error. One way to think

about this is that training data are "dropped down" the tree to determine how well the tree performs. The training data are "resubstituted" when tree performance is evaluated.

In some implementations of bagging, one can do better. For each tree, observations not included in the bootstrap sample (called "out-of-bag" observations) can be treated as a test dataset. These are then dropped down the tree instead of the data used to build the tree. A record is kept of the class with which each out-of-bag observation is labeled, as well as its values on all of the predictors. Then in the averaging process, it is these assigned values that are used as class labels, and based on these, a confusion table contructed. In other words, the averaging for a given observation over trees is done only using the trees for which that observation was not used in the fitting process. Thus, a fitting enterprise has been turned into a genuine forecasting enterprise. This leads to more honest fitted values and more honest confusion tables.

It is important to emphasize that the improvement will usually be seen in forecasting accuracy. If bagged CART results are compared to the results from a single classification tree, the single tree may seem to perform better. But this is misleading. If resubstituted values are used to construct the confusion tables for both the single tree and the bagged trees, the bagged trees should look worse. The bagging results have been adjusted for overfitting, at least in part. When out-of-bag data are used to construct the confusion table for the bagged trees, the bagged results will appear to suffer even more by comparison. A fair competition between the performance of a single tree and a set of bagged trees requires confusion tables for both procedures constructed from test data. On this level playing field, bagged trees will usually perform better than single trees. Some exceptions are considered shortly.

In summary, by assigning cases to categories using a majority vote over a set of bootstrapped classification trees, overfitting can be dramatically curtailed. Forecasting accuracy is improved because generalization error is reduced. Using the out-of-bag observations can further curb the potential overfitting.

4.4 Some Thinking on Why Bagging Works

The core of bagging's potential is found in the averaging over results from a substantial number of bootstrap samples. As a first approximation, the averaging helps to cancel out the impact of random variation. However, there is more to the story, some details of which are especially useful for understanding a number of statistical learning procedures discussed in subsequent chapters.

4.4.1 More on Instability in CART

One can get an initial sense of the need for bagging from Figures 4.1 to 4.3. The three figures are three classification trees constructed from the same data,

but each uses a different bootstrap sample (i.e., sampled with replacement from the data). The data were collected to help forecast incidents of domestic violence within households served by a sheriff's department from a large metropolitan area. For a sample of households to which sheriff's deputies were dispatched for domestic violence incidents, the deputies collected information on a series of possible predictors of future domestic violence. For example, they determined whether police officers had been called to that household in the recent past. Then, the households were followed for two months and any new incidents of domestic violence recorded. The data were used to construct a forecasting algorithm so that when information was collected on new households, forecasts of the likelihood of more domestic violence incidents could be made.

It is clear that the three figures are very different. Although each tree's initial splitting variable is the number of times the police had been called to that household before, different break points are chosen. More important, the subsequent splits vary widely across the three trees. It is clear that in this instance, CART does not produce trees that are likely to be stable under different random samples from the same population. It would follow that interpretations of the results would be unreliable.

This is a very important lesson. Interpretations from the results of a single tree can be quite risky when CART performs in this manner. And recall from the previous chapter that CART can produce unstable results because of any number of common problems: small sample sizes, heterogeneous terminal nodes, or highly correlated predictors.

Also problematic may be the classes that CART assigns to nodes. For each of the figures, the sample sizes in the terminal nodes are generally quite small. This can increase substantially the instability of the classes assigned. With node distributions such as 4 to 3, 5 to 4, or even 9 to 6, changes in the composition of the data from sample to sample could easily alter how the observations in a node are classified or even whether the node is constructed at all.

However, if over trees the different nodes in which a given case might fall tend to classify that case in the same manner, instability in tree structure does not necessarily translate into instability in the class assigned. In other words, when CART is used solely as a classification tool, the classes assigned may be relatively stable even if the tree structure is not. Experience suggests that such is sometimes the case. Recall that much the same phenomenon can be found in conventional regression when predictors are highly correlated. The regression coefficients estimated for particular predictors may be very unstable, but it does not necessarily follow that the fitted values will be unstable as well.

Finally, each tree has many terminal nodes. As a result, each tree represents a very flexible fitting function; there are a large number of conditional proportions estimated. As a result, the number of classification errors is likely to be relatively few for each tree individually. Each fit, therefore, may be quite

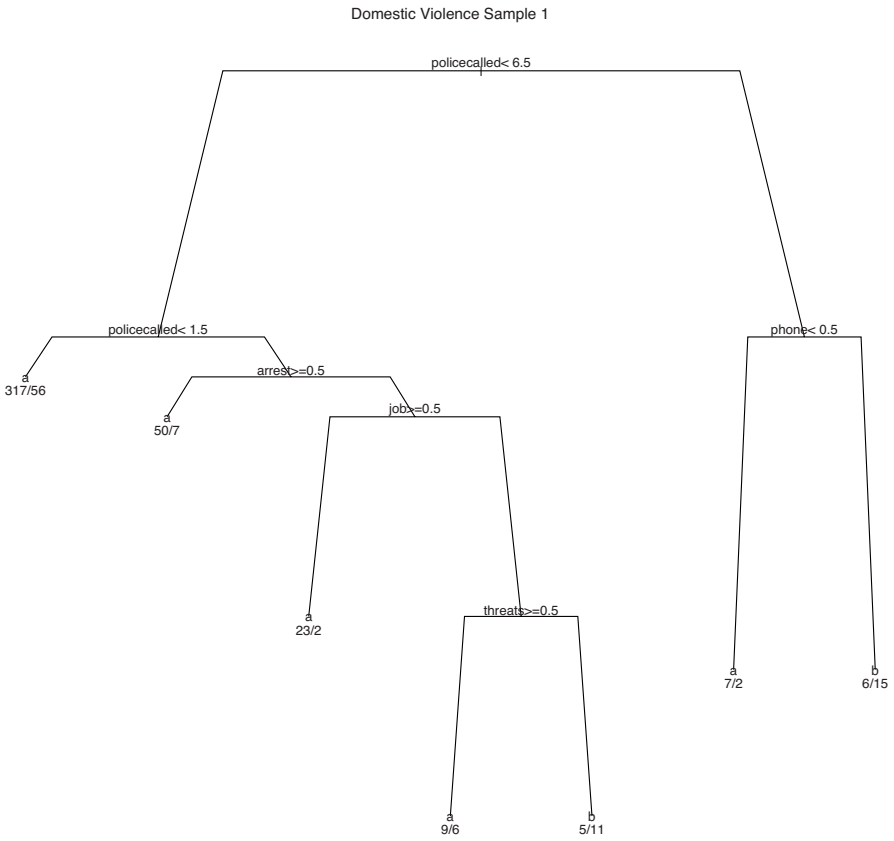


Fig. 4.1. Tree diagram for first bootstrap sample.

good. Were the original data a random sample from a well-defined population, one might be able to argue that the bias in the assigned classes is small.

At the same time, one must be clear about what is being estimated. Suppose there is a real population and CART were applied to all of the data in that population. Then, if CART is applied to a random sample of observations from that population, one might be able to grow a tree providing unbiased estimates of the splits and the fitted values. A requirement would be to have a large enough sample so that a sufficiently large tree could be grown with the sample data. That is, all of the terminal nodes in the population tree

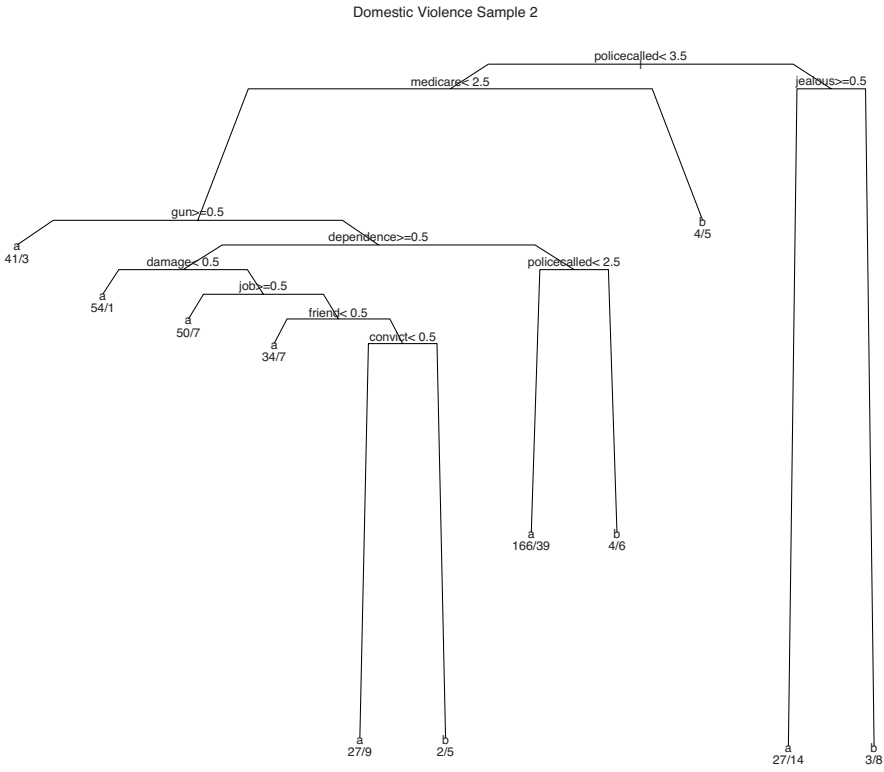


Fig. 4.2. Tree diagram for second bootstrap sample.

could be reproduced with the sample data. The deeper problem is that there is no guarantee whatsoever that the population tree, let alone the sample tree, captures the way in which the population data were generated. In the most obvious case, there may be no information in the population about all predictors that in fact were relevant.

At a more abstract level, the same concerns apply to data said to be a product of a particular stochastic process. If that stochastic process really functions through mechanisms that comport with a classification or regression tree, and if the inputs to that stochastic process are included in the dataset being analyzed, there is again the possibility of obtaining unbiased tree esti-

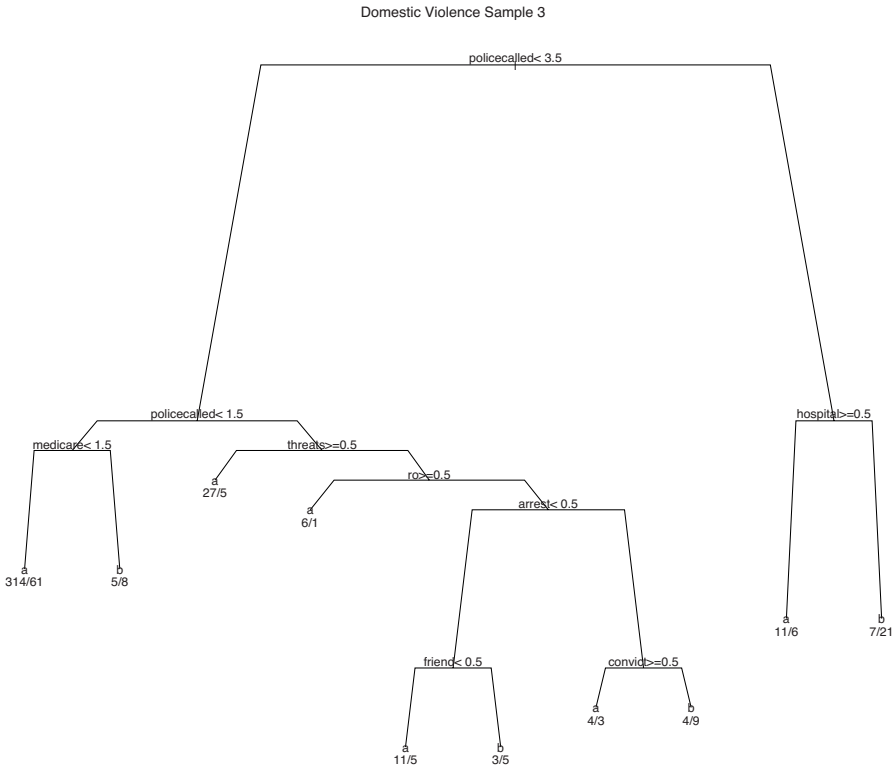


Fig. 4.3. Tree diagram for third bootstrap sample.

mates. But if the stochastic process does not comport with a classification or regression tree, or if the requisite predictors are unavailable, bias will likely result.

In short, it is not clear how much bias exists in the three trees. But it is clear that the variance across trees is large. Bagging can help with the variance.

4.4.2 How Bagging Can Help

Consider the classifications that would follow from each tree. Suppose that for each observation one averaged over trees to determine the class assigned. With a binary outcome, the averaging would take form of a vote across trees. Because there are three trees in this illustration, a majority vote would be two out of three or three out of three. If an observation were classified as having a new incident of domestic violence (i.e., "b") in two of the three trees or in three of the three trees, it would be classified as a high-risk domestic violence household. If an observation were classified as having a new incident of domestic violence in none of the three trees or one of the three trees, it would not be classified as a chronic domestic violence household.

As an averaging process, voting over trees tends to cancel out the impact of random sampling error on the classes assigned to observations (Brieman, 1996; 2000). Idiosyncratic results from tree to tree can be averaged away and more stable estimates can follow. The variance in the assigned classes can be reduced as a consequence.

The idea of independent random samples from a population must not be confused with bootstrap samples from the data. Independent random samples from a population are the conceptual foundation for conventional (frequentist) statistical inference. One works within the thought experiment of a limitless number of independent random samples from a population or a limitless number of independent realizations of a stochastic process. The definitions of the bias and variance for a statistic computed from the data on hand follow from this thought experiment.

Bootstrap samples are probability samples with replacement from the data on hand. Often such procedures are justified as an effort to simulate the thought experiment. For bagging, however, the bootstrap samples serve another purpose: they are the foundation for the averaging process by which the bias–variance tradeoff may be constructively addressed. Statistical inference is not the motivation. One can think of the averaging as a kind of shrinkage that can, as before, increase the stability of the fitted values. For each observation, fitted values are pulled toward their mean over bootstrap samples.

There is actually no requirement in bagging that the samples drawn from the training data be with replacement. It seems that in general, one can use samples without replacement and obtain virtually the same results as long as a particular relationship is maintained between the size of the larger samples with replacement and the size of the smaller samples without replacement (Buja and Stuetzle, 2006). If there are N observations in the training data, a sample without replacement of $N/2$ effectively will produce the same bagged results as a sample with replacement of size N. So, the key idea is working with a large number of random samples of the training data. Whether the sampling is with or without replacement does not by itself seem to be a critical factor.

The results developed by Buja and Steutze (2006) make clear that when sampling with replacement, the nominal sample size can be larger than N, and

as that sample size increases, the equivalent sample size for sampling without replacement can approach N. However, there is no definitive message about what the ideal sample size should be, whether with or without replacement. Therefore, the discussion that follows emphasizes a sample of size N sampling with replacement, consistent with the traditional bootstrap.

Finally, bagging can have implications for the bias (Bühlmann and Yu, 2002). The basic concern is this: bagging acts as a smoother for the step functions CART produces. If the underlying $f(X)$ is smooth, bagging will tend to reduce bias by "sanding off" the corners of the step functions. If the underlying $f(X)$ has the same jagged structure as step functions, "sanding off" the corners can increase the bias. Apparently, neither of these consequences were anticipated in the initial work on bagging but were eventually recognized as a byproduct of the averaging that bagging employs. More is said about bagging and bias shortly.

4.4.3 A Somewhat More Formal Explanation

We can now formalize these ideas a bit (Breiman, 1996) by applying concepts from conventional regression analysis. We begin with a discussion of the variance and a simple illustration to set the stage.

Bagging and Variance

Consider first a given predictor value x_0 and an associated response. Imagine a single random draw from a population, conditional on x_0. The value of the response for that draw is an unbiased estimate of the mean response for all observations in the population with the same value of x, x_0. But because that estimate is constructed from a single observation, the estimate can vary a lot from sample to sample if the response is not homogeneous at x_0. Had a random sample of, say, ten observations at x_0 been drawn instead, the mean of those 10 values would still be an unbiased estimate of the mean of the responses at x_0. But now, the sampling variability would likely be much smaller because the sample size is much larger. With more observations one can shrink the variance, and in this case, still have an unbiased estimate.

Consider now a more complicated illustration. We assume for the moment that the response variable is quantitative. We focus on a single observation. For that observation, assume there is a true function of the predictor $f(x_0)$ through which the response is related to x_0. That true function can be found in the population or in the stochastic process responsible for the data. What is the mean squared error of an observed value of the response variable y with respect to the fitted value $\hat{f}(x_0)$?

The mean squared error over repeated random samples (or realizations) can be decomposed into the sum of three parts: !1) an irreducible error, (2) the bias in the fitted value, and (3) the variance of the fitted value. More explicitly (Hastie et al., 2001: 197),

$$E[(y-\hat{f}(x_0))^2|x=x_0] = \sigma_\varepsilon^2 + [E\hat{f}(x_0) - f(x_0)]^2 + E[\hat{f}(x_0) - E\hat{f}(x_0)]^2. \quad (4.1)$$

There is nothing that can be done about the σ_ε^2. It reflects the variance of y around its true conditional mean at x_0. Generally, the more complex the model $\hat{f}$, the smaller the squared bias, shown in Equation 4.1 as $[E\hat{f}(x_0) - f(x_0)]^2$. A more complex model will generally fit the data better. But with greater complexity, a greater number of degrees of freedom is used up. The likely result is greater variance, shown in Equation 4.1 as $E[\hat{f}(x_0) - E\hat{f}(x_0)]^2$. Put another way, the available information in the sample is being spread more thinly over the fitted values being estimated. The bias–variance tradeoff is with us again.

Bagging can, in principle, usefully address the link between the bias and the variance. For any given amount of bias, averaging over many bootstrap samples produces a far more stable collection of fitted values than is likely from any single sample. It is as if one had a large number of samples (or realizations) generated by the frequentist thought experiment. Moreover, because bagging helps to produce more stable estimates, one is more free to fit complex functions to the data. If there is a subject matter rationale for fitting a tree with a large number of terminal nodes, for example, concerns about high variance need not automatically be a serious constraint.

Equation 4.1 should be understood as illustrative. In particular, it does not literally apply when the response variable is categorical. When models for categorical data are used, the bias of the fitted values is related to the variance of the fitted values. The simple partitioning shown in Equation 4.1 does not follow. Nevertheless, the same general implications apply.

Bagging and Bias

Having addressed the variance, we turn to the bias. Figure 4.4 illustrates how bagging can affect the bias. To keep the graph simple, there is a single predictor with the $f(X)$ the smooth S-shaped function shown linking the predictor to a binary 0/1 response. Imagine now that CART is applied one time to each of four different bootstrap samples of the data. Each time, only one break in the predictor is allowed. (Such trees are sometimes call "stumps.") The four step functions that result are overlaid.

Consider now a single value of the predictor x_0 of 14. At x_0, the value of $f(X)$ is about .8. If only the single step function on the far right were available, $\hat{f}(X)$ would be around .9. If only the single step function just to the left were available, $\hat{f}(X)$ would be around .75. Yet, the average of the two would be pretty close to .8. More generally, with a greater number of CART step functions averaged, the S-shaped $f(X)$ is better approximated. Bagging can reduce bias by what is, in effect, smoothing (Bühlmann and Yu, 2002). The key is that $f(X)$ is a smooth function to begin with.

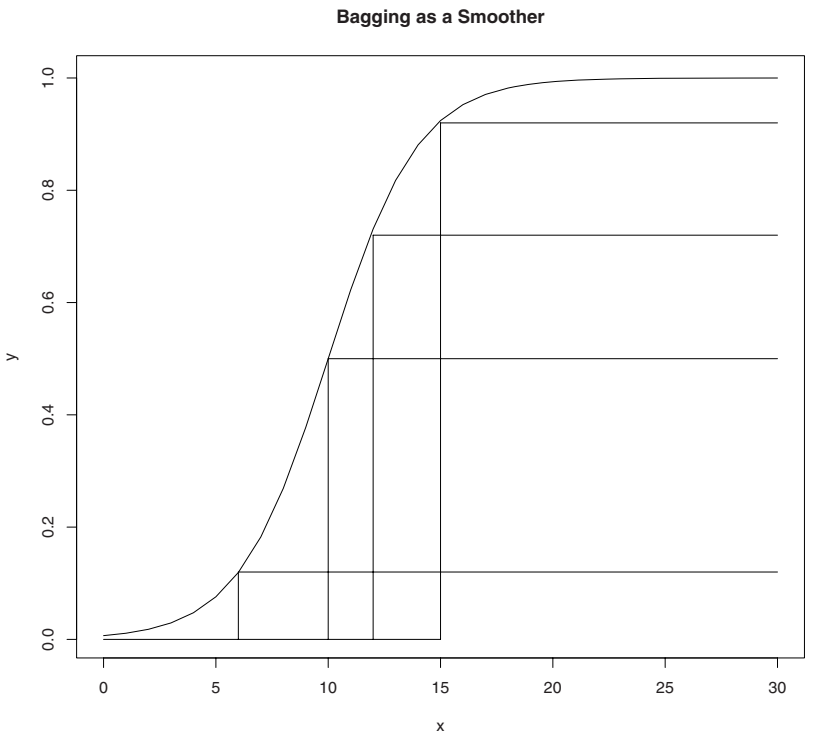


Fig. 4.4. How bagging smooths.

4.5 Some Limitations of Bagging

Bagging has been used recently in a number of interesting ways beyond classification and regression trees (Hothorn and Lausen, 2003). The principles that bagging exploits are quite general. But there are also important limitations.

4.5.1 Sometimes Bagging Does Not Help

Bagging only returns different fitted values from those that could be obtained from one pass over the original data if the fitting procedure is a nonlinear or an adaptive function of the data. For example, all of the smoothers considered earlier were, with the predictors treated as fixed, linear in the data. Recall that the fitted values were just a linear combination of the original values of the response variable. There are no gains from bagging such estimators. The fitted values from bagging would be effectively the same as the fitted values from the original data with no sampling (and identical if the number of bootstrap samples increases without limit).

4.5.2 Sometimes Bagging Can Make the Bias Worse

Look again at Figure 4.4. Suppose $f(X)$ is really very jagged, much like a step function. Then, the smoothing that bagging accomplishes can increase bias because the smoothing on the average moves the fitted values away from the correct $f(X)$. One does not want the sharp corners of the CART estimates sanded off. Classification can also be adversely affected.

Weak classifiers can also create problems, especially when the distribution of the response is highly unbalanced. Weak classifiers are sometimes defined as those that do no better than the marginal distribution. Suppose the marginal distribution of the response is unbalanced so that it is very difficult for a model using the predictors to perform better than the marginal distribution. Under those circumstances the rare class will likely be misclassified most of the time because votes will be typically be won by the class that is far more common.

To illustrate this point, suppose there is a binary response variable, and for the moment, we are interested in a single observation that happens to be a "success." For a given set of trees, that observation is classified as a success about two times out of ten. So, the classification for that observation will be wrong about 80% of the time. But if one classifies by majority vote, the class assigned would be a failure and that would be wrong 100% of the time. Because the classifier does a poor job, the majority vote produces a disappointing result. And if the other observations in the training data tend to be affected by the same difficulty, bagging will perform less well than CART. Bias is increased.

In practice, such problems will be rare if the data analyst pays attention to how the classifier performs before bagging is applied. A key question is how the estimated functions being bagged correspond to the function being estimated. If serious mismatches are avoided, one important source of bias can be reduced. In addition, one should always proceed with great caution if one has very weak classifiers. We show in later chapters that if one has weak classifiers, alternative procedures may be called for.

4.5.3 Sometimes Bagging Can Make the Variance Worse

Bagging sometimes can also perform poorly with respect to the variance (Grandvalet, 2004). Figure 4.5 shows a scatterplot with a binary outcome. The observations are represented by shaded rectangles. Two are far darker than the rest. Both are outliers in x. Consider now their role for the fitted values.

Suppose that the response is a linear function of the x. The fitted values, therefore, should also be a linear function of x. Working with a linear function makes the exposition much easier, and the general lessons from the discussion that follows apply when the response and the fitted values are a nonlinear

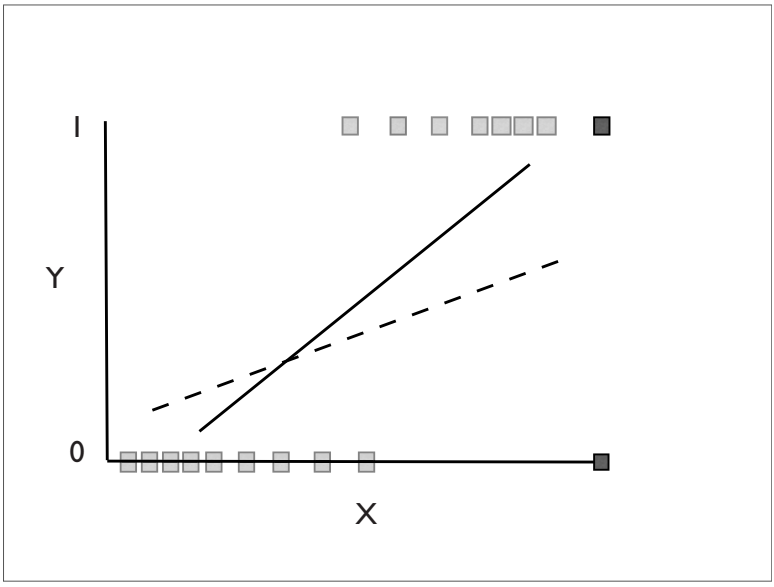


Fig. 4.5. The role of influence in bagging.

function of x. The lessons carry over as well to fitting exercises when there is more than one predictor.

The solid line shows the fitted values with the lower-right outlier excluded from the data. The dotted line shows the fitted values with the lower-right outlier included. The lines are rather different, implying that whether that value is included in the analysis alters the response function substantially. Therefore, the outlier is influential.

In contrast, whether the outlier in the upper-right part of the scatterplot is included makes little difference in the fitted values. It happens to fall very near the line generated by the other fitted values. Deleting it does not change the fit a great deal. Therefore, it is not influential.

However, because the upper-left outlier increases substantially the variance of x without increasing the variance of the residuals very much, it helps to anchor the fitted values. Within the thought experiment of independent random samples of training data from a well-defined population, the fitted values will be more stable if values such as the upper-left outlier are present. Recall that to have substantial influence, an observation needs to be away from the mass of the data in the space defined by the predictors and also needs to have a large disparity between its fitted value of the response and its actual value. For an accessible discussion in the case of linear regression see Cook and Weisberg (1999: 360) and Peña (2005).

Now think about a set of bootstrap samples of the data. If there is an observation like the lower-right outlier, the fitted values will vary a great deal

depending on whether that influential observation happens to be in the sample. But then, averaging over bootstrap samples can help to stabilize the fitted values. This means that in the canonical thought experiment, the variance over random samples from the same population will be reduced. Bagging is doing just what it is supposed to do; the bagged results are an improvement.

The outlier observation in the upper right is not influential but helps to stabilize the fitted values. As a result, bootstrap sampling tends to destabilize the fitted values. When that outlier is by chance not included in the bootstrap sample, the fitted values derived from the other observations will tend to vary more over bootstrap samples. An observation that helps to anchor the fit is absent.

In practice, instances in which bagging can increase the variance sometimes can be spotted. A good place to start is with the univariate statistics for all predictors and the usual search for outliers and highly skewed or unbalanced distributions. Insofar as outliers can be excluded from the analysis on subject matter grounds (e.g., an observation is so atypical that it probably represents some kind data of error), the risks to bagging can be reduced. In the same spirit, highly skewed distributions might be transformed toward more symmetric distributions.

For highly unbalanced, categorical predictor variables with three or more classes, it can help to collapse classes. In the binary case, sometimes it is possible to combine two or more predictors in a way that still makes subject matter sense and restores some balance. For example, people with a PhD could be combined with people holding other advanced degrees to define a new variable equal to 1 if there is any post college education, and 0 otherwise. And there is always the option of dropping highly unbalanced predictors from the analysis.

However, problems with univariate distributions may not prove to have serious consequences. The predictor in question may not figure importantly in the fit because its relationship with the response is weak. Indeed, it may be excluded from the model altogether. In addition, the region in the predictor space where the instability is most manifest may be of little subject matter interest. For example, there may be little interest in the fitted values near the tails of the predictor distribution. Finally, where the mass of the data are, the impact of the instability may be modest. Thus in Figure 4.5, the two lines are much the same toward the middle of the distribution of x. In short, some trial and error can be useful before a final decision is made to exclude outliers.

There are extensions of conventional influence statistics that can be applied to bagging before the bagging begins (Grandvalet, 2004: 267–268). Although they have yet to be battled tested, they may be able to help in finding observations that are likely to be influential. But, the problem for bagging is somewhat different. One needs to find observations that ought to be influential because they are outliers in the space defined by the predictors, but that are actually not influential because they fall on the path of the fitted values constructed from the other observations. In Grandvalet's words, such

values provide "good" influence. Unfortunately, good influence leads to "bad" bagging.

The problems with bagging just described have their analogues for quantitative responses. Bagging is at its best when the problem to overcome is instability. Bagging when the fitted values are already very stable (or when the fitted values contain large amounts of bias) can make things worse. It is important to examine the data carefully before bagging is applied.

4.5.4 Losing the Trees for the Forest

Even when bagging performs as advertised, the price for averaging over trees can be high. There is no longer a single tree structure to interpret and, therefore, no tree diagram. Consequently, there is no direct way to consider how the inputs are related to the output. This is a very serious problem to which we return in the next chapter.

With no tree to interpret, the basic output from bagging is the predicted class for each case. Commonly there is an estimate of the classification error and a cross-tabulation of the classes predicted by the classes observed. This is nothing more than a confusion table but now based on averaging over trees. In addition, there can be separate error calculations for the different response classes, and a comparison of the number of false negatives to the number of false positives. If out-of-bag data are used, the confusion table is an even more honest representation of the results. Sometimes the software will store each of the trees as well, although these are rarely of any interest because the amount of information is typically overwhelming.

4.5.5 Bagging Is Only an Algorithm

Bagging may be seen less as an extension of CART and more as an illustration of what Breiman (2001b) calls "algorithmic modeling." Algorithmic models are computer algorithms designed to solve very particular data analysis problems. Linking inputs to outputs so that classification errors are small is a key example. Although there may also be an interest in describing how the inputs are linked to outputs, there is no effort to represent in the algorithm the mechanisms by which the linkage occurs. Thus, algorithmic models are not causal models. For researchers who want causal models, bagging is not the procedure.

4.6 An Example

Table 4.1 shows the bagged confusion table for the domestic violence data. Before bagging was applied, some CART results were examined to determine if in general CART might be appropriate for these data. Taking the empirical

distribution as the prior and using the default of equal costs for false negatives and false positives, CART seemed to help.

According to the bagged results in Table 4.1, there are 516 observations overall with .29 of them misclassified. About .15 of the households are incorrectly classified as having chronic domestic violence problems, and about .82 of the households are incorrectly classified as not having chronic domestic violence problems. The proportion of incorrect no DV classifications is .22, and the proportion of incorrect DV classifications is .75. Table 4.1 uses the out-of-bag (OOB) data to construct the fitted values, so the confusion table is more honest than the CART confusion tables in which the test data are the same as the training data. A confusion table was constructed from CART output using the same data, but with no bagging applied. The proportion of misclassified cases overall was .24, down from .29. CART was a bit too optimistic.

	Predict No DV	Predict DV	Model Error
No DV	347	60	.15
DV	89	20	.82
Use Error	.22	.75	Overall Error = .29

Table 4.1. Bagged CART confusion table for estimates of domestic violence.

4.7 Bagging a Quantitative Response Variable

Bagging works by the same general principles when the response variable is quantitative. Recall that CART constructs a regression tree by maximizing the reduction in the error sum of squares at each split. Each case is placed in a terminal node with a conditional mean. That mean is the predicted value for all cases of that terminal node.

All of the concerns about overfitting apply, especially given the potential impact that outliers can have on the fitting process when the response variable is quantitative. Recall that with the sum of squares fitting function, a few cases that fall a substantial distance from the mass of the data can produce results that do not characterize well the data on hand and do not generalize well either.

At the same time, overfitting is not always a problem. The consequences of overfitting can be unimportant if

1. The number of observations is large.
2. The number of predictors is small.
3. The number of terminal nodes is small.

4. There are no observations that fall some distance away from the mass of the data for the joint distribution of response variable and the predictors.

With a numerical response variable, bagging averages over trees in much the same way it averages over trees when the response variable is categorical. For each tree, each observation is placed in a terminal node and assigned the mean of that terminal node. Then, the average of these assigned means over trees is computed for each observation. This average value for each case is the bagged fitted value used. It is an average of conditional means for a large number of regression trees. The averaging process will tend to cancel out the impact of trees producing extreme conditional means and in so doing, helps to reduce the impact of overfitting. If for each tree, it is the OOB data that are placed in terminal nodes, the overfitting problems can be reduced even more.

As just noted however, overfitting is not necessarily a problem for CART analyses. To illustrate, the CART regression analysis undertaken earlier for high school grade point average was done again with bagged regression trees. Recall that there were approximately 8000 observations. This time eight predictors were used.

There was no evidence of outliers in the joint distribution of the predictors and the response. A series of bivariate scatterplots was first examined, and no apparent outliers were spotted. However, a series of bivariate plots is not the same as a single multivariate plot. So, the model implied by the CART results was re-estimated using linear regression with appropriate interaction terms. Then Cook's distance was computed for each observation. As expected, with so large a sample and relatively few predictors, no single observation stood out as problematic. Taken together, these two approaches are not iron clad proof that all is well, but make it a reasonable working premise.

It was not surprising, therefore, that bagging did not make an important difference. The root mean squared error (i.e., the standard deviation of the residuals) was .4136 for the CART results and .4132 for the bagged CART results. The grade point average response variable ranged from 1.0 to 5.0, therefore a difference of the root mean square error in the fourth decimal place is effectively noise.

4.8 Software Considerations

In R, the bagging procedure (i.e., bagging() in the *ipred* library) can be applied to classification, regression, and survival trees. The arguments from these procedures can be passed to ipred(). For example, one set the prior in rpart() to take misclassification costs into account, and this information is used in ipred(). The library was written by Andrea Peters (Andrea.Peters@imbe.imed.uni-erlangen.de) and Torsten Hothorn (Torsten.Hothorn@rzmail.uni-erlangen.de). The package maintainer is Andrea Peters. Perhaps the key concern is that when confusion tables are constructed, or when other measures of

performance are computed, one must be clear on what is being done. There are at least three possibilities.

1. Trees are bagged as usual, and bagged classifications or bagged conditional means constructed. These are then compared to the actual classifications or response variable values in the original data from which bootstrap samples were drawn.
2. Trees are bagged as usual, and bagged classifications or bagged conditional means constructed. The software stores the predictor values leading to each terminal node. New data (from a test sample) are dropped down the bagged tree and assigned to terminal nodes based on the values of their predictors. Each of the new observations is assigned a class or conditional mean determined by the class or conditional mean of the terminal node in which it lands. These fitted values for the new data are compared to the actual values of the response in the new data.
3. CART is used to grow a single tree using a bootstrap sample of the data. As usual, classifications or conditional means are constructed for each terminal node. The set of predictor values leading to each terminal node is stored. Observations not included in the bootstrap sample are noted. These are the out-of-bag observations for that tree. The out-of-bag observations are then dropped down the tree and assigned the class or conditional mean of the terminal node in which they land. The same process is repeated for a number of trees. When the votes are cast to determine class membership or when conditional means over trees are averaged, the only trees considered for a given observation are the ones for which that observation was not in the training data. That is, for a given case i, the only trees that count are the trees for which case i was not used (i.e., it was among the out-of-bag observations). Generally about one-third of the original data are not chosen to be included in each training sample. Even with a relatively small number of trees, therefore, each of the observations will have several votes or conditional means to average.

The third method was used for analysis of grade point average, just reported. The second and third methods are generally more honest than the first because the separation between training data and the test data is more complete. But all three methods are often better than no bagging at all. The drawback to the second method is that a second random sample from the same population is needed. The drawback to the third method is that it will often be useful to construct a larger number of trees because for each tree, only about a third of the data figure in the voting. For example, although 25 trees may be good enough for methods one and two, 100 trees may be a good number for method three. But usually, 100 trees will not be a prohibitive computational burden.

A rather different set of issues can sometimes be raised by the bootstrap sampling process. In particular, a given sample may produce predictors or response variables that are constants. This is more likely when predictor or

response observations are categorical and unbalanced. For example, if in a sample of 300 inmates only 30 are Asian-Americans, a bootstrap sample may include no Asian-Americans whatsoever.

The problem for the software is what to do in such situations. One approach would be to discard samples in which any of the variables were constants. Another approach would be to throw out the offending variables. However, discarding variables is not an option if the response variable is one of the set. In any case, the worst outcome is for the software to crash. It can be well worth the time to read the software documentation especially carefully if there are highly unbalanced variables in the dataset.

4.9 Summary and Conclusions

Bagging is an important conceptual advance and a useful tool in practice. The conceptual advance is to aggregate fitted values from a large number of bootstrap samples. Ideally, many sets of fitted values, each with low bias but high variance, may be averaged in a manner than can effectively reduce the bite of the bias–variance tradeoff. Thanks to bagging, there can be a way to usefully address this long-standing dilemma in statistics. Moreover, the ways in which bagging aggregates the fitted values is the basis for other statistical learning developments.

In practice, bagging can generate fitted values that often reproduce the data well and forecast with considerable skill. Both masters are served without making unrealistic demands on available computing power. Bagging can also be usefully applied to a wide variety of fitting procedures.

But bagging also suffers from several problems. Perhaps most important, there is no way within the procedure itself to depict how the predictors are related to the response. One can obtain a more honest set of fitted values and a more honest evaluation of how good the fitted values really are. But as a descriptive device, bagging is pretty much a bust. Other tools are needed, which are considered in the next chapter.

A second problem is that because the same predictors are available from tree to tree, the sets of fitted values are not fully independent. The averaging is not as effective as it could be if the sets of fitted values were closer to independent. This too is addressed shortly.

Third, bagging can stumble badly if the fitting function is consistently and substantially inappropriate. Large and systematic errors in the fitted values are just reproduced a large number of times and do not, therefore, cancel out in the averaging process. For categorical response variables, bagging a very weak classifier can sometimes make things worse.

Fourth, the bootstrap sampling can lead to problems when categorical predictors or outcomes are highly unbalanced. For any given bootstrap sample, the unbalanced variable can become a constant. Depending on the fitting function being bagged, the entire procedure may abort.

Finally, bagging can actually increase instability if there are outliers that help to anchor the fit. Such outliers will be lost to some of the bootstrap samples. Bagging can be extended so that many of these problems are usefully addressed, even if full solutions are not available. We turn to some of these solutions in the next chapter. And in their potential solutions is found another form of statistical learning, still farther away from conventional regression analysis.

Exercises

Problem Set 1

The goal of this first exercise is to compare the performance of linear regression, CART, and bagging applied to CART. Construct the following data set in which the response is a quadratic function of a single predictor.

```
x1=rnorm(500)
x12=x1^2
y=1+(2*(x12))+(2*rnorm(500))
```

1. Plot the $1 + (2 \times \text{x12})$ against x1. This is the "true" relationship between the response and the predictor without the complication of the disturbances. This is the $f(X)$ you hope to recover from the data.

2. Proceed as if you know that the $f(X)$ is quadratic. Fit a linear model with x12 as the predictor. Then plot the fitted values against x1. You can see how well linear regression does when the functional form is known.

3. Now suppose that you do not know that the $f(X)$ is quadratic. Apply linear regression to the same response variable using x1(not x12) as the sole predictor. Construct the predicted values and plot the fitted values against x1. How do the fitted values compare to what you know to be the correct $f(X)$? (It is common to assume the functional form is linear when the functional form is unknown.)

4. Apply CART to the same response variable using rpart() and x1(not x12) as the sole predictor. Use the default settings. Construct the predicted values, using predict(). Then plot the fitted values against x1. How do the CART fitted values compare to what you know to be the correct $f(X)$? How do the CART fitted values compare to the fitted values from the linear regression with x1 as the sole predictor?

5. Apply bagging to the same response variable using ipred() and x1 as the sole predictor. Use the default settings. Construct the predicted values using predict(). Then plot the fitted values against x1. How do the bagged fitted values compare to the linear regression fitted values?

6. You know that the relationship between the response and x1 should be a smooth parabola. How do the fitted values from CART compare to the fitted values from bagging? What feature of bagging is highlighted?

Problem Set 2

Load the dataset "Freedman" from the *car* library. For 100 American cities, there are four variables: the crime rate, the population, population density, and proportion nonwhite. As before, the crime rate is the response and the other variables are predictors.

1. Use rpart() and its default values to fit a CART model. Compute the root mean square error for the model. One way to do this is to use predict.rpart() to obtain the fitted values and with the observed values for the variable "crime," compute the root mean square error in R. Then use bagging() from the library *ipred* and the out-of-bag observations to obtain a bagged value for the root mean square error for the same CART model. Compare the two estimates of fit and explain why they differ.

2. Using sd(), compute the standard deviation for the CART fitted values and the bagged fitted values. Compare the two standard deviations and explain why they differ.

Problem Set 3

Load the dataset "frogs" from the library *DAAG* Using "pres.abs" as the response build a CART model under the default settings.

1. Construct a confusion table with "pres.abs" and the predicted classes from the model. Now, using bagging() from the library *ipred*, bag the CART model using the out-of-bag observations. Construct a confusion table with "pres.abs" and the bagged predicted classes from the model. Compare the two confusion tables and explain why they differ.

2. Cross-tabulate using table() or xtab() the fitted classes from CART and the bagged CART. Examine the two cells for cases in which the two sets of fitted classes do not agree. Why is the number of observations in each about the same?

5

Random Forests

5.1 Introduction and Overview

Just as in bagging, imagine constructing a large number of trees with bootstrap samples from a dataset. But now, as each tree is constructed, take a random sample of predictors before each node is split. For example, if there are twenty predictors, choose a random five as candidates for defining the split. Then construct the best split, as usual, but selecting only from the five chosen. Repeat this process for each node. And as in bagging, do not prune. Thus, each tree is produced from a random sample of cases, and at each split a random sample of predictors. Finally, just as in bagging, classify by a majority vote of the full set of trees. Breiman calls the set of such trees a "random forest" (Breiman, 2001a).

The random forest algorithm is, therefore, very much like the bagging algorithm. Again let N be the number of observations and assume for now that the response variable is binary.

1. Take a random sample of size N with replacement from the data.
2. Take a random sample without replacement of the predictors.
3. Construct the first CART partition of the data.
4. Repeat Step 2 for each subsequent split until the tree is as large as desired. Do not prune.
5. Drop the out-of-bag data down the tree. Store the class assigned to each observation along with each observation's predictor values.
6. Repeat Steps 1–5 a large number of times (e.g., 500).
7. Using only the class assigned to each observation when that observation is not used to build the tree, count the number of times over trees that the observation is classified in one category and the number of times over trees it is classified in the other category.
8. Assign each case to a category by a majority vote over the set of trees. Thus, if 51% of the time over a large number of trees a given case is classified as a "1," that becomes its estimated classification.

R.A. Berk, *Statistical Learning from a Regression Perspective*,
DOI: 10.1007/978-0-387-77501-2_5,

5.1.1 Unpacking How Random Forests Works

It should be clear that random forests draws on many features of procedures discussed in the last two chapters. To begin, random forests uses CART as a key building block. An important benefit is that one can capitalize on CART's strengths and flexibility. For example, large trees can be effective tools for reducing bias, and the averaging over trees can substantially reduce instability that might otherwise result. In addition, the relative costs of false negatives and false positives can be explicitly considered. Especially for policy-related applications, this can be vital.

It should also be apparent that random forests is bagging, but more so. By working with a random sample of predictors at each possible split, the fitted values across trees are more independent. Consequently, the gains from averaging over a large number of trees can be more dramatic. But there is more to the story.

If the individual trees in a random forest are unbiased in their fitted values and estimated splits, the gains from random forest are solely with respect to the variance. In practice, of course, there is no way to know if this is true and there are usually lots of reasons for skepticism. Sometimes, therefore, random forests can be seen as helping to reduce the bias.

Perhaps most directly, random forests is able to work with a very large number of predictors, even more predictors than there are observations. In addition to conventional regression modeling, all of the statistical learning procedures considered thus far have required that the number of predictors be less than the number of observations (usually much less). An obvious gain with random forests is that more information may be brought to bear on the fitting process. More predictors can weigh in, which can reduce bias. Variables that should play a role and that otherwise would have been excluded, can participate.

A more subtle gain is that different sets of predictors can be evaluated for different splits so that different "models" can be applied as needed. To appreciate how this works recall the CART splitting criterion:

$$\Delta I(s, A) = I(A) - p(A_L)I(A_L) - p(A_R)I(A_R), \tag{5.1}$$

where $I(A)$ is the value of the parent impurity, $p(A_R)$ is the probability of a case falling in the right daughter node, $p(A_L)$ is the probability of a case falling in the left daughter node, $I(A_R)$ is the impurity of the right daughter node, and $I(A_L)$ is the impurity of the left daughter node. CART tries to find the predictor and the split for which $\Delta I(s, A)$ is as large as possible.

The key point is that the usefulness of a split is a function of the two new impurities and the probability of cases falling into either of the prospective daughter nodes. Suppose there is a predictor that could produce splits in which one of the daughter nodes is very homogeneous but has relatively few observations whereas the other node is quite heterogeneous but has relatively many observations. Suppose there is another predictor that could generate

two nodes of about the same size, each of which is only moderately homogeneous. If these two predictors were forced to compete against each other, the second predictor might well be chosen, and the small local region that the first predictor would address be ignored. However, if the second predictor were not in the pool of competitors, the first might be selected instead.

Similar issues arise with predictors that are substantially correlated. There may be little difference empirically between the two so that when they compete to be a splitting variable, one might be chosen almost as easily as the other. But they would not partition the data in exactly the same way. The two partitions that would be defined would largely overlap. But each partition would have unique content as well. The unique content defined by the predictor not chosen would not be included in that step. Moreover, with the shared area now removed from consideration, the chances that the neglected predictor would be selected later would be significantly reduced. But if the two variables each had an opportunity to be selected without competing against each other, each might be able to contribute.

Both kinds of competitions are in practice likely to involve many variables, especially if there are a large number of predictors. Then, there can be a few predictors that in a procedure such as CART will dominate the fitting process because on the average they consistently perform just a bit better than their competitors. Consequently, many other predictors, which could be useful for very local features of the data, are rarely selected as splitting variables.

The same issues can arise for a single predictor. Recall that in CART, all predictors are evaluated, even predictors previously selected. The same predictor may be selected more than once. But each new selection for a given predictor implies the construction of a new basis function for that predictor (i.e., a different break point). In the competition between all predictors at each stage, basis functions that might be very important, but only for a small fraction of the data, risk being overlooked.

With random forests computed for a large enough number of trees, each predictor will have at least several opportunities to be the predictor defining a split. And in those opportunities, it will have very few competitors. Moreover, if there are a relatively small number of dominant predictors, much of the time a dominant predictor will not be included. As a result, predictors that might ordinarily be overlooked have the opportunity to contribute to the fit. The same implications follow for different basis functions for given predictors. With a changing mix of competitors, highly specialized basis functions may have the opportunity to define a split.

The sampling of predictors also has beneficial effects for the variance because of the averaging over trees. In effect, the predictor sampling and averaging leads to shrinkage of the impact of each predictor on the fitted values. The impacts on the fitted values when a predictor is included are averaged with the impacts on the fitted values when that predictor is not included. This regularization can help to reduce the variance beyond what would follow solely from the sampling of observations.

To help fix these ideas, Figure 5.1 shows a partitioning diagram much like the one used earlier when CART was first introduced. As before, there are two predictors, x and z. There is a choice to be made between the vertical split on z represented by the solid line and the horizontal split on x represented by the dashed line. The horizontal split might not be selected because its one very homogeneous partition has relatively few observations. But if the predictor z were not available, the horizontal split shown might well be chosen.

Very much the same issues arise if the vertical split is made, but then the choice is between a horizontal split and another vertical split, which would imply a second basis function for z. Consider the right vertical partition, for example. The potential horizontal split shown would have to compete against all possible vertical splits of z. But if z were not among the competitors for the second split, the horizontal split might well be chosen.

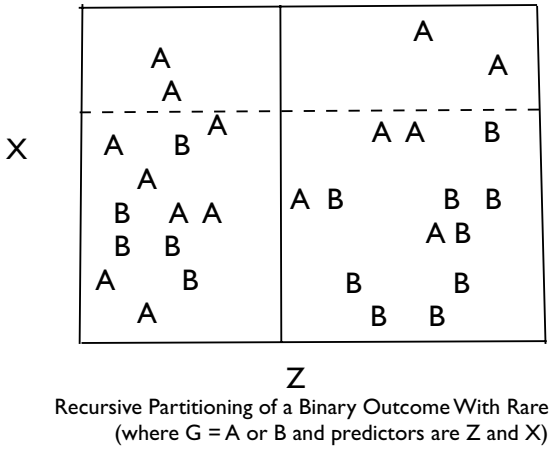


Fig. 5.1. Partitioning for some rare cases.

Consider now Figure 5.2, a reproduction of Figure 3.8. This figure was used earlier to illustrate CART instabilities when two predictors were highly correlated. Here the point is that whether the partition defined by the dashed line or the partition defined by the dotted line were selected, it would make no difference for the three As in the upper-right hand corner. But whether the fourth A, just below the dashed line, would be included depends on which partition won in the fitting competition. If the horizontal partition were to win, the fourth A would be placed in a very heterogeneous grouping. If the vertical partition were to win, the fourth A would be placed in a very homogeneous

grouping. And if the former, the fact that the fourth A was very similar to its three A neighbors would probably be lost to the analysis. With the three As already in their own partition, it is unlikely that another partition would be defined later which was also very high on x and z. If x were not available when the second split were determined, the vertical slice shown might well be the one selected.

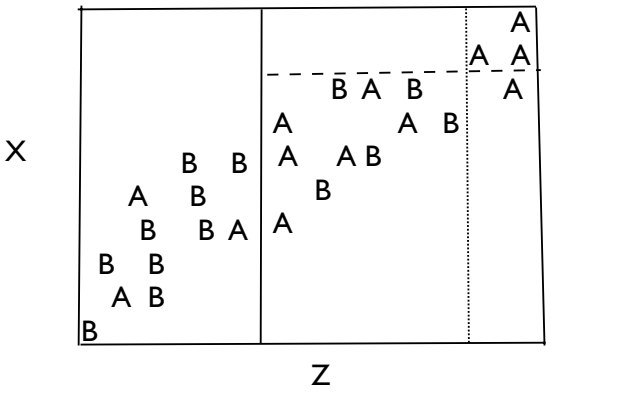


Fig. 5.2. Competition between highly correlated predictors

The goal of finding a role for highly specialized predictors is an argument for growing very large, unpruned trees. Generally, this seems to be a wise strategy. However, large trees can sometimes lead to very unstable results when there are a substantial number of predictors that at best are weakly related to the response and correlated substantially with one another (Segal, 2003). In effect, this becomes a problem with multicolinearity that the averaging over trees becomes more difficult. The instability is too large to be readily averaged out. In practice, therefore, it can occasionally be useful to work with smaller trees, especially when there are a large number of weak predictors that are strongly associated with one another. If before a data analysis is begun there are reasons to worry about such problems, tree size can be used as a tuning parameter in yet another manifestation of the bias–variance tradeoff.

Geurts and his colleagues (2006) have proposed another method for selecting predictors for enhancing independence across trees and further opening up the predictor competition. They do not build each tree from a bootstrap sample of the data. Rather, for each random sample of predictors, they select

splits for each predictor at random (with equal probability), subject to some minimum number of observations in the smaller of the two partitions. Then, as in random forests, the predictor that reduces heterogeneity the most is chosen to define the two subsets of observations. They claim that this approach will reduce the overall heterogeneity at least as much as other ensemble procedures without a substantial increase in bias. However, this conclusion would seem to depend on how good the predictors really are. Moreover, if one is interested in interpreting the manner in which inputs are related to outputs, their method risks serious subject matter errors. In the averaging process over trees, model results characterized by optimal splits are weighted the same as model results characterized by random spits.

In summary, with forecasting accuracy as a criterion, bagging is in principle an improvement over CART. And by this same criterion, random forests is in principle an improvement over bagging. Indeed, random forests is among the very best classifiers invented to date (Breiman, 2001a). A key reason is the ability to consider a very large number of predictors, even more predictors than observations. This can lead to reductions in the bias and reductions in the variance.

5.2 An Initial Illustration

Table 5.1 shows some results for the domestic violence data described earlier. As before, there are a little over 500 observations, and even if just double interactions are considered, well over 100 predictors. This time, the goal is not to forecast new calls for service to the police department that likely involve domestic violence, but only those calls in which there is evidence that felony domestic violence has actually occurred. Such incidents represent about 4% of the cases. They are very small as a fraction of all domestic violence calls for service. And as such, they would normally be extremely difficult to forecast with better skill than using the marginal distribution of the response alone. One would make only four mistakes in 100 households if one classified all households as not having new incidents of serious domestic violence.

Using the response variable as the only source of information would in this case mean never correctly identifying serious domestic violence households. The policy recommendation might be for the police to assume that the domestic violence incident to which they had been called would be the last serious one for that household. This would almost certainly be an unsatisfactory result, which implies that there are significant costs from false negatives.

Using a cost ratio of 10 to 1 for false negatives to false positives favored by the police department (more on how to do that shortly), Table 5.1 shows that random forests incorrectly classifies households 13 times out of 100 overall. If equal costs were used and the empirical distribution of the response variable taken as the prior distribution, random forests would likely do at least as well as the marginal distribution (i.e. four mistakes per 100 households).

But whenever costs other than equal ones are introduced, the overall error proportion will increase, so this result is no surprise. It is also not a problem.

More instructive measures of performance are found in the row proportions. Random forests manages to correctly identify the very rare serious domestic violence households about half the time with a model error of only .30 for households without these problems. As one would expect, when a logistic regression was applied to the data, not a single incident of serious domestic violence was identified, either correctly or incorrectly. The logistic regression performed no better than the marginal distribution of the response.

	No DV Forecasted	DV Forecasted	Model Error
No DV	341	146	.30
DV	15	14	.51
Use Error	.04	.91	Overall Error = .13

Table 5.1. Confusion table for the ten-to-one random forest model for new domestic violence incidents.

The use errors (i.e., column proportions) also look promising. When no future incidents of domestic violence are forecasted, that forecast is correct about 96 times out of 100. When future incidents of domestic violence are forecasted, that forecast is correct about 1 time in 10. Although that might seem to be disappointing, it is a reflection of the costs assigned. The results imply that the police department is prepared to live with nine false positives for every true positive. If that tradeoff is indeed acceptable, then the forecasting exercise works as intended.

5.3 A Few Formalities

With some initial material on random forests behind us, it is useful to take a bit more formal look at the procedure. We build on an exposition by Breiman (2001a). The concepts considered make more rigorous some ideas that we have used in the past two chapters, and provide important groundwork for material to come. As before, we focus on categorical, and especially binary, response variables.

We also need to change notation just a bit. Bold type is used for vectors and matrices. Capital letters are used for random variables.

5.3.1 What Is a Random Forest?

With categorical response variables, a random forest is a classifier. More than two classes can be used. The intent is to assign classes to observations using

information contained in a set of predictors. A random forest is constructed from a set of K classification trees, each based in part on chance mechanisms.

We formally represent the random forest classifier as a collection of tree-structured classifiers $\{f(\mathbf{x}, \Theta_k), k = 1, \dots\}$, where $\mathbf{x}$ is an input vector of P predictor values used to assign a class, and k is an index for a given tree. Each Θ_k is a random vector constructed for the kth tree so that it is independent of past random vectors $\Theta_1, \dots, \Theta_{k-1}$, and is generated from the same distribution. For bagging, it is the means by which observations are selected at random with replacement from the training data. For random forests, it is also the means by which subsets of predictors are sampled without replacement for each potential split. In both cases, Θ_k is a collection of integers. Integers for both sampling procedures can be represented by Θ_k.

But we are getting ahead of ourselves. The output from a given classifier is an assigned class for each observation, determined by the input values $\mathbf{x}$. In CART, for example, the class assigned to an observation is the class associated with the terminal node in which an observation falls. With random forests, the class assigned to each observation is determined by a vote over the set of tree classifiers for which that observation was not part of the training dataset. That is, classes are assigned to observations much as they are in bagging. It is conceptually very important to distinguish between the class assigned by the kth classifier and the class ultimately assigned by a vote.

5.3.2 Margins and Generalization Error for Classifiers in General

Suppose there is a training dataset with predictor values and associated values for a categorical response. A training dataset has been drawn at random. This means that the data on hand can be treated as a realization of two kinds of random variables: a set of predictors and a response variable.

Consider now a single data point from the training data. If the training data have, for instance, 250 observations and 20 variables, that data point might be the 27th row in a 250 by 20 data matrix. The values in this row will change from sample to sample. Consequently, the set of P predictor values can be represented by the random variable $\mathbf{X}$. The actual class for that data point will be represented by the random variable Y.

Suppose there is an ensemble of K classifiers, $f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_K(\mathbf{x})$. For the moment, we do not consider how these different classifiers are constructed. The margin function at the data point $\mathbf{X}, Y$ is then defined as

$$mg(\mathbf{X}, Y) = av_k I(f_k(\mathbf{X}) = Y) - \max_{j \neq Y} av_k I(f_k(\mathbf{X}) = j), \tag{5.2}$$

where $I(.)$ is an indicator function, j is an incorrect class, av_k denotes averaging over the set of classifiers for a single realized data point, and max denotes the largest value.

For a given set of x-values and the associated observed class, the margin is the average number of votes for the correct observed class minus the maximum

average number of votes for any other incorrect class. Thus, the margin is the smallest spread between the number of correct and incorrect votes for a given data point. When there are only two classes, there is only one spread to worry about, and the word "maximum" is unnecessary.

From the definition of the margin function, generalization error is then

$$g = P_{\mathbf{X},Y}(mg(\mathbf{X}, Y) < 0), \tag{5.3}$$

where P means probability (easily confused with p, which can be the number of predictors.

The probability is over the space represented by the random variables $\mathbf{X}, Y$, so that generalization error addresses what happens to the margin over different realizations of the data point. Generalization error, therefore, is the probability over realizations of the data point that the classification vote will be overturned (i.e., go from positive to negative), and the assigned class changed. A low probability indicates the class assigned is likely to be stable over random samples from the same population.

This definition can be somewhat confusing. Generalization error, sometimes called "prediction error"or "test error," is more typically defined as the expected loss over all sources of randomness built into the full set of fitted values. Then, generalization error can be written as

$$E[L(Y, f(\mathbf{X}))]. \tag{5.4}$$

The loss function $L(Y, f(\mathbf{X}))$, is derived from disparities between the values of the response predicted and the values of the response observed. Sometimes a "hat" is placed over the "f" to indicate that it is an estimated function.

Here, we are using a "1–0 loss" because the assigned class is compared to the observed class via the indicator function. It would also be possible to compare the observed class to the predicted probability of membership in that class. Then, a popular loss function is the deviance. But using the deviance is more appropriate when constructing a given classifier, and if used instead of the 1–0 loss here would fundamentally change random forests.

5.3.3 Generalization Error for Random Forests

Now, suppose the classifier $f_k(\mathbf{X}) = f(\mathbf{X}, \Theta_k)$; the classifier is a random forest. Breiman proves (2001a) that as the number of trees increases, the estimated generalization error converges to the population generalization error, which is

$$P_{\mathbf{X},y}(P_{\Theta}(f(\mathbf{X}, \Theta) = Y) - \max_{j \neq Y} P_{\Theta}(f(\mathbf{X}, \theta) = j) < 0). \tag{5.5}$$

The arguments for $P_{\mathbf{X},Y}$ are (1) the probability of the correct classification over trees, and (2) the maximum probability over trees of a wrong classification. The number of these trees increases without limit. Then the estimated

generalization error converges to the probability over $\mathbf{X}, Y$ that a vote will be overturned.

There is a lot going on here. The data used for a given tree are a bootstrap sample of the training dataset. The training dataset is a realization of the random variables. Thus, two different chance mechanisms are involved, the first reflected in P_{Θ} and the second reflected in $P_{\mathbf{X},Y}$.

The importance of the convergence is that demonstrably random forests does not overfit as more trees are grown. One might think that with more trees one would get an increasingly false sense of how well the results would generalize. Breiman proves that this is not true. Given all of the concern about the problem of overfitting, this is an important result.

It is also important to appreciate the limitations of what is being claimed. The convergence is to a value for the generalization error in the population. (This implies that the data are a random sample from that population or a random realization from a specified stochastic process.) This says nothing about the quality of the population classifier responsible for that generalization error. That classifier could well generate fitted values some distance from the actual response variable's observations, and these fitted values could contain substantial systematic error. As a result, the target generalization error could be some distance from the "true" generalization error were the classifier in the population the "correct" classifier (Traskin, 2008). In short, Breiman does not prove that the classifier estimated from the data is a consistent estimator of the $f(X)$, even if one has access to all of the necessary predictors perfectly measured. We return to this point in a slightly different context shortly.

As noted earlier, however, if there is no postulated $f(X)$, or if the data analyst is prepared to accept that with the data available the $f(X)$ will not be well estimated, then concerns about whether the population classifier is "correct" are moot. Breiman's proof remains relevant nevertheless because one might otherwise believe that growing lots of trees in order to assemble a random forest would lead to overfitting.

5.3.4 The Strength of a Random Forest

The margin function for a given realized data point in a random forest (not just any classifier) is defined as

$$mr(\mathbf{X}, Y) = P_{\Theta}(f(\mathbf{X}, \Theta) = Y) - \max_{j \neq Y} P_{\Theta}(f(\mathbf{X}, \Theta) = j), \tag{5.6}$$

where $f(\mathbf{X}, \Theta)$ is a set of classifications that varies because of the chance mechanism represented by Θ. Thus, considering all possible trees and that realized data point, the margin function of a random forest is the probability that a classification will be correct minus the maximum probability that it will be incorrect.

The margin function for a given random forest takes the training data as fixed. If one allows for different realizations of the data point, the margin function for a random forest will vary from realization to realization. But if we take the expected value of Equation 5.6, over realizations of the data point, the strength of a random forest is defined as

$$s = E_{\mathbf{X},\mathbf{y}} mr(\mathbf{X}, \mathbf{y}). \tag{5.7}$$

Thus, the strength of a random forest is essentially the average margin over randomly drawn training data. Clearly, the larger this expected value is, the better.

5.3.5 Dependence

For a given data point, the trees from which a random forests is constructed differ from each other because of the chance mechanism represented in Θ_k. Recall that for both the selection of observations in the bootstrap sample and the selection of potential predictors at each split, the chance mechanism generates a set of integers.

For the bootstrap sample, the chance mechanism generates the number of times each observation in the training data is selected. For the predictors, the chance mechanism generates integers denoting which predictors are chosen. For example, if there are 10 predictors $1, \ldots, 10$, and a decision is made to select three of them, the integers 1, 3, and 8 might be drawn at random.

Ideally, both chance mechanisms should perform so that the output from each tree is as independent as possible. That allows the averaging to occur most effectively. But what output in particular?

For the binary outcome case, there is a relatively straightforward result. Draw an observation from the relevant population. For any one tree, classify that realized data point. If the classification is correct, record a 1. If the classification is incorrect, record a 0. Repeat the process beginning with random sampling of a single observation. The average correlation between these coded values, which represent whether the classification is correct, over samples and trees, is the appropriate measure of dependence. Ideally, this correlation should be small. The probability that a given observation chosen at random is classified correctly should be unrelated to whether any other randomly chosen observation is classified correctly.

5.3.6 Implications

The central concept in this more technical discussion is the margin. From the margin comes the definition of generalization error. Generalization error, in turn, depends on the strength of the collection of classifiers and the dependence between them. When the dependence between trees is small and

the strength of the trees is large, the generalization error will be small. More precisely, Breiman shows that the upper bound for the generalization error is

$$g^* = \frac{\bar{\rho}(1 - s^2)}{s^2}, \tag{5.8}$$

where $\bar{\rho}$ is the average correlation over realizations of the data and trees, and s is the strength of the set of trees. We want the former to be small and the latter to be large.

In addition, generalization error will not increase with the number of trees. Indeed, more trees are generally better than fewer trees because the true value of the generalization error will be more accurately approximated. In practice, this means that unless one is limited by the computer being used, a random forests can usefully include several thousand trees.

Another practical lesson is that a useful random forest classifier will on the average produce large margins as observations are classified. Put another way, one should have less confidence in random forest results when the margins tend to be small. This point and related ones have been made informally in earlier discussions.

A final lesson is that the random selection of predictors helps to make random forests more desirable than bagging because dependence is reduced. Other advantages of random forests are considered shortly.

5.4 Random Forests and Adaptive Nearest Neighbor Methods

A conceptual link was made earlier between CART and adaptive nearest neighbor methods. Not surprisingly, similar links can be made between random forests and adaptive nearest neighbor methods. But for random forests, there are a number of more subtle issues (Lin and Jeon, 2006). These are important not just for a deeper understanding of random forests, but for some extensions of random forests considered at the end of this chapter.

Recall that in CART, each terminal node represented a region of nearest neighbors. The boundaries of the neighborhood were constructed adaptively when the best predictors and their best splits were determined. With the neighborhood defined, all of the observations inside were used to compute a mean or proportion. This value became the measure of central tendency for the response within that neighborhood. In short, each terminal node and the neighborhood represented had it own conditional mean or conditional proportion. Figure 3.1 might be a useful memory refresher.

Consider the case in which equals costs are assumed. This makes for a much easier exposition, and no key points are lost. The calculations that take place within each terminal node, in effect, rely on a weight given to each value of the response variable. (Meinshausen, 2006; Lin and Jeon, 2006). For a given

terminal node, all observations not in that node play no role when the mean or proportion is computed. Consequently, each such observation has a weight of zero. For a given terminal node, all of the observations are used when the mean or proportion is computed. Consequently, each value of the response variable in that node has a weight equal to $1/n_\tau$, where n is the number of observations in terminal node τ. Once the mean or proportion for a terminal node is computed, that mean or proportion can serve as a fitted value for all cases that fall in that terminal node.

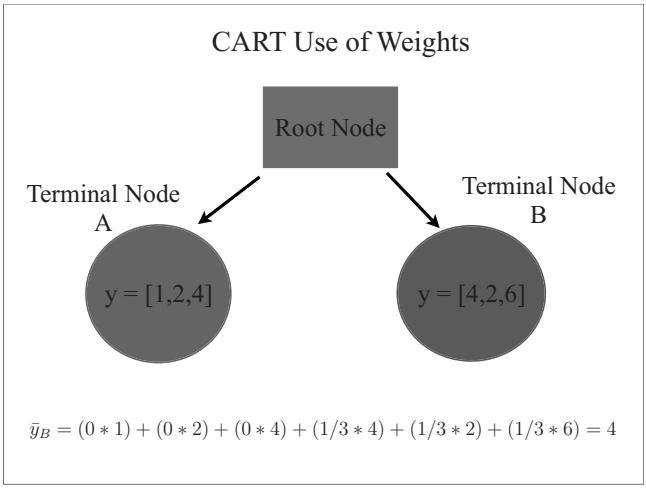


Fig. 5.3. CART weighting.

Figure 5.3 shows a toy rendering of this account. The tree has but a single partitioning of the data. There are three values of the response variable in each terminal node. Consider terminal node B. The mean for terminal node B is 4, computed with weights of 1/3 for the values in that node and weights of 0 otherwise. Each of the three observations landing in terminal node B can be assigned a value of 4 as their fitted value. If the response variable had been binary, the numbers in the two terminal nodes would have been replaced by 1s and 0s. Then a conditional proportion for each terminal node would be the outcome of the weighted averaging. And from this an assigned class could be determined as usual.

A bit more formally, a conditional mean or proportion for any terminal node τ is

$$\bar{y}_\tau | x = \sum_{i=1}^{N} w_{(i,\tau)} y_i, \tag{5.9}$$

where the sum is taken over the entire training dataset, and w_i is the weight for each y_i. The sum of the weights over all observations is 1.0. In practice, most

of the weights will be zero because they are not associated with the terminal node in question. This is really no different from the manner in which nearest neighbor methods can work when summary measures of a response variable are computed.

It may be important to underscore that each observation in the training dataset has a set of weights, one weight for each y_i. So, if there are 150 observations in the training data, there will be 150 weight values. It is these weights and y-values from which the mean or proportion for each terminal node is computed. Then each observation can have one fitted value depending on the node in which it comes to rest.

When a classification or regression tree becomes part of a random forest, there are a large number of trees with which to contend. For a given observation, there will now be a set of weights produced by each tree. Once again, most of the weights will be 0, with the rest fractions. Because of the stochastic nature of each tree, the weights will vary.

Consider a given observation $\mathbf{x}_0$ characterized by a set of predictor values and a value for the response variable. From tree to tree, the observations in the terminal nodes in which $\mathbf{x}_0$ falls will vary. Consequently, the node mean will likely vary as well. It follows that the set of weights used to compute the fitted value for $\mathbf{x}_0$ will change. In effect, random forests averages these weights for $\mathbf{x}_0$ (and for all other observations). Then, the average weight replaces the tree-specific weight in Equation 5.9. Weights greater than zero are sometimes called "voting points" when the mean or proportion for $\mathbf{x}_0$ is computed (Lin and Jeon, 2006: 579–580). The sum of the average weights over all observations is 1.0. Clearly, the links to adaptive nearest neighbors remain. It is just that each weight is an average weight over trees.

To help firm up these ideas, Table 5.2 carries on with the toy example for a single case. There are six observations in the dataset represented in the table as columns. A very small random forest of three trees is grown. To keep the table format manageable, assume that despite sampling with replacement, each observation appears just once in each bootstrap sample. The cell entries for the first three rows are the CART weights for each of the three trees. The last row is the average of each column. Each row sums to 1.0 and the average weights sum to 1.0 (except for rounding error).

Tree	i=1	i=2	i=3	i=4	i=5	i=6
1	0	.33	.33	0	.33	0
2	.5	.5	0	0	0	0
3	.25	.25	0	.25	0	.25
Average	.25	.36	.11	.083	.11	.083

Table 5.2. Weights from three random trees for a single observation.

Table 5.2 contains three sets of CART weights and their average for a given observation. When in random forests the fitted value is computed for that observation, the average weights in the last row are used. Each value of $y_i, i = 1, 2, \ldots, 6$, is multiplied by its weight and summed. There would be a corresponding table for each of the six observations in the dataset.

There are at least three interesting implications that follow from formulating the random forest fitted values in terms of weights. First, a plot of the weights in the space defined by the predictors can be instructive. To begin, consider a given $\mathbf{x}_0$ and all of the nonzero weights for all of the observations associated with that target point. Then, find each x-value in the predictor space and mark that point with a symbol for the weight. Figure 5.4 is an example for CART in which just two predictors are shown: age and income. The response might be years of education. For the moment, assume that age and income are unrelated. For ease of exposition, we can ignore the other predictors used in the analysis such as gender or ethnicity. For simplicity, we also ignore the weight values.

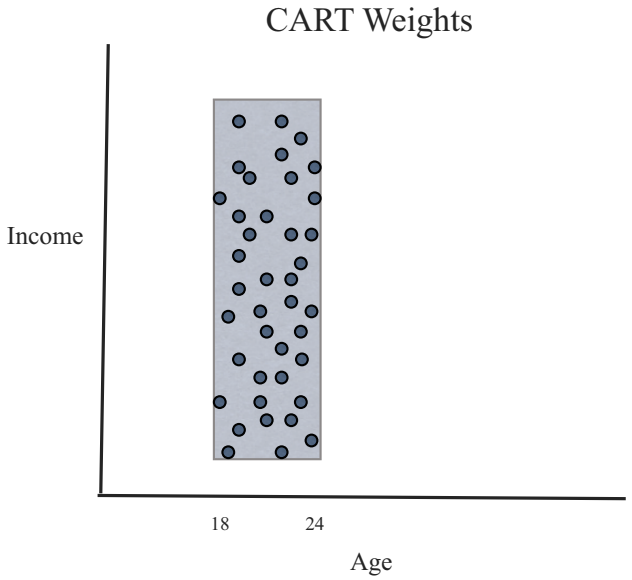


Fig. 5.4. Weights with two unrelated predictors.

Predictors that define the terminal node in which $\mathbf{x}_0$ falls will have all of nonzero weights (1) clustered along the dimensions of $\mathbf{x}$ used to define $\mathbf{x}_0$'s terminal node and (2) spread out along the dimensions of $\mathbf{x}$ not used to define

$\mathbf{x}_0$'s terminal node. Suppose, for example, age alone is used to define the terminal node. The terminal node only includes individuals between 18 and 24 years of age. Then, the plotted weights would be found within the range of 18 to 24 years old.

For Figure 5.4, income is not a defining feature of the node; it was not used to define the relevant splits. Then, the weights would be spread out over its range. Income is not by itself important for how the terminal node is defined and plays no systematic role in how the node mean or proportion is computed. In other words, such plots are characterized by tight clustering along some dimensions and little clustering at all along others, much as in Figure 5.4.

Suppose now that age and income are related. Perhaps older people tend to have higher incomes. The basic ideas just presented still apply, but the plot will be less dramatic. There will now be some clustering along the income dimension as well because of the association between the two predictors.

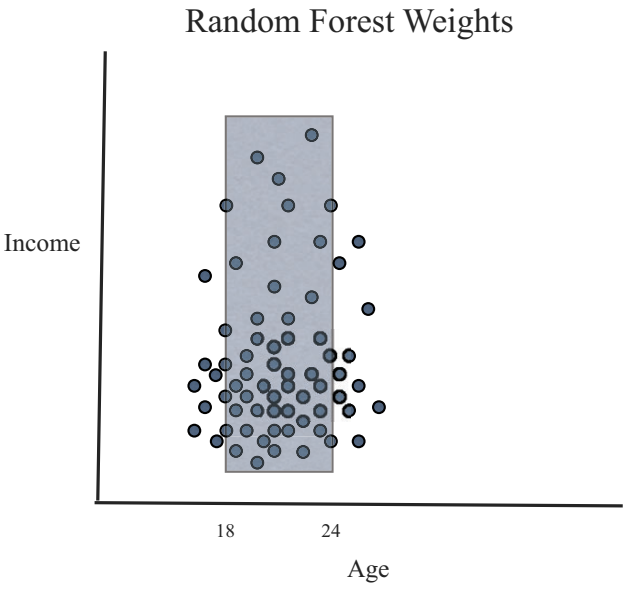


Fig. 5.5. Weights with two related predictors.

Random forests leads to somewhat different looking plots. As Figure 5.5 shows, the distinction between predictors that are included and predictors that are excluded is far less dramatic. One reason is that even weak predictors may be selected on occasion for use in partitioning the data if the predictors against which it is competing are even weaker. Another reason is that depending on

the competing predictors selected by chance for consideration, a splitpoint for strong predictors may sometimes be chosen that would otherwise be ignored.

Consequently, although there will be more nonzero weights clustering within regions defined by stronger predictors than elsewhere, there will also be instances where weights for weak predictors or weak regions within strong predictors will be greater than zero. Notice that in Figure 5.5 there is clustering in the vertical direction implying that now there is a cutpoint separating moderate to low incomes from high incomes. When income does not compete with age, a useful partitioning of the data tends to be found. Notice also that there are now voting points for years of age outside of 18 to 24.

It follows that information from income that would have been pushed aside under CART can be brought to bear in random forests. The fitting burden is shared, at least a bit, between age and income because nonzero weights are more widely distributed. In this context, income is an illustration of a highly specialized predictor that random forests is able to exploit. This is also another way to think about how random forests regularizes the fitted values.

Compared to CART more generally, random forests can include a wider variety of locations in the predictor space when conditional means or proportions are computed. There can be more voting points for a larger set of predictors and partitions within predictors. It is this ability to make better use of predictors that in part explains random forests' successes.

A second implication of the weighting formulation is that once one recognizes that random forests computes its fitted values as weighted averages, other uses can be made of the weights (Meinshausen, 2006). In particular, when one has a quantitative response variable (more on this later), the weights can be used to construct the cumulative distribution of the response values for each configuration of x-values. And with this in hand, one can compute for quantitative variables any conditional quantiles of interest such as the median or the 90th percentile. In effect, one can do random forests quantile regression.

Table 5.3 provides a simple illustration for a given target value $\mathbf{x}_0$. There are ten response values available for $\mathbf{x}_0$, listed in order, that have nonzero average weights across trees. The mean is computed by multiplying each response value by its average weight and adding the products. In this case, the mean is 83. However, quantiles are also available. The 10th quantile is 66. The 50th quantile (the median) is 82. The 90th quantile is 92. In short, if information such as that found in Table 5.3 is available, one is not limited to the mean of each $\mathbf{x}_0$.

There will sometimes be interest in the fitted values of conditional medians to "robustify" random forest results or to consider a central tendency measure unaffected by the tails of the distribution. Sometimes, there is subject-matter interest in learning about the conditional distribution of a very high or very low quantile, especially if those conditional distributions differ from one another and from the conditional distribution for the median.

For example, in today's world of school accountability based on standardized tests, perhaps students who score especially poorly on standardized tests

Average Weight	Response Value	Cumulative Weight
.10	66	.10
.11	71	.21
.12	74	.33
.08	78	.41
.09	82	.50
.10	85	.60
.13	87	.73
.07	90	.80
.11	98	.91
.09	99	1.0

Table 5.3. Weights and cumulative weights for a target value $\mathbf{x}_0$

.

respond better to smaller classroom sizes than students who excel on standardized tests. The performance distribution on standardized tests, conditioning on classroom size, differs for good versus poor performers. Random forests quantile regression can address such concerns. A real illustration is provided later in this chapter.

A final implication is that the weights provide another way to think about how a correct classifier would perform. In the population or stochastic process responsible for the data, the $f(X)$ leads to weights that in turn generate as a weighted average the systematic part of the response variable. A desirable classifier working with a random sample from this population or a random realization from the stochastic process might provide unbiased, or at least consistent, estimates of those weights. From these estimated weights, unbiased or consistent estimates of the fitted values could be constructed. By this standard, CART and random forest do not measure up.

5.5 Taking Costs into Account in Random Forests

Just as in CART, there is a need to consider the relative costs of false negatives and false positives. Otherwise, for each tree, one again has to live with the default values of equal costs and a prior distribution for the response variable that is the same as its empirical distribution in the data.

Perhaps the most conceptually direct method would be to allow for a cost matrix just as CART does. To date, this option is not available in random forest software, and there are suspicions that it might not work effectively if it were.

There are four approaches that have been seriously considered for the binary class case. They differ by whether costs are imposed on the data before each tree is built, as each tree itself is built, or at the end when classes are assigned.

1. Just as in CART, one can use a prior distribution to capture costs when each tree is built. This has the clear advantages of being based on the mechanics of CART and a straightforward way in the binary case to translate costs into an appropriate prior.
2. After all of the trees are built, one can differentially weight the classification votes over trees. For example, one vote for classification in the less common category might count the same as two votes for classification in the more common category. This has the advantage of being easily understood.
3. After all of the trees are built, one can abandon the majority vote rule and use thresholds that reflect the relative costs of false negatives and false positives. For instance, rather than classifying as "1" all observations for which the vote is larger than 50%, one might classify all observations as "1" when the vote is larger than 33%. This too is easy to understand.
4. When each bootstrap sample is drawn before a tree is built, one can oversample one class of cases relative to the other class of cases in much the same spirit as disproportional stratified sampling used for data collection (Thompson, 2002: Chapter 11). Before a tree is built, one oversamples the cases for which forecasting errors are relatively more costly. Conceptually, this is a lot like altering the prior distribution.

All four approaches share the problem that the actual ratio of false negatives to false positives in the confusion table may not sufficiently mirror the cost ratio. In practice, this means that whatever method is used to introduce relative costs, that method is simply considered a way to "tune" the results. With some trial and error, an appropriate ratio of false negatives to false positives can usually be achieved.

Although some very tentative experience suggests that in general all four methods can tune the results as needed, there may be some preference for tuning by the prior or by stratified bootstrap sampling. Both of these methods will affect the confusion table through the trees themselves. The structure of the trees themselves responds to the costs introduced. Changing the way votes are counted or the thresholds used only affects the classes assigned, and leaves the trees unchanged. The defaults of equal costs and the empirical prior remain in effect. It would seem that by allowing the trees themselves to respond to cost considerations, more responsive forecasts should be produced. Moreover, any output beyond a confusion table will reflect the role of costs. More is said about such output shortly.

There is one very important situation in which the stratified sampling approach is likely to be superior to the other three approaches. If the response variable is highly unbalanced (e.g., a 95–5 split), any given bootstrap sample may fail to include enough observations for the rare category. Then, a useful tree will be difficult to construct. As observed earlier, it will often be difficult under these circumstances for CART to move beyond the marginal distribution of the response. Oversampling rare cases when the bootstrap sample is

drawn will generally eliminate this problem. Using a prior that makes the rare observations less rare can also help, but that help applies in general and will not be sufficient if a given bootstrap sample makes the rare cases even more rare. We consider some examples in depth shortly. But a very brief illustration is provided now to prime the pump.

5.5.1 A Brief Illustration

	Forecast No Misconduct	Forecast Misconduct	Model Error
No Misconduct	3311	1357	.29
Misconduct	58	80	.42
Use Error	.02	.94	Overall Error = .29

Table 5.4. Confusion table for forecasts of serious prison misconduct.

Table 5.4 was constructed using data from the prison misconduct study described earlier. In this example, the response is incidents of very serious misconduct, not the garden-variety kind. As noted previously, such misconduct is relatively rare. Less than about 3% of the inmates had such reported incidents. So, just as for the domestic violence data shown in Table 5.1, it is difficult to do better than the marginal distribution under the usual CART defaults.

Suppose that the costs of forecasting errors for the rare cases were substantially higher than the costs of forecasting errors for the common cases. These relative costs can be effectively introduced by taking a stratified bootstrap sample, oversampling the rare cases. And by making the rare cases less rare, problems that might follow from the highly unbalanced response variable can sometimes be overcome.

For Table 5.4, the bootstrap samples for each of the response categories was set to equal 100. The "50–50" bootstrap distribution was selected by trial and error to produce a cost ratio of false negatives to false positives of about 20 to 1. This may be too high for real policy purposes, but it is still within the range considered reasonable by prison officials.

Why 100 cases each? Experience to date suggests that the sample size for the less common response category should equal about two-thirds of the number of cases in the class. If a larger fraction of the less common cases is sampled, the out-of-bag sample size may be too small.

With the number of bootstrap observations for the less common category determined to be 100, the 50–50 constraint leads to 100 cases being sampled for the more common response category. In practice, one determines the sample size for the less common outcome and then adjusts the sample size of the more common outcome as needed.

Table 5.4 can be interpreted just as any of the earlier confusion tables. For example, the overall proportion of cases incorrectly identified is .29. Random forests forecasts 42% of the incidents of misconduct incorrectly and 29% of the no misconduct cases incorrectly. Were prison officials to use these results for forecasting, a forecast of no serious misconduct would be wrong only 2 times out of 100, and a forecast of serious misconduct would be wrong 94 times out of 100. But for very serious inmate misconduct, having 1 true positive for about 20 false positives may be an acceptable tradeoff. The misconduct represented can include homicide, assault, sexual assault, and narcotics trafficking.

To summarize, random forests provides several ways to take the costs of false negatives and false positives into account. Ignoring these options does not mean that costs are not affecting the results. The default is equal to the costs and the use of the marginal distribution of the response variable as the empirical prior. However, there is to date no formal justification for preferring one costing method over the others, and the early hands-on experience is far from conclusive.

5.6 Determining the Importance of the Predictors

Just as for bagging, random forests leaves behind so many trees that collectively they are useless for interpretation. Yet, a central goal of statistical learning is to explore how inputs are related to outputs. Exactly how best to do this is currently unresolved, but there are several useful options available. We begin with a discussion of variable "importance."

5.6.1 Contributions to the Fit

One approach to predictor importance is to record the decrease in the fitting measure (e.g., Gini index) each time a given variable is used to define a split. The sum of these reductions for a given tree is a measure of importance for that variable when that tree is built. For random forests, one can average this measure of importance over the set of trees.

As with variance partitions, however, reductions in the fitting criterion ignore the forecasting skill of a model, which many statisticians treat as the gold standard. Fit measures are computed with the data used to build the classifier. They are not computed from test data.

Moreover, it can be difficult to translate contributions to a fit statistic into practical terms. Simply asserting that a percentage contribution to a fit statistic is a measure of importance is circular. Importance must be defined outside of the procedure used to measure it. And what is it about contributions to a measure of fit that makes a predictor more or less important? Even if an external definition is provided, is a predictor important if it can account for, say, 10% of the reduction in impurity?

In any case, one must be fully clear that contributions to the fit by themselves are silent on what would happen if in the real world a predictor is manipulated. Causality can only be established by how the data were generated, and causal interpretations depend on there being a real intervention altering one or more predictors (Berk, 2003).

5.6.2 Contributions to Forecasting Skill

Breiman (2001a) has suggested another form of randomization to assess the role of each predictor. This method is implemented in the R version of random forests. It is based on the reduction in predictive accuracy when a predictor is shuffled so that it cannot make a systematic contribution to a forecast. Reductions in predictive accuracy can be translated into practical terms. Would a reduction of, say, 10% in forecasting accuracy matter in real applications? In contrast to fit statistics, forecasting skill has direct implications for actual decisions.

Breiman's approach has much in common with the concept of Granger causality (Granger and Newbold, 1986: Section 7.3). Imagine two times series, Y_t and X_t. If the future conditional distribution of Y given current past values of Y is the same as the future conditional distribution of Y given current and past values of Y and X, X does not Granger cause Y. If the two future conditional distributions differ, X is a *prima facie* cause of Y.

These ideas generalize so that for the baseline conditional distribution, one can condition not just on current and past values of Y but on current and past values of other predictors (but not X). Then X Granger causes Y, conditional on the other predictors, if including X as a predictor changes the future conditional distribution of Y. In short, the idea of using forecasting skill as a way to characterize the performance of predictors has been advanced in both the statistical and econometrics literature.

Breiman's importance measure of forecasting skill differs perhaps most significantly from Granger's in that Breiman does not require time series data and randomly shuffles the values of predictors rather than dropping (or adding) predictors from the forecasting model. The latter has some important implications discussed shortly.

For Breiman's approach a categorical response variable is constructed using the following algorithm.

1. Construct a measure prediction error ν for each tree as usual by dropping the out-of-bag (OOB) data down the tree. Note that this is a real forecasting enterprise because data not used to build the tree are used to evaluate its predictive skill.
2. If there are p predictors, repeat Step 1 p times, but each time with the values of a given predictor randomly shuffled. The shuffling makes that predictor on the average unrelated to the response and all other predictors. For each shuffled predictor j, compute new measures of prediction error, ν_j.

3. For each of the p predictors, average over trees the difference between the prediction error with no shuffling and the prediction error with the jth predictor shuffled.

The average increase in the forecasting error when a given predictor j is shuffled represents the importance of that predictor for forecasting skill. That is,

$$\mathrm{I}_j = \sum_{k=i}^{K} \left[\frac{1}{K} (\nu_j - \nu) \right], \quad j = 1, \ldots, p, \tag{5.10}$$

where there are K trees, ν_j is the forecasting error with predictor j shuffled, and ν is the forecasting error with none of the predictors shuffled. It is sometimes possible for forecasting accuracy to improve slightly when a variable is shuffled because of the randomness introduced. A negative measure of forecasting importance follows. Negative forecasting importance can be treated as no decline in accuracy or simply can be ignored.

As written, Equation 5.10 is somewhat open-ended. The measures of forecasting error (ν and ν_j) are not defined. One could imagine using the number of forecasting errors, the percentage of cases forecasted incorrectly, the change in the margin, or some other measure. Currently, the preferred measure is the same as the one used when confusion tables are constructed: the proportion (or percentage) of cases misclassified. This has the advantage of allowing direct comparisons between predictor importance and either the row or column totals in the table. In addition, all of the other measures considered to date have been found less satisfactory for one reason or another. For example, some measures are misleadingly sensitive; small changes in the number of classification errors can lead to large changes in the importance measure.

One significant complication is that Equation 5.10 will almost always produce different importance measures for given predictors for different categories of the response. That is, there will be for any given predictor a measure of importance for each class forecasted, and the measures will not generally be the same. For example, if there are three response classes, there will be three measures of importance for each predictor that will generally not be the same. Moreover, this can lead to different rankings of predictors in the forecasting importance depending on which response category is being considered. Although this may seem odd, it follows directly from the fact that the number of observations in each response class and the margins for each class will typically differ. Consequently, a given increase in the number of misclassifications can have different impacts. A detailed illustration is be presented shortly.

Partly in response to such complications, one can standardize the declines in forecasting skill. The standard deviation of Equation 5.10 can be computed over the K trees. In effect, one has a bootstrap estimate over trees of the standard error associated with the increase in forecasting error, which can be used as a descriptive measure of stability. Larger values imply less stability.

Then, one can divide Equation 5.10 by this value. The result can be interpreted as a z-score so that importance measures are now all on the same scale. And with a bit of a stretch, confidence intervals can be computed and conventional hypothesis tests performed. It is a stretch because the sampling distribution of the predictor importance measure is usually not known. Perhaps more important, the descriptive gains from standardization are modest at best, as the illustrations that follow make clear.

One of the drawbacks of the shuffling approach to variable importance is that only one variable is shuffled at a time. There is no role for joint importance over several predictors. This can be an issue when predictors are not independent. There will be a contribution to forecasting skill that is uniquely linked to each predictor and a joint contributions shared between two or more predictors.

There is currently no option in the random forest software to shuffle more than one variable at a time. However, it is relatively easy to apply the prediction procedure in random forests using as input the original dataset with two or more of the predictors shuffled. Then, Equation 5.10 can be employed as before, where j would now be joined by other predictor subscripts. The main problem is that the number of potential joint contributions can be very large. In practice, some subset selection procedure is likely to be needed, perhaps based on substantive considerations.

It might seem that Granger's approach of examining forecasting skill with and without a given predictor included is effectively the same as Breiman's shuffling approach. And if so, one might consider, for instance, dropping sets of predictors to document their joint contribution. But actually, the two strategies are somewhat different. In Granger's approach, dropping or adding predictors to the model means that the model itself will be re-estimated each time. So, the comparisons Granger favors are the result of different predictors being included and different models. Under Breiman's approach, the model is not reconstructed. The shuffling is undertaken as an additional procedure with the model fixed.

In summary, for many scientists the ability to forecast accurately is the gold standard of a model's worth. If one cannot forecast well, it means that the model cannot usefully reproduce the empirical world. It follows that such a model has little value. And as now stressed a number of times, a model that fits the data well will not necessarily forecast well. The take-home message is simple: if forecasting skill is the gold standard (or even just a very important criterion by which to evaluate a model), then a predictor's contribution to that skill is surely one reasonable measure of that predictor's importance.

Some Examples

Figures 5.6 to 5.9 show how predictor importance can be represented as a reduction in forecasting accuracy. The data are, once again, from the prison study with the response variable very serious misconduct in prison. In each

figure, importance is on the horizontal axis and predictor names are on the vertical axis.

Figure 5.6 displays predictor importance for the "misconduct" response category. Importance is just the average decline over trees in forecasting accuracy. It is, therefore, "unscaled" or "unstandardized." Taking all the variables into account, Table 5.4 indicates that serious misconduct is correctly forecasted about 58% of the time. That accuracy drops to about 51% (i.e., about 7%) if the variable "Term" is shuffled. It drops to about 52% (i.e., about 6%) if the variable "Gang" is shuffled. None of the other predictors meaningfully affect forecasting accuracy. The two most important predictors are "Term" and "Gang." Recall that "Term" refers to sentence length and "Gang" refers to street or prison gang activity.

It is useful to keep in mind that the importance represented is the decline in forecasting accuracy uniquely attributable to each predictor. Predictive skill shared between predictors is not included. Thus, for example, the sum of the declines in forecasting accuracy for all predictors is usually less, often far less, than overall forecasting accuracy.

Figure 5.7 shows the unscaled importance of the predictor for the no misconduct response category. Now the base is different because the absence of misconduct is forecast with about 71% accuracy. Moreover, there is no particular reason why the predictors that play a major role in forecasting misconduct should play a major role in forecasting no misconduct. Because this may seem to be counterintuitive, some discussion is warranted.

Recall how classification is accomplished in random forests. The class is assigned by majority vote. Two features of those votes are especially relevant here: the margin and the number of actual class members.

Consider a simple example. Suppose a given inmate receives a vote of 25 to 24 to be assigned to the misconduct class category. Suppose that in fact that inmate has a reported incident of serious misconduct; the forecast is correct. Now a predictor is shuffled. The vote might be very different. But suppose it is now 24 to 25. Only one vote has changed. Yet, the inmate is now placed in the no misconduct class. This increases the forecasting error by one inmate.

Is that one inmate increase enough to matter? It depends on how many inmates were correctly predicted to have a serious misconduct incident and how many were incorrectly predicted to have a serious misconduct incident. Suppose 10 inmates actually had an incident of serious misconduct, with 7 correctly predicted to have an incident of serious misconduct and 3 incorrectly predicted to have an incident of serious misconduct. Changing just one inmate from a true positive to a false negative reduces forecasting accuracy from 70% to 60%. If there had been 100 inmates who actually had incidents of serious misconduct with 70 true positives and 30 true negatives, changing that one inmate would reduce forecasting accuracy from 70% to 69%.

In summary, if the margins tend to be small, dropping a predictor can easily change the class assigned for a significant number of cases. Then, if the number of true class members is small as well, the change of even a few

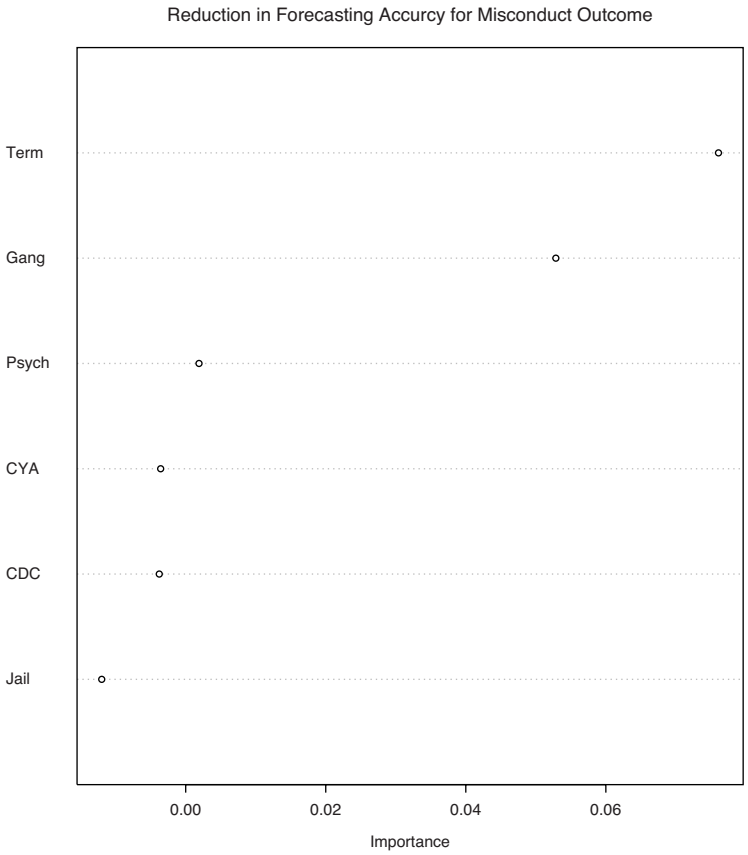


Fig. 5.6. Unscaled forecasting importance for misconduct

cases from one assigned class to another can dramatically affect the proportion of cases whose class membership was accurately forecasted. Because both the margins and the class sizes can differ depending on which response category is considered, forecasting importance of the predictors can differ as well. Thus, Figure 5.6 looks somewhat different from Figure 5.7. Comparisons such as these seem to argue for some kind of standardization, perhaps through the z-scores mentioned earlier.

In Figure 5.7, none of the predictors affect forecasting skill very much. Yet, "Term" is still the most important predictor. A previous sentence in a state juvenile facility (CYA) now comes in second. Gang activity drops to third place.

The differences between Figures 5.6 and 5.7 illustrate the kinds of complications just noted. As an empirical matter, the number of no misconduct

observations is large. Other things being equal, many cases would have to change from true negative to false positive for forecasting accuracy to decline a substantial amount. But it is more complicated than that because margins for each case will likely differ from the margins when misconduct is the response category.

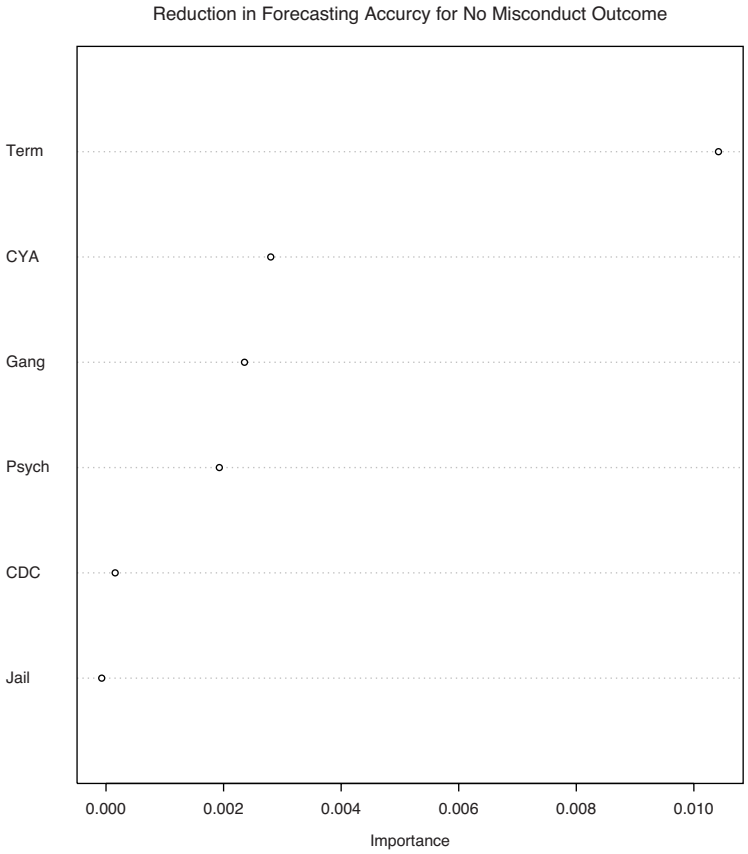


Fig. 5.7. Unscaled forecasting importance for no misconduct.

Figures 5.8 and 5.9 replay the same analysis in standardized scores. The horizontal axis is now in z-scores. Because forecasting importance is scaled to be in units of the same size, the two figures can be more easily compared. However, we already know that forecasting skill declines very little for the no misconduct class when predictors are shuffled. Those are the facts. It is not clear, therefore, how standardizing helps if the goal is to characterize predictors by their forecasting skill. Two predictors may be deemed strong

and equally important by the z-score metric when in fact one substantially affects forecasting skill and the other does not.

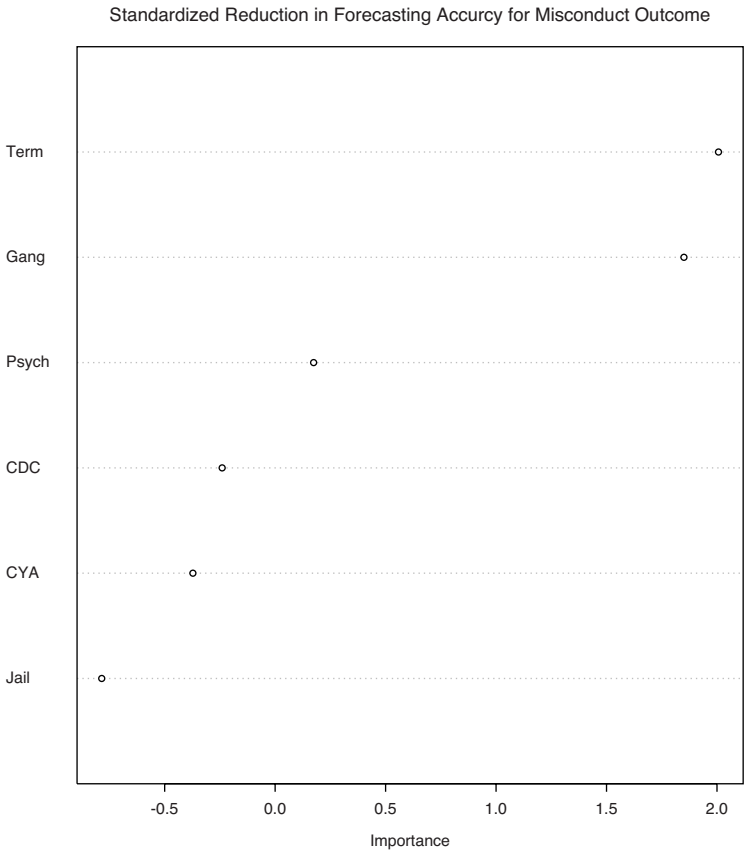


Fig. 5.8. Scaled forecasting importance for misconduct.

Let's return to the matter of stability over trees and take "Term" for the misconduct outcome as an example. The standard deviation over trees of the measure of forecasting importance is about .03. Thus for term length, one can say that although mean importance over trees is about .07, importance will typically vary from about .04 to about .10. If, however, the standard deviation over trees were .10, importance would typically vary from the lower bound of 0.0 to about .17. Clearly, one has a much worse fix on how important term length really is for the second case.

Insofar as the distribution of raw importance scores over trees is approximately normal, formal hypothesis tests and confidence intervals can follow.

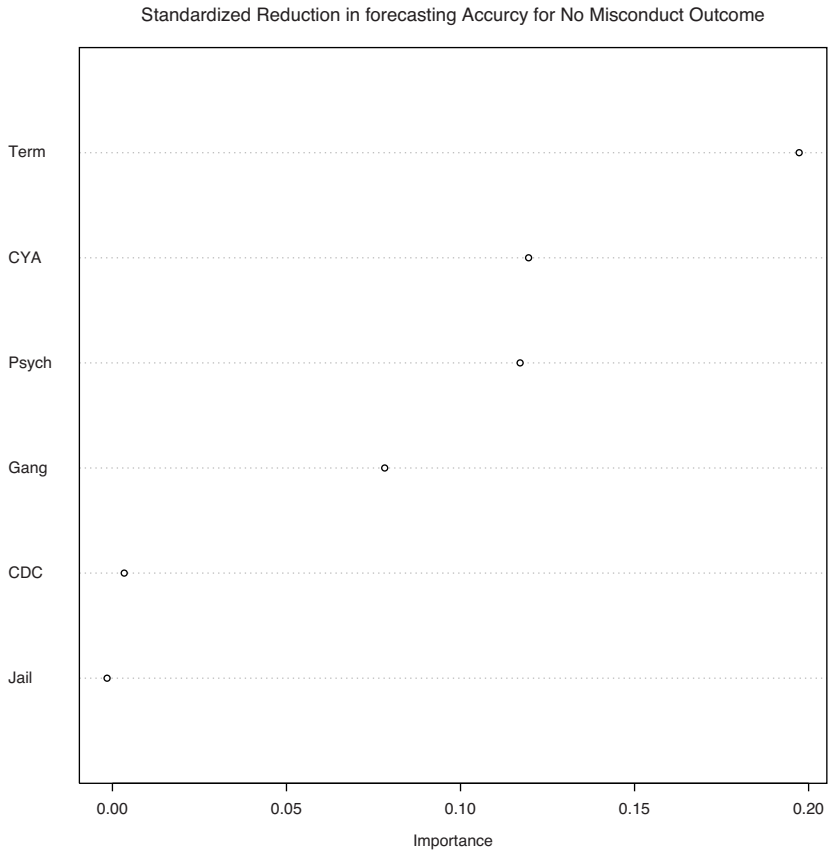


Fig. 5.9. Scaled forecasting importance for no misconduct.

For example, if term length had a z-score of over 2.0, one might justifiably take that as a rejection of the null hypothesis at the .05 level that the importance of term length is 0.0. If one is not prepared to bet that the distribution of importance is approximately normal, one can in principle resort to resampling tests directly over the set of trees. This capability is currently not available in R's random forests software, but with very modest changes in the code it could be.

One must be clear that the uncertainty being assessed comes from the bootstrap sampling and predictor sampling within random forests itself. Nothing whatsoever is being said about the stability of importance measures over sets of training data selected by probability sampling. Random forests outputs a single measure of importance for each predictor as an average over trees. If one were interested in the overall uncertainty in this single measure for each

predictor, one would at least need to address in addition the implications of random samples of training data. A possible approach would be to embed the random forest procedure in bootstrap samples of the existing training data.

In summary, the decision to standardize or not standardize raises the old saw of substantive versus statistical significance. The unstandardized measure of forecasting importance addresses substantive significance. The standardized measure of forecasting importance addresses (ideally) statistical significance. Both can be important, but they are different.

The implications of these illustrations generalize to response variables with more than two categories. There can be scaled or unscaled plots for each response category. This underscores a point made earlier. Moving from two response categories to more than two does not change anything fundamental. But analysis complexity will increase dramatically with each additional response category.

5.7 Response Functions

Predictor importance is only part of the story. In addition to knowing the importance of each predictor, it can be very useful to have a description of how each predictor is related to the response. The set of response functions needs to be described.

One useful solution, based on an earlier suggestion by Breiman and his colleagues (1984) is "partial dependence plots" (Friedman, 2001; Hastie et al., 2001: Section 10.13.2). For tree-based approaches such as CART, one proceeds as follows.

1. Grow a forest.
2. Suppose x_1 is the initial predictor of interest, and it has v distinct values in the training data. Construct v data sets as follows.
   a) For each of the v values of x_1, make up a new dataset where x_1 only takes on that value, leaving all other variables untouched.
   b) For each of the v datasets, predict the response using random forests. There will be a single value averaged over all observations.
   c) Average each of these predictions over the trees.
   d) Plot the average prediction for each value for each of the v datasets against the v values of x_1
3. Go back to Step 2 and repeat for each predictor.

Partial dependence plots show the relationship between a given predictor and the response averaged within the joint values of the other predictors as they are represented in a tree structure. In this way, the other predictors are being "held constant" by matching. Consequently, no assumptions are made about how the predictors are related to one another or to the response variable. One price for this approach is that interaction effects are not represented

unless the appropriate interaction variables are constructed in advance and included among the set of predictors.

Unlike the plots of fitted values constructed by smoothers (e.g., from the generalized additive model), partial dependence plots impose no smoothness constraints, and the underlying tree structure tends to produce somewhat bumpy results. In practice, one usually imposes an "eyeball" smoother when the plot is interpreted. Alternatively, it is often possible to overlay a smoother if the software stores the requisite output.

Partial plots can be constructed for quantitative responses and for responses with more than two categories. For quantitative response variables, the units represented on the vertical axis usually are the natural units of the response, whatever they happen to be. For categorical response variables, the units of the response represented on the vertical axis are unconventional and easily misunderstood. Because partial dependence plots can be so very useful for applied work, the metric used needs to be examined in some detail. We begin with the binomial case.

It is common to see a logistic regression equation written as

$$\log\left(\frac{p}{1-p}\right) = \mathbf{X}\boldsymbol{\beta}, \tag{5.11}$$

where p is the probability of success. The term on the left-hand side is the log of the odds of a success, often called the "logit." The change in the response for a unit change in a predictor is in "logits."

For the multinomial case, the most common approach to logistic regression builds up from the familiar binary formulation. If there are K response categories, there are $K-1$ equations, each of the same general form as Equation 5.11. However, one equation of the K possible equations is redundant because the response categories are exhaustive and mutually exclusive. Thus, if an observation does not fall in categories $1, \ldots, K-1$, it must fall in the Kth category. This implies that a single category can be chosen as the reference category, just as in the binomial case (i.e., there are two possible outcomes and one equation). Then, for each of the $K-1$ equations, the logit is the log of the odds for a given category compared to the reference category.

Suppose there are four response categories, and the fourth is chosen as the reference category. There would then be three equations with three different responses, one for $\log(p_1/p_4)$, one for $\log(p_2/p_4)$, and one for $\log(p_3/p_4)$. The predictors would be the same for each equation, but each equation would have its own set of regression coefficients differing in values across equations.

One might think that partial dependence plots would follow a similar convention. But they don't. The choice of the reference category determines which logits will be used, and the logits used affect the regression coefficients that result. Although the overall fit is the same no matter what the reference category, and although one can compute from the set of estimated regression coefficients what the regression coefficients would be were another reference

category used, the regression coefficients reported are still different when different reference categories are used.

There is no statistical justification for choosing one reference category or another. The choice is usually made on subject matter grounds to make interpretations easier, and the choice can easily vary from data analyst to data analyst. So, the need for a reference category can complicate interpretations of the results and means that a user of the results has to undertake considerable additional work if regression coefficients using another reference category are desired.

In response to these complications, partial dependence plots are based on a somewhat different approach. There are K, rather than $K - 1$, response functions, one for each response variable class. For the logistic model, these take the form of

$$p_k(X) = \frac{e^{f_k(X)}}{\sum_{k=1}^{K} e^{f_k(X)}}. \tag{5.12}$$

There is still a redundancy problem to solve. The solution employed by partial dependence plots is to constrain $\sum_{k=1}^{K} f_k(X) = 0$. This leads to the multinomial deviance loss function and the use of a rather different kind of baseline.

Instead of using a given category as the reference, the unweighted mean of the proportions in the K categories is used as the reference. In much the same spirit as analysis of variance, the response variable units are then in deviations from a mean. More specifically, we let

$$f_k(X) = \log[\mathrm{p}_k(X)] - \frac{1}{K}\sum_{k=1}^{K} \log[\mathrm{p}_k(X)]. \tag{5.13}$$

Thus, the response is the disparity between the logged proportion for category k and the average of the logged proportions for all K categories. The units are essentially logits but with the mean over the K classes as the reference. Consequently, each response category can have its own equation and, therefore, its own partial dependence plot. This approach is applied even when there are only two response categories, and the conventional logit formulation might not present interpretive problems.

To help fix these ideas, consider an example of a single data point for a binary outcome. Here, a single data point is defined by a single value of a given predictor. For that data point, the partial dependence algorithm classifies all of the observations as described earlier. For each of the response variable categories, there is a proportion of observations assigned. Then Equation 5.13 is applied.

To illustrate, consider once again the prison data. Suppose term length in years was the predictor whose relationship with the binary misconduct response variable was of interest. And suppose for a term length of say, 1 year, the proportion of inmates engaging in an incident of misconduct was

.20 (computed using the partial dependence algorithm). If the proportion of success is .20, the value plotted is $\log(.2) - [\log(.2) + \log(.8)]/2 = -0.693$ (using natural logarithms). This is the value that would be plotted on the vertical axis for the term length value on the horizontal axis of 1.0.

The same approach could be used for the proportion of inmates with no misconduct. The proportion of failures is necessarily .80, so that value plotted for failures is $\log(.8) - [\log(.2) + \log(.8)]/2 = 0.693$, also associated with a term length of 1.0. In the binary case, essentially the same information is obtained no matter which response class is examined. The partial dependence function is centered on 0.0 because of the constraint that the sum of the response functions is zero; these are deviation scores in logit units. So, the distance above zero for the response function of one class is the same as the distance below zero for the response function of the other class. One response function is the mirror image of the other. Thus, one partial dependence plot is the mirror image of the other partial dependence plot, and only one of the two is required for interpretation.

In the binary case, it is easy to get back into more familiar logit units. The values produced by Equation 5.13 are half the usual log of the odds. And from there, one can easily get back to the relevant probabilities. For example, multiplying –.693 by 2 and exponentiating yields an odds of .25. Then, solving for the numerator probability results in a value of .20. We are back where we started.

Equation 5.13 would be applied for each value of term length. Thus, for 1.5 years, the proportion of inmates engaging in misconduct might be .25. Then the value plotted on the horizontal axis would be 1.5, and the value on the vertical axis would be $\log(.25) - [\log(.25) + \log(.75)]/2 = -.549$.

The value of –.549 is at a region where the response function is increasing. With .5 units (i.e., six months) increase in term length, the value of the response increases .144 (i.e., from –.693 to –.549). And all other values produced for different term lengths can be interpreted in a similar way. Consequently, one can get a sense of how the response variable changes with changes in a given predictor, all other predictors held constant.

For more than two response variable categories, each of the response categories can be usefully plotted. Suppose, for example, there are three response categories: no misconduct, minor misconduct, and serious misconduct. And suppose the respective proportions when term length is 1.0 years are .70, .20, and .10. The three values computed for the three response categories are respectively 1.066, –.187, and –.880. Note that as before the sum of the values is again 0.0. Each of these values would likely change as the value of the predictor of interest changed. For each, the sum would still be zero. But the changing values could not be represented as a set of mirror images. Three partial dependence plots would follow.

To summarize, the vertical axis in partial dependence plots is the response function as defined by Equation 5.13. It is derived from Equation 5.12 with the constraint that the sum of response functions $f_k(X)$ is equal to zero. Thus,

the sum of the values from Equation 5.13 for the categories of the response variable is zero as well.

5.7.1 An Example

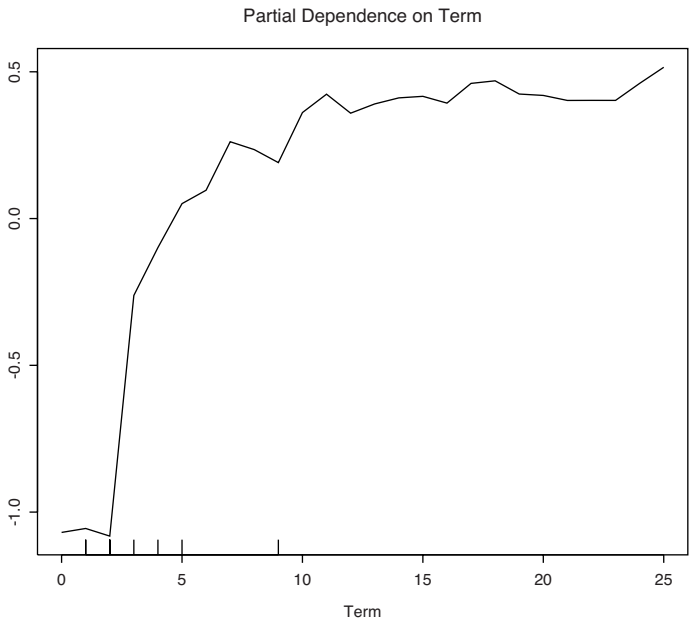


Fig. 5.10. Response function for misconduct and term.

Figures 5.10 and 5.11 show partial dependence plots constructed from the prison data. Figure 5.10 is for the misconduct response category and Figure 5.11 is for the no misconduct response category. For both, term length in years is the predictor. In both cases the vertical axis is in the logit units just discussed, and the horizontal axis is in years. From the discussion just completed, one plot should be the mirror image of the other. For the binary case, one partial plot is sufficient.

Both plots indicate that the odds of serious misconduct generally increase with term length. The increase is relatively rapid for terms from two to ten years. There seems to be no relationship between term length and misconduct for terms less than two years, and the rate of increase is relatively slow for terms greater than about ten years.

In order to get a practical sense of whether misconduct varies a lot with term length, it can be useful to transform the logits back to their underlying

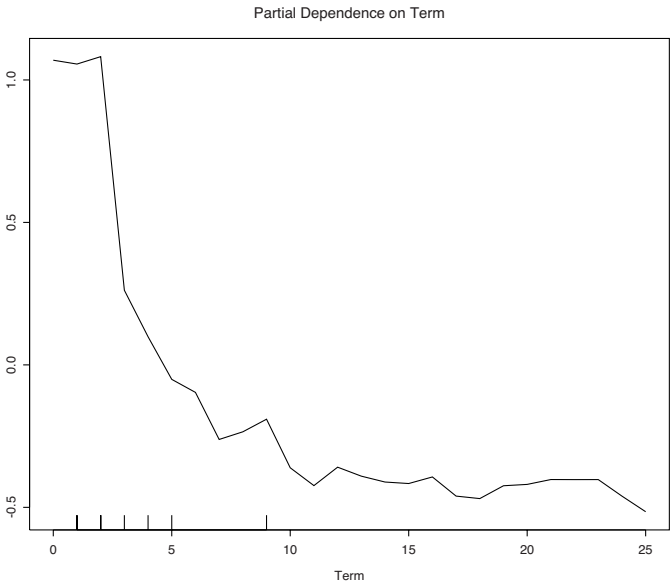


Fig. 5.11. Response function for no misconduct and term.

probabilities. For example, a logit value of –1 is equal to a probability of about .12. A logit value of .5 is equal to a probability of about .73. Because –1 and .5 represent the approximate minimum and maximum values of the response in logit units, the probability of serious misconduct increases from about .12 to about .72 as term length increases from two years to about ten years. This is a large effect in practical terms. Note that this increase is in deviation scores and that, therefore, it is the difference that matters, not the values themselves.

Finally, it is important to keep in mind that the response functions displayed in partial dependence plots reflect the relationship between a given predictor and the response, conditioning on all other predictors. All other predictors are being "held constant" in a manner that is equivalent to matching. That is why the plots are called partial dependence plots. Consequently, Figures 5.10 and 5.11 show how term length is related to serious misconduct, with gang activity, age at the time of admission to prison, and all other predictors included in the analysis held constant.

Figures 5.12 – 5.14 show the response functions for three classes of inmate misconduct and sentence length. The three classes, as before, are no misconduct, minor misconduct, and serious misconduct. Three partial dependence plots are necessary because although the values of the three response functions sum to zero, no plot is the mirror image of another. The baseline is, in the sense discussed above, the typical proportion of inmates over the three response classes.

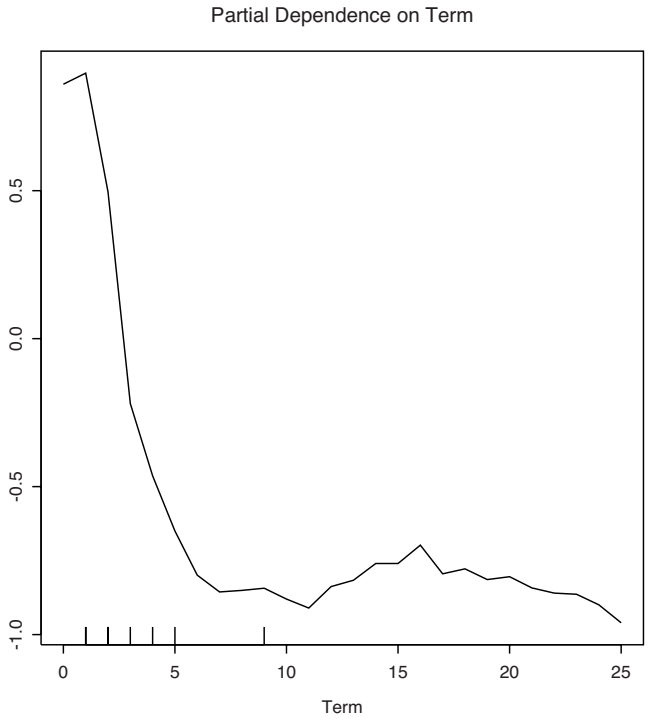


Fig. 5.12. Response function for no misconduct and term for a three class response.

Figure 5.12 shows the partial dependence of no misconduct on sentence length. The proportion of inmates with no reported incidents of misconduct decreases rapidly for sentences up to five years. Then the response function becomes flat.

Figure 5.13 shows the partial dependence of minor misconduct on sentence length. The proportion of inmates with reported incidents of minor misconduct increases rapidly for sentences up to about five years, levels off, and then declines for sentences of more than ten years. The downward trend ends with sentences of about 18 years. After that, it may even increase a bit.

Figure 5.14 shows the partial dependence of serious misconduct on sentence length. The proportion of inmates with reported incidents of serious misconduct increases rapidly up to a sentence of about five years and then increases much less rapidly thereafter.

Viewed as a group, the three figures complement one another and show associations that are large in practical terms. With increasing sentence length, the proportion of inmates who engage in no misconduct drops off rapidly until a sentence of about five years. Over those same shorter sentences, the propor-

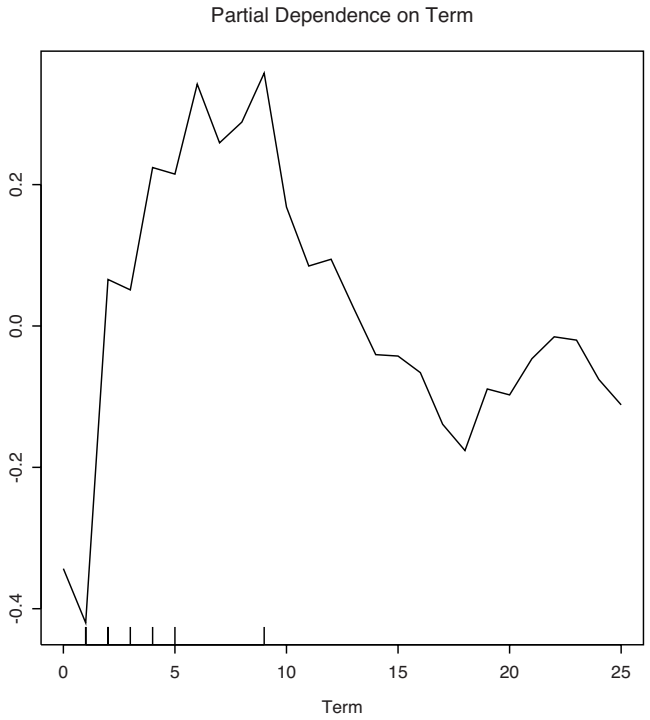


Fig. 5.13. Response function for minor misconduct and term for a three class response.

tion of inmates who engage in minor misconduct increases commensurately. And over the same short sentences, the proportion of inmates who engage in serious misconduct also increases commensurately.

But for sentences of around ten years or more, the proportion of inmates who engage in minor misconduct falls off, and the proportion of inmates who engage in serious misconduct continues to increase. Longer sentences are associated with increases in the likelihood of both minor and serious misconduct, but for very long sentences, the association is only with serious misconduct. One interpretation is that with very long sentences, inmates who might commit acts of minor misconduct now commit acts of serious misconduct.

5.8 The Proximity Matrix

It can be useful to determine the degree to which individual observations tend to be classified alike. In random forests, this information is contained in the "proximity matrix." The proximity matrix is constructed as follows.

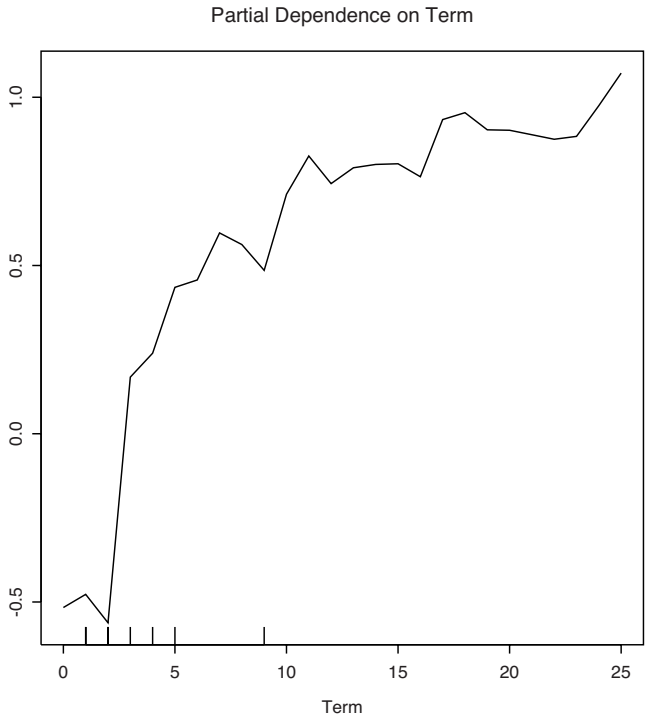


Fig. 5.14. Response function for serious misconduct and term for a three class response.

1. Grow a tree as usual.
2. Drop all the training data (in-bag and out-of-bag) down the tree.
3. For all possible pairs of cases, if a pair lands in the same terminal node, increase their proximity by one.
4. Repeat Steps 1–4 until the designated number of trees has been grown.
5. Normalize by dividing by the number of trees.

The result is an $n \times n$ matrix with each cell showing the proportion of trees for which each pair of observations winds up in the same terminal node. The higher that proportion, the more alike those observations are in how CART places them, and the more "proximate" they are.

However, it can be very demanding to store an $n \times n$ matrix and even more demanding to operate on it. The storage problem can be partly addressed by only storing the upper or lower triangle, but working with tens of thousands of proximity values (or more) remains a serious difficulty. When feasible, it can help to work with a random sample of the training data instead of the full set of observations. Another work around is to store only the largest proximity

values. There are currently efforts under way to find more elegant and formally defensible solutions.

The proximity matrix is also usually far too large to be directly examined in a meaningful manner. But helpful information can be extracted from the proximity matrix in several different ways. We consider three applications that are sufficiently well developed to be of some practical use.

5.8.1 Clustering by Proximity Values

The proximity matrix can be treated as a similarity matrix and subjected to multidimensional scaling. Plots of the observations in the first two dimensions extracted can help show whether the data tend to cluster in the space defined by predictors and whether those clusters tend to differ by the class to which the observations in each cluster belong. This information can give an initial sense of whether a classification exercise is likely to be successful.

The existing methods for displaying the scaling results are currently in some flux. A lot depends on first solving the computational problems associated with the proximity matrix. In addition, the current graphic display will no doubt be refined as hands-on experiences accumulates.

5.8.2 Using Proximity Values to Impute Missing Data

There are two ways in which random forests can impute missing data. The first and quick method relies on a measure of location. If a predictor is quantitative, the median of the available values is used. If the predictor is categorical, the modal category from the available data is used. If there are small amounts of missing data, this method may be satisfactory, especially given the computational demands of the second method.

The second method capitalizes on the proximity matrix in the following manner.

1. The "quick and dirty" method of imputation is first applied to the training data, a random forest is constructed, and the proximity values computed.
2. If the missing value is from a quantitative variable, the weighted average of the values of the nonmissing cases for that variable is used. The proximity values between that missing observation and all of the nonmissing observations are used as the weights. So, cases that are more like the cases with the missing data are given greater weight. All missing values for that variable are computed in the same fashion.
3. If the missing value is from a categorical value, the inputed value is the most common nonmissing value for the variable, with the frequencies weighted by proximity. Again, cases more like the case with the missing data are given greater weight. All missing values for that variable are are computed in the same fashion.

The step using proximity values is then iterated several times. Experience to date suggests that four to six iterations is sufficient. But the use of imputed values tends to make the OOB measures of fit too optimistic. There is really less information being brought to bear in the analysis than the random forest algorithm knows about. The computational demands are also quite daunting and may be impractical for many datasets until more efficient ways to handle the proximities are found.

5.8.3 Using Proximities to Detect Outliers

The proximity matrix can be used to spot outliers in the space defined by the predictors. The basic idea is that outliers are observations whose proximities to all other observations in the data are small. Currently, the procedures in R's version of random forests to detect outliers are not implemented for quantitative response variables. For categorical response variables, outliers are defined within categories of the response variable. For each observed outcome class, each observation is given a value for its "outlyingness" computed as follows.

1. For a given observation, compute the sum of the squares of the proximities with all of the other observations in the same outcome class. Then take the inverse. A large value will indicate that on the average the proximities are small for that observation. Do the same for all other observations in that class. One can think of these values as unstandardized.
2. Compute the median and mean absolute deviation around the median of the unstandardized values.
3. Subtract the median from each of the unstandardized values and divide by the mean absolute deviation. In this fashion, the unstandardized values are standardized.
4. Values less than zero are set to 0.0.

These steps are then repeated for each category of the response variable. Observations with values larger than about ten can be considered outliers.

Especially if the number of observations overall is modest (e.g., less than 100), it can be instructive to drop the outliers from the training data, repeat the random forest analysis, and see if the results change by a meaningful amount. If the number of observations is large, it is very unlikely that a few outliers will make an important difference in the results.

When the data analyst considers dropping one or more outlying cases, a useful diagnostic tool can be a cross-tabulation of the classes assigned for the set of observations that the two random forest analyses have in common. If the observations are, by and large, classified in the same way in both analyses, the outliers do not make an important difference to the classification process.

5.9 Quantitative Response Variables

There is not very much new that needs to be said about quantitative response variables once one appreciates that random forests handles quantitative response variables much as CART does. Recall that for CART, impurity when trees are constructed is defined as the within-node error sum of squares. A new partition of the data is determined by the split that would most reduce the within-node error sum of squares. Predicted values are determined by the mean of the response variable in each of the terminal nodes. For each observation, the mean of its terminal node is the value assigned.

For regression trees, therefore, there are no classification errors, only residuals. Concerns about false negatives and positives and their costs are no longer relevant. There are no confusion tables and no measures of importance based on predictor errors.

To turn a regression tree into a fully operational random forest, there are several operations required.

1. Just as in the classification case, each tree is constructed from a random sample (with replacement) of the training data.
2. Just as in the classification case, at each potential partitioning of the data, a random sample (without replacement) of predictors is used.
3. Just as in the classification case, the out-of-bag data are used to construct predicted values. After a tree is built, the OOB observations are dropped down the tree. From these observations, a mean is computed for each terminal node. These means serve as the predicted values for the observations in their respective terminal nodes. The predicted values are not (and cannot be) membership in a particular class.
4. Then, random forest averages in much the way as it does for classification problems. For a given observation, the average of the tree-by-tree predicted values is computed using only the predicted values from trees in which that observation was not used to build the tree. This is the predicted value that random forest returns. Then the deviations between these over-tree predicted values and the observed values are used to construct the mean square error reported for the collection of trees that constitutes a random forest. The value of the mean square error can be used to compute a pseudo R^2 as $(1 - \text{MSE})/\text{Var}(Y)$.
5. Construction of partial dependence plots is done in the same manner as for classification trees, but now the fitted response is the set of conditional means for different predictor values, not a set of transformed fitted proportions.
6. Importance is computed using the shuffling approach as before. And as before there is a "resubstitution" measure and a forecasting measure. For the resubstitution measure, consider a single tree. Each time a given variable is used to define a partitioning of the data, the reduction in the within-node error sum of squares is recorded. When the tree is complete,

the reductions are summed. The result is the error sum of squares that can be attributed to each predictor. These totals, one for each predictor, are then averaged over trees.

7. The forecasting measure uses the OOB observations. For each tree, the OOB observations are used to compute the predicted values and the within-node mean square error around them. Then a given predictor is shuffled, and the OOB predicted values and mean square error computed again. An increase in this mean square error is a decrease in accuracy. These decreases are averaged over trees to get an average decrease in accuracy for that predictor. The standard deviation of these decreases over trees can be used to standardize the average decrease, if that is desirable.

Despite the tight connection between regression trees and random forests, there are a few features found in some implementations of regression trees that have yet to be introduced into random forests. Perhaps most important, random forests is currently limited to the normal regression model. There are, for instance, no accommodations for count data, where some form of Poisson regression might be appropriate. Likewise, there are no accommodations for bounded response variable distributions such as might be found for survival data. However, such generalizations are likely to come soon.

For example, there is a new procedure in R called quantregForest() that computes for each terminal node quantiles of the user's choosing. Instead of storing only the mean of each terminal node as trees are grown, the entire distribution is stored. Recall the earlier discussion surrounding Table 5.3. Once the user decides which quantiles are of interest, they can be easily computed.

If one is worried about the impact of within-node outliers on the conditional mean, the conditional median can be used instead. If for substantive reasons there is interest in the first or third quartile, those can be used. Perhaps most interestingly, the quantile option provides an interesting way to take the costs of forecasting errors into account. For example, if the 75th quantile is chosen, the consequences of underestimates are three times more costly than the consequences of overestimates. However, such calculations only affect what is done with the information contained in the terminal nodes across trees. This approach does not require that the trees themselves be grown again with a linear loss function, let alone a loss function with asymmetric costs. In other words, the trees grown under quadratic loss are not changed. As a result, the quantile adjustments are not complete. An example is discussed later in this chapter.

5.10 Tuning Parameters

Despite the complexity of the random forest algorithm and the large number of potential tuning parameters, most of the usual defaults work well in practice. The tuning parameters most likely to require some manipulation are the following.

1. *Node Size*—Unlike in CART, the number of observations in the terminal nodes of each tree can be very small. The goal is to grow trees with as little bias as possible. The high variance that would result can be tolerated because of the averaging over a large number of trees. In the R implementation of random forests, the default sample sizes for the terminal nodes are one for classification and five for regression. These seem to work well. But one must also keep in mind the concerns raised earlier when there are a large number of predictors weakly related to the response and at least moderately related to each other. If such predictors are not dropped, it is usually wise to grow smaller trees. If one is interested in estimating a quantile, such as in quantile random forests, then terminal node sizes about twice as large will often be necessary. If there are only five observations in a terminal node, for instance, it will be difficult to get a good read on, say, the 90th percentile.
2. *Number of Trees*—The number of trees used to constitute a forest needs to be at least several hundred and probably no more that several thousand. In practice, 500 trees is often a good compromise. It sometimes makes sense to do most of the intitial development (see below) with about 500 trees and then confirm the results with a run using about 3000 trees.
3. *Number of Predictors Sampled*—The number of predictors sampled at each split would seem to be a key tuning parameter that should affect how well random forests performs. Although it may be somewhat surprising, very few predictors need to be randomly sampled at each split, and with sensible bounds on the number sampled, it does not seem to matter much for the OOB error estimates. With a large number of trees, each predictor will have an ample opportunity to contribute, even if very few are drawn for each split. For example, if the average tree in a random forest has ten terminal splits, and if there are 500 trees in the random forest, there will be 5000 chances for predictors to weigh in. Sampling two or three each time should then be adequate.

   But a lot depends on the number of predictors and whether all have good potential or whether some do and some don't. In the manual for the FORTRAN version of random forests, Breiman recommends starting with the number of predictors sampled equal to the square root of the number of predictors available. Then, trying a few more or a few less as well can be instructive.

   In the R implementation of random forests, one can search for the best number of predictors to sample using the OOB error statistic as a criterion. This is an excellent tool in principle. In practice, large differences in performance are rarely found. Also, one must be careful not to overtune and introduce the overfitting that random forests is designed to prevent.

The feature of random forests that will usually make the biggest difference in the results is how the costs of false negatives and false positives are handled.

These costs have already been extensively discussed and are not reconsidered now. At the same time, costs are not really a tuning parameter, but a key aspect of how the data are to be analyzed.

5.11 An Illustration Using a Binary Response Variable

Industrialized fishing is dramatically reducing the stock of predatory fish throughout the oceans of the world. Large-scale commercial fishing affects not just the target species but other species that become the "bycatch." The impact on dolphin populations of commercial fishing for tuna is perhaps the most visible illustration and has been the subject of a National Research Council committee report (Committee on Reducing Porpoise Mortality from Tuna Fishing, 1992). Over the past decade, international cooperation to reduce dolphin mortality has led to efforts to monitor tuna fishing practices and penalize offenders. The political and technical issues are very complex.

Dolphin are put at risk in tuna fishing because they are often used to locate large schools of tuna. For reasons that are not fully understood, dolphin are often found swimming above schools of tuna and because the dolphin typically swim close to the surface, they can be seen by fishermen some distance away. Then, when large nets are deployed to catch the tuna, the dolphin can be caught as well. Over the past two decades fishing technology and procedures have been changed so that dolphin mortality can be dramatically reduced, but the mortality is far from zero.

The Inter-American Tropical Tuna Commission (ITTC), which oversees the international purse-seine fishery for tuna in the eastern Pacific Ocean, has provided data on dolphin mortality. The dataset includes over 100,000 observations. An observation is a "set," defined as placing a large net into the water to encircle a school of tuna. There are over 200 predictors. Here, the intent is to determine the circumstances under which dolphin mortality is likely to be high.

For example, a major cause of dolphin deaths apparently is whether the net "collapses" as it is drawn to the boat. A net collapse is a relatively rare event, but one to be actively avoided. For similar reasons, it would be helpful to learn which other predictors are associated with dolphin mortality so that preventive actions might be taken by fishermen.

Sanctions can be applied to fishermen if any dolphin are killed. There is zero tolerance for any dolphin mortality. This suggests treating the response as a binary outcome: whether any dolphin are killed or not. And this is how we proceed here.

For this illustration, we use the predictors listed below. In discussions with the ITCC, these predictors were singled out as by far the most promising. We could have included well over 100 predictors, but some of the graphics would have been unnecessarily cluttered and difficult to explain.

1. capskill: Captain skill coded "0 for less than 30 dolphin sets/year and "1" for 30 or more.
2. biomass: The number of animals encircled in the net.
3. cwtunay: Catch weight of yellowfin tuna in metric tons.
4. cwtunao: Catch weight of other tuna in metric tons.
5. encircle: Duration of the encirclement phase of the set in decimal hours.
6. netretrieval: Duration of the prebackdown net retrieval in decimal hours.
7. backdown: Duration of the backdown procedure in decimal hours.
8. netcanopy: Coded "1" if a net canopy present and "0" if a net canopy is not present. Net canopies are associated with collapsed nets.
9. diver: Coded "1" if divers were used to help dolphin escape and "0" if not.

Using a random sample of 10,000 observations, we consider first the results under the default costs. A failure to identify correctly a set in which dolphin were killed has the same costs as a failure to identify correctly a set in which no dolphin were killed. In addition, the prior used is the empirical distribution of the binary response variable.

	Predict No deaths	Predict deaths	Model Error
No Deaths	7859	142	.02
Deaths	797	202	.78
Use Error	.09	.41	Overall Error = .10

Table 5.5. Confusion table for forecasting dolphin deaths using equal costs.

Table 5.5 shows the confusion table. Overall, random forests is able to forecast with about 90% accuracy. Given the unbalanced nature of the response, this is not a very impressive feat. It is clear that most of this accuracy comes from the predictions of true negatives (i.e., no dolphin deaths), which random forests incorrectly identifies only about 2 times out of 100. About 78 times out of 100, random forests incorrectly identifies true positives (i.e., dolphin deaths). Were the random forest results used for forecasting by ITTC administrators, ship captains or on-board observers, they would be wrong only about 9 times out of 100 when they forecasted no dolphin deaths, but about 41 times out of 100 when they forecasting dolphin deaths.

Figure 5.15 shows a plot of predictor importance for the response category in which dolphin were killed. It is clear that the presence of a net canopy is the dominant predictor, followed by the length of the backdown procedure, and then two measures of the size of the tuna catch. Given that random forests identifies correctly on 22% of the time sets in which dolphin are killed, the accuracy reductions are large. Shuffling the net canopy variable reduces the model's forecasting accuracy from .22 to .15.

Figure 5.16 shows a plot of predictor importance for the response category in which no dolphin are killed. It is here that random forests stumbles badly.

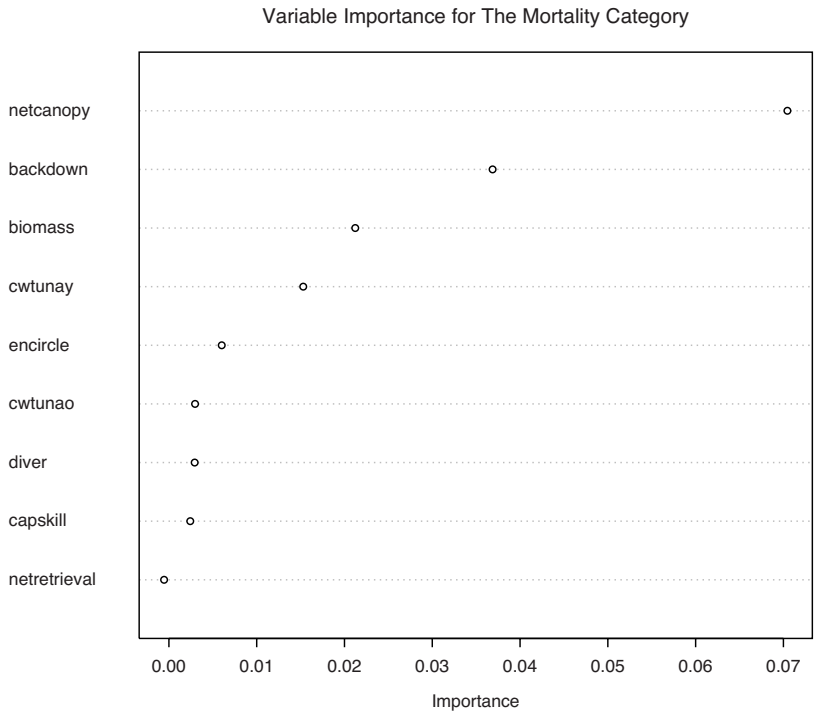


Fig. 5.15. Variable importance when the outcome is dolphin deaths and the costs are equal.

In general the same predictors are important, but their contributions to forecasting accuracy are trivial. A key reason is that it is very difficult for random forests to do better than always concluding that no dolphin were killed.

Partial dependence plots can be constructed for each of the predictors. We will consider just one to illustrate the sorts of insights that can be obtained and to highlight some important limitations. Thus, Figure 5.17 shows the empirical response function for the predictor backdown time. Backdown time is how long it takes for the net, once the tuna are encircled, to be drawn to the boat. Long backdown times are thought to be dangerous for dolphin because they increase the risk of dolphin getting caught in the net and drowning.

Figure 5.17 suggests that for very short backdown times, which are rare and probably reflect serious reporting errors, increases in backdown time are associated with decreases in dolphin mortality. This is a result that should not be taken seriously. For backdown times between about 15 minutes and an hour, increases in backdown time are, as expected, associated with substantial increases in dolphin mortality. Beyond an hour, where again the number of

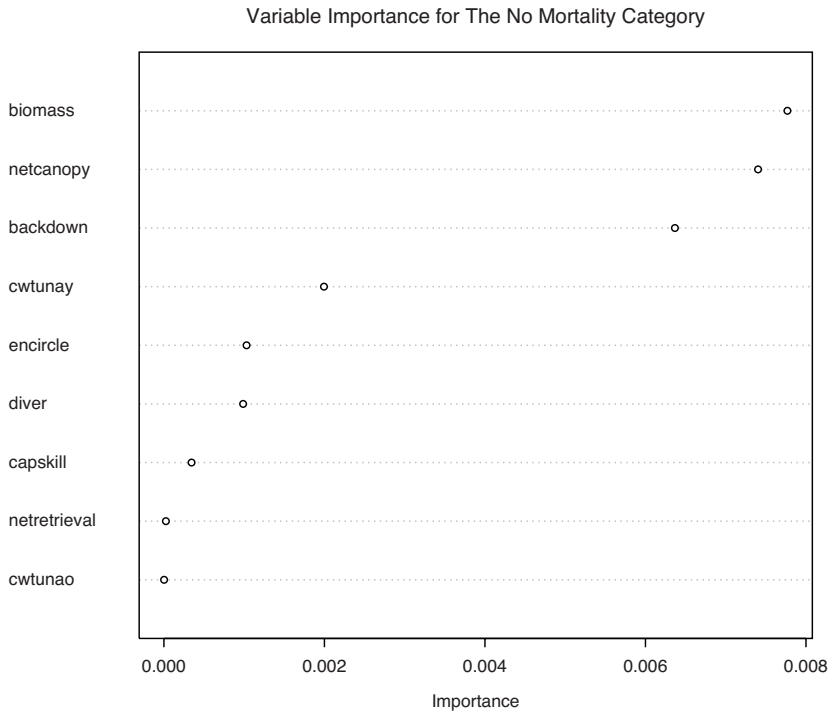


Fig. 5.16. Variable importance when the outcome is no dolphin deaths and the costs are equal.

observations is very few, the relationship essentially become flat. This too should not be taken seriously.

Recall that the units on the vertical axis are not probabilities or conventional logits. Thus, it is difficult to judge in subject matter terms whether the changes in the response shown are large enough to be important. But as shown earlier, the logit units can be transformed back into probabilities, and for Figure 5.17, the change in the logits is large enough to be very significant. When the backdown time is under 20 minutes, the chances of any dolphin deaths are less than 1 in 20 below what is typical. When the backdown time is over 40 minutes, the chances of death are around 15 in 20 above what is typical. So, the probabilities are increased a maximum of about .70.

From the off-diagonal cells in Table 5.5, one can see that there are a little over five false negatives for every false positive. Therefore, false positives are being treated as about five times more costly than false negatives. Discussion with representatives of the ITTC led to the conclusion that false negatives were actually much more costly than false positives. There were few harmful

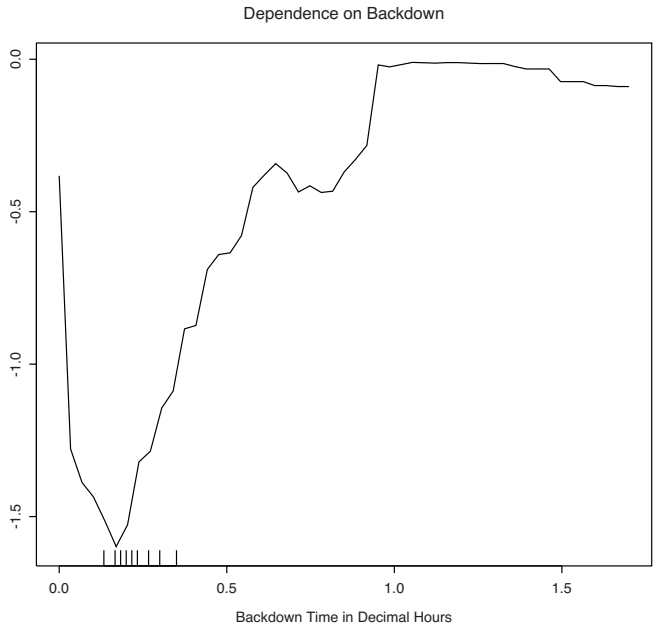


Fig. 5.17. Partial dependence on backdown time when the costs are equal.

consequences from failing to identify sets with no dolphin mortality, but many from failing to identify sets in which dolphin were killed. They suggested a ratio of about one to ten for the ratio of false negatives to false positives.

	Predict No Deaths	Predict Deaths	Model Error
No Deaths	5468	2533	.32
Deaths	271	728	.27
Use Error	.04	.78	Overall Error = .31

Table 5.6. Confusion table for forecasting dolphin deaths using a cost ratio of one to ten.

Table 5.6 shows the confusion table when the one to ten cost ratio is used. The changes are substantial. Just as one would expect, overall forecasting error increases substantially from .10 to .31.

Consistent with the cost applied, random forests now does a much better job identifying sets in which dolphin are killed. The proportion of sets incorrectly identified drops from .78 to .27. At the same time, random forests does much worse identifying sets in which dolphin are not killed. The proportion of

sets incorrectly identified increases from .02 to .32. These changes stem from the new ratio of false negatives to false positives, which by intent is about one to ten.

If forecasts of no dolphin deaths are made, they will be incorrect about 4 times out of 100. If forecasts of dolphin deaths are made they will be incorrect about 78 times out of 100. This properly reflects the new cost ratio. Decision-makers are now more prepared to predict sets in which dolphin will be killed because the costs of false positives are relatively low. Table 5.6 indicates that there will be a bit more than three false positives for every true positive.

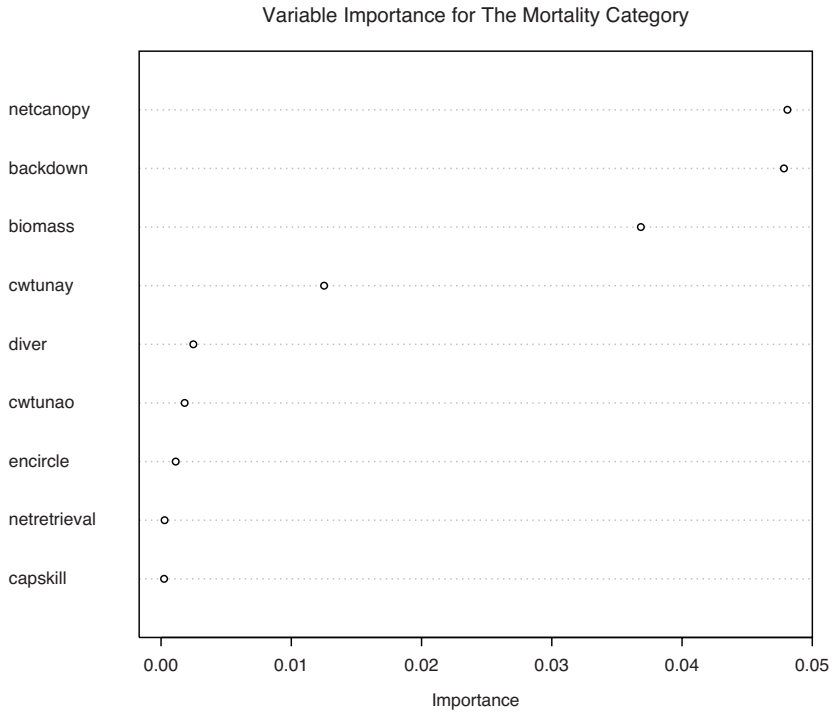


Fig. 5.18. Variable importance when the outcome is dolphin deaths and the costs are one to ten.

Figures 5.18 and 5.19 show the two importance plots. It is again apparent that the predictors matter far more for forecasts of dolphin deaths than for forecasts of no dolphin deaths. But the change in a cost ratio of one to ten has altered a bit the importance of some variables. For example, in predictions of dolphin deaths, the presence of a net canopy and backdown time are now about equally important. Under equal costs, backdown time was a little less

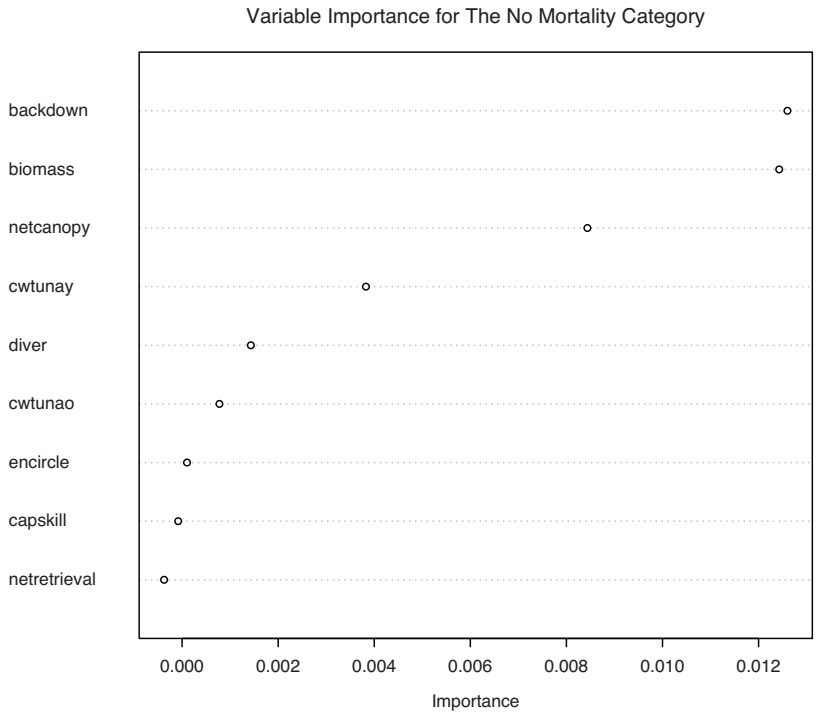


Fig. 5.19. Variable importance when the outcome is no dolphin deaths and the costs are one to ten.

important. And under the new cost ratio, the biomass of the tuna caught has moved up somewhat in importance.

It is not unusual to see the importance of variables change with changes in relative costs. In effect, there is a new weighting of observations. Variables that predict well the observations now given more weight will increase in importance.

Finally, Figure 5.20 shows the partial dependence plot for backdown time under the one to ten cost ratio. The overall shape of the curve is basically the same, but the increase in dolphin mortality is not quite as large.

5.12 An Illustration Using a Quantitative Response Variable

A recent effort was made to count the number of homeless in Los Angeles County (Berk et al., 2008). There are over 2000 census tracts in the county, and enumerators were sent to a sample of a little over 500. The details of the

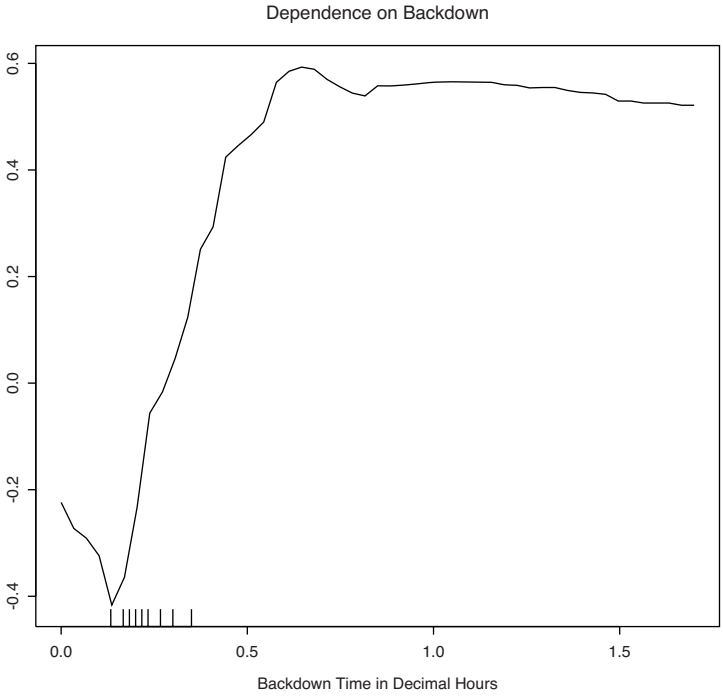


Fig. 5.20. Partial dependence on backdown time when the costs are one to ten

sampling need not trouble us here, and in the end the overall county total was estimated to be about 90,000.

In addition to countywide totals, there was a need to have estimated counts for tracts not visited. Various stakeholders might wish to have estimates at the tract level for areas to which enumerators were not sent. Random forests was used with tract-level predictors to impute the homeless counts for these tracts. About 21% of the variance in the homeless counts was accounted for by the random forests model.

Figure 5.21 is an importance plot for three of the most useful predictors. When the response variable is quantitative, the "external" and "internal" measures of importance differ from when the response variable is qualitative. The external measure, based on the OOB observations, is the average percentage increase in mean square forecasting error over trees when a given predictor is randomly shuffled. The internal measure is the average reduction in the error sum of squares over trees when a given predictor is used to define a split, which can also be called the increase in node purity.

For example, when median household income is shuffled, the mean square forecasting error increases about 7%. For the percentage of dwellings in a tract

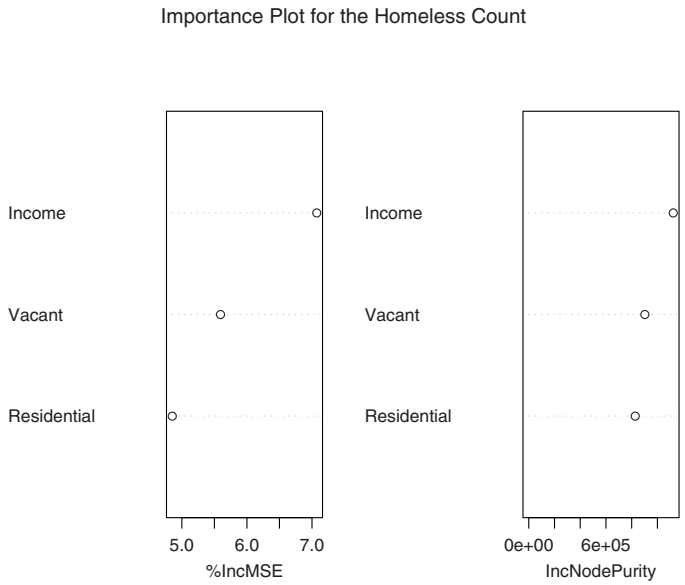


Fig. 5.21. Variable importance when the outcome is the number of homeless in a census tract.

that is vacant, shuffling increases forecasting error about 6%. For the percentage of land that is devoted to residential use, shuffling increases forecasting error about 5%. In this case, therefore, all three variables have about the same impact on forecasting skill. The rank ordering of importance is the same when the contribution to fit is used, but it is far more difficult to tell whether the contributions are large or small. It is difficult to think in raw error sum of squares units.

Figures 5.22 to 5.24 show the partial dependence plots for each predictor. For a quantitative response, the vertical axis is the conditional mean of the response for different values of the predictor in question, with all other variables fixed. Compared to the categorical response variable case, the only feature of the partial dependence algorithm that has changed is the units in which the response is represented.

From Figure 5.22, one can see that the fitted values for the number of homeless individuals in a census tract drops from a high of around 150 when median income is less than about $20,000 a year to around 30 when median income is $50,000 or more. Overall, the relationship is strongly negative. But the drop is precipitous, implying what some have called a "tipping effect."

The visual story is much the same in Figure 5.23. When most of the land in a census tract is not used for residential dwellings, the number of homeless individuals is about 130. That figure drops to about 30 when a quarter of

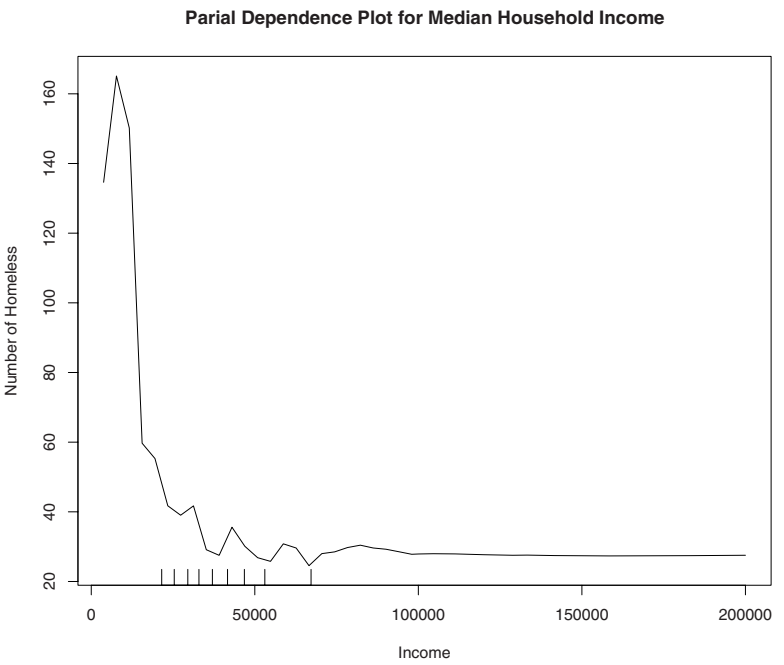


Fig. 5.22. The response function when median household income is the predictor.

the land or more is used for residents. Overall, the relationship is strongly negative with more evidence of a tipping effect.

Figure 5.24 shows a positive association between the percentage of the residential dwellings that are vacant and the number of homeless. When vacancy is near zero, the average number of homeless is about 10 per tract. When the vacancy percent is above approximately 10%, the average count increases to between 60 and 70 (with a spike right around 10%). Once again the change is very rapid.

In summary, a larger number of homeless are to be found in low income census tracts with relatively few occupied dwellings. Perhaps more interesting is that the transition from tracts with few homeless individuals to tracts with many homeless individuals occurs over a very small range of predictor values. An important methodological point is that the highly nonlinear response functions would not likely have been found using conventional regression procedures unless there were a strong a priori belief in a tipping effect and the the ability to specify a functional form that would find it. Given the sharp transition for predictor values that would have been difficult to anticipate, determining the functional form would have probably been difficult.

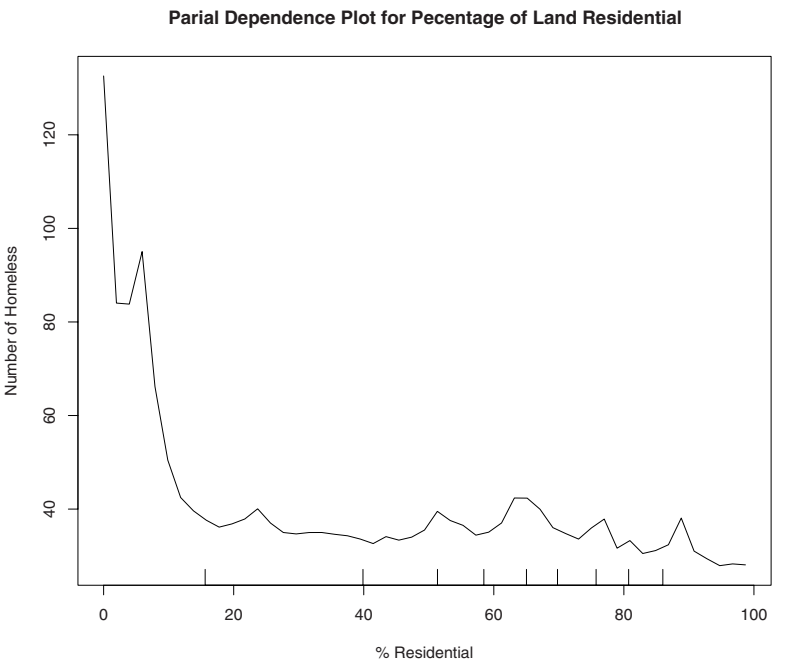


Fig. 5.23. The response function when residential land use is the predictor.

Unfortunately, even with the power of random forests, there is a marked tendency to underestimate the few very largest homeless counts. These census tracts matter a great deal in the overall amount of resources allocated to take care of homeless individuals and how those resources are allocated. As suggested earlier, this is precisely where quantiles might be more instructive than the mean.

Figure 5.25 shows a plot of the actual census tract counts against the fitted census tract counts using the conditional .05 quantile. Overestimates are being treated as far more important than underestimates: about 19 to 1. A 1-to-1 line is overlaid.

The mean absolute disparity between the fitted values and the actual values is 29.4. This is quite large considering that most of the measured homeless counts are under 50. Ideally, moreover, all of the points should fall to the 1-to-1 line. Most of the points fall above the 1-to-1 line, indicating underestimated counts much of the time. Finally, one can see that although the actual counts are sometimes larger than 400, the largest fitted count is a little over 80.

Figure 5.26 shows a plot of the actual census tract counts against the fitted census tract counts using the conditional .50 quantile: the median. Using the

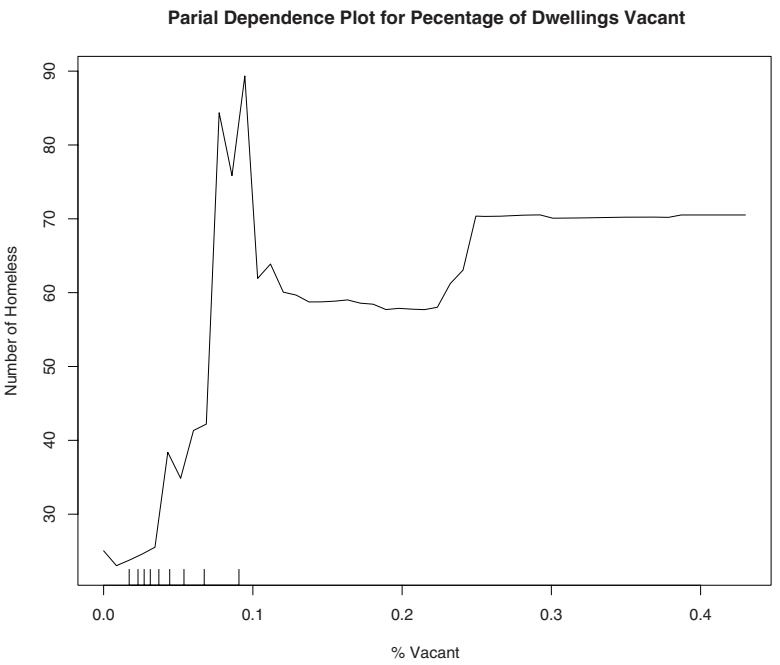


Fig. 5.24. The response function when vacant dwellings is the predictor.

median implies that the costs of overestimates are the same as the costs of underestimates. Again, a 1-to-1 line is overlaid.

Overall the fit looks quite good. The mean absolute disparity is only 3.5. In addition, the fitted counts can now be as large as about 450, which is a clear improvement if large underestimates are a serious concern. However, the very largest counts are still substantially underestimated. Allowing the overestimates and underestimates to have the same costs produces results much like those produced by the conditional mean.

Figure 5.27 shows a plot of the actual census tract counts against the fitted census tract counts using the conditional .95 quantile. Now the costs of underestimates are 19 times larger than the costs of overestimates. A 1-to-1 line is again overlaid.

Virtually all of the points fall below the 1-to-1 line, and the mean absolute disparity is 36.0. Overestimates dominate the plot. There are several fitted counts in excess of 800, which in most cases are also overestimates. On the other hand, the very highest count is fitted perfectly. Clearly, one can have very different fitted values depending on which quantiles are used.

Much as in our earlier discussion, there is no purely statistical way to determine what costs should be used. The costs need to be determined by how

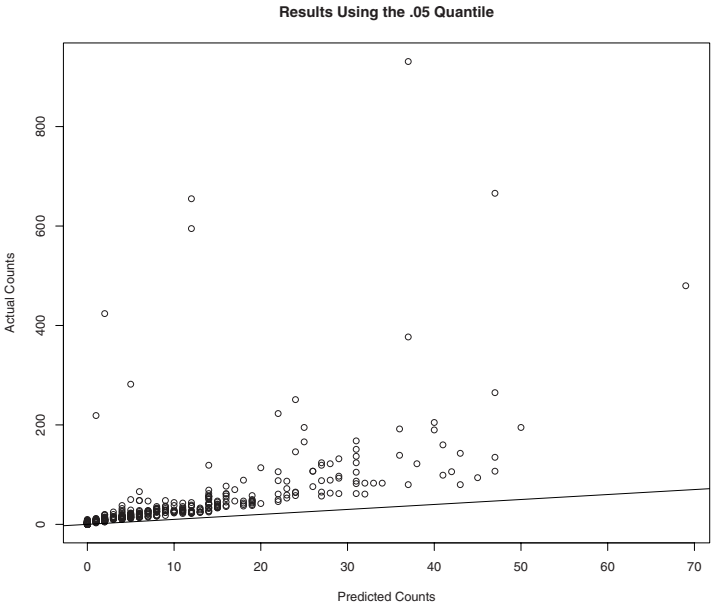


Fig. 5.25. Actual counts plotted against fitted counts using the .05 quantile: MAD = 29.4.

the results are to be used and by how various stakeholders view the consequences of those uses. For example, although using the conditional median minimizes the sum of the absolute deviations between the actual values and the fitted values, the mean absolute deviation assumes that underestimates have the same costs as overestimates. As such, it will be a misleading measure of fit when equal costs do not apply.

Figure 5.28 shows the results when the conditional .80 quantile is used. Underestimates are taken to be four times more costly than overestimates. Stakeholders might find these results the most congenial. The mean absolute disparity of 7.4 is relatively small, and the very largest counts are fitted values about as well as possible, given their variability. For our purposes, however, the statistical point is that quantile regression provides a way to employ asymmetric cost functions with random forests.

Quantile random forests is hardly the final answer to the need for asymmetric cost functions in statistical learning for quantitative response variables. As noted earlier, the trees are grown as usual using the random forests algorithm. Also, one has to be happy with linear loss. Finally, the importance plots and partial dependence plots currently available in the quantile random forests procedure are still those from the underlying random forest algorithm.

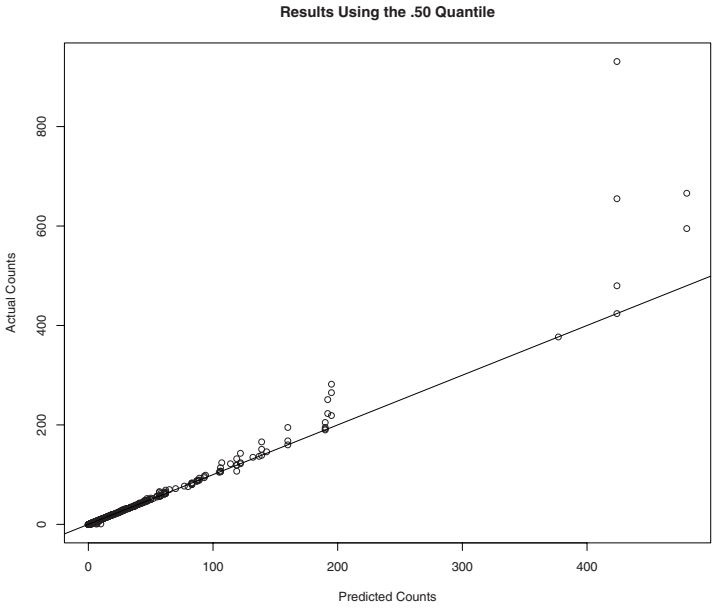


Fig. 5.26. Actual counts plotted against fitted counts using the .50 quantile: MAD = 3.5.

5.13 Software Considerations

At the moment, there are three species of random forests available. One can obtain from Leo Breiman's Web site (http://stat-www.berkeley.edu/users/breiman/RandomForests/) a FORTRAN version of random forests and some supporting documentation. Leo Breiman and Adele Cutler are the software's authors. There are some features that are otherwise not available and some features available elsewhere that are not included. It is perhaps the least user-friendly of the three.

The version of random forests available in R is far more user friendly, has better documentation, and has the key advantage of an R computing environment. It has features that are unique but lacks some of the highly experimental tools found in the FORTRAN version. The R port was undertaken by Andy Liaw and Matthew Weiner. Andy Liaw (andy_liaw@merck.com) is the maintainer.

Quantile random forests, which draws so heavily on conventional random forests, is also an R-based procedure. Quantile random forests (quantregForest) was written and is maintained by Nicolai Meinshausen.

The version of random forests available from Salford Systems (http://www.salford-systems.com/) is by far the most user-friendly. But, the user

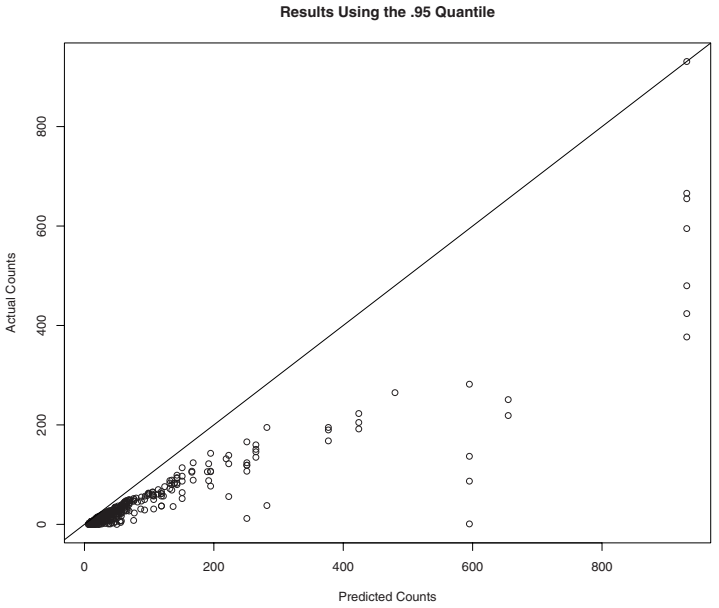


Fig. 5.27. Actual counts plotted against fitted counts using the .95 quantile: MAD = 36.0.

gives up considerable control and in general, the features included are several iterations behind those that can be found in the R version. And unlike the FORTRAN and R implementations, there is a substantial charge for the software.

It is important to stress that random forests is a new procedure still very much under development. Although it is unlikely that its main algorithm will change significantly, lots of the special routines and displays of output will. If history is any measure, one can expect significant updates of random forests at least once a year, often sooner. And unfortunately, it is often difficult for the documentation to keep up. There are also likely to be spinoffs from random forests, such as quantile random forests and others. One can imagine some of these being very handy for certain kinds of problems.

For example, Geurts and his colleagues (2006) propose what they call "extremely randomized trees." No bootstrap sample is used; the full training sample is used to grow each tree. Then the algorithm proceeds as follows.

1. For each potential partitioning, choose a random sample of predictors without replacement.
2. For these selected predictors, choose the break points at random.
3. Compute the reduction in heterogeneity for each predictor at its randomly chosen break point.

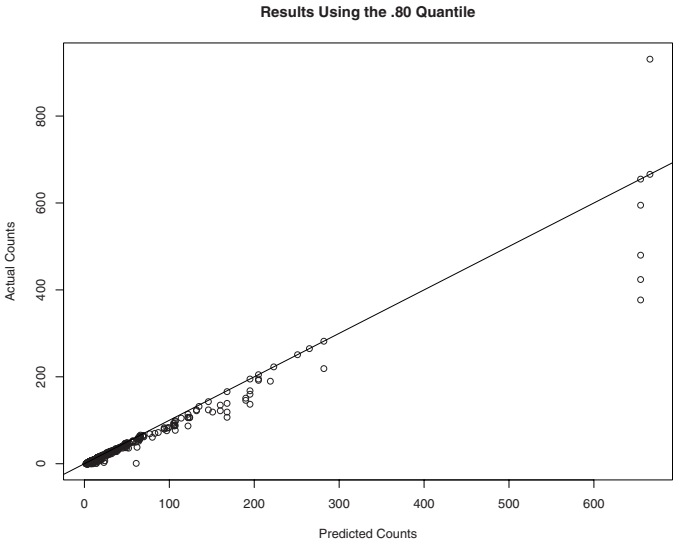


Fig. 5.28. Actual counts plotted against fitted counts using the .80 quantile: MAD = 7.4.

4. Choose the predictor that reduces the heterogeneity the most.
5. Repeat Steps 1–4 for each subsequent split.
6. Average over trees as usual.

The underlying rationale is that by selecting break points at random, greater independence is achieved across the trees in a forest compared to conventional random forests. That is, the sets of fitted values will be less dependent. As a result, instability is more effectively controlled. The price for this reduction in instability can be an increase in the bias because the random break points are not likely to be the optimal break points. Ideally, the reduction in the instability will more than offset the increase in the bias.

When the set of predictors is weak, extremely randomized trees may perform at least as well as random forests; the tradeoff works because random break points do not perform much worse than optimal break points. When the predictors are not weak, extremely randomized trees is not likely be a good choice.

There will also in the future likely be important improvements in how the results from random forests are visualized. For example, it might be useful to have a receiver operating characteristic (ROC) curve that would depend on such things as the relative costs for false negatives to false positives or the values of tuning parameters. Such a plot would have the number of true positives on the vertical axis and the number of false positives on the horizontal axis. The best result would fall in the upper-left hand corner: all true positives and

no false positives. The worst result would be in the lower-right hand corner: no true positives and all false positives. Locations between these extremes would represent different tradeoffs between the two, and one could see how these tradeoffs changed with alterations of the model's features. Drummond and Holte (2006) have suggested an interesting alternative, consistent with much of the earlier discussion, in which on the vertical axis is the (normalized) expected total cost of the classification errors. This too could come in handy.

5.14 Summary and Conclusions

There is growing evidence that random forests is a very powerful statistical learning tool. If forecasting accuracy is one's main performance criterion, there are no other tools that have been shown to consistently perform any better. We consider a chief competitor in the next chapter.

Random forests seems to get its leverage from five features of the algorithm:

1. Growing large, low bias trees
2. Using bootstrap samples as training data when each tree is grown
3. Using random samples of predictors for each partitioning of the data
4. Constructing fitted values and output summary statistics from the out-of-bag data
5. Averaging over trees.

At the same time, very few of random forest's formal properties have been proven, and there remains the nettlesome problem that if one is interested in knowing the $f(X)$, a random forest estimate is not consistent. At a deeper level, the precise reasons why random forests performs so well and why it does better with some datasets than others is fully understood. There is some hard work ahead for theoretical statisticians.

Exercises

5.14.1 Problem Set 1

The goal of this first exercise is to compare the performance of linear regression, CART, and random forests. Construct the following dataset in which the response is a quadratic function of a single predictor.

```
x1=rnorm(500)
x12=x1^2
y=1+(-5*x12)+(5*rnorm(500))
```

1. Plot the $1 + (-5 \times \text{x12})$ against x1. This is the "true" relationship between the response and the predictor without the complication of the disturbances. This is the $f(X)$ you hope to recover from the data.
2. Proceed as if you know that the relationship between the response and the predictor is quadratic. Fit a linear model with x12 as the predictor. Then plot the fitted values against x1. The results show how the linear model can perform when you know the correct function form.
3. Now suppose you do not know that the relationship between the response and the predictor is quadratic. Apply CART to the same response variable using rpart() and x1 as the sole predictor. Use the default settings. Construct the predicted values, using predict(). Then plot the fitted values against x1. How do the CART fitted values compare to the linear regression fitted values? How well does CART seem to capture the true $f(X)$?
4. Apply random forests to the same response variable using randomForests() and x1 as the sole predictor. Use the default settings. Construct the predicted values using predict(). Then plot the fitted values against x1. How do the random forest fitted values compare to the linear regression fitted values? How well does random forests seem to capture the true $f(X)$?
5. How do the fitted values from CART compare to the fitted values from random forests? What feature of random forests is highlighted?
6. Construct a partial dependence plot with x1 as the predictor. How well does the plot seem to capture the true $f(X)$?
7. Why in this case does the plot of the random forest fitted values and the partial dependence plot look so similar?

5.14.2 Problem Set 2

Load the dataset SLID from the *car* library. Learn about the data set using the help() command. Treat the variable "wages" as the response and all other variables as predictors. The data have some missing values you will want to remove. Try using na.omit().

1. Using the default settings, apply random forests and examine the fit quality.
2. Set the argument *mtry* at 4. Apply random forests again and examine fit quality. What if anything of importance has changed?
3. Now set ntrees at 100 and then at 1000 applying random forests both times. What if anything of importance has changed?

4. Going back to the default settings, apply random forests and examine the variable importance plots with no scaling for each predictor's standard deviation. Explain what is being measured on the horizontal axis on both plots when no scaling for the standard deviation is being used. Interpret both plots. If they do not rank the variables in the same way, why might that be? Now scale the permutation-based measure and reconstruct that plot. Interpret the results. If the ranks of the variables differ from the unscaled plot, why might that be? Focusing on the permutation-based measures (scaled and unscaled) when might it be better to use one rather than the other?

5. Construct partial dependence plots for each predictor and interpret them.

5.14.3 Problem Set 3

Load the *MASS* library and the dataset called Pima.tr. Read about the data using help().

1. Apply random forests to the data using the diagnosis of diabetes as the response. Use all of the predictors and random forest default settings. Study the confusion table.
   a) How accurately does the random forests procedure forecast overall?
   b) How accurately does the random forests procedure forecast each of the two outcomes?
   c) If the results were used to forecast either outcome, what proportions of the time would each of the forecasts be incorrect?

2. Construct variable importance plots for each of the two outcomes. Use the unscaled plots of forecasting accuracy. Compare the two plots.
   a) Which predictors are the three most important in forecasts of the presence of diabetes compared to forecasts of the absence of diabetes? Why might they not be the same?
   b) Why are forecasting contributions for the less common outcome generally larger than the forecasting contributions for the more common outcome?

3. Construct and interpret partial dependence plots of each predictor.

4. Suppose now that medical experts believe that the costs of failing to identify future cases of diabetes are four times larger than the costs of falsely identifying future cases of diabetes. For example, if the medical treatment is to get overweight individuals to lose weight, that would likely be beneficial even if the individuals were not at high risk for diabetes. But failing to prescribe a weight loss program for an overweight individual might be an error with very serious consequences. Repeat the analysis just completed but now taking the costs into account by using the stratified bootstrap sampling option in random forests.

a) How has the confusion table changed?
b) How have the two variable importance plots changed?
c) How have the partial dependence plots changed?

6

Boosting

6.1 Introduction

One of the reasons why random forests is so effective for complex response functions is that it capitalizes on very flexible fitting procedures. As a result, it can respond to highly local features of the data. Such flexibility is desirable because it can substantially reduce the bias in fitted values compared to the fitted values from parametric regression, unless based on prior information, the parametric regression happens to hit upon an appropriate functional form.

The flexibility in random forests comes in part from individual trees that can find nonlinear relationships and interactions. Another source of the flexibility is large trees that are not precluded from having very small sample sizes in their terminal nodes. Yet another source of the flexibility is the sampling of predictors. Predictors that work well, but only for a very few observations, have the opportunity to participate.

But as now stated many times, that flexibility comes at a price: the risk of overfitting. Random forests consciously addresses overfitting by using OOB observations to construct the fitted values and measures of fit, and by averaging over trees. Experience to date suggests that this two-part strategy—very flexible fitting functions and averaging over OOB observations—can be highly effective.

But the two-part strategy, broadly conceived, can be implemented in other ways. An alternative method to accommodate highly local features of the data is to give the observations responsible for the local variation more weight in the fitting process. If in the binary case, for example, a fitting function misclassifies those observations, that function can be applied again, but with extra weight given to the observations misclassified. Then, after a large number of fitting attempts, each with difficult-to-classify observations given relatively more weight, overfitting can be reduced if the fitted values from the different fitting attempts are combined in a sensible fashion. Ideas such as these lead to very powerful statistical learning procedures that can compete with random forests. These procedures are called "boosting."

R.A. Berk, *Statistical Learning from a Regression Perspective*,
DOI: 10.1007/978-0-387-77501-2_6,

Boosting gets its name from its ability to take a "weak learning algorithm," which performs just a bit better than random guessing, and "boosting" it into an arbitrarily "strong" learning algorithm (Schapire, 1999: 1). It "combines the outputs from many 'weak' classifiers to produce a powerful 'committee' " (Hastie et al., 2001: 299). So, boosting has some of the same look and feel as random forests.

But, boosting formally differs from random forests in at least four important ways. First, in traditional boosting, there are no chance elements built in. At each iteration, boosting works with the full training sample and all of the predictors. Some recent developments in boosting exploit random samples from the training data, but these developments are enhancements that are not fundamental to the usual boosting algorithms. Second, with each iteration the observations that are misclassified, or otherwise poorly fitted, are given more relative weight. No such weighting is used in random forests. Third, the ultimate fitted values are a combination over a large set of earlier fitting attempts. But the combination is not a simple average as in random forests. Finally, the fitted values and measures of fit quality are usually constructed from the "within-sample" data. There are no out-of-bag observations, although some recent developments make that an option.

To appreciate how these pieces can fit together, we turn to Adaboost, which is perhaps the most widely known boosting procedure (Freund and Schapire, 1996). For reasons we soon examine, the "ada" in Adaboost stands for "adaptive" (Schapire, 1999: 2). Adaboost illustrates well boosting's key features and despite a host of more recent boosting procedures is still among the best classifiers available (Mease and Wyner, 2008).

6.2 Adaboost

Adaboost is the poster child for boosting and provides a useful introduction to the method. It was designed originally for classification problems, which once again are discussed first.

Consider a binary response coded as 1 or –1. Adaboost then has the following general structure. The pseudocode that follows is basically a reproduction of what Hastie et al. (2001) show on their page 301.

1. Initialize the observation weights $w_i = 1/N, i = 1, 2, \ldots, N$.
2. For $m = 1$ to M:
   a) Fit a classifier $G_m(x)$ to the training data using the weights w_i.
   b) Compute: $err_m = \frac{\sum_{i=1}^{N} w_i I(y_i \neq G_m(x_i))}{\sum_{i=1}^{n} w_i}$.
   c) Compute $\alpha_m = \log[(1 - err_m)/err_m]$.
   d) Set $w_i \leftarrow w_i \cdot \exp[\alpha_m \cdot I(y_i \neq G_m(x_i))], i = 1, 2, \ldots, N$.
3. Output $G(x) = \text{sign}\left[\sum_{m=1}^{M} \alpha_m G_m(x)\right]$.

There are N cases and M iterations. $G_m(x)$ is a classifier for pass m over the data. Any number of procedures might be used to build a classifier, but highly truncated trees (called "stumps") are common. The operator I is an indicator variable equal to 1 if the logical relationship is true, and 0 otherwise. The binary response is coded 1 and –1 so that the sign defines the outcome. The classification error for pass m over the data is denoted by err_m.

The value of err_m is transformed into a logit, which then defines α_m. The new weights, one for each case, are then computed as w_i. All cases incorrectly classified are "up-weighted" relative to the previous pass over the data by e^{α_m}. Consequently, Adaboost will pay relatively more attention in the next iteration to the cases that were misclassified. In some expositions of Adaboost (Freund and Schapire, 1999), α_m is defined as $\frac{1}{2}\log(1 - err_m/err_m)$. Then, incorrectly classified cases are up-weighted by e^{α_m} and correctly classified cases are down-weighted by $e^{-\alpha_m}$. In the end, classification is determined by a vote over the M classifiers G_m, with each vote weighted by α_m.

To summarize, Adaboost combines a large number of fitting attempts of the data. Each fitting attempt is undertaken by a classifier using weighted observations. The observation weights are a function of how poorly an observation was fitted in the previous iteration. The fitted values from each iteration are then combined as a weighted average. There is one weight for each fitting attempt, applied to all of the fitted values, which is a function of the overall classification error of that fitting attempt. The observation weights and the iteration weights both are a function of the classification error, however, their forms and purposes are quite different.

There are now several variants on the basic Adaboost algorithm (Friedman et al., 2000). For example, one can think of Adaboost as "discrete" Adaboost because the fitting function produces a binary response. "Real" Adaboost exploits a fitting function that generates class membership probabilities instead. For these probabilities, the log-odds of class membership can be computed, which in turn, are used instead of an assigned class when weights are updated. Because the output from real Adaboost is less lumpy than from Adaboost, a claim is made that the algorithm may perform a bit better. "Gentle Adaboost" is a more robust version of Adaboost whose loss function gives less weight to extreme values. Limited experience to date with realistic datasets suggests that all three procedures classify about equally well. But there are exceptions, and it is possible to construct datasets in which one or the other will perform substantially better. More will be said about such performance assessments later.

6.2.1 A Toy Numerical Example of Adaboost

To help fix these ideas, it is useful to go through a numerical illustration with very simple data. There are five observations with response variable values for $i = 1, 2, 3, 4, 5$ of $1, 1, 1, -1, -1$, respectively.

1. Initialize the observations with each weight $w_i = 1/5$.

2. For the first iteration using the equal weights, suppose the fitted values for observations $i = 1, 2, 3, 4, 5$ are $1, 1, 1, 1, 1$. The first three are correct and the last two are incorrect. The error for this first iteration is:

$$err_1 = \frac{(.20 \times 0) + (.20 \times 0) + (.20 \times 0) + (.20 \times 1) + (.20 \times 1)}{1} = .40.$$

3. The weight to be given to this iteration is

$$\alpha_1 = \log\frac{(1 - .40)}{.40} = \log(.60/.40) = \log(1.5) = .41.$$

4. The new weights are:

$$\begin{aligned} w_1 &= .20 \times e^{(.41 \times 0)} = .20 \\ w_2 &= .20 \times e^{(.41 \times 0)} = .20 \\ w_3 &= .20 \times e^{(.41 \times 0)} = .20 \\ w_4 &= .20 \times e^{(.41 \times 1)} = .30 \\ w_5 &= .20 \times e^{(.41 \times 1)} = .30 \end{aligned}$$

5. Now we begin the second iteration. We fit the classifier again and for $i = 1, 2, 3, 4, 5$ get $1, 1, 1, 1, -1$. The first four are correct and the last one is incorrect. The error for the second iteration is

$$err_2 = \frac{[(.20 \times 0) + (.20 \times 0) + (.20 \times 0) + (.30 \times 0) + (.30 \times 1)]}{1.2} = .25$$

6. The weight to be given to this iteration is

$$\alpha_2 = \log\frac{(1 - 25)}{.25} = \log(.75/.25) = 1.1.$$

7. We would normally keep iterating, beginning with the calculation of a third set of weights. But suppose we are done. The classes assigned are:

$$\begin{aligned} \hat{y}_1 &= \text{sign}[(1 \times .41) + (1 \times 1.1)] > 0 \Rightarrow 1 \\ \hat{y}_2 &= \text{sign}[(1 \times .41) + (1 \times 1.1)] > 0 \Rightarrow 1 \\ \hat{y}_3 &= \text{sign}[(1 \times .41) + (1 \times 1.1)] > 0 \Rightarrow 1 \\ \hat{y}_4 &= \text{sign}[(1 \times .41) + (1 \times 1.1)] > 0 \Rightarrow 1 \\ \hat{y}_5 &= \text{sign}[(1 \times .41) + (-1 \times 1.1)] < 0 \Rightarrow -1. \end{aligned}$$

One can see in this toy example how in the second iteration, the misclassified observations are given relatively more weight. One can also see that the class assigned (i.e., +1 or –1) is just a weighted average of the classes assigned at each iteration. The second iteration had fewer wrong (one out of five rather than two out of five) and so was given more weight in the ultimate averaging. These principles would apply even for very large datasets and thousands of iterations.

6.2.2 A Statistical Perspective on Adaboost

Adaboost has of late been studied by statisticians, (e.g., Ridgeway, 1999; Friedman et al., 2000; Wyner, 2003; Bühlmann and Yu, 2004; Friedman et al., 2004; Zhang and Yu, 2005; Mease at al., 2007; Mease and Wyner, 2008) and in response, by its original inventors (Shapire, 2002). From this recent work several interesting features of Adaboost have been clarified.

Adaboost fits a stagewise additive model using basis functions in much the same spirit as CART and random forests. In CART, the basis functions are indicator variables that determine the optimal splits. Once a split is defined, it is fixed. Later splits have no impact on earlier splits. Each terminal node is characterized by a set of indicator variables that define the basis functions for that node. The weighted sum of the basis functions is the classifier for all of the data.

In random forests, the basis functions are the individual tree classifiers, each a function of X. Earlier trees are unaffected by later trees. Classes are assigned to observations by determining the class most commonly assigned over trees. Votes are summed over trees, with each tree weighted the same.

In Adaboost, as with random forests, the basis functions are the individual "weak" classifiers, each also a function of X. Often these weak classifiers are trees. Earlier classifications are unaffected by later classifications. Classes are assigned by a weighted sum over classifiers, with weights determined by the values of α_m.

Given the broad similarities between Adaboost and random forests, it is not surprising that many of the same concepts can be applied to both. Just as in random forests, for example, there is in Adaboost a margin, which plays much the same role as the margin in random forests. Thus, a larger margin implies less generalization error. There can also be in Adaboost population generalization error, although unlike random forests, there is no convergence to that value as the number of iterations increases without limit. The lack of convergence raises some important issues (addressed shortly) about how Adaboost should be used in practice.

There also are a number of formal links between Adaboost and a variety of statistical procedures that provide a very useful bridge between the two and a statistical framework within which to place boosting (Breiman, 1999; Friedman et al., 2000). With this framework in place, several important conclusions can be derived that have significant implications for work with real data.

Recall that in conventional parametric regression with a quantitative response variable, the goal is to fit conditional means of the response variable. If the intent is description, the regression hyperplane for the training data ideally goes right through the conditional means. If the enterprise is estimation, the regression hyperplane provides unbiased estimates of the conditional means in the population, or the conditional means implied by the underlying stochastic process. As noted in Chapter 1, the same motives can drive

any of the statistical learning procedures we have discussed when there is a quantitative response. When the response variable is categorical, a similar motivating framework can be applied, but the units of the response are different. For example, it is common to fit or estimate the conditional log odds (i.e., conditional logits) of the response categories. What is Adaboost trying to fit?

Hastie et al. (2001: 306–308) show that the Adaboost iterations are implicitly targeting

$$f(X) = \frac{1}{2}\log\frac{\hat{P}(Y=1|X)}{\hat{P}(Y=-1|X)}. \tag{6.1}$$

This is just one-half the usual log-odds (logit) function for $P(Y=1|X)$. The 1/2 implies using the sign to determine the class. In other words, the "solution" Adaboost is seeking is the population conditional probabilities, or if within-sample results are all that matter, the conditional proportions. This is familiar territory. However, the means to the end shown in Equation 6.1 is minimizing the loss function $e^{-yf(x)}$. Adaboost is attempting to minimize exponential loss with the observed class and the predicted class as its arguments. This focus on exponential loss raises at least two important issues.

First, relying on the mathematical relationship between the exponential loss function and conditional probabilities can miss a key point in practice. Mease and Wyner (2008) show that although at each stage the true conditional probability is indeed the minimizer, over stages there can be gross overfitting of the estimated probabilities. Mease et al. (2007) had earlier demonstrated that because classification depends only on the sign of the classifier, the computed probabilities of class membership are pushed toward 0.0 or 1.0 as the number of iterations increases. In other words, even when there is no evidence of overfitting for class membership, there can still be massive overfitting of the conditional probabilities (Buja et al., 2008). Indeed, the massive overfitting is desirable because it implies large margins for the fitted classes. Possible solutions to this form of overfitting are discussed later, but one must be very cautious about making too much of the estimated conditional probabilities.

Second, the exclusive attachment to the exponential loss function naturally raises the question of whether there are other loss functions that might perform better. Hastie and his colleagues (2001: 306–309) show that minimizing negative binomial log likelihood (i.e., the deviance) is also (as in Adaboost) in service of finding the true conditional probabilities, or the within-sample conditional proportions. Might this loss function, implemented as "Logitboost," be preferred?

On the matter of overfitting conditional probabilities, the answer is no. The same overfitting problems surface (Mease et al., 2007). With respect to estimating class membership, the answer is maybe. Hastie et al. (2001: 308–312) show that the Logitboost loss function is somewhat more robust to outliers than the Adaboost loss function. They argue that, therefore, Logitboost may be preferred if a significant number of the observed classes on

the response variable are likely to be systematically wrong or noisy. Biased or noisy measurement could produce large disparities between the observed and fitted classes that would tend to dominate the fit. However, Mease and Wyner (2007) show through simulation counterexamples that this advice can often be wrong. There is no doubt that exponential loss is more vulnerable to outliers in principle, but the implications of this for practice are not yet clear. Perhaps the best advice when analyzing a given dataset is to try both procedures with the training data, and then with test data see which classifies more accurately.

Despite these and other controversies, there are some real gains to be had placing boosting within a statistical framework of loss function minimization. Statisticians have done a lot of thinking about loss functions. We turn to one particularly useful approach shortly.

6.3 Why Does Adaboost Work So Well?

There is no formal stopping rule for Adaboost and as a result, Adaboost can overfit (Jiang, 2004). The number of passes over the data is a tuning parameter that in practice depends on trial and error, often indexed by a measure of fit. One such measure is the cross-validation statistic, but there are several others that each penalize model complexity a bit differently. Often the number of classification errors will decline up to a certain number of passes over the data and then begin to increase. The point of inflection can sometimes be treated as a useful stopping point. But, there is nothing in boosting implying convergence.

Indeed, for a given sample size, "boosting forever" is not consistent (Mannor et al., 2002). But, for a given stopping point, Zhang and Yu (2005) show that under fairly general conditions, boosting will estimate the population generalization error as the number of observations increases without limit. The population characteristic being estimated is essentially the same as Breiman's for random forests, but the number of observations rather than the number of iterations increases without limit. Importantly, the same caveat holds: the proof of consistency says nothing about the quality of the population classifier responsible for generalization error.

There are also some interesting twists on the usual problem of what can be learned from asymptotics about the results from some data on hand. For example, in a given sample, there may be by random selection no observations where there are several critical turning points in the $f(X)$. It is likely, therefore, that at least those turning points will be fitted in a misleading manner, which a proof of consistency cannot usefully address. That is, even if an estimator can be shown to be consistent under certain conditions, a given sample may miss key features of the $f(X)$.

Still, there is broad consensus that Adaboost performs remarkably well. Part of the reason may be that as does random forests, Adaboost provides an

opportunity for many different predictors to usefully contribute. Rather than working with random subsets of predictors, Adaboost reweights the data so that predictors that might have contributed little to the fit in earlier stages may do so for later stages. In the same spirit, a basis function for a given predictor that may work poorly at an early stage may work better at a later stage. Just as in random forests, the result is a very flexible fitting procedure that can help reduce bias in the fitted values. Then in the weighted averaging process, the variance can be brought under better control.

6.3.1 Least Angle Regression (LARS)

One can obtain a useful window on this process through Least Angle Regression (Efron et al., 2004). Although the initial focus for LARS was on model selection, there are some insights to be had on boosting. LARS proceeds in a stagewise fashion in the spirit of forward stepwise regression. But rather than making an all-or-nothing decision about how a prospective regressor should participate in the model, each variable selected has its role somewhat diluted.

Recall conventional forward stepwise regression. For a given response variable,

1. Find the predictor that has the largest absolute correlation with the response.
2. Compute the one-predictor regression equation and the residuals.
3. Find among the remaining predictors the one that has the largest absolute correlation with the residuals.
4. Add that predictor to the model, and compute the two-predictor regression equation and the residuals.
5. Keep adding predictors in this fashion until there is an insufficient improvement in the model's performance.

A usual feature of this approach is that the basis function that produces the largest reduction in the error sum of squares is the one added to the model. Such algorithms are sometimes characterized as "greedy" because by a specified criterion they make an optimal decision at each step that does not necessarily lead to a global optimum. Greedy algorithms have the advantage, however, of being practical and often give very good results. The alternative of searching over all possible models for a global optimum is typically far too taxing and in some cases, effectively impossible.

The full impact of the included predictors is at each step transmitted through the fitted values to the residuals. One result is residuals that are uncorrelated with the regressors currently in the model. Another result is that other predictors correlated with the regressors already selected may have their chances of being included seriously compromised. Their potential fitting capabilities may be to some extent pre-empted by predictors already in the model.

Consider now stagewise regression. Stagewise regression has some of the same look and feel as forward stepwise regression, but there are important differences. As before, there is a response variable and for ease of exposition, it is common to assume predictors that have been standardized to have a mean of 0.0 and a standard deviation of 1.0. Then,

1. Find the predictor that has the largest absolute correlation with the response.
2. Compute a "regression" coefficient proportional to that correlation coefficient.
3. Compute the fitted values and residuals.
4. Find among the remaining predictors the one that has the largest absolute correlation with the new residuals.
5. Compute a "regression" coefficient proportional to that correlation coefficient.
6. Update the fitted values and compute new residuals.
7. Keep adding predictors in this fashion until there is an insufficient improvement in the model's performance.

Underneath Steps 3 and 6 is an updating procedure for the fitted values of the following form,

$$\hat{y} \rightarrow \hat{y} + \varepsilon \cdot \text{sign}(\hat{c}_j) \cdot x_j, \tag{6.2}$$

where $\hat{y}$ is the fitted values of the response, x_j is the predictor selected at a given stage, c_j is proportional to correlation with the residualized response, and ε is a small constant (Efron et al., 2004: 410). Then, residuals are computed as one would expect by subtracting the updated fitted values from the values of the observed response. These residuals serve as the response variable for the next stage.

At each stage, therefore, a new fitting equation with a single new predictor is computed from which new residuals follow. The impact of each new predictor is discounted substantially by the multiplicative factor ε. The result is residuals whose correlation with each newly added predictor is reduced but not eliminated. Prospective predictors, correlated with the included predictors are then more likely to remain in play and have a greater chance of being included in later stages. Also, predictors used in earlier stages can be used again in later stages. In this sense, the "greediness" of the algorithm is reduced.

How is the value of ε determined? To date, there is no formal way to determine the value of ε. The standard recommendation is that ε should be very small (e.g., .01) and that the fitting process should be allowed to run for several thousand stages. There is some experience suggesting that in this manner, a very flexible fitting function will result. In short, the fitted values should be allowed to very gradually arrive at a satisfactory result.

LARS can be seen in part as a computational shortcut for stagewise regression. Rather than incrementing the fitted values in tiny steps over many

stages, LARS only requires up to as many stages as there are regressors. It arrives more quickly at the desired regression function by using different weights in the updating process. At any given stage, the residuals from the previous stage serve as the response varible. But when a new predictor is added to the model, all of the earlier included predictors, as well as the new one, participate in the fitting process. Moreover, they are each forced to have equal impact on the fitted values regardless of what might have happened had, say, forward stepwise regression been applied to the data.

Like for stagewise regression, the LARS residuals are not forced to be uncorrelated with the included predictors, and regressors can have more than one opportunity to contribute to the fitted values. Also, predictor variables not yet in the model have a better chance of being included in later stages. In the language we have been using, weak and highly specialized predictors can participate in the fitting process. The result is a very flexible fitting function.

But, what has all this to do with boosting? In conventional regression, regressors are either included in the model or not. In some circumstances this is too ham-fisted. Predictors compete for the opportunity to be included, and there are clear winners and clear losers. LARS illustrates how that the all-or-nothing strategy can be improved. Often it will be better to blend the impact of a wide variety of predictors rather than choose among them. Random forests and boosting share much the same perspective. And this can be very effective when highly flexible fitting functions are needed and when there is not a clear distinction between the regressors that belong in the model and those that do not. Another benefit is a kind of shrinkage that can increase the stability of the fitted values. As new predictors are added to the model, their weights are discounted and their potential impact on the fitted values is spread across other predictors.

Finally, the links among LARS, the lasso, and boosting mean that one can see boosting in part as a shrinkage procedure (Bühlmann and Yu, 2006; Bühlmann, 2006). If one looks backwards from the final boosting pass through the data, there is shrinkage at work. The shrinkage is more dramatic as one gets closer to the initial iteration.

In summary, there is no definitive explanation of why Adaboost works as well as it does. But it is likely that two factors are important: the weighting of residuals over many passes through the data so that weak and strong predictors can play a role, and the averaging across sets of fitted values that help to reduce overfitting and to increase stability. These assets carry over to a wider range of boosting procedures to which we can turn now.

6.4 Stochastic Gradient Boosting

Boosting can be approached from an unusually wide variety of perspectives (Shapire, 2002). Many different classifiers can be boosted using many differ-

ent algorithms and loss functions. It also seems that boosting is related not just to a number of traditions in statistics but to game theory, linear programming, other core areas in applied mathematics, and computer science. Finally, boosting is "hot." The half-dozen journals or so that publish work on statistical learning have in almost every issue a useful paper on boosting or related approaches.

To have so much new and interesting work is surely a joy for the researchers. But practitioners are faced with the serious problem of needing stable tools already programmed that reflect the best of these recent developments. Yet, it is not even clear at this point which procedures work best for which kinds of data analyses or even any consensus on how "best" is to be defined. And the availability of software depends as much on happenstance as a well-considered response to what the user community needs.

In so confused and volatile an environment, moving from discussions of the various procedures in principle to tools ready for serious practice means making a number of educated guesses about what will be most productive. To that end, the material that follows emphasizes boosting additive trees. Just as in random forests, CART serves as the base classifier. We also limit ourselves to several rather conventional loss functions. Finally, we stay within R and what is one of the best implementations of boosting widely available (Ridgeway 2005), based on stochastic gradient boosting (Friedman, 2001, 2002). Experience to date suggests that stochastic gradient boosting using trees provides a very flexible boosting framework without, in general, sacrificing fitting performance. Moreover, stochastic gradient boosting can be used with either categorical response variables or quantitative response variables, depending on the loss function used. Finally, stochastic gradient boosting is closely linked to a number of common statistical traditions which, given the exposition style of this book, will seem familiar.

The basic logic behind stochastic gradient boosting is very clever. What follows is a first approximation of gradient boosting with a few key details overlooked for now. As has become our didactic practice, we start with a binary response variable.

Suppose that the response variable is binary and coded as 1 or 0. From a regression tree, not a classification tree, fitted values $\hat{y}_i$ can be obtained. For each observation, there is also the observed value of the response y_i. After applying a monotonic transformation to y_i (details later), the transformed values of $\hat{y}_i$ are subtracted from the y_i to obtain what are, in effect, residuals. In the next iteration, a regression tree is fit to these residuals. The new set of fitted values is then added to the old fitted values (details later) to obtain a new set of fitted values. After a sufficient number of such iterations, the last set of fitted values can be used to assign classes. Commonly, observations with $\hat{y}_i > 0.5$ are assigned a "1," and observations with $\hat{y}_i \leq 0.5$ are assigned a "0."

Larger positive or negative residuals imply that for those observations, the fitted values are less successful. When the regression tree attempts to

maximize the quality of the fit overall, it will respond more to the observations with larger positive or negative residuals. In effect, therefore, these residuals serve as weights and hence, provide a key connection to boosting as originally proposed.

Now consider gradient boosting more formally. The discussion that follows on boosting trees draws heavily on Ridgeway (1999) and on Hastie et al. (2001: Sections 4.9–4.11.

A given tree can be represented as

$$T(x;\Theta) = \sum_{j=1}^{J} \gamma_j I(x \in R_j), \tag{6.3}$$

with, as before, the tree parameters $\Theta = \{R_j, \gamma_j\}$, where j is an index of the terminal node, $j, \ldots, J$, R_j a predictor-space region defined by the jth terminal node, and γ_j is the value assigned in each observation in the jth terminal node. The goal is to construct values for the unknown parameters Θ so that the loss function is minimized. At this point, no particular loss is specified, and we seek

$$\hat{\Theta} = \arg\min_{\Theta} \sum_{j=1}^{J} \sum_{x_i \in R_j} L(y_i, \gamma_j). \tag{6.4}$$

How this can be done for a given tree was discussed when CART was examined. The problem now is more difficult. We seek to minimize the loss over a set of trees. We once again proceed in a stagewise fashion so that at iteration m we need to find

$$\hat{\Theta}_m = \arg\min_{\Theta_m} \sum_{i=1}^{N} L(y_i, f_{m-1}(x_i) + T(x_i;\Theta_m)), \tag{6.5}$$

where $f_{m-1}(x_i)$ are the results as of the previous tree. Given the results from the previous tree, the intent is to reduce the loss as much as possible using the fitted values from the next tree. This can be accomplished through an astute determination of $\hat{\Theta}_m = [R_{jm}, \gamma_{jm}]$ for $j = 1, 2, \ldots, J_m$. Thus, Equation 6.5 is a way to update the fitted values in an optimal manner.

Equation 6.5 can be reformulated as a numerical optimization task. In this framework, g_{im} is the gradient for the ith observation on iteration m, defined as the partial derivative of the loss with respect to the fitting function. Thus,

$$g_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x_i)=f_{m-1}(x_i)}. \tag{6.6}$$

Equation 6.6 represents for each observation the potential reduction in the loss as the fitting function $f(x_i)$ is altered. The larger the absolute value of g_{im}, the greater is the change in the loss as $f(x_i)$ changes. So, an effective fitting function would respond most to the larger absolute values of g_{im}.

The g_{im} will generally vary across observations. A way must be found to exploit the g_{im} so that over all of the observations, the loss is reduced the most it can be. One approach is to use a numerical method called "steepest descent," in which a "step length" ρ_m is found so that

$$\rho_m = \arg\min_{\rho} L(\mathbf{f}_{m-1} - \rho \mathbf{g}_m). \tag{6.7}$$

In other words, a scalar ρ_m is determined for iteration m so that when it multiplies the vector of gradients, the loss function from the previous iteration is reduced the most it can be.

The link between the method of steepest descent and gradient boosting is g_{im}. Consider the disparities between tree-generated fitted values and the actual values of the response. Those disparities are a critical input to the loss function. The size of the loss depends on all of the N disparities, but larger disparities make greater contributions to the loss than smaller disparities. Thus, a fitting function will reduce the loss more substantially if it does an especially good job at reducing the larger disparities between its fitted values and the actual values. There is a greater payoff in concentrating on the larger disparities. Thus, disparities resulting from the fitting process play much the same role as the gradients in the method of steepest descent.

And now the payoff. Friedman (2002) show that if one uses certain transformations of the disparities as the gradients (details soon), there is a least squares solution to finding the best parameter values for the fitting function. That is,

$$\tilde{\Theta}_m = \arg\min_{\Theta} \sum_{i=1}^{N} (-g_{im} - T(x_i; \Theta))^2. \tag{6.8}$$

What this means in practice is that if one fits successive regression trees by least squares, each time using as the "response variable" a certain transformation of the disparities produced by the previous regression tree, one can obtain a useful approximation of the required parameters. For a binary outcome, the classifier that results, based on a large number of combined regression trees, is much the same as Adaboost. Moreover, by recasting the boosting process in gradient terms, many useful variants follow.

We turn, then, to the steps involved in gradient boosting as implemented in gbm(), the software R we soon apply. The algorithm is also called stochastic gradient boosting because of the random sampling in Step 2b below.

Consider a training dataset with N observations and p variables, including the response y and the predictors x.

1. Initialize $f_0(x)$ so that the constant κ minimizes the loss function: $f_0(x) = \arg\min_{\kappa} \sum_{i=1}^{N} L(y_i, \kappa)$.
2. For m in $1, \ldots, M$, do Steps a through e.
   a) For $i = 1, 2, \ldots, N$ compute the negative gradient as the working response

$$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}.$$

b) Randomly select without replacement $W \times p$ cases from the data set, where W is less than the total number of observations. Note that this is a simple random sample, not a bootstrap sample. How large W should be is discussed shortly.
c) Using the randomly selected observations, fit a regression tree with J_m terminal nodes to the gradients r_{im}, giving regions R_{jm} for each terminal node $j = 1, 2, \ldots, J_m$.
d) For $j = 1, 2, \ldots, J_m$, compute the optimal terminal node prediction as

$$\gamma_{jm} = \arg\min_{\gamma} \sum_{x_i \in R_{jm}} L(y_i, f_{m-1}(x_i) + \gamma),$$

where region R_{jm} is denotes the set of x-values that define the terminal node j for iteration m.
e) Still using the sampled data, update $f_m(x)$ as

$$f_m(x) = f_{m-1}(x) + \nu \cdot \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm}).$$

where ν is a "shrinkage" parameter that determines the learning rate. The importance of ν is discussed shortly.

3. Output $\hat{f}(x) = f_M(x)$.

Ridgeway (1999) has shown that by using this algorithmic structure, all of the procedures within the generalized linear model, plus several extensions of it, can properly be boosted by the stochastic gradient method. Stochastic gradient boosting relies on an empirical approximation of the true gradient (Hastie et al., 2001: Section 10.10). The trick is determining the right r_i for each special case. The "residuals" need to be defined. Among the definitions of r_{im} are the following.

1. Gaussian: $y_i - f(x_i) \ldots$ the usual regression residual.
2. Bernoulli: $y_i - \frac{1}{1+e^{-f(x_i)}} \ldots$ the difference between the binary outcome coded 1 or 0 and the fitted ("predicted") proportion for the conventional logit link function.
3. Poisson: $y_i - e^{f(x_i)} \ldots$ the difference between the observed count and the fitted count for the conventional log link function.
4. Laplace: $\text{sign}[y_i - f(x_i)] \ldots$ the sign of the difference between the values of the response variable and the fitted medians.
5. Adaboost: $-(2y_i - 1)e^{-(2y_i-1)f(x_i)} \ldots$ not within the generalized linear model so not as easily seen as a kind of "residual."

There are a number of other gradient boosting possibilities. Hastie and his colleagues (2001: 321) provide the gradient for a Huber robust regression. Ridgeway (2005) offers boosted proportional hazard regression in gbm().

Kriegler (2007) has added to gbm() a Laplace loss function that through an analogy to quantile regression, allows for asymmetric loss. More is said about Kriegler's work later.

Stochastic gradient boosting also can be linked to various kinds of penalized regression of the general form discussed in earlier chapters. One insight, implied earlier, is that the sequences of results that are produced with each pass over the data can be seen as a regularization process akin to shrinkage (Bühlmann and Yu, 2004; Friedman et al., 2004). There is less shrinkage with each successive pass over the data.

In short, with gradient boosting, each tree is constructed much as a conventional regression tree. The difference is how the "target" for the fitting is defined. By using disparities defined in particular ways, a wide range of fitting procedures can be boosted.

6.4.1 Tuning Parameters

The stochastic gradient boosting algorithm just described has two important innovations beyond the original version of gradient boosting. First, a page is taken from bagging with the use of random sampling in Step 2b to help control overfitting. The sampling is done without replacement, but as noted earlier, there can be an effective equivalence between sampling with and without replacement, at least for conventional bagging (Buja and Stuetzle, 2006).

The sample size, whether with or without replacement can be a tuning parameter. The issues are rather like those that arise when the number of folds in N-fold cross-validation is considered. And as with N-fold cross-validation, there seems to be no formal and general answer. Practice seems to favor a conventional sample size of N when sampling with replacement and a conventional sample size of $N/2$ when sampling without replacement. But it can make sense for any given data analysis to try sample sizes that also are about 25% smaller and larger and choosing the best sample size based on out-of-sample performance.

Second, it can be very useful to reduce the rate at which the updating occurs by setting ν to a value substantially less than 1.0 (Step 2e). A value of .001 often seems to work reasonably well, but values larger and smaller by up to a factor of 10 are usually worth trying as well. Again, the value of the tuning parameter is usually determined best by out-of-sample performance.

By slowing down the rate at which the algorithm "learns," a larger number of basis functions can be computed. The flexibility of the fitting process is increased, and the small steps lead to shrinkage at each pass through the data. A cost is a larger number of passes through the data. Fortunately, one can usually slow the learning process down substantially without a prohibitive increase in computing.

Third, a tuning parameter that also affects the flexibility of the fitting function is the "depth" of the interaction variables desired: no interactions, two-way, three-way, and so on. In other words, by allowing for what are, in

effect, sets of product variables, one can increase the "dictionary" of basis functions evaluated. This may seem unnecessary because CART already has the capacity, at least in principle, for building interaction effects basis functions. But, depending on how the partitioning proceeds and on the ways in which predictor variables are related to one another, CART may fail to find some needed interactions or represent them improperly. For example, it may miss entirely a given two-way interaction or represent it as a three-way interaction. By explicitly building in interaction variables, one increases the chances that for many passes through the data CART will get it right.

The price, once again, can be computational. For example, one could include main effects plus all two-way and three-way interaction effects. But even for a small number of predictors, all two-way interactions alone will dramatically increase the number of terms evaluated at each CART partitioning of the data. In practice, therefore, it is rare to go beyond all two-way interactions. And unless the response function is thought to be rather complex, including only main effects may well suffice.

Fourth, yet another tuning parameter that affects fitting function flexibility is the minimum number of observations in each tree's terminal node. Smaller node sizes imply larger trees and a more flexible fitting function. Minimum terminal node sizes of between 5 and 15 seem to work reasonably well in many settings, but it can be worth experimenting with somewhat larger terminal node sizes if computational constraints are significant and if the number of observations used to construct each tree is large. The risk is that with larger terminal node sizes and the smaller trees that can result, some important nonlinearities may be missed.

Finally, the number of passes over the data needs to be determined. Because there is no convergence and no clear stopping rule, the usual practice is to run a large number of iterations and inspect a graph of the fitting error (e.g., residual deviance) plotted against the number of iterations. Usually, the error will decline rapidly at first and then level off. It can even start to increase when the number of iterations is very large. If there is an inflection point at which the fitting error starts to increase, the number of iterations can be stopped just short of that number. If there is no inflection point, the number of iterations can be determined by when reductions in the error effectively cease.

Determining the number of iterations is rarely a serious problem in practice. One proceeds in steps. Several hundred iterations are run, and the performance of the fitting procedure examined. There are often useful tools, such as cross-validation statistics, to help evaluate performance. If the results are unsatisfactory, more iterations are run. This process stops when the performance of the fitting procedures no longer seems to be improving. Stochastic gradient boosting results can be quite robust to the number of iterations used, once it is apparent that there are no important gains to be made. But there are exceptions. In particular, when the response is binary and interest centers

on the fitted probabilities, the fitted probabilities can be quite sensitive to the number of iterations. An example is provided later.

There are some important relationships between the tuning parameters. The usual goal of a data analysis is to construct a set of fitted values with low bias and low variance. Larger trees, higher-order interaction variables, and smaller steps can contribute to reducing the bias. Smaller steps can also help reduce the variance by not allowing a few sets of widely varying fitted values to destabilize the procedure, and by indirectly increasing the need for more passes through the data. Random sampling, which will increase the independence between the sets of fitted values, also can help increase stability.

But exactly how the possible values for each of the tuning parameters should be tuned as a group is not apparent. Is one better off, for instance, to proceed with larger trees and small learning steps? More generally, can certain values for one tuning parameter compensate for certain values for the another? At this point, it is difficult to find clear guidance (Buja et al., 2008).

To summarize, there are several tuning parameters associated with stochastic gradient boosting. Fortunately, much of the available software comes with sensible defaults, and it is often a good idea to stick with these, at least at first. Then some trial-and-error tuning can also be useful. The only tuning parameter likely to need immediate attention is the number of trees to grow. The program gbm(), for example, offers useful information on how many trees are needed, but the user is free to do what seems appropriate. Perhaps the most important message is that gradient boosting can be quite forgiving in general with respect to its tuning parameters.

6.4.2 Output

The key output from stochastic gradient boosting is much the same as the key output from bagging: predicted classifications, predicted probabilities, error rates, and confusion tables. However, unlike bagging and random forests, there are not the usual out-of-bag observations. Therefore, the confusion tables commonly depend on resubstituted data; the data used to build the model are also used to evaluate its performance. As a result, it can be important to have both a training dataset and a test dataset. Confusion tables should be constructed from the test data set. If a simple random sampling option is available, a kind of out-of-bag data is available for evaluation. Recall these are not what is excluded from random samples drawn with replacement, but a fraction of the total training dataset not chosen when a subset of the observations is selected for each tree. Exactly how these data are used will depend on the software.

Just as for bagging and random forests, the use of multiple trees means that it is impractical to examine tree diagrams to learn how individual predictors perform. The solutions currently available are much like those implemented for random forests. There are partial dependence plots that are effectively the same as those used in random forests. However, these plots must be treated

cautiously when the outcome variable is binary. Recall that in an effort to classify well, boosting can push the fitted probabilities away from .50 toward 0.0 and 1.0, and in the case of stochastic gradient boosting, the fitted probabilities can be very sensitive to values of the tuning parameters. Consequently, the fitted probabilities can be misleading. For partial dependence plots with binary predictors, the vertical axis is a function of these fitted probabilities, usually in a logit metric. If the probabilities are suspect, so are the logits.

There are also importance measures for each predictor. The exact form these importance measures can take depends on the software used. But one common option is the reduction of the loss function normalized to 100. The software stores how the loss decreases when each predictor is chosen for particular splits over trees. The average decrease over trees is the raw contribution each predictor makes to the fit. Then these contributions are summed, and each contribution is reported as a proportion of the total. In gbm(), there is on a somewhat experimental basis a random shuffling approach to importance based on predictive skills, but to date it does not use the out-of-bag observations. So it does not represent true forecasting accuracy. Recall that for random forests, importance is defined by contributions to forecasting skill.

6.5 Some Problems and Some Possible Solutions

Because there are so many different kinds of boosting, it is difficult to arrive at any overall assessments of strengths and weaknesses. Moreover, the menu of boosting options continues to grow partly in response to concerns about the performance of older boosting methods. Nevertheless, a few provisional observations may be useful for practitioners.

6.5.1 Some Potential Problems

Boosting is a very powerful tool whose reach will no doubt expand in the near future. For many data analysis problems, it performs well and can be a legitimate competitor to random forests when either approach could be properly applied. But boosting also has some serious drawbacks.

As with random forests, the existing proofs of consistency are not fully satisfying. Suppose in the population there is some function of X, $h(X)$, constructed that links inputs of outputs. Under certain reasonable conditions, including a training dataset that is a random sample from that population, boosting will provide a consistent estimate of $h(X)$. But unless $h(X)$ is the same as the true mechanism $f(X)$ linking inputs to outputs, the function estimated from the training data will not be a consistent estimate of $f(X)$. That is, in large samples boosting can get the wrong function approximately right.

Boosting also can overfit the data. Unlike random forests, there is no mechanism in boosting for capitalizing on random samples of the data and then averaging the results over these samples. Some implementations of boosting have the option of cross-validation measures of fit or other measures that can provide useful guidance on when to stop the boosting process. But even a very good cross-validation stopping rule does not necessarily imply that all is well. The problems with binary outcomes and the fitted probabilities noted earlier are an instructive example.

A related matter is that costs are addressed solely within the functional form of the loss. In Adaboost, for example, classification errors are weighted exponentially but symmetrically. There is no distinction between false positives and false negatives and hence, no way to take their different costs directly into account. In stochastic gradient boosting, classification problems are transformed into regression problems when the residuals are defined. Thus, fitting errors as used in the algorithm are no longer categorical, and there are no longer false negatives and false positives. And the loss functions are once again symmetric. A positive residual of a given size is treated the same as a negative residual of the same size.

Similar issues carry over when the response is quantitative. Although the concepts of false negatives and false positives no longer apply, one might still wish for an asymmetric loss function. If the intent, for instance, is to characterize how the number of homeless people in a census tract is related to features of that tract, overestimates of the number of homeless might have very different consequences from underestimates of the number of homeless. Homeless advocates would perhaps see underestimates as more costly than overestimates. Local public officials might take the opposite view. But neither would like see the costs of overestimates and underestimates treated the same (Berk et al., 2008)

In short, the inability to take asymmetric costs into account means that symmetric costs are being assumed. In practice, this can be untenable. A closely related consequence is that there is no principled way to address the problems that can follow when a response variable is highly skewed. For classification exercises, then, it can be very difficult for boosting to perform better than assigning classes solely from the modal response variable category.

Finally, tuning parameters on occasion can make an important difference in the results. Then, one has no choice but to experiment with different sets of tuning parameter values. Unfortunately, this can be at best a trial-and-error process with too often no definitive resolution.

6.5.2 Some Potential Solutions

Many of boosting's vulnerabilities, just as for any statistical procedure, are exposed by inadequate data. Stated a bit differently, it will be rare indeed, for boosting to be able to solve problems stemming from weaknesses in the information on which it operates. Little more need be said about the importance

of large samples, a rich set of predictors, and accurate measurement. Better data are always better, and data analysis difficulties that may seem to result from the boosting procedure applied, can be remedied if by data of higher quality.

The difficulties that can arise by assuming symmetric costs and/or working with highly skewed response variables can be addressed within the same broad framework. The key is to allow for asymmetric costs. For stochastic gradient boosting and quantitative response variables, Kriegler (2007) suggests attaching weights to the loss functions that can capture asymmetric costs. These weights, in turn, are carried forward when the empirical gradients are constructed so that the CART fitting process at each pass through the data takes them into account. To date, this idea has been employed for the Laplace, Gaussian, and Poisson distributions. Applications to real datasets look promising. One can use the weights to make the costs of forecasting errors responsive to policy and where appropriate, use such costs to adjust for skewed distributions. Asymmetric weighting for the Laplace distribution has been implemented in gbm(). An illustration is provided later.

When the response variable is binary, Mease and his colleagues (2007) argue for weighting the classification errors directly and asymmetrically within an Adaboost (not stochastic gradient boosting) framework. Imagine that one can estimate accurately the probability that a given case is in a given class. It is common to assign that case to a particular class if the associated probability is greater than .50. As noted in earlier chapters, the .50 threshold implies that the costs of false positives and false negatives are the same, and by raising the threshold above or below .50, asymmetric costs can be taken into account. For example, if the threshold were placed at .75, it would imply that the costs of falsely placing a case in the specified class are three times higher than the costs of incorrectly failing to place a case in that class. If one thinks of these thresholds as quantiles, there is a direct connection between the use of such quantiles and the use of costs in classification exercises. Using the quantiles to classify has been called quantile classification (Mease et al., 2007).

In practice, quantile classification is undertaken by oversampling or undersampling in much the same way it is done in random forests. The algorithm is called Jous-Boost and is available in R. The direct links between asymmetric costs, quantile classification, and disproportional stratified sampling allow one to implement costs sensitive boosting within an Adaboost framework. Moreover, one can then obtain appropriate estimates of the probability function. The details can be found in a paper by Mease and his colleagues (2007).

About all that can be said about problems with determining the values of tuning parameters is that it is important to be systematic in one's search of the parameter space. Increasingly, there is software to aid in this process. The procedure tune() in the *e1071* library is one example. It can also be important to appreciate at a deeper level that one usually tunes in response to some function of fit quality. For many applications this is appropriate. But there is no necessary connection between fit quality and scientific or

policy responsiveness. As noted several times in earlier chapters, a better-fitting model may be less instructive than a worse-fitting model. Tuning for fit quality, therefore, is no assurance of sensible results. Finally, one must be cautious about boosting output that is highly sensitive to values of the tuning parameters. For example, one might reasonably decide that the fitted probabilities from a binomial model should not be used or interpreted. And, it would not be a good idea under these circumstances to use fitted probabilities to construct propensity scores (McCaffrey et al., 2004).

6.6 Some Examples

6.6.1 A Garden Variety Data Analysis

Among the most common kinds of analyses in the social sciences are regressions of wages on various biographical variables. We turn to some survey data from the Panel of Income Dynamics to do just such an analysis. We boost using a Gaussian loss function in gbm() to provide a relatively straightforward illustration.

Figure 6.1 shows a boosting performance plot. On the horizontal axis is the number of iterations. On the vertical axis is the change in the normal log-likelihood computed, in this case, from out-of-bag observations. These are the observations not used when the fractional simple random samples were drawn for each tree. They can provide a more conservative assessment of how well the iterations are doing than the resubstituted data. Because the point is to determine how well the interations are doing with the data actually being processed, it is not clear that a more conservative estimate is called for. No measure of fit is being computed that will be generalized to out-of-sample data. In this instance, and as expected, the big improvements come early with no substantial gains after about iteration 4000.

The purpose of Figure 6.1 is to see how the fitting proceeds as the number of iterations increases and to choose a cutoff point. If there is evidence that the performance has not bottomed out, additional iterations can be undertaken. If the performance curve has become effectively flat, there is important information about the useful number of additional iterations needed. Iterations beyond the cutoff point can be discarded.

Commonly, there is statistic a computed from OOB data or through cross-validation that evaluates whether the improvement in performance in a given iteration is worth the increase in complexity. Recall that each additional iteration can be viewed as adding another basis function, which makes the fitting procedure more complex. In this case, the cutoff was determined to be iteration 7746.

Figure 6.2 is an importance plot. Importance is measured by the reduction in the log-likelihood attributable to each predictor, then normalized so that the contributions to the fit add to 100. Recall that for CART the contribution

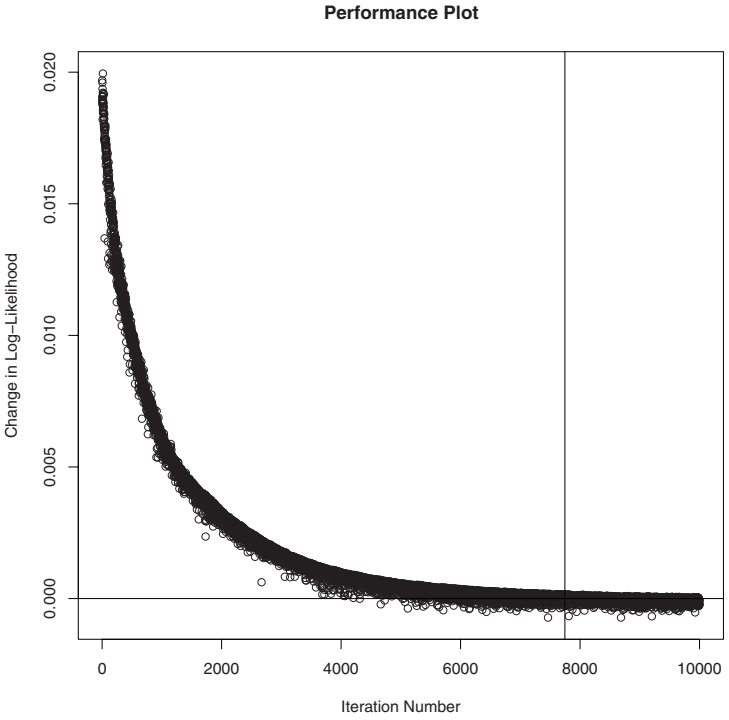


Fig. 6.1. Performance of Gaussian boosting for wages.

of each predictor to the fit of a given tree can be easily calculated. When a predictor is chosen as the splitting variable, the reduction in heterogeneity is determined. The sum of such reductions over the entire tree is that predictor's importance. Random forests averages each predictor's importance over trees. Stochastic gradient boosting, as implemented in gbm(), does the same. In Figure 6.2, a little more than 50% of the fit can be attributed to age. Education accounts for about 35% of the fit. Sex accounts for about 12% of the fit. Language spoken makes almost no contribution.

Finally, Figure 6.3 is the partial dependence plot for age. The vertical axis is in dollars per hour. The horizontal axis is in years of age. One can see that after about age 20, increases in age are associated with increases in wages, but that after about age 40, the relationship flattens out and after age 60, may even decline a bit. Note how one would have been misled had a linear relationship been assumed, and a quadratic form would have only done a little better. Even a cubic polynomial, had that been anticipated, would have missed some of the more interesting features of the relationship, such as the flat part up to about age 20.

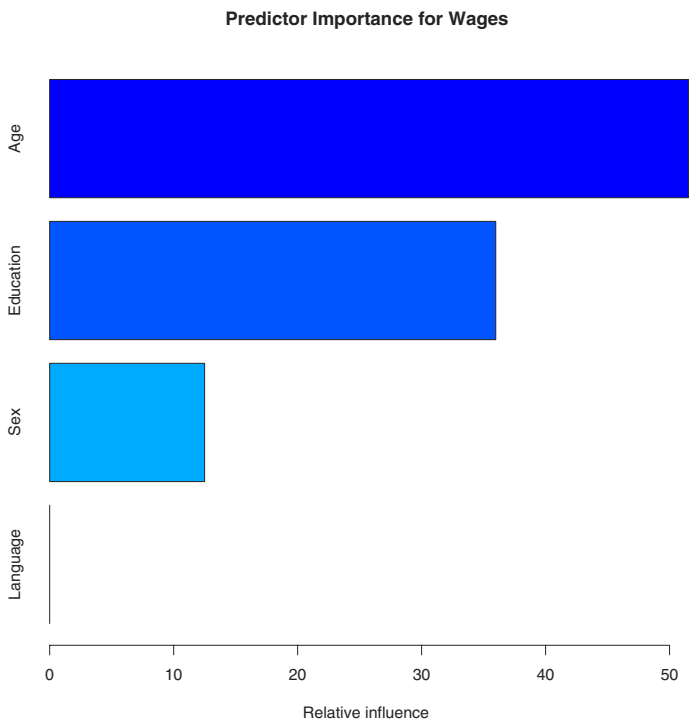


Fig. 6.2. Predictor importance for Gaussian boosting for wages.

In general, it is useful to construct partial dependence plots for all quantitative predictors as long as there are a sufficient number of different predictor values and a substantial number of observations for each. Recall that there is often no point in trying to overlay a smoother when the predictor values are few. But, a lot depends on the complexity of the partial response function. If the function is simple, a partial dependence plot based on few unique predictor values can be helpful. One must also consider whether there are a sufficient number of observations for each unique predictor value. Regions where the data are sparse risk unstable results that can make the response function look more complex than it really is.

It is not appropriate to construct dependence plots for the categorical variables such as sex and language. The values on the horizontal axis would be meaningless. Bar charts are a more useful option. For each category, the conditional mean is plotted. Recall that this is the strategy employed by random forests.

There are no confusion tables for quantitative response variables. But in principle, one has access to all of the usual regression diagnostics. For this

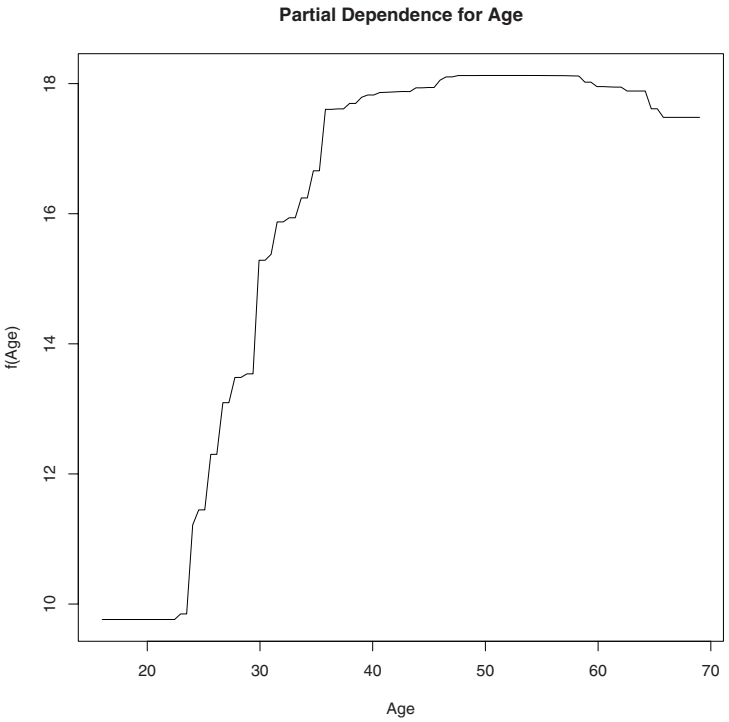


Fig. 6.3. Partial dependence on age for Gaussian boosting for wages.

boosted Gaussian regression, about 35% of the variance is accounted for by functions of the predictors. A conventional linear regression accounted for about 29% of the variance. Boosting clearly improves the fit in this example, although it also uses a larger effective number of parameters. Most of the improvement comes from the nonlinear relationship between age and income. Boosting can help most when one or more relationships between the response variable and predictors is complex.

Figure 6.4 shows four common plots used to evaluate the quality of a regression fit. Beginning with the plot in the upper-left corner, it is clear that the variance in wages increases with the average wage. This is confirmed by the plot at the lower left corner. In both, there also seems to be some evidence of a cluster of outliers suggesting an omitted categorical predictor. The plot on the upper-right corner indicates that the residuals are quite close to normal. The plot on the lower-right hand corner initially gives the impression that there are several influential observations, but the values they have for Cook's distance are very small.

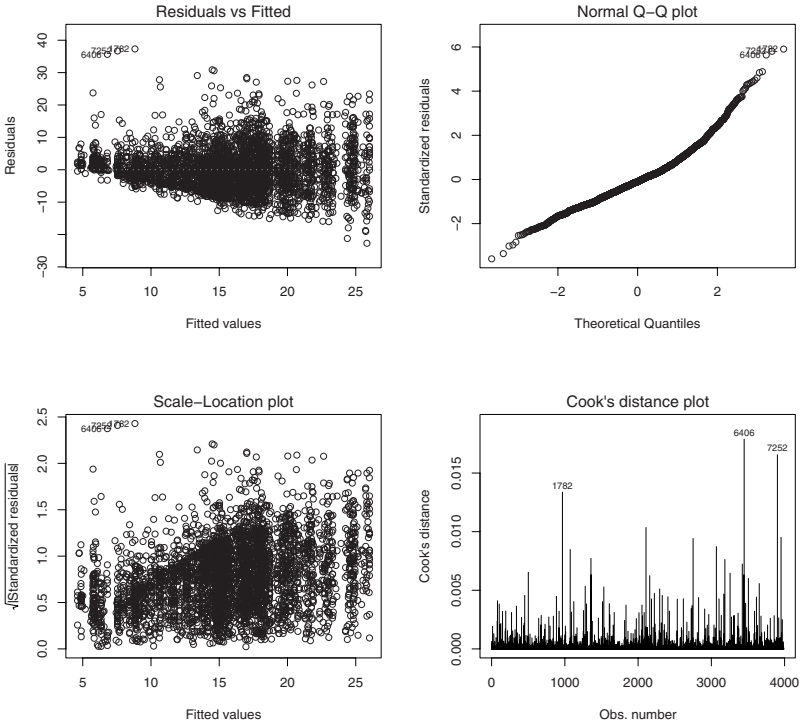


Fig. 6.4. Diagnostic plots for Gaussian boosting of wages.

We make more of such diagnostics in a later example. For now it may suffice to note that the importance of these diagnostics depends in part on the goals of the analysis. Just as in conventional regression, whether the residuals are normal, for instance, will typically not matter much unless the sample is small and traditional hypothesis tests and/or confidence intervals are desired.

6.6.2 Inmate Misconduct Again

Although boosting can be expected in general to perform at least as well as Gaussian regression models, it will sometimes shine when such a conventional regression model does not fit the data very well. The boosting process can improve the fit, sometimes dramatically. The same holds for binomial regression models. But when the response variable is highly unbalanced, there can be serious problems. To make this point, we return to the prison inmate data. We once again consider inmate misconduct using the same predictors as before.

Figure 6.5 shows a boosting performance plot. As expected, the big improvements come early with few real gains after about iteration 4000. The cutoff was determined to be iteration 6148.

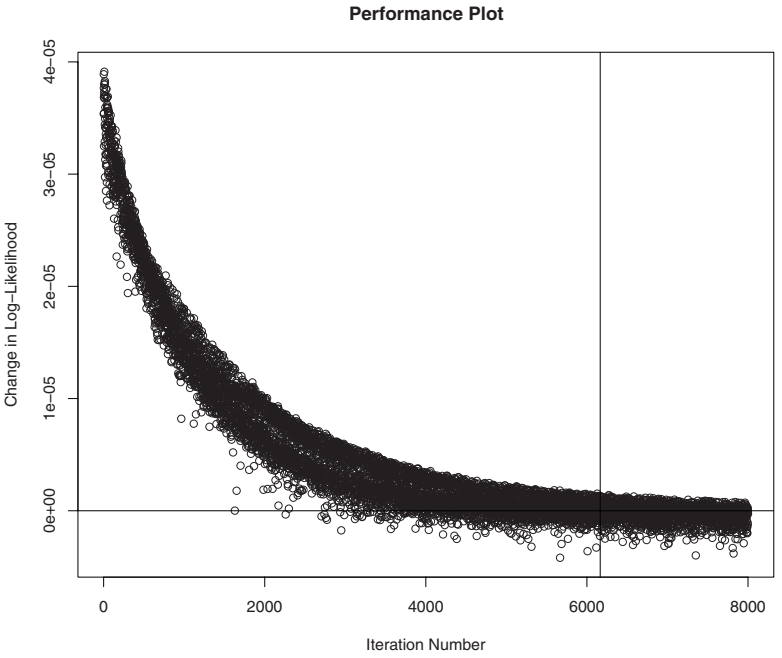


Fig. 6.5. Performance of binomial boosting for inmate misconduct.

Table 6.1 shows the confusion table using the training data, not OOB data or test data. Overall, the fit is quite good. Only about a fifth of the data are misclassified. However, all of the success comes from predicting the nomisconduct class well. And this is pretty easy to do with no predictors all because if the marginal distribution alone is used, and no misconduct is predicted, about 21% of the cases will be identified incorrectly. Moreover, boosting fails miserably when trying to identify inmates who engage in misconduct. Out of every 100 inmates who engaged in misconduct, only about 9 are correctly identified as such.

Table 6.1 is not necessarily an accurate rendering of how the classifier would perform in practice because only training data are used to construct the table. But it is likely from the reported use errors that if no misconduct was predicted, the forecast would be correct the vast majority of time. If misconduct was predicted, the forecast would be correct somewhat more than half the time.

	None Predicted	Misconduct Predicted	Model Error
No Misconduct	3745	62	0.01
Misconduct	900	99	0.91
Use Error	.19	.39	Overall Error = .21

Table 6.1. Confusion table for binomial boosting of inmate misconduct.

But the errors in use, as well as the model errors, depend on the costs of classification errors. And one can see where at least part of the problem lies. The cost ratio of false negatives to false positives is approximately .02. About one false positive is equal to about 50 false negatives. Recall that this is completely upside down from the point of view of corrections officials. And there is currently no way to intervene in the stochastic gradient boosting process and alter these relative costs.

In short, this is about the best that boosting is likely to do and indeed, probably overly optimistic because the confusion table is constructed from the training data. On these same data with the same predictors, random forests performs a bit better, but neither really shines with the default ratio of false negatives to false positives. Costs must be better taken into account. Then, using the cost ratio favored by prison administrators, random forests does dramatically better predicting the true positives.

Figure 6.6 shows predictor relative importance through their contributions to the fit. Sentence length dominates, followed by the two age variable, and gang activity is close behind. This is roughly consistent with our earlier random forest results but difficult to compare directly because Figure 6.6 is derived from contributions to the fit, not forecasting accuracy.

As one illustration of an estimated response function, Figure 6.7 shows the partial dependence plot for sentence length. The vertical axis is in logits as previously defined for partial dependence plots. Recall, $f_k(X) = \log[\mathrm{p}_k(X)] - \frac{1}{K}\sum_{k=1}^{K}\log[\mathrm{p}_k(X)]$, where $\mathrm{p}_k(X)$ is the proportion of observations in category k. However, because of the sensitive nature of the conditional probabilities, it is not clear how seriously one can take the logit values shown. For what it may be worth, inmate misconduct increases rather linearly with sentence length up to a sentence of around six years and then levels off.

We now repeat the analysis using very serious misconduct as the response. Recall that such behavior is very rare. A bit less than 3% of the inmates are reported for incidents of very serious misconduct. Figure 6.8 shows how the boosting algorithm performs. Many fewer iterations are required this time. Also, it is clear from the wider vertical spread of the points that the boosting results are much less stable than before. A likely explanation is that the margins associated with each iteration are substantially smaller and if so, it suggests that the boosted model is not faring well.

Indeed, boosting binomial regression fares quite badly in this case. The confusion table reproduces the marginal distribution of the response because

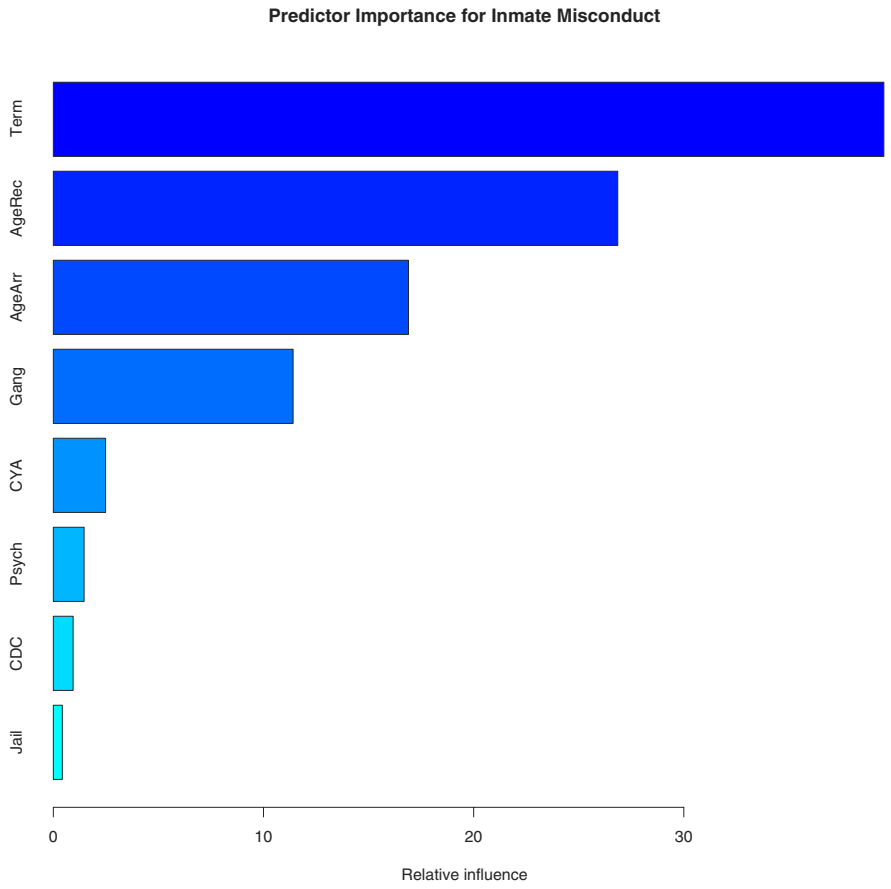


Fig. 6.6. Predictor importance for binomial boosting for inmate misconduct.

not a single inmate is identified as have engaged in very serious misconduct although 138 actually had. Figure 6.9 underscores how bad the performance is. The figure is a histogram for the subset of cases in which there actually was an incident of very serious misconduct. On the horizontal axis are the fitted probabilities. The largest of these values is less than .20, whereas a value of more than .50 is needed for an inmate to be classified as a serious risk. About the best that can be said about the analysis is that because no cases of very serious misconduct are correctly identified, the potential problems with fitted probabilities pushed toward 0 and 1 do not materialize.

Figure 6.10 shows the relative importance of each predictor for the quality of the fit. The pattern is largely the same, but gang activity has moved up to second place, and the gap in importance between sentence length and the

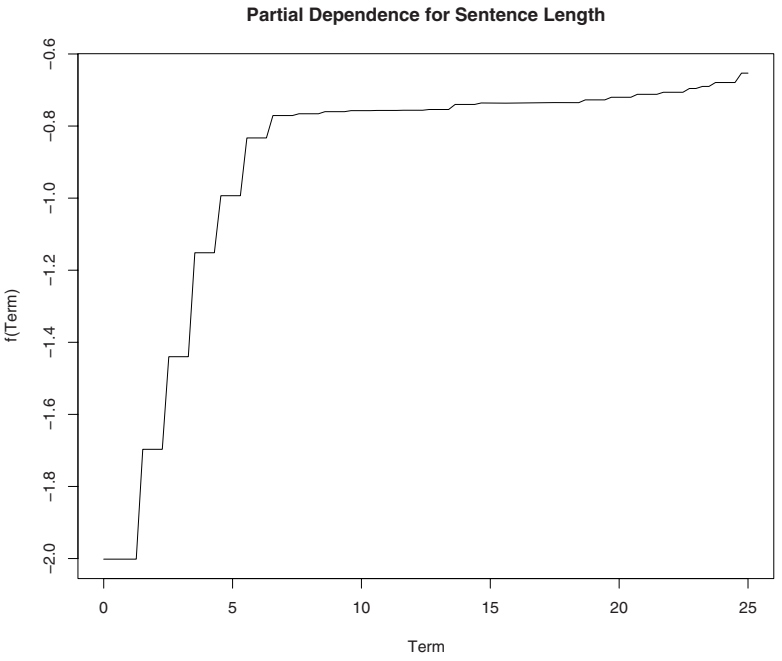


Fig. 6.7. Partial dependence on sentence length in years for binomial boosting for inmate misconduct

other predictors has increased. Thus, sentence length and gang activity are more important for the fit of very serious misconduct compared to the full range of inmate misconduct.

Finally, Figure 6.11 shows the partial dependence plot for sentence length. The response function is now roughly linear over all sentence lengths and does not flatten out for very long sentences. This may help to explain why sentence length has gained in its relative importance. For the reasons discussed earlier, however, it is may not be wise to make much of the response function shown.

In summary, in this application and for the default symmetric costs, boosting does a bit worse than random forests for incidents of general misconduct and much worse than random forests for very serious incidents of misconduct. Boosting, just as conventional regression can stumble badly with highly unbalanced response variables. Moreover, for a binary outcome and stochastic gradient boosting, there is currently no direct way to build asymmetric costs into the fitting process. Consequently, the views of corrections administrators cannot be taken into account. The best one can do is change the classification threshold so that fitted probabilities other than .50 determine the assigned class. This assumes that one has confidence in those probabilities.

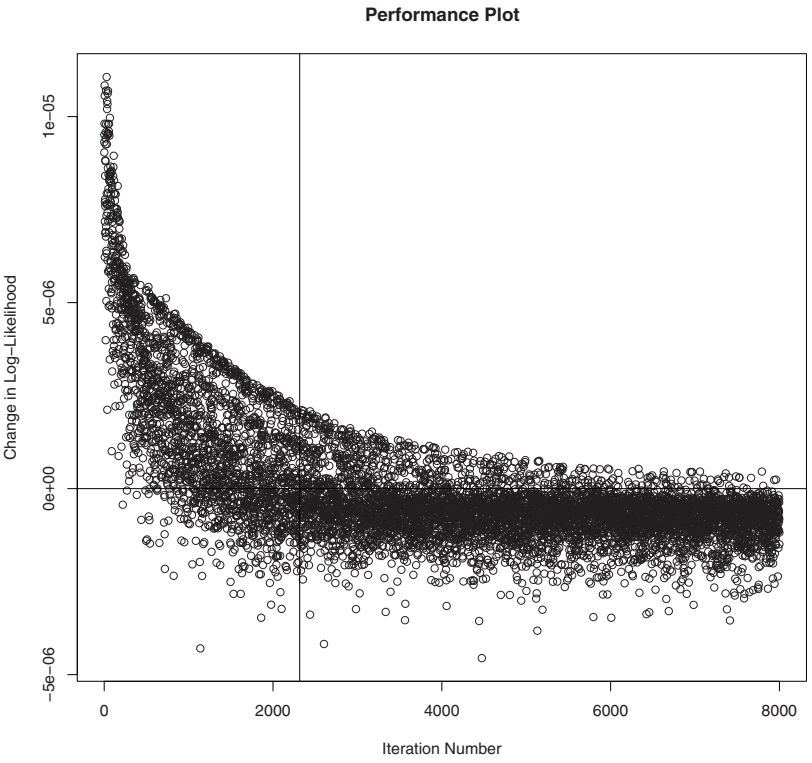


Fig. 6.8. Performance of binomial boosting for very serious inmate misconduct

6.6.3 Homicides and the Impact of Executions

Boosting cannot be expected to improve on conventional Gaussian regression if that regression already fits the data very well. There is nothing to boost. To see how this plays out when the conventional regression already performs with near perfection, and to raise some new issues, we return to the homicides data.

Once again, there are for all 50 states over a 21-year period, the number of homicides per year. As predictors we use the number of executions lagged by one year and then state and year as factors. The key question is whether once one controls for the average number of homicides in a state over the 21 years and the average number of homicides by year over each of the states, the number of executions is related to the number of homicides. For purposes of this illustration, we assume that the number of homicides is conditionally Poisson. Using the generalized additive model we are able to account for well over 95% of the deviance.

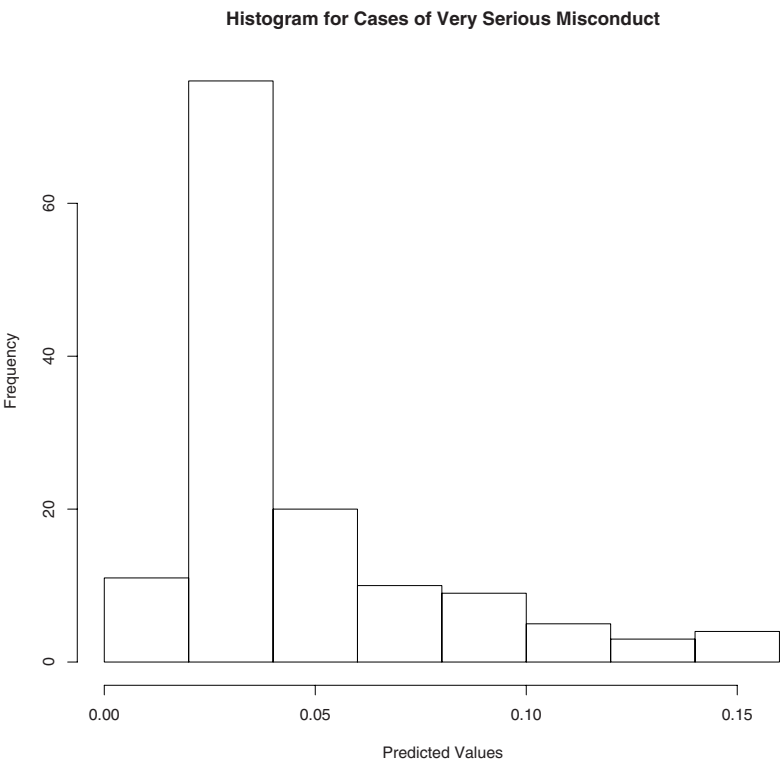


Fig. 6.9. Predicted probability of very serious misconduct.

Figure 6.12 shows how gradient boosting performs in this application. As usual, the changes in the log-likelihood are large early and after about 2000 iterations, the gains are small. Still, the optimal number of iterations is a little less than 10,000.

Figure 6.13 shows that virtually all of the fitting story belongs to the state categorical variable. Its relative contribution is 99.6 out of 100. In contrast, the relative contribution of the number of executions is .008 out of 100.

In Figure 6.14 is plotted the partial dependence of the number of homicides on the number of executions. For less than five executions in a given state in a given year, the relationship is flat. For five or more executions in a given state in a given year, the relationship is negative. But as pointed out earlier, only about 1% of the states have five or more executions in a given year. The apparent evidence for a deterrent effect can only be found where there are almost no data. There is no evidence for deterrence for most states in most years, and too little data to tell when the number of executions is five or more.

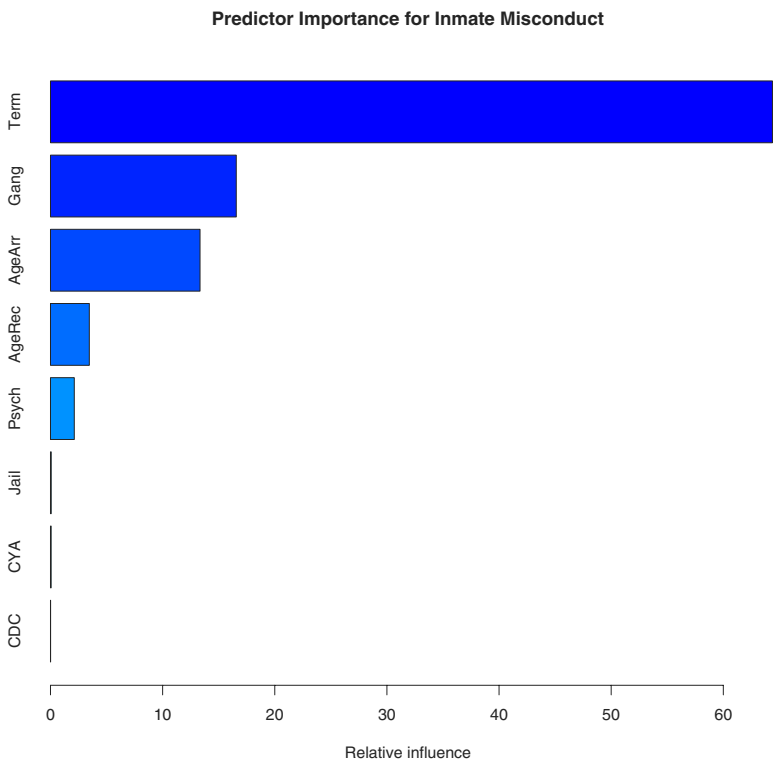


Fig. 6.10. Predictor importance for binomial boosting for very serious inmate misconduct.

A confusion table makes no sense for count data. But we have access to most of the usual regression diagnostics. To begin, the predictors as a group account for about 98% of the deviance. So, the fit is excellent. In addition, Figure 6.15 shows in clockwise order beginning at the upper left 1) the actual number of homicides plotted against the predicted number of homicides, 2) a normal–normal plot of the residuals, 3) a plot of the transformed residuals against the predicted number of homicides, and 4) a plot of Cook's distance by observation number.

From these plots we learn that there are three clumps of fitted values. The two clumps on the right side of the first plot suggest that two smaller subsets of years and/or states may differ from the rest. There is lots of daylight between the clumps. From the first and third plot, we learn that roughly consistent with the Poisson model, the conditional variance of the residuals increases with the conditional mean. However, there is also ample evidence of overdispersion. The second graph indicates that, as one would expect, the

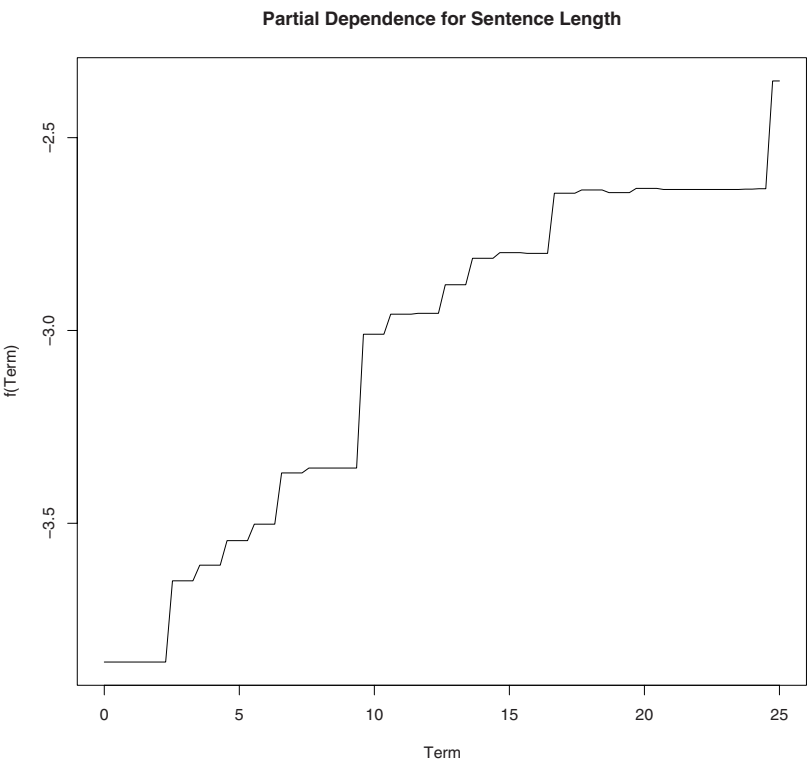


Fig. 6.11. Partial dependence on sentence length in years for binomial boosting for very serious inmate misconduct.

residuals are far from normal, especially at the tails. In fact, the residuals are strongly skewed to the right. This is to be expected given an outcome assumed to be conditionally Poisson. However, the skewing may be linked to a few influential variables. The fourth graph reveals that there are several large influential observations that may well be affecting the fit in significant ways. All of these diagnostics suggest problems with the model, even though most of the variation is accounted for by the predictors.

Figure 6.16 reproduces the partial dependence plot but from data reanalyzed with observations having five executions or more removed. A little more than 1% of the data are lost. Even with only five values for the predictor, the plot is instructive; the plot is a flat straight line. If the response function is even a little more complex, the plot would not have been helpful, even if the computer agreed to construct the plot. There are too few values for the predictor.

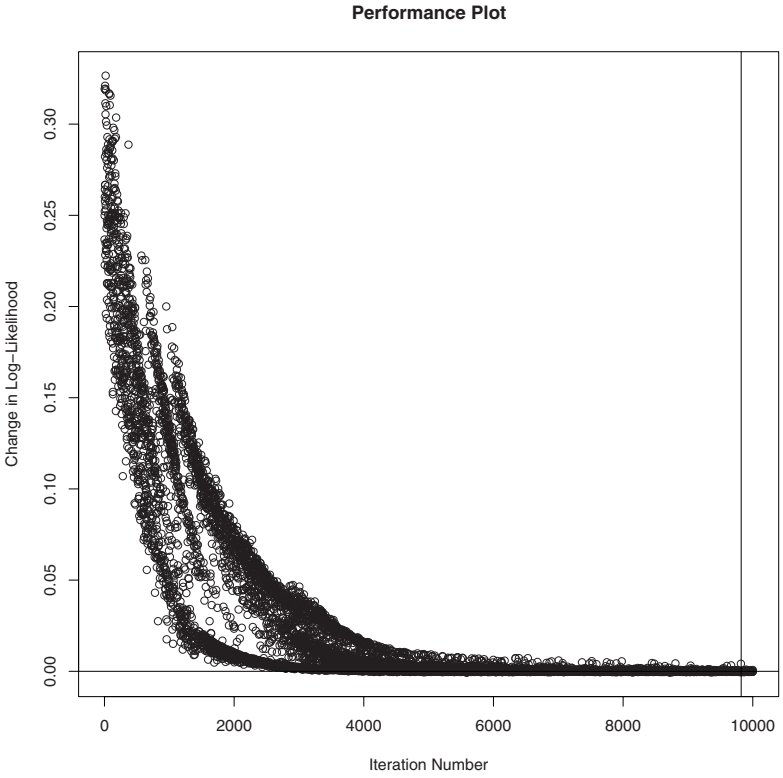


Fig. 6.12. Performance of Poisson boosting for the number of homicides

There is no evidence whatsoever of any deterrence. So, the few influential observations really did affect the boosting results. This is an important lesson. Just as in conventional regression, outliers can make a very big difference. Not surprisingly, the four diagnostic plots (not shown) now look a lot better.

One can in this case obtain virtually the same story using the generalized additive model, including the story about the influential observations. Boosted Poisson regression fits the data slightly better, but not enough to matter; both models fit the data nearly perfectly. And the subject matter conclusions are the same; the gains from boosting compared to parametric regression are slight. The intent in boosting is to take weak predictors and make them strong. There is not much point in boosting predictors that are already very strong.

6.6.4 Imputing the Number of Homeless

Consider again the problem of imputing the number of homeless individuals in Los Angeles County census tracts (Berk et al., 2008). Recall that when

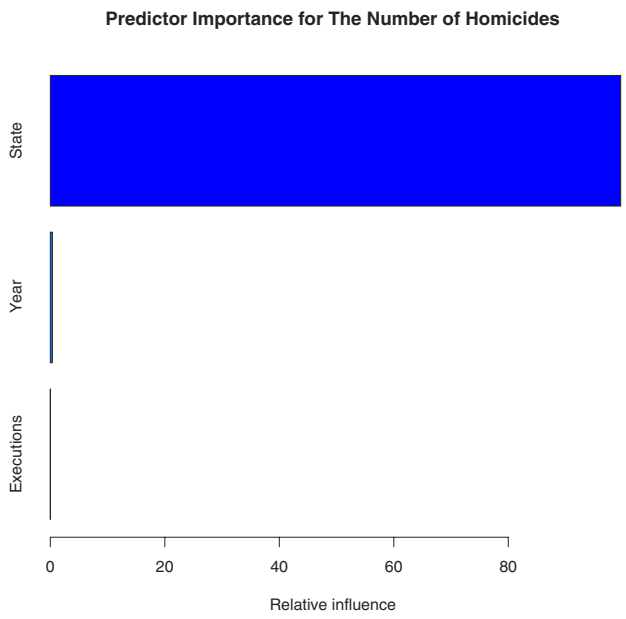


Fig. 6.13. Predictor importance for Poisson boosting for the number of homicides.

random forests was discussed, quantile random forests was applied in order to respond to a few especially high counts. At that time, it was noted that quantile random forests takes the composition of the forest as given, and only adjusts for the summary statistics extracted. A key problem with that approach was that variable importance measures and partial dependence plots were unaltered because they rely on the random forest.

Kriegler (2007) has developed a procedure to weight the loss function in stochastic gradient boosting so that asymmetric costs can be taken into account, not just at the end when summary statistics are constructed, but as the boosting procedure proceeds. Consequently, measures of predictor importance and partial dependence plots are altered accordingly. To date, cost-weighting has been applied to linear (Laplace) loss, Gaussian loss, and Poisson loss, and linear loss has been implemented in gbm().

Figure 6.17 shows for the homeless data, observed street counts plotted again predicted street counts for the weighted linear loss function. Smoothers are overlaid. For the 1/11th quantile, corresponding to weighting overestimates as ten times more costly than underestimates, the fitted values change little, and counts larger than about ten are captured poorly. For the 1/2 quantile, corresponding to equal costs, the overall fit is quite good, but observed counts larger than 50 are not captured well. For the 10/11th quantile, cor-

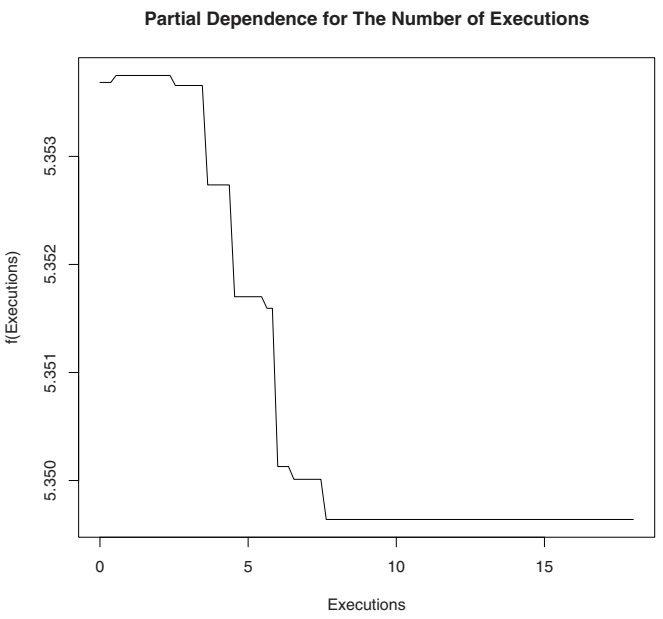


Fig. 6.14. Partial dependence on Executions for Poisson Boosting for the Number of Homicides

responding to costing underestimates as ten times more than overestimates, the overall fit is disappointing, but the larger observed counts are much more effectively fitted. However, still larger relative costs for underestimates would be needed if the very highest observed counts were to be fitted well.

In this instance, the partial dependence plots for key predictors do not change shape materially across different cost ratios. One would tell pretty much the same story about how the predictors are related to the response for a wide range of cost ratios. However, there are some rearrangements of variable importance suggesting that several predictors vary in their forecasting skill depending on which quantile in the response distribution is the target. The details need not concern us here.

As before, there is no statistical answer to the question of which set of fitted values should be preferred. That decision depends on which relative costs are appropriate for the decisions to be made. But it is likely that for counts of the homeless, homeless advocates would see underestimates as far more costly than overestimates. It is less clear what position public officials would take. Larger counts might lead to criticisms to their policies but might also help generate increased funding.

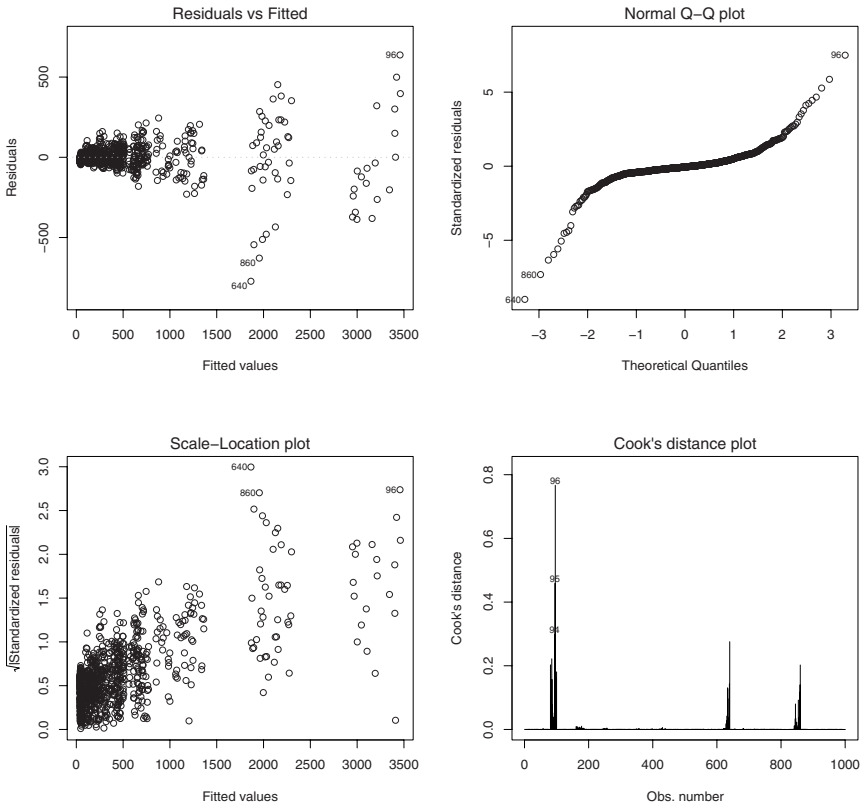


Fig. 6.15. Fit diagnostics for Poisson boosting for the number of homicides.

6.6.5 Estimating Conditional Probabilities

As a final illustration, consider a random sample of American female adults and whether they are in the labor force. The data come from the "Mroz" data in the *car* library in R. The response variable is binary. For this example, the predictors are age, the log of expected wage, household income, and whether there is a child under six in the household.

Two different fit statistics were used to determine when to stop iterating: one based on the OOB data and one based on the data used to build the model. The former indicated that 4000 iterations would be about right. The latter indicated that 10,000 iterations would be about right.

Both stopping rules led to 72% of the observations being correctly classified; classification accuracy was virtually identical. However, the fitted proportions that might be used as estimates of conditional probabilities differed

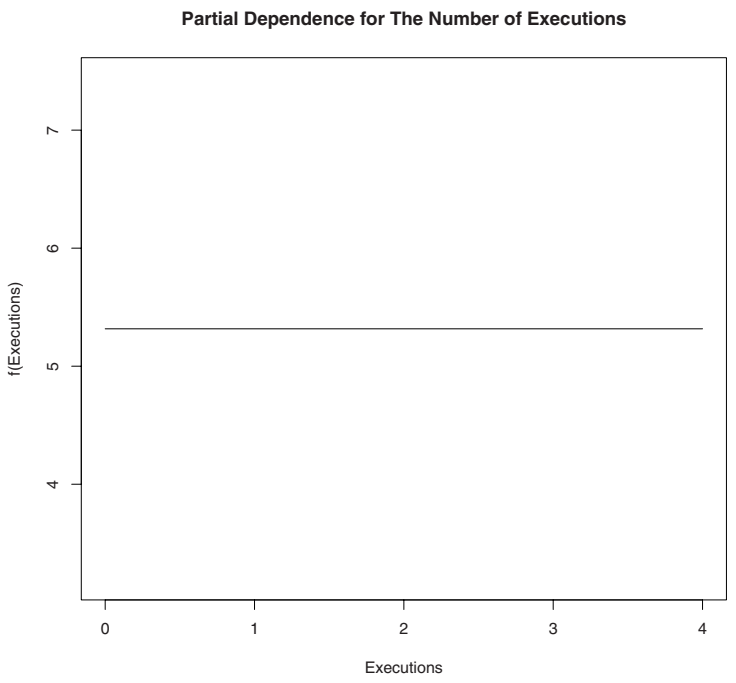


Fig. 6.16. Partial dependence on executions for Poisson boosting for the number of homicides—outliers excluded.

substantially. Figure 6.18 is a scatterplot of the two sets of fitted proportions with a 1-to-1 line overlaid. Because of the stochastic content in the algorithm, the two sets of fitted proportions cannot be exactly the same, but they should cluster tightly around the 1-to-1 lines. The plot shows that fitted proportions are significantly more spread out when there are 10,000 iterations rather than 4000 iterations. On the average the two sets of values differ most at the tails, especially the lower tail.

There is some craft lore arguing that determining the number of iterations using the OOB data can lead to underfitting, which implies that the fitted proportions based on 10,000 iterations should be preferred. Even if this is true, the sensitivity of the fitted proportions to at least one tuning parameter when classification skill is nearly the same, is a bit unsettling. And although craft lore can be very helpful, it comes with no guarantees. If there is a keen interest in the fitted proportions, perhaps as estimates of conditional probabilities, it may be important to consider using Jous-Boost, described briefly earlier (Mease et al., 2007). .

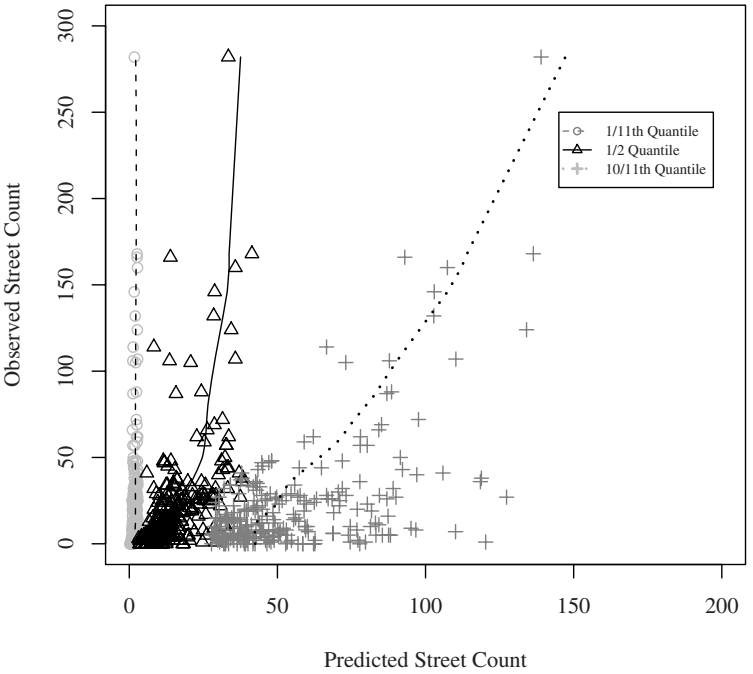


Fig. 6.17. Observed and fitted values for different quantiles.

6.7 Software Considerations

As noted earlier, boosting is in a great state of flux, and nowhere is this more evident than in the software available. The boosting analyses reported in this chapter were done with the procedure gbm() in R. It performs very well using gradient boosting, allowing for a wide variety of loss functions. It also has a number of useful tuning parameters and several helpful forms of graphical output. The name "gbm" stands for Generalized Boosted Models. The package is written by Greg Ridgeway, who also is the maintainer (gregr@rand.org).

There are several other boosting procedures in R. The procedure mboost(), for example, does gradient boosting for the generalized linear model and the generalized additive model. GAMBoost() boosts the generalized additive model using likelihood based approaches. The procedures boost() and ada() implement Adaboost, Logitboost, and other classification procedures. To date, an important advantage that gbm() has over the alternatives in R

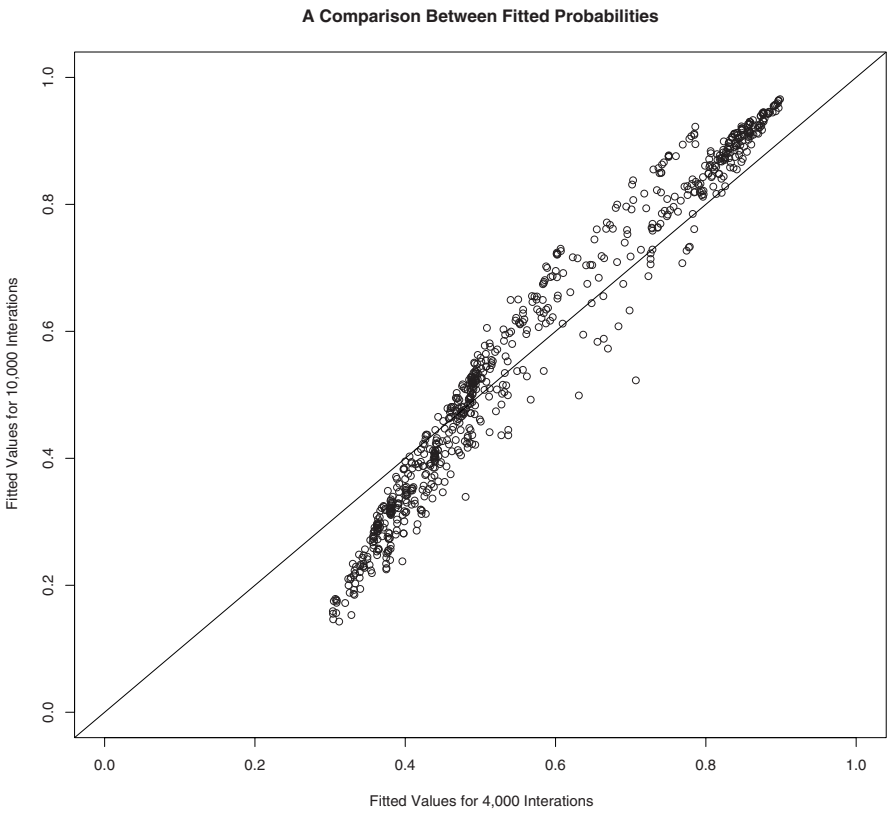


Fig. 6.18. Fitted proportions for two different stopping rules.

is a better range of outputs. The documentation that goes with gbm() is also especially good.

The only visible private sector provider currently is, once again, Salford Systems. Salford Systems offers a program called Multiple Additive Trees (MART), which is essentially gradient boosting. As with all Salford System products, the user interface is very friendly and the documentation excellent. But the price is substantial and the advertising hype a bit offputting.

6.8 Summary and Conclusions

Boosting is a very slick approach to statistical learning. The underlying concepts are interesting and their use to date creative. Boosting has also stim-

ulated very productive interactions among researchers in statistics, applied mathematics, and computer science. Perhaps most important, boosting has been shown to be very effective for certain kinds of data analysis.

However, there are important limitations to keep in mind. First, boosting is designed to improve the performance of weak predictors. Trying to boost predictors that are already strong is not likely to be productive. A set of strong predictors can lead to an effective fit within a conventional regression model. Then, the residuals are essentially noise. Then, there is no more information to extract.

Unfortunately, there is no convincing way from the data alone to know if the residuals really lack any systematic information. But if the list of variables represents all the predictors known to be important, if these predictors are well measured, and if the partial response plots are consistent with widely accepted and detailed theory, the chances are good that boosting will not help much.

Second, if the goal is to fit conditional probabilities, boosting can be a risky way to go. One useful alternative was discussed, but it has yet to be extensively field-tested with real data. It follows that calculations using the fitted probabilities can be highly suspect.

Third, boosting is not alchemy. Boosting can improve the performance of many weak fitting procedures, but the improvements may fall far short of the performance needed. Boosting cannot overcome variables that are measured poorly or important predictors that have been overlooked. The moral is that (even) boosting cannot overcome a seriously flawed measurement and badly executed data collection. The same applies to all of the statistical learning procedures discussed in this book.

Finally, when compared to other statistical learning procedures, especially random forests, boosting will often allow for a wider range of applications, and for the same kinds of applications, perform competitively. In addition, its clear links to common and well-understood statistical procedures can help make boosting understandable. However, boosting's usual reliance on symmetrical loss functions is a major difficulty, especially for research results that will be used to inform practical decisions and broader public policy.

Exercises

Problem Set 1

Generate the following data. The systematic component of the response variable is quadratic. If 1000 observations are too large for your computer to easily handle, work with a sample of 500 observations.

```
x1=rnorm(1000)
x12=x1^2
```

```
ysys=1+(-5*x12)
y=ysys+(5*rnorm(1000))
dta=data.frame(y,x1,x12)
```

1. Plot the the systematic part of y against the predictor $x1$. This represents the $f(x)$ you are trying to recover. Plot y against $x1$. This represents the data to be analyzed. Why do they look different?

2. Apply gbm() to the data. There are a lot of tuning parameters and parameters that need to be set for later output so, here is some code to get you started.

   ```
   out<-gbm(y~x1,distribution="gaussian",n.trees=10000,
        data=dta1,cv.folds=5)
   gbm.perf(out,method="cv")
   ```

   Construct the partial dependence plot using

   ```
   plot(out,n.trees=???)
   ```

   where the ??? is the number of trees, which is the same as the number of iterations. Make five plots, one each of the following number of iterations: 100, 500, 1000, 5000, and the number recommended by the cross-validiation method in the second step above. Study the sequence of plots and compare them to the plot of the true $f(X)$. What happens to the plots as the number of iterations approaches the recommended number? Why does this happen?

Problem Set 2

From the *car* library load the data "Leinhardt." Analyze the data using gbm(). The response variable is infant mortality.

1. Plot the performance of gbm(). Interpret the two lines that are plotted and explain what their divergence implies.

2. What is the recommended number of iterations?

3. Construct a graph of the importance of the predictors. Which variables seem to affect the fit substantially and which do not?

4. Construct the marginal partial dependence plot for each predictor Interpret each plot.

5. Construct all of the two-variable plots (see examples in help(gbm)). Interpret each plot.

6. Construct the three-variable plot (see examples in help(gbm)) Interpret the plot.

7. Consider the quality of the fit. How large is the improvement compared to when no predictors are used?

8. Write a paragraph or so, on what the analysis of these data has revealed about correlates of infant mortality at a national level.

9. Now repeat the analysis using random forests. How do the results compare to the results from stochastic gradient boosting? Would you have arrived at substantially different conclusions depending on whether you used random forests or stochastic gradient boosting?

Problem Set 3

From the *MASS* library, analyze the dataset called Pima.tr. The outcome is binary: diabetes or not (coded as "Yes" and "No"). Assume that the costs of failing to identify someone who has diabetes are three times higher than the costs of falsely identifying someone who has diabetes. The predictors are all of the other variables in the dataset.

The goal is to analyze these data using several different procedures and then make comparisons across the results. The statistical procedures to compare are logistic regression, the generalized additive model, random forests, and stochastic gradient boosting. You will need to make a number of decisions so that the methods are as comparable as possible (e.g., what loss function to use for stochastic gradient boosting). But also feel free to try several different versions of each procedure (e.g., "Adaboost" v. "bernoulli" for stochastic gradient boosting).

1. Construct confusion tables for each model. Be alert for whether the fitted values are for "resubstituted" data. Do some procedures fit the data better than others? Why or why not?

2. Cross-tabulate the fitted values for each model against the fitted values for each other model. How do the sets of fitted values compare?

3. Compare the "importance" assigned to each predictor. This is tricky. For example, how can sensible comparisons be made between the output of a logistic regression and the output of random forests?

4. Compare partial response functions. This too is tricky. For example, what can you do with logistic regression?

5. If you had to make a choice to use one of these procedures, which would you select? Why?

7

Support Vector Machines

Support vector machines (SVM) will seem somewhat far afield from the statistical learning procedures discussed to this point. SVM was developed as a classifier, largely in computer science, with its own set of research questions, conceptual frameworks, technical language, and culture. In addition, a substantial amount of the initial interest in support vector machines stemmed from the important theoretical work surrounding it (Vapnick, 1996). The early applications were not especially compelling.

Over the past few years, the applications to which support vector machines has been applied have broadened (Christianini and Shawe-Taylor, 2000; Moguerza and Munõz, 2006), and the available software has responded (Joachims, 1998; Hsu et al., 2007; Chen et al., 2004). Support vector machines now can have more the look and feel of regression. Formal links to statistical learning in statistics have been made so that there are increasingly shared concepts and language across interested disciplines (Hastie et al., 2001: Sections 12.1–12.3; Bishop, 2006: Chapters 6–7). And although there are still some very important components of a legitimate regression analysis that are a bit beyond SVM's reach, these drawbacks will probably be remedied reasonably soon. It is useful therefore to spend some time providing a brief summary of how support vector machines works. It introduces a number of intriguing ideas and has considerable promise for regressionlike applications.

7.1 A Simple Didactic Illustration

Consider a very simple classification problem. A guidance counsellor in a small high school is trying to determine which students are at risk for dropping out. There are data on students from the past several years from which one can determine which students dropped out. There is also a promising set of covariates. In looking at the data, the guidance counsellor notices that all students who were reading at second-grade level dropped out of school. At the other extreme, none of the students who were reading at a twelfth-grade level

R.A. Berk, *Statistical Learning from a Regression Perspective*,
DOI: 10.1007/978-0-387-77501-2_7,

ever dropped out. For both of these sets of students, the high-risk students and the low-risk students, the classification job is done because this eyeball classifier works perfectly.

What about the rest of the students? The eyeball classifier stumbles because it is not apparent how to subset the students any further so that the two classes of students are homogeneous. For example, among those students reading at tenth-grade level, seven out of every ten graduate. It might make sense, therefore, to employ a more powerful classifier for these students. And the quality of the classifier overall will depend on how accurately the students between the two extremes can be classified. In other words, variation in the loss function will depend only on how accurately the middle group of students is assigned to the binary outcome.

Focusing only on the middle group of students, a useful classifier might try to put the students into one of the two classes so that a pair of objectives are achieved. First, there should be a small number of misclassifications: students incorrectly labeled as dropouts and students incorrectly labeled as graduating high school. There is nothing new in this.

Second, subject to this small number of misclassifications, the students in the two classes should be as different as possible in their reading ability. If there is a continuum of reading ability monotonically related to dropping out of school, greater separation between the two groups can imply more stable classifications. For example, if the two groups can be divided so that the highest reading level for the dropout group is at least one grade level below the lowest reading level for the higher reading group, a relatively clear distinction has been made. In contrast, if that gap is only 5% of a grade level, the distinction is not nearly so clear. One grade level implies approximately nine months of class time, whereas 5% of a grade level may imply only two weeks of class time. The larger distinction is desirable because it implies that the classifications are more stable under random perturbations of the data. Generalization error will be smaller.

Operationally, suppose the guidance counsellor is prepared to live with 20 misclassifications. Given 20 misclassifications, what reading level should be chosen to separate the two classes? A fourth-grade level? A sixth-grade level? An eighth-grade level? It would make sense to choose a threshold so that on either side, the difference in reading ability between the two groups was as large as possible.

This method of partitioning the data has some important similarities to nearest neighbor methods. The distinction between the two groups of students only depends on where students are located on the measure of reading ability. Students who have similar reading abilities will be classified in the same manner. It is the distance between students that matters, not their raw score. Note that there is no concern with how the data were generated and no substantive interest in how predictors might be related to a response. The goal is solely to construct accurate and stable classifications by classifying similar students similarly.

An accurate and stable classification procedure could give the guidance counsellor a helpful forecasting device. In the future, students similar to those who had been classified as dropouts because of their inability to read well would be seen as having a great risk of dropping out. Interventions of various sorts might follow.

This very simple example illustrates several fundamental features of SVM. It is a classifier that partitions the data, but unlike classification trees, does not do so in a stagewise fashion. In addition, only observations near the classification boundary figure directly in the fitting process. Then, the partitioning is accomplished so that the distance in the predictor space between the two groups is as large as possible; separation is maximized conditional on a predetermined and tolerable number of classification errors. Maximizing the separation serves much the same purpose as large margins in bagging and random forests. Finally, the key information extracted from the predictors is, in effect, a matrix of distances. Observations sufficiently near one another in predictor space will be treated alike.

No simple example can be made to map exactly onto support vector machines. Moving back and forth between the guidance counsellor illustration and the diagrams and equations that follow will reveal an imperfect match. But with some central issues now raised, the new material may be somewhat more accessible.

7.2 Support Vector Machines in Pictures

With the simple example behind us, we can turn to a bit more formal exposition of SVM. Despite its roots in computer science and its focus on classification per se, at its core support vector machines can be treated in a manner introduced in Chapter 2. Hastie et al. (2001: 380–381) point out that one can use the familiar formulation of fitting a set of observed values of a response variable subject to a complexity penalty. But if we skip to that punch line, the underlying intuitions will likely be lost.

7.2.1 Support Vector Classifiers

Suppose there is a binary response variable coded, as is often done in boosting, as"1" and "–1." There is a fitting function $f(x)$, where x can be one or more predictors. If the $f(x)$ returns a positive number, the label "1" is assigned to the observation. If the $f(x)$ returns a negative number, the label "–1" is assigned to the observation. A fitting function can be written as

$$f(x) = \beta_0 + h(x)^T \beta, \tag{7.1}$$

where, as before, $h(x)^T$ are basis functions of x.

In the SVM literature, the response variable is often called the "target variable," and the intercept in Equation 7.1 is often called the "bias." Nevertheless, the basic idea is the same as before: there is a series of basis functions $h(x)$, where x can represent more than one predictor. The basis functions are additively combined, with the vector βs as the weights. Then the goal, as usual, is to minimize the loss without making the fitting function unnecessarily complex.

But beneath the surface, the fitting exercise is quite novel. A key feature is that when undertaking a classification task, some observations are treated very differently from others. In much the same spirit as boosting, the observations that are more difficult to classify receive more attention. But unlike boosting, a qualitative distinction is first made between the observations that are difficult to classify and observations that are not. Then, the problematic observations determine the criterion by which classes are to be assigned. This, in turn, implies the use of an unusual loss function. Thinking back to the guidance counsellor illustration, observations that fall in regions where there is a mix of students who drop out and students who graduate are the observations that are difficult to classify and consequently, the observations that determine the precise location of the threshold to be imposed separating high-risk from low-risk students.

Another key feature is that the predictors affect the class assigned by how they locate observations in the space defined by the predictors. What matters is where observations fall with respect to one another. This is a significant difference between the statistical learning procedures in this chapter and those discussed in earlier chapters. Details are provided a little later. But it is a bit like what real estate agents often say: what matters is "location, location, location."

Figure 7.1 shows a partitioning diagram. As before, there are two predictors (x and z) and a binary response y, that can take on values of A or B. B might represent dropping out of school and A might represent graduating. (A could be coded as 1 and B could be coded as –1.) The two predictors might be reading grade level and the number of truancies per semester. In this figure, the As and the Bs are each located in quite different areas of the two-dimensional space defined by the predictors. In fact, there is lots of daylight between the two groupings.

With data of this sort, one can apply a "support vector classifier." The goal is to locate a "decision boundary," here represented by the dark straight line, using information from the predictors so that the partitions are as homogeneous as possible. In spirit, this is lot like CART. The decision boundary is also called a "separating hyperplane." Observations that fall on one side of the decision boundary are assigned to one class, and observations that fall on the other side of the decision boundary are assigned to the other class. In this instance, each observation above and to the right would be correctly classified as an A. Each observation below and to the left would be correctly

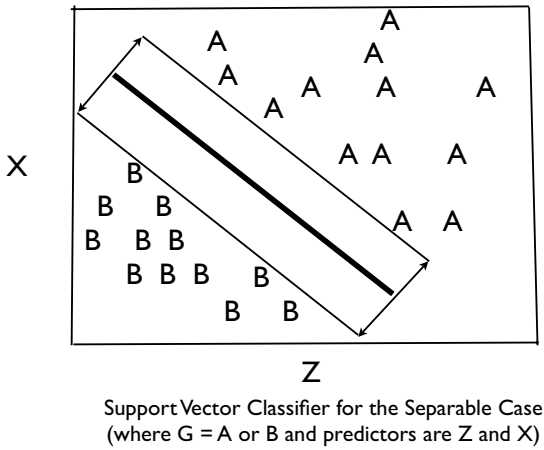


Fig. 7.1. Support vector classifier when there are two separable classes.

classified as a B. Future observations would then be forecasted to an A or a B depending on where in the predictor space they fell.

For Figure 7.1, finding a decision boundary is easy. In fact, it is too easy. There are a limitless number of linear decision boundaries one could draw that would also define two perfectly homogeneous partitions. A way needs to be found to determine the "best" decision boundary among all those that sensibly could be constructed.

For data such as those shown in Figure 7.1, the support vector classifier solves this problem by constructing two parallel lines on either side of, and the same distance from, the decision boundary. The two lines are as far apart as possible without including any observations within the space between them. One can think of the two lines as fences defining a data "buffer zone." Observations can fall right on either fence but not across them, inside the buffer zone. In Figure 7.1, the total width of the buffer zone is shown with the two double-headed arrows.

The distance between the two fences or the distance between the decision boundary and either fence is called the "margin." The different definitions amount to the same thing in practice, and justification for maximizing the margin has a familiar ring. The wider the margin is, the greater the separation between the two classes. Larger margins are desirable because generalization error will usually be smaller. A more definitive and, therefore, more stable distinction is being made between the two classes. Thus, the buffer zone's margin plays much the same role as the margin used in bagging and random

forests, although it is not defined in the same way. Sometimes the two fences are called the "margin boundary."

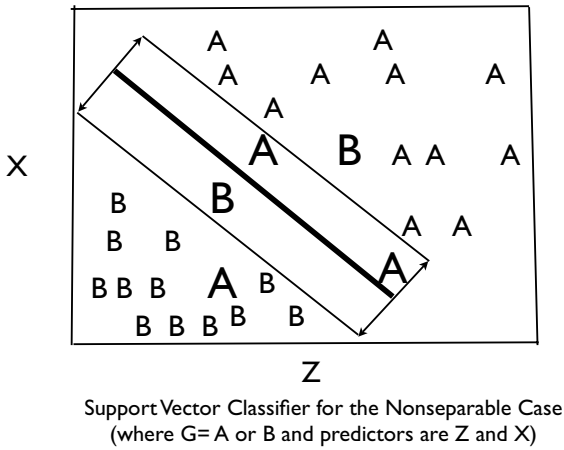


Fig. 7.2. Support vector classifier when there are two classes that are not separable.

The data shown in Figure 7.1 are very cooperative, and such cooperation is in practice rare. Figure 7.2 shows a partitioning diagram that is much like Figure 7.1, but the two sets of values are no longer separable with a straight line. The three larger *A*s and the two larger *B*s violate the margin boundary. They are on the wrong side of their respective buffer zone fences. The large *A*s are too far toward the lower left of the figure, and the large *B*s are too far toward the upper right. Moreover, there is no way to relocate and/or narrow the buffer zone so there is a separating hyperplane able to partition the space into two perfectly homogeneous regions. There is no longer any linear solution to this classification problem.

One possible response is to permit some misclassifications. One could specify some fraction of the observations that could be allowed to fall on the wrong side of the decision boundary. That is, one could try to live with a result that looked a lot like Figure 7.2. The idea would be to maximize the width of the buffer zone subject to some specified number of misclassifications.

But that is not quite enough. Some misclassified observations fall just across the decision boundary and some fall far away. Some correctly classified observations fall on the wrong side of their fence, violating the need for wide separation between the two classes. But these observations can differ on how far on the wrong side of the fence they fall. In response, the distance between

the relevant fence and the location of an observation can be taken into account for both kinds of violations. For example, there are two large *A*s whose distance from their fence is small, and one whose distance is substantial. The first two are correctly classified. The last one is not. There is a large *B* close to its fence and a large *B* much farther away. The former is correctly classified. The latter is not.

The sum of such distances can be viewed as a measure of how permissive one has been when the margin is maximized. If one is more permissive by allowing for a larger sum, it usually possible to construct a larger margin. Again, larger margins are good. More stable classifications can follow. But more permissive solutions imply more bias. There will be a greater number of misclassifications and/or the misclassifications will be more in error. The bias–variance tradeoff reappears. It follows that the sum of the distances can be a tuning parameter when a support vector classifier is applied to data.

Sometimes, several observations will fall right on the margin boundary. These observations, in addition to the observations on the wrong side of the margin boundary are used to determine, in the space defined by the predictors, the precise location of the decision boundary and the buffer zone. Because such observations "support" the location of the decision boundary, they are called "support vectors." No other observations play that role.

The support vectors represent the observations most difficult to correctly classify. These As and Bs inhabit much the same space that the predictors define. They can differ from each other in their response values even though they are near each other and, therefore, alike on their predictor values. Thus, the special attention given to the support vectors is in the same spirit as the extra weight given to misclassified cases in boosting.

Figure 7.3 is almost the same as Figure 7.2, but the buffer zone has been enlarged and reoriented a bit to represent a solution to the classification problem. There are two kinds of support vectors: those on the wrong side of the margin boundary and those on top of the margin boundary. The support vectors are shown in large letters. Classification is determined by the side of the decision boundary on which an observation falls. In Figure 7.3, one *A* is misclassified and one *B* is misclassified.

7.2.2 Support Vector Machines

There is a still better solution to the classification problem when the classes are nonseparable. Suppose one allows for a nonlinear decision boundary. In somewhat the same manner of the smoothers we considered in Chapter 2, the data are used to help determine an appropriate nonlinear function, within a given category of functions, that can separate the two classes of data. The added flexibility makes it much more likely that the decision boundary will classify accurately and with a large margin.

Once again, a basis function approach can be applied. One can enlarge the predictor space in which the observations sit and construct a nonlinear

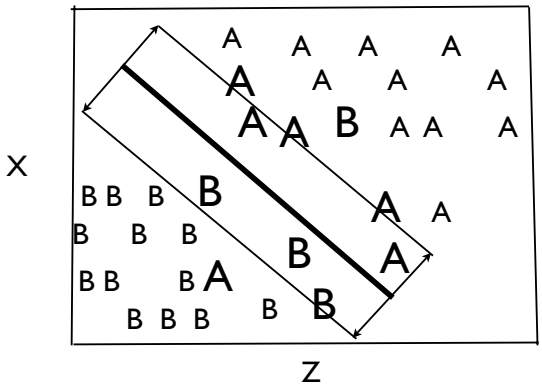


Fig. 7.3. Support vector classifier when there are two classes that are not separable with support vectors shown.

function of the original variables. Support vector machines capitalizes on these ideas by allowing for a very large number of transformations of the predictors, but only for particular classes of transformations. A few of these transformation are considered shortly. We show that with the focus on the location of observations in the predictor space, and especially on their proximity to one another, the transformations are applied not to each predictor by itself, but to the predictor space they define. This is a useful and very clever approach.

There are some of the same tuning issues for support vector machines as there are for support vector classifiers. One tuning parameter is a function of the sum of the distances of the observations on the wrong side of the margin boundary. This sum serves the same purpose as it did for support vector classifiers. One or more other tuning parameters are also common, depending on which predictor transformations are employed. We will see that, somewhat in contrast to random forests and stochastic gradient boosting, variation across a set of reasonable values for the tuning parameters can have a large impact on the results.

In summary, the goal of support vector classifiers is to linearly partition the space defined by the predictors so that two homogeneous regions are defined. The decision boundary for those two regions is determined by making the margin as large as possible. This often leads to a compromise in which some observations are allowed to cross over into, and even beyond, the region inside the margin boundary. Support vector machines takes support vector classifiers

one significant step further by allowing the decision boundary and the margin boundary to be nonlinear. But in both cases, the tradeoff between variance and bias remains.

7.3 Support Vector Machines in Statistical Notation

Support vector classifiers and support vector machines can be considered along with some other procedures as maximum margin classifiers. It is probably fair to say that support vector machines is at this point the most popular. But despite its popularity, or perhaps because of its popularity, there are several somewhat different ways it can be formulated and presented. What follows draws heavily on Hastie and his colleagues (2001) and on Bishop (2006). As before, we start with support vector classifiers to introduce some of the key concepts.

7.3.1 Support Vector Classifiers

There is a set of p predictors and a binary outcome. The goal is to find a linear boundary so that in the space defined by the predictors, the data are partitioned into two regions, both of which are perfectly homogeneous with respect to the response. In addition, the margin is to be as large as possible.

The Separable Case

There are N observations in the training data. Each observation has a value for each of p predictors and a value for the response. The response is coded "1" or "–1." A separating hyperplane of dimension $p - 1$ is defined as

$$f(x) = \beta_0 + x^T\beta = 0. \tag{7.2}$$

Classification is then undertaken by the following rule,

$$G(x) = \text{sign}(\beta_0 + x^T\beta). \tag{7.3}$$

The $f(x)$ in Equation 7.2 can be used to compute the signed distance of any point from the separating hyperplane. Thus, one can determine for any i whether $y_i f(x_i) > 0$ and, therefore, is correctly classified. Because y is coded as 1 and –1, products that are positive represent correctly classified cases. One can also determine if that observation is on the wrong side of its fence and if so, how far. Finally, for observations that are misclassified, one can determine how badly.

The ability to generate signed distances greater than 0 between observations and the separating hyperplane, conditional on the values of β and β_0, implies that one can maximize the margin by how β and β_0 are chosen. Values

of β and β_0 are sought so that the distance from the decision boundary to the closest observations is as large as possible. This implies the use of a novel loss function that is discussed later.

More formally, the goal is to

$$\max_{\beta,\beta_0,||\beta||=1} C, \tag{7.4}$$

subject to

$$y_i(\beta_0 + x_i^T\beta) \geq C, \quad i = 1, \ldots, N. \tag{7.5}$$

The left-hand side of Equation 7.5 is the distance between the decision boundary and an observation. In this formulation, C is the distance from the decision boundary to the margin boundary; $2C$ is the margin. Because C is a distance centered on the decision boundary, Equation 7.5 identifies correctly classified observations on or beyond the margin boundary. Because Equation 7.5 applies to cases $1, \ldots, N$, no cases are inside their fences. Thus, C is sometimes characterized as producing a "hard boundary" that is impermeable. In short, the goal is to find the values of the coefficients that maximize the margin, such that there are no observations inside the fences. This basically characterizes the support vector classifier for the separable case.

It is sometimes useful to work with an equivalent formulation (Hastie et al., 2001: 372):

$$\min_{\beta,\beta_0} ||\beta|| \tag{7.6}$$

subject to

$$y_i(\beta_0 + x_i^T\beta) \geq 1, \quad i = 1, \ldots, N. \tag{7.7}$$

A key change from Equation 7.4 is that the norm constraint on the coefficients has been discarded, so that one can set $C = 1/||\beta||$ (Hastie, 2001: 108–109). Equation 7.7 then can follow. Equation 7.7 requires that the point closest to the decision boundary has a distance 1.0 and that all other observations are farther away (i.e., distance > 1). This particular normalization is arbitrary but does not change the basic problem and can lead to a more direct and easily comprehended solution (Bishop, 2006: 327–328).

Minimizing the norm can be thought of as trying to make the values of the coefficients as small as possible, which forces the margin to be as large as possible. Then, at least one observation must fall on the margin boundary (Bishop, 2006: 328). One can do no better than this and maintain separation.

The Nonseparable Case

To move toward the non separable case, some misclassifications have to be tolerated. Define a set of "slack" variables $\xi = (\xi_1, \xi_2, \ldots, \xi_N)$, $\xi_i \geq 0$, that measure how far incorrectly classified observations are on the wrong side of their fence. We let $\xi_i = 0$ for observations that are correctly classified and on the proper side of their fence or right on top of it; they are not in the buffer

zone. The farther an observation moves across its fence into and even through the buffer zone, the larger is the value of the slack variable. In other words, the value for a slack variable i denotes how inaccurate the classification exercise is for observation i.

The slack variables lead to a revised setting in which the coefficients are chosen so that C is maximized. Specifically,

$$y_i(\beta_0 + x_i^T\beta) \geq C(1 - \xi_i) \tag{7.8}$$

for all $\xi_i \geq 0$, and $\sum_{i=1}^{N} \xi_i \leq K$, with K as some constant.

In Equation 7.8, the "hard boundary" C has been transformed into a "soft" boundary $C(1 - \xi_i)$. The boundary moves from observation to observation depending on the value of ξ_i so that crossing into the buffer zone and even to its other side can be permitted (Bishop, 2006: 331–332). Because slack variables cannot be negative, the constraint becomes smaller as the value of the slack variable increases. It is as if the margin were shrunk proportionally for observations that have a slack variable defined.

For a response variable coded "1" or "–1",

$$\xi_i = |y_i - f(x_i)|. \tag{7.9}$$

Equation 7.9 then implies that

1. $0 < \xi_i \leq 1$ for observations that violate the buffer zone but are correctly classified.
2. $\xi_i = 1$ for observations that are correctly classified and right on top of the decision boundary.
3. $\xi_i > 1$ for misclassified observations, implying that the constraint in Equation 7.8 is negative.

These are the values whose sum cannot exceed K. Because for all misclassified observations $\xi_i > 1$, K can be interpreted as the upper bound on the number of misclassifications. As a result, K can become a useful tuning parameter. With a larger K, there are more support vectors. The decision boundary that follows is more stable; there is less variance from sample to sample. But the price is more misclassifications. There is greater bias. There is no "right" value for K beyond how the classifier performs, and as we show later, choosing the "best" value for K depends heavily on craft lore.

For completeness, we again offer an equivalent formulation much like the one provided earlier (Hastie et al., 2001: 373).

$$\min_{\beta,\beta_0} ||\beta|| \tag{7.10}$$

subject to

$$y_i(\beta_0 + x_i^T\beta) \geq 1 - \xi_i, \quad i = 1, \ldots, N, \tag{7.11}$$

for all $\xi_i \geq 0$, and $\sum_{i=1}^{N} \xi_i \leq K$, with K as some constant. In exposition coming from computer science traditions, Equations 7.10 and 7.11 are considered "canonical." But canonical does not mean trouble free. Bishop (2006:

322) observes that the minimization process can be distorted by outlier values of ξ_i. A very small number of badly misclassified observations can drive the solution.

7.3.2 Support Vector Machines

We now turn from the support vector classifier to the support vector machine. A key difference is the use of nonlinear transformations of the predictors. Thus, beginning with Equation 7.2, x is replaced by $h(x)$. However, the nature of the transformations is unlike any of those considered in earlier chapters.

Suppose one has an $N \times M$ data matrix D of data, which includes a response variable and a set of predictors. N is the number of observations and M is the number of variables. A conventional scatterplot of the data will locate each observation in a space defined by the variables. A cross-product matrix of the data will be $M \times M$ and symmetric with each off-diagonal entry a measure of how one variable is related to another. With proper scaling the cross-product matrix can become a conventional correlation matrix. One can learn which variables are strongly associated with other variables.

One can alternatively proceed with a transpose of D. Now rows are variables and the columns are observations. A different kind of scatterplot might follow in which variables are located in a space defined by observations. Compared to the usual scatterplot, the roles of variables and observations are reversed. A cross-product matrix is now $N \times N$ and symmetric with each off-diagonal entry a measure of how one observation is related to another. Again, with proper scaling the cross-products can be turned into correlations. One can learn which observations are strongly associated with other observations.

Figure 7.4 shows the two different scatterplots. To keep the plots manageable, there are for both just two variables and two observations. The plot on the left shows a conventional scatterplot with variables defining the space. The plot on the right shows an alternative scatterplot with observations defining the space.

This dual representation of data can be applied to any dataset or a subset thereof. In SVM, the predictors are arrayed in observation space so that the relationships between observations can be exploited. The key to all this is a set of inner products. For example, if a is an $N \times 1$ vector, the inner product is $a^T a$, a scalar that is the sum of the squared values of a. If there are two $N \times 1$ vectors a and b, the inner product is $a^T b = ba^T$, which is the sum of their cross-products.

For every pair of cases i and j, one computes $x_i^T x_j$. If there are two predictors, for example, there will be two predictor values for case i and two predictor values for case j. The inner product would be the sum of two cross-products. This sum can be viewed as the distance between case i and case j, or their similarity. And again, with proper scaling the sum of the cross-products can be turned into a correlation.

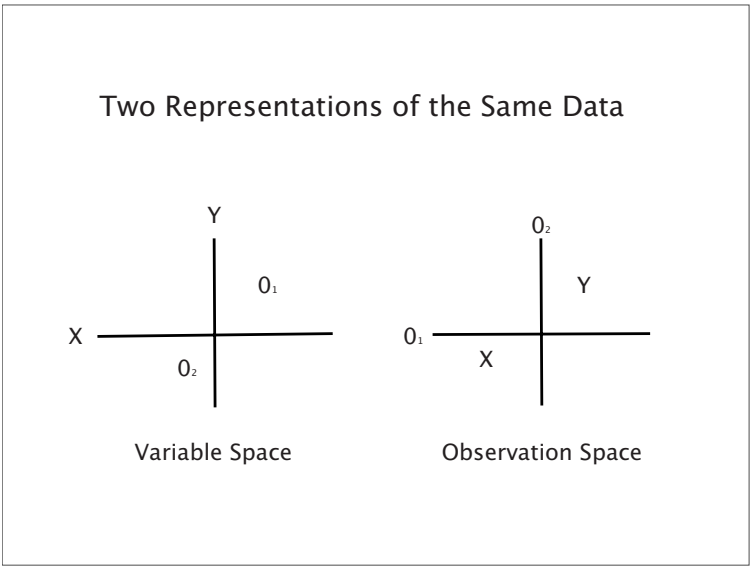


Fig. 7.4. Variable space and observation space.

In order to allow for more flexible fitting, the cross-products can be transformed using a "kernel function" $K(x_i, x_j)$. For example, a relatively simple kernel is the polynomial, which can be represented as

$$K(x_i, x_j) = (1 + x_i^T x_j)^d, \tag{7.12}$$

where d is the order of the polynomial.

The term "kernel" used in SVM is different from the term "kernel" used in smoothing. In smoothing, a kernel refers to a region of the predictor space in which some calculations are to be undertaken. In SVM, a kernel represents a transformation applied to the transposed cross-product matrix of predictors.

Consider a simple numerical example. For observation i, the value for the first variable is 2, and the value for the second variable is 3. For observation j the value for the first variable is 1, and the value for the second variable is 3. So the inner product is $(2 \times 1) + (3 \times 3)$. With a second-order polynomial, the result is

$$K(x_i, x_j) = [1 + (2 \times 1) + (3 \times 3)]^2 = [1 + 2 + 9]^2 = 144. \tag{7.13}$$

For observation i and observation j, 144 is the value of the transformed predictors. The goal of the transformation is to alter how the two observations are related to each other. More generally, it is as if observations are moved around so that separation is more effectively achieved.

There are many different kinds of kernels currently in use or recently proposed (Bishop, 2006: 295–297), but at this point, there is no universal set nor

much consensus about which work best for which kinds of data. In each case, the result must be an $N \times N$ symmetric, positive definite matrix K. This is the input into the optimization procedure whose solution has the following form (Hastie et al., 2001: 378; Bishop, 2006: 329),

$$\hat{f}(x) = \hat{\beta}_0 + \sum_{i=1}^{N} \hat{\alpha}_i y_i K(x, x_i), \tag{7.14}$$

where $\hat{\alpha}_i$ is a positive weight given to each observation estimated from the data (Hastie et al., 2001: Section 12.3.3).

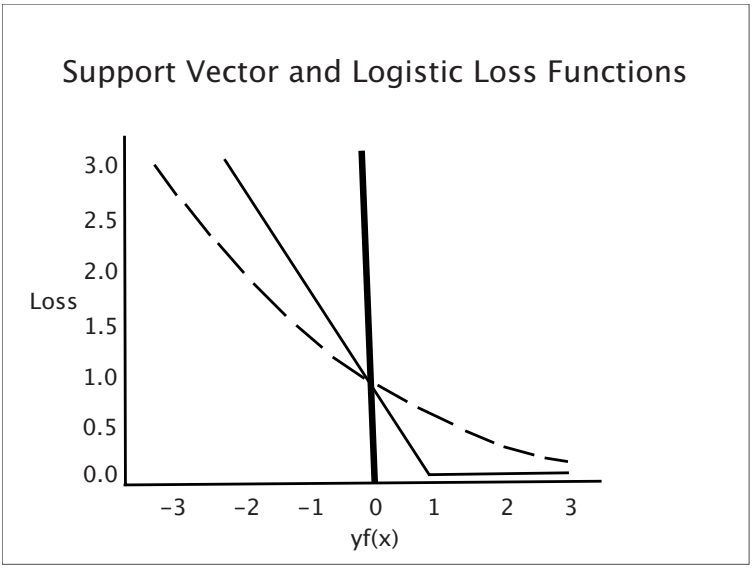


Fig. 7.5. A comparison of classification loss functions.

Interestingly, the minimization exercise for support vector classifiers can be written in regularized sum of squares form (Hastie et al., 2001: 380; Bishop, 2006: 293):

$$\min_{\beta_0, \beta} \sum_{i=1}^{N} [1 - y_i f(x_i)]_+ + \lambda ||\beta||^2, \tag{7.15}$$

where the $+$ indicates that only the positive values are used, and λ plays the same role as K. Note that these residuals are treated in a linear fashion, implying less sensitivity to extreme values than, for instance, a quadratic function.

The use of a regularized sum of squares naturally raises questions about characteristics of the loss function for support vector classifiers and support

vector machines (Hastie et al., 2001: 380–381; Bishop, 2006: 337–338). Figure 7.5 shows the "hinge" loss function (the thin solid line) used by both and for purposes of comparison, the logistic loss function (the thin broken line). The logistic loss function has been rescaled to facilitate a comparison.

Some speak of the support vector loss function as a "hockey stick." The thick vertical line represents the separating hyperplane. Values of $yf(x)$ to the left denote observations that are misclassified. Values of $yf(x)$ to the right denote observations that are properly classified. The product of y and $f(x)$ will be ≥ 1 if a correctly classified observation is on the proper side of its fence.

Consider the region defined by $yf(x) < 1$. Moving from left to right, both loss functions decline. At $yf(x) = 0$, an observation is correctly classified, but in the buffer zone. The loss is equal to 1.0. Moving toward $yf(x) = 1$, both loss functions continue to decline. The support vector loss function is equal to 0 at $yf(x) = 1$. The logistic loss function is greater than 0. For $yf(x) > 1$, the support vector loss function has a loss of 0. The logistic loss function continues to decline, but still has a loss greater than 0.

One could argue that the two loss functions are not dramatically different. Both can be seen as an approximation of misclassification error. The misclassification loss function would be a step function equal to 1.0 to the left $yf(x) = 0$ and equal to 0.0 at or to the right of $yf(x) = 0$. It is not clear in general when the support vector loss function or the logistic loss function should be preferred although it would seem that the support vector loss function would be somewhat less affected by outliers.

7.3.3 SVM for Regression

Support vector machines can be altered to apply to quantitative response variables. One common approach is in the fitting process to only use residuals smaller in absolute value than some constant (called "e-insensitive" regression). These are somewhat analogous to observations on the incorrect side of the fence. For the other residuals, one has a linear loss function. The result is a robustified kind of regression. The relative advantage in practice of support vector machine regression compared to any of several forms of robust regression is not clear.

7.4 A Classification Example

Support vector machines is sufficiently different from the procedures discussed earlier that a graduated illustration may be helpful. We begin with a relatively simple analysis using the Mroz dataset from the *car* library.

The data come from a sample survey of 783 husband–wife households. The response variable will be whether the wife is employed. For now, two predictors are selected: household income exclusive of the wife's earnings in

tens of thousands of dollars, and the log of the wife's expected weekly wage. Expected wage is the earnings anticipated if the wife finds a job. The data are divided randomly into a training dataset of 400 observations and a test data set of 383 observations. About 60% of the wives are employed, so the response variable is reasonably well balanced, and there is nothing else in the data to make an analysis of labor force participation especially problematic.

7.4.1 SVM Analysis with a Linear Kernel

	Predict Unemployed	Predict Employed	Model Error
Unemployed	38	136	.78
Employed	38	188	.17
Use Error	.50	.41	Overall Error = .44

Table 7.1. SVM confusion table for forecasting employment: linear kernel, $cost = 1$, $\gamma = NULL$, 339 support vectors.

Table 7.1 shows the SVM confusion table with a linear kernel, and tuning parameters set at their default values. The linear kernel is used for simplicity. For pairs of observations, the transformation is of the form $(x_i^T x_j)$. The tuning parameter that svm() calls γ is not relevant to the linear kernel and is set to "NULL". The tuning parameter that svm() calls "cost" plays the same role as λ in Equation 7.15 or K in Equation 7.8. It determines how much weight is given to the penalty function or alternatively, how tolerant one is prepared to be about observations that are not fit well. It is through this tuning parameter that one trades bias against variance. The default sets cost to 1.0. Resubstituted data are used to build the table, but with the linear kernel, two predictors, and a sample of 400, overfitting is not likely to be problematic.

The overall proportion of cases misclassified is .44. The SVM model misclassifies unemployed wives 78% of the time and employed wives 17% of the time. Clearly, it is more difficult in this case to accurately classify employed women.

Figure 7.6 shows a classification plot with the linear boundary implied by the linear kernel. The darker area to the lower right represents that part of the predictor space in which all cases are classified as unemployed. The lighter area to the upper left represents that part of the predictor space in which all cases are classified as employed. The "x" symbols denote support vectors. The "o" symbols denote vectors that are not used to locate the decision boundary. The plotting procedure in svm() also has the ability to show which observations represent employed wives and which observations represent unemployed wives, but one has to be able to plot in color. R does, but there are no color reproductions in this book.

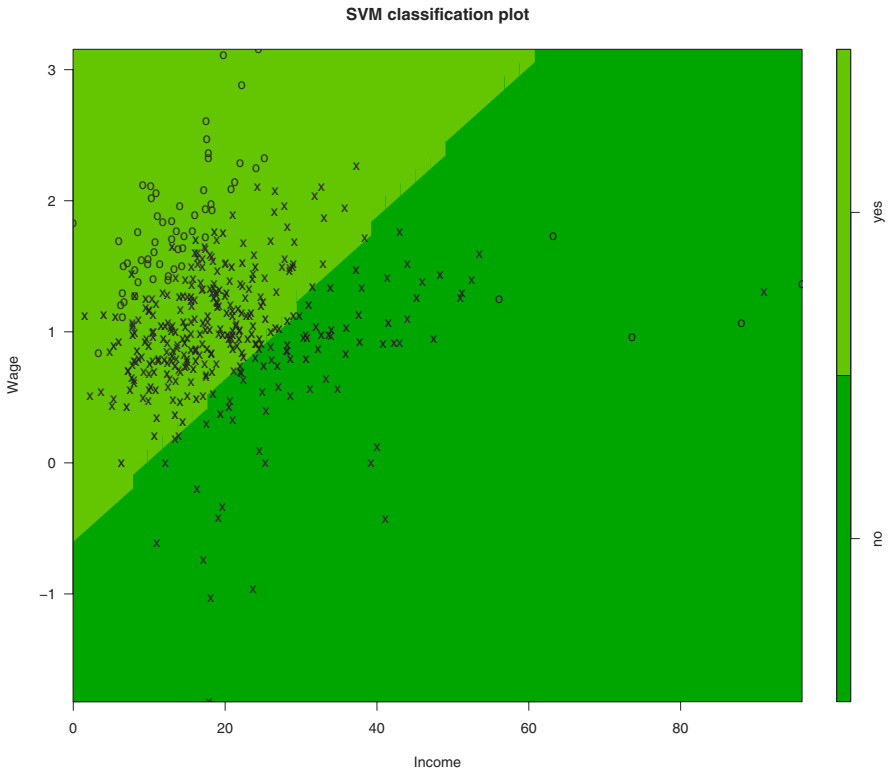


Fig. 7.6. Plot of observations, support vectors, and classifications for linear kernel.

Employed wives are identified as coming from households with lower incomes but who have higher expected wages (in log units). The diagonal decision boundary implies that an additive model has been applied. Overall, the story seems sensible enough, but perhaps a nonlinear kernel can do better.

	Predict Unemployed	Predict Employed	Model Error
Unemployed	110	64	.37
Employed	52	174	.23
Use Error	.32	.19	Overall Error = .29

Table 7.2. SVM Confusion table for employment: radial kernel, $cost = 1$, $\gamma = .50$, 275 support vectors.

7.4.2 SVM Analysis with a Radial Kernel

Table 7.2 shows how well SVM classifies when a nonlinear kernel is used, in this case the radial kernel. The radial kernal is popular because it is relatively easy to understand and seems to perform well. Specifically, we use

$$K(x_i, x_j) = e^{-\gamma|x_i - x_j|^2}. \tag{7.16}$$

The distance between a pair of observations is squared, multiplied by $-\gamma$, and exponentiated. Because of the negative sign, for a given value of γ, larger distances between observations have smaller kernel values. It also follows that larger values of γ for a given distance have smaller kernel values. In other words, smaller values of γ make differences in the distances between observations in the predictor space more important; observations are spread out more with respect to one another. Therefore, γ can be a useful tuning parameter.

For Table 7.2, the same tuning parameters are as before, but γ is set to the default value of .5. Somewhat fewer support vectors are found implying that fewer observations are difficult to classify. Overall, there is a dramatic improvement in fitting skill, implying the linear kernel was missing important relationships between the two predictors and the response that were uncovered by the radial kernel.

The overall proportion of cases misclassified is .29. The SVM model misclassifies unemployed wives 37% of the time and employed wives 23% of the time. The better performance, compared to the linear kernel comes from a substantial improvement in the ability to correctly classify unemployed wives.

But the improved classification skill comes at a high price. The classification plot shown in Figure 7.7 is challenging to interpret. Wives who fall in the middle ranges of the log of expected wages tend to be unemployed almost regardless of household income. In other words, a U-shaped relationship between the log of expected wages and employment surfaces that holds across most income levels. Wives with relatively high or relatively low expected wages are more likely to be employed than wives with middling expected wages. Why wives with relatively low expected wages behave as do wives with relatively high expected wages is not apparent and perhaps reflects the role of predictors not included in the model. The exception to this pattern is found when household income is very low. Then, wages do not seem to matter at all. Wives from very low income households are classified as employed.

7.4.3 Varying Tuning Parameters

It usually important to vary the SVM tuning parameters to see which give the best results. Hsu and his colleagues (2007) suggest trying a range of cost values between 2^{-5} and 2^{15}, and γ values between 2^{-15} and 2^{3}, ideally in a highly systematic manner. Over such a wide range of values, there will likely be dramatic changes in performance.

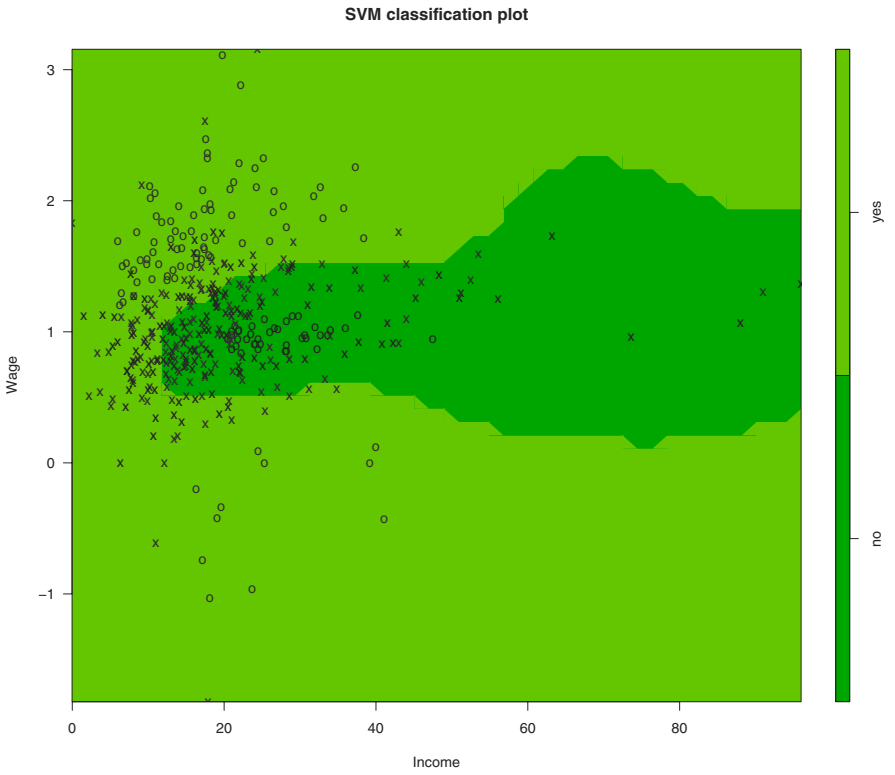


Fig. 7.7. Plot of Observations, Support Vectors, and Classifications for Radial Kernel.

The overall classification skill shown in Table 7.2 is somewhat robust to variation in the values for cost. Essentially, the same overall classification skill materializes with values for cost between .1 and 10,000. However, the ratio of false positives to false negatives can change enough to matter and the classification plots can change substantially. In particular, the unemployment "hole" changes shape and size as the cost parameter increases. And with costs much above 10, the classification plot begins to fragment in a manner that makes little substantial sense. Figure 7.8 shows one example when costs are set at 10. Overall classification is not materially affected because the same region, located to the center left of the plot, contains largely the same observations classified the same way regardless. Put another way, the shape and size change most where there are very few observations. One important lesson is much like that learned for smoothers. Great caution is required if one tries to draw substantive conclusions in regions where there are few data.

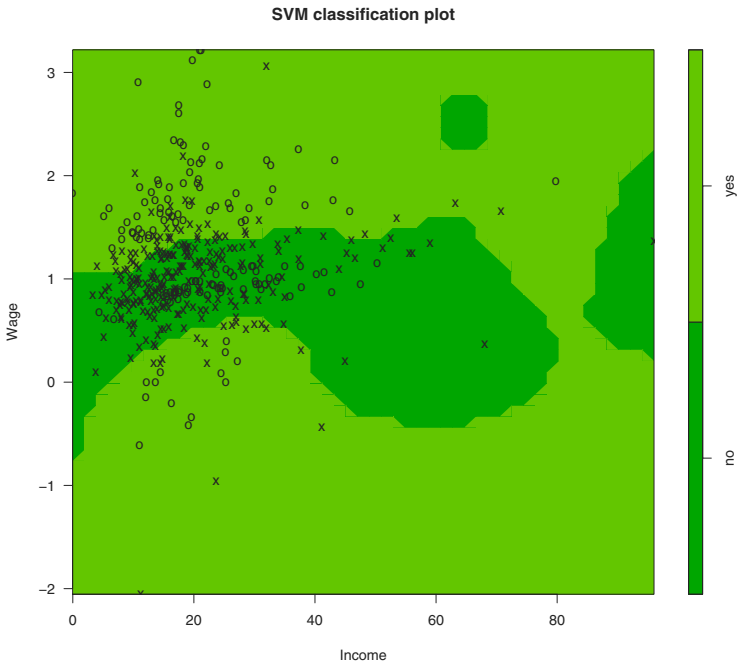


Fig. 7.8. Plot of Observations, Support Vectors, and Classifications for Radial Kernel with cost=10

Varying γ between .0001 and 10 can have very important effects. Until a value for γ of about .005 is reached, there is virtually no classification skill beyond the marginal distribution. Useful classification skill can be obtained thereafter, but γ values larger than about 5 lead again to fragmentation of the classification plots that is very difficult to interpret. Figure 7.9 is a representative illustration.

There is no formal justification for the ranges of cost and γ values tried and unfortunately, there seems to be no principled way to choose among the results that can follow by varying the values of the SVM tuning parameters. One has the option of trial and error with some measure of classification skill as the criterion. Indeed, the procedure tune.svm() will undertake a grid search over specified values of several tuning parameters with some overall measure of performance the arbiter. A key problem with such trial-and-error approaches is that substantive concerns have no direct way to contribute. One risks results that are substantive nonsense. Another concern is that one cannot easily take into account differential classification skill for the true positives and true negatives. Finally, overtuning is encouraged. Even with test data, repeated trials over a large grid will soon lead to a model tuned to the test

Fig. 7.9. Plot of observations, support vectors, and classifications for radial kernel with γ=5.

data. The same dangers exist with cross-validiation or bootstrap approaches.

7.4.4 Taking the Costs of Classification Errors into Account

We have not yet addressed the issue of how to value false positives and false negatives. In part because of the concern with overall classification accuracy, there is little discussion in the SVM literature about taking the costs of false positives and false negatives into account. But in svm(), one can manipulate the number of false negatives and false positives in a confusion table by weighting the response categories. In the spirit of CART and random forests, one can implicitly alter the prior distribution of the response by weighting the data.

The prior distribution of the response variable has 43% of the wives unemployed and 57% of the wives employed. Suppose one gives unemployed wives a weight of .60 and unemployed wives a weight of .40. The intent is to change

the 43–57 marginal distribution into something closer to 60–40. One might employ weights because of what the proper marginal distribution should be (e.g., to correct for oversampling employed wives) or to alter the relative number of false positives and false negatives so that their ratio is more responsive to policy considerations.

	Predict Unemployed	Predict Employed	Model Error
Unemployed	154	17	.10
Employed	98	131	.43
Use Error	.39	.11	Overall Error = .29

Table 7.3. SVM confusion table for employment with weighting: radial kernel, $cost = 1$, $\gamma = .50$, 268 support vectors.

Table 7.3 shows the result. The overall amount of forecasting error has not changed materially, but predictive skill has shifted toward unemployed wives. We are doing better with unemployed wives and worse with employed wives. Consistent with this, the implicit relative costs of false negatives and false positives has changed substantially. Without weighting, the costs of false negatives and false positives were about the same. There were 64 incorrectly classified unemployed wives and 52 incorrectly classified employed wives. Now, there are 17 incorrectly classified unemployed wives and 98 incorrectly classified employed wives. In short, weighting may in practice be a useful approach for altering the prior distribution of the response and the relative costs of false positives and false negatives. For this simple analysis, however, there seems to be no clear rationale for choosing a particular prior other than the empirical prior, which is the implicit default.

7.4.5 Comparisons to Logistic Regression

One might wonder how SVM compares to a logistic regression analysis of the same data. If a radial SVM kernel is used with the default values of the tuning parameters, logistic regression's overall forecasting accuracy is about 50% worse than SVM's when both SVM and logistic regression are evaluated with the test data ($N = 353$). One explanation might be that there are important nonlinear relationships between the two predictors and the response, and SVM is able to find them. Logistic regression cannot find them because logistic regression is committed to whatever functional forms one builds in. Here a simple additive model is used. Such an interpretation is consistent with the earlier comparisons between the SVM results with a linear kernel and a radial kernel.

Alternatively, the nonlinear relationships that SVM finds might be papering over the impact of omitted variables. In other words, although SVM is

better able to link the two predictors to the response, the true roles of the two predictors may be misrepresented in the classification plots. Recall that just this concern was raised earlier. Given the two predictors, SVM is able to show in the classification plots how those predictors are related to the decision boundary. But one must be cautious about attributing to these two variables substantive relationships that may really be in part a function of predictors not included in the analysis. If a richer set of predictors were available, the performance differences between SVM and logistic regression might be reduced. Put another way, a two-predictor description may look very different from a description using more than two predictors.

Consider a somewhat more complicated analysis. The response variable is the same as before but there are now five additional predictors: (1) the number of children in the household 5 years of age or younger, (2) the number of children in the household 6 to 18 years old, (3) whether the wife attended college, (4) whether the husband attended college, and (5) the wife's age.

	Predict Unemployed	Predict Employed	Model Error
Unemployed	89	65	.42
Employed	51	148	.26
Use Error	.36	.30	Overall Error = .32

Table 7.4. Logistic regression confusion table for forecasting employment.

Table 7.4 shows the confusion table for the logistic regression analysis. The table is constructed from the test data. Table 7.5 shows the confusion table for the SVM analysis. It too is constructed from the test data. The overall forecasting error is the same. There are modest differences in forecasting skill for employed wives and unemployed separately. Logistic regression does somewhat better with employed wives and SVM does somewhat better with the unemployed wives. There are related differences in the ratio of false positives to false negatives. But on its face, it is not apparent which model is preferable. One possible implication is that once a richer set of predictors is used, the linear relationships imposed by logistic regression are sufficient. There are no important nonlinearities remaining, at least as a function of the predictors available.

It would be useful if one could examine a classification plot for the SVM analysis. Unfortunately, on this svm() stumbles. Because there are seven predictors, the space defined by the predictors is in seven dimensions. What svm() offers is to plot any plane in that space. There are in practice a very large number of such planes (in theory a limitless number) and no principled way to determine which should be examined. Moreover, if the goal is to understand how predictors are related to the response, a plane through a high-dimensional space is not likely to be helpful unless the decision boundary is very simple.

	Predict Unemployed	Predict Employed	Model Error
Unemployed	111	43	.28
Employed	71	128	.36
Use Error	.39	.25	Overall Error = .32

Table 7.5. SVM Confusion Table for Forecasting Employment: $cost = 1$, $\gamma = .125$, 278 Support Vectors

More likely, the decision boundary will change in important ways from plane to plane in ways that will defy any simple explanation.

One possible solution would be to construct marginal plots, in the spirit of partial dependence plots, by integrating out all but one predictor dimension at a time. Such plots would be far more interpretable but at the price of throwing out a lot of information. In particular, interaction effects would be lost unless interaction variables are constructed and included among the predictors.

In summary, SVM has its roots in classification problems so that classification accuracy has been a driving concern. Less attention has been paid to representing how inputs are related to outputs so that from a regression perspective, SVM has some serious limitations. However, there seems to be no reason in principle why analogues to importance plots and partial dependence plots could not be developed, and it is likely that they will be introduced in the near future.

7.5 Software Considerations

There are several free software packages available to download that have good implementions of SVM. There are also lots of free modules that can be downloaded and cobbled together (see http//www.kernel-machines.org/). In R, the procedure SVM (in the R library *e1071*) implements four different kernel functions, two kinds of classification procedures and two kinds of regression procedures. Initial experience with the software is good. It runs well and seems to be largely bug-free. Two other SVM programs in R that seem to work satisfactorily are in the library *svmpath* and *ksvm*. The discussion that follows concentrates on the SVM implementation in *e1071*, but the comments apply more generally.

For reasons discussed earlier, it is not easy to learn from SVM how inputs are related to outputs. The SVM software in R relies on plots much the same as those used in this chapter. Multivariate relationships are, therefore, very difficult to visualize and in addition, one is effectively limited to quantitative predictors. There is also no direct help in determining predictor importance.

A second difficulty with the software is that the user must specify the kernel function. As noted earlier, guidance is pretty thin on which kinds of kernel functions work better for which kinds of data. Typically, there is a bit

of theory and some simulations suggesting that if the data are generated by a certain kind of stochastic process, one kind of kernel may perform better than another kind of kernel. However, in practice one usually has little idea what stochastic process has generated the data, so such information is not very helpful. In effect, the kind of kernel becomes a tuning parameter.

A third difficulty is that the software requires at least one tuning parameter, and usually two. One must always specify the value of the tuning parameter that determines the weight given to the slack observations. However, there is usually no clear way to link that value to subject matter or decision-making concerns. For most kernels, a second tuning parameter is needed, whose relationships to the data and the goals of the analysis are even more distant.

The current advice, at least for "beginners," is to proceed by trial and error using a cross-validation measure of generalization error (Hsu et al., 2007). For each combination of tuning parameters, a cross-validation measure of performance is computed with the training data. Eventually, the set of values with the smallest generalization error is selected. These are then applied when SVM is used on the full set of training data to arrive at final results. Alternatively, the trial-and-error process can be automated with tune.svm(), as noted earlier

This strategy has several related problems. To begin, the informal grid search strategy recommended has a large number of grid points. The strategy is time consuming. An automated approach can help a lot in cutting the amount of human labor required, but substantive considerations get short shrift.

In addition, the application of a cross-validation measure of performance is usually often a good idea and in principle, overfitting can be usefully addressed. However, the grid search procedure goes back to the same well many times, so the assets of cross-validation are gradually dissipated. The same data are being recycled over and over. Similar problems affect the use of test data and the bootstrap. And overtuning leads to overfitting.

Also, the benchmark for the grid search procedure is some estimate of overall forecasting error. This ignores potentially important differences between the errors that result from trying to forecast "successes" and the errors that result from trying to forecast the "failures." Misleading results can follow. At the very least, important information is not taken into account.

A final difficulty with the software is that there is no direct way to address the relative costs of false negatives and false positives. The software provides a method to reweight the data to adjust for unbalanced response variable distributions and it seems that the weights can be used much as a prior distribution. In effect, the weighting becomes another tuning parameter. But, the formal justification for this approach is thin and one could in principle try to address differential costs with combinations of other tuning parameters.

Perhaps the best advice at this point is to undertake the data analysis with statistical learning procedures that have the requisite tools for examining the output and that seem to be easier to implement in an informed

manner. Then, SVM can be applied to the same data. In the comparisons across statistical learning tools, the performance of SVM may be better understood and appreciated. Perhaps then it can be used in a more informed manner.

7.6 Summary and Conclusions

Support vector machines has some real strengths. SVM was developed initially for classification problems and seems to perform well in a variety of real classification applications. As a form of robust regression, it may also prove to be useful when less weight needs to be given to more extreme residuals. And, the underlying fundamentals of support vector machines rest on well-considered and sensible principles.

However, SVM was not developed to capture the $f(X)$ and therefore, does not fit well within a regression framework in which one cares about how predictors are related to a response. It also stumbles badly as an exploratory tool to inform future work on some $f(X)$. Finally, although SVM can often fit the data at least as well as random forests and stochastic gradient boosting, it is rare to find applications for which it performs dramatically better.

At the same time, it is difficult to arrive at an overall assessment. Support vector machines is still evolving rapidly with new kernel functions and computer algorithms appearing on a regular basis. The tradeoffs between the various kernels and between them and alternative statistical learning procedures is not at all clear. In short, it is difficult at this time to make a strong case one way or another for support vector machines compared to procedures such as random forests and boosting even if the primary goal of the primary data analysis is to fit the response as well as possible.

Exercises

Problem Set 1

Construct a dataset as follows. You will need to load the *Rlab*.

```
w<-rnorm(500)
z<-rnorm(500)
w2<-w^2
x<-(-1+3*w2-1*z)
p<-exp(x)/(1 + exp(x))
library(Rlab)
y<-rbern(500,p)
```

As you proceed, you will need to read carefully the help commands for the various procedures to determine which require that y be a factor, and the form in which the data are expected.

1. Regress y on w and z using logistic regression and construct a confusion table with the resubstituted data. You know that the model has been misspecified. Examine the regression output and the confusion table. Now regress y on $w2$ and z using logistic regression and construct a confusion table with the resubstituted data. You know that the model is correct. Compare the two sets of regression coefficients, their hypothesis tests, and two confusion tables. How does the output from the two models differ? Why?

2. Repeat the two analyses using svm() from the library *e1071*. Use the default settings. First regress y on w and z and then regress regress y on $w2$ and z. Compare the confusion tables and classification plots from the two analyses. How does the output from the two models differ? Why?

3. What do you answers to the first two question tell you how about how SVM responds to nonlinear relationships that are not introduced explicitly in the predictors used?

Problem Set 2

From the *MASS* library, load the Pima.tr dataset. The variable "type" is the response. All other variables are predictors. Apply svm() to the data (in library *e1071*). Use all of the predictors and the defaults. Study the output by constructing a confusion table and some classification plots. There are a very large number of possible classification plots. To get a feel for what they can reveal, use"glu" and "bp" to define the two axes and set the other predictors at their means. So, you get to see the decision boundary for "glu" and "bp" with all other variables fixed at their average values. Now see what happens when the predictors are set at their first quartile and then their third quartile. There is no "right" classification plot. Each shows a different part of the decision boundary in different locations of the predictor space.

1. Apply svm() to the data (in library *e1071*). Use all of the predictors and the defaults. Study the output by constructing a confusion table and a classification plot. Use the first plot constructed above. Now change the default radial kernel to a linear kernel. Study the output by constructing a confusion table and a classification plot. Compare the output from the radial kernel to the output from the linear kernel. How are they much the same and how are they somewhat different. Why?

2. For the response variable type, about a third of the observations are "yes" and about two-thirds are "no." Try imposing a prior with the proportions reversed. For example, the weights can be constructed with

```
wts=table(type)
wts[1]=.3
wts[2]=.7
```

Repeat the earlier analysis with the radial kernel. Also consider the ratio of false positives to false negatives. How have things changed? Are the changes consistent with the alternation in the attempt to alter the prior? Look at both a confusion table and a classification plot. For the latter, use"glu" and "bp" to define the two axes and set the other predictors at their means.

3. With the weighting as before (the default), repeat the analysis with cost parameter values of .1 and 10. How have things changed? Consider both a confusion table and a classification plot. For example, what does the classification plot convey when the cost is set at 10?

4. With the weighting as before and the cost parameter set to 1, try a values of γ of .01, .5, 1, and 5. How have things changed? Consider both a confusion table and a classification plot as before. How does the output change? From the confusion table alone, which value of γ produces the most accurate classifications? Is this the best model? Why or why not?

8

Broader Implications and a Bit of Craft Lore

It is difficult to arrive at any overall assessment about the state of statistical learning. Statistical learning is currently undergoing very rapid growth and change. Efforts to evaluate the relative merits of various procedures are further complicated by little agreement on what the performance yardsticks should be: goodness-of-fit, forecasting accuracy, robustness, interpretabilty, computational feasibility, consistency proofs, and so on.

One consequence is that the number of alternative procedures is large, growing, and changing. Another consequence is that there are no clear winners among the list of credible contenders. A final consequence is that because there is little policing of claims made, one must evaluate all assertions about performance with considerable care. This may be especially true about claims about statistical learning procedures made by for-profit enterprises.

Nevertheless, it is probably useful to assemble what craft lore there may be and to provide some general observations and suggestions. Both are offered with some trepidation. They could easily change as more experience is gained.

8.1 Some Fundamental Limitations of Statistical Learning

It is important to start at the top. Ideally, what is statistical learning supposed to get done? What are its goals? Recall that in most formal expositions of statistical learning, especially those within a regression framework, there exists in nature an explicit data-generation mechanism. Key features of that mechanism can be represented in a function linking a set of predictors to a response. Then, the function, symbolized by $f(X)$, can be assigned one of two roles. If the conditional distribution of the response is assumed to be the result of a causal process, $f(X)$ represents the causal machinery. It depicts how independent manipulation of each predictor alters the value of the response. Alternatively, the $f(X)$ can be used to describe how the conditional

R.A. Berk, *Statistical Learning from a Regression Perspective*,
DOI: 10.1007/978-0-387-77501-2_8,

distribution of the response varies with different x-values, but with no causal interpretation. One has the conditional distribution, but nothing more.

If either of these accounts is credible, the goal of statistical learning can be to determine the function linking predictors to the response. One must have in the dataset all of the predictors, and each must be very well measured. Then, if the training data can be plausibly treated as a random sample from an appropriate population or a random realization from the relevant stochastic process, many statistical learning procedures can be seen as an effort to obtain a good estimate of the $f(X)$. "Good" can be defined in several different ways, but minimizing generalization error is a common yardstick.

Even under ideal circumstances, however, statistical inference can be problematic. If the goal is to construct confidence intervals, very little may be known about the relevant sampling distribution. If a priori ignorance prevails about the $f(X)$ as well, there may be no sensible hypotheses to test that were posed before the data were examined. Hypotheses generated as part of the data analysis process violate a key assumption of statistical tests. The computed p-values will likely be too small.

More fundamentally, one must be clear that there are no formal mathematical results stating that statistical learning procedures will accurately capture the $f(X)$. There also are no proofs of unbiased estimation, and the consistency proofs that exist to date address generalization error for some population model. There is no guarantee that the population model is in any sense "right" or even useful. Moreover, there remains the usual difficulty of figuring out what asymptotic results convey about the results from the data on hand.

In practice, matters are usually worse. One will rarely have all of the required predictors, and it will be rare indeed for all of them to be well measured. Therefore, statistical learning applications will more typically be exploratory and descriptive, and occasionally be the basis for forecasting.

Perhaps most important, one must not see statistical learning as the magic bullet of data analysis. Over the past several decades great promises have been made for any number of statistical procedures, which sometimes proved to be useful, but hardly the revolutions that many of their advocates claimed: SEM, ARIMA, HLM, robust regression, raking, specification tests, sliced inverse regression, Rasch models, ARCH, meta-analysis, life history analysis, log-linear models, latent class analysis, and on and on. So that there be no mistake, no such claims are being made here about statistical learning.

8.2 Some Assets of Statistical Learning

The benefits of statistical learning can be organized into three large categories. Some of the benefits boil down to an attitude adjustment. Others come from an ability to better address certain kinds of data analysis tasks. A final set involves improving other statistical procedures.

8.2.1 The Attitude Adjustment

Statistical learning may be seen as implicit criticism of conventional causal modeling that has for several decades dominated much research in the social and life sciences. There are certainly many overt critiques of business as usual (Box, 1976; Leamer, 1978; Rubin, 1986; Freedman, 1987; Manski, 1990; Heckman, 1999; Breiman, 2001b: Berk, 2003: Freedman, 2005). Statistical learning, by contrast, delivers its message by example. And this message has several related themes.

First, statistical learning can proceed quite happily without worrying much about cause and effect. Inputs are associated with outputs, but the inputs are not necessarily causes and the outputs are not necessarily effects. One can use causal insights to help determine inputs and outputs when prior information about cause and effect is available. But, within the framework emphasized in this book, outputs are nothing more than variables whose conditional distributions are of interest. Inputs are nothing more than the variables to be used in the conditioning. Consequently, the inputs may be selected to help in classification or forecasting even if they play no causal role whatsoever.

Second, the data do not have to be generated by a real intervention. This means that randomized experiments, for example, usually deliver no special leverage. There is also no formal need to proceed as if the predictors in an observational dataset were treatments that could be manipulated. It is often said that experiments are the gold standard for causal inference. Unless determining the effects of causes is a key goal of a statistical learning data analysis, the underpinnings and tools of causal inference are likely to be irrelevant

Third, the most important statistical benchmark for a successful data analysis is successful forecasting. If fitted values correspond well to observed values one may be on the right track. But a good fit is not good in and of itself. Rather, it implies that the forecasts may be good as well. This means that goodness-of-fit statistics and tests are in statistical learning rarely of much interest. The broader message is that one must look beyond the data used to build a model in order to determine if that model has genuine merit.

Fourth, with forecasting skill as the standard, many statistical learning procedures perform well, and very often substantially better than conventional causal modeling. One reason can be that a statistical learning procedure may use more of the information in a dataset than a conventional model does. Another reason can be that the conventional models represent the associations between the predictors and the response less well than an inductive model does. Finally, statistical learning procedures are often designed to maximize forecasting skill. Conventional models rarely are.

At the same time, there is nothing in statistical learning that precludes causal thinking. With conventional causal modeling, the regression coefficients associated with each predictor are supposed to reveal what the average change in the response would be if the predictor values for a given observation were actually altered (e.g., a person with no high school degree obtained one). In

statistical learning, partial response plots make no such claim. They simply show how, on the average, the response variable varies depending on the value of a given predictor, with the values of all other predictors fixed at their current levels. Likewise, plots of predictor importance are not plots of regression coefficients; they do not represent causal importance. And one must not think that tree diagrams are anything like path diagrams.

Partial plots, importance plots, classification plots, and tree diagrams can, however, provide some ideas about how the response variable might change if a predictor were manipulated. These can be very useful for designing future studies to get directly at possible causal relationships (e.g., with a randomized experiment). Partial plots, importance plots, classification plots, and tree diagrams can also generate new ideas and even theoretical insights. In short, statistical learning results can be useful for understanding possible causal relationships but unlike causal models, they are not meant to be surrogates for real experiments.

Finally, perhaps the most important attitude adjustment is that description is a noble and useful scientific enterprise. One can do good science and not do causal modeling or experiments.

8.2.2 Selectively Better Performance

Although statistical learning of the sort described in this book is certainly more than a niche player, it performs far better at some tasks than others. Here are some tasks at which statistical learning can excel.

1. Determining Functional Forms—When the response functions are not known but are likely to be highly nonlinear, statistical learning procedures can shine. Even if no claims are made that the true $f(X)$ has been determined, important information about that function may be revealed. For example, there may be good evidence that the function is roughly S-shaped and for which x-values the response is changing especially rapidly. This can be extremely instructive, even if the particular S-shaped curve (e.g., cumulative normal v. cumulative logistic) cannot be identified. Recall the many earlier examples in which unanticipated nonlinear functions were found.
2. Discovering Unexpected Predictors—The ability of many statistical learning procedures to exploit a large number of inputs means that some predictors, or transformations of predictors, that might have unanticipated relationships with the response variable can be found. This is true for CART, but especially boosting and random forests. For example, in research on racial bias in decisions to charge with a capital crime, some unexpected two-way and three-way interactions are sometimes needed (Berk et al., 2005a).
3. Discovering Which Predictors Matter—Even if there is a broad consensus about the predictors that need to be included in the training data, there

may be little agreement about how important each of the predictors is. If one finds contribution to the fit or to forecasting skill a useful definition of "importance," several statistical learning procedures can provide instructive measures of predictor importance. For example, particularly for the most serious form of prison misconduct, sentence length delivers the most forecasting skill (Berk et al., 2006).

4. Providing Useful Regression Diagnostics—By being able to find useful predictors whose roles were unanticipated and by being able to reveal unanticipated response functions, statistical learning procedures can serve as very useful diagnostic tools for parametric regression models (Berk et al., 2005a). More is said about this shortly.
5. Avoiding or Compensating for Overfitting—In exploratory data analysis, overfitting is almost unavoidable. But bagging can correct in part for overfitting, and random forests does not overfit. Boosting can overfit in principle, but as commonly used does not seem to overfit significantly. Moreover, statistical learning is embedded in a statistical tradition where concerns about overfitting and tools to counter overfitting are very salient. There are several measures of fit adjusting for degrees of freedom, cross-validation, the use of test data, and a variety of regularization methods.
6. Forecasting—Especially if the data do not constitute a time series, random forests and boosting are probably state-of-the-art forecasting tools. For example, there seems to be some success to be had using random forests to forecast future incidents of serious domestic violence (Berk et al., 2005b)
7. Responding to asymmetric costs—All of the procedures discussed require a commitment to a particular costs/loss function. But when the response variable is categorical, CART and random forests are especially able to take account of asymmetric costs. For many applied problems, asymmetric costs are critical because the costs of false negatives can be very different from the costs of false positives. Many examples were provided in earlier chapters. Asymmetric costs for quantitative variables are at a very early stage. But quantile random forests has some promise as do methods being developed for stochastic gradient boosting.
8. Exploiting Many Predictors—When there are a very large number of predictors, most procedures attempting to link inputs to outputs will stumble, especially if the number of predictors exceeds the number of observations. CART and random forests are particularly strong in this regard and can handle hundreds of predictors even if there are far more predictors than cases. The main limitation is not the statistical procedure, but computing power. Some have claimed that because SVM works with the cross-products between observations, the curse of dimensionality is lifted. This turns out to be at least an exaggeration (Hastie et al., 2001: 384–385).

8.2.3 Improving Other Procedures

The results of a statistical learning analysis can be a useful intermediate step in another statistical procedure. The key attribute that statistical learning brings to these tasks is an ability to construct fitted values that can correspond especially well to the actual values. When a good fit is very important, statistical learning procedures can be especially effective.

A useful example is in the construction of propensity scores (Rosenbaum, 2002; McCaffrey, 2004). If one is interested in the causal impact of some categorical intervention, such as a school program to enhance reading, and if that intervention is not provided through random assignment, there will be the substantial likelihood of selection bias. In the school illustration, students who participate in such a program will probably differ on the average from students who do not. These "pre-existing" differences are then confounded with estimates of the impact of the reading program.

However, if an unbiased estimate of the probability of program participation can be constructed, that estimated probability can be used as an effective statistical control through matching or other means (Imbens, 2004). The probability of program participation is called a "propensity score." In principle, using propensity scores to adjust for selection bias in estimates of the treatment effects can be an effective tool.

For example, propensity scores can be used as weights in what is called the "difference-in-differences" estimator (Heckman et al., 1998). The data analysis problem is still the same: selection bias into the experimental and control groups. But now there is a pretest and a posttest. The difference-in-differences estimator compares the members of the experimental group to members of the control group through the change in their performance between the pretest and the posttest, with those changes weighted by propensity scores. If those propensity scores are estimated in a consistent manner, the difference-in-differences estimator will in principle deliver a consistent estimate of the treatment effect. (But see Freedman and Berk, 2008).

Usually propensity scores are estimated with logistic regression, and one always has to wonder how close to unbiased the fitted values really are. Statistical learning procedures can improve the quality of those fitted values by making them closer to the actual values. Then, instructive estimates of the treatment effects may be more likely. But the issues are a little tricky. Conventional boosting procedures applied to classification problems would seem to be a tool of choice, but it risks pushing any estimated probabilities away from .50 toward 0.0 and 1.0. Recall that an alternative approach was discussed.

Another useful asset of statistical learning procedures is the very flexible functional forms that are often constructed. These, in turn, can be instructive as conventional regression models are built. As noted earlier, statistical learning procedures may reveal predictors or functional forms not anticipated by conventional regression models. These models might then be revised accordingly. Conversely, predictors that seem to be important within a conven-

tional regression model may vanish when more appropriate functional forms are used. This might suggest dropping such regressions from the regression analysis.

Yet another example is improving covariance adjustments. Recall that covariance adjustments for a given predictor depend on residualizing that predictor and the response with respect to all other predictors. That is, any linear dependence between the response and the other predictors is removed and any linear dependence between the given predictor and the other predictors is removed. But the quality of this "purging" depends on using the right predictors with the correct functional forms. Many statistical learning procedures can exploit hundreds of predictors and search for the functional form that fits the data best. Then, one option is applying statistical learning tools to residualize the response and key predictors for the "nuisance" covariates before the important relationships are examined.

8.3 Some Practical Suggestions

Just as for any other set of statistical procedures, practice is guided significantly by craft lore. In that spirit, we turn to a bit of craft lore about the use of statistical learning. It is important to keep in mind, however, the craft lore can change dramatically with experience, and the experience with statistical learning to date is somewhat spotty.

8.3.1 Matching Tools to Jobs

To begin, it can be useful to reconsider which procedures are likely to be most effective for which data analysis tasks. The smoothers discussed in Chapter 2 are primarily visualization aids that can be applied in a wide variety of settings. They can be taken as standalone tools, as when one smooths a two-dimensional or three-dimensional scatterplot. They can be used as a component of other procedures, such as the generalized additive model. Their main strength is providing information in a very accessible manner about how predictors are related to a response variable.

If the primary goal is good fit, accurate classification, and/or accurate forecasting, the procedures discussed in Chapters 4 through 7 are likely to be a better choice: random forests, boosting, and support vector machines. CART can be a handy way to hunt for possible interaction effects and can serve as a intermediate step for more powerful statistical learning procedures. But otherwise, CART has largely been superseded.

Random forests, boosting, and support vector machines can all perform well. It is not clear yet which perform better for which kinds of datasets, or even if the differences in performance are likely to matter a great deal in practice. It is easy to get caught up in differences of a few percent in forecasting accuracy, which are too small to matter and may not hold up.

For example, suppose the goal is to forecast which high school students are at the greatest risk for dropping out. A classifier might be trained on data from several cohorts of high school students and evaluated with test data that are a random sample from the same population. But as a practical matter, the forecasting tool developed would be applied to new cohorts of students that would likely differ from the training and test samples by more than random sampling error. One might well expect a gradual drift in the background of incoming freshmen and the mix of incentives to remain in school. As a result, forecasting accuracy could decline by an amount that could easily swamp the performance differences between competing classifiers, and a new analysis might show that the classifier that had previously performed best no longer did. In short, differences in performance that may be of methodological interest may also be of no practical importance.

Therefore, one key factor in choosing between statistical learning tools can be the quality of the output. To date, there are important differences among random forests, boosting, indexboosting and support vector machines beyond the predicted values and a confusion table. If one needs to examine response functions and evaluate predictor importance, an implementation of support vector machines may not have what is needed. There are implementations of random forests and boosting that do, although at the moment, the measures of variable performance in random forests, which exploit the out-of-bag data for forecasting, is probably more desirable.

Another key factor in choosing among statistical learning tools is their ability to address in a flexible manner the relative costs of false negatives and false positives. Currently, random forests is likely to do this better for classification problems than either boosting or support vector machines. More generally, asymmetric loss functions can be important. There are very recent developments for random forests and stochastic gradient boosting that have the promise of allowing for certain special cases.

One should also consider the relationship between the sample size and the number of predictors. If the number of predictors exceeds the number of observations, random forests may be the only viable choice among the better-performing statistical learning procedures. There may even be reason to prefer random forests if the number of predictors is large compared to the number of observations, even if there are fewer predictors than observations.

A final factor is the range of response variables that can be properly analyzed. At this point, boosting may be the most flexible, at least within the gradient boosting approach. But there seems to be no reason in principle why random forests and support vector machines cannot be made more broadly applicable, and it is likely that the range of response variable types that can be handled by these procedures will increase over the next several years.

8.3.2 Getting to Know Your Software

There is not yet, and not likely to be in the near future, a consensus on how any of the various statistical learning procedures should be implemented in software. For example, a recent check on software available for support vector machines found working code for over a half dozen procedures. There is, as well, near anarchy in naming conventions, and notation. Thus, the term "cost," for instance, can mean several different things and a symbol such as γ can be a tuning parameter in one derivation and a key feature in another derivation.

One cannot assume that a description of a procedure in a textbook or journal article corresponds fully to software using the very same name. Consequently, it is very important to work with software for which there is good technical documentation on the procedure and algorithms being used. There also needs to be clear information on how to introduce inputs, outputs, and tuning parameters. Two computer programs can use the same name for different items, or use very different names for the same item. And in either case, the naming conventions may not correspond to the naming conventions in the technical literature.

Even when the documentation looks to be clear and complete, a healthy dose of skepticism is useful. There are sometimes errors in the documentation, or in the software, or in both. So, it is usually important to "shake down" any new software with data that have previously been analyzed properly to determine if the new results come out as expected. In addition, it is usually helpful to experiment with various tuning parameters to see if the results make sense. In short, *caveat emptor*.

It is also very important keep abreast of software updates, which can come as often as two or three times a year. As a routine matter, new features are added, bugs fixed and documentation rewritten. These changes are often far more than cosmetic. Working with an older version of statistical learning software can lead to unnecessary problems.

Finally, a key software decision is whether to work primarily with shareware such as R or with commercial products. The tradeoffs have been discussed earlier at various points. Cost is certainly an issue, but perhaps more important is the tension between having the most current software and having the most stable software and documentation. Shareware is more likely to be on the leading edge, but often lacks the convenience and stability of commercial products. One possible strategy for individuals who are unfamiliar with a certain class of procedures is to begin with a good commercial product, and then once some hands-on skill has been developed, migrate to shareware.

8.3.3 Not Forgetting the Basics

It is very easy to get caught up in the razzle-dazzle of statistical learning and for any given data analysis, neglect the more simple fundamentals. All data

explorations must start with an effort to get "close" to the data. This requires a careful inspection of elementary descriptive statistics: means, standard deviations, histograms, cross-tabulations, scatterplots and the like. It also means understanding how the data were generated and how the variables were measured. Moving into a statistical learning procedure without this groundwork can lead to substantial grief. For example, sometimes numeric values are given to missing data. Treating these values as legitimate measures can seriously distort any data analysis, including ones undertaken with statistical learning.

It will usually be helpful to apply one or more forms of conventional regression analysis before moving to statistical learning. One then obtains an initial sense of how good the fit is likely to be, the likely signs of key relationships between predictors and the response, and hints of problems that might be more difficult to spot later (e.g., high correlations among some predictors). An important implication is that it will often be handy to undertake statistical learning within a larger computing environment in which a variety of statistical tools can be applied to the same data. This can weigh against single-purpose statistical learning software.

To take one simple example, a tuning parameter in random forests may require a distinct value for each response class. But the order in which those arguments are entered into the function for the tuning parameter may be unclear. In the binary case, for example, which category comes first? Is it $\omega = c(1, 0)$ or $\omega = c(0, 1)$? It is easy to make the wrong choice. Random forests runs just the same and generates sensible-looking output. But the analysis has not been tuned as it should have been. It can be difficult to spot such an error unless one knows the marginal distribution of the response variable and the likely sign of relationships between each predictor and the response. A few cross-tabulations and a preliminary regression analysis can help enormously.

To take a little more complicated example, one of the few graphical displays of output from support vector machines depends on specifying a slice of the control variables used to the subset of the data so that a plane can be plotted. If there are several control variables, it is easy to choose a slice in which there are no data, or too few observations to construct a useful plot. Yet, no error message may be produced, and misleading interpretation can follow. A series of cross-tabulations or conditional plots can help a lot.

8.3.4 Getting Good Data

As noted many times, there is no substitute for good data. The fact that boosting, for example, can make a weak classifier much stronger, does not mean that boosting can make weak data stronger. There are no surprises in what properties good data should have: a large number of observations, little measurement error, a rich set of predictors, and a reasonably well-balanced response variable distribution. The clear message is that it is very important to invest time and resources in data collection. One cannot count on statistical learning successfully coming to the rescue. Indeed, some forms of statistical

learning are quite fragile and easily pulled off course by noisy data, let alone data that have systematic measurement error.

The case for having legitimate test data is a bit more ambiguous. Statistical learning procedures that use out-of-bag data or the equivalent do not formally need a test dataset that is a random sample from the same population as the training data. The out-of-bag observations serve that purpose. But most statistical learning procedures currently are not designed to work with random samples of the data, even when that might make a lot of sense. Therefore, having access to test data is usually very important.

Even for random forests, test data beyond the out-of-bag observations can come in handy. Comparisons between how random forests performs and how other approaches (including conventional regression) perform are often very instructive. Yet such comparisons cannot be undertaken unless there are test data shared by all of the statistical procedures applied. Finally, having a true test dataset can help a great deal if random forests is applied repeatedly to the same training data after changes in the tuning parameters. At the very end of the tuning process, there is still the opportunity to get an honest measure of performance from data that until that moment have not been used.

8.3.5 Being Sensitive to Overtuning

We have discussed several summary measures of model performance that can be used to help tune models. Tuning can be an important process in model development. However, if the tuning process goes on for very long, the desirable properties of the summary measures can be badly diluted.

For example, neither the AIC or BIC take into account the number of parameters estimated for earlier models that were considered and rejected. Cross-validation can be compromised when the same dataset is used over and over. The independence between the training data and the test data is gradually lost.

These concerns suggest a strategy, just noted, in which there is a hold-out sample, or another random sample from the same population, that is used at the very end to evaluate the final model. Ideally there would be a large number of hold-out or random samples that could be used for tuning as well. In addition, it will generally be a good idea to show great restraint when there is an opportunity to tune, particularly when the number of observations is relatively small and the number of predictors is relatively large.

8.3.6 Matching Your Goals to What You Can Credibly Do

Much of the literature on statistical learning is formulated around some $f(X)$. There is a real mechanism by which the data were generated. A key goal of a data analysis is to recover the data-generation function from a dataset. It can be very tempting, therefore, to frame all data analyses in a similar manner.

But, one of the themes of this book has been that in reality, more modest goals are likely to be appropriate. Perhaps most important, one will not have access to all of the requisite predictors, let alone predictors that are all well measured. In addition, various kinds of data snooping will often be difficult to avoid, and even the best adjustments for overfitting may prove insufficient. For these and other reasons, description will be what the data analysis is really about.

This does not mean that the stability of one's results cannot be usefully addressed. It also does not mean that causal thinking is unimportant. And it certainly does not mean that one cannot achieve an improved understanding of what the $f(X)$ might be. For example, entire classes of functions can sometimes be effectively eliminated.

But what it does mean is that much of the formal rationale for any statistical procedure, including statistical learning procedures, cannot be relied upon. It can be very difficult to know, for example, what use to make of proofs of consistency. It also means that packaging one's results as function estimation or as a model of how the data were generated can be false advertising. If description is the enterprise, it needs to be labeled as such.

8.4 Some Concluding Observations

Statistical learning has considerable potential, and its reach and power will likely increase in the next several years. But with that potential comes almost certain misuse. There are already some instructive examples.

As just noted, it can be very seductive to proceed as if the goal were to estimate $f(X)$ even when one does not have the requisite predictors or predictors of the requisite quality. Then, the actual work being undertaken is description. Concepts such as bias and consistency no longer apply and cannot be appealed to. And the work cannot properly be packaged as function estimation.

Another error is to undertake statistical inference as part of a statistical learning analysis when the p-values are not likely to make much sense. The p-values may be wildly misleading because the data are not a random sample or random realization of anything, because the statistical learning procedure invalidates the required assumptions, and/or because the necessary sampling distributions are unknown or not credibly estimated.

Still another error is to accept statistical learning results uncritically. The very flexibility with which fitted values are constructed can lead to results that are factual nonsense. There is also the possibility that software will malfunction or be fundamentally flawed. In general, all results must pass the sniff test of subject matter credibility.

Finally, data snooping can lead to significant data analysis errors. Even data analysts who are well aware of its risks can inadvertently lower their guard and allow data snooping errors to affect their results. For example, a

number of different applications of random forests, using different tuning parameters, may be examined before selecting a single model and the particular values of its tuning parameters.

The concluding message, therefore, is to users of statistical learning results. At the very least, demand that all results to be taken seriously rest on test data or their equivalent. And if the results do not make subject matter sense, skepticism is a sensible stance. Ask that each step in the data analysis, including how the data were collected, be reviewed. If anomalies persist, consider getting an independent third party involved. As with any new and complicated procedure, there is lots of room for mistakes and even a substantial amount of fakery.

References

Akaike, H. (1973) "Information Theory and an Extension to the Maximum Likelihood Principle." In B.N. Petrov and F. Casaki (Eds.), *International Symposim on Information Theory* Budapest: Akademia Kiado: 267–281.

Bartlett, P.L., Jordon, M.I., and J. D. McAuliffe (2006) "Comment." *Statistical Science* 21(3): 341–346.

Berk, R.A. (2003) *Regression Analysis: A Constructive Critique*. Newbury Park, CA.: Sage.

Berk, R.A. (2005a) "New Claims About Executions and General Deterrence: Déjà Vu All Over Again?." *Journal of Empirical Legal Studies* 2(2)L 303–330 .

Berk, R.A. (2005b) "Randomized Experiments as the Bronze Standard." *Journal of Experimental Criminology*, 1(4):417–433.

Berk, R.A. and S. Rothenberg (2004) "Water Resource Dynamics in Asian Pacific Cities." Statistics Department Preprint Series, UCLA.

Berk, R.A., He, Y., and S. Sorenson (2005b) "Developing a Practical Forecasting Screener for Domestic Violence Incidents." *Evaluation Review* 29(4): 358–382.

Berk, R.A., Kriegler, B., and D. Ylvisaker (2008) "Counting the Homeless in Los Angeles County." In D. Nolan and S. Speed (Eds.), *Probability and Statistics: Essays in Honor of David A. Freedman*, Monograph Series for the Institute of Mathematical Statistics, forthcoming.

Berk, R.A., Kriegler B. and J. Baek (2006) "Forecasting Dangerous Inmate Misconduct: An Application of Ensemble Statistical Procedures. *Journal of Quantitative Criminology* 22(2): 131-145. forthcoming.

Berk, R.A., Li, A. and L. Hickman (2005a) "Statistical Difficulties in Determining the Role of Race in Capital Cases: A Re-analysis of Data from the State of Maryland." *Journal of Quantitative Criminology*, 21(4): 365–390.

Berk, R.A., Ladd, H., Graziano, H., and J. Baek (2003) "A Randomized Experiment Testing Inmate Classification Systems." *Journal of Criminology and Public Policy*, 2, No. 2: 215–242.

Bickel, P.J., Ritov, Y., and A Zakai (2006) "Some Theory for Generalized Boosting Algorithms." *Journal of Machine Learning Research* 7: 705–769.

Bishop, C.M. (2006) *Pattern Recognition and Machine Learning*. New York: Springer.

Bousquet, O. and B. Scholköph (2006) "Comment." *Statistical Science* 21(3): 337–340.

Box, G.E.P. (1976) "Science and Statistics." *Journal of the American Statistical Association* 71: 791–799.

Breiman, L. (1996) "Bagging Predictors." *Machine Learning* 26:123–140.

Breiman, L. (1999) "Prediction Games and Arcing Algorithms." *Neural Computation* 11: 1493–1517.

Breiman, L. (2000) "Some Infinity Theory for Predictor Ensembles." *Technical Report 522*, Department of Statistics, University of California, Berkeley.

Breiman, L. (2001a) "Random Forests." *Machine Learning* 45: 5–32.

Breiman, L. (2001b) "Statistical Modeling: Two Cultures" (with discussion). *Statistical Science* 16: 199–231.

Breiman, L, (2004) "The 2002 Wald Memorial Lectures: Population Theory for Boosting Ensembles." *The Annals of Statistics* 32 (1): 1–11.

Breiman, L., Friedman, J.H., Olshen, R.A., and C.J. Stone, (1984) *Classification and Regression Trees*. Monterey, CA: Wadsworth Press.

Brown, L.D., Zhao, L., Mao, V., and J. Xu. (2005) "A Pseudo-Bayesian Procedure for Nonparametric Regression Estimation and Pointwise Confidence Bands." Invited presentation at Bernoulli/IMS meeting, Barcelona.

Bühlmann, P. (2006) "Boosting for High Dimensional Linear Models." *The Annals of Statistics* 34 (2): 559–583.

Bühlmann, P. and B. Yu (2002), "Analyzing Bagging." *The Annals of Statistics* 30: 927–961.

Bühlmann, P. and B. Yu (2004), "Discussion." *The Annals of Statistics* 32: 96–107.

Bühlmann, P. and B. Yu (2006) "Sparse Boosting." *Journal of Machine Learning Research* 7: 1001–1024.

Buja, A. and W. Rolke (2007) "Calibration for Simultaneity: (Re) Sampling Methods for Simultaneous Inference with Application to Function Estimation and Functional Data." Working paper at www-stat. wharton.upenn.edu/ buja/.

Buja, A. and W. Stuetzle (2000) "Smoothing Effects of Bagging." Working paper at www-stat. wharton. upenn.edu/ buja/.

Buja, A. and W. Stuetzle (2006) "Observations on Bagging." *Statistica Sinica* 16(2) 323–352.

Buja, A., Mease, D., and A.J. Wyner (2008) "Discussion of Bühlmann and Hothorn." *Statistical Science*, forthcoming.

Buja, A., Stuetzle, W. and Y. Shen (2005) "Loss Functions for Binary Class Probability Estimation and Classification: Structure and Applications."

Unpublished manuscript, Department of Statistics, The Wharton School, University of Pennsylvania.

Cai, T.T. and J. Lv (2007) "Discussion: The Dantzig Selector: Statistical Estimation When p is much larger than n." *Annals of Statistics* 35(6): 2365–2369.

Camacho, R., King, R., and A. Srinivasan (2006) "14th International Conference on Inductive Logic Programming." *Machine Learning* 64: 145–287.

Candes, E. and T. Tao. (2007) "The Dantzig Selector: Statistical Estimation When p is Much Larger than n" (with discussion). *Annals of Statistics* 35(6): 2313-2351.

Carlin, B.P. and T.A. Lewis (1996) *Bayes and Empirical Bayes Methods for Data Analysis*. New York: Chapman & Hall.

Chen, P., Lin, C., and B. Schölkopf (2004) " A Tutorial on ν-Support Vector Machines." Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan.

Chipman, H.A., George, E.I., and McCulloch, R.E. (1998) "Bayesian CART Model Search. *Journal of the American Statistical Association*

Christianini, N, and J. Shawe-Taylor. (2000) *Support Vector Machines*. Cambridge, UK: Cambridge University Press. 93(443): 935-948.

Cleveland, W. (1979) "Robust Locally Weighted Regression and Smoothing Scatterplots." *Journal of the American Statistical Association* 78: 829–836.

Cleveland, (W.) 1993) *Visualizing Data*. Summit, New Jersey: Hobart Press.

Committee on Reducing Porpoise Mortality from Tuna Fishing (1992) *Dolphins and the Tuna Industry*. Washington, DC, National Academy Press.

Cook, D.R. and S. Weisberg (1999) *Applied Regression Including Computing and Graphics*. New York: John Wiley and Sons.

Crawley, M.J. (2007) *The R Book*. New York: John Wiley and Sons.

Dalgaard, P. (2002) *Introductory Statistics with R*. New York: Springer.

Damšar, J. (2006) "Statistical Comparisons of Classifiers over Multiple Data Sets." *Journal of Machine Learning Research* 7: 1–30.

Danilov, D.L. and J.R. Magnus (2004) "On the Harm that Ignoring Pre-Testing Can Cause." *Journal of Econometrics* 122: 27–46.

Dasu, T. and T. Johnson (2003) *Exploratory Data Mining and Data Cleaning*. New York: John Wiley and Sons.

de Boors, C. (2001) *A Practical Guide to Splines*, revised edition. New York: Springer.

Deloncle, A., Berk, R.A., D'Andrea, F., and M. Ghil (2006) "Weather Regmine Prediction Using Statistical Learning." UCLA Department of Statistics Preprint # 435.

Drummond, C. and R.C. Holte (2006) "Cost Curves: An Improved Method for Visualizing Classifier Performance." *Machine Learning* 65 (1): 95–130.

Efron, B. (1981) "Nonparametric Standard Errors and Confidence Intervals." *Canadian Journal of Statistics* 9: 139–172.

Efron, B. (1983) "Estimating the Error Rate of a Prediction Rule: Improvement on Cross-Validation." *Journal of the American Statistical Association* 78: 892-898.

Efron, B. (1987) "Better Bootstrap Confidence Intervals." *Journal of the American Statistical Association* 82: 171–185.

Efron, B. (2004) "The Estimation of Prediction Error: Covariance Penalties and Cross-Validation" (with discussion). *Journal of the American Statistical Association* 99: 619–642.

Efron, B. and R. Tibshirani (1993) *Introduction to the Bootstrap*. New York: Chapman & Hall.

Efron, B. and R. Tibshirani (1997) "Improvements on Cross-Validation: the 632+ Bootstrap Method," *Journal of the American Statistical Association* 92: 548–560.

Efron, B., Hastie, T., Johnstone, I., and R. Tibshirani (2004) "Least Angle Regression" (with discussion). *The Annals of Statistics* 32(2): 407–499.

Efron, B., Hastie, T., and R. Tibshirani (2007) "Discussion: The Dantzig Selector: Statistical Estimation When p is much larger than n." *Annals of Statistics* 35(6): 2358–2364.

Eibl, G. and K. Pfeiffer (2005) "Multiclass Boosting for Weak Classifiers." *Journal of Machine Learning Research* 6: 189–210.

Fan, J. and I. Gijbels. (1996) *Local Polynomial Modeling and its Applications*. New York: Chapman & Hall.

Fan, J. and Li. R. (2006) "Statistical Challenges with Dimensionality: Feature Selection in Knowledge Discovery." In M. Sanz-Sole, J. Soria, J.L. Varona, J. Verdera (eds.), Proceedings of the International Congress of Mathematicians Vol. III, 595–622.

Fan G., and B. Gray (2005) "Regression Tree Analysis Using TARGET." *Journal of Computational and Graphical Statistics* 14: 206–218.

Faraway, J. (2004) "Human Animation Using Nonparametric Regression." *Journal of Computational and Graphical Statistics* 13: 537–553.

Freedman, D.A. (1987) "As Others See Us: A Case Study in Path Analysis" (with discussion). *Journal of Educational Statistics* 12: 101–223.

Freedman, D.A. (2004) "Graphical Models for Causation and the Identification Problem." *Evaluation Review* 28: 267–293.

Freedman, D.A. (2005) *Statistical Models* Cambridge, UK: Cambridge University Press.

Freedman, D.A. and R. Berk (2008) "Weighting Regressions by Propensity Scores." *Evaluation Review*, forthcoming.

Freedman, D.A., Navidi, W., and S.C. Peters (1988) "On the Impact of Variable Selection in Fitting Regression Equations." In T.K. Dijkstra (ed.), *On Model Uncertainty and Its Statistical Implications*, 1–16, Springer, Berlin

Freund, Y., and R. Schapire. (1996) "Experiments with a New Boosting Algorithm. *Macine Learning: Proceedings for the Thirteenth International Conference* 148–156. Morgan Kaufmann, San Francisco.

Freund, Y., and R.E. Schapire (1999) "A Short Introduction to Boosting." *Journal of the Japanese Society for Artificial Intelligence* 14: 771–780.

Friedman, J.H. (1991) "Multivariate Adaptive Regression Splines (with discussion)," *The Annals of Statistics* 19: 1–82.

Friedman, J. H. (2001) "Greedy Function Approximation: A Gradient Boosting Machine," *The Annals of Statistics* 29: 1189–1232

Friedman, J.H. (2002) "Stochastic Gradient Boosting," *Computational Statistics and Data Analysis* 38: 367–378.

Friedman, J.H. and P. Hall (2000) "On Bagging and Non-Linear Estimation." *Technical Report*, Department of Statistics, Stanford University.

Friedman, J.H., Hastie, T., and R. Tibshirani (2000). "Additive Logistic Regression: A Statistical View of Boosting" (with discussion). *Annals of Statistics* 28: 337–407.

Friedman, J.H., Hastie, T., Rosset S., Tibshirani, R., and J. Zhu (2004) "Discussion of Boosting Papers," *Annals of Statistics* 32: 102–107.

Gareth, M. and P. Radchenko (2007) "Sparse Generalized Linear Models." Working Paper, Department of Statistics, Marshall School of Business, University of California.

Gareth, M. and J. Zhu (2007) "Funtional Linear Regression That's Interpretable." Working Paper, Department of Statistics, Marshall School of Business, University of California.

Geurts, P., Ernst, D., and L. Wehenkel (2006) "Extemely Randomized Trees." *Machine Learning* 63 (1): 3–42.

Gifi, A. (1990) *Nonlinear Multivariate Analysis*. New York: John Wiley and Sons.

Goldenshluger, A. and A. Tsybakov (2001) "Adaptive Prediction and Estimation in Linear Regression with Infinitely Many Parameters." *Annals of Statistics* 29(6): 1601–1619.

Grandvalet, Y. (2004) "Bagging Equalizes Influence." *Machine Learning* 55: 251–270.

Granger, C.W.J. and P. Newbold, (1986) *Forecasting Economic Time Series*. New York: Academic Press.

Green, P.J. and B.W. Silverman (1994) *Nonparametric regression and Generalized Linear Models*. New York: Chapman & Hall.

Green, W.H. (2003) *Econometric Analysis*, fifth edition. New York: Prentice Hall.

Hand, D., Manilla, H., and P. Smyth (2001) *Principles of Data Mining*. Cambridge, MA: MIT Press.

Hastie, T., and J. Zhu (2006) "Comment." *Statistical Science* 21(3): 352–356.

Hastie, T., Tibshirani, R. and J. Friedman (2001) *The Elements of Statistical Learning*. New York: Springer-Verlag.

Hastie, T.J. and R.J. Tibshirani. (1990) *Generalized Additive Models*. New York: Chapman & Hall.

Hastie, T.J. and R.J. Tibshirani. (1996) "Discriminant Adaptive Nearest Neighbor Classification." *IEEE Pattern Recognition and Machine Intelligence* 18: 607–616.

He, Y. (2006) "Missing Data Imputation For Tree-Based Models." PhD dissertation for the Department of Statistics, UCLA.

Heckman, J. (1999) "Causal Parameters and Policy Analysis in Economics: A Twentieth Century Retrospective." *The Quarterly Journal of Economics*, February: 45–97.

Heckman, J, Ichimura, H., and P. Todd (1998) "Matching as an Economic Evaluation Estimator," *Review of Economic Studies* 65: 261–294.

Hoeting, J., Madigan, D., Raftery, A., and C. Volinsky (1999) "Bayesian Model Averaging: A Practical tutorial." *Statistical Science* 14: 382–401.

Horváth, T. and A. Yamamoto (2006) "International Conference on Inductive Logic Programming." *Journal of Machine Learning* 64:3–144.

Hothorn, T., Berthold, L., Brenner, A., and M. Radespiel-Tröger. (2002) "Bagging Survival Trees." Department of Medical Informatics, Biometry and Epidemiology, Freidrich-Alexander University Erlangen-Nuremberg, preprint.

Hothorn, T., and B. Lausen (2003) "Double-Bagging: Combining Classifiers by Bootstrap Aggregation." *Pattern Recognition* 36: 1303–1309.

Hothorn, T., Hornik, K., and A. Zeileis (2006) "Unbiased Recursive Partitioning: A Conditional Inference Framework." *Journal of Computational and Graphical Statistics* 15(3): 651–674.

Hurvich, C.M., and C. Tsai (1989) "Regression and Time Series Model Selection in Small Samples." *Biometrika* 76(2): 297–307.

Hsu, C., Chung, C., and C. Lin (2007) "A Practical Guide to Support Vector Classification," Department of Computer Science and Information Engineering, National Taiwan University, Taipsi, Taiwan. http://www.csie.ntu.edu.tw/ cjlin/libsvm.

Imbens, G. (2004) "Nonparametric Estimation of Average Treatment Effects Under Exogeneity: A Review." *The Review of Economics and Statistics*: 86: 4–29.

Jiang, W. (2004) "Process Consistency for Adaboost." *Annals of Statistics* 32: 13–29.

Joachims, T (1998) "Making Large-Scale SVM Learning Practical," in B. Schölkopf, C.J.C. Burges, and A.J. Smola (Eds.) *Advances in Kernel Methods — Support Vector Learning* Cambridge, MA: MIT Press.

Krieger, A., Long, C., and A. Wyner (2001) "Boosting Noisy Data." *Prroceedings of the International Conference on Machine Learning*, Amsterdam: Mogan Kauffman.

Kriegler, B. (2007) "Boosting the Quantile Distribution: A Cost-Sensitive Statistical Learning Procedure." Department of Statistics, UCLA, working paper.

Lafferty, J. and L. Wasserman (2008) "Rodeo: Sparse Greedy Nonparametric Regression." *Annals of Statistics* 36(1): 28-63.

Leamer, E.E. (1978) *Specification Searches: Ad Hoc Inference with Non-Experimental Data*. New York, John Wiley.

LeBlanc, M. and R. Tibshirani (1996) "Combining Estimates on Regression and Classification." *Journal of the American Statistical Association* 91: 1641–1650.

Leeb, H. and B.M. Pötscher 2006) "Can One Estimate the Conditional Distribution of Post-Model-Selection Estimators?" *The Annals of Statistics* 34(5): 2554–2591.

Lennert-Cody, C.E. and R.A. Berk (2005) "Statistical Learning Procedures for Monitoring Regulatory Compliance: An Application to Fisheries Data. UCLA Department of Statistics Preprint Series #426.

Li, Y., Lie, Y., and J. Zhu (2007) "Quantile Regression in Reproducing Kernel Hilbert Spaces." *Journal of the American Statistical Association* 102: 255–268.

Lin, Y. and Y. Jeon (2006) "Random Forests and Adaptive Nearest Neighbors." *Journal of the American Statistical Association* 101: 578–590.

Lipton, P. (2005) "Testing Hypotheses: Prediction and Prejudice." *Science* 307: 219–221.

Little, R. and D. Rubin (2002) *Statistical Analysis with Missing Data*, second edition. New York: John Wiley.

Loader, C. (1999) *Local Regression and Likelihood*. New York: Springer-Verlag.

Loader, C. (2004) "Smoothing: Local Regression Techniques," in J. Gentle, W. Hardle, and Y. Mori, *Handbook of Computational Statistics*. NewYork: Springer-Verlag.

Loh, W.-Y. (2002) "Regression Trees with Unbiased Variable Selection and Interaction Detection." *Statistica Sinica* 12: 361–386.

Lugosi, G. and N. Vayatis (2004) "On Bayes Risk Consistency of Regularized Boosting Methods." *Annals of Statistics* 32(1): 30–55.

Luo, X, Stefanski, L.A., and D.B. Boos (2006) "Tuning Variable Selection Procedures by Adding Noise." *Technometrics* 48(2): 165–175.

Lutz, R.W. and P. Büehlmann (2006) "Conjugate Direction Boosting." *Journal of Computational and Graphical Statistics* 15: 287–311.

Maindonald, J. and J. Braun (2007) *Data Analysis and Graphics Using R*, second edition. Cambridge, UK: Cambridge University Press.

Madigan, D., Raftery, A.E., Volinsky, C., and J. Hoeting (1996) "Bayesian Model Averaging." In *AAAI Workshop on Integrating Multiple Learned Models*: 77–83, AAAI Press.

Manly, B.F.J. (1997) *Randomization, Bootstrap and Monte Carlo Methods in Biology*. New York: Chapman & Hall.

Mannor, S., Meir, R. and T. Zhang (2002) "The Consistency of Greedy Algorithms for Classification." In J. Kivensen and R.H. Sloan (Eds.), COLT 2002, LNAI 2375: 319–333.

Manski, C.F. (1990) "Nonparametric Bounds on Treatment Effects." *American Economic Review Papers and Proceedings* 80: 319–323.

McCaffrey, D., Ridgeway, G., and A. Morral (2004) "Propensity Score Estimation with Boosted Regression for Evaluating Adolescent Substance Abuse Treatment." *Psychological Methods* 9: 403–425.

McCullagh, P. and J.A. Nelder (1989) *Generalized Linear Models*, second edition. New York: Chapman and Hall.

Mease, D. and A.J. Wyner (2008) "Evidence Contrary to the Statistical View of Boosting." *Journal of Machine Learning* 9: 1-26

Mease, D., Wyner, A.J. and A. Buja (2007) "Boosted Classification Trees and Class Probability/Quantile Estimation." *Journal of Machine Learning* 8: 409–439.

Meinshausen, N. (2006) "Quantile Regression Forests." *Journal of Machine Learning Research* 7: 983–999.

Meinshausen, N. and P. Bühlmann (2006) "High Dimensional Graphs and Variable Selection with the Lasso." *The Annals of Statistics* 34(3) 1436–1462.

Meinshausen, N., Rocha, G., B. Yu. (2007) "Discussion: The Dantzig Selector: Statistical Estimation When p is much larger than n." *Annals of Statistics* 35(6): 2373–2384.

Mocan, H.N. and K. Gittings (2003) "Getting off Death Row: Commuted Sentences and the Deterrent Effect of Capital Punishment." *Journal of Law and Economics* 46: 453-478.

Moguerza, J.M. and A. Munõz (2006) "Support Vector Machines with Applications." *Statistical Science* 21(3): 322–336.

Mojirsheibani, M. (1997) "A Consistent Combined Classification Rule." *Statistics & Probability Letters* 36: 411-419.

Mojirsheibani, M. (1999) "Combining Classifiers vis Discretization." *Journal of the American Statistical Association* 94: 600-609.

Murrell, P. (2006) *R Graphics*. New York: Chapman & Hall/CRC.

Peña, D. (2005) "A New Statistic for Influence in Linear Regression." *Technometrics* 47: 1–12.

Polzehl, J. and S. Spokoiny (2000) "Adaptive Weights Smoothing with Applications to Image Restoration." *Journal of the Royal Statistical Association*, Soc. B, 62, Part 2: 335–354.

Polzehl, J. and S. Spokoiny (2002). "Varying Coefficient Regression Modeling by Adaptive Weights Smoothing." Manuscript.

Quinlan, R. (1993) *Programs in Machine Learning*. San Mateo, CA: Morgan Kaufman.

Raftery, A.D. (1995) "Bayesian Model Selection in Social Research." *Sociological Methodology* 25: 111–163.

Reunanen, J. (2003) "Overfitting in Making Comparisons between Variable Selection Methods." *Journal of Machine Learning Research* 3: 1371–1382.

Ridgeway, G. (1999) "The State of Boosting." *Computing Science and Statistics*, 31: 172–181.

Ridgeway, G. (2005) "Generalized Boosted Models: A Guide to the gbm Package." Available at http: //i-pensieri.com/gregr/papers/gbm-vignette.pdf

Ripley, B.D., (1996) *Pattern Recognition and Neural Networks*. Cambridge, UK: Cambridge University Press.

Rosenbaum, P.R. (2002) *Observational Studies*. second edition. New York: Springer-Verlag.

Rosset, S. and J. Zhu (2007) "Piecewise Linear Regularized Solution Paths." *The Annals of Statistics* 35 (3): 1012–1030.

Rubin, D. B. (1986) "Which Ifs Have Causal Answers." *Journal of the American Statistical Association* 81: 961–962.

Ruppert, D. (1997) "Empirical-Bias Bandwidths for Local Polynomial Nonparametric Regression and Density Estimation." *Journal of the American Statistical Association* 92: 1049–1062.

Ruppert, D. and M.P. Wand (1994) "Multivariate Locally Weighted Least Squares Regression." *Annals of Statistics* 22: 1346–1370.

Ruppert, D., Wand, M.P. and R.J Carroll (2003) *Semiparametric Regression*. Cambridge, UK: Cambridge University Press.

Schwartz, G. (1978) "Estimating the Dimension of a Model." *Annals of Statistics* 6: 461–464.

Segal, M.R. (2003) "Machine Learning Benchmarks and Random Forest Regression." Working paper, Department of Biostatistics University of California San Francisco (http://www.ics.uci.edu/ mlearn/MLRepository.html).

Shakhnarovich, G. (Ed.) (2006) *Nearest-Neighbor Methods in Learning and Vision: Theory and Practice*. Cambridge, MA: MIT Press.

Shapire, R.E. (1999) "A Brief Introduction to Boosting." *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*.

Shapire, R.E. (2002) "The Boosting Approach to Machine Learning: An Overview." *MSRI Workshop on Non-Linear Estimation and Classification*.

Simonoff, J.S., and C. Tsai (1999) "Semiparametric and Additive Model Selection Using an Improved Akaike Information Criterion." *Journal of Computational and Graphical Statistics* 8(1): 22–40.

Stein, C. (1956) "Inadmissability of the Usual Estimator for the Mean of Multivariate Normal Distribution." *Proceedings of the 3rd Berkeley Symposium* 1: 197–206.

Strobl, C., Bouestreix, A., Zeileis, A., and T. Hothorn (2007) "Bias in Random Forest Variable Importance Measures: Illustrations, Sources, and a Solution." *Bioinformatics* 8(25).

Su, X., Wang, M., and J. Fan (2004) "Maximum Likelihood Regression Trees." *Journal of Computational and Graphical Statistics* 13: 586–598.

Sutton, R.S. and A.G. Barto. (1999) *Reinforcement Learning*. Cambridge, MA: MIT Press.

Svetnik, V., Liaw, A., and C.Tong. (2003) "Variable Selection in Random Forest with Application to Quantitative Structure-Activity Relationship." Working paper, Biometrics Research Group, Merck & Co., Inc.

Therneau, T.M. and E.J. Atkinson (1997) "An Introduction to Recursive Partitioning Using the RPART Routines." Technical Report # 61, Mayo Foundation.

Thompson, S.K. (2002) *Sampling*, second edition. New York: John Wiley.

Tibshirani, R.J. (1996) "Regression Shrinkage and Selection Via the LASSO." *Journal of the Royal Statistical Society*, Series B, 25: 267–288.

Traskin, M. (2008) "The Role of Bootstrap Sample Size in the Consistency of the Random Forest Algorithm." Technical Report, Department of Statistics, University of Pennsylvania.

Vapnick, V. (1996) *The Nature of Statistical Learning Theory*. New York: Springer-Verlag.

Wahba, G. (2006) "Comment." *Statistical Science* 21(3): 347–351

Wang, H., Li, G., and F. Jiang (2007) "Robust Regression Shrinkage and Consistent Variable Selection Through the LAD-Lasso." *Journal of Business and Economic Statistics* 25(3): 347–355.

Witten, I.H. and E. Frank. (2000). *Data Mining*. New York: Morgan and Kaufmann.

Wood, S.N. (2000) "Modelling and Smoothing Parameter Estimation with Multiple Quadratic Penalties." *Journal of the Royal Statistical Society*, B 62(2):413–428.

Wood, S.N. (2003) "Thin Plate Regression Splines." *Journal of the Royal Statistical Society* B 65(1):95–114.

Wood, S.N. (2004) "Stable and Eficient Multiple Smoothing Parameter Estimation for Generalized Additive Models." *Journal of the American Statistical Association*, Vol. 99: 673–686.

Wood, S.N. (2006) *Generalized Additive Models* New York: Chapman & Hall.

Wu, Y., Tjelmeland, H., and M. West (2007) "Bayesian CART: Prior Specification and Posterior Simulation." *Journal of Computational and Graphical Statistics* 16(1): 44–66.

Wyner, A.J. (2003) "Boosting and Exponential Loss." In C.M. Bishop and B.J. Frey (Eds.) *Proceedings of the Ninth Annual Conference on AI and Statistics* Jan 3–6, Key West, Florida.

Xu, M. and M.W. Golay (2006) "Data-Guided Model Combination by Decomposition and Aggregation." *Machine Learning* 63 (1): 43–67.

Zellner, A., Keuzenkamp, H.A., and M. McAleer (Eds.) (2001) *Simplicity, Inference and Modelling*. Cambridge, UK: Cambridge University Press.

Zhang, C. (2005) "General Empirical Bayes Wavelet Methods and Exactly Adaptive Minimax Estimation." *The Annals of Statistics* 33(1): 54–100.

Zhang, H. and B. Singer (1999) *Recursive Partitioning in the Health Sciences*. New York: Springer-Verlag.

Zhang, T. and B. Yu (2005) "Boosting with Early Stopping: Convergence and Consistency." *Annals of Statistics*, 33(4): 1538–1579.

Zou, H. (2006) "The Adaptive Lasso and Its Oracle Properties." *The Journal of the American Statistical Association* 101 (467) 1418–1429.

Zou, H. and T. Hastie (2005) "Regularization and Variable Selection via Elastic Net." *Journal of the Royal Statistical Association*, series B, 67, Part 2: 301–320.

Zou, H. and Hastie, T., and R. Tibshirani (2005) "Space Principal Component Analysis." *Journal of Computational and Graphical Statistics* 15: 265–286.

Index

(continued from page ii)

Iacus: Simulation and Inference for Stochastic Differential Equations: With R Examples
Ibrahim/Chen/Sinha: Bayesian Survival Analysis
Jiang: Linear and Generalized Linear Mixed Models and Their Applications
Jolliffe: Principal Component Analysis, 2nd edition
Konishi/Kitagawa: Information Criteria and Statistical Modeling
Knottnerus: Sample Survey Theory: Some Pythagorean Perspectives
Kosorok: Introduction to Empirical Processes and Semiparametric Inference
Küchler/Sørensen: Exponential Families of Stochastic Processes
Kutoyants: Statistical Inference for Ergodic Diffusion Processes
Lahiri: Resampling Methods for Dependent Data
Lavallée: Indirect Sampling
Le Cam: Asymptotic Methods in Statistical Decision Theory
Le Cam/Yang: Asymptotics in Statistics: Some Basic Concepts, 2nd edition
Le/Zidek: Statistical Analysis of Environmental Space-Time Processes
Liese/Miescke: Statistical Decision Theory: Estimation, Testing, Selection
Liu: Monte Carlo Strategies in Scientific Computing
Manski: Partial Identification of Probability Distributions
Mielke/Berry: Permutation Methods: A Distance Function Approach, 2nd edition
Molenberghs/Verbeke: Models for Discrete Longitudinal Data
Morris/Tibshirani: The Science of Bradley Efron, Selected Papers
Mukerjee/Wu: A Modern Theory of Factorial Designs
Nelsen: An Introduction to Copulas, 2nd edition
Pan/Fang: Growth Curve Models and Statistical Diagnostics
Politis/Romano/Wolf: Subsampling
Ramsay/Silverman: Applied Functional Data Analysis: Methods and Case Studies
Ramsay/Silverman: Functional Data Analysis, 2nd edition
Reinsel: Elements of Multivariate Time Series Analysis, 2nd edition
Rosenbaum: Observational Studies, 2nd edition
Rosenblatt: Gaussian and Non-Gaussian Linear Time Series and Random Fields
Särndal/Swensson/Wretman: Model Assisted Survey Sampling
Santner/Williams/Notz: The Design and Analysis of Computer Experiments
Schervish: Theory of Statistics
Shaked/Shanthikumar: Stochastic Orders
Shao/Tu: The Jackknife and Bootstrap
Simonoff: Smoothing Methods in Statistics
Song: Correlated Data Analysis: Modeling, Analytics, and Applications
Sprott: Statistical Inference in Science
Stein: Interpolation of Spatial Data: Some Theory for Kriging
Taniguchi/Kakizawa: Asymptotic Theory for Statistical Inference for Time Series
Tanner: Tools for Statistical Inference: Methods for the Exploration of Posterior Distributions and Likelihood Functions, 3rd edition
Tillé: Sampling Algorithms
Tsaitis: Semiparametric Theory and Missing Data
van der Laan/Robins: Unified Methods for Censored Longitudinal Data and Causality
van der Vaart/Wellner: Weak Convergence and Empirical Processes: With Applications to Statistics
Verbeke/Molenberghs: Linear Mixed Models for Longitudinal Data
Weerahandi: Exact Statistical Methods for Data Analysis

Springer
the language of science

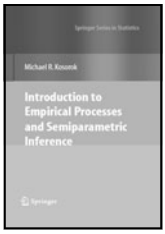
springer.com

Time Series Analysis with Applications in R

Johnathon D. Cryer and Kung-Sik Chan

Time Series Analysis With Applications in R, Second Edition, presents an accessible approach to understanding time series models and their applications. Although the emphasis is on time domain ARIMA models and their analysis, the new edition devotes two chapters to the frequency domain and three to time series regression models, models for heteroscedasticty, and threshold models. All of the ideas and methods are illustrated with both real and simulated data sets. A unique feature of this edition is its integration with the R computing environment.

2008. 2nd Ed., 494p. (Springer Texts in Statistics) Hardcover
ISBN 0-387-75958-6

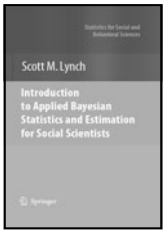


Introduction to Empirical Processes and Semiparametric Inference

Michael R. Kosorok

This book provides a self-contained, linear, and unified introduction to empirical processes and semiparametric inference. These powerful research techniques are surprisingly useful for developing methods of statistical inference for complex models and in understanding the properties of such methods. The targeted audience includes those who have an interest in learning about and doing research in empirical processes and semiparametric inference but who would like to have a friendly and gradual introduction to the area.

2008. 476 pp. (Springer Series in Statistics) Hardcover
ISBN 978-0-387-74977-8



Introduction to Applied Bayesian Statistics and Estimation for Social Scientists

Scott M. Lynch

This book covers the complete process of Bayesian statistical analysis in great detail from the development of a model through the process of making statistical inference. The key feature of this book is that it covers models that are most commonly used in social science research, including the linear regression model, generalized linear models, hierarchical models, and multivariate regression models, and it thoroughly develops each real-data example in painstaking detail.

2007. 357 pp. (Statistics for Social and Behavioral Sciences) Hardcover
ISBN 978-0-387-71264-2

Easy Ways to Order► Call: Toll-Free 1-800-SPRINGER ▪ E-mail: orders-ny@springer.com ▪ Write: Springer, Dept. S8113, PO Box 2485, Secaucus, NJ 07096-2485 ▪ Visit: Your local scientific bookstore or urge your librarian to order.