# Data Mining und Information Retrieval

**Skript zur Vorlesung an der TH Darmstadt im Wintersemester 1995/96**

**Reginald Ferber**

# Vorbemerkung zur Vorlesung Data Mining und Information Retrieval

Datenbanken enthalten große Mengen an Informationen, die Nutzenden durch Information Retrieval Methoden zugänglich gemacht werden müssen. Während diese Methoden einen durch eine Anfrage (Query) ausgedrückten Informationsbedarf mit Datensätzen der Datenbank bedienen, die mehr oder weniger den tatsächlichen Informationsbedarf der Anfragenden befriedigen, bleiben Informationen über das Verhältnis der Dokumente untereinander meistens weitgehend unzugänglich.

Ziel von Knowledge Discovery oder Data Mining Methoden ist es, Abhängigkeiten und Regelmäßigkeiten zwischen Datenbankeinträgen zu finden und in eine für Nutzende verständliche Form zu bringen. Dabei ergeben sich eine Reihe von Problemen: Aus der ungeheuren Menge möglicher Beziehungen zwischen Datensätzen müssen solche herausgesucht werden, die eine (für Nutzende) sinnvolle, genaue, effektive und einsichtige Unterscheidung bzw. Kategorisierung erlauben. Dabei muß berücksichtigt werden, daß gerade große Datenbanken oft unklare, fehlende oder widersprüchliche Informationen enthalten. Bei Text-, Bild- und Multimediadaten müssen darüber hinaus geeignete Featuredetektoren ausgewählt bzw. konstruiert werden.

In der Vorlesung werden nach einer Einführung in das Thema verschiedene Lösungsansätze vorgestellt. Dazu gehören solche, die sich an Methoden aus dem Bereich Machine Learning orientieren, aber auch probabilistische, korpusbasierte und Feature Selection Methoden.

Die Vorlesung ist in vier Teile untergliedert:

- In der Einführung (Teil 1) werden die Aufgaben beschrieben, die zu lösen sind, und es werden Anforderungen vorgestellt, die dabei erfüllt werden müssen.
- Der Teil 2 stellt zunächst die “klassischen” Modelle im Information Retrieval vor. In den Kapiteln 2.2 und 2.3 werden dann auch neuere Entwicklungen besprochen (die teilweise in Teil 4 wieder aufgenommen werden).
- Teil 3 behandelt Data-Mining bzw. Knowledge Discovery Methoden. Dabei werden vor allem Verfahren vorgestellt, die auf streng strukturierten Daten, in der Regel also Faktendatenbanken, arbeiten und oft aus dem Bereich Machine Learning stammen.
- Im 4. Teil wird die Fragestellung des Data-Mining bzw. Knowledge Discovery auf Information Retrieval Systeme übertragen. Dabei zeigt sich, daß einige ältere und neuere Forschungsansätze auch unter dieser Perspektive gesehen und beschrieben werden können.

Knowledge Discovery in Databases (KDD) ist ein sehr junges Forschungsgebiet, zu dem verschiedene Disziplinen beitragen. Entsprechend ist nicht nur das Vokabular uneinheitlich (und meistens englisch bzw. amerikanisch), sondern es gibt bisher (meines Wissens) auch kein zusammenfassendes Lehrbuch zum Thema. Die Grundlagen für diese Vorlesung sind entsprechend vielfältig: Lehrbücher zum Thema Information Retrieval: Van Rijsbergen (1979, liegt in einer HTML-Version im Netz), Salton and McGill (1983), Salton (1988), Frakes and Baeza-Yates (1992)) und das Skript Fuhr (1995), zum KDD die Skripte: Holsheimer und Siebes 1994, Wüthrich 1994, vor allem aber Originalbeiträge aus Fachzeitschriften, Tagungsbänden und Technical Reports.

Entsprechend ist auch die Auswahl der Themen der Vorlesung: Neben den Grundlagen der jeweiligen Gebiete werden viele einzelne Implementierungen und experimentelle Ergebnisse vorgestellt.

Viele Ergebnisse und Softwarepakete zu den Themen Knowledge Discovery in Databases und Machine Learning werden über das WWW verbreitet. Ein guter Einstiegspunkt ist die “Knowlede Discovery Mine“ von Piatetsky-Shapiro (html).

## Literaturverzeichnis


Frakes, W. B., & Baeza-Yates, R., Eds. (1992). *Information Retrieval Data Structures & Algorithms*. Prentice Hall, Englewood Cliffs, New Jersey.

Fuhr, N. (1995). Information retrieval. Skriptum zur Vorlesung im SS 93, Universität Dortmund, http://ls6–www.informatik.uni-dortmund.de/teaching/courses/ir.

Holsheimer, M., & Siebes, A. P. J. M. (1994). Data mining: the search for knowledge in databases. Technical report, Centrum voor Wiskunde en Informatica, ftp://ftp.cwi.nl/pub/CWIreports/AA/CS-R9406.ps.Z.

Piatetsky-Shapiro, G. Knowledge discovery mine. WWW-HTML Page: http://info.gte.com/~kdd/index.html.

Salton, G., Ed. (1988). *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*. Reading, Massachusetts: Addison-Wesley.

Salton, G., & McGill, M. J. (1983). *Introduction to Modern Information Retrieval*. New York: McGraw-Hill.

Van Rijsbergen, C. J. (1979). *Information Retrieval*. London: Butterworth, als HTML-Version im WWW: http://www.dcs.glasgow.ac.uk/Keith/Preface.html.

Wüthrich, B. (1994). Knowledge discovery in databases. Electronically available manuscript, The Hong Kong University of Science and Technology, ftp://ftp.cs.ust.hk/pub/techreport/postscript/tr94–2.ps.gz.

# Inhalt

**Vorbemerkung zur Vorlesung Data Mining und Information Retrieval . . . . . i**
**Literaturverzeichnis . . . . . ii**

**Teil 1 Einführung . . . . . 1**
**Literaturverzeichnis . . . . . 9**

**Teil 2 Wissensrepräsentation und Retrievalmodelle . 10**
**2.1 Bool'sches Retrieval . . . . . 12**
2.1.1 Logik des Bool'schen Retrievals . . . . . 12
2.1.2 Implementation mit einer invertierten Liste . . . . . 13
**2.2 Linguistische Verfahren der Repräsentation, Thesauri . . 15**
2.2.1 Wortorientierte Verfahren . . . . . 16
2.2.2 Allgemeinere Wissenstrukturen . . . . . 21
2.2.3 Klassifikationen . . . . . 21
2.2.4 Thesauri . . . . . 22
**2.3 Vektorraummodelle . . . . . 25**
2.3.1 Das Modell . . . . . 25
2.3.2 Vektorraummodell und Bool'sches Retrieval . . . . . 25
2.3.3 Ähnlichkeitsfunktionen . . . . . 27
2.3.4 Gewichtungsmethoden . . . . . 30
**2.4 Evaluationsverfahren . . . . . 32**
2.4.1 Einflußfaktoren . . . . . 33
2.4.2 Relevanz . . . . . 34
2.4.3 Precision und Recall . . . . . 35
2.4.4 Mittelwertbildungen . . . . . 37
2.4.5 Testkollektionen . . . . . 38
**Literaturverzeichnis . . . . . 39**

**Teil 3 Data Mining . . . . . 42**
**3.1 Lernen . . . . . 42**
3.1.1 Induktives Lernen . . . . . 44
**3.2 Automatisiertes Lernen . . . . . 46**
3.2.1 Faktendatenbanken . . . . . 46
3.2.2 Kategorisieren . . . . . 46
3.2.3 Überwachtes Lernen . . . . . 48
3.2.4 Nichtüberwachtes Lernen . . . . . 48
3.2.5 ID3 (Quinlan) . . . . . 49
3.2.6 Endlichkeit des ID3 auf konsistenten Trainingssets . . . . . 52
3.2.7 Wertebereiche . . . . . 53

3.2.8 Inkonsistente Trainingssets . . . 55
3.2.9 Unvollständige Beispiele . . . 55
3.2.10 Größe des Trainingssets . . . 56
3.2.11 Fenstertechnik . . . 56
3.2.12 Inkrementelles Lernen . . . 57
3.2.13 Overfitting . . . 57
3.2.14 Bewertung . . . 57
3.2.15 Darstellung als Regeln . . . 59
3.2.16 Mächtigere Repräsentationen . . . 64
3.2.17 Top-Down und Bottom-Up Methoden . . . 65
3.2.18 Nachoptimierung von Bäumen . . . 66
3.2.19 Suchstrategien . . . 66
3.2.20 Überlappende Einteilungen . . . 69
**3.3 Ausgewählte KDD Verfahren . . . 69**
3.3.1 AQ . . . 69
3.3.2 Externe Wissensstrukturen . . . 75
3.3.3 DBlearn (Han, Cai & Cercone 1992) . . . 76
3.3.4 Assoziative Regeln (Klemettinen et al 1994) . . . 80
**3.4 Fuzzy Sets . . . 82**
**Literaturverzeichnis . . . 88**
**Teil 4 Knowledge Discovery und IR Systeme . . . 90**
**4.1 Modelle des Information Retrieval . . . 90**
4.1.1 Probabilistisches Modell . . . 90
4.1.2 Logische Modelle des Information Retrieval . . . 98
**4.2 Korpusbasierte Verfahren . . . 104**
4.2.1 Der assoziative Ansatz . . . 104
4.2.2 Kookurrenzverfahren . . . 106
4.2.3 Latent Semantic Indexing . . . 112
**4.3 Schlußbemerkung . . . 112**
**Literaturverzeichnis . . . 113**
**Stichwortverzeichnis . . . 115**

# Teil 1 Einführung

In diesem Teil werden die Problemstellungen, die in der Vorlesung behandelt werden, vorgestellt.

## 1.1 Information Retrieval

Das klassische Feld des Information Retrieval sind Literaturdatenbanken. Dabei geht es darum, zu einer (wissenschaftlichen) Fragestellung die Literatur zu finden, die sich mit dem Thema befaßt. Es gibt eine ganze Reihe von Herangehensweisen, z. B:

- jemanden fragen, der/die sich auskennt
- ein Buch suchen, das eine Einführung in das Gebiet gibt, aus dem die Fragestellung stammt
- sich über Literaturverweise weiterhangeln
- in einer nach Sachgruppen geordneten Abstractsammlung nachsehen
- in einer Literaturdatenbank suchen

Dabei stößt man auf spezifische Probleme:

- Man muß jemanden finden, die/der sich auskennt und Zeit und Lust hat, sich mit der Frage zu beschäftigen. Dabei besteht die Gefahr, daß das Problem nur in der Sichtweise der Befragten angegangen wird.
- Bücher erscheinen häufig erst, wenn ein neues Fachgebiet bzw. eine neue Sichtweise schon eine gewisse Zeit existiert. Sie sind langsam.
- Das Beschaffen von Literatur dauert häufig lange. Es ist oft schwer einzuschätzen, inwieweit ein angegebener Artikel eine Fragestellung abdeckt.
- Indices sind hierarchisch nach Sachgruppen aufgebaut. Ihre Verwendung setzt daher Kenntnisse über die Sachgruppenhierarchie (und damit das Gebiet der Fragestellung) voraus. Schwierige Suchprobleme zeichnen sich häufig gerade dadurch aus, daß die Fragestellungen nicht innerhalb einer etablierten Theorie bleiben. Sie sind damit auch schwer in die vorgegebene Sachgruppenhierarchie einzuordnen.
- Literaturdatenbanken sind meist teuer und noch komplex zu bedienen. Sie bieten Zugang über Sachgruppen, Stichwörter und Freitextsuche.

## 1.2 Beispiel einer Datenbankrecherche

Angenommen, wir suchen Literatur zum Stand der Forschung im Bereich Retrieval Systeme für Multimedia-Objekte, wobei uns insbesondere interessiert, wie Bilder behandelt werden.

Die Datenbank INSPEC besteht aus Dokumenten, die Artikel und Bücher beschreiben, indem sie neben bibliographischen Angaben eine Kurzzusammenfassung (Abstract), eine Einordnung in ein hierarchisches Indexsystem und Stichwörter enthalten. Ein Beispiel ist in Abbildung 1 dargestellt. Auf die Eingabe von (Kombinationen von) Wörtern liefert das (Bool'sche) Retrievalsystem von INSPEC die Dokumente, die die Wörter in der angegebenen Form enthalten (Genaueres siehe Kapitel 2.1).

Deshalb überlegen wir uns zunächst, welche Stichwörter das Problem besonders gut beschreiben. Das sollten Wörter sein, die spezifisch für die Fragestellung sind, aber dann doch wieder so allgemein, daß man annehmen kann, daß sie in jedem Artikel, der für uns wichtig ist, vorkommen.

Eine Anfrage mit den Stichwörtern “Retrieval Systems”, “Multimedia” und “Images” könnte z. B. so aussehen:

**Abbildung 1: Document aus der Literaturdatenbank INSPEC**

```
 INSPEC XXXXXXXXX
 Doc Type:      Journal Paper
 Title:         Images database management system: a 'server-client
                producer' system on a local network and on the Internet
 Authors:       Ageron, P.; Besson, F.; Desfarges, P.
 Affiliation:   Univ. Lumiere, Lyon, France
 Journal:       Computer Networks and ISDN Systems
                Vol: 26  Iss: suppl., no.2-3  p. S101-6
                Date: 1994
                Country of Publication: Netherlands
                ISSN: 0169-7552  CODEN: CNISE9
                CCC: 0169-7552/94/$07.00
 Language:      English
 Treatment:     Practical
 Abstract:      Lumiere University offers a full range of arts and social
   sciences. The research resources are very large and are specialized
   in politics, economics, finance, arts and archeology. As soon as the
   first network services were available, researchers tried to find
   information on the network about digitised photo management and
   everything about multimedia communication. The 'Images Database
   Management System' program tries to manage all information about
   digitised images: to give somebody access to, to modify and work on
   images and to export images into other information retrieval systems.
   This application program is based on a special definition of an
   'entity'. This is a physical entity (Photo CD) or a folder or
   directory. It may also be a book, if images are reproduced from a
   document. These entities are named CDphoto, folder or book while
...
   Network like WAIS, WWW, or GOPHER and hope to place this tool at the
   Internet's users' disposal. (0 Refs.)
 Classification: C6160S (Spatial and pictorial databases); C6130M
   (Multimedia); C5620L (Local area networks); C6150N (Distributed
   systems software); C5620W (Other computer networks)
 Thesaurus: Client-server systems; File servers; Internet; Local area
   networks; Multimedia communication; Multimedia computing; Visual
   databases
 Free Terms: Images database management system; Server-client producer
   system; Local network; Internet; Digitised photo management;
   Multimedia communication; Information retrieval systems; Photo CD;
   Text information; SUN servers; OS Macintosh; HyperCard; JPEG Format;
   Communication protocol; Unix directories; GIF format
```

`RETRIEVAL SYSTEMS and MULTIMEDIA and IMAGES`. Sie wird vom Retrieval System so interpretiert: Suche alle Dokumente, in denen die Zeichenketten “RETRIEVAL SYSTEMS”, “MULTIMEDIA” und “IMAGES” alle drei irgendwo im Text vorkommen. Dabei wird zwischen Groß- und Kleinbuchstaben nicht unterschieden, und das Leerzeichen zwischen RETRIEVAL und SYSTEMS kann auch ein anderer “white space” sein (also z. B. ein Zeilenumbruch, auch in Verbindung mit mehreren Leerzeichen)

Die Suche liefert für die Zeit von Januar bis Juni 1995 drei Einträge:

- Image Engine: an object-oriented multimedia database for storing, retrieving and sharing medical images and text
- Multimedia information retrieval using knowledge in encyclopedia texts
- Images database management system: a 'server-client producer' system on a local network and on the Internet (Siehe Abbildung 1)

Die entsprechenden Dokumente können nun angesehen, ausgewählt und ausgedruckt oder auf Diskette gespeichert werden.

Nun sind drei Dokumente nicht gerade viel. Wir können also versuchen, die Anfrage etwas allgemeiner zu machen. Wenn die folgende Anfrage verwendet wird:

`RETRIEVAL and MULTIMEDIA and IMAGE$`

erhalten wir immerhin 35 Dokumente für den selben Zeitraum ($ ist eine Wildcard, d.h. IMAGE$ bezeichnet alle Wörter, die mit der Zeichenkette IMAGE beginnen). Dabei finden sich Titel wie:

- PhotoFile: a digital library for image retrieval
- Spatial knowledge representation and retrieval in 3–D image databases
- Multimedia retrieval technology
- A WWW interface to the OMNIS/Myriad literature retrieval engine
- Problems of content-based retrieval in image databases

die auf den ersten Blick nicht weniger relevant aussehen als die oben genannten.

Weitere Verallgemeinerungen der Anfrage und ihre Treffermengen sind in Tabelle 2 angegeben.

**Abbildung 2: Anzahl der in INSPEC gefundenen Dokumente für die Zeit vom Januar bis Juni 1995**

| Anfrage | Anzahl der gefundenen Dokumente |
|---|---|
| RETRIEVAL SYSTEMS and MULTIMEDIA and IMAGES | 3 |
| RETRIEVAL SYSTEMS and MULTIMEDIA and IMAGE$ | 5 |
| RETRIEVAL and MULTIMEDIA and IMAGE$ | 35 |
| RETRIEVAL and MULTIMEDIA | 148 |
| RETRIEVAL or MULTIMEDIA | 2559 |
| RETRIEVAL or MULTIMEDIA or IMAGE$ | 9364 |

## 1.3 Fakten Retrieval

Während die Dokumente in Literaturdatenbanken zwar in verschiedene Felder strukturiert sind, innerhalb vieler dieser Felder aber freien Text beliebiger Länge enthalten, sind die Einträge in Faktendatenbanken i. a. stark strukturiert. D. h. sie bestehen (logisch) aus Tupeln von Werten vorgegebener Typen (im allgemeinen werden Faktendatenbanken als relationale Datenbanken implementiert). Abbildung 3 zeigt eine Beispieldatenbank aus Fuhr (1995 Kapitel 1 S. 6). Die starke Strukturierung erleichtert den Zugriff auf die Einträge und das Arbeiten mit ihnen, da die Typisierung das Format vorgibt und Vergleiche ermöglicht. Diese Stärke kommt vor allem zum Tragen, wenn exakte Anfragen gestellt werden, wie, um bei der Beispieldatenbank zu bleiben, “gibt es einen PC mit 80386 CPU, 80 MB Festplatte und einer Zugriffszeit von weniger als 30 ms” . Sobald die Anfragen allerdings vager werden, müssen die Werte noch interpretiert werden, wenn für Anfragende nützliche Ergebnisse erziehlt werden sollen. So kann man in der Anfrage: “Suche kostengünstigen PC mit mindestens 25 MHZ, 80 MB und maximaler Zugriffszeit von 25 ms” die einzelnen Vorgaben als Richtwerte auffassen und wird mit “B” oder “C” Anfragende glücklicher machen als mit “D”. (Das Beispeil ist von 1993!)

Neben vagen Anfragen können vor allem in Eigenschaften der Datenbankobjekte Gründe dafür liegen, daß Faktendatenbanken nur bedingt nützlich sind. So ist es in einer Faktendatenbank einfach festzustellen, daß ein Buch mit 200 Seiten mehr Seiten hat als eines mit 150 (weil der Datentyp Intervallskalenniveau hat, oder anders gesagt, weil man für zwei verschiedene reelle Zahlen immer sagen kann, welche größer ist). Dagegen ist es erheblich schwieriger zu sagen, ob ein Buch, das einen “umfassenden und fundierten Überblick” über ein Thema liefert, mehr Stoff enthält als eines, das

**Abbildung 3: Beispiel-Datenbank über PCs (aus Fuhr 1995)**

| | CPU | MEM | CLOCK-RATE | DISK-SIZE | ACC-TIME | PRICE |
|---|---|---|---|---|---|---|
| A | 80386 | 4 | 20 | 40 | 40 | 1500 |
| B | 80386 | 4 | 25 | 80 | 28 | 2000 |
| C | 80386 | 4 | 25 | 75 | 26 | 2000 |
| D | 80386 | 4 | 25 | 80 | 24 | 3000 |
| E | 80386 | 4 | 25 | 85 | 28 | 2500 |

*Die Werte der letzten 5 Spalten lassen sich als reelle Zahlen vergleichen (Intervallskalen), wobei die Interpretation, was besser ist, von den Attributen die die einzelnen Spalten beschreiben abhängt. (Im Allgemeinen sind MEM, CLOCK-RATE und DISK-SIZE besser, wenn sie größere Werte annehmen, während ACC-TIME und PRICE besser sind, wenn sie kleinere Werte annehmen). Die Spalte CPU enthält Daten, die lediglich einen Typ bezeichnen (Nominalskala) wobei allerdings auch hier im allgemeinen angenommen wird, daß sich zwei Prozessoren bezüglich ihrer Leistung vergleichen lassen (was zu einer Teilordnung führt).*

eine “gründliche Einführung in alle Bereiche” anbietet (weil für diese “Attributwerte kein offensichtlicher Vergleichsoperator vorhanden ist). Andererseits ist die Frage berechtigt, welche der beiden Arten von Information nützlicher ist, wenn es darum geht, sich zwischen zwei Büchern zu entscheiden.

## 1.4 Definition und Abgrenzung

(Siehe auch Fuhr 1995 Kapitel 2)

Für den Begriff bzw. das Gebiet des Information Retrieval (IR) gibt es keine allgemein akzeptierte Definition oder Abgrenzung. Historisch gesehen wurde IR zum besseren (Wieder)auffinden von wissenschaftlicher Literatur entwickelt. Auch wenn dieses Gebiet nach wie vor einer der Schwerpunkte des IR ist, haben sich sowohl der Bereich der Objekte, mit denen IR umgeht, als auch die Aufgabenstellungen erweitert. Eine Beschreibung gibt die Fachgruppe Information Retrieval der Gesellschaft für Informatik (zitiert nach Fuhr 1995 )

> *Im Information Retrieval (IR) werden Informationssysteme in bezug auf ihre Rolle im Prozeß des Wissenstransfers vom menschlichen Wissensproduzenten zum Informations-Nachfragenden betrachtet. Die Fachgruppe “Information Retrieval” in der Gesellschaft für Informatik beschäftigt sich dabei schwerpunktmäßig mit jenen Fragestellungen, die im Zusammenhang mit vagen Anfragen und unsicherem Wissen entstehen. Vage Anfragen sind dadurch gekennzeichnet, daß die Antwort a priori nicht eindeutig definiert ist. Hierzu zählen neben Fragen mit unscharfen Kriterien insbesondere auch solche, die nur im Dialog iterativ duch Reformulierung (in Abhängigkeit von den bisherigen Systemantworten) beantwortet werden können; häufig müssen zudem mehrere Datenbasen zur Beantwortung einer einzelnen Anfrage durchsucht werden. Die Darstellungsform des in einem IR-System gespeicherten Wissens ist im Prinzip nicht beschränkt (z. B. Texte, multimediale Dokumente, Fakten, Regeln, semantische Netze). Die Unsicherheit (oder die Unvollständigkeit) dieses Wissens resultiert meist aus der begrenzten Repräsentation von dessen Semantik (z. B. bei Texten oder multimedialen Dokumenten); darüber hinaus*

*werden auch solche Anwendungen betrachtet, bei denen die gespeicherten Daten selbst unsicher oder unvollständig sind (wie z. B. bei vielen technisch-wissenschaftlichen Datensammlungen). Aus dieser Problematik ergibt sich die Notwendigkeit zur Bewertung der Qualität der Antworten eines Informationssystems, wobei in einem weiteren Sinne die Effektivität des Systems in bezug auf die Unterstützung des Benutzers bei der Lösung seines Anwendungsproblems beurteilt werden sollte.*

Van Rijsbergen (1979, zitiert nach der HTML-Version) gibt im ersten Kapitel tabellarisch eine Abgrenzung zwischen Information Retrieval und Fakten Retrieval (Data Retrieval), deren Punkte er dann im Text weiter erläutert (Siehe dort). In dieser Vorlesung werden wir zu Beginn des 4. Teils nochmal auf Fragen der theoretischen Modellierung von IR Systemen zurückkommen.

**Abbildung 4: Abgrenzung von Information Retrieval und Fakten Retrieval (Data Retrieval) nach Van Rijsbergen (1979)**

| | Data Retrieval | Information Retrieval |
|---|---|---|
| Matching | Exact match | Partial match, best match |
| Inference | Deduction | Induction |
| Model | Deterministic | Probabilistic |
| Classification | Monothetic | Polythetic |
| Query language | Artificial | Natural |
| Query specification | Complete | Incomplete |
| Items wanted | Matching | Relevant |
| Error response | Sensitive | Insensitive |

## 1.5 Knowledge Discovery / Data Mining

Im folgenden werden die Begriffe Knowledge Discovery in Databases (KDD) und Data Mining (DM) synonym verwendet. Als deutsche Übersetzung könnte man von Wissensextraktion oder Regelextraktion aus Datenbanken sprechen (was aber selten getan wird). Weitere Bezeichnungen und Definitionsvorschläge finden sich z. B. in der F(requently)A(sked)Q(uestions)-Seite der "Knowlede Discovery Mine" (Piatetsky-Shapiro html).

In dieser Vorlesung wird die folgende eher allgemeine Definition zugrunde gelegt (streng genommen ist das natürlich keine Definition):

*KDD beschreibt automatisierte Verfahren, mit denen Regelmäßigkeiten in Mengen von Datensätzen gefunden und in eine für Nutzende verständliche Form gebracht werden.*

Der erste Teil dieser Definition ist auch eine Beschreibung von "Machine Learning (ML)". Dieser Forschungsbereich befaßt sich damit, Klassifizierungsverfahren zu entwickeln, die mit einer Menge von Beispielen "trainiert" werden und anschließend in der Lage sind, diese und andere Datensätze (möglichst richtig) zu klassifizieren. KDD ist eng verwandt mit Machine Learning. Einige Autoren definieren KDD als Machine Learning, bei dem das Trainingset eine Datenbank ist (Holsheimer und Siebes 1994). Die Unterschiede

bestehen zum einen darin, daß die Anzahl der Datensätze in einer Datenbank erheblich größer sein kann als die Anzahl der Datensätze in den Trainingsets von ML Verfahren und daß die Datensätze weniger konsistent sind als ein Trainingsset. Zum anderen fordert die obige Definition auch, daß die Regelmäßigkeiten in eine für Nutzende verständliche Form gebracht werden müssen. Das können z. B. Regeln der Form "Wenn X und Y dann Z" sein, oder auch statistische Aussagen wie "in X% der Fälle, in denen Y eintritt, tritt auch Z ein". Diese Forderung ist z. B. für neuronale Netze eine hohe Hürde.

Die meisten bisher in der Literatur vorgestellten Systeme arbeiten auf Faktendatenbanken, d. h. sie arbeiten mit stark strukturierten Records aus wohldefinierten Typen, in der Regel also mit binären, ganzzahligen oder reellen Werten, wie z. B. Verkaufsdaten, Umsatzzahlen, Testwerte oder Preise. Als Ergebnisse liefern sie Verfahren, mit denen neue Datensätze in Kategorien eingeteilt werden können. Gern zitierte Beispiele sind Regelmäßigkeiten im Kaufverhalten von Konsumenten, die aus einer Kundendatenbank extrahiert werden können, um gezielter werben zu können; oder die Analyse von Daten einer Kreditkartengesellschaft, um bessere und einheitlichere Regeln für die Aufnahme neuer Kunden zu ermitteln (Carter and Catlett 1987). Häufig werden auch (anonymisierte) Patientendaten verwendet, um ärztliche Diagnosen aus den verschiedenen Labor- und Testwerten vorherzusagen.

Diese Beispiele zeigen auch die Problematik des Ansatzes. Es gehört nicht viel Phantasie dazu, sich vorzustellen, wie die Verfahren zum Beispiel bei der Vergabe von Arbeitsplätzen, dem Abschluß von Versicherungsverträgen oder ähnlichem angewendet werden können. Erinnert sei auch an die Diskussion um die Rasterfahndung in den 70er und 80er Jahren.

## 1.6 Beispiel eines Machine Learning Programms

Carter and Catlett (1987) beschreiben ein Machine Learning Programm, das Entscheidungen darüber trifft, ob ein Antrag auf eine Kreditkarte bewilligt werden soll oder nicht. Traditionell wurden diese Entscheidungen von den Mitarbeitenden des jeweiligen Instituts aufgrund ihrer Erfahrungen und anhand von Richtlinien getroffen. Ein anderes Verfahren ist eine sogenannte Scoring Table. Dabei werden für bestimmte Attribute, wie z. B. Höhe des Einkommens, Höhe des Bankguthabens, Grundbesitz, ... Punkte vergeben (für reellwertige Attribute werden Intervalle festgelegt). Wenn die Summe der Punkte einen Schwellenwert übersteigt, wird eine Kreditkarte vergeben, sonst nicht.

Anstelle dieses einfachen Summationsverfahrens kann auf der Basis von Fallbeispielen mit Machine Learning Verfahren ein Entscheidungsbaum erzeugt werden.

Carter and Catlett machen leider keine genaueren Angaben über die verwendeten Attribute (vermutlich, weil das System tatsächlich angewendet werden soll), sondern geben nur hypothetische Beispiele. So ist in Abbildung 5 eine Scoring Tabelle dargestellt und in Abbildung 6 ein Trainingsset. Abbildung 7 zeigt einen Entscheidungsbaum, der die Beispiele aus dem Trainingsset aus Abbildung 6 richtig kategorisiert. Man beachte, daß (in diesem hypothetischen Beispiel) nur zwei der vier Attribute verwendet werden.

## 1.7 Knowledge Discovery in Information Retrieval Systemen

Über Anwendungen von Knowledge Discovery Methoden im Bereich von Information Retrieval Systemen ist bisher selten berichtet worden. Allerdings lassen sich einige der Methoden, die in diesem Bereich benutzt werden, auch unter dem Aspekt des KDD interpretieren. Wir werden darauf im 4. Teil zurückkommen.

**Abbildung 5: Scoring Table aus Carter and Catlett (1987)**

| | | | | |
|---|---|---|---|---|
| Home status | boarder<br>*5* | rent<br>*8* | mortgage<br>*15* | owner<br>*20* |
| Time at address | 0-1<br>*4* | 1-2<br>*7* | 2-3<br>*10* | 3-4<br>*15* |
| Age of car | None<br>*0* | 0-1<br>*10* | 1-2<br>*15* | 2-3<br>*11* |
| Monthly disposable income | 0-$124<br>*0* | $125-249<br>*10* | $250-349<br>*15* | $350 up<br>*25* |

*In der linken Spalte stehen die Attribute, in den Zellen jeweils oben die möglichen Werte und unten (kursiv) die dazugehörigen Punktzahlen. (Home status beschreibt die Wohnsituation. Ein boarder ist z. B. ein Untermieter, mortgage ist eine Hypothek, dieser Attributwert beschreibt also jemanden, der eine eigene hypothekenbelastete Wohung bewohnt.)*

**Abbildung 6: Ein kleines Trainingsset (aus Carter and Catlett, 1987)**

| No | Attributes | | | | Class |
|---|---|---|---|---|---|
| | account | balance | employed | monthly expense | |
| 1 | bank | 700 | yes | 200 | accept |
| 2 | bank | 300 | yes | 600 | reject |
| 3 | none | 0 | yes | 400 | reject |
| 4 | other inst | 1200 | yes | 600 | accept |
| 5 | other inst | 800 | yes | 600 | reject |
| 6 | other inst | 1600 | yes | 200 | accept |
| 7 | bank | 3000 | no | 300 | accept |
| 8 | none | 0 | no | 200 | reject |

Einige einfache Beobachtungen kann man allerdings schon aus Anfragen an eine herkömmliche Datenbank abgeleiten: z. B. findet INSPEC (Januar bis Juni 95) für die Anfrage

`Information retrieval system`

92 Einträge, für die Anfrage

`Information retrieval systems`

219, für die Schnittmenge der beiden Anfragen aber nur 30. D. h daß die Singular- und die Pluralformen des Suchterms `Information retrieval system` sehr verschiedene Dokumentmengen finden.

**Abbildung 7: Entscheidungsbaum zum Trainingsset aus Abb. 6 (nach Carter and Catlett, 1987)**

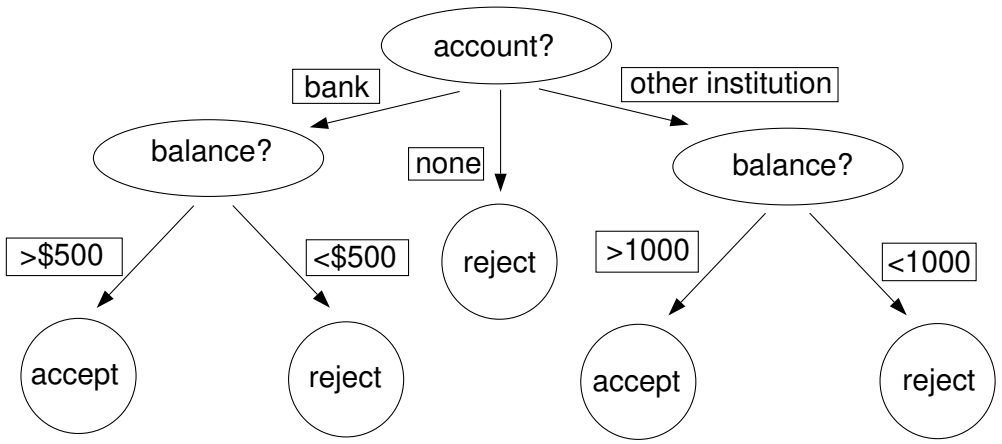



**Abbildung 8: Anzahl der in INSPEC gefundenen Dokumente für die Zeit vom Januar bis Juni 1995**

| Anfrage | IRS als Indexterm | Anzahl der gefundenen Dokumente |
|---|---|---|
| Information retrieval system | | 92 |
| Information retrieval systems | | 219 |
| | ja | 203 |
| Information retrieval system and Information retrieval systems | | 30 |
| Information retrieval system | nein | 68 |
| Information retrieval systems | nein | 16 |
| Information retrieval system | ja | 24 |

*Anzahl der in INSPEC gefundenen Dokumente, wenn der Indexterm information retrieval systems nicht beachtet wird, vorhanden sein muß (ja), nicht vorhanden sein darf (nein)*

”Information retrieval systems” ist ein Indexterm von Inspec. Er sollte als solcher den Dokumenten zugewiesen werden, die sich mit diesem Thema beschäftigen und bei denen es nicht nur am Rande erwähnt wird. Weitere Trefferzahlen für Anfragen, bei denen die Benutzung als Indexterm einbezogen wurde, finden sich in Tabelle 8. Es zeigt sich, daß die relative Häufigkeit des Indexterms bei der Pluralform ca. zehn mal so hoch ist wie bei der Singularform. Gründe für dieses Ergebnis lassen sich zunächst nur vermuten. Vielleicht werden von dem System, mit dem die Artikel indexiert werden, im Text (Titel oder Abstract) auftretende Indexterme automatisch erkannt und dem (menschlichen) Indexer vorgeschlagen. Vielleicht ist es aber auch so, daß die Pluralform eher verwendet wird, wenn über ein System berichtet wird, und die Singularform, wenn es nur als Beispiel am Rande erwähnt wird.

## Literaturverzeichnis


Carter, C., & Catlett, J. (1987). Assessing credit caord applications using machine learning. *IEEE Expert Fall 1987*, 71-79.

Fuhr, N. (1995). Information retrieval. Skriptum zur Vorlesung im SS 93, Universität Dortmund, http://ls6–www.informatik.uni-dortmund.de/teaching/courses/ir.

Holsheimer, M., & Siebes, A. P. J. M. (1994). Data mining: the search for knowledge in databases. Technical report, Centrum voor Wiskunde en Informatica, ftp://ftp.cwi.nl/pub/CWIreports/AA/CS-R9406.ps.Z.

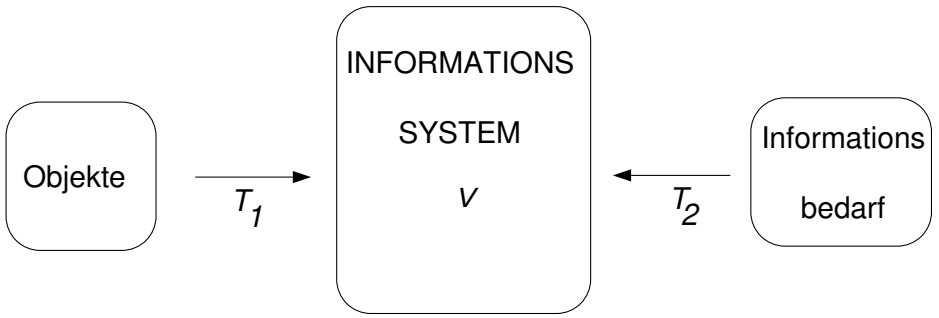
Piatetsky-Shapiro, G. Knowledge discovery mine. WWW-HTML Page: http://info.gte.com/~kdd/index.html.

Van Rijsbergen, C. J. (1979). *Information Retrieval*. London: Butterworth, als HTML-Version im WWW: http://www.dcs.glasgow.ac.uk/Keith/Preface.html.

# Teil 2 Wissensrepräsentation und Retrievalmodelle

Information Retrieval beschäftigt sich mit dem Suchen und Finden von Informationen. Häufig werden dabei Datenbanken verwendet, um Sammlungen von Objekten zu verwalten, d. h. die Datenbank enthält nicht die Objekte selbst, sondern **Repräsentationen** der Objekte. Diese Repräsentationen müssen die Eigenschaften der Objekte so gut beschreiben, daß die Objekte voneinander unterschieden und von Nutzenden gefunden werden können. Andererseits müssen Anfragen an das System so gestellt werden, daß sie auf den Repräsentationen der Objekte arbeiten können. Für das ganze System sind also mindestens zwei Transformationen notwendig: Beim Aufbau der Datenbank müssen für die Objekte Repräsentationen erzeugt und in der Datenbank abgelegt werden. Bei einer Anfrage müssen die Informationswünsche der Nutzenden in eine Repräsentation überführt werden, mit der die in der Datenbank abgelegten Repräsentationen verglichen werden können. Abbildung 9 stellt die Situation schematisch dar.

**Abbildung 9: Schematische Darstellung eines Information Retrieval Systems**




*Die Objekte werden durch die Transformation $T_1$ in Repräsentationen überführt, die in der Datenbank gespeichert werden. Eine Anfrage wird durch eine Transformation $T_2$ in eine Repräsentation überführt, die mit den Objekten der Datenbank verglichen wird.*

Als Beispiel kann man herkömmliche Literaturdatenbanken betrachten. Sie enthalten Angaben über Artikel und Bücher, wie sie im Beispiel 1 dargestellt sind, nicht aber die Artikel und Bücher selbst. Die Repräsentation der Objekte, also der Artikel, wird hier durch menschliche Indexierende erzeugt, die die Artikel lesen und die bibliografischen Angaben, Titel und Abstract sowie eine Einordnung in den systematischen Index zu einem **Dokument** zusammenstellen, das dann in der Datenbank abgelegt wird. (Bei dieser Umsetzung handelt es sich streng genommen natürlich nicht um eine Transformation, da das Ergebnis nicht nur vom Objekt abhängt, sondern auch von den Indexierenden. Aber darauf soll hier erstmal nicht weiter eingegangen werden.) Ob Volltextdatenbanken die Objekte selbst enthalten oder wiederum nur Repräsentationen, hängt davon ab, ob man einen Text als eine Repräsentation der in ihm enthaltenen Ideen und Aussagen auffaßt, oder als ein Objekt; beide Ansätze haben Vor- und Nachteile.

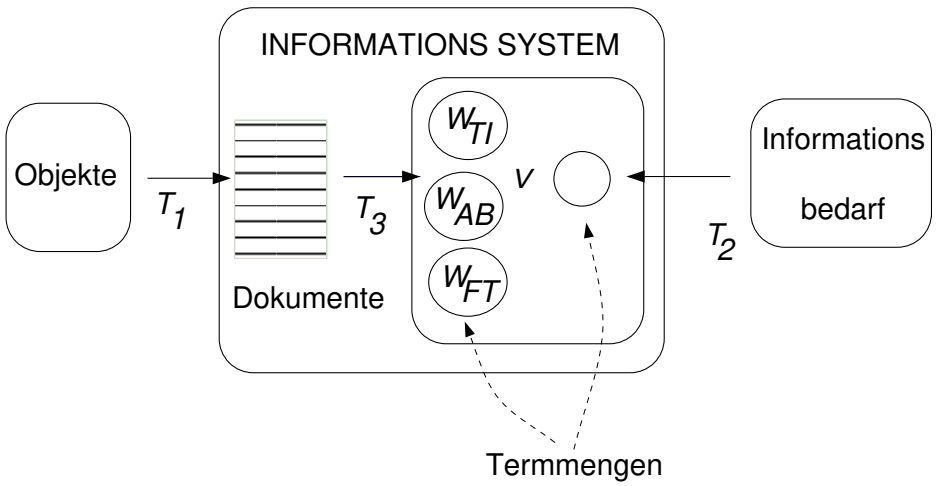
Soll nun eine Anfrage an das System gestellt werden, so muß der oder die Anfragende ihren Informationsbedarf in eine bestimmte Form bringen, die vom System bearbeitet werden kann. Im Beispiel aus Abschnitt 1.2 war das zunächst die Bool'sche Anfrage `RETRIEVAL SYSTEMS and MULTIMEDIA and IMAGES`.

Aufgabe des Systems ist es jetzt, alle zu dieser Anfrage "passenden" Dokumente zu finden. Dazu werden die von den Indexierenden erstellten Dokumente in Wörter zer-

legt, und es findet ein Abgleich statt. Die Repräsentation der Artikel durch Dokumente wird also nicht direkt genutzt, um den Vergleich zwischen Anfrage und Dokumenten durchzuführen, sondern aus den Dokumenten werden nochmal vereinfachte Repräsentationen gemacht, die dann mit der Anfrage verglichen werden. Dieser Vorgang ist in Abbildung 10 dargestellt.

Wie beim Bool'schen Retrieval werden in fast allen IR Systemen die Dokumente intern nochmals vereinfacht repräsentiert. Diese Vereinfachung ist nicht nur technisch bedingt, sie ist gleichzeitig auch eine Art Verallgemeinerung des Inhalts der Dokumente, die notwendig ist, um erfolgreich suchen zu können (vergl. die in Abbildung 4 dargestellte Tabelle von Van Rijsbergen): Wenn zum Suchen immer eine vollständige Beschreibung des Inhalts gegeben werden müßte, wäre die Suche in der Regel überflüssig. Eines der Haupthemen der IR Forschung der letzten 30 Jahre ist die Entwicklung und der Vergleich von internen Repräsentationsformen und darauf beruhenden Vergleichsverfahren.

**Abbildung 10: Schematische Darstellung eines textbasierten Bool'schen Information Retrieval Systems**




*Nachdem die Objekte durch die Transformation $T_1$ in Dokumente überführt wurden, werden diese vom IR System zum Vergleich mit der Anfrage in eine Repräsentation als Menge von Termen transformiert ($T_3$). Dabei werden die verschiedenen Felder der Dokumente in separate Termmengen $W_{TI}, W_{AB}, W_{FT}, ...$ umgewandelt, mit denen dann Mengenoperationen durchgeführt werden können.*

Was für Textdokumente gesagt wurde, gilt auch für andere Objekte, wie Bilder, Videos, Audiodokumente: ihre Speicherung in Rechnern erlaubt fast nie eine inhaltliche Erschließung. Um aus einem Pixelmuster zu schließen, ob es sich dabei um ein Gemälde von Rembrandt, ein Foto von Helmut Kohl oder um eine Bierreklame handelt, sind weitere Verfahren notwendig, die gegenwärtig Thema der Forschung sind und es auch noch eine ganze Weile bleiben dürften.

## 2.1 Bool'sches Retrieval

Bool'sches Retrieval, ist nach wie vor der Standard bei den meisten kommerziellen IR Systemen. Es zeichnet sich durch logische Klarheit aus, deren Angemessenheit im Bezug auf Sprache allerdings nicht unumstritten ist. Für ungeübte Anwendende sind die heutigen Systeme im allgemeinen nicht ganz einfach zu bedienen. Das liegt allerdings auch daran, daß viele Benutzeroberflächen ziemlich primitiv gestaltet sind. Bool'sche Systeme liefern auf eine Anfrage eine ungeordnete Menge von Dokumenten.

### 2.1.1 Logik des Bool'schen Retrievals

Grundidee des Bool'schen Retrievals sind einfache Mengenoperationen auf Mengen von Dokumenten, die durch Attributwerte der Dokumente charakterisiert sind.

Eine Anfrage ist eine Verknüpfung von Paaren aus einem Attribut und einem dazugehörigen Attributwert (**Attribut-Wert-Paare**) durch die Operatoren AND, OR und NOT. Ein Attribut-Wert-Paar steht in der Anfrage für die Menge der Dokumente, bei denen das entsprechende Attribut den angegebenen Wert annimmt. Ein **Attribut** ist dabei eine Abbildung, die einem Dokument einen Wert zuordnet. Bezeichne $D$ die Menge aller Dokumente und $t : D \rightarrow T$, $t(d) = t_i$ ein Attribut, so ist

$$D_{t,t_i} = t^{-1}(t_i) = \{d \in D \mid t(d) = \ t_i\} \tag{1}$$

die Menge der Dokumente, die durch den Attributwert $t_i$ charakterisiert sind. Diese Menge wird auf eine Anfrage, die nur aus $(t, t_1)$ besteht, zurückgeliefert. Sind in einer Anfrage zwei Attribut-Wert-Paare durch einen Bool'schen Operator verknüpft, so wird damit die entsprechende Verknüpfung der Dokumentmengen bezeichnet: $(t, t_1)$ AND $(s, s_1)$ bezeichnet den Durchschnitt $D_{t,t_1} \cap D_{s,s_1}$, der Ausdruck $(t, t_1)$ OR $(s, s_1)$ die Vereinigung $D_{t,t_1} \cup D_{s,s_1}$ und der unäre Operator NOT $(t, t_1)$ das Komplement $D \setminus D_{t,t_1}$. Letzterer wird i. a. nur in Verbindung mit dem AND Operator eingesetzt, um bestimmte Dokumente aus einer Ergebnismenge auszuschließen.

#### 2.1.1.1 Bool'sches Retrieval für Textdokumente

Sind die Dokumente Textdokumente, wie z. B. in einer Literaturdatenbank, so sind die wichtigsten Attribute das Auftreten von **Termen** in den verschiedenen Feldern der Dokumente. Terme müssen dafür in geeigneter Weise als zusammenhängende Zeichenketten aus Buchstaben und bestimmten Sonderzeichen, die durch Leerzeichen, Interpunktionszeichen und andere Sonderzeichen begrenzt sind, definiert werden.

Ein Attribut $TI_{t_1} : D \rightarrow \{false, \ true\}$ wäre z. B. das Auftreten eines Terms $t_1$ im Titelfeld der Dokumente. Die dadurch charakterisierte Dokumentmenge

$$D_{TI_{t_1},true} = \{d \in D \mid d \ enthaelt \ t_1 \ im \ Titelfeld\} \tag{2}$$

enthält diejenigen Dokumente, bei denen der Term $t_1$ im Titel vorkommt.

Man kann auch das Attribut $x : D \rightarrow \mathfrak{P}(V)$ als die Menge der Terme, die im Titelfeld des Dokumentes vorkommen, definieren. Dabei bezeichnet $V$ die Menge der möglichen Terme und $\mathfrak{P}(V)$ die Potenzmenge (Menge der Teilmengen) von $V$. Dann ist $x$ eine mengenwertige Funktion, und die durch einen Term $t_1$ charakterisierte Menge ergibt sich als

$$D_{x,t_1} = x^{-1}(\{t_1\}) = \{d \in D \mid d \ enthaelt \ t_1 \ im \ Titelfeld\} \tag{3}$$

d. h. ein in einer Anfrage angegebener Term muß als Element einer einelementigen Menge aufgefaßt werden.

Eine Anfrage ist jetzt eine Verknüpfung von Paaren aus Feldbezeichnern und Termen durch die Operatoren AND, OR und NOT. Bei den meisten Systemen wird beim Fehlen eines Feldbezeichners als Voreinstellung das ganze Dokument – also die Vereinigung aller Felder – angenommen. Ein Term steht für die Menge der Dokumente, die den Term im entsprechenden Feld enthalten. Sind in einer Anfrage zwei Terme $t_1$ und $t_2$ mit AND verknüpft, so werden alle Dokumente angezeigt, die beide Terme enthalten. Sind sie mit einem OR verknüpft, werden die Dokumente angezeigt, die den einen oder den anderen Term (oder beide) enthalten. Lautet die Anfrage $t_1$ AND NOT $t_2$ werden die Dokumente ausgewählt, die zwar den Term $t_1$, nicht aber den Term $t_2$ enthalten. Die Anfragen können mit den Operatoren durch Klammerung beliebig kompliziert geschachtelt werden.

Über diese Grundidee hinaus bieten die meisten Bool'schen Systeme weitere Verfeinerungen:

- Für Felder, die Zahlen (i. a. Jahreszahlen) enthalten, stehen entsprechende Vergleichsoperatoren wie kleiner (<) oder größer (>) zur Verfügung.
- Werden mehrere Terme ohne verbindenden Operator angegeben, wird nach der entsprechenden Kette von Termen gesucht.
- Als Verallgemeinerung gibt es Funktionen, die den Abstand zwischen zwei Termen in einem Textfeld berücksichtigen, also nur solche Dokumente auswählen, bei denen zwischen dem Term $t_1$ und dem Term $t_2$ höchstens (genau) $n$ andere Terme stehen. Dabei kann auch die Reihenfolge der Terme vorgegeben werden.
- Terme können trunkiert werden. D. h. es können durch Sonderzeichen (Wildcards) alle Terme bezeichnet werden, die ein bestimmtes Zeichenmuster in einer bestimmten Position enthalten. (i. a. ein Wortanfang, der mit einem Trunkierungszeichen endet und damit alle Terme, die mit der Zeichenkette beginnen, selektiert).

### 2.1.2 Implementation mit einer invertierten Liste

(Siehe auch Frakes Baeza-Yates 1992)

Bool'sche Retrieval Systeme werden i. a. mit Hilfe von **invertierten Listen** implementiert, d. h. es wird eine Liste (oder andere Speicherstruktur) angelegt, in der zu jedem Term eingetragen wird, in welchen Dokumenten er vorkommt. Dieses Verfahren ermöglicht einen schnellen Zugriff, aber es erfordert dafür auch erhebliche Mengen an Speicherplatz. Zudem muß bei einer Erweiterung des Dokumentbestandes die invertierte Liste neu berechnet werden.

Zur Konstruktion einer invertierten Liste muß ein **kontrolliertes Vokabular**, also eine Liste aller Terme, nach denen gesucht werden kann, vorgegeben werden bzw. es müssen Regeln angegeben werden, mit denen bestimmt werden kann, ob ein Wort zu diesem Vokabular gehört oder nicht. Dieses Vokabular sollte zum einen möglichst alle "sinntragenden" Wörter enthalten, die in den Dokumenten auftreten, andererseits sollte es aus Gründen der Speicherplatzökonomie möglichst nicht zu groß werden. Insbesondere sehr häufige Terme benötigen durch die vielen Einträge viel Speicherplatz. Folgendes Vorgehen ist bei der Konstruktion eines kontrollierten Vokabulars üblich:

- Es werden Regeln festgelegt, nach denen Texte in Terme zerlegt werden. Das betrifft z. B. die Behandlung von Interpunktionszeichen, Bindestrichen etc.
- In einer Stoppwortliste werden die häufigsten Wörter, wie Artikel, Präpositionen etc. gesammelt, die nicht in die Liste aufgenommen werden.

- Es werden Regeln festgelegt, nach denen bestimmte Zeichenketten aussortiert werden, wie z. B. solche, die nur aus Ziffern bestehen, Römische Zahlen etc.
- Die Zeichenketten, die nach Anwendung dieser Regeln übrigbleiben, sind die Terme des kontrollierten Vokabulars.

Die Konstruktion der invertierten Liste vollzieht sich im Prinzip in folgenden Schritten:

- Anhand der Regeln zur Konstruktion eines kontrollierten Vokabulars werden die Dokumente in Terme zerlegt.
- Zu den Termen werden jeweils das Dokument und die Position des Auftretens im Dokument geschrieben. (-> Word within location list)
- Diese Paare aus Termen und ihren Positionen werden (primär nach Termen alphabetisch, sekundär nach Positionen) sortiert.
- Paare mit gleichen Termen werden zusammengefaßt, wobei die Positionen in einer sortierten Liste an den Term angefügt werden.
- Die Terme und die Listen mit den Positionen werden getrennt, wobei die Terme in eine Indexdatei geschrieben werden, die zu jedem Term einen Pointer auf die zugehörige Liste enthält (bzw. einen Offset in einem Postings File, das alle Positionen enthält).

Die Trennung der Terme und Listen in verschiedene Dateien dient dazu, den Zugriff auf die Terme zu beschleunigen. Es muß in diesem Fall bei der sequentiellen Suche pro Term, der vor dem gesuchten Term liegt, nur ein Pointer überlesen werden und nicht die ganze Liste der Positionen.

Eine Anfrage wird nun folgendermaßen bearbeitet:

- Es werden die Terme in der Anfrage isoliert.
- Zu jedem Term wird in der Indexdatei der invertierten Liste der Pointer auf die Liste mit den Positionen, also den Nummern der Dokumente bzw. der Dokumentfelder, in denen der Term vorkommt, ermittelt.
- Die Listen zu den verschiedenen Termen werden, je nachdem, durch welchen Operator die Terme verbunden sind, vereinigt, es wird der Durchschnitt gebildet oder es wird eine Differenz berechnet.
- Die Dokumente, die in der resultierenden Liste übrigbleiben, werden aus der Dokumentdatei geholt und als Resultat der Anfrage präsentiert.

Der aufwendigste Schritt bei der Berechnung der invertierten Liste ist das Sortieren, insbesondere, wenn die ganze Liste auf einmal sortiert wird. Deshalb setzen an diesem Punkt diverse Verbesserungsmethoden an. Zum einen können die Daten aufgeteilt werden, so daß die Datenmengen, die zwischengespeichert werden müssen, handhabbar bleiben. Zum anderen können die Terme in den Knoten eines binären Baums gespeichert werden, an denen dann Listen mit den Positionsangaben angehängt werden.

Ein weiteres Verfahren ist der Fast Inversion Algorithm von Fox and Lee (1991 beschrieben in Frakes Baeza-Yates 1992).

## 2.2 Linguistische Verfahren der Repräsentation, Thesauri

Im vorigen Kapitel wurden Terme als Zeichenketten, die nach bestimmten Regeln ausgewählt werden, definiert. Dieser Ansatz bietet eine gewisse Flexibilität in bezug auf verschiedene Formen eines Wortes. So kann durch Trunkierung am Wortende eine ganze Reihe von verschiedenen Formen eines Wortes zusammengefaßt werden. Diese Möglichkeiten beschränken sich allerdings auf Wörter, deren Schreibweise eine solche Erfassung verschiedener Formen zuläßt. In Abbildung 11 ist ein Beispiel angegeben, bei dem ein solcher Versuch mißlungen ist.

**Abbildung 11: Trunkierungen, die nicht nur Tiere ausschließen (aus Ferber, Wettler, Rapp 1995)**

```
3  273 FIND CT D Vertebrates
4   24 FIND CT D Invertebrates
5  346 FIND ALL Animal$
6 2981 FIND ALL [Cat$;Dog$;Pidgeon$;Monkey$;
           Rat$;Dog$;Mouse;Mice;Rabbit$]
7 3264 FIND 3 TO 6
8   15 FIND 2 NOT 7
```

*Die erste Spalte ist eine Zeilennummer, die zweite gibt die Anzahl der gefundenen Dokumente an, hinter FIND steht die zur Zeile gehörende Anfrage. Die Nummern in den Anfragen referieren auf die Dokumente, die in der entsprechenden Zeile gefunden wurden. In Zeile 6 werden Wortanfänge von Tiernamen angegeben, die häufig als Versuchstiere verwendet werden. Dabei werden aber auch Begriffe wie Category, Dogmatism, oder Rating mit erfaßt. Alle Dokumente, die einen dieser Terme enthalten, werden in Zeile 8 von dem Suchergebnis ausgeschlossen. In der Recherche sollten aber vermutlich nur solche Dokumente, die sich mit Tierexperimenten beschäftigen ausgeschlossen werden.*

Dieses Beispiel aus einer psychologischen Datenbank zeigt den Versuch über die Verwendung von Zeichenketten eine bestimmte Gruppe von Dokumenten zu erfassen (bzw. auszuschließen), nämlich solche, die sich mit Tierexperimenten beschäftigen. Für einen Leser bzw. eine Leserin ist das eine verhältnismäßig einfache Aufgabe. Sie setzt allerdings voraus, daß die Texte gelesen und zu einem gewissen Grad verstanden werden und daß allgemeines Wissen über Tiere und Experimente mit Tieren vorhanden ist, das genutzt werden kann, um die gelesenen Sätze so weit wie möglich zu interpretieren. Denn selbst wenn Lesende die genannte Tierart und die Experimentalanordnung nicht kennen, werden sie i. a. in der Lage sein zu entscheiden, ob es sich um einen Artikel über Experimente mit Tieren handelt oder nicht. Diese Möglichkeit zusätzliches Wissen zu nutzen, um Anfragen zu beantworten, bietet ein boolesches Retrieval System, das mit dem Vergleich von Zeichenketten arbeitet, zunächst nicht. Im obigen Beispiel wurde versucht das auszugleichen, indem Bezeichner für Tiere, die häufig in psychologischen Experimenten verwendet werden, gesammelt wurden und damit quasi ein Konzept “Versuchstier” konstruiert wurde. Probleme, die sich dabei ergeben, können exemplarisch als Motivation für die in diesem Abschnitt behandelten Ansätze genommen werden.

In Zeile 6 aus Abbildung 11 wurde versucht verschiedenen Wortformen zusammenzufassen. Bei “Cat”, “Dog” etc. sollte das erreicht werden, indem durch die Trunkierung ein mögliches Plural “s” mit erfaßt werden sollte. Dabei wurden allerding (vermutlich unbeabsichtigt) viele weitere Terme mit ausgewählt. Bei “Mouse” und “Mice” versagt das

Trunkierungsverfahren, weil sich der Stamm des Wortes beim Übergang zur Mehrzahl ändert. In den Zeilen 3 und 4 wurden Oberbegriffe (“Vertebrates”, “Invertebrates”) verwendet, die aber offensichtlich in weniger Dokumenten auftreten wie die Zeichenkette “Animal”.

Das sprachliche und faktische Wissen, das Menschen beim Lesen die richtige Zuordnung erlaubt, läßt sich eben nur schwer durch Regeln auf der Ebene von Zeichenketten ersetzen, wenn es darum geht, Inhalte vergleichbar zu machen.

Es gibt verschiedene Ansätze, dieses Problem der **Vergleichbarkeit von Inhalten** durch geeignete Repräsentation anzugehen. Grundsätzlich kann man zwei verschiedene Herangehensweisen unterscheiden:

- Versuche, die natürliche Sprache so zu repräsentieren, daß “inhaltliche Ähnlichkeiten” erkennbar werden.
- Versuche, die zulässigen Mittel zur inhaltlichen Beschreibung so einzuschränken, daß sie Ähnlichkeiten abbilden.

Wir werden uns zunächst mit Ansätzen der ersten Art beschäftigen.

### 2.2.1 Wortorientierte Verfahren

Der **computerlinguistische Ansatz** versucht, Terme in Texten nicht als Zeichenketten zu definieren, sondern als bestimmte Formen eines Wortes aufzufassen. Es werden also z. B. verschiedene Flexionsformen eines Wortes als zusammengehörig oder sogar als identisch betrachtet. Man unterscheidet dabei zwischen der **Grundformreduktion**, die die Wörter auf ihre grammatikalische Grundform (bei Substantiven auf den Nominativ singular, bei Verben auf den Infinitiv) zurückführt, und **Stammformreduktion**, die die Wortformen auf ihren Stamm zurückführt, der im allgemeinen keine in der Sprache als Wort vorkommende Form ist und z. B. für ein Verb und ein Substantiv gleich sein kann. Diese Reduktionen von verschiedenen Formen eines Wortes auf eine Grundform oder einen Stamm werden auch Lemmatisierungen genannt. Sie führen zum einen dazu, daß sehr verschiedene Zeichenketten als gleich angesehen werden, andererseits sollten im Idealfall identische Zeichenketten, die von verschiedenen Wortstämmen abgeleitet werden, als verschieden angesehen werden. So werden “finden” und “gefundenes” auf den selben Stamm zurückgeführt, sind aber als Zeichenketten ziemlich verschieden, während die Zeichenkette “Winden” auf die Stämme “Wind”, “Winde” und “winden” zurückgeführt werden kann.

Im Information Retrieval verfolgt man mit der Lemmatisierung zwei Ziele:

- Durch das Zusammenführen von verschiedenen Wortformen zu einem Term werden invertierte Listen bzw. andere Verweisstrukturen von Termen auf Dokumente kleiner, da weniger Terme verwaltet werden müssen.
- Dadurch, daß verschiedene Wortformen in einem Term zusammengefaßt werden, tritt dieser Term tendenziell in mehr Dokumenten auf. Inhaltlich gesehen stellt das eine Verallgemeinerung der Bedeutung des Terms dar. Bei der Suche sollten daher mit dem zusammengefaßten Term mehr Dokumente zu einem Thema gefunden werden.

Zur Frage, wieweit die zweite Annahme zutrifft, sind eine Reihe von Experimenten durchgeführt worden (siehe Frakes und Baeza-Yates 1992, Hull, 1995). Die Ergebnisse sind uneinheitlich. Wichtige Einflußfaktoren für den Erfolg scheinen u. a. die Homogenität der Dokumentsammlung in der gesucht wird und die Länge der Anfrage zu sein.

Um in einem Text Grundformen oder Wortstämme als Terme zu isolieren, kann man zunächst versuchen, allgemeine Regeln aufzustellen, nach denen Zeichenketten auf Grundformen bzw. Stämme zurückgeführt werden können. Für das Englische gibt es eine ganze Reihe solcher Verfahren. Einige werden im 8. Kapitel des Buches von Frakes und Baeza-Yates (1992) behandelt. Hier soll der Algorithmus von Kuhlen (1977), kurz vorgestellt werden.

#### 2.2.1.1 Lexikographische Grundformenreduktion nach Kuhlen

Kuhlen gibt ein Regelwerk an, mit dem die meisten englischen Wörter auf ihre formale bzw. lexikographische Grundform oder die Stammform zurückgeführt werden können. Dabei bezeichnet er (Seite 57) als

- **lexikographische Grundform** die Form, in der das Wort in einem Wörterbuch zu finden ist. Die durch Flexion möglicherweise entstandenenen graphematischen Veränderungen der Grundformen werden rückgängig gemacht d. h. die Wörter werden deflektiert und anschließend rekodiert.
- **formale Grundform** " ... Wortfragmente, bei denen die 'normalen' englischen und fremdsprachigen (hauptsächlich lateinischen) Flexionsendungen abgetrennt werden, ohne daß die entstandenen Wortfragmente rekodiert würden."
- **Stammform nach linguistischen Prinzipien** die Zeichenketten, die durch Deflexion und Abtrennen von Derivationsendungen entstehen. Diese Zeichenketten sollen soweit wie möglich durch Rekodierung vereinheitlicht werden.

Als Beispiel sind in Abbildung 12 die verschiedenen Formen der Wörter mit der Stammform ABSORB angegeben.

Einige der Regeln sind in Abbildung 13 angegeben.

Die Evaluierung seines Algorithmus' faßt Kuhlen folgendermaßen zusammen (S. 11): "Mit 95% Wahrscheinlichkeit ist zu erwarten, daß die Fehlerquote bei einem Fachtext von ca. 72 000 unterschiedlichen Wörtern nicht mehr als 0.5 % ausmachen wird." Die Reduktionsquote liegt bei diesem Korpus bei 13% für die lexikographischen Grundformenreduktion und bei 27,3% für die Stammformenreduktion; d. h. die Anzahl verschiedener Wörter im Korpus verringert sich um die genannten Prozentsätze, wenn alle Wörter durch ihre Grundformen bzw. Stammformen ersetzt werden.

Diese Form der Reduktion ist im Englischen verhältnismäßig erfolgreich, weil die Wörter sich in der Sprache wenig ändern, d.h. wenige Flexionsformen besitzen. Zudem gibt es wenig zusammengesetzte Wörter. Im Deutschen sieht das ganz anders aus. Hier lassen sich solche Regeln nicht erfolgreich konstruieren und anwenden. Deshalb muß auf lexikonbasierte Verfahren zurückgegriffen werden.

#### 2.2.1.2 Lexikonbasierte Morphologieprogramme

Probleme bereiten im Deutschen die vielfältigen Veränderungen auch des Wortstammes. So werden nicht nur am Ende Suffixe angehängt, wie bei "des Hauses", sondern es werden Umlaute eingefügt (Apfel - Äpfel, alt - älter, Kuß - Küsse), von "ß" zu "ss" gewechselt und Präfixe vorangestellt (laufen - gelaufen). Bei den Präfixen muß noch unterschieden werden, ob das Präfix abgetrennt werden kann (mitbringen: er brachte den Brief mit; nicht: er mitbrachte den Brief) oder nicht (überbringen: er überbrachte den Brief; nicht: er brachte den Brief über). Abgetrennte Präfixe können die Bedeutung eines Wortes auch verändern (Professorin Mayer schlug ihren Assistenten (für die Stelle vor)). Diese Eigenschaften lassen sich kaum in allgemeine Regeln fassen; sie sind von den einzelnen Wörtern abhängig. Um sie zu speichern, muß ein Lexikon

**Abbildung 12: Die verschiedenen Reduktionsformen nach Kuhlen am Beispiel (aus Kuhlen 1977, S. 58)**

| *Formale Grundform* | *Textwörter* | *Lexikalische Grundform* | *Stamm-form* |
|---|---|---|---|
| ABSORB | ABSORB<br>ABSORBED<br>ABSORBING<br>ABSORBS | ABSORB | ABSORB |
| | ABSORBER<br>ABSORBERS | ABSORBER | |
| ABSORBAB | ABSORBABLE<br>ABSORBABLY | ABSORBABLE | |
| ABSORBANC | ABSORBANCE<br>ABSORBANCES | ABSORBANCE | |
| | ABSORBANCY<br>ABSORBANCIES | ABSORBANCY | |
| ABSORBENT | ABSORBENT<br>ABSORBENTS<br>ABSORBENTLY | ABSORBENT | |
| ABSORPTION | ABSORPTION<br>ABSORPTIONS | ABSORPTION | |
| ABSORPTIV | ABSORBTIVELY<br>ABSORBTIVE | ABSORPTIVE | |

angelegt werden, in dem zu jedem Wort die jeweiligen Eigenschaften abgelegt werden. Dieses Lexikon muß zumindest teilweise von Hand gepflegt werden, was einen enormen Aufwand bedeuten kann. Besondere Probleme macht dabei die Tatsache, daß im Deutschen häufig Komposita gebildet werden, indem mehrere Substantive, Adjektive, Adverbien und auch Verben zu einem neuen Wort zusammengefügt werden. Dabei werden verschiedene "**Verfugungen**" verwendet, die keiner Regelmäßigkeit folgen (Schwein-kram, Schwein-s-haxe, Schwein-s-braten oder Schwein-e-braten), aber durchaus Probleme bei der Auflösung von Komposita machen können (Arbeit-s-amt, Arbeit-samt).

Lexikonbasierte Morphologieprogramme können entweder als Vollformenlexikon, bei denen zu jeder möglichen Form der Stamm aufgeführt wird, implementiert werden oder als Wortstammlexikon. Im letzteren Fall werden zu jedem Stamm nur soviele Informationen gespeichert, daß alle Wortformen generiert werden können. Dazu werden verschiedene Flexionsklassen unterschieden und Besonderheiten vermerkt. In "Morphy", einem Morphologieprogramm mit Stammformenlexikon von Lezius (1995), wird z. B. für ein Substantiv abgespeichert, in welche von 62 verschiedenen Deklinationsklassen es fällt (dadurch ist auch das Genus bestimmt), und ob bei der Pluralbildung ein ß/ss Wechsel oder eine Vokalumlautung stattfindet. Bei einem ß/ss Wechsel wird das

**Abbildung 13: Übersicht über die Regeln zur lexikographischen Grundformenreduktion nach Kuhlen (1977)**

---

IES → Y
*xy*ES → *xy* für *xy* = *k*O, CH, SH, SS, ZZ oder *x*X
*xy*S → *xy* für *xy* = *xk*, xE, *v*Y, *v*O, OA oder EA
IES' → Y
*x*ES' → *x*
*x*S' → *x*
*x*'S → *x*
*x*' → *x*
*xy*ING → *xy* für *xy* = *kk*, *xv*, *x*X
*xy*ING → *xy*E für *xy* = *vk*
IED → Y
*xy*ED → *xy* für *xy* = *kk*, *xv*, *x*X
*xy*ED → *xy*E für *xy* = *vk*

---

*Angeführt sind Wortenden der vollen Wortformen und die Enden durch die diese überführt werden. Dabei bezeichnen x und y (klein und kursiv gedruckt) einen beliebigen aber festen Buchstaben, k bezeichnet einen beliebigen Konsonanten und v einen beliebigen Vokal. Zu jeder Regel gibt es noch Listen mit Ausnahmen, d. h. Wörtern, bei denen sie nicht angewendet werden dürfen.*

erste von rechts vorkommende “ß” durch “ss” ersetzt, bei der Vokalumlautung der erste von rechts vorkommende Vokal durch seinen Umlaut.

Während die Grundformenreduktion im Wesentlichen nur erlaubt, vorhandene Formen auf Grundformen abzubilden, können mit den lexikonbasierten Verfahren zu einer Grundform bestimmte oder auch alle Wortformen konstruiert werden. Die Reduktion auf lexikalische Grundformen wird in “Morphy” nach dem Prinzip “Lemmatisierung durch Generierung” durchgeführt. Es wird dabei folgendermaßen vorgegangen:

- Suche nach der Wortform in einem kleinen Hochfrequenzwörterbuch. Wird das Wort gefunden, ist die Lemmatisierung beendet.
- Flexionsanalyse: sukzessive werden die letzen Buchstaben abgeschnitten und die verbleibende Zeichenkette im Stammformenlexikon gesucht. Wird ein Stamm gefunden, wird untersucht, ob damit die Wortform generiert werden kann. Bei der Suche im Stammformenlexikon werden die bei dem jeweiligen Stamm zugelassenen Umwandlungen (ß/ss, Umlautung) berücksichtigt (also gegebenenfalls in der Wortform rückgängig gemacht). Es werden alle Stämme geprüft, um Mehrdeutigkeiten zu finden. Werden Grundformen gefunden, werden sie mit der Wortklassenzugehörigkeit ausgegeben und die Lemmatisierung ist beendet. (vergl. Abbildung 14)
- Kompositionsanalyse: Es werden von rechts rekursiv die jeweils längsten Wortformen abgeschnitten, die im Lexikon gefunden werden. Läßt sich das Wort so in Teilwörter zerlegen, wird es als Kompositum bezeichnet und die Lemmatisierung beendet.

- Konnte das Wort nicht lemmatisiert werden, wird auf der Basis einer empirischen Häufigkeitstabelle für Endungen eine Hypothese über die Wortklassenzugehörigkeit ausgegeben.

**Abbildung 14: Die Wortform "Flüssen" in der Flexionsanalyse bei Lezius (1995)**

| Fall / Endung | - | n | en | sen | ... |
|---|---|---|---|---|---|
| normal | Flüssen- | Flüsse-n | Flüss-en | Flüs-sen | ... |
| Umlaut | Flussen- | Flusse-n | Fluss-en | Flus-sen | ... |
| ß/ss | Flüßen- | Flüße-n | Flüß-en | Flü-ßen | ... |
| beides | Flußen- | Fluße-n | **Fluß**-en | Flu-ßen | ... |

Lexikonbasierte Morphologieprogramme sind natürlich erheblich aufwendiger als die Grundformenreduktion mit lexikographischen Ersetzungsregeln. Sie bieten dafür aber erheblich mehr und genauere Informationen über die untersuchten Wortformen. Insbesondere für Verfahren, die nicht nur isolierte Wörter sondern auch Wörter im (syntatktischen) Satzusammenhang auswerten sind diese zusätzlichen Informationen über die Wortform nützlich oder sogar zwingend notwendig. So gibt es auch für das Englische lexikonbasierte Morphologieprogramme. Eines davon (auch für die Sprachen Französisch, Deutsch, Italienisch und Spanisch) wird vom Xerox-Forschungszentrum in Grenoble (Xerox html) angeboten.

Auch nach der Reduktion auf Grund- oder Stammformen bleibt das Problem der mehrdeutigen Wortformen bzw. Wörter bestehen. Es wird als **Polysemie**– oder **Ambiguitätsproblem** bezeichnet und ist für isolierte Wörter nicht zu lösen. Man kann aber die Situationen betrachten, in denen die Reduktionen vorgenommen werden, und versuchen, die Mehrdeutigkeiten in dem jeweiligen Kontext zu lösen.

Die Bestimmung von Termen ist im Information Retrieval bei der Indexierung und bei der Bearbeitung von Anfragen notwendig (in der schematischen Darstellung des Information Retrieval Prozesses in Abbildung 10 sind das die Transformationen $T_2$ und $T_3$). Bei der Bestimmung von Termen als Zeichenketten sind die Regeln, nach denen Terme isoliert werden, in beiden Fällen sehr ähnlich. Sie bestehen im wesentlichen aus Regeln auf Zeichenebene, die durch die Stoppwortliste und einige Ausschlußregeln für bestimmte Zeichenmuster ergänzt werden. Bei der Bearbeitung von Anfragen (also bei der Transformation $T_2$) müssen über diese Regeln hinaus noch trunkierte Zeichenketten expandiert werden.

Bei der Bestimmung von Grundformen bzw. Stämmen als Terme unterscheidet sich die Situation bei der Indexierung und der Queryverarbeitung. Genauer gesagt: Weil die Bestimmung der Terme erheblich schwieriger ist, muß man sich die jeweilige Situation zunutze machen, um das Problem möglichst gut zu lösen: Bei der Bestimmung von Anfragetermen ($T_2$) kann man versuchen, die Auswahl der Terme im Dialog mit den Nutzenden zu lösen. Dazu muß allerdings zunächst erkannt werden, daß ein Term mehrdeutig ist. Das ist nur mit einem lexikonbasierten oder einem statistischen System möglich. Bei der Indexierung ($T_3$) kann man versuchen, den Kontext der Dokumente zu nutzen, um die vorgefundenen mehrdeutigen Zeichenketten dem richtigen Term zuzuordnen. Auch dazu ist allerdings zusätzliches Wissen wie ein Lexikon oder eine Assoziationstabelle nötig.

**Abbildung 15: Beispiel für die Kontextabhängigkeit der Dereferenzierung des Pronomen “es”**

---

**Das gelbe Auto überholte das rote Auto obwohl es stärker war**
**Das gelbe Auto überholte das rote Auto obwohl es schwächer war**
**Das gelbe Auto überholte das rote Auto obwohl es dunkel war**
**Das gelbe Auto überholte das rote Auto obwohl es regnerisch war**
**Das gelbe Auto überholte das rote Auto obwohl es glatt war**
**Das gelbe Auto überholte das rote Auto obwohl es verboten war**

---

Das zur Polysemie entgegengesetzte Problem ist das der Synonymie: gleiche Objekte werden mit verschiedenen Begriffen bezeichnet. Dieses Problem kann auch mit externem Wissen, wie einem Synonymlexikon, angegangen werden.

### 2.2.2 Allgemeinere Wissenstrukturen

Ein allgemeinerer Lösungsansatz für die Probleme der Polysemie und Synonymie ist die Repräsentation von Textinhalten mithilfe von komplexeren Wissenstrukturen, wie **semantische Netze**, **Skripts** und **Frames**. Ziel solcher Verfahren ist es, Wissen unabhängig von der jeweiligen sprachlichen “Oberflächenstruktur” zu repräsentieren. Sie stammen meist aus dem Bereich der künstlichen Intelligenz (KI). Dort dienen sie dazu, Wissen, aus dem automatisch Folgerungen abgeleitet werden sollen, zu speichern.

Bisher gibt es nur wenige experimentelle Systeme, die solche Wissensrepräsentationen im Bereich des Information Retrieval einsetzen. Die Probleme, die sich dabei ergeben, sind erheblich:

- Solche Systeme benötigen große Wissensbasen, um erfolgreich arbeiten zu können.
- Sie sollten in der Lage sein, Wissen aus Dokumenten automatisch zu extrahieren. Die dazu benötigten Verfahren im Bereich der Verarbeitung natürlicher Sprache (z. B. Parsing) sind bisher nicht vorhanden.
- Selbst wenn das Problem der Verarbeitung natürlicher Sprache gelöst wäre, würden für die Interpretation der Resultate weitere Verfahren zur Integration von “Weltwissen” benötigt. (vergl. Abbildung 15)
- Sie benötigen Methoden um Wissensinhalte zu vergleichen.

### 2.2.3 Klassifikationen

(vergl. auch Fuhr 1995)

Wir kommen nun zum zweiten Repräsentationsansatz, mit dem man versuchen kann inhaltliche Ähnlichkeiten vergleichbar zu machen, zu den Methoden, die die zulässigen Mittel zur inhaltlichen Beschreibung einschränken bzw. vorgeben.

Mit **Klassifikationen** werden Wissensgebiete nach einem formalen Schema strukturiert. Sie sind insbesondere dann notwendig, wenn physikalische Objekte systematisch angeordnet werden sollen. So kann ein Buch in einer Bibliothek (sinnvollerweise) nur an einem Platz stehen, oder ein Bild in einer Austellung nur in einem Saal hängen. Solche Klassifikationen können damit natürlich nur einer bestimmten Sichtweise gerecht werden. Wie das Beispiel der Bibliothek zeigt, müssen für andere Sichtweisen Verweissysteme, wie z. B. ein alphabetischer Katalog nach Autoren, überlagert werden.

Klassifikationen können als Baumstrukturen realisiert werden, bei denen die verschiedenen Ebenen unterschiedliche Abstraktionsniveaus darstellen können. Die tatsächlichen Objekte sind die Blätter der Struktur. Die Bezeichnung ihrer Kategorie entspricht dem Pfad von der Wurzel bis zu dem entsprechenden Blatt.

#### 2.2.3.1 Universelle Dezimalklassifikation

Ein solches Klassifikationssystem ist die Universelle Dezimalklassifikation für Buchbestände, die einen sachgebietsorientierten Zugang ermöglicht. Wie der Name schon sagt, ist diese Klassifikation an das Dezimalsystem angepaßt, d. h. sie kann auch als Baum mit einem Verzweigungsgrad 10 dargestellt werden.

Zunächst werden 10 Hauptabteilungen definiert (vergl. Abbildung 16). Jede dieser Abteilungen ist ihrerseits wieder in (bis zu) 10 Abteilungen untergliedert usw.

**Abbildung 16: Die 10 Hauptabteilungen der Universellen Dezimalklassifikation (zitiert nach Fuhr 1995)**

| |
|---|
| 0 Allgemeines |
| 1 Philosophie |
| 2 Religion, Theologie |
| 3 Sozialwissenschaften, Recht, Verwaltung |
| 4 (Zur Zeit nicht belegt) |
| 5 Mathematik, Naturwissenschaften |
| 6 Angewandte Wissenschaften, Medizin, Technik |
| 7 Kunst, Kunstgewerbe, Photographie, Musik, Spiel, Sport |
| 8 Sprachwissenschaft, Philologie, Schöne Literatur, Literaturwissenschaft |
| 9 Heimatkunde, Geographie, Biographien, Geschichte |

So gelangt man zu sehr speziellen Klassifikationen, wie sie in Abbildung 17 dargestellt sind.

An jedem Knoten dieser Klassifizierungen muß eine Entscheidung getroffen werden, die in vielen Fällen diskutierbar sein wird.

### 2.2.4 Thesauri

In gewisser Weise bilden Thesauri das sprachliche oder terminologische Pendant zu Klassifikationen. Sie haben vor allem zwei Funktionen:

- Sie definieren ein kontrolliertes Vokabular.
- Sie stellen Beziehungen zwischen den Termen dieses Vokabulars her.

Dabei werden allerdings nicht nur hierarchische Systeme definiert, wie bei der Klassifikation, sondern es können eine Vielzahl von weiteren Beziehungen zwischen Wörtern dargestellt werden.

Nach DIN 1463 ist ein Thesaurus eine geordnete Zusammenstellung von Begriffen mit ihren (natürlichsprachlichen) Beziehungen (zitiert nach Fuhr 1995). Man sollte dabei allerdings zwischen allgemeinen Thesauri und den in einem IR System verwendeten Thesauri unterscheiden.

**Abbildung 17: Ein Pfad durch die Universelle Dezimalklassifikation (nach Fuhr 1995)**

| | |
|---|---|
| 3 | Sozialwissenschaften, Recht, Verwaltung |
| 33 | Volkswirtschaft |
| 336 | Finanzen. Bank- und Geldwesen |
| 336.7 | Geldwesen. Bankwesen. Börsenwesen |
| 336.76 | Börsenwesen. Geldmarkt. Kapitalmarkt |
| 336.763 | Wertpapiere. Effekten |
| 336.763.3 | Obligationen. Schuldverschreibungen |
| 336.763.31 | Allgemeines |
| 336.763.311 | Verzinsliche Schuldbriefe |
| 336.763.311.1 | Langfristige verzinsliche Schuldbriefe |

Ein **allgemeiner Thesaurus** listet zu jedem Wort, neben einer kurzen Definition des Begriffs, verschiedene Relationen zu Gruppen von anderen Wörtern auf:

- Synonyme
- Antonyme (Gegensatz)
- Verwandte Wörter
- Oberbegriffe

Bei mehrdeutigen Wörtern werden diese Gruppen für jede Bedeutung einzeln aufgelistet. Ein klassisches Beispiel eines allgemeinen Thesaurus ist “Roget’s Thesaurus” von 1852, in dem die Wörter der englischen Sprache “nach den Ideen, die sie ausdrücken” (Vorwort der Originalausgabe) sortiert sind. Solche Thesauri sollen vor allem dazu dienen, Kreativität und Vielfalt bei der Wortwahl zu unterstützen (oder vorzutäuschen). Sie werden teilweise auch von Textverarbeitungsprogrammen angeboten.

Im Gegensatz dazu steht bei den meisten **Thesauri in IR-Systemen** die Klassifikation der Dokumente der Datenbank im Vordergrund. Kernstück eines solchen Thesaurus ist ein **kontrolliertes Vokabular** mit dem vor allem die Probleme der Synonymie und Polysemie in den Griff bekommen werden sollen. In diesen Kernbestand von zugelassenen Deskriptoren werden nur genau ausgewählte Terme aufgenommen, die gleichzeitig einen Teilbereich des Sachgebiet beschreiben. Sie können aus einem oder mehreren Wörtern bestehen. Nur diese Terme dürfen bei der Indexierung mit dem Thesaurus verwendet werden. Sie sind hierarchisch gegliedert, d. h. zu jedem Term ist ein Oberbegriff und eine Reihe spezifischere Begriffe angegeben. Durch diese Gliederung wird wie bei der Klassifikation eine Baumstruktur erzeugt, in die die Dokumente eingeordnet werden können. Im Unterschied zur Klassifikation können einem Dokument aber i. a. mehrere Deskriptoren auch aus dem selben Baum zugeordnet werden, sodaß es nicht mehr nur in einer Klasse vorkommen kann.

Die Struktur der Deskriptoren kann weiter kompliziert werden, indem zu einem Begriff mehrere Oberbegriffe zugelassen werden. Das kann sinnvoll sein, wenn verschiedene Kriterien für die Bildung von Ober– und Unterbegriffen verwendet werden. Die Systematik, die durch die Deskriptoren erzeugt wird ist also allgemeiner als z. B. die Dezimalklassifikation. Das führt zu einer größeren Flexibilität, die allerdings nur genutzt werden kann, wenn die entsprechenden Retrievalwerkzeuge zur Verfügung stehen. So

ermöglicht es die hierarchische Struktur der Deskriptoren alle Unterbegriffe eines Begriffes bei einer Suche mit einzubeziehen.

Wären im Beispiel aus Abbildung 11 die Terme “Vertebrates” und “Invertebrates” in diesem Sinne als Deskriptoren verwendet worden, so hätten die Tiere als Unterbegriffe einbezogen werden können. Andererseits wären in einem biologischen Sinn aber auch die Menschen als Versuchspersonen unter die “Vertebrates” (Wirbeltiere) gefallen. Mit einem Deskriptor “Animal” der Tiere im Gegensatz zu Menschen bezeichnet hätte man mehr Glück haben können, er wäre aber dann wieder nicht mit den “Vertebrates” und “Invertebrates” kompatibel gewesen. Das Beispiel zeigt in einem sehr einfachen Fall, daß Thesauri selten universell definiert werden können, sondern am besten für verschiedene Fachgebiete jeweils spezifisch angepaßt werden. Das führt allerdings dazu, daß Suchende in jedem Fachgebiet zunächst den entsprechenden Thesaurus “lernen” müssen.

Neben der hierarchischen Kernstruktur der zugelassenen Deskriptoren kann ein Thesaurus weiter Begriffe und Relationen enthalten. Als Relationen können z. B. verwandte Deskriptoren angegeben werden. Als weiter Terme können Begriffe aufgenommen werden, die häufig im Gebiet, für das der Thesaurus konstruiert wird, verwendet werden, aber nicht als Deskriptoren zugelassen sind. Bei diesen Begriffen wird dann angegeben welche Deskriptoren statt des Begriffs verwendet werden können (bzw. sollten). Weiter finden sich verschiedene Auflistungsformen, wie z. B. ein Rotated Index, bei dem Deskriptoren, die aus mehreren Termen bestehen, unter jedem der Terme alphabetisch zu finden sind.

Bisher wurden Thesauri i. a. “von Hand” erstellt; meistens in Zusammenhang mit bibliographischen Arbeiten, wie Bibliographien, Bibliotheks- oder Museumskatalogen, Abstraktsammlungen und bibliographischen Informationsdiensten. Dabei arbeiten meist mehrere Personen oder auch Institutionen zusammen, die mit dem Fachgebiet, für das der Thesaurus konstruiert werden soll, befaßt sind. Diese Arbeiten sind sehr aufwendig, damit teuer und auch langsam. Prallel dazu gibt es verschiedenen Versuche Thesauri automatisch auf der Basis von großen Korpora aus dem entsprechenden Gebiet zu erstellen. Solche automatisch generierten Thesauri weisen meistens nicht die gleiche strenge Struktur auf, wie manuell bzw. intellektuell konstruierte. Sie haben dafür den Vorteil, daß sie billiger, schneller und stärker auf eine Domäne spezialisier sein können. Zudem kann ihre Struktur klarer auf die Datengrundlage zurückgeführt werden als die im Laufe vieler Jahre gewachsenen manuellen Thesauri.

## 2.3 Vektorraummodelle

Im Kapitel 2.1 über Bool'sches Retrieval hatten wir Retrievalverfahren betrachtet, die auf dem Auftauchen von Termen in Texten oder – allgemeiner – von Attributwerten in Dokumenten aufgebaut sind. Im letzten Kapitel 2.2 haben wir näher untersucht, was alles ein Term sein kann. Dabei war aber immer noch davon ausgegangen worden, daß ein Term in einem Dokument entweder auftaucht oder nicht. In diesem Kapitel geht es nun um Verfahren, die die Terme, die in einem Dokument auftreten, gewichten. Das kann zum einen dadurch geschehen, daß den Termen von vornherein unterschiedliche Gewichtungen gegeben werden, zum anderen können aus der Art des Auftretens in den Dokumenten unterschiedliche Gewichtungen abgeleitet werden.

### 2.3.1 Das Modell

Zunächst werden wir das Vektorraummodell allgemein definieren; es geht von einem festen Vokabular an Termen aus:

***Definition 2.3.1.1 Vektorraummodell:*** *Sei $T = \{t_1, ..., t_n\}$ eine endliche Menge von Termen und $D = \{d_1, ..., d_m\}$ eine Menge von Dokumenten. Für jedes Dokument $d_i \in D$ sei zu jedem Term $t_k \in T$ ein Gewicht $w_{i,k} \in \mathbb{R}$ gegeben. Die Gewichte des Dokumentes $d_i$ lassen sich zu einem Vektor $w_i = (w_{i,1}, ..., w_{i,n}) \in \mathbb{R}^n$ zusammenfassen. Dieser Vektor beschreibt das Dokument im Vektorraummodell: er ist seine Repräsentation und wird* ***Dokumentvektor*** *genannt.*

*Auch Anfragen werden durch Vektoren $q \in \mathbb{R}^n$ dargestellt. Wie bei der Repräsentation der Dokumente wird die Anfrage durch eine Menge gewichteter Terme, den* ***Anfragevektor*** *oder* ***Queryvektor****, dargestellt.*

*Zwischen zwei Dokumenten $x, y \in \mathbb{R}^n$ sei eine* ***Ähnlichkeitsfunktion*** *$s : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ definiert.*

Der Ähnlichkeitswert, der für jedes Vektorenpaar mit der Ähnlichkeitsfunktion berechnet werden kann, kann benutzt werden, um zu einem vorgegebenen Vektor ähnliche Vektoren zu bestimmen. Auf diese Weise können Dokumente untereinander und Anfragen und Dokumente miteinander verglichen werden. Insbesondere können zu einer Anfrage, bzw. ihrem Anfragevektor die ähnlichen Dokumente in der Datenbank gefunden werden.

### 2.3.2 Vektorraummodell und Bool'sches Retrieval

In diesem Modell kann man auch die einfachen Operationen des Bool'sche Retrieval als Spezialfälle darstellen. Dadurch läßt sich der Zusammenhang zwischen den beiden Verfahren verdeutlichen. Zunächst erlaubt man als Gewichte nur die Werte $0$ und $1$, also $w_i \in \{0, 1\}^n$ wobei $w_{i,j} = 1$ bedeutet, daß Term $j$ im Dokument $i$ auftritt, und $w_{i,j} = 0$ bedeutet, daß Term $j$ im Dokument $i$ nicht auftritt.

Dasselbe Verfahren verwendet man, um eine Anfrage aus mit AND verknüpften Termen zu repräsentieren: An den Stellen der Terme, die in der Anfrage vorkommen, wird in einem Anfragevektor $q \in \{0, 1\}$ eine $1$ gesetzt, an den anderen Stellen eine $0$. Die zur Anfrage gehörenden Dokumente sind dann genau die, bei denen an allen Stellen, an denen im Anfragevektor eine $1$ steht, auch eine $1$ steht.

Um eine Anfrage aus mit OR verknüpften Termen zu bearbeiten, kann man zunächst genauso verfahren; nur sind dabei Dokumente bereits zur Anfrage gehörend, wenn an einer einzigen Stelle, an der in der Anfrage eine $1$ steht, auch eine $1$ steht.

Man kann diese Operationen mit dem Skalarprodukt der Vektoren darstellen:

***Definition 2.3.2.1 Skalarprodukt:*** *Seien $w_i = (w_{i,1}, ..., w_{i,n}) \in \mathbb{R}^n$ und $q = (q_1, ..., q_n) \in \mathbb{R}^n$ zwei Vektoren, dann ist das Skalarprodukt zwischen ihnen definiert durch*

$$w_i \cdot q = \sum_{k=1}^{n} w_{i,k} q_k \qquad (4)$$

*Das Skalarprodukt ist also eine Abbildung $\mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$*

Eine aus $r$ durch AND verknüpften Termen bestehende Anfrage kann jetzt mithilfe des Skalarproduktes so ausgedrückt werden: Das Dokument mit dem Gewichtsvektor $w_i = (w_{i,1}, ..., w_{i,n}) \in \mathbb{R}^n$ gehört zur Anfrage $q = (q_1, ..., q_n) \in \mathbb{R}^n$ (ist ihr ähnlich) wenn $w_i \cdot q = r$ gilt. Formal läßt sich das so schreiben:

$$s_{AND}(w_i, q) = \begin{cases} 1 & falls\ w_i \cdot q = \sum_{k=1}^{n} q_k \\ 0 & sonst \end{cases} \qquad (5)$$

die Antwortmenge definiert sich über das Attribut $S_{AND,q}(w) = s_{AND}(w, q)$ als $D_{AND,q} = S^{-1}_{AND,q}(\{1\})$

Entsprechend gehört ein Dokument mit dem Gewichtsvektor $w_i = (w_{i,1}, ..., w_{i,n}) \in \mathbb{R}^n$ zu einer aus $m$ durch OR verknüpften Termen bestehenden Anfrage, wenn $w_i \cdot q \geq 1$ gilt:

$$s_{OR}(w_i, q) = \begin{cases} 1 & falls\ w_i \cdot q \geq 1 \\ 0 & sonst \end{cases} \qquad (6)$$

Die Antwortmenge ergibt sich über das Attribut $S_{OR,q}(w) = s_{OR}(w, q)$ als $D_{OR,q} = S^{-1}_{OR,q}(\{1\})$.

Die Bool'schen Operationen bestehen also bei dieser Konstruktion im Kern aus dem Skalarprodukt und einer Schwellenfunktion, die die Aufgabe hat, das gewünschte binäre Resultat zu erzeugen. Bei AND ist die Schwelle maximal, bei OR ist sie minimal.

Läßt man die Schwellenfunktion weg, erhält man einen durch das Skalarprodukt gegebenen Ähnlichkeitswert. Er gibt an, wieviele Terme in der Anfrage und in dem jeweiligen Dokument übereinstimmen. Durch den Ähnlichkeitswert können die Dokumente in eine Rangreihe gebracht werden; d. h. die Dokumente, die in vielen Termen mit der Anfrage übereinstimmen, werden zuerst ausgegeben, die, die in nur wenigen Dokumenten übereinstimmen, am Ende. Die Ausgabe der Resultate in einer solchen Rangreihe ist in einem Information Retrieval System sehr viel nützlicher als eine Ausgabe als eine ungeordnete Menge, wenn sie denn "richtig" ist. Sie bietet den Nutzenden zuerst die wichtigsten Informationen an und sortiert weniger wichtige Informationen eher ans Ende der Ausgabe.

Die geschilderten Verfahren sind natürlich nicht auf die Wertemenge $\{0, 1\}$ beschränkt. Das Skalarprodukt kann auch mit reellen Zahlen berechnet werden. In diesem Fall sind die Rangreihen natürlich feiner untergliedert als bei ganzen Zahlen.

### 2.3.3 Ähnlichkeitsfunktionen

Bei der im vorigen Abschnitt geschilderten Ähnlichkeitsfunktion, die das Skalarprodukt benutzt, hängen die Ähnlichkeitswerte stark von der Anzahl der Werte im Vektor, die ungleich $0$ sind, also der Anzahl der Terme in der Anfrage (oder im Dokument) ab. Der Maximalwert, den das Skalarprodukt zwischen einem Anfragevektor und den Dokumentvektoren einer Dokumentmengen annehme kann, ist durch die Anzahl der Terme in der Anfrage begrenzt.

Längere Dokumente haben statistisch gesehen daher bei langen Anfragen größere Chancen hohe Ähnlichkeitswerte zu bekommen als kurze. Andererseits ist z. B. bei Literaturdatenbanken die Länge des Abstracts häufig durch äußere Restriktionen bestimmt und nicht so sehr durch die Wichtigkeit des Artikels. Einflüsse dieser Restriktionen würden beim Skalarprodukt direkt auf die Ähnlichkeitswerte durchschlagen und damit auf die Retrievalergebnisse.

Um die Eigenschaften verschiedener Ähnlichkeitsmaße zu vergleichen, kann man sich überlegen, wie die Vektoren verteilt sind, die zu einem gegebenen Vektor $w_i$ die gleiche Ähnlickeit haben. Sie sind in Jones & Furnas (1987) für einige Ähnlichkeitsmaße dargestellt.

Beim Skalarprodukt liegen diese Vektoren auf einer Hyperebene, die parallel zu derjenigen ist, die senkrecht auf dem Vektor steht. Im zweidimensionalen Fall ist das eine Gerade, die senkrecht (rechtwinklig) zu dem Vektor ist.

Sei $(a, b)$ der Referenzvektor, zu dem der Vektor $(x, y)$ die Ähnlichkeit $c$ habe; dann gilt

$$ax + by = c \tag{7}$$

und folglich

$$y = -\frac{a}{b}x + \frac{c}{b} \tag{8}$$

Für $c = 1$ ergibt sich die Senkrechte auf dem Vektor.

Das Skalarprodukt ist linear, d. h. unter anderem, daß ein Faktor auf einen der Vektoren sich als Faktor auf die Ähnlichkeit auswirkt (daher die parallelen Hyperebenen). Die Ähnlichkeit, die ein Vektor zu sich selbst hat ist deshalb gerade das Quadrat seiner euklidischen Länge. Das heißt in einem Information Retrieval System, das mit dem Vektorraummodell und dem Skalarprodukt als Ähnlichkeitsmaß arbeitet, daß die euklidische Länge der Dokumentvektoren, wie oben beschrieben, einen starken Einfluß auf die Ähnlichkeitswerte beim Vergleich mit einem Anfragevektor hat. Um dies auszugleichen, kann man Maße verwenden, die das vermeiden.

#### 2.3.3.1 Das Cosinusmaß

Ein solches Maß ist das Cosinusmaß (der Cosinus im $\mathbb{R}^n$ vergl Abb. 18):

$$cos(w_i, q) = \frac{\sum_{k=1}^{n} w_{i,k} q_k}{\sqrt{\sum_{k=1}^{n} w_{i,k}^2} \sqrt{\sum_{k=1}^{n} q_k^2}} \tag{9}$$

Wie der Cosinus im Eindimensionalen liegen die Ähnlichkeitswerte zwischen zwei Vektoren immer im Intervall $[-1, 1]$. Die beiden Wurzeln im Nenner sind gerade die euklidischen Längen der Vektoren $w_i$ und $q$. Man kann sich die Formel also auch als

einfaches Skalarprodukt der normierten Vektoren

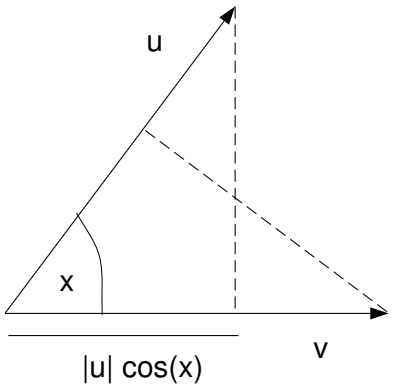
$$\frac{w_{i,k}}{\sqrt{\sum_{k=1}^{n} w_{i,k}^2}} \tag{10}$$

und

$$\frac{q_k}{\sqrt{\sum_{k=1}^{n} q_k^2}} \tag{11}$$

vorstellen. Das zeigt auch schon den Grund für die Wahl des Maßes an. Die Ähnlichkeitswerte sind unabhängig von der euklidischen Länge der Vektoren, also der Anzahl und Größe der von $0$ verschiedenen Einträge. Sie hängen lediglich von der "Richtung" der Vektoren ab, also dem Verhältnis der Vektoreinträge untereinander. Werden nun zwei Vektoren mit dem Cosinusmaß verglichen, so ist ihre Ähnlichkeit am größten, wenn sie die gleiche Richtung haben, unabhängig von ihrer Länge, d. h. wenn sie sich nur durch einen positiven Faktor voneinander unterscheiden. Die Vektoren mit gleicher Ähnlichkeit zu einem Referenzvektor sind also die, die den selben Winkel mit dem Referenzvektor bilden. Im zweidimensionalen sind das zwei Strahlen, die symmetrisch zum Referenzvektor von $(0, 0)$ ausgehen.

**Abbildung 18: Der Cosinus im $\mathbb{R}^2$**




### 2.3.3.2 Das Pseudo-Cosinusmaß

Beim Pseudo-Cosinusmaß sind die euklidischen Längen durch die Summe der Vektoreinträge (die $L_1$ Länge) ersetzt.

$$s_p(w_i, q) = \frac{\sum_{k=1}^{n} w_{i,k} q_k}{\left(\sum_{k=1}^{n} w_{i,k}\right)\left(\sum_{k=1}^{n} q_k\right)} \tag{12}$$

In diesem Sinne kann auch dieses Maß als Skalarprodukt ($L_1$–) normierter Vektoren aufgefaßt werden. Solange die Vekoreinträge bzw. Gewichte $w_{i,j}$ alle positiv sind, reagiert diese Normierung vor allem weniger stark auf große Werte unter den Einträgen. Die Vektoren mit gleicher Ähnlichkeit zu einem Referenzvektor sind hier auch radiale Strahlen aus $(0, 0)$ allerdings nicht mehr durch den Winkel charakterisiert. Im zweidimensionalen Fall ergibt sich folgendes: Sei wieder $(a, b)$ der Referenzvektor, zu dem

der Vektor $(x, y)$ die Ähnlichkeit $c$ habe, dann gilt:

$$\frac{ax + by}{(a + b)(x + y)} = c \tag{13}$$

$$ax + by = c(a + b)(x + y) \tag{14}$$

$$y = \frac{(c - 1)a + b}{a - (c - 1)b}x \tag{15}$$

Für den Fall $c = 1$ folgt

$$y = \frac{b}{a}x \tag{16}$$

Für andere Werte von $c$ zeigt sich aber, daß das Verhalten nicht symmetrisch zum Referenzvektor ist wie beim Cosinus. Setzt man größere Werte von $c$ ein, so wandert der Strahl mit den Vektoren mit Abstand $c$ in Richtung auf die Achse, die den kleineren Winkel zum Referenzvektor hat. Im Grenzwert wird der Ähnlichkeitswert am größten für den Strahl, der in $(0, 0)$ senkrecht auf dem Referenzvektor steht. Läßt man nur positive Einträge in den Vektoren zu, so ist die Ähnlichkeit am größten für einen Vektor, der genau in die Richtung der Achse der größten Komponente des Vektors zeigt, also nur für diese Komponente einen Eintrag ungleich $0$ besitzt.

Ein solches Verhalten ist im Allgemeinen nicht wünschenswert.

#### 2.3.3.3 Das Dice-Maß

$$s_d(w_i, q) = \frac{2\sum_{k=1}^{n} w_{i,k}q_k}{\sum_{k=1}^{n} w_{i,k} + \sum_{k=1}^{n} q_k} \tag{17}$$

Das Dice-Maß kann nicht mehr als Skalarprodukt normierter Vektoren aufgefaßt werden, weil die Summen im Nenner addiert und nicht multipliziert werden. Betrachtet man wieder die Vektoren gleicher Ähnlichkeit, so ergibt sich im $\mathbb{R}^2$ folgendes Verhalten:

$$\frac{2(ax + by)}{a + b + x + y} = c \tag{18}$$

$$2ax + 2by = c(a + b + x + y) \tag{19}$$

$$y = \frac{(c - 2a)}{(2b - c)}x + \frac{c(a + b)}{(2b - c)} \tag{20}$$

d. h. die größten Ähnlichkeitswerte ergeben sich für die Gerade $y = -x - (a + b)$. Auch dieses Verhalten ist nicht unbedingt wünschenswert.

#### 2.3.3.4 Das Overlap-Maß

Der Vollständigkeit halber wird noch die Formel des Overlap-Maßes angegeben. Durch die Minima, die in der Formel auftreten, sind die Bilder, wie sie in Jones & Furnas (1987) dargestellt sind, nicht mehr durch Geraden dominiert, sondern es treten komplexere Gebiete gleicher Ähnlichkeit auf, die durch die Fallunterscheidungen bei der Minimumbildung verursacht werden.

$$s_o(w_i, q) = \frac{\sum_{k=1}^{n} \min(w_{i,k}, q_k)}{\min\left(\sum_{k=1}^{n} w_{i,k}, \sum_{k=1}^{n} q_k\right)} \tag{21}$$

#### 2.3.3.5 Das Jaccard-Maß

Schließlich soll noch das Jaccard-Maß angegeben werden. Es läßt sich im Sinne des Bool'schen Retrieval interpretieren.

$$s_J(w_i, q) = \frac{\sum_{k=1}^{n} w_{i,k} q_k}{\sum_{k=1}^{n} w_{i,k} + \sum_{k=1}^{n} q_k - \sum_{k=1}^{n} w_{i,k} q_k} \tag{22}$$

Geht man wieder von der Repräsentation der Dokumente und Anfragen mit Vektoren aus $\{0,1\}^n$ aus, so steht im Zähler gerade die Anzahl der Terme aus der Anfrage, die im Dokument vorkommen, und im Nenner die Anzahl der Terme, die in der Anfrage oder im Dokument vorkommen.

Bei der Betrachtung der Orte mit den höchsten Ähnlichkeitswerten muß beachtet werden, daß natürlich nur die vorhandenen Dokumente mit einer Anfrage verglichen werden. Ist die Teilmenge des $\mathbb{R}^N$, in dem die Dokumentvektoren tatsächlich liegen, geeignet gewählt (bzw. wird sie von den Dokumentvektoren geeignet erzeugt), können auch diese Maße durchaus sinnvoll sein.

### 2.3.4 Gewichtungsmethoden

In letzten Abschnitt hatten wir verschiedenen Ähnlichkeitsfunktionen für Dokument- und Anfragevektoren betrachtet, ohne zu sagen, woher die Gewichte, die in den Vektoren stehen, kommen. Lediglich im Fall des Bool'schen Retrievals hatten wir mit Werten aus $\{0,1\}$ gearbeitet, jenachdem ob ein Term in einem Dokument oder einer Anfrage vorkam oder nicht. In diesem Abschnitt wollen wir Methoden ansprechen, mit denen Gewichtsvektoren für Dokumente und Anfragen bestimmt werden können.

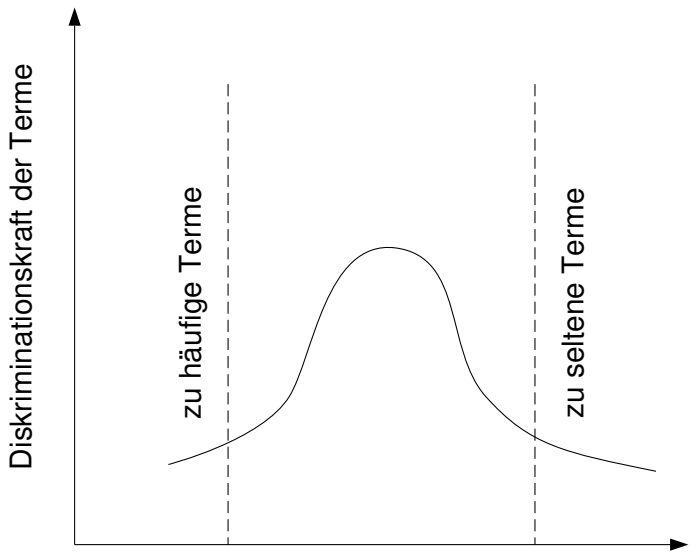
Zunächst können Gewichtungen von Termen von Menschen eingegeben werden. Das kann zum einen bei der Indexierung geschehen, insbesondere bei der Vergabe von Termen aus einem Thesaurus oder bei der Vergabe von freien Schlüsselwörtern. Dabei stellt sich vor allem das Problem der Konsistenz der Gewichtungen von verschiedenen Indexierenden. Zum anderen können bei einer Anfrage die Terme von den Anfragenden gewichtet werden. Diese Gewichtungen können durch **Feedbackmethoden** verfeinert werden. In beiden Fällen sind die Gewichtungen von dem jeweiligen Kontext abhängig, also bei der Indexierung von dem Objekt bzw. Dokument, das indexiert wird, bei der Anfrage von dem Informationsbedürfnis der Anfragenden.

Andererseit gibt es auch Kriterien für die Gewichtung von Termen, die nicht vom Kontext abhängig sind. So sind allgemeine Terme, die nicht an ein bestimmtes Fachgebiet

gebunden sind (“Ergebnis, Methode, Verfahren, Zusammenfassung, ...”), schlechte Suchterme, während Terme, die nur in bestimmten Wissensgebieten vorkommen, gute Terme sein sollten.

In einer ersten Näherung an dieses Problem kann man die Häufigkeit von Termen in Dokumenten betrachten. Terme, die in sehr vielen Dokumenten vorkommen, haben eine schlechtere Diskiminationskraft als Terme die in weniger Dokumenten vorkommen. Andererseits werden mit Termen, die nur in sehr wenigen Dokumenten vorkommen im allgemeinen keine umfassenden Suchergebnisse erziehlt werden können (vergl. Abbildung 19).

**Abbildung 19: Schematische Darstellung der Diskriminationskraft von Termen gegen die Häufigkeit aufgetragen**




Neben dieser allgemeinen Häufigkeit kann versucht werden den Kontext eines Dokumentes zu berücksichtigen. Eine Annäherung an die Wichtigkeit eines Termes in einem Dokument kann durch seine Häufigkeit in dem Dokument versucht werden.

Im Folgende werden Beispiele von Formeln angegeben, die in experimentellen Systemen verwendet wurden. Dabei bezeichnet $h(i,j)$ die Häufigkeit von Term $t_j$ im Dokument $d_i$, $d(j)$ die Anzahl der Dokumente in denen Term $t_j$ vorkommt, $D = (d_1, ..., d_m)$ wieder die Menge der Dokumente, und $T = (t_1, ..., t_n)$ die der Terme.

### 2.3.4.1 Häufigkeitsformeln

Formeln, die die Häufigkeit eines Terms in einem Dokument berücksichtigen:

$$h(i,j) \tag{23}$$

$$\frac{h(i,j)}{d(j)} \tag{24}$$

$$\frac{h(i,j)}{1+h(i,j)} \tag{25}$$

$$K + (1 - K)\frac{h(i,j)}{\max\limits_{l \in \{1,\ldots n\}} h(i,l)} \tag{26}$$

#### 2.3.4.2 Inverted Dokument Frequency

Formeln, die die Häufigkeit eines Terms in der Dokumentenmenge berücksichtigen (**IDF = Inverted Document Frequency**):

$$\ln\left(\frac{m}{d(i)}\right) \tag{27}$$

$$\ln\left(\frac{m - d(i)}{d(i)}\right) \tag{28}$$

#### 2.3.4.3 Häufigkeits- und IDF Formeln

$$\widetilde{w_{i,j}} = \frac{1}{2}\left(1 + \frac{h(i,j)}{\max\limits_{k \in \{1,\ldots,n\}} \{h(i,k)\}}\right) \ln\left(\frac{m}{d(i)}\right) \tag{29}$$

bzw. die normierte Form davon:

$$w_{i,j} = \frac{\widetilde{w_{i,j}}}{\sqrt{\sum\limits_{k=1}^{n} \widetilde{w_{i,k}}^2}} \tag{30}$$

Diese Formel wurde im experimentellen Retrieval System SMART (Salton & McGill 1983) für die automatische Gewichtung von Termen verwendet.

## 2.4 Evaluationsverfahren

Am Ende der in der Einleitung gegebenen Beschreibung des Information Retrievals durch die Fachgruppe Information Retrieval der Gesellschaft für Informatik hatte es geheißen: “Aus dieser Problematik ergibt sich die Notwendigkeit zur Bewertung der Qualität der Antworten eines Informationssystems, wobei in einem weiteren Sinne die Effektivität des Systems in bezug auf die Unterstützung des Benutzers bei der Lösung seines Anwendungsproblems beurteilt werden sollte.”

Will man diesem Anspruch gerecht werden, so lassen sich IR Systeme nur mit ungeheurem Aufwand beurteilen und vergleichen: Man müßte eine repräsentative Auswahl von Anwendungsproblemen und Benutzenden zur Verfügung haben und den Einfluß des IR Systems auf die Lösung der Anwendungsprobleme isolieren und bewerten können. Eine solche Bewertung ist im allgemeinen weder theoretisch noch praktisch möglich. Die meisten Evaluierungen beschränken sich deshalb auf Evaluierungen anhand von einigen meist kleinen Dokument- und Anfragesammlungen. Bei diesen Methoden bleiben allerdings zwangsläufig eine ganze Reihe von Einflußfaktoren unberücksichtigt.

### 2.4.1 Einflußfaktoren

Zunächst soll deshalb eine Aufzählung von Einflußfaktoren, wie sie bei Salton und McGill (1983) zu finden ist, angegeben werden.

- Auswahl und Eingabe der Dokumente (*input policies*) bestimmen die Ausrichtung der Inhalte der Datenbank, die "Tiefe" der Behandlung von Themen, die Aktualität der Dokumente und die Fehlerrate in den Dokumenten
- Die Dokumentformate (*physical input form*) bestimmen, in welcher Form Dokumente gespeichert werden und was sie beinhalten können. Das kann z. B. die Länge der Dokumente betreffen, also ob nur Titel, Abstracts oder ganze Artikel gespeichert werden. Es kann auch beinhalten, in welchen Formaten Bilder, Tondokumente oder andere multimediale Dokumente abgelegt werden. Außerdem können dadurch die Repräsentation von Dokumenten und damit auch der Zugriff beim Suchen festgelegt werden.
- Die Indexierungsmethode (*indexing language*) gibt ganz allgemein die Repräsentation der Dokumente an. Spezieller kann sie z. B. ein kontrolliertes Vokabular und Regeln für die Indexierung angeben. Dabei muß auch festgelegt werden, wie spezifisch bzw. wie breit die Dokumente indexiert werden sollen. Ob also jeweils die spezifischsten Terme, die ein Dokument noch beschreiben, verwendet werden sollen, um einen genauen Zugriff zu ermöglichen, oder eher allgemeinere Terme, um das Dokument auch noch zu finden, wenn mit allgemeineren Termen gesucht wird.
- Wird die Indexierung von Hand vorgenommen, so ist auch der Indexierungsvorgang (*indexing operation*) ein Einflußfaktor. Die Erfahrungen und Interessen, ebenso wie die Motivation der Indexierenden, beeinflußen die Konsistenz der Indexierung durch verschiedene Indexierende.
- Die Anfrageformulierung durch die Nutzenden ist von vielen Faktoren abhängig. Unter anderem vom fachlichen Kenntnisstand der Anfragenden und von ihrer Vertrautheit mit dem IR System.
- Bei der Suche selbst (*search operation*) sollten neben den verschiedenen Methoden auch Präferenzen der Anfragenden berücksichtigt werden, z. B. ob sie mehr Wert auf eine vollständige Abdeckung der Frage legen oder auf eine überschaubare Antwortmenge.
- Schließlich entscheidet auch die Präsentation der Ergebnisse mit darüber, ob die Anfragenden Nutzen aus der Recherche ziehen können. Insbesondere bei Systemen mit Feedback, bei denen also mehrere aufeinander aufbauende Fragen gestellt werden können, ist es wichtig, daß die Anfragenden die dargebotenen Dokumente optimal erfassen und beurteilen können.

Schon diese unvollständige Aufzählung zeigt, daß eine systematische Variation all dieser Einflußfaktoren kaum zu bewältigen ist.

Neben dieser Aufteilung der Evaluation nach den verschiedenen Systemkomponenten kann man auch eine Aufteilung nach den verschiedenen Bewertungskriterien vornehmen. Dabei kann man zwischen Effizienz und Effektivität unterscheiden.

Unter **Effizienz** versteht man dabei den möglichst sparsamen Umgang mit Resourcen wie Rechenzeit und Speicherplatz. **Effektivität** beschreibt die Fähigkeit des Systems den Nutzenden die benötigte Information bei möglichst geringen Kosten an Zeit und Anstrengung anzubieten. Während die Frage nach der Effizienz bei genügend genauer Analyse der verwendeten Algorithmen weitgehend analytisch und durch Benchmarktests geklärt werden kann, ist man für die Ermittlung der Effektivität auf empirische

Methoden angewiesen. Wir werden uns im Folgenden nur mit der Frage der Effektivität beschäftigen.

### 2.4.2 Relevanz

Das zentrale Problem bei der Beurteilung von Retrievalergebnissen ist, daß, um die Antwort des Systems zu bewerten, die richtige Antwort bekannt sein muß. Konkret: Um zu überprüfen, ob zu einer Anfrage die richtigen Dokumente gefunden wurden, muß bekannt sein, welche Dokumente in der Datenbank vorhanden sind, die zu der Anfrage "gehören". Dazu verwendet man das Konstrukt der **Relevanz**, einer Beziehung die zwischen einer Anfrage und einem Dokument besteht. Salton und McGill (1983) zitieren dazu eine Studie von Cuadra und Katter:

> *relevance is the correspondence in context between an information requirement statement (a query) and an article (a document), that is, the extend to which the article covers the material that is appropriate to the requirement statement*

Diese Definition läßt natürlich die entscheidenden Fragen offen. Denn die Frage, wann ein Dokument oder ein Artikel "das Material abdeckt, das für die Anfrage angemessen ist", ist ja nichts weiter als eine neue Umschreibung des eigentlichen Problems des Information Retrievals. Tatsächlich wird in der Praxis auch keine Definition von Relevanz benutzt, wenn es darum geht, zu bestimmen, welche Artikel zu einer Anfrage relevant sind, sondern es werden Personen gebeten, die Relevanz einzuschätzen. Man verwendet also den intuitiven, umgangsprachlichen oder naiven Begriff, um Relevanz zu bestimmen (oder zu operationalisieren, wie die Psychologen sagen würden). So gesehen sind Evaluierungen von Retrieval Systemen Untersuchungen, die versuchen festzustellen, inwieweit das System zu den gleichen Ergebnissen kommt wie Versuchspersonen (wobei die Versuchspersonen auch Experten auf dem Gebiet sein können), die also menschliches Verhalten simulieren wollen. Dieser Aspekt wird in vielen Untersuchungen wenig klargestellt, vielmehr wird gerne der Anschein erweckt, daß es sich bei den verwendeten Methoden um von menschlichen Einflüssen unabhängige Maße handele.

***Definition 2.4.2.1 Relevanz*** *Die Relevanz eines Dokumentes für eine Anfrage ist eine Relation, $r : D \times Q \rightarrow R$ wobei $D = \{d_1, ..., d_m\}$ die Menge der Dokumente, $Q$ die Menge der Anfragen und $R$ eine Menge von "Wahrheitswerten", i. a. die Menge $\{0, 1\}$ ist. (Im folgenden wird $R = \{0, 1\}$ angenommen, wenn nichts anders gesagt wird.)*

*Die Relation $r$ wird im allgemeinen durch Befragen von Experten zu konkreten Anfragen und Dokumentmengen ermittelt und dann ensprechend festgelegt*

Diese Definition scheint auf den ersten Blick wenig auszusagen. Sie schränkt aber ganz erheblich ein, was als Relevanz bezeichnet werden kann. So hängt nach dieser Definition die Relevanz eines Dokuments für eine Anfrage lediglich von der Anfrage und dem Dokument ab. Sie kann nicht durch die anderen Dokumente, die bereits als relevant beurteilt wurden, beeinflußt werden, und auch nicht durch den Wissensstand der Anfragenden.

### 2.4.3 Precision und Recall

Auf der Definition der Relevanz bauen die beiden am häufigsten verwendeten Evaluierungsmaße auf:

***Definition 2.4.3.1 Precision und Recall*** *Sei $D = \{d_1, ..., d_m\}$ eine Menge von Dokumenten, $q \in Q$ eine Anfrage und $D_q$ die Menge der in $D$ zur Anfrage $q$ gefundenen Dokumente. Sei ferner $r : D \times Q \to \{0, 1\}$ eine Relevanzrelation und $r_q : D \to \{0, 1\}$; $r_q(d) := r(d, q)$ die zur Anfrage $q$ gehörende Relevanzfunktion. Dann heißt*

$$P(q, D) := \frac{\mid D_q \cap r_q^{-1}(\{1\}) \mid}{\mid D_q \mid} \tag{31}$$

***Precision, Präzision** oder **Genauigkeit** der Antwort auf die Anfrage $q$ und*

$$R(q, D) := \frac{\mid D_q \cap r_q^{-1}(\{1\}) \mid}{\mid r_q^{-1}(\{1\}) \mid} \tag{32}$$

***Recall** oder **Vollständigkeit** der Antwort auf die Anfrage $q$.*

Precision gibt also den Anteil der relevanten Dokumente unter den gefundenen Dokumenten an, Recall gibt den Anteil der relevanten Dokumente an, die gefunden wurden. Optimal, nämlich gleich $1$, sind die Werte für Precision und Recall natürlich genau dann, wenn $D_q = r_q^{-1}(\{1\})$ gilt, wenn also genau alle relevanten Dokumente als Antwortmenge zurückgeliefert werden.

Die beiden Maße sind in gewisser Weise gegenläufig. Zur Illustration kann man die beiden Extremfälle betrachten: Wenn $D_q = D$ gilt, wenn also alle Dokumente auf die Anfrage hin zurückgeliefert werden, ist der Recall gleich $1$:

$$R(q, D) := \frac{\mid D \cap r_q^{-1}(\{1\}) \mid}{\mid r_q^{-1}(\{1\}) \mid} = \frac{\mid r_q^{-1}(\{1\}) \mid}{\mid r_q^{-1}(\{1\}) \mid} = 1 \tag{33}$$

(jedenfalls, wenn es überhaupt relevante Dokumente in der Datenbank gibt). Die Precision wird aber sehr schlecht sein. (Jedenfalls, wenn nicht alle Dokumente relevant sind).

Wird umgekehrt nur ein einziges relevantes Dokument $d_r \in D$ gefunden, so ist

$$P(q, D) := \frac{\mid \{d_r\} \cap r_q^{-1}(\{1\}) \mid}{\mid \{d_r\} \mid} = \frac{\mid \{d_r\} \mid}{\mid \{d_r\} \mid} = 1 \tag{34}$$

der Recall wird aber sehr schlecht sein (jedenfalls, wenn es mehrere relevante Dokumente in der Datenbank gibt).

In der Regel werden die Antwortmengen aber zwischen diesen beiden Extremen liegen. Dann ergibt sich i. a. bei einer Verkleinerung der Antwortmenge durch eine spezifischere Anfrage eine bessere Precision, aber ein schlechterer Recall, bei einer Vergrößerung der Antwortmenge durch eine allgemeinere Anfrage ein größerer Recall aber eine kleinere Precision. Ähnliche Situationen können auftreten, wenn Precision- und Recallmaße dazu verwendet werden verschiedene Systeme zu vergleichen. Eindeutige Aussagen darüber, ob ein System besser ist als das andere, können nur gemacht werden, wenn für das eine System sowohl der Precisionwert als auch der Recallwert besser ist als bei dem anderen System. Ist bei einem System z. B. die Precision

besser, dafür aber der Recall schlechter, so eignen sich die System zwar für unterschiedliche Aufgaben, es kann aber nicht allgemein gesagt werden, welches besser ist.

Die Gegenläufigkeit der Maße wird besonders interessant, wenn die Größe der Antwortmenge des Systems durch einen Parameter steuerbar ist, wie dies bei den Vektorraummodellen (siehe Abschnitt 2.3) der Fall ist. Dort wurden die Antwortmengen durch eine Ähnlichkeitsschranke bestimmt, bzw. es wurden als Antwortmengen Rangreihen von Dokumenten geliefert. Mit einer solchen Rangreihe kann man nun den Zusammenhang zwischen Precision und Recall darstellen, und gegebenenfalls die Schwelle so festlegen, daß die gewünschte Art der Antwortmenge erzielt wird.

***Definition 2.4.3.2 Precision-Recall-Diagramm*** *Sei $D_q = (d_{s_1}, ..., d_{s_k})$ eine vollständig geordnete Antwortmenge, $R_q = r_q^{-1}(\{1\})$ die Menge der zu der Anfrage $q$ relevanten Dokumente und $(d_{t_1}, ..., d_{t_l})$ die Schnittmenge $D_q \cap R_q$ in der Anordnung aus $D_q$. Dann bezeichnet die Folge $(R_i(q, D), P_i(q, D))_{i=1,...,l}$ mit*

$$R_i(q, D) := \frac{\mid (d_{t_1}, ..., d_{t_i}) \mid}{\mid r_q^{-1}(\{1\}) \mid} \qquad (35)$$

*und*

$$P_i(q, D) := \frac{\mid (d_{t_1}, ..., d_{t_i}) \mid}{\mid (d_{s_1}, ..., d_{s_j}) \mid} \qquad (36)$$

*mit $d_{s_j} = d_{t_i}$ das zu $q$ gehörige **Precision-Recall-Diagramm***

*Dieses Diagramm kann durch Punkte im Quadrat $[0, 1]^2$ dargestellt werden.*

Abbildung 20 gibt ein Beispiel und die zugehörige graphische Darstellung an.

Falls die Antwortmenge nicht vollständig geordnet ist, also z. B. beim Vektorraummodell mehrere Dokumente die selbe Ähnlichkeit zur Anfrage haben, muß darauf geachtet werden, daß die Werte in der Folge nicht durch willkürliche Vergabe der Rangplätze beeinflußt werden. Wäre im Beipiel der Abbildung 20 jeder Block eine Gruppe von Dokumenten mit gleicher Ähnlichkeit zur Anfrage, so könnten die Dokumente innerhalb der Blöcke zufällig angeordnet werden. Das würde das Precision Recall Diagramm beeinflussen. So könnte das 14. Element der Folge auch $\left(\frac{1}{17}, \frac{14}{31}\right)$ oder $\left(\frac{1}{17}, \frac{14}{40}\right)$ lauten.

Eine Möglichkeit, das Problem zu lösen, ist Precision – Recall Paare nur für die verschiedenen Ähnlichkeitswerte zu berechnen, also für jede Ähnlichkeitsschranke eine eigenständige Auswertung zu machen. Im Beispiel der Abbildung 20 würde sich (für den konstruierten Fall, daß jeder Block eine Ähnlichkeitsstufe darstellt) eine Folge aus 19 Gliedern ergeben, die folgendermaßen anfängt: $\left(\frac{6}{30}, \frac{6}{10}\right), \left(\frac{11}{30}, \frac{11}{20}\right), \left(\frac{13}{30}, \frac{13}{30}\right), \left(\frac{14}{30}, \frac{14}{40}\right), \left(\frac{17}{30}, \frac{17}{50}\right), ...$ Vergleicht man die Precision an gleichen Recallwerten, zeigt sich, daß die erste Art der Berechnung bessere Precisionwerte liefert. Das liegt daran, daß der Precisionwert in diesem Fall nur für die Dokumente berechnet wird, die in der Ordnung vor dem jeweiligen relevanten Dokument liegen, während bei der Blockberechnung eventuell auch nichtrelevante Dokumente einbezogen werden, die zwischen dem relevanten Dokument und der Blockgrenze liegen.

Weitere Verfahren zur Berechnung von Precision Recall Diagrammen im Fall von nicht vollständig geordneten Antwortmengen finden sich in Fuhr (1995).

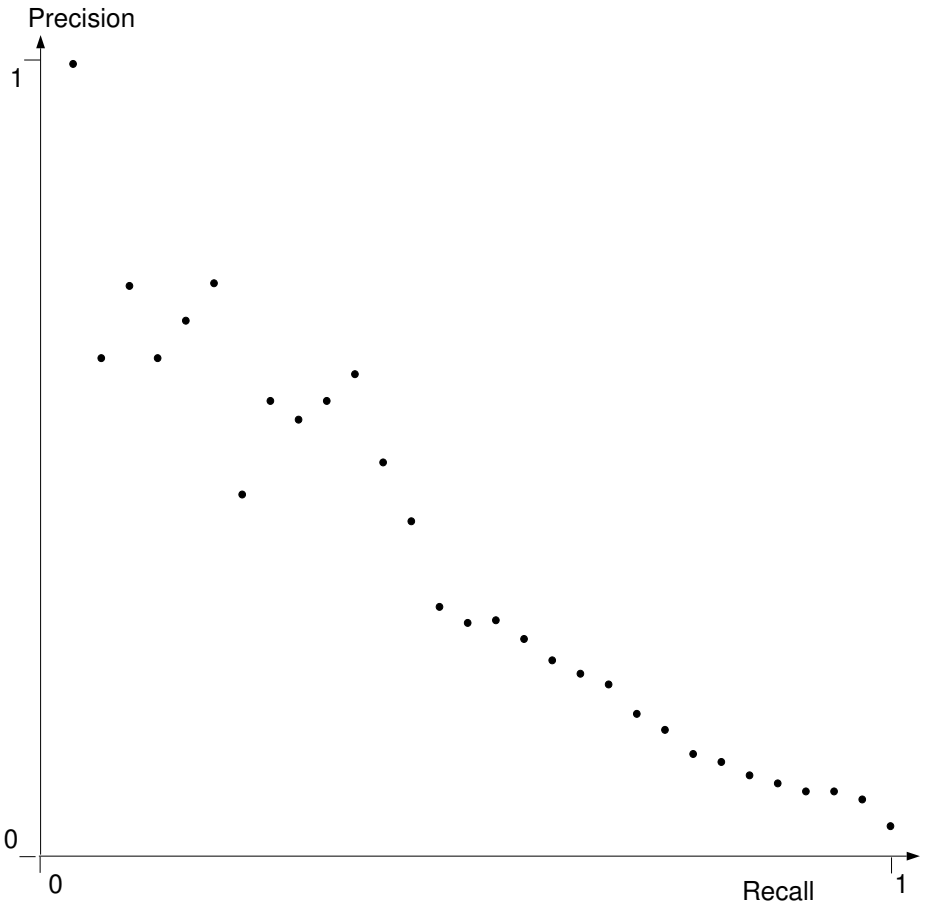
**Abbildung 20: Beispiel eines Precision Recall Diagramms**

Es sei die folgende vollständig geordnete Antwortmenge gegeben, bei der ein `R` für ein relevantes Dokument und ein `U` für ein nicht relevantes Dokument steht.

```
RURRURRRUU URRURRRUUU URUUUURUUU UUUUURUUUU RURUUUUURU
UUURUUUURU UUUUURUUUU UUUUURUUUU UUUUUUUURU UUUUUUUURU
UUUUUUUURU UUUUUUUUUU UUUUURUUUU UUUUUUURUU UUUUUURUUU
UUUUUUURUU UUUUUUUUUU RUUUUUUUUU UUUUUURUUU UU...
```

Die Folge $(R_i(q,D), P_i(q,D))_{i=1,\ldots,30}$ sieht dann so aus:

$\left(\frac{1}{30}, \frac{1}{1}\right), \left(\frac{2}{30}, \frac{2}{3}\right), \left(\frac{3}{30}, \frac{3}{4}\right), \left(\frac{4}{30}, \frac{4}{6}\right), \left(\frac{5}{30}, \frac{5}{7}\right)$
$\left(\frac{6}{30}, \frac{6}{8}\right), \left(\frac{7}{30}, \frac{7}{12}\right), \left(\frac{8}{30}, \frac{8}{13}\right), \left(\frac{9}{30}, \frac{9}{15}\right), \left(\frac{10}{30}, \frac{10}{16}\right)$
$\left(\frac{11}{30}, \frac{11}{17}\right), \left(\frac{12}{30}, \frac{12}{22}\right), \left(\frac{13}{30}, \frac{13}{27}\right), \left(\frac{14}{30}, \frac{14}{36}\right), \left(\frac{15}{30}, \frac{15}{41}\right)$
$\left(\frac{16}{30}, \frac{16}{43}\right), \left(\frac{17}{30}, \frac{17}{49}\right), \left(\frac{18}{30}, \frac{18}{54}\right), \left(\frac{19}{30}, \frac{19}{59}\right), \left(\frac{20}{30}, \frac{20}{66}\right)$
$\left(\frac{21}{30}, \frac{21}{76}\right), \left(\frac{22}{30}, \frac{22}{89}\right), \left(\frac{23}{30}, \frac{23}{99}\right), \left(\frac{24}{30}, \frac{24}{109}\right), \left(\frac{25}{30}, \frac{25}{126}\right)$
$\left(\frac{26}{30}, \frac{26}{138}\right), \left(\frac{27}{30}, \frac{27}{147}\right), \left(\frac{28}{30}, \frac{28}{158}\right), \left(\frac{29}{30}, \frac{29}{171}\right), \left(\frac{30}{30}, \frac{30}{187}\right)$




*Die Punkte werden häufig durch Geraden verbunden. Das geschieht besonders dann, wenn mehrere Diagramme zum Vergleich in eine Graphik eingezeichnet werden, um die zum selben Diagramm gehörenden Punkte leichter zuordnen zu können. Diese Linien haben dann aber keine interpolierende Bedeutung. Zwischen den einzelnen Recallwerten sind keine Precisionwerte definiert!*

## 2.4.4 Mittelwertbildungen

Um aussagekräftige Ergebnisse zu bekommen, müssen die Precision und Recallwerte

über mehrere (repräsentative) Anfragen gemittelt werden. Dazu gibt es zwei verschiedene Möglichkeiten: Die sog. **Makrobewertung** oder der **nutzerorientierte** (*user-oriented*) Ansatz bildet das arithmetische Mittel über die Werte:

$$P_u(D) := \frac{1}{N} \sum_{i=1}^{N} \frac{\mid D_{q_i} \cap r_{q_i}^{-1}(\{1\}) \mid}{\mid D_{q_i} \mid} \tag{37}$$

$$R_u(D) := \frac{1}{N} \sum_{i=1}^{N} \frac{\mid D_{q_i} \cap r_{q_i}^{-1}(\{1\}) \mid}{\mid r_{q_i}^{-1}(\{1\}) \mid} \tag{38}$$

Dabei wird die Größe der beteiligten Antwortmengen nicht berücksichtigt. Ob ein Wert aufgrund von wenigen oder vielen relevanten Dokumenten zustande gekommen ist, spielt keine Rolle. (Nutzerorientiert, weil jede Anfrage gleich in die Bewertung eingeht).

Die **Mikrobewertung** oder der **systemorientierte** (*system oriented*) Ansatz setzt die Summen der Dokumentanzahlen in die Formeln für Recall und Precision ein:

$$P_s(D) := \frac{\sum_{i=1}^{N} \mid D_{q_i} \cap r_{q_i}^{-1}(\{1\}) \mid}{\sum_{i=1}^{N} \mid D_{q_i} \mid} \tag{39}$$

$$R_s(D) := \frac{\sum_{i=1}^{N} \mid D_{q_i} \cap r_{q_i}^{-1}(\{1\}) \mid}{\sum_{i=1}^{N} \mid r_{q_i}^{-1}(\{1\}) \mid} \tag{40}$$

Hier wird der Mittelwert gemäß der Anzahl der beteiligten Dokumente berechnet. Anfragen mit wenigen relevanten Dokumenten spielen eine kleinere Rolle als solche mit vielen relevanten Dokumenten. (Systemorientiert, weil die relevanten Dokumente die Basis der Bewertung bilden.)

Salton und McGill (1983) geben ein nettes Beispiel für die beiden Methoden: Wenn $10$ Seminare stattfinden, davon $5$ mit je $99$ Studenten und $5$ mit je nur einem Studenten, dann ist die Mikrobewertungs– oder die “class level”– Durchschnittsgröße eines Seminars $50$ Studenten, die Makrobewertungs- oder “student-level” Durchschnittsgröße ist aber $98,02$.

### 2.4.5 Testkollektionen

Schon im Abschnitt 2.4.2 war darauf hingewiesen worden, daß eine “objektive” oder zumindest definitorische Bestimmung von Relevanz nicht möglich ist. Um die Relevanz eines Dokumentes für eine Anfrage zu bestimmen, werden i. a. Experten befragt. Das ist eine sehr aufwendige Methode. Sollen Retrieval Systeme bewertet oder verglichen werden, so sollten sie im Idealfall an großen Datenbanken mit einer repräsentativen Menge von Anfragen überprüft werden. Dazu wäre es notwendig, zu jeder Anfrage alle relevanten Dokumente der Datenbank zu kennen. Dabei darf die Menge der relevanten Dokumente natürlich nicht durch ein anderes Retrievalsystem bestimmt werden, da in diesem Fall das zu testende System nicht besser werden könnte als das zur Bestimmung der relevanten Dokumente verwendete System. Zur Bestimmung der zu einer Anfrage relevanten Dokumente (oder zumindest deren Anzahl) kann man verschiedene Vorgehensweisen wählen (siehe auch Fuhr (1995)):

- Man kann eine **repräsentative Stichporbe** aus der Gesamtmenge der Dokumente ziehen und die Dokumente der Stichprobe beurteilen. Problematisch dabei ist,

daß für einigermaßen verläßliche Schätzungen bereits sehr große Stichproben (und damit sehr viele Relevanzurteile) notwendig sind.

- "**Document-Source-Methode**": Man wählt zufällig ein Dokument aus und formuliert eine Anfrage, für die dieses Dokument relevant ist. Anschließend überprüft man, ob das Dokument mit der Anfrage gefunden wird. Durch Wiederholung mit anderen Anfragen kann man die relative Häufigkeit, mit der das Dokument gefunden wird, als Näherung des Recalls verwenden. Diese Methode hat den Nachteil, daß die Anfragen keine echten Benutzeranfragen sind. Sie ist zudem natürlich sehr anfällig für bewußte oder unbewußte Einflußnahmen bei der Konstruktion der Anfragen.
- **Frageerweiterung**: Durch Verallgemeinerung der Anfrage versucht man weitere relevante Dokumente zu finden. Dazu können auch verschiedene Recherchierende beitragen.
- Externe Quellen: Man versucht über Experten, Bibliographien und andere Wissensquellen weitere relevante Dokumente bzw. Zugänge zu solchen zu finden.

Alle diese Methoden sind in einem streng empirischen Sinn kaum haltbar, da sie in der Regel von nur wenigen Personen, die i. a. auch noch mit der Untersuchung befaßt sind, durchgeführt werden. Ausserdem sind sie für die Bewertenden sehr anstrengend, so daß zu befürchten ist, daß die Motivation im Laufe der Bewertung nachläßt oder aber zumindest nicht konstant bleibt. Ein weiteres Problem ist die Veränderung des Wissensstandes der Bewertenden im Laufe der Bewertung, der vermutlich zu einer Veränderung des Bewertungsmaßstabes führen dürfte.

Trotz aller dieser Mängel sind als mehr oder weniger befriedigende Lösung, im Laufe der letzten 30 Jahre eine Reihe von sogenannten **Testkollektionen** (**Document Test Collections**) zusammengestellt worden, die jeweils aus einer Sammlung von Dokumenten, einer Sammlung von Anfragen und den Angaben der für die Anfragen relevanten Dokumente in der Sammlung bestehen. Einige davon sind in Abbildung 21 zusammengestellt.

Neben diesen Testkollektionen gibt es seit 1992 jährlich die sog. TREC Experimente (Text Retrieval Conference), die vom National Institute of Standards and Technology (NIST) der USA durchgeführt werden. Bei diesen "Konferenzen" erhalten die teilnehmenden Institute bis zu einer Million Dokumente (2 GB) als Trainigs- und Testmaterial, die im Laufe eines Jahres bearbeitet werden müssen. Retrieval Ergebnisse zu 250 Anfragen werden eingereicht und vom NIST bewertet.

## Literaturverzeichnis


Dumais, S. T. (1991). Improving the retrieval of information from external sources. *Behavior Research Methods, Instruments, & Computers 23*(2), 229-236.

Ferber, R., Wettler, M., & Rapp, R. (1995). An associative model of word selection in the generation of search queries. *Journal of the American Society for Information Science (JASIS) 46*(9), 685-699.

Fox, E. A., & Lee, W. C. (1991). FASIT-INV: A fast algorithm for building large inverted files. Technical report, VPI&SU Department of Computer Science, Blackburg, VA.

Frakes, W. B., & Baeza-Yates, R., Eds. (1992). *Information Retrieval Data Structures & Algorithms*. Prentice Hall, Englewood Cliffs, New Jersey.

Fuhr, N. (1995). Information retrieval. Skriptum zur Vorlesung im SS 93, Universität Dortmund, http://ls6–www.informatik.uni-dortmund.de/teaching/courses/ir.

**Abbildung 21: Testkollektionen (nach Griffiths Luckhurst & Willett 1986 und Dumais, 1991)**

| Name | Gebiet | Inhalt | 4. Dokumente | 5. Terme / Dok. | 6. Anfragen | 7. Terme / Anfrage | 8. rel./ Anfrage |
|---|---|---|---|---|---|---|---|
| Keen | Bibliothekswesen, Information Science | Titel, Indexterms (manuell) | 800 | 9,8 | 63 | 10,3 | 14,9 |
| Cranfield | Aerodynamik | Indexterms (manuell) | 1400 | 28,7 | 225 | 8,0 | 7,2 |
| Evans | INSPEC | Titel | 2542 | 6,6 | 39 | 27,5 | 23,1 |
| Harding* | INSPEC | Titel, Abstracts | 2472 | 36,3 | 65 | 32,4 | 22,6 |
| LISA** | Library and Information Abstracts Database 1982 | Titel, Abstracts | 6004 | 39,7 | 35 | 16,5 | 10.8 |
| INSPEC | INSPEC | Titel, Abstracts | 12684 | 36,0 | 77 | 17,9 | 33,0 |
| UCKIS | Chemical Abstract Service | Titel | 27361 | 6,7 | 182 | 7,4 | 58,9*** |
| MED | Biomedizin | (Titel?) Abstracts | 1033 | 50 | 30 | 10 | 23 |
| CISI | Bibliothekswesen, Information Science | (Titel?) Abstracts | 1460 | 45 | 35 | 8 | 50 |
| TIME | World news (Time Magazine) | Artikel | 425 | 190 | 83 | 8 | 4 |
| ADI | Bibliothekswesen | Abstract | 82 | 16 | 35 | 5 | 5 |

*Spalte 4 gibt die Anzahl der Dokumente an, Spalte 5 die durchschnittliche Anzahl der Terme in einem Dokument, Spalte 6 die Anzahl der Anfragen, Spalte 7 die durchschnittliche Anzahl der Terme in einer Anfrage und Spalte 8 die durchschnittliche Anzahl der relevanten Dokumente zu einer Anfrage.*

** Harding ist eine Teilmenge der Dokumente aus Evans, angereichert mit Abstracts und zusätzlichen Anfragen. ** Die Relevanzurteile in LISA wurden durch Suche in der gedruckten Ausgabe und durch vereinzelte Datenbankrecherchen erhoben. *** Die Relevanzurteile in UCKIS sind teilweise unvollständig.*

Griffiths, A., Luckhurst, H. C., & Willett, P. (1986). Using interdocument similarity information in document retrieval systems. *Journal of the American Society for Information Science 37*(1), 3–11.

Jones, W. P., & Furnas, G. W. (1987). Pictures of relevance: A geometric analysis of similarity measures. *Journal of the American Society for Information Science 38(6)*, 420-442.

Kuhlen, R. (1977). *Experimentelle Morphologie in der Informationswissenschaft*. München: Verlag Dokumentation.

Lezius, W. (1995). Morphologie und tagging. Technical report, Universität -GH - Paderborn, FB 2, http://www.uni-paderborn.de/extern/fb/2/KoPro/projekte/morpho.html.

Salton, G., & McGill, M. J. (1983). *Introduction to Modern Information Retrieval*. New York: McGraw-Hill.

Xerox. MultiLingual Theory and Technology. WWW-HTML Page: http://www.xerox.fr.grenoble/mltt/Mos/Tools.html.

# Teil 3 Data Mining

In der Einleitung war Knowledge Discovery in Databases als eine Theorie beschrieben worden, die sich unter anderem aus dem Bereich Machine Learning entwickelt hat. Machine Learning wiederum ist inspiriert durch psychologische Lerntheorien. Daher zunächst ein kleiner Exkurs zum Thema Lernen.

## 3.1 Lernen

In der Psychologie bezeichnet man **Lernen** als eine Veränderung von **Verhaltensdispositionen** durch **Erfahrung**. Man spricht von *Verhalten* und nicht von Wissen, weil nur Verhalten beobachtet werden kann. Wissen kann nur durch Verhalten für andere manifest werden. Man spricht von Verhaltens*disposition*, also der Fähigkeit oder Möglichkeit, ein bestimmtes Verhalten zu zeigen, weil das Lernen unabhängig davon ist, ob das Verhalten tatsächlich gezeigt wird. Der Einfluß von Erfahrung grenzt Lernen von Vorgängen wie Reifung oder Entwicklung ab. Die (mehr oder weniger) dauerhafte Veränderung der Verhaltensdisposition beruht auf Veränderungen im **Gedächtnis** bzw. der Gedächtnisinhalte; daher ist Lernenforschung immer eng mit Gedächtnisforschung verknüpft.

Lernen ist also dadurch beschreibbar, daß sich eine Verhaltensdisposition d. h. die Fähigkeit oder Möglichkeit, ein bestimmtes Verhalten zu zeigen, durch einen äußeren Einfluß ändert, ohne daß es sich dabei primär um eine mechanisch–körperliche Veränderung handelt. Das kann zum Beispiel die Fähigkeit sein

- auf eine Frage in einer bestimmten Sprache zu antworten oder auf einen Satz in der Sprache zu reagieren, nachdem man ein Lehrbuch dieser Sprache durchgearbeitet hat
- aus einer Folge von Buchstaben ein Wort oder einen Satz vorzulesen, nachdem man die Grundschule besucht hat
- eine bestimmte Art von Rechenaufgaben zu lösen
- einen Gegenstand mit einem Begriff zu bezeichnen, nachdem man beobachtet hat, daß andere diesen Begriff für den Gegenstand verwenden
- auf einem Fahrrad zu fahren, nachdem man es einige Male erfolglos versucht hat
- einen Schweißausbruch zu bekommen, wenn man eine bestimmte Person sieht
- als Ratte den richtigen Weg durch ein Labyrinth zu finden (operantes Lernen)
- als Hund auf einen Signalton hin Speichel abzusondern (klassische Konditionierung)

Umgangssprachlich bezeichnet man das Ergebnis eines Lernvorgangs als

- **wissen**, wenn es sich um sprachlich-begriffliche Fähigkeiten handelt (“Ich weiß, wie ich zum Luisenplatz komme”)
- **kennen**, wenn es sich um sinnliche Wahrnehmungen handelt (“Ich kenne den Luisenplatz”)
- **können**, wenn es sich um motorische Fähigkeiten handelt (“Ich kann mit dem Fahrrad zum Luisenplatz fahren”)

Diese für Informatikerinnen und Informatiker sicherlich etwas umständliche Art, Lernen zu definieren, zeigt den Unterschied, zwischen einer empirischen Wissenschaft, wie der Psychologie und einer Ingenieurwisssenschaft wie der Informatik. Während die letztere sich die Modelle und Strukturen vorgibt, die sie erforscht, muß die erstere versuchen, die im Alltagswissen und in der Umgangssprache verwendeten Begriffe so zu präzisieren und systematisieren, daß sie beobachtete Phänomene beschreiben und sie einer wissenschaftlichen Erforschung erschließen. Dabei steht zunächst die Beschreibung der Phänomene durch Beobachtung und Experimente im Vordergrund, und die Erklärung im Sinne eines kausalen Modells ist erst der zweite (oder n-te) Schritt.

Die Forschung zur künstlichen Intelligenz (KI) bewegt sich in einem Spannungsfeld zwischen diesen beiden Ansätzen. Häufig werden die Aufgaben und insbesondere die erwarteten Lösungen der Aufgaben eher ad hoc formuliert und nicht empirisch ermittelt. Es werden Modelle entwickelt, die einzelne Phänomene z. B. aus dem Bereich des Spracheverstehens so simulieren, wie sie die Untersuchenden verstehen. Sprache – insbesondere ihr syntaktischer Aspekt – ist ein beliebtes Forschungsgebiet, weil es dort anerkannte Regeln gibt. An diese Regeln halten sich in der Umgangssprache allerdings nur wenige, trotzdem klappt die Verständigung i. a. ziemlich gut (oft sogar besser, als wenn sich die Sprechenden an die Regeln halten). Genau diese Fähigkeit der “natürlichen Intelligenz”, mit fehlerhaften, ungenauen und vagen Objekten umzugehen, ist aber eines der Hauptprobleme der (symbolorientierten) KI. Um dieses Problem anzugehen, kann man die verschiedenen Arten des Lernens betrachten und versuchen, die Prozesse, von denen man vermutet, daß sie die Robustheit der “natürlichen Intelligenz” gegenüber “Fehlern” bewirken, nachzubilden.

Dazu kann man zunächst die Lernprozesse nach einem Input – Output – Schema klassifizieren, also die Frage stellen: Welche Erfahrungen führen zu welchen Verhaltensdispositionen? Man beginnt bei einfachen Lernprozessen und schreitet zu schwierigeren fort. Der Einschätzung der Schwierigkeit liegt die Annahme zugrunde, daß Lernen nicht nur ein Abspeichern von Zuordnungen von Reaktionen zu Reizmustern ist, sondern eine Informationsverarbeitung, die Regeln verwendet. Unter dieser Annahme ist der einfachste Fall das Lernen eines konkreten Verhaltens durch üben, z. B. Lernen eines Labyrinths bei Ratten, Fahrradfahren, Balancieren, Vokabelnlernen etc. Es wird das Verhalten geübt, das auch abgeprüft wird.

Bei den schwierigeren Aufgaben kann man zwischen der Komplexität des Inputs und der des Ouputs unterscheiden. Wenn wir beim Output bei der Überprüfung von Beispielen bleiben, ist die nächst komplexere Aufgabe das verallgemeinernde Lernen. Es werden Beispiele wahrgenommen, die den später zu lösenden Aufgaben zwar ähnlich, aber nicht mit ihnen identisch sind, die also eine **Generalisierung** verlangen. Solche Aufgaben sind z. B. das Lösen von bestimmten Arten von Rechenaufgaben oder das Kategorisieren von Objekten anhand von Beispielen, die untereinander ähnlich sind. Die nächsten Aufgaben dieser Art sind die, bei denen Regeln, Theorien oder Modelle gelernt werden, die dann an Beispielen überprüft werden. (Ob diese Aufgaben schwieriger sind als die Generalisierung, hängt natürlich auch von den Regeln, den Beispielen und dem Grad der Generalisierung ab.)

Werden beim Output allgemeine Regeln über die Beispiele, also Theorien oder Modelle überprüft, ist der einfachste Fall der, bei dem Regeln gelernt wurden. Schwieriger sind die Aufgaben, bei denen Beispiele als Lernmaterial vorgegeben werden und eine Theorie abgeprüft wird. Diese Aufgaben erfordern die Konstruktion eines Modells oder einer Theorie.

Aufgrund dieser theoretischen Überlegungen kann man Rückschlüsse auf die Lern-

**Abbildung 22: Mathematische Logik oder probabilistisches Schließen. Nach Ferber, Wettler & Rapp (1995)**

---

**Vorgaben:**
**Einige Lehrer sind Ökologen**
**Einige Ökologen fahren Fahrrad**

**Schlussfolgerung:**
**Einige Lehrer fahren Fahrrad**

---

*Die Schlußfolgerung aus den beiden Voraussetzungen ist im logischen Sinne falsch. Sie ist aber im Alltagswissen plausibel und durch probabilistisches Schließen erklärbar: Die Wahrscheinlichkeit, daß ein Lehrer Ökologe ist, ist größer als* $0$*, die Wahrscheinlichkeit, daß ein Ökologe Fahrrad fährt ist größer als* $0$*. Die Wahrscheinlichkeit für das gemeinsame Auftreten zweier unabhängiger Ereignisse ist gleich dem Produkt der Einzelwahrscheinlichkeiten und damit in diesem Beispiel größer als* $0$*. Da die Anzahl der Lehrer groß ist, ist es plausibel, daß es fahrradfahrende Lehrer gibt.*

vorgänge versuchen. So könnte man annehmen, daß beim Lernen von Beispielen keine Modellbildung notwendig ist und deswegen aus "resourcenökonomischen" Gründen auch nicht stattfindet. Empirische Untersuchungen deuten aber darauf hin, daß diese Annahme so nicht haltbar ist. So lernen z. B. Ratten schneller, ein Labyrinth zu durchschwimmen, wenn sie vorher gelernt haben, es zu durchlaufen. Auch Versuchspersonen geben auf Befragen i. a. an, daß sie bei Lernexperimenten, in denen Paare gelernt werden mußten, nach Regeln gesucht haben. Vermutlich hat es sich im Laufe der Evolution als nützlich erwiesen, Regeln, Modelle oder Theorien über die Umwelt zu bilden, um deren Verhalten bzw. die Konsequenzen des eigenen Verhaltens in ihr vorherzusagen. Wie diese Verarbeitung der Umwelteinflüsse allerdings aussieht und ob sie mit den Regeln der mathematischen Logik, die die selbe Terminologie verwendet wie die Alltagssprache, übereinstimmt ist dadurch noch nicht geklärt. Tatsächlich akzeptieren z. B. ca. $80\%$ der befragten Versuchspersonen eine logisch falsche aber probabilistisch einsichtige Folgerung der Art, wie sie in Abbildung 22 gezeigt ist (Mayer, 1983).

Die Untersuchung der Prozesse, die an der Lösung dieser Aufgaben beteiligt sind, ist Thema der Kognitionspsychologie, Gedächtnispsychologie und der Cognitive Science.

### 3.1.1 Induktives Lernen

Auch wenn über die Natur der mentalen Modellbildung noch nicht allzuviel bekannt ist, kann man die verschiedenen Arten des Lernens, wie sie oben beschrieben wurden, in Analogie zu den logischen Operationen bezeichnen.

- **deduktives Lernen** bezeichnet das Folgern von Beispielen aus (bekannten) Regeln
- **induktives Lernen** bezeichnet das Ableiten von Regeln aus Beispielen.
- **abduktives Lernen** bezeichnet den Rückschluß auf (hypothetische) Ursachen aufgrund von beobachteten Beispielen bzw. Ergebnissen

Dedukives Lernen ist also (zumindest in der Theorie) allgemein richtig, bzw. so richtig wie die Regeln, aus denen die Richtigkeit einer Antwort abgeleitet wird (wenn in der Ableitung keine "Fehler" gemacht werden).

**Abbildung 23: Beispiel für die verschiedenen Arten des Lernens**

---

**deduktiv:**
**Es regnet.**
**Wenn es regnet, wird die Straße naß.**
**→ Die Straße ist naß. (Darum ziehe ich feste Schuhe an)**

---

**induktiv:**
**Als gestern die Straße naß war, hat es geregnet.**
**Heute ist die Straße naß und es regnet.**
**Letzte Woche war die Straße naß und es hat geregnet.**
**→ Wenn es regnet, wird die Straße naß.**

---

**abduktiv:**
**Die Straße ist naß.**
**Wenn es regnet wird die Straße naß.**
**→ Es regnet. (Darum nehme ich den Regenschirm mit)**

---

*Der deduktive Fall geht von der Regel aus. Wenn die Voraussetzung erfüllt ist, ist auch die Folgerung richtig. Dabei wird die Richtigkeit der Regel nicht in Frage gestellt.*

*Der induktive Fall geht von der Beobachtung aus. Die induzierte Regel ist richtig, solange es keine überdachten oder beheizten Straßen gibt, wenn man lange genug raus schaut, falls es gerade anfängt zu regnen etc...*

*Der abduktive Fall ist richtig, wenn die Regel eineindeutig, also umkehrbar ist. In diesem Fall also z. B., wenn es keine Rohrbrüche, Straßensprengwagen oder putzende bzw. autowaschende Anwohnende gibt. Ansonsten erzeugt die Abduktion aber eine sinnvolle Hypothese.*

Induktives Lernen kann nur in Abhängigkeit von einer gegebenen Beispielmenge “richtig” oder “falsch” sein. Die aus Beispielen gewonnenen Regeln werden im Allgemeinen an Mengen von Beispielen überprüft. Sie können z. B. die Beispiele, aus denen sie entwickelt wurden, vollständig abdecken, aber neue Beispiele nicht mehr richtig klassifizieren. In vielen Fällen können sie auch die vorgegebenen Beispiele nur teilweise erklären, wenn diese z. B. widersprüchlich sind oder zu zahlreich, um vollständig berücksichtigt zu werden. In solchen Fällen sollten möglichst “gute” und “plausible” Regeln konstruiert werden. (Widersprüchlichkeit bezieht sich hier immer auf den Repräsentationsrahmen, also den Teil der “Wirklichkeit”, der in der Beschreibung der Beispiele dargestellt wird.)

Abduktives Lernen schließlich setzt die Regeln voraus und schließt rückwärts von der Folgerung auf eine mögliche Ursache. Hier hängt die Richtigkeit des Ergebnisses davon ab, ob die Regel umkehrbar ist, bzw. ob sie in der betrachteten Beispielmenge umkehrbar ist.

## 3.2 Automatisiertes Lernen

Knowledge Discovery in Databases ist eine Anwendung des Prinzips des induktiven Lernens auf Datenbanken. Im allgemeinen wird es auf Faktendatenbanken angewendet, also auf Sammlungen stark strukturierter Objekte.

### 3.2.1 Faktendatenbanken

Faktendatenbanken lassen sich – wie in Abschnitt 2.1 beschrieben – als eine Menge von Dokumenten $D$ definieren, auf die Attribute

$$A_i : D \rightarrow R_i \ , \ i = 1, ..., n \tag{41}$$

angewendet werden können. Jedes dieser Attribute hat eine vorgegebene Menge $R_i$ von Werten, die es annehmen kann (Wertebereich). Wie bei der in Abbildung 10 gezeigten schematischen Darstellung eines Information Retrieval Systems gibt es eine interne Darstellung der Objekte als Mengen von Attribut-Wert-Paaren, also als Elemente oder **Tupel** aus der Menge

$$R_1 \times R_2 \times ... \times R_n = \prod_{k=1}^{n} R_k \tag{42}$$

In der Terminologie der relationalen Datenbanken ist die Menge der Dokumente als eine Relation über die Attribute definiert, Teilmengen der Dokumente sind Tabellen über den Attributen.
Definiert man die Transformation

$$T : D \rightarrow \prod_{k=1}^{n} R_k, \ T(d) = (A_1(d), ..., A_n(d)) \tag{43}$$

als die Abbildung, die die Objekte durch die Attribute $A_i, \ i = 1, ..., n$ in die Menge der Tupel abbildet, so ist das gerade die Transformation $T_3$ in Abbildung 10. Objekte, die unter dieser Transformation auf das gleiche Tupel abgebildet werden, sind in der Faktendatenbank nicht zu unterscheiden. Wenn sie aufgrund von Anforderungen an das System unterschieden werden müssen, ist die Auswahl der Attribute ungeeignet für die Anwendung.

### 3.2.2 Kategorisieren

Im Abschnitt 1.5 war Knowledge Discovery in Databases folgendermaßen definiert worden:

> *KDD beschreibt automatisierte Verfahren, mit denen Regelmäßigkeiten in Mengen von Datensätzen gefunden und in eine für Nutzende verständliche Form gebracht werden.*

Diese Definition wird im allgemeinen von Data Mining Systemen nicht ausgeschöpft. Es werden vor allem Verfahren verwendet, die die einzelnen Objekte bzw. Tupel und ihre Eigenschaften betrachten und bewerten. Abhängigkeiten zwischen den Objekten bzw. Tupeln werden selten gesucht. Das zeigt sich in der am weitesten verbreiteten Aufgabenstellung, der Kategorisierung: Eine Teilmenge von Tupeln wird nach einem Kriterium in Kategorien eingeteilt. Ziel ist es, Regeln zu finden, die für die einzelnen Tupel aufgrund ihrer Eigenschaften vorhersagen, in welche Kategorie sie gehören.
Formal läßt sich das so definieren:

***Definition 3.2.2.1 Kategorisierung:*** *Sei $D$ eine Menge von Dokumenten und $K_i \subset D,\ K_i \cap K_j = \emptyset,\ \forall\ i,j \in \{1,...,r\},\ i \neq j$ mit*

$$D = \bigcup_{k=1}^{r} K_i \tag{44}$$

*eine Zerlegung der Menge $D$ in disjunkte Teilmengen. Dann heißt $\{K_1,...,K_r\}$ eine Kategorisierung der Menge $D$.*

*Ist $T$ die Menge der Tupel, die bezüglich der Attribute*

$$A_i : D \to R_i\ \ ,\ i = 1,...,n \tag{45}$$

*zu den Dokumenten gehören, dann kann man $T$ nach dem Attribut $A_i$ kategorisieren, indem man die Urbildmengen der Elemente aus $R_i$ als Kategorien betrachtet:*

$$\left\{A_i^{-1}(\{x\}) \mid x \in R_i\right\} \tag{46}$$

*Ist umgekehrt $T$ durch $\{K_1,...,K_r\}$ kategorisiert, kann man ein Attribut $A_{n+1} : D \to \{1,...,r\},\ A_{n+1}(d) = i \Leftrightarrow d \in K_i$ definieren.*

*Man kann also ohne Beschränkung der Allgemeinheit annehmen, daß eine kategorisierte Menge von Tupeln nach einem Attribut kategorisiert ist.*

Um Regelmäßigkeiten zu finden, verwendet man ein **Trainingsset**, das ist eine endliche Menge von Objekten bzw. Tupeln, auf denen der KDD Algorithmus arbeitet. Um die Ergebnisse zu überprüfen, verwendet man eine andere Menge von Beispielen, das sogenannte **Testset**. Das besteht i. a. aus endlich vielen Tupeln, die nicht zum Lernen verwendet wurden und so für das Regelwerk neu sind. Zeigen die gefundenen Regeln auch auf diesem Testset gute Ergebnisse, so kann man erstmal annehmen, daß den gefundenen Regeln eine gewisse allgemeine Gültigkeit innerhalb des Bereichs, aus dem die Beispiele stammen, zukommt.

Die Definition des Trainings- und Teststes als Mengen ist insofern etwas problematisch, als in Mengen gleiche Elemente nicht mehrfach auftreten können. Bei den Objekten ist das unproblematisch, da auch identische Objekte natürlich verschieden sind. Bei den Tupeln, die durch Anwendung der Attribute auf die Objekte entstehen, sollten identische Objekte auf gleiche Tupel abgebildet werden, und damit nur einmal in einer Menge vorkommen. Formal läßt sich das durch einen zusätzlichen eindeutigen Bezeichner für die Tupel mit gleichen Attributwerten lösen. Es soll hier aber nur daraufhingewiesen werden, daß Tupel mit identischen Attributwerten mehrfach in den Test und Trainingssets vorkommen können.

### 3.2.3 Überwachtes Lernen

Sind bei den Beispielen des Trainingssets die Attributwerte und damit auch die gesuchte Kategorisierung bekannt und werden von dem KDD Algorithmus genutzt, spricht man von **supervised learning**, **überwachtem Lernen** oder **Lernen aus Beispielen**. Diese Form des Lernens hat den Vorteil, daß die bekannten Werte des Trainingssets bei der Konstruktion der Regeln verwendet werden können und dadurch die Algorithmen wesentlich effektiver werden. Sie hat den Nachteil, daß ein (i. a. großes) Trainingsset und ein Testset bereitgestellt werden müssen, bei denen die Bewertung bzw. Kategorisierung vorhanden ist. Solche Datensets sind häufig nur unter großen Kosten zu beschaffen.

Eine abgeschwächte Form des überwachten Lernens ist das bewertete Lernen. Dabei werden nicht die exakten Kategorisierungen, also die gesuchten Attributwerte, zur Verfügung gestellt, sondern es werden Kategorisierungsleistungen bewertet. Das ist oftmals erheblich weniger aufwendig, als die genauen Attributwerte angeben zu müssen.

### 3.2.4 Nichtüberwachtes Lernen

Beim nichtüberwachten Lernen sind im Gegensatz zum überwachten Lernen die Kategorisierungen nicht vorgegeben. Der Algorithmus muß eine Einteilung in Kategorien selbst finden. Hat man Ähnlichkeitsmaße auf der Menge der Tupel definiert, so handelt es sich bei diesen Verfahren um **Clusterverfahren** in einem sehr allgemeinen Sinn. Diese Verfahren sind vor allem dann sinnvoll, wenn auf der Menge der Tupel Ähnlichkeitsstrukturen vorhanden sind oder vermutet werden, die inhaltlich interpretiert werden können.

Nichtüberwachte Lernverfahren haben den Vorteil, daß keine kategorisierten oder bewerteten Trainingssets vorliegen müssen. Allerdings braucht man auch für solche Verfahren eine Bewertung der Ergebnisse der Kategorisierung. Diese kann aber noch stärker als beim bewerteten Lernen von den Einzelergebnissen abstrahieren. Sie kann z. B. durch die nachträgliche Begutachtung der gefundenen Kategorisierung durch Experten erreicht werden.

Bei Data Mining Algorithmen wird überwachtes Lernen eingesetzt. Wir können aufgrund der vorangehenden Definitionen die Aufgabe des Data Mining genauer formulieren:

***Definition 3.2.4.1 Data Mining:*** *Sei $T$ ein Trainingsset von Tupeln über den Attributen $A_i : D \rightarrow R_i \ , \ i = 1, ..., n$. Die Menge der Attribute sei in zwei Teilmengen*

$$A_i : D \rightarrow R_i \ , \ i \in Z \subset \{1, ..., n\} \qquad (47)$$

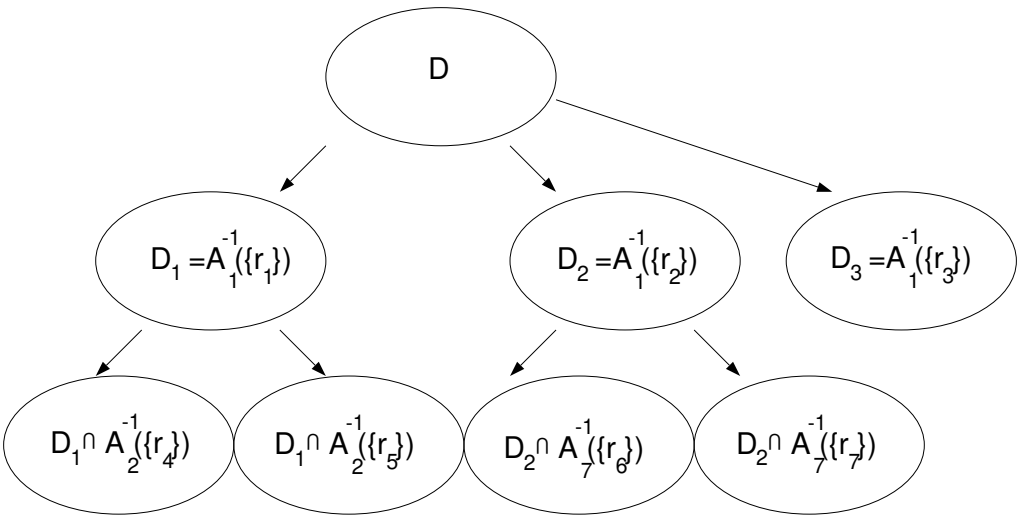
*die **vorherzusagenden Attribute*** (predicted attributes) *und*

$$A_i : D \rightarrow R_i \ , \ i \in \{1, ..., n\} \setminus Z \qquad (48)$$

*die **vorhersagenden Attribute*** (predicting attributes) *unterteilt. Ein **Data Mining Algorithmus** ist ein Algorithmus, der aufgrund der Elemente des Trainingssets Regeln oder Regelwerke – wie Entscheidungsbäume – generiert, mit denen für jedes $t \in T$ aufgrund der Werte der vorhersagenden Attribute $A_i, \ i \in \{1, ..., n\} \setminus Z$ Werte für die vorherzusagenden Attribute $A_i \ , \ i \in Z$ bestimmt werden können.*

Als Beispiel soll zunächst eher informal und prototypisch der ID3 Algorithmus von Ross Quinlan beschrieben werden, wie er in Holsheimer und Siebes (1994) dargestellt ist. Dieser Algorithmus generiert einen Entscheidungsbaum, also eine Struktur, bei der zur Klassifizierung eines Tupels an der Wurzel begonnen wird und solange Abfragen, die jeweils einen Ast bestimmen, durchgeführt werden, bis ein Blatt des Baumes erreicht ist, das die Klassifizierung des Tupels angibt. Ein Beipiel ist in Abbildung 24 gezeigt.

**Abbildung 24: Entscheidungsbaum nach dem ID3 Algorithmus**




*In den Knoten sind die Tupelmengen angegeben, die in dem jeweiligen Knoten aufgeteilt werden. D ist die Gesamtmenge der Trainingstupel. Das Attribut $A_1$ hat die Wertemenge $\{r_1, r_2, r_3\}$ das Attribut $A_2$ hat die Wertemenge $\{r_4, r_5\}$ und das Attribut $A_7$ hat die Wertemenge $\{r_6, r_7\}$*

## 3.2.5 ID3 (Quinlan)

Bei der Konstruktion des Entscheidungsbaumes wird in jedem Knoten ein einzelnes Attribut dazu verwendet, die Menge der Beispiele aufzuteilen, bis die Beispiele einer Teilmenge alle aus einer Klasse sind.

### 3.2.5.1 Algorithmus im Überblick

1. Ordne alle Beispiele einem (Wurzel) Knoten zu.
2. Wähle ein Attribut für den Knoten, füge für jeden möglichen Wert des Attributes einen Kindknoten an und verteile die Beispiele des Knotens gemäß den Werten des Attributes auf die Kindknoten.
3. Falls in einem Kindknoten nur Beispiele einer Kategorie vorliegen, beende diesen Zweig, andernfalls gehe mit den Kindknoten nach Schritt 2

### 3.2.5.2 Auswahlkriterium

Das ID3 Verfahren versucht jeweils dasjenige Attribut zur Selektion zu verwenden, das die Informationsmenge minimiert, die für die restlichen Entscheidungen benötigt wird. Dazu wird das Attribut $A$ gewählt, für das

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I(p_i, n_i) \tag{49}$$

minimal wird. Dabei ist

$$I(p_i, n_i) = -\frac{p_i}{p_i + n_i} ln\left(\frac{p_i}{p_i + n_i}\right) - \frac{n_i}{p_i + n_i} ln\left(\frac{n_i}{p_i + n_i}\right) \tag{50}$$

wobei $p_i$ die Anzahl der positiven Beispiele in der Menge der Beispiele mit Attributwert $v_i$ bezeichnet und $n_i$ die negativen.

#### 3.2.5.3 Formale Beschreibung des ID3 Algorithmus

Nach dieser eher informalen Beschreibung soll nun eine formalere und ausführlichere Beschreibung gegeben werden, zunächst für die Konstruktion des Entscheidungsbaumes:

**3.2.5.3.a Algorithmus** Sei $A_1, ..., A_n$ eine Menge vorhersagender Attribute auf einer Menge $D$ von Beispielen und $A_0 : D \rightarrow R_0 = \{z_1, ..., z_m\}$ ein vorherzusagendes Attribut. Im folgenden rekursiven Algorithmus bezeiche $N = (A, K, z)$ den aktuellen Knoten, wobei $A \in \{A_1, ..., A_n\}$ das aktuelle Attribut, $K \subset D$ die Menge der Beispiele, die dem Knoten zugeordnet sind, und $z \in R_0$ eine Kategorie bezeichnen.

1. Ordne alle Beispiele einem (Wurzel-) Knoten zu:

$$N = (A, K, z) = N_{0,0} = (A, K_{0,0}, z),\ K_{0,0} = D \tag{51}$$

2. Falls alle Tupel aus $K$ der selben Kategorie angehören, falls also gilt $A_0(d_i) = z_k\ \forall\ d_i \in K$ und ein $z_k \in R_0$, setze $A := A_0$ und $z = z_k$.
   Andernfalls wähle ein Attribut $A_i : D \rightarrow R_i,\ \ i \in \{1, ..., n\}$, das in $K$ mindestens $2$ verschiedene Werte annimmt ($\exists d_1, d_2 \in K$ mit $A_i(d_1) \neq A_i(d_2)$), als aktuelles Attribut für den Knoten $N$ und füge für jeden Wert $r_j \in R_i$ mit $A_i^{-1}(\{r_j\}) \cap K \neq \emptyset$ des Attributes einen Kindknoten $N_{i,r_j} = \left(A, K_{i,r_j}\right)$ mit unbestimmtem Attribut und

$$K_{i,r_j} = A_i^{-1}(\{r_j\}) \cap K \tag{52}$$

   (also mit den Beispielen aus $K$, bei denen das Attribut $A_i$ den Wert $r_j$ annimmt) an.
3. Falls ein unbearbeiteter Kindknoten $N_{i,r_j}$ exisitiert, setze $N := N_{i,r_j}$. Andernfalls prüfe, ob es einen unbearbeiteten Geschwisterknoten $N_{i,r_k}$ gibt und setze $N := N_{i,r_j}$. Ist das auch nicht der Fall, setzte $N$ auf den Elternknoten.
4. Falls $N \neq N_{0,0}$ gehe nach Schritt 2, andernfalls beende den Alogrithmus.

**3.2.5.3.b Auswahlkriterium** In Schritt 2 muß ein Attribut $A_i : D \rightarrow R_i$ ausgewählt werden, das zur Selektion verwendet werden soll. Wir nehmen zur Vereinfachung an, daß $R_0 = \{0, 1\}$ gilt, also nur zwei Kategorien existieren, positive Beispiele mit $A_0(d) = 1$ und negative Beispiele mit $A_0(d) = 0$. Für alle Attribute, die in Frage kommen, wird der Wert

$$E(A_i, K_{j,r_m}) = \sum_{k \in R_i} \frac{p_k + n_k}{p + n} I(p_k, n_k) \tag{53}$$

mit

$$I(p_k, n_k) = -\frac{p_k}{p_k + n_k} ln\left(\frac{p_k}{p_k + n_k}\right) - \frac{n_k}{p_k + n_k} ln\left(\frac{n_k}{p_k + n_k}\right) \tag{54}$$

berechnet. Dabei ist $p_k :=\mid A_i^{-1}(\{r_k\}) \cap K_{j,r_m} \cap A_0^{-1}(\{1\}) \mid$ die Anzahl positiver Beispiele mit Attributwert $r_k$ und $n_k :=\mid A_i^{-1}(\{r_k\}) \cap K_{j,r_m} \cap A_0^{-1}(\{0\}) \mid$ die Anzahl negativer

Beispiele mit Attributwert $r_k$, $p :=\mid K_{j,r_m} \cap A_0^{-1}(\{1\}) \mid$ bezeichnet die Gesamtzahl positiver und $n :=\mid K_{j,r_m} \cap A_0^{-1}(\{0\}) \mid$ die Gesamtzahl negativer Beispiele in dem entsprechenden Knoten. Es wird das Attribut zur Selektion gewählt, bei dem der Wert $E(A_i, K_{j,r_m})$ maximal ist.

Wenn mehr Kategorien bearbeitet werden sollen, muß in Formel (54) entsprechend über mehr Werte summiert werden.

Betrachtet man die Beispiele eines Knotens als Informationsquelle über die Zugehörigkeit zu den Zielkategorien, dann gibt die Formel (54) den Informationsgehalt der Aufteilung nach Attribut $A_i$ und Attributwert $r_j$ an. Formel (53) gibt den mittleren Informationsgehalt über alle Werte des Attributes $A_i$ an. Die Idee dieser Wahl ist es, den Informationsgehalt der Entscheidungen zu maximieren.

Um ein Tupel $t = (t_1, ..., t_n)$ mit dem Entscheidungsbaum zu kategorisieren, werden die Beispielmengen $K_{i,r_j}$ nicht mehr benötigt. Bezeichne $N$ wieder den aktuellen Knoten des Baumes und $A$ das diesem Knoten zugeordnete Attribut.

1. Setze $N = N_{0,0}$
2. Falls $A = A_0$ gilt, ordne dem Tupel $t = (t_1, ..., t_n)$ den aktuellen Kategoriewert $z$ zu und beende den Algorithmus. Falls $A = A_i$ setze $N = N_{i,t_i}$ also als den Knoten, der durch den Wert von $t_i$ bezeichnet ist.
3. Gehe zu Schritt 2.

Der ID3 Algorithmus ist der “klassische” Ansatz für Entscheidungsbäume. Er und weitere entschiedungsbaumbasierte Verfahren werden in einer ganzen Reihe von Softwareprodukten angeboten. In der Softwareliste, die in der “Knowledge Discovery Mine” von G. Piatetsky-Shapiro (html) im WWW geführt wird, finden sich folgende kommerziellen und public domain Produkte:

- AC2
- C4.5
- Clementine
- IND
- Information Harvester
- KATE-tools
- Knowledge Seeker
- Linear Machine Decision Trees (LMDT)
- OC1
- Recon ™ Data Mining System
- SE-Learn

Die aktuelle Version finden sich am besten unter:
http://info.gte.com/~kdd/siftware.html.

Ein weiteres System “Skycat” wurde im Video vorgestellt. Siehe dazu auch die Presseerklärung auf der Vorlesungshomepage
http://www.darmstadt.gmd.de/~ferber/vorlesung/index.html

Im Folgenden werden wir den ID3 Algorithmus näher betrachten und an ihm Probleme und Fragestellungen entwickeln, die allgemeiner für KDD Algorithmen gelten. Zunächst setzten wir vorraus, daß die Beispiele des Trainingssets **konsistent** sind.

***Definition 3.2.5.4 Konsistenz*** *Sei $D$ eine Menge von Dokumenten, $A_0, ..., A_n$ eine Menge von Attributen auf den Dokumenten, wobei $A_0, ..., A_{m-1}$ vorherzusagende Attribute sind und $A_m, ..., A_n$ die vorhersagenden Attribute sind. Sei $T$ die Menge der Tupel, die mit den Attributen aus den Dokumenten gewonnen werden. $D$ bzw. $T$ heißt* ***konsistente Beispielsammlung****, wenn für alle $s = (s_0, ..., s_n),\ t = (t_0, ..., t_n) \in T$ gilt:*

$$s_i = t_i\ \forall i \in \{m, ..., n\} \Rightarrow s_j = t_j\ \forall j \in \{0, ..., m-1\} \qquad (55)$$

Gibt es nur ein vorherzusagendes Attribut, sagt die Definition gerade, daß dieses durch die vorhersagenden Attribute eindeutig bestimmt sein muß, daß also nicht zwei Tupel, die auf den vorhersagenden Attributen gleiche Werte haben, in verschiedenen Kategorien liegen dürfen. Aus dieser Definition folgt unmittelbar die Aussage:

***3.2.5.5 Aussage:*** *Zu jeder endlichen konsistenten Beispielsammlung gibt es einen Kategorisierungsalgorithmus, der die Tupel der Beispielsammlung richtig kategorisiert.*

Im einfachsten Fall ist dieser Algorithmus eine Aufzählung der Beispiele.

Die Annahme von Konsistenz klingt zunächst nicht sehr einschränkend. Die folgenden Punkte zeigen aber, daß bei vielen praktischen Anwendungen von KDD-Verfahren nicht von einer konsistenten Beispielmenge ausgegangen werden kann. Das ist z. B. der Fall, wenn

- die Attribute nicht für die KDD Aufgabe entwickelt wurden, sondern aus vorhandenen Datenbankeinträgen abgeleitet werden
- die Anzahl der Dokumente sehr groß wird
- die Kategorisierung intellektuell und von mehreren Personen vorgenommen wird
- die Kategorisierung sich über einen längeren Zeitraum erstreckt
- die Kategorisierung an den Objekten und nicht an den Tupeln vorgenommen wird

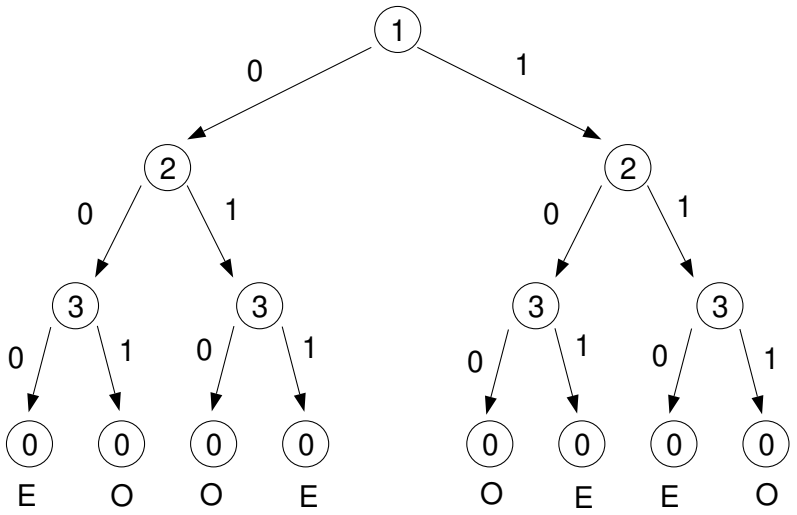
## 3.2.6 Endlichkeit des ID3 auf konsistenten Trainingssets

Nehmen wir ein (endliches) konsistentes Trainingsset an, so wird in jedem Knoten des Baumes ein Attribut getestet, und die Tupel werden gemäß ihres Wertes in diesem Attribut aufgeteilt. Jedes Attribut kann in einem Ast des Baumes höchstens einmal auftreten, da sonst die Forderung, daß es zwei verschiedene Werte innerhalb der Beispielmenge des Knotens annehmen muß, nicht mehr erfüllt ist. Spätestens, wenn alle Attribute abgearbeitet sind, stehen in einem Endknoten des Baumes nur noch Tupel, die auf allen vorhersagenden Attributen identische Werte annehmen, also zur gleichen Kategorie gehören. Die Höhe des Baumes ist also durch die Anzahl der vorhersagenden Attribute beschränkt. Die tatsächliche Höhe hängt natürlich auch von der Anzahl der Kategorien und ihrer Verteilung auf die Tupel ab. Die Extremfälle sind die, in denen die Anzahl der (nicht leeren) Kategorien gleich der Anzahl der möglichen Tupel ist, bzw. die, in denen nur eine Kategorie existiert. Im ersten Fall muß der Baum maximale Höhe haben, und alle Pfade im Baum haben die volle Länge. Im zweiten Fall ist keine Kategorisierung notwendig, der Baum hat die Höhe $1$.

Daß aber neben der Anzahl der Kategorien auch deren Definition entscheidenden Einfluß auf die Größe des Baumes hat, zeigt Abbildung 25. Hier bilden die Tupel mit

gerader Quersumme die Kategorie E und die mit ungerader Quersumme die Kategorie O. Der Baum hat maximale Größe, obwohl nur zwei Kategorien vorliegen. Das liegt daran, daß die Quersumme nur bestimmt werden kann, wenn alle Attribute tatsächlich einbezogen werden. Daraus ergibt sich auch, daß es mit den vorgegebenen Attributen keinen kleineren Entscheidungsbaum geben kann.

**Abbildung 25: Maximaler Entscheidungsbaum mit 2 Kategorien**




*Beispiele sind die Elemente aus $\{0,1\}^3$. Die Kategorie E enthält die Tupel mit gerader Quersumme, die Kategorie O die mit ungerader. In den Knoten steht die Nummer des Attributs, das ausgewertet wird. Aus der Kommutativität der Addition ergibt sich, daß die Reihenfolge allerdings beliebig gewählt werden kann, ohne daß sich die Größe des Baumes ändert.*

Würde man z. B. als Kategorie die Tupel mit einer Quersumme im Bereich $[0,2]$ bzw. $3$ wählen, ergäbe sich ein wesentlich kleinerer Baum (siehe Abbildung 26)

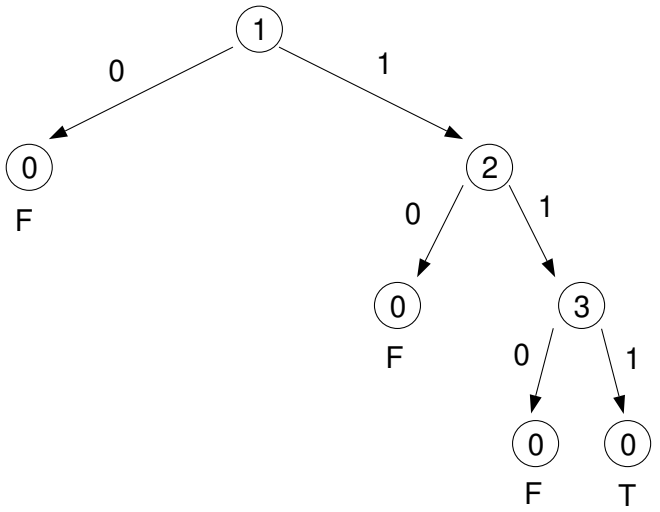
Das Beispiel zeigt auch, daß die Wahl der Attribute (wenn denn eine Wahl besteht) entscheidenden Einfluß auf die Aufwendigkeit der Lösung eines Problems hat.

### 3.2.7 Wertebereiche

In den Beispielen, die wir bisher betrachtet haben, waren die Attribute maximal dreiwertig. (Abbildung 7). Dabei wurden die Wertebereiche $R_i$ stets nur als Mengen betrachtet. Weitere Strukturen wie z. B. Ähnlichkeiten zwischen Werten wurden nicht berücksichtigt. Nun wird aber für jeden Wert, den ein Attribut in der Beispielmenge annimmt, ein neuer Kindknoten angelegt. Das kann zum einen dazu führen, daß es bei großen Wertebereichen sehr viele Knoten gibt und damit der Baum sehr groß wird, zum anderen kann es bedeuten, daß z. B. bei reellen Meßwerten Werte, die sich nur minimal unterscheiden, also z. B. durch zufällige Schwankungen an einem Meßgerät, genauso als verschieden angesehen werden wie Werte mit maximaler Differenz.

Um diese Fälle zu analysieren, sollen zunächst verschiedene Grade der inneren Ordnung von Mengen, sog. **Skalenniveaus**, eingeführt werden. Sie geben an, wie stark eine Menge in sich strukturiert ist.

**Abbildung 26: Entscheidungsbaum mit 2 Kategorien**




*Beispiele sind die Elemente aus $\{0,1\}^3$. Die Kategorie T enthält die Tupel mit Quersumme 3, die Kategorie F alle anderen Tupel. In den Knoten steht die Nummer des Attributs, das ausgewertet wird. Aus der Kommutativität der Addition ergibt sich wieder, daß die Reihenfolge beliebig gewählt werden kann, ohne daß sich die Größe des Baumes ändert.*

#### 3.2.7.1 Nominalskala

Den niedrigsten Organisationsgrad besitzen sog. Nominalskalen. Hier handelt es sich lediglich um Mengen, bei denen jedes Element benannt werden kann, aber keine weiteren Beziehungen zwischen den Elementen vorliegen. Solche Mengen können z. B. Vornamen sein (allerdings lassen die meistens schon auf das Geschlecht schließen) oder Kennziffern, die vergeben werden.

#### 3.2.7.2 Ordinalskala

Bei einer Ordinalskala sind die Elemente vollständig geordnet, d. h. für je zwei verschiedene Elemente ist eindeutig festgelegt, welches von beiden größer ist. Es können aber nicht unbedingt Abstände angegeben werden. Beispiele: hoch tief; ruhend, langsam, schnell, sehr schnell.

#### 3.2.7.3 Intervallskala

Bei Daten, die Intervallskalenniveau haben, kommt zu der vollständigen Ordnung noch die Möglichkeit hinzu, Werte zu addieren.

#### 3.2.7.4 Rationale Skala

Ist schließlich auf einer Intervallskala auch noch die Multiplikation definiert, so spricht man von einer rationalen Skala, mit deren Werten wie gewohnt gerechnet werden darf.

Die Definition des ID3 Algorithmus, wie sie bisher vorgestellt wurde, nutzt also nur Nominalskaleneigenschaften. Weiteres Wissen über die Wertebereiche der Attribute bleibt ungenutzt. Dieses Wissen kann allerdings in die Definition der Attribute gesteckt werden, wie es in dem Beispiel von Carter and Catlett (1987) (siehe Abbildung 7) getan wurde. Dort waren Schwellwerte angegeben worden, und das Attribut war eine binäre Schwellwertfunktion der Form

$$A(x) = \begin{cases} 1 & falls\ x > \theta \\ 0 & sonst \end{cases} \tag{56}$$

In dieser Weise wird das Attribut vereinfacht. Die Schwellwertfunktion hat als Input die reelle Zahl $x$, der Output aber hat nur noch Ordinalskalenniveau.

Eine Verfeinerung dieser Methode, die Anzahl der Kindknoten klein zu halten, ist die Angabe von mehr als zwei Intervallen. Aber auch in diesem Fall hat das Ergebnis der Transformation ein geringeres Skalenniveau. Es geht Information verloren. Dieser Verlust von Information kann als Generalisierung aber auch ein erwünschter Effekt sein.

### 3.2.8 Inkonsistente Trainingssets

Ist das Trainingsset des ID3 Algorithmus inkonsistent, so bricht das Verfahren zusammen, genauer gesagt: es bricht nicht ab. In diesem Fall existieren zwei Tupel $s = (s_0, ..., s_n),\ t = (t_0, ..., t_n) \in T$, die auf den vorhersagenden Attributen $A_0, ..., A_{m-1}$ die selben Werte haben, auf den vorherzusagenden Attributen $A_m, ..., A_n$ aber verschiedene Werte, für die also gilt:

$$s_i = t_i\ \forall i \in \{m, ..., n\}, \exists j \in \{0, ..., m-1\}\ s.d.\ s_j \neq t_j \tag{57}$$

Diese beiden Tupel werden stets dem selben Knoten zugeordnet. Dadurch wird die Beispielmenge dieses Knotens niemals nur Elemente einer Kategorie enthalten und somit der Algorithmus nicht enden.

Um das zu verhindern, kann man das Trainingsset vor der Anwendung des Algorithmus auf Konsistenz überprüfen und gegebenenfalls inkonsistente Beispiele aussondern. Dabei wird allerdings in der Regel die Repräsentativität des Trainingssets leiden. Andere Möglichkeiten sind eine Neudefinition der entsprechenden Attribute oder die Zusammenfassung von Kategorien. In jedem Fall muß eine Lösung gefunden werden, die die Ergebnisse möglichst wenig beeinträchtigt. Dabei werden äußere Anforderungen an das System eine entscheidende Rolle spielen. Diese Verfahren sind aber nicht Bestandteil des eigentlichen Algorithmus.

In vielen Fällen ist es unzweckmäßig, das Trainingsset in eine deterministische Form zu zwingen, weil sie dem Bereich, aus dem die Beispiele stammen, nicht angemessen ist. In diesen Fällen kann eine statistische Aussage, die z. B. Wahrscheinlichkeiten für die Zugehörigkeit von Beispielen zu verschiedenen Kategorien angibt, viel nützlicher sein.

### 3.2.9 Unvollständige Beispiele

Neben inkonsistenten Beispielen können bei KDD Verfahren auch unvollständige Beispiele auftreten, wenn z.B. einzelne Angaben fehlen, oder neue Objekte auftreten, auf die die vorhandenen Attribute nicht angewandt werden können. Zur Behandlung dieser Fälle gibt es einige Standardverfahren zur **Vorbehandlung** (*preprocessing*) des Trainingssets, die je nach den Anforderungen an das System verwendet werden können. Zunächst kann man zu jedem Wertebereich einen zusätzlichen Wert $undefined$ hinzufügen. Das wird in der Regel das Skalenniveau auf das einer Nominalskala absenken. Für Verfahren, die ein höheres Skalenniveau der Wertebereiche ausnützen, ist dieses Verfahren daher problematisch. Andere Möglichkeiten bestehen darin, bei Nominalskalen den Wert, der von dem entsprechenden Attribut am häufigsten im Trainingsset angenommen wird, einzusetzen. Bei Ordinalskalen kann der Median, bei Intervallskalen ein Mittelwert (arithmetisches Mittel, gewichtetes Mittel) eingesetzt werden. Auch hier gilt aber, daß diese Verfahren nicht Bestandteil des eigentlichen KDD-Algorithmus' sind, sondern Methoden, die das Trainingsset mehr oder weniger künstlich in eine Form bringen, die von dem KDD-Algorithmus bearbeitet werden kann. Besser wäre es

natürlich einen Algorithmus zu entwickeln, der die Unvollständigkeit berücksichtigt und entsprechende Regeln generiert.

Eine aufwendigere Methode ist die, die fehlenden Attributwerte mit Hilfe des selben Verfahrens vorherzusagen, das für die vorherzusagenden Attribute verwendet wird: Man nutzt die Beispiele, bei denen der Attributwert vorhanden ist, als Trainingsset, mit dessen Hilfe ein Algorithmus entwickelt wird, mit dem die fehlenden Werte bestimmt werden.

### 3.2.10 Größe des Trainingssets

Datenbanken enthalten häufig sehr viele Einträge. Ähnlich wie bei der in Abschnitt 2.4.5 beschriebenen Auswahl von Testkollektionen stellt sich auch bei Trainingssets das Problem, wie eine handhabbare, aber doch ausreichend große Menge an Beispielen zusammengestellt werden kann. Anders als bei den Testkollektionen stellt sich hier das Problem der Repräsentativität nicht so sehr wegen der korrekten Bewertung der Ergebnisse (dafür wird das Testset verwendet), sondern damit alle möglichen Fälle abgedeckt sind. Neben dem Abdecken möglichst vieler Fälle hat aber die Häufigkeit der Tupel mit gleichen Attributwerten beim ID3 Algorithmus auch Einfluß auf die Auswahl der Attribute, nach denen die Beispielmengen aufgeteilt werden.

Da bei einer größeren Anzahl von Attributen mit eventuell zahlreichen möglichen Werten die Anzahl der Beispiele in einem repräsentativen Trainingsset sehr schnell sehr groß und damit nicht mehr handhabbar wird, müssen häufig andere Methoden verwendet werden, um ein geeignetes Trainingsset zu konstruieren. Eine entscheidende Beobachtung für die Entwicklung solcher Methoden ist die, daß die Anwendung des Kategorisierungsalgorithmus auf ein Beispiel i. a. viel weniger aufwendig ist als der Rechenaufwand, der beim Lernen anteilig auf ein Beispiel des Trainingssets entfällt. Daraus hat man folgendes Verfahren entwickelt:

### 3.2.11 Fenstertechnik

Zunächst wird mit einer kleinen Teilmenge, - einem **Fenster** - der Beispiele als Trainingsset ein Lerndurchgang berechnet. Dann werden weitere Beispiele des Trainingssets mit dem so gewonnenen Algorithmus kategorisiert. Die Beispiele, bei denen die Kategorisierung nicht stimmt, werden dem Trainingsset hinzugefügt. Anschließend wird ein neuer Lerndurchgang gestartet. Auf diese Weise soll die Anzahl der Beispiele, die von dem Algorithmus richtig kategorisiert werden, gesteigert werden. Es kann aber natürlich vorkommen, daß nach dem Hinzufügen von neuen Beispielen zum Trainingsset Beispiele, die nicht im ursprünglichen Trainingsset lagen, aber richtig kategorisiert worden waren, nicht mehr richtig kategorisiert werden.

Als Beispiel kann man annehmen, daß das Trainingsset inkonsistent ist, also zwei Dokumente $s$ und $t$ exisitieren, wie sie in Gleichung (57) definiert wurden, die nicht im zunächst gewählten Fenster liegen. Falls $s$ durch den im ersten Durchgang berechneten Algorithmus richtig kategorisiert wurde und $t$ falsch, wird $t$ im ersten kategorisierenden Kontrolldurchgang durch das Trainingsset zur Menge der Beispiele, aus denen der Algorithmus gelernt wird, hinzugefügt. Dadurch kann $s$ durch den daraus konstruierten neuen Algorithmus nicht mehr richtig kategorisiert werden. Wird ein weiterer Kontrolldurchgang durchgeführt, wird sich die Inkonsistenz zeigen. Solche Effekte können mehreren inkonsistenten Beispielen mit identischen Werten auf den vorhersagenden Attributen noch komplexer werden.

Insgesamt betont die Verwendung dieser Fenstertechnik die “schwierigen” Beispiele im Trainingsset. Insbesondere wenn das Trainingsset nur einen kleinen Ausschnitt aus

einer großen Menge möglicher Wertkombinationen darstellt, können dadurch Verzerrungen bei den Ergebnissen des Lernens auftreten.

### 3.2.12 Inkrementelles Lernen

Die Fenstertechnik wird natürlich erst richtig effektiv, wenn es Verfahren gibt, die **inkrementell lernen** können, d. h. Verfahren, bei denen ein einmal berechneter Algorithmus durch neue Trainingsbeispiele weiter verbessert werden kann. Das ist bei der vorgestellten Form des ID3 Algorithmus nicht möglich. Hier muß jedesmal mit dem neuen Trainingsset von Grund auf neu gelernt werden. Inkrementelles Lernen ist insbesondere auch bei Datensammlungen interessant, die sich im Laufe der Zeit verändern. Hier kann es auch sinnvoll sein, neue Informationen stärker zu gewichten als alte. Das gilt für alle Szenarien, bei denen sich nicht nur die Beispiele ändern, sondern auch die Situation, in der sie auftreten. Z. B. wird ein System, das Aktienwerte aufgrund von Firmendaten vorhersagen soll, die Regeln den sich ändernden Verhältnissen anpassen müssen. Dabei muß das richtige Gleichgewicht zwischen den dauerhaften Regeln der Ökonomie und den aktuellen Trends gefunden werden.

### 3.2.13 Overfitting

In vielen Anwendungsfällen kann das Trainingsset aufgrund der exponentiell wachsenden Anzahl der möglichen Kombinationen von Attributwerten nicht alle Attributwertkombinationen abdecken. Wie oben beschrieben, kann man sich dann auf ein repräsentatives Trainingsset oder auf ein Trainingsset, in dem vor allem “schwierige” Fälle enthalten sind, beschränken. In solchen Fällen kann es zu sogenanntem Overfitting kommen. Das bedeutet, daß die Regeln sich an Eigenheiten des Trainingssets orientieren, die nicht für die gesamte Datenmenge charakteristisch sind, und dann bei neuen Datensätzen schlechtere Ergebnisse liefern, als es allgemeinere Regeln oder breitere Klassifikationen getan hätten.

Das gilt insbesondere bei Verfahren wie der oben beschriebenen Fenstertechnik, die gezielt solche Beispiele ins Trainingsset aufnehmen, die schwierig zu kategorisieren sind und dadurch kein repräsentatives Trainingsset verwenden. Hier können vor allem die unterschiedlichen Wahrscheinlichkeiten, mit denen Beispiele zu erwarten sind, ins Gewicht fallen. Wird beim Lernen das Gewicht zu Lasten von häufigen Beispielen auf sehr seltene und schwierige Beispiele gelegt, kann die Leistung des resultierenden Systems auf den häufigen Beispielen nachlassen, was bei einer Gesamtbewertung natürlich viel stärker zu Buche schlagen kann, als schlechtere Leistungen auf seltenen Beispielen.

### 3.2.14 Bewertung

Für die Bewertung von KDD-Algorithmen verfährt man im Prinzip ähnlich wie bei der Bewertung von Information Retrieval Verfahren. Man verwendet ein Testset, das möglichst repräsentativ für die Verteilung der Beispiele in dem Bereich sein sollte, der mit dem erzeugten Algorithmus bearbeitet werden soll. Aufgrund der richtig bzw. falsch kategorisierten Beispiele des Testsets kann man wie in Abschnitt 2.4.3 Precision– und Recallwerte für jede Kategorie berechnen. Allerdings heißt die Precision dabei häufig **classification accuracy**. Recall wird auch **Abdeckung** oder *Coverage* genannt.

Formal läßt sich das so formulieren:

### 3.2.14.1 Accuracy und Coverage

*Sei $D = \{d_1, ..., d_m\}$ eine Menge von Dokumenten und $K = \{K_1, ..., K_r\}$ eine vorgegebene Kategorisierung von $D$. Sei $Q = \{Q_1, ..., Q_r\}$ eine Kategorisierung, die durch einen Algorithmus erzeugt wurde, dann heißt*

$$P(K_i, Q_j) := \frac{\mid K_i \cap Q_j \mid}{\mid Q_j \mid} \tag{58}$$

*Genauigkeit oder Accuracy und*

$$R(K_i, Q_j) := \frac{\mid K_i \cap Q_j \mid}{\mid K_i \mid} \tag{59}$$

*Abdeckung oder Coverage der berechneten Kategorie $Q_j$ bezüglich der gegebenen Kategorie $K_i$*

Man sieht unmittelbar, daß diese Definition eine Verallgemeinerung der Definiton 2.4.3.2 ist: Setzt man als gegebene Kategorisierung $K = \left\{r_q^{-1}(\{0\}), r_q^{-1}(\{1\})\right\}$ und als berechnete Kategorisierung $Q = \{D \setminus D_q, D_q\}$ ein, hat man die dort gegebenen Definitionen von Precision und Recall. Die Definition zeigt auch, daß in einer symmetrischen Sichtweise, wenn nämlich die vorgegebene Kategorisierung und die berechnete einfach als zwei verschiedene Kategorisierungen mit gleich vielen Kategorien betrachtet werden, die Precision durch Austauschen der Rollen der beiden Kategorisierungen aus dem Recall hervorgeht und umgekehrt.

Im allgemeinen ist es bei Kategorisierungen nicht sinnvoll, Precision-Recall-Diagramme zu benutzen, da die Algorithmen keine Ähnlichkeitswerte zu einer Anfrage liefern, sondern ein binäres Ergebnis: Ein Dokument ist entweder in der Kategorie oder nicht.

Werden Kategorien durch Attribute auf höherem Skalenniveau (also z. B. Intervallskalenniveau) definiert, kann es sinnvoll sein, auch wieder Ähnlichkeiten zu berechnen und Precision-Recall-Diagramme zu verwenden.

**Abbildung 27: Beispielmenge von Tupeln mit Kategorisierung**

| Beispiele | |
|---|---|
| 0 0 0 0 $\rightarrow K_1$ | 1 0 0 0 $\rightarrow K_1$ |
| 0 0 0 1 $\rightarrow K_4$ | 1 0 0 1 $\rightarrow K_3$ |
| 0 0 1 0 $\rightarrow K_4$ | 1 0 1 0 $\rightarrow K_1$ |
| 0 0 1 1 $\rightarrow K_3$ | 1 0 1 1 $\rightarrow K_1$ |
| 0 1 0 0 $\rightarrow K_2$ | 1 1 0 0 $\rightarrow K_1$ |
| 0 1 0 1 $\rightarrow K_5$ | 1 1 0 1 $\rightarrow K_2$ |
| 0 1 1 0 $\rightarrow K_2$ | 1 1 1 0 $\rightarrow K_1$ |
| 0 1 1 1 $\rightarrow K_1$ | 1 1 1 1 $\rightarrow K_1$ |

**Abbildung 28: Entropiewerte, nach denen die Attribute bei der Konstruktion des ID3 Baums aus den Beispielen aus Abbildung 27 selektiert werden.**

<table>
<tr><td colspan="4">$A_1$ : 2.2952<br>$A_2$ : 1.9061<br>$A_3$ : 2.5677<br>$A_4$ : 2.3944</td></tr>
<tr><td colspan="2">$A_3 = 0$:<br>$A_1$ : 2.4260<br>$A_2$ : 2.0794<br>$A_4$ : 1.9486</td><td colspan="2">$A_3 = 1$:<br>$A_1$ : 1.3862<br>$A_2$ : 1.6020<br>$A_4$ : 1.6020</td></tr>
<tr><td>$A_1 = 0$:<br>$A_2$ : 1.3863<br>$A_4$ : 0.69315</td><td>$A_1 = 1$:<br>$A_2$ : 0.6935<br>$A_4$ : 1.3863</td><td>$A_2 = 0$:<br>$A_1$ : 1.3863<br>$A_4$ : 1.3863</td><td>$A_2 = 1$:<br>$A_1$ : 1.3863<br>$A_4$ : 1.3863</td></tr>
</table>

## 3.2.15 Darstellung als Regeln

Ein Entscheidungsbaum ist eine sehr spezifische Darstellung der gewonnenen Regelmäßigkeiten. Es lassen sich auch ander Darstellungsarten wählen.

Bei der Konstruktion des Entscheidungsbaumes waren wir von einer Menge von Attributen ausgegangen, die jeweils eine bestimmte Wertemenge haben. Mit Hilfe der Attribute läßt sich ein Entscheidungsbaum leicht in eine Menge von durch AND verknüpfte Bedingungen an Attribute bringen. Dazu definieren wir zunächst für ein Attribut $A_i : D \to R_i$ und einen Wert $r_j \in R_i$ des Attributs ein Attribut – Wert Paar oder eine **elementare** oder **atomare Bedingung**, die durch Teilmengen von $D$ charakterisiert ist:

$$(A_i = r_j) := A_i^{-1}(\{r_j\}) = \{d \in D \mid A_i(d) = r_j\} \tag{60}$$

Solche elementaren Bedingungen lassen sich zu **zusammengesetzten Bedingungen** verknüpfen, dabei heißt die Verknüpfung

$$(A_i = r_k) \wedge (A_j = r_l) := A_i^{-1}(\{r_k\}) \cap A_j^{-1}(\{r_l\}) \tag{61}$$

**Konjunktion** oder AND-Verknüpfung und die Verknüpfung

$$(A_i = r_k) \vee (A_j = r_l) := A_i^{-1}(\{r_k\}) \cup A_j^{-1}(\{r_l\}) \tag{62}$$

**Disjunktion** oder OR-Verknüpfung. Schließlich läßt sich noch das **Komplement** einer Bedingung bilden:

$$\neg(A_i = r_k) := D \setminus A_i^{-1}(\{r_k\}) =: (A_i \neq r_k) \tag{63}$$

Aufgrund der Definition über Mengen lassen sich die Verknüpfungen unmittelbar auf zusammengesetzte Bedingungen übertragen. Sie stimmen mit den im Abschnitt 2.1 über Bool'sches Retrieval verwendeten AND und OR Verknüpfungen überein. Aus den De Morgan'schen Regeln lassen sich Distributivgesetze ableiten.

Als Beispiel kann die Menge der Beispiele im Blatt des linkesten Asts aus dem Baum in Abbildung 24 folgendermaßen dargestellt werden: $(A_1 = r_1) \wedge (A_2 = r_4) = A_1^{-1}(\{r_1\}) \cap A_2^{-1}(\{r_4\})$. Nehmen wir an, daß die Beispiele dieses Blattes des Baumes aus der

**Abbildung 29: Der ID3 Baum, der sich aus den Beispielen aus Abbildung 27 ergibt, mit den Beispielmengen der Knoten**

<table>
<tr><td colspan="4">0 0 0 0 → $K_1$, 0 0 0 1 → $K_4$, 0 0 1 0 → $K_4$, 0 0 1 1 → $K_3$<br>0 1 0 0 → $K_2$, 0 1 0 1 → $K_5$, 0 1 1 0 → $K_2$, 0 1 1 1 → $K_1$<br>1 0 0 0 → $K_1$, 1 0 0 1 → $K_3$, 1 0 1 0 → $K_1$, 1 0 1 1 → $K_1$<br>1 1 0 0 → $K_1$, 1 1 0 1 → $K_2$, 1 1 1 0 → $K_1$, 1 1 1 1 → $K_1$</td></tr>
<tr><td colspan="2">$A_3 = 0$:<br>0 0 0 0 → $K_1$, 0 0 0 1 → $K_4$<br>0 1 0 0 → $K_2$, 0 1 0 1 → $K_5$<br>1 0 0 0 → $K_1$, 1 0 0 1 → $K_3$<br>1 1 0 0 → $K_1$, 1 1 0 1 → $K_2$</td><td colspan="2">$A_3 = 1$:<br>0 0 1 0 → $K_4$, 0 0 1 1 → $K_3$<br>0 1 1 0 → $K_2$, 0 1 1 1 → $K_1$<br>1 0 1 0 → $K_1$, 1 0 1 1 → $K_1$<br>1 1 1 0 → $K_1$, 1 1 1 1 → $K_1$</td></tr>
<tr><td>$A_1 = 0$:<br>0 0 0 0 → $K_1$<br>0 0 0 1 → $K_4$<br>0 1 0 0 → $K_2$<br>0 1 0 1 → $K_5$</td><td>$A_1 = 1$:<br>1 0 0 0 → $K_1$<br>1 0 0 1 → $K_3$<br>1 1 0 0 → $K_1$<br>1 1 0 1 → $K_2$</td><td>$A_2 = 0$:<br>0 0 1 0 → $K_4$<br>0 0 1 1 → $K_3$<br>1 0 1 0 → $K_1$<br>1 0 1 1 → $K_1$</td><td>$A_2 = 1$:<br>0 1 1 0 → $K_2$<br>0 1 1 1 → $K_1$<br>1 1 1 0 → $K_1$<br>1 1 1 1 → $K_1$</td></tr>
<tr><td>$A_2 = 0$:<br>0 0 0 0 → $K_1$<br>0 0 0 1 → $K_4$</td><td>$A_2 = 0$:<br>1 0 0 0 → $K_1$<br>1 0 0 1 → $K_3$</td><td>$A_4 = 0$:<br>0 0 1 0 → $K_4$<br>1 0 1 0 → $K_1$</td><td>$A_1 = 0$:<br>0 1 1 0 → $K_2$<br>0 1 1 1 → $K_1$</td></tr>
<tr><td>$A_2 = 1$:<br>0 1 0 0 → $K_2$<br>0 1 0 1 → $K_5$</td><td>$A_2 = 1$:<br>1 1 0 0 → $K_1$<br>1 1 0 1 → $K_2$</td><td>$A_4 = 1$:<br>0 0 1 1 → $K_3$<br>1 0 1 1 → $K_1$,</td><td>$A_1 = 1$:<br>1 1 1 0 → $K_1$<br>1 1 1 1 → $K_1$</td></tr>
<tr><td>$A_4 = 0$:<br>0 0 0 0 → $K_1$</td><td>$A_4 = 0$:<br>1 0 0 0 → $K_1$</td><td>$A_1 = 0$:<br>0 0 1 0 → $K_4$</td><td>$A_4 = 0$:<br>0 1 1 0 → $K_2$</td></tr>
<tr><td>$A_4 = 1$:<br>0 0 0 1 → $K_4$</td><td>$A_4 = 1$:<br>1 0 0 1 → $K_3$</td><td>$A_1 = 1$:<br>1 0 1 0 → $K_1$</td><td>$A_4 = 1$:<br>0 1 1 1 → $K_1$</td></tr>
<tr><td>$A_4 = 0$:<br>0 1 0 0 → $K_2$</td><td>$A_4 = 0$:<br>1 1 0 0 → $K_1$</td><td>$A_1 = 0$:<br>0 0 1 1 → $K_3$</td><td></td></tr>
<tr><td>$A_4 = 1$:<br>0 1 0 1 → $K_5$</td><td>$A_4 = 1$:<br>1 1 0 1 → $K_2$</td><td>$A_1 = 1$:<br>1 0 1 1 → $K_1$</td><td></td></tr>
</table>

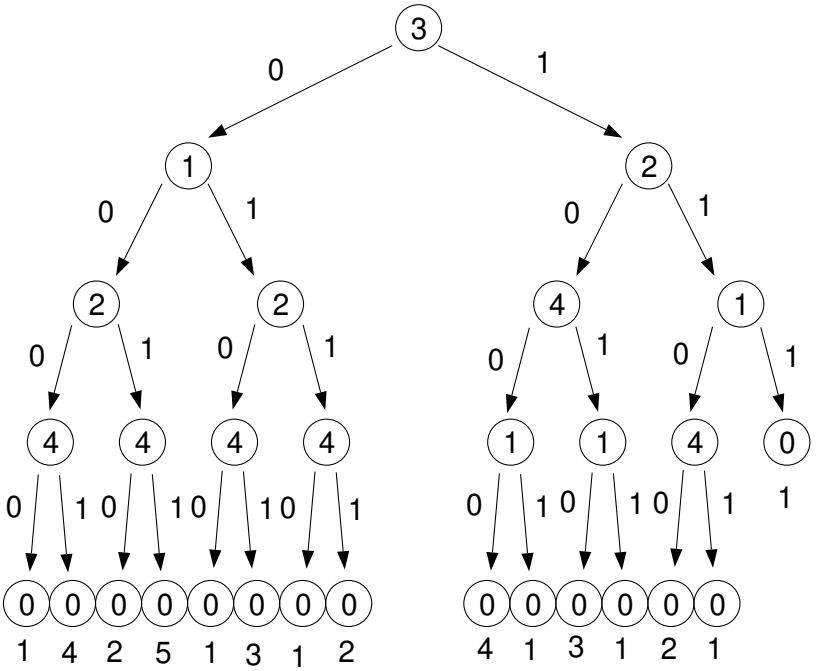
*Die Knoten des Baumes, die dieselbe Höhe haben, befinden sich zwischen den doppelten Horizontallinien der Tabelle. Bis zur dritten Ebene ist der Baum also normal dargestellt. Die Kindknoten der vierten Ebene sind untereinander dargestellt. Die letzten vier Tabellenzeilen stellen die Knoten der fünften Ebene dar. Dabei sind die viert- und drittletzte Tabellenzeile die Kindknoten der sechstletzten Tabellenzeile, die vorletze und letzte Tabellenzeile die der fünftletzten. Die Baumstruktur ist besser in Abbildung 30 zu erkennen. Um die Verbindung herzustellen, kann man jeweils das Attribut, nach dem aufgeteilt wird, suchen. Hier steht es über den Kindknoten, in Abbildung 30 steht seine Nummer in dem zu den Kindknoten gehörenden Elternknoten.*

Kategorie $K_j$ sind, so kann man diesen Ast etwas lax in eine “Produktionsregel” der Form

$$IF\ (A_1(d) = r_1) \wedge (A_2(d) = r_4)\ THEN\ d \in K_j \tag{64}$$

umwandeln.

**Abbildung 30: Der ID3 Baum aus Abbildung 29 ohne Beispielmengen**




*In dieser Darstellung sind in den Knoten die Nummern der Attribute angegeben, nach deren Werten bei der Konstruktion des Baumes die Beispiele der Knoten aufgeteilt werden, bzw. die bei der Kategorisierung überprüft werden.*

Streng genommen müßte man bei den elementaren Bedingungen immer angeben, auf welche Gesamtmenge sich die Definition bezieht. Man müßte statt Formel (60) also schreiben

$$(A_i = r_j)_D := A_i^{-1}(\{r_j\}) = \{d \in D \mid A_i(d) = r_j\} \tag{65}$$

Die Produktionsregel (64) würde dann lauten:

$$IF\ (A_1 = r_1)_{\{d\}} \wedge (A_2 = r_4)_{\{d\}}\ THEN\ d \in K_j \tag{66}$$

wobei der $THEN$ Teil dann erfüllt ist, wenn die Menge, die hinter $IF$ steht, nicht leer ist.

Produktionsregeln haben die Vorteile, daß

- sie häufig als Repräsentationsform z. B. in Expertensystemen verwendet werden
- sie für Menschen verständlich sind
- jede einzelne für sich allein verwendet werden kann
- sie sich leicht generalisieren und spezialisieren lassen

Sie haben den Nachteil, daß viele der Bedingungen, nämlich die in der Nähe der Wurzel des Entscheidungsbaumes, vielfach gespeichert werden müssen.

Man kann solche Regeln in Normalformen bringen, die eine automatisierte Verarbeitung erleichtern.

***Definition 3.2.15.1 Konjunktive Normalform*** *Sei $A_i : D \to R_i,\ i \in \{1, ..., n\}$, eine Menge von Attributen auf einer Dokumentenmenge $D$. Eine Formel ist in **konjunktiver Normalform**, wenn sie als Konjunktion von Disjunktionen von elementaren Bedingungen geschrieben ist, d. h. die Form*

$$\bigwedge_{X_1,...,X_m \subset \{\{1,...,n\},R_k\}} \left( \bigvee_{(k,r) \in X_l} A_k = r \right) \tag{67}$$

*hat.*

*Sie ist in disjunktiver Normalform, wenn sie als Disjunktion von Konjunktionen von elementaren Bedingungen geschrieben ist:*

$$\bigvee_{X_1,\ldots,X_m \subset \{\{1,\ldots,n\},R_k\}} \left( \bigwedge_{(k,r)\in X_l} A_k = r \right) \qquad (68)$$

*Dabei gilt die Notation* $\bigwedge_{i=1}^{N} D_i = D_1 \wedge \ldots \wedge D_N$ *bzw.* $\bigwedge_{i\in\{1,\ldots,N\}} D_i = D_1 \wedge \ldots \wedge D_N$ *wie bei Summenzeichen etc.*

Die Produktionsregeln, die aus einem Entscheidungsbaum gewonnen werden, der durch einen ID3 Algorithmus erzeugt wurde, haben eine noch einfachere Form: sie sind, wie das obige Beispiel, Konjunktionen von elementaren Bedingungen. Diese einzelnen elementaren Bedingungen können wegen der Kommutativität der Durchschnittsbildung beliebig umsortiert werden.

**Abbildung 31: Einige Formeln, die sich aus dem Entscheidungsbaum aus Abbildung 30 ableiten lassen**

if $(A_3(d) = 0) \wedge (A_1(d) = 0) \wedge (A_2(d) = 0) \wedge (A_4(d) = 0)$ then $d \in K_1$
if $(A_3(d) = 1) \wedge (A_2(d) = 0) \wedge (A_4(d) = 1) \wedge (A_1(d) = 0)$ then $d \in K_3$
if $(A_3(d) = 1) \wedge (A_2(d) = 1) \wedge (A_1(d) = 1)$ then $d \in K_1$

Um die Redundanz der Gesamtmenge der Regeln zu verringern, können die Regeln in eine Liste geschrieben werden, die der Reihe nach abgearbeitet wird. Solche sogenannten **Entscheidungslisten** (*decision lists*) bestehen aus Paaren aus je einer Bedingung und einer Kategorieangabe. Im letzten Element enthält die Liste nur eine leere elementare Angabe (die immer richtig ist). Um ein Beispiel zu kategorisieren, wird am Anfang der Liste beginnend für jede Bedingung geprüft, ob sie von dem Beispiel erfüllt wird. Ist das der Fall, wird die zugehörige Kategorie zugewiesen. Andernfalls wird die nächste Bedingung geprüft. Ist das letzte Paar der Liste erreicht, ohne daß eine Bedingung erfüllt wurde, wird – da dort die Bedingung ja immer erfüllt ist – die Kategorie, die zu dieser Bedingung gehört, zugewiesen. Abbildung 32 zeigt eine Liste, die aus den Beispielen aus Abbildung 27 konstruiert wurde. Sie ist im Vergleich zu dem Entscheidungsbaum in Abbildung 30 recht handlich. Berücksichtigt man allerdings die Anzahl der benötigten Vergleiche, zeigt sich, daß sie erheblich mehr Rechenaufwand benötigt als der Baum. Ob sich der größere Programmier- und Verwaltungsaufwand für den Entscheidungsbaum lohnt, hängt von dem zu bearbeitenden Problem ab.

Entscheidungslisten sind insbesondere dann interessant, wenn es eine große Kategorie gibt, in die die Mehrzahl der zu erwartenden Beispiele fällt, und nur einzelne unsystematische Ausnahmen getrennt behandelt werden müssen. Sie sind nicht ganz einfach zu handhaben, da die Reihenfolge, in der die Regeln in die Liste eingetragen werden, natürlich entscheidend für die Kategorisierung ist. Spezialfälle bzw. Ausnahmen von allgemeineren Regeln müssen am Anfang der Liste stehen, damit in diesen Fällen abgebrochen wird, bevor die allgemeineren (und i. a. häufigeren) Fälle erreicht werden. Das kann dazu führen, daß das System verhältnismäßig ineffektiv wird, da erst

**Abbildung 32: Konstruktion einer Entscheidungsliste aus den Beispielen aus Abbildung 27**

| Beispiele | Beispiele sortiert | Entscheidungsliste |
|---|---|---|
| 0 0 0 0 $\rightarrow K_1$ | $K_1$: | 1 0 0 1 $\rightarrow K_3$ |
| 0 0 0 1 $\rightarrow K_4$ | 0 0 0 0 | |
| 0 0 1 0 $\rightarrow K_4$ | 0 1 1 1 | 1 1 0 1 $\rightarrow K_2$ |
| 0 0 1 1 $\rightarrow K_3$ | 1 0 0 0 | |
| 0 1 0 0 $\rightarrow K_2$ | 1 0 1 0 | 1 * * * $\rightarrow K_1$ |
| 0 1 0 1 $\rightarrow K_5$ | 1 0 1 1 | |
| 0 1 1 0 $\rightarrow K_2$ | 1 1 0 0 | * 1 0 1 $\rightarrow K_5$ |
| 0 1 1 1 $\rightarrow K_1$ | 1 1 1 0 | |
| 1 0 0 0 $\rightarrow K_1$ | 1 1 1 1 | * 1 1 1 $\rightarrow K_1$ |
| 1 0 0 1 $\rightarrow K_3$ | | |
| 1 0 1 0 $\rightarrow K_1$ | $K_2$: | * 1 * * $\rightarrow K_2$ |
| 1 0 1 1 $\rightarrow K_1$ | 0 1 0 0 | |
| 1 1 0 0 $\rightarrow K_1$ | 0 1 1 0 | * * 0 0 $\rightarrow K_1$ |
| 1 1 0 1 $\rightarrow K_2$ | 1 1 0 1 | |
| 1 1 1 0 $\rightarrow K_1$ | | * * 1 1 $\rightarrow K_3$ |
| 1 1 1 1 $\rightarrow K_1$ | $K_3$: | |
| | 0 0 1 1 | * * * * $\rightarrow K_4$ |
| | 1 0 0 1 | |
| | | |
| | $K_4$: | |
| | 0 0 0 1 | |
| | 0 0 1 0 | |
| | | |
| | $K_5$: | |
| | 0 1 0 1 | |

*Die Tupel in der Entscheidungsliste sind als durch eine Konjunktion verknüpfte elementare Bedingungen mit den Attributen, die an der entsprechenden Position stehen, und dem angegebenen Wert zu interpretieren. Steht an einer Position ein Stern (*), so wird das entsprechende Attribut weggelassen.*

*Die Entscheidungsliste ist nicht nach einem Algorithmus konstruiert, sondern "freihändig" zusammengebaut worden. Dabei zeigt sich, daß unwillkürlich einzelne Attribute abgearbeitet werden. Es ist auch unmittelbar einsichtig, daß die Reihenfolge der Elemente der Liste entscheidend für die Richtigkeit des Ergebnisses ist.*

*Im Vergleich zu dem Entscheidungsbaum aus Abbildung 30 sieht die Entscheidungsliste mit nur acht Regeln einfach aus. Bedenkt man aber, daß gerade bei den ersten Regeln jeweils alle Attribute abgeprüft werden müssen, und daß bei dem Entscheidungsbaum für eine Entscheidung maximal vier Attribute geprüft werden müssen, so sieht man, daß der Rechenaufwand für die Kategorisierung in den meisten Fällen erheblich höher sein dürfte.*

viele Spezialfälle abgeprüft werden müssen, ehe allgemeinere Fälle bearbeitet werden können. Zudem werden Bedingungen, die inhaltlich zusammengehören, in der Liste nicht unbedingt nahe beieinander stehen, weil die strikte Sequentialität dies verhindert.

Um die strikte Abhängigkeit von der Reihenfolge aufzuheben, kann man sog. **Ripple-down Regelmengen** verwenden. Diese Mengen fassen Regeln und ihre Ausnahmen

in genesteten *if-then*-Strukturen zusammen. Durch die lokale Anbindung der Ausnahmen an die Regeln entfällt die Notwendigkeit, die Regeln anzuordnen bzw. in einer vorgegebenen Reihenfolge abzuarbeiten. Die Regeln können nach inhaltlichen Gesichtspunkten oder Häufigkeit gebildet und angeordnet werden. So können z. B. die Regeln, die Beispiele erkennen, die besonders häufig auftreten, an den Anfang der Liste gestellt werden, um die Bearbeitungszeiten zu optimieren, oder es können verschiedene Reihenfolgen gewählt werden, je nachdem, welcher Bereich eines Themas bearbeitet wird.

Abbildung 33 zeigt die Konstruktion eines Ripple-down-sets aus der Entscheidungsliste aus Abbildung 32. In diesem Fall wurde die Entscheidungsliste nach dem ersten Attribut in zwei Regeln zerlegt. Sie ließe sich natürlich auch in kleinere Teile aufbrechen, wenn z. B. in der ersten if-Abfrage keine elementare, sondern eine verknüpfte Bedingung stünde.

**Abbildung 33: Konstruktion eines Ripple-down Sets aus der Entscheidungsliste aus Abbildung 32**

| Entscheidungsliste | Ripple-down-Set |
|---|---|
| 1 0 0 1 $\rightarrow K_3$<br><br>1 1 0 1 $\rightarrow K_2$<br><br>1 * * * $\rightarrow K_1$<br><br>* 1 0 1 $\rightarrow K_5$<br><br>* 1 1 1 $\rightarrow K_1$<br><br>* 1 * * $\rightarrow K_2$<br><br>* * 0 0 $\rightarrow K_1$<br><br>* * 1 1 $\rightarrow K_3$<br><br>* * * * $\rightarrow K_4$ | if $(A_1 = 1)$ then {<br>* 0 0 1 $\rightarrow K_3$<br>* 1 0 1 $\rightarrow K_2$<br>* * * * $\rightarrow K_1$<br>}<br><br>if $(A_1 = 0)$ then {<br>if $(A_2 = 1)$ {<br>* * 0 1 $\rightarrow K_5$<br>* * 1 1 $\rightarrow K_1$<br>* * * * $\rightarrow K_2$<br>}<br>else {<br>* * 0 0 $\rightarrow K_1$<br>* * 1 1 $\rightarrow K_3$<br>* * * * $\rightarrow K_4$<br>}<br>} |

## 3.2.16 Mächtigere Repräsentationen

Die bisher für KDD Wissen besprochenen Repräsentationsformen sind auf die einzelnen Attribute und ihre Werte konzentriert. Beziehungen zwischen Attributen können nur über den Vergleich von Attributwerten ausgedrückt werden. So kann die Bedingung, daß zwei Attribute den gleichen Wert annehmen sollen ($A_i(d) = A_j(d)$ für $d \in D_q$), nur dadurch ausgedrückt werden, daß dies für jeden einzelnen Wert der Wertemengen aufgeschrieben bzw. abgeprüft wird (Vergl. Abbildung 34). Um leichter Beziehungen zwischen Attributen und Objekten herstellen zu können, kann man mächtigere Repräsentationsformen wie First Order Logic, semantische Netze, Frames und Schemata verwenden. Das hat den Vorteil, daß viele Relationen einfacher und manche

überhaupt erst dargestellt werden können; es hat den Nachteil, daß der Suchraum für mögliche Regeln enorm anwächst. Es werden dann gute Heuristiken oder sinnvolle Einschränkungen des Suchraumes benötigt, um noch in der Lage zu sein, in annehmbaren Zeiten Regeln zu finden.

**Abbildung 34: Regeln, die auch Vergleiche von Attributen zulassen, auf einer Teilmenge der Beispiele aus Abbildung 27.**

| Beispiele sortiert | | |
|---|---|---|
| K1:<br>0 0 0 0<br>0 1 1 1<br>1 0 0 0<br>1 0 1 0<br>1 0 1 1<br>1 1 0 0<br>1 1 1 0<br><br>K2:<br>0 1 0 0<br>0 1 1 0<br>1 1 0 1<br><br>K4:<br>0 0 0 1<br>0 0 1 0 | if (A1=1){<br>if (A4=1) & (A3=0) →K2<br>else →K1<br>}<br>else {<br>if (A2=0) {<br>if (A3=1) → K4<br>else{<br>if (A4=1) → K4<br>else → K1<br>}<br>}<br>if (A4=1) → K1<br>else → K2<br>} | if (A1=A4) {<br>if (A2=1) → K2<br>else {<br>if (A3=A4) → K1<br>else → K4<br>}<br>else {<br>if (A3=0 & A4 =1) → K4<br>else → K1<br>}<br>} |

*& steht für die AND Verknüpfung* $\wedge$

## 3.2.17 Top-Down und Bottom-Up Methoden

Der ID3 Algorithmus beginnt die Suche nach Regelmäßigkeiten mit allen Beispielen des Trainingssets. Diese Beispielmenge wird solange durch Bedingungen an Attributwerte differenziert, bis die Kategorien genau gefunden sind. D. h. bis durch geeignete Vereinigung der Beispielmengen, die den einzelnen Knoten zugeordnet sind, die Kategorien erzeugt werden können. Man bezeichnet dieses Vorgehen als **Top-down Methode**: Beginnend mit der allgemeinsten Beschreibung aller Beispiele, werden die Regeln durch Hinzufügen von Bedingungen an zusätzliche Attribute immer weiter ausdifferenziert, bis sie gerade die Regeln des Trainingssets beschreiben.

Das umgekehrte Verfahren, die **Bottom-up Methode**, beginnt mit den Beispielen der gesuchten Kategorien im Trainingsset und verallgemeinert solange die Regeln, wie dies möglich ist, ohne daß die Kategorisierung verletzt wird. Verallgemeinerungen oder Generalisierungen lassen sich besonders gut bei Produktionsregeln beschreiben. Produktionsregeln in konjunktiver Normalform können verallgemeinert werden, indem in den Disjunktionen elementare Bedingungen hinzugefügt werden (Siehe Abbildung 35) oder in den Konjunktionen Glieder weggelassen werden.

Formal könnte man in einem Bottom-Up Verfahren mit Produktionsregeln folgendermaßen vorgehen, um zu jeder Kategorie einer Menge von Beispielen eine einzige Produktionsregel zu generieren:

1. Sortiere die Beispiele nach Kategorien
2. Verbinde die Attribut-Wert Paare in jedem Beispiel durch AND Verknüpfungen, also zu Konjunktionen von elementaren Bedingungen
3. Verbinde die Konjunktionen innerhalb einer Kategorie disjunktiv zu einer Produktionsregel für diese Kategorie
4. Verallgemeinere diese Regeln solange durch
   - Weglassen von elementaren Bedingungen in den Konjunktionen
   - Zusammenfassen von identischen Gliedern der Disjunktion

   wie dadurch keines der Beispiele des Trainingssets falsch kategorisiert wird.

**Abbildung 35: Verallgemeinerung einer Bedingung durch Hinzufügen einer elementaren Bedingung zu einer Disjunktion**

$$\begin{aligned} ((A_i = r_s) \vee (A_i = r_t)) \wedge (A_j = r_x) \\ = (D_{i,s} \cup D_{i,t}) \cap (D_{j,x}) \\ = (D_{i,s} \cap D_{j,x}) \cup (D_{i,t} \cap D_{j,x}) \\ = ((A_i = r_s) \wedge (A_j = r_x)) \vee ((A_i = r_t) \wedge (A_j = r_x)) \\ \supset (A_i = r_s) \wedge (A_j = r_x) \end{aligned} \tag{69}$$

## 3.2.18 Nachoptimierung von Bäumen

Ein Entscheidungsbaum, der mit einem Algorithmus berechnet wurde, kann durch geeignete Umformungen in einen äquivalenten Baum transformiert werden, der beim Kategorisieren effektiver ist. Neben äquivalenten Bäumen können mit solchen Methoden auch Bäume berechnet werden, die Generalisierungen von ursprünglichen Bäumen bzw. deren Regeln sind.

## 3.2.19 Suchstrategien

Beim ID3 Algorithmus wurde in jedem Knoten eine (endgültige) Entscheidung getroffen, welches Attribut zur Aufteilung der Beispielmenge verwendet wurde. Als Auswahlkriterium wurde die eher allgemeine Entropieregel verwendet. Diese heuristische Entscheidung garantiert nicht, daß dadurch insgesamt ein optimaler Baum konstruiert wird. Sie kann nur versuchen, an dem jeweiligen Punkt die beste Entscheidung zu treffen. Diese Art Verfahren werden Hill-Climbing genannt.

### 3.2.19.1 Hill Climbing

Hill Climbing Verfahren gehen davon aus, daß die Algorithmen bzw. deren Teile, die im Laufe eines KDD-Verfahrens entwickelt werden, einen Raum aufspannen und zu jedem Algorithmus eine Bewertung in Form einer reellen Zahl gegeben ist, die seine “Güte” angibt. Weiter wird angenommen, daß

- auf dem Raum der Algorithmen eine Metrik existiert
- die Bewertungsfunktion stetig bezüglich dieser Metrik ist
- der Abstand zu einem Algrithmus, der durch einen Schritt in der Konstruktion erreicht werden kann, beschränkt (und klein) ist,

**Abbildung 36: Verallgemeinerung von Regeln, die aus einigen Beispielen aus Abbildung 27 gewonnen wurden**

| Beispiele | Beispiele sortiert |
|---|---|
| 0 0 0 0 $\rightarrow K_1$<br>0 0 0 1 $\rightarrow K_4$<br>0 0 1 0 $\rightarrow K_4$<br>0 1 0 0 $\rightarrow K_2$<br>0 1 1 0 $\rightarrow K_2$<br>0 1 1 1 $\rightarrow K_1$<br>1 0 0 0 $\rightarrow K_1$<br>1 0 1 0 $\rightarrow K_1$<br>1 0 1 1 $\rightarrow K_1$<br>1 1 0 0 $\rightarrow K_1$<br>1 1 0 1 $\rightarrow K_2$<br>1 1 1 0 $\rightarrow K_1$ | $K_1$:<br>0 0 0 0<br>0 1 1 1<br>1 0 0 0<br>1 0 1 0<br>1 0 1 1<br>1 1 0 0<br>1 1 1 0<br><br>$K_2$:<br>0 1 0 0<br>0 1 1 0<br>1 1 0 1<br><br>$K_4$:<br>0 0 0 1<br>0 0 1 0 |
| IF $(A_1 = 1) \wedge (A_2 = 0) \wedge (A_3 = 0) \wedge (A_4 = 0)$ THEN $K_1$<br>IF $(A_1 = 1) \wedge (A_2 = 0) \wedge (A_3 = 1) \wedge (A_4 = 0)$ THEN $K_1$<br>IF $(A_1 = 1) \wedge (A_2 = 0) \wedge (A_3 = 1) \wedge (A_4 = 1)$ THEN $K_1$<br><br>IF $(A_1 = 1) \wedge (A_2 = 1) \wedge (A_3 = 0) \wedge (A_4 = 0)$ THEN $K_1$<br>IF $(A_1 = 1) \wedge (A_2 = 1) \wedge (A_3 = 1) \wedge (A_4 = 0)$ THEN $K_1$<br>IF $(A_1 = 1) \wedge (A_2 = 1) \wedge (A_3 = 0) \wedge (A_4 = 1)$ THEN $K_2$ | |
| IF $(A_1 = 1) \wedge (A_2 = 0)$ THEN $K_1$<br>IF $(A_1 = 1) \wedge (A_2 = 0)$ THEN $K_1$<br>IF $(A_1 = 1) \wedge (A_2 = 0)$ THEN $K_1$<br><br>IF $(A_1 = 1) \wedge (A_2 = 1) \wedge (A_4 = 0)$ THEN $K_1$<br>IF $(A_1 = 1) \wedge (A_2 = 1) \wedge (A_4 = 0)$ THEN $K_1$<br>IF $(A_1 = 1) \wedge (A_2 = 1) \wedge (A_4 = 1)$ THEN $K_2$ | |
| IF $(A_1 = 1) \wedge (A_2 = 0)$ THEN $K_1$<br>IF $(A_1 = 1) \wedge (A_2 = 1) \wedge (A_4 = 0)$ THEN $K_1$<br>IF $(A_1 = 1) \wedge (A_2 = 1) \wedge (A_3 = 0)$ THEN $K_2$ | |

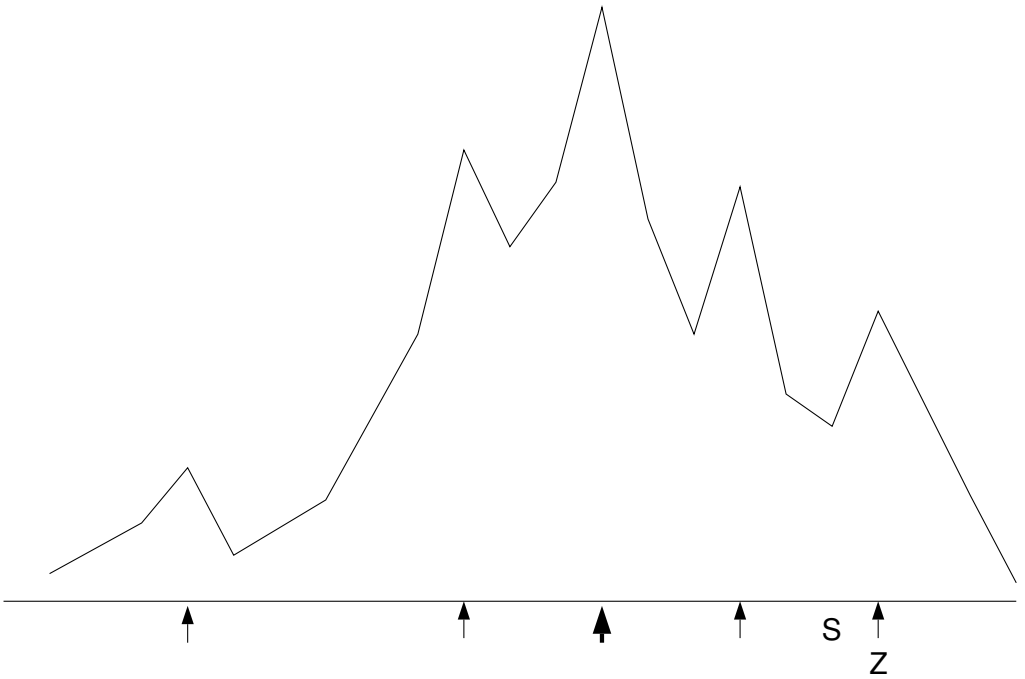
*Zunächst kann jedes Beispiel als konjunktive Produktionsregel formuliert werden. Dann prüft man, welche elementaren Bedingungen weggelassen werden können, ohne daß die Kategorisierung falsch wird. Schließlich werden identische Regeln zusammengefaßt.*

also daß die Bewertungsfunktion eine “Gütelandschaft” mit Bergen und Tälern über dem

Raum der möglichen Algorithmen bildet. Gesucht wird nun der höchste Berg in dieser Landschaft.

Die Idee eines Hill Climbing Verfahrens ist nun, von dem Ort, an dem man sich gerade befindet, (also dem gegenwärtigen Zustand des Algorithmus aus) den Schritt zu tun, bei dem der Gütewert am meisten steigt. Dieses Verfahren kann bei einer ungünstigen "Gütelandschaft" natürlich auf einen kleinen Hügel statt auf den höchsten Berg führen, sprich ein lokales Maximum erreichen statt eines globalen (Vergl. Abbildung 37).

**Abbildung 37: Gütefunktion mit lokalen Maxima**




*Die dünnen Pfeile bezeichnen lokale Maxima, der dicke Pfeil bezeichnet das globale Maximum der Gütefunktion. Falls ein Hill-Climbing Verfahen an der Stelle S startet und sich um eine Einheit weiterbewegen kann, endet es im lokalen Maximum Z und nicht im absoluten Maximum.*

Um nicht in einem lokalen Maximum zu landen, kann man versuchen, statt der Entropieregel Verfahren zu finden, die die folgenden Schritte der Entwicklung des Baumes einbeziehen. Ein optimaler Baum kann natürlich dadurch gefunden werden, daß man alle möglichen Bäume konstruiert, sie nach einem gegebenen Kriterium bewertet und den besten auswählt. Diese **vollständige Suche** stößt aber sehr schnell an Grenzen, da die Anzahl der Bäume mit der Anzahl der Attribute und Attributwerte exponentiell wächst.

Ein Kompromiß zwischen den beiden Verfahren berechnet jeweils eine begrenzte Anzahl von aussichtsreichen Alternativen vergleicht deren Effektivität und wählt schließlich die beste aus. Dieses Verfahren wird **Beam-Search** genannt. Die Auswahlkriterien für die Verfahren, die weiterverfolgt werden sollen, können dabei verschiedene Sichtweisen und Eigenschaften berücksichtigen. So können zum Beispiel verschiedene Bewertungsfunktionen verwendet werden, die sich in unterschiedlichen Situationen bewährt haben.

### 3.2.20 Überlappende Einteilungen

Neben der in Definition 3.2.2.1 beschriebenen Kategorisierung von Beispielen, bei der die Menge der Beispiele in disjunkte Teilmengen zerlegt wurde, können auch Einteilungen sinnvoll sein, bei denen sich die Mengen, die die verschiedenen Arten von Beispielen beschreiben, überlappen. Solche Einteilungen sind keine Zerlegungen der Beispielmenge mehr: Zum einen sind die Teilmengen, die sich dabei ergeben, nicht mehr disjunkt, zum anderen braucht die Vereinigung der Teilmengen nicht mehr die gesamte Beispielmenge zu ergeben. Solche Einteilungen treten typischerweise auf, wenn zu einigen Protopyen ähnliche Beispiele gesucht werden. Abbildung 38 gibt zu drei Tupeln aus Abbildung 34 die Mengen der Tupel an, die sich nur in einer Stelle unterscheiden.

In diesem einfachen Beispiel haben alle Elemente einer Menge die gleiche Ähnlichkeit zum Prototypen. Ist das nicht der Fall, wie z. B. bei einem reellwertigen Abstand, können die Tupel in eine Rangreihe bezüglich ihres Abstands zum Prototypen gebracht werden. Der Abstand kann dann als Maß der Zugehörigkeit zur Menge betrachtet werden. Solche Zugehörigkeitsfunktionen werden in der Theorie der Fuzzy Sets verwendet. Es kann auch vorkommen, daß die Bestimmung der Ähnlichkeit mit Unsicherheiten behaftet ist. Dann könnte man eine Wahrscheinlichkeit angeben, mit der ein Beispiel in eine Menge gehört.

## 3.3 Ausgewählte KDD Verfahren

Im folgenden sollen einige Verfahren vorgestellt werden, die einige der im vorigen Abschnitt beschriebenen Probleme und Möglichkeiten aufgreifen. Zunächst wird der AQ Algorithmus von Michalski, Mozetic, Hong und Lavrac (1986) beschrieben, der mit Produktionsregeln arbeitet.

### 3.3.1 AQ

Unter dem Namen AQ Algorithmus werden eine ganze Reihe Verfahren zusammengefaßt, die für verschiedene Probleme entwickelt wurden. Eine Einführung findet sich in Michalski 1983. Hier soll der auch in Holsheimer und Siebes (1994) dargestellte AQ15 Algorithmus in einer einfachen Form vorgestellt werden. Der AQ15 Algorithmus generiert IF-THEN-Regeln, die Kategorien in einem "Variable-Valued Logic System 1" ($VL_1$) genannten Formalismus beschreiben.

Im folgenden sei wieder $D$ eine Menge von konsistenten Beispielen mit Attributen $A_i : D \rightarrow R_i \ , i = 1, ..., n.$

***Definition 3.3.1.1 Selektor, Komplex, Abdeckung*** *Eine Disjunktion von elementaren Bedingungen an ein Attribut wird **Selektor** genannt. Selektoren lassen also in der Form*

$$\begin{aligned}(A_i = r_1) \vee (A_i = r_2) \vee ... \vee (A_i = r_k) &= (A_i \in \{r_1, ..., r_k\}) \\ &= A_i^{-1}(\{r_1, ..., r_k\})\end{aligned} \qquad (70)$$

*schreiben. Eine Konjunktion von Selektoren heißt **Komplex**. Ein Komplex ist also eine spezielle konjunktive Normalform. Der Bedingungsteil einer Regel besteht aus einer Disjunktion von Komplexen, die **Abdeckung** (Cover) genannt wird.*

**Abbildung 38: Teilmengen, die sich in maximal einer Stelle von einem Prototypen unterscheiden, sind nicht disjunkt.**

| Beispiele | |
|---|---|
| 0 0 0 0<br>0 0 0 1<br>0 0 1 0<br>0 1 0 0<br>0 1 1 0<br>0 1 1 1 | 1 0 0 0<br>1 0 1 0<br>1 0 1 1<br>1 1 0 0<br>1 1 0 1<br>1 1 1 0 |
| Prototyp: 0 0 1 1 | 0 0 0 1<br>0 0 1 0<br>1 0 1 1<br>0 1 1 1 |
| Schnittmenge: | 0 0 1 0<br>1 0 1 1 |
| Prototyp: 1 0 1 0 | 1 0 1 0<br>0 0 1 0<br>1 0 0 0<br>1 0 1 1<br>1 1 1 0 |
| Schnittmenge: | 1 0 0 0<br>1 0 1 0 |
| Prototyp: 1 0 0 0 | 0 0 0 0<br>1 0 0 0<br>1 0 1 0<br>1 1 0 0 |
| Fehlende Beispiele: | 0 1 0 0<br>0 1 1 0<br>1 1 0 1 |

*Die den drei Prototypen zugeordneten Mengen sind nicht disjunkt, ihre Vereinigung ist nicht gleich der Menge aller Beispiele. Das Abstandmaß, das durch die Anzahl der verschiedenen Stellen in einem Tupel definiert ist, heißt auch Hamming Distance.*

Für jede Kategorie in einer konsistenten Beispielsammlung kann man eine Abdeckung finden, die die Kategorie beschreibt. Dazu schreibt man, wie im Abschnitt 3.2.17 beschrieben, die Tupel aus einer Kategorie als Konjunktionen ihrer Attribut-Wert-Paare und bildet über diese (sehr einfachen) Komplexe eine Disjunktion. Diese Beschreibung durch Aufzählung ist aber natürlich nicht sehr nützlich. Es müssen also einfachere Beschreibungen gefunden werden.

Dazu führt AQ15 eine heuristische Suche im Raum aller Abdeckungen durch, um diejenigen zu finden, die genau die Beispiele einer Kategorie beschreiben und keine anderen. Dabei können von Nutzenden Präferenzkriterien angegeben werden, die bei mehreren passenden Regeln die bevorzugte auswählen. Solche Präferenzkriterien können zum Beispiel die Anzahl der Selektoren in einem Komplex oder auch spezifische Kostenfunktionen für einzelne Selektoren sein.

Die Suche im Raum der Abdeckungen wird konstruktiv durch Operationen, die **generalisieren**, und solche, die **spezialisieren**, durchgeführt. In einem einfachen Fall kann man dazu die folgenden Operationen verwenden:

- Hinzufügen von erlaubten Werten zu einem Selektor (Generalisierung)
- Hinzufügen eines Komplexes zu einer untersuchten Abdeckung (Generalisierung)
- Schneiden eines Komplexes mit einem weiteren Selektor (Spezialisierung)

Der Algorithmus geht entweder von einer (z. B. durch Nutzende) gegebenen Abdeckung aus, oder er beginnt mit der leeren Abdeckung. Dann werden zu einem zufälligen Beispiel, das zur Kategorie, aber nicht zur Abdeckung gehört, alle Komplexe gesucht, die das Beispiel abdecken, in denen aber kein Beispiel des Trainingssets liegt, das nicht zur Kategorie gehört. Die Vereinigung über diese Komplexe ist eine Abdeckung, die die Kategorie beschreibt.

Das zentrale Konstrukt des Algorithmus ist das des **Sterns** (*star*), das hier in einer eingeschränkten Form definiert werden soll:

***Definition 3.3.1.2 Stern*** *Der Stern eines Beispiels $d \in D$ unter der* ***Einschränkung*** *(*constraint*) $Y \subset D$ ist die Menge aller Komplexe $K$, die $d$ enthalten, aber zu $Y$ disjunkt sind, für die also gilt: $d \in K$ und $K \cap Y = \emptyset$. Ein* ***beschränkter Stern*** *(*bounded star*) ist ein Stern, der maximal eine fest vorgegebene Menge von Komplexen enthält.*

Im Folgenden werden Sterne häufig innerhalb der Elemente einer Kategorie $X \subset D$ berechnet, d. h. wir setzen $Y = D \setminus X$, und die zweite Bedingung geht in die äquivalente Bedingung $K \subset X$ über.

Der Algorithmus zum Lernen einer einzelnen Kategorie $X \subset D$ aus einer Menge von Beispielen $D$ kann folgendermaßen beschrieben werden:

### 3.3.1.3 Algorithmus AQ15

1. Sei $C \subset X$ eine vorgegebene oder die leere Abdeckung, die eine Teilmenge der Kategorie $X$ beschreibt.
2. Falls $C$ die Kategorie $X$ vollständig abdeckt, falls also $C = X$ gilt, beende den Algorithmus. Andernfalls wähle ein Beispiel $d \in X \setminus C$ aus der Kategorie, das noch nicht von der Abdeckung $C$ abgedeckt wird und berechne die maximalen Komplexe $K_1, ..., K_k$ von $d$ in $X$.
3. Wähle nach dem vorgegebenen Kriterium den besten Komplex $K_m$ aus $K_1, ..., K_k$, füge ihn (disjunktiv) der Abdeckung an ($C := C \cup K_m$) und gehe nach Schritt 2

Ein maximaler Komplex eines Beispiels $d \in D$ in $X$ ist ein Komplex $K \subset X$, für den gilt: Falls ein Komplex $Q \neq K$ mit $K \subset Q$ existiert, folgt $Q \not\subset X$. Ein Element kann mehrere maximale Komplexe haben (zwischen denen dann aber natürlich keine Teilmengenrelation herrschen kann).

Um eine Annäherung an die maximalen Komplexe eines Elementes $d \in X$ in einer Kategorie $X$ zu berechnen, kann man in Schritt 2 einen beschränkten Stern für $d$ in $X$ berechnen:

### 3.3.1.4 Algorithmus: AQ15 Partieller Stern

1. Sei $P = (K_1, ..., K_k)$ eine Liste mit $k$ Komplexen und $K_j = (A_1 \in R_1) = D$ für alle $j = 1, ..., k$. $P = (K_1, ..., K_k)$ wird ein **partieller Stern** genannt, da er nur für Teilmengen der Kategorie berechnet wird.

2. Falls $K_j \subset X$ für alle $j = 1, ..., k$ gilt, beende den Algorithmus. Andernfalls wähle ein Beispiel $r \in (D \setminus X) \cap K_j$, $j \in \{1, ..., k\}$ und berechne die maximalen Komplexe im Stern von $d$ unter der Einschränkung $\{r\}$. Diese Komplexe werden gerade durch die Selektoren der Attribute, in denen sich $d$ und $r$ unterscheiden, beschrieben.
3. Schneide die Komplexe aus $P$ paarweise mit den generierten Komplexen, wähle die nach dem externen Kriterium besten $k$ Komplexe aus, setze sie als $K_1, ..., K_k$ in den partiellen beschränkten Stern $P$ ein und gehe nach Schritt 2.

Die mit dem Algorithmus gefundene Abdeckung $C$ kann durch weitere Generalisierungen optimiert werden.

**Abbildung 39: Beispiele aus Abbildung 36 nach Kategorien sortiert**

| Beispiele nach Kateogrien sortiert | | |
|---|---|---|
| $K_1$:<br>0 0 0 0<br>0 1 1 1<br>1 0 0 0<br>1 0 1 0<br>1 0 1 1<br>1 1 0 0<br>1 1 1 0 | $K_2$:<br>0 1 0 0<br>0 1 1 0<br>1 1 0 1 | $K_4$:<br>0 0 0 1<br>0 0 1 0 |

Abbildung 40 zeigt die Anwendung des AQ15 Algorithmus auf die Beispiele aus Abbildung 39. Zunächst wird ein Beispiel $d = (0, 1, 0, 0)$ aus der zu beschreibenden Kategorie gewählt und dazu der partielle Stern berechnet. Dabei wurden die Beispiele $r \in (D \setminus X) \cap K_j$, $j \in \{1, ..., k\}$ so gewählt, daß sie sich in möglichst wenigen Attributen von dem Beispiel $d$ unterscheiden. Dadurch wird die Anzahl der maximalen Komplexe klein gehalten und das Beispiel überschaubar. Es können natürlich auch andere Auswahlkriterien angewendet werden.

Die Beschränkung des partiellen Sterns auf eine feste Anzahl von Komplexen wurde nicht verwendet (d. h. die zulässige Anzahl an Komplexen wurde so groß gewählt, daß sie nicht erreicht wurde). Dadurch werden alle Komplexe mitgeführt.

Beim ersten Beispiel $d = (0, 1, 0, 0)$ zeigt sich der Einfluß der Auswahl des Komplexes, der in die Abdeckung übernommen wird. Wäre statt des Komplexes mit dem Selektor `(A3=0)`, der andere Komplex mit `(A4=0)` gewählt worden, wäre das zweite Beispiel $d = (0, 1, 1, 0)$ bereits mit abegedeckt gewesen. Eine Regel, die eine solche Wahl bewirkt hätte, wäre z. B. die, solche Komplexe zu wählen, die möglichst viele positive Beispiele abedecken.

Beim dritten Beispiel $d = (1, 1, 0, 1)$ aus der Kategorie wurde mit $r = (0, 0, 0, 1) \in (D \setminus X) \cap K_j$ zunächst ein Beispiel gewählt, das sich in vielen Attributen von $d$ unterscheidet. Dadurch ergeben sich erheblich mehr Komplexe. In Abbildung 40 sind zunächst alle Schnittmengen aufgeführt. Man sieht aber, daß eine ganze Reihe davon doppelt auftreten. Nach dem zweiten P sind die doppelten Komplexe weggelassen.

Weiter sieht man, daß durch die Durchschnittsbildung Komplexe entstehen, die Teilkomplexe von anderen Komplexen sind. Hier könnte man im Sinne einer maximalen Generalisierung die Teilmengen weglassen. Die einzige Forderung an den partiellen Stern, die dabei immer erfüllt sein muß, ist die, daß er $d$ enthalten muß und $r$ nicht enthalten darf. Darüber hinaus können Operationen und Auswahlregeln angewendet

**Abbildung 40: Anwendung des AQ15 Algorithmus auf die Kategorien aus Abbildung 39.**

```
d=0 1 0 0
   r=1 1 0 0
   P: (A1=0)
   r=0 0 0 0
   P: (A1=0)&(A2=1)
   r=0 1 1 1
   (A3=0), (A4=0)
   P: (A1=0)&(A2=1)&(A3=0), (A1=0)&(A2=1)&(A4=0)
C=(A1=0)&(A2=1)&(A3=0)

d=0 1 1 0
   r=0 1 1 1
   P: (A4=0)
   r=1 1 1 0
   P: (A4=0)&(A1=0)
   r=0 0 1 0
   (A2=1)
   P: (A4=0)&(A1=0)&(A2=1)
C=(A1=0)&(A2=1)&(A3=0)v(A4=0)&(A1=0)&(A2=1)


d=1 1 0 1
   r=0 0 0 1
   P: (A1=1), (A2=1)
   r=1 1 1 0
   (A3=0), (A4=1)
   P: (A1=1)&(A3=0), (A2=1)&(A3=0), (A1=1)&(A4=1), (A2=1)&(A4=1)
   r=1 1 0 0
   (A4=1)
   P: (A1=1)&(A3=0)&(A4=1), (A2=1)&(A3=0)&(A4=1),
      (A1=1)&(A4=1)&(A4=1), (A2=1)&(A4=1)&(A4=1)
   r=0 1 1 1
   (A1=1), (A3=0)
   P: (A1=1)&(A3=0)&(A4=1), (A1=1)&(A2=1)&(A3=0)&(A4=1),
      (A1=1)&(A4=1), (A1=1)&(A2=1)&(A4=1),
      (A1=1)&(A3=0)&(A4=1), (A2=1)&(A3=0)&(A4=1),
      (A1=1)&(A3=0)&(A4=1), (A2=1)&(A3=0)&(A4=1)
   P: (A1=1)&(A3=0)&(A4=1), (A1=1)&(A2=1)&(A3=0)&(A4=1),
      (A1=1)&(A4=1), (A1=1)&(A2=1)&(A4=1), (A2=1)&(A3=0)&(A4=1)
   r=1 0 1 1
   (A2=1), (A3=0)
   P: (A1=1)&(A2=1)&(A3=0)&(A4=1), (A1=1)&(A2=1)&(A3=0)&(A4=1),
      (A1=1)&(A2=1)&(A4=1), (A1=1)&(A2=1)&(A4=1),
      (A2=1)&(A3=0)&(A4=1), (A1=1)&(A3=0)&(A4=1),
      (A1=1)&(A2=1)&(A3=0)&(A4=1), (A1=1)&(A3=0)&(A4=1),
      (A1=1)&(A2=1)&(A3=0)&(A4=1), (A2=1)&(A3=0)&(A4=1)
   P: (A1=1)&(A2=1)&(A3=0)&(A4=1), (A1=1)&(A2=1)&(A4=1),
      (A2=1)&(A3=0)&(A4=1), (A1=1)&(A3=0)&(A4=1)
C=(A1=0)&(A2=1)&(A3=0)v(A4=0)&(A1=0)&(A2=1)v(A2=1)&(A3=0)&(A4=1)
```

*Durchführung des AQ15 Algorithmus zur Beschreibung der Kategorie $K_2$ aus Beispiel 39. Die eingerückten Absätze sind die Berechnung des partiellen Sterns zu den ausgewählten Beispielen $d$. & steht für die Durchschnittbildung bzw. AND Verknüpfung. Im letzten Block sind unter P zunächst alle Komplexe angegeben, die sich durch die Durchschnittsbildung ergeben. Danach sind nach dem zweiten P die doppelten Komplexe weggelassen.*

werden, die sinnvoll erscheinen, um das Verfahren zu vereinfachen und zu beschleunigen. So könnte man wieder vor allem solche Komplexe übernehemen, die möglichst viele Beispiele aus der zu beschreibenden Kategorie enthalten, und möglichst wenige, die nicht in diese Kategorie fallen. Man könnte so, im Gegensatz zur oben vorgeschlagenen Verallgemeinerung, solche Komplexe weglassen, die Obermengen von anderen Komplexen sind und noch Gegenbeispiele enthalten während die Untermengen ganz in der Kategorie liegen. So ist im letzten Block von Abbildung 40 `(A1=1)&(A4=1)` Obermenge von `(A1=1)&(A3=0)&(A4=1)`. Der erste Komplex enthält aber mit $(1, 0, 1, 1)$ noch ein Beispiel, daß nicht in der zu beschreibenden Kategorie liegt, der zweite enthält dagegen keines mehr. In diesem Fall ist es klar, das der erste Komplex zugunsten des zweiten weggelassen werden kann. Wenn es aber verschiedenen Spezialisierungen gäbe, die alle zwar weniger, aber doch immer Beispiele enthalten, die nicht in der Kategorie liegen, wäre es nicht so eindeutig, welche Spezialisierung günstiger ist.

Das betrachtete Beispiel enthält einen großen Teil der möglichen Tupel. In vielen praktischen Fällen werden vergleichsweise wenige der möglichen Tupel auch tatsächlich im Trainingsset auftreten. Sind die Wertebereiche der Attribute z. B. die reellen Zahlen oder reelle Intervalle, so schneidet die Konstruktion über maximale Komplexe nur jeweils die Punkte der Gegenbeispiele heraus. Es ist aber z. B. bei Ergebnissen physikalischer Experimente oder bei medizinischen Tests sehr unwahrscheinlich, daß nur die exakten Werte der Beispiele aus der Kategorie herausfallen. Viel wahrscheinlicher ist es, daß ganze Intervalle entfernt werden müßten, für die ein Gegenbeispiel jeweils nur ein Prototyp ist. In diesen Fällen kann es sinnvoll sein, neben den hier betrachteten Spezialisierungen auch Generalisierungen zu verwenden.

#### 3.3.1.5 Generalisierungsoperationen

Michalski (1983) gibt die folgende Aufstellung von Generalisierungsoperationen, zu denen sich natürlich jeweils die umgekehrten Spezialisierungsoperationen ableiten lassen.

- Weglassen einer Bedingung (*dropping condition rule*) in einer Konjunktion:

$$(A_1 = r_1) \wedge (A_2 = r_2) \subset (A_1 = r_1) \tag{71}$$

- Hinzufügen einer Alternative (*adding condition rule*) in einer Disjunktion:

$$(A_1 = r_1) \subset (A_1 = r_1) \vee (A_2 = r_2) \tag{72}$$

- Erweitern eines Wertebereiches (*extending reference rule*) in einem Selektor:

$$(A_1 \in X) \subset (A_1 \in Y) \tag{73}$$

  wenn $X \subset Y \subset R_1$ gilt
- Als Spezialfall davon: Schließen eines Intervalls (*closing interval rule*) in einem Selektor:

$$(A_1 \in \{1, 3\}) \subset (A_1 \in [1, 3]) \tag{74}$$

  für $R_1 = [0, 5]$
- Eine Konjunktion in eine Disjunktion umwandeln (*turning conjunction into disjunction rule*):

$$(A_1 = r_1) \wedge (A_2 = r_2) \subset (A_1 = r_1) \vee (A_2 = r_2) \tag{75}$$

In der Liste der Softwareprodukte, die von G. Piatetsky-Shapiro (html) im WWW geführt wird, finden sich folgende kommerziellen und public domain Produkte, die Regeln verarbeiten bzw. aus Beispielen extrahieren:

- BRUTE
- CN2
- Data Surveyor
- EMERALD
- PQ->R
- QM
- Q-YIELD
- SIPINA-W

Die aktuellen Beschreibungen und Links zu den Anbietern finden sich unter http://info.gte.com/~kdd/siftware.html.

### 3.3.2 Externe Wissensstrukturen

Im Abschnitt 3.2.7 über verschiedene Wertebereiche von Attributen hatten wir Nominalskalen, Ordinalskalen, Intervallskalen und rationale Skalen betrachtet. Man kann Wertebereiche von Attributen aber auch noch anders strukturieren. Man kann zum Beispiel bestimmte Teilmengen aus einem Wertebereich betrachten. So könnte man die Lebenszeit einer Person einteilen in Kindheit, Jugend, Berufsfähigkeit, Rentenalter, oder in die verschiedenen Grade der Geschäftsfähigkeit: 0–6, 7–14, 14–16, 16–18, 18–21, 21–25, 25–40, > 40. Wieder eine andere Einteilung wären z. B. Jahre, in denen der Geburtstag auf einen Sonntag fällt, oder, für Personen, die an einem 29. Februar geboren sind, Schaltjahre.

Diese verschiedenen Teilmengen bilden eine Teilordung bezüglich der Enthaltensrelation: Bei manchen Paaren von Teilmengen ist die eine in der anderen enthalten, bei anderen gibt es keine solche Beziehung. Nimmt man noch den gesamten Wertebereich dazu, so ist jede Teilmenge in dieser Menge enthalten. Man nennt den Wertebereich dann ein maximales Element der Teilordnung.

***Definition 3.3.2.1 Teilordnung*** *Sei $R$ eine Menge und $\prec$ eine Relation, die zwischen einigen der Elemente aus $R$ besteht. Sei weiter $m \in R$ ein Element, für das gilt $x \prec m \; \forall x \in R$, Dann heißt das Paar $(R, \prec)$ eine **teilgeordnete Menge** mit **maximalem Element** $m$.*

Auf einer Menge kann es mehrere Teilordnungen geben.

Durch Teilordungen kann **externes Wissen** oder **Hintergrundwissen** (*background knowledge*) gespeichert werden, das zur Erschließung allgemeiner Regeln aus Beispielen genutzt werden kann. So kann durch Aufsteigen in der Teilordnung generalisiert werden: Im obigen Beispiel wäre das dritte Lebensjahr z. B. ein Teil der Kindheit, oder umgekehrt die Kindheit eine Verallgemeinerung des dritten Lebensjahres. Ein KDD-Verfahren, das mit solchen Teilordnungen arbeitet, soll im folgenden vorgestellt werden.

### 3.3.3 DBlearn (Han, Cai & Cercone 1992)

DBLearn arbeitet auf relationalen Datenbanken und verwendet zusätzlich Hintergrundwissen in Form von Teilmengen von Attributwerten, die mit Bezeichnungen versehen sind. Einige solche **Konzepthierarchien** aus Han,Cai & Cercone (1992) sind in Abbildung 41 dargestellt.

Formal kann man in der Terminologie des AQ Algorithmus eine Konzepthierarchie als verschiedene Selektoren zu einem geeigneten Attribut $A_i : D \rightarrow R_i$ auffassen: In elementaren Bedingungen werden die Elemente des Wertebereichs $R_i$ einzeln verwendet, Selektoren sind als Disjunktionen von elementaren Bedingungen oder Teilmengen des Wertebereichs definiert (vergl. Definition 3.3.1.1):

$$\begin{aligned}(A_i = r_1) \vee (A_i = r_2) \vee ... \vee (A_i = r_k) \\ = (A_i \in \{r_1, ..., r_k\}) \\ = A_i^{-1}(\{r_1, ..., r_k\})\end{aligned}$$

Hat nun die Teilmenge $\{r_1, ..., r_k\} \subset R_i$ in der Menge der beschriebenen Objekte eine Bedeutung bzw. eine Bezeichnung, wie etwa die Lebensjahre 1–14 als Kindheit bezeichnet werden können, kann man diese Bezeichnung für den Selektor übernehmen und damit eine sprachlich einfachere Regel formulieren. Ein solcher Selektor kann auch als **Konzept** (*concept*) bezeichnet werden. Allerdings ergibt sich dabei das Problem, daß man definieren müßte, was ein “Begriff” oder eine “Bedeutung” ist. Um das zu vermeiden, werden die Begriffe Konzept bzw. Konzepthierarchie im folgenden nur dort verwendet, wo sie bei Han, Cai & Cercone (1992) auftreten. Man kann sich unter einem Konzept hier einen (von außen) vorgegebenen Selektor vorstellen.

Umgekehrt kann man aus “externem Wissen”, also Alltagswissen und Wissen von Experten, Konzepthierarchien in Form einer Teilordnung konstruieren und Verallgemeinerungen nur zu solchen Selektoren erlauben, die mit Teilmengen aus der Konzepthierarchie gebildet werden. So sind in Abbildung 41 die Städtenamen zunächst nach Provinzen, dann nach Ländern etc. zusammengefaßt. In dieser Vorgabe steckt also das Wissen über die (politische) Zugehörigkeit der Städte.

In real existierenden Datenbanken werden im allgemeinen die Attribute nicht so “schön” vorliegen, daß mit ihnen direkt Konzepthierarchien gebildet werden können. Man kann aber häufig vorhandene Attribute kombinieren, um Konzepthierarchien zu konstruieren. In dem von Han, Cai & Cercone (1992) verwendeten Beispiel einer Studentendatenbank gibt es die Attribute “city”, “province” und “country”. Diese Attribute werden zu einem Attribut “place” kombiniert, zu dem eine Konzepthierarchie konstruiert wird. Dabei macht man sich zunutze, daß die vorhandenen Attribute in einem Beispiel bereits eine Hierarchie beschreiben: Die im Attribut “city” angegebene Stadt liegt in der im Attribut “province” angegebenen Provinz, die wiederum im in “country” angegebenen Land liegt. Der Wertebereich des Attributs “province” kann also als erste Stufe in der Konzepthierarchie verwendet werden, der des Attributes “country” als zweite, etc. Die Zugehörigkeiten zu den Konzepten können einfach aus den Tupeln der Datenbank abgelesen werden, in denen die entsprechenden Werte vorkommen (falls die Datenbank in diesem Teilbereich konsistent ist).

Andere Konzepthierarchien lassen sich schlechter aus den Attributen vorhersagen. So könnte man als Konzepthierarchie die Sprache, die in der Geburtsstadt gesprochen wird, verwenden. In diesem Fall müßte zusätzliches Wissen über die Sprachen, die in verschiedenen Städten und Provinzen gesprochen werden, eingebracht werden. Zudem würden sich natürlich auch Probleme mit mehrsprachigen Städten ergeben.

**Abbildung 41: Konzepthierarchien aus (Han, Cai & Cercone 1992)**

| |
|---|
| $\{biology, chemistry, computing, ..., physics\} \subset science$ * |
| $\{literature, music, ..., painting\} \subset art$ * |
| $\{science, art\} \subset ANY(major)$ |
| $\{freshmen, sophomore, junior, senior\} \subset undergraduate$ * |
| $\{M.S., M.A., Ph.D.\} \subset graduate$ * |
| $\{undergraduate, graduate\} \subset ANY(status)$ |
| $\{Burnaby, ..., Vancouver,\ Victoria\} \subset British\ Columbia$ * |
| $\{Calgary, ..., Edmonton,\ Lethbridge\} \subset Alberta$ * |
| $\{Hamilton, Toronto, ..., Waterloo\} \subset Ontario$ * |
| $\{Bombay, ..., New\ Delhi\} \subset India$ * |
| $\{Beijing, Nanjing, ..., Shanghai\} \subset China$ * |
| $\{China, India, Germany, ..., Switzerland\} \subset foreign$ |
| $\{Alberta, British\ Columbia, ..., Ontario\} \subset Canada$ |
| $\{foreign, Canada\} \subset ANY(place)$ |
| $\{0.0 - 1.99\} \subset poor$ * |
| $\{2.0 - 2.99\} \subset average$ * |
| $\{3.0 - 3.49\} \subset good$ * |
| $\{3.5 - 4.0\} \subset excellent$ * |
| $\{poor, average, good, excellent\} \subset ANY(grade)$ |

*Die einzelnen Blöcke der Tabelle enthalten jeweils eine Hierarchie für ein Attribut. Die mit einem Stern bezeichneten Zeilen enthalten Elemente aus den Wertemengen der jeweiligen Attribute (die wiederum nur bei den maximalen Elementen der Teilordnungen ANY angegeben sind.)*

DBLearn erzeugt allgemeine Regeln einer vorgegebenen Komplexität über eine Kategorie in Bezug auf eine Auswahl von Attributen und Hintergrundwissen über diese Attribute in Form von Konzepthierarchien, indem die Tupel aus der Datenbank, die in die Kategorie fallen und in denen die ausgewählten Attribute auftreten, nach verschiedenen Regeln generalisiert werden. Die verwendeten Tupel können bei relationalen Datenbanken aus einer oder mehreren Relationen stammen.

Han, Cai & Cercone (1992) unterscheiden drei verschiedene Arten von Regeln:

- **charakteristische Regeln** (*characteristic rules*) beschreiben Eigenschaften, die von allen (oder zumindest vielen) Beispielen einer Kategorie erfüllt werden. Dabei wird nichts darüber ausgesagt, wieviele andere Beispiele diese Regeln auch erfüllen.
- **Unterscheidungsregeln** (*discriminant rules*) beschreiben Eigenschaften, die die Beispiele einer Kategorie von denen einer anderen Kategorie unterscheiden.
- **Quantitative Regeln** (*quantitative rules*) sind Regeln, bei denen angegeben wird, für welchen Anteil der Beispiele einer Kategorie sie gelten.

Zur Generierung dieser Regeln werden unterschiedliche Varianten des Algorithmus' verwendet. Für die quantitativen Regeln werden die Tupel der Kategorie um ein Attribut

“vote” mit dem Anfangswert Eins erweitert, das als Wertebereich die natürlichen Zahlen hat und bei Verallgemeinerungen angibt, auf wieviele Beispiele sich eine verallgemeinerte Regel stützt.

DBLearn ist wie AQ ein Bottom-Up Verfahren, bei dem zunächst die Tupel der Kategorie als Komplexe aus elementaren Bedingungen dargestellt werden. Auf diese Komplexe werden dann bei der Generalisierung die folgenden Regeln und Strategien angewendet:

#### 3.3.3.1 Nur Selektoren generalisieren

Bei der Generalisierung werden nur einzelne Attribute, also Selektoren bearbeitet. Dabei dürfen in Selektoren auch Bezeichner für Teilmengen von Attributwerten aus den Konzepthierarchien verwendet werden.

#### 3.3.3.2 Aufsteigen in Konzepthierarchien

Liegt die Anzahl der verschiedenen Attributwerte und Konzeptbezeichner eines Attributes in den Komplexen höher als eine vorgegebene Schwelle, werden die Attributwerte oder Konzeptbezeichner durch den nächsthöheren Konzeptbezeichner in der Konzepthierarchie ersetzt. Das ist die Generalisierungsoperation “Erweitern eines Wertebereichs” aus Abschnitt 3.3.1.5. Es ist gleichzeitig die einzige Generalisierungsregel, die in DBLearn verwendet wird.

#### 3.3.3.3 Selektoren mit dem maximalen Element der Teilordnung weglassen

Bezeichnet der Konzeptbezeichner eines Attributes das maximale Element der Teilordnung, also den Wertebereich des Attributes, kann dieses Attribut weggelassen werden, denn bei der Konstruktion der Komplexe ändert der Schnitt mit der Gesamtmenge der Beispiele den Komplex nicht.

#### 3.3.3.4 Gleiche Tupel zusammenfassen

Tupel, die bis auf das Attribute “vote” identisch sind, werden zu einem Tupel zusammengefaßt. Dabei wird das Attribut “vote” gleich der Summe der Werte der “vote” Attribute der zusammengefaßten Tupel gesetzt.

#### 3.3.3.5 Einfachere Form erzwingen

Falls die Anzahl der Komplexe über einer vorgegebenen Schwelle liegt (also die Regel noch zu wenig allgemein ist), und trotzdem die Bedingung aus der Regel zum Aufsteigen in der Konzepthierarchie für keines der Attribute erfüllt ist, wird die Schwelle aus dieser Regel gesenkt.

Der Algorithmus zur Extraktion von charakteristischen Regeln zu einer Kategorie kann jetzt folgendermaßen geschrieben werden.

#### 3.3.3.6 Eingabe:

- Die Beispiele (Relationen) aus der Datenbank, in denen die Kategorie beschrieben wird.
- Die Menge $A = \{A_1, ..., A_s\}$ der Attribute, mit denen die Kategorie beschrieben werden soll, und ihre Konzepthierarchien.
- Grenzwert $k$ für die maximal zulässige Anzahl von verschiedenen Attributwerten je Attribut in der Menge der zu erzeugenden Komplexe.
- Grenzwert $n$ für die maximal zulässige Anzahl an Komplexen.

### 3.3.3.7 Algorithmus

1. Bilde Komplexe aus den Selektoren mit den relevanten Attributen und den Werten der Attribute in den Beispielen der Kategorie.
2. Falls die Anzahl der Komplexe $< n$ und die Anzahl der verschiedenen Attributwerte für alle Attribute $< k$ ist, gehe nach 7.
3. Für alle Attribute $A_i \in A$: Solange die Anzahl der verschiedenen Attributwerte $\geq k$ ist, steige mit allen Werten in der Konzepthierarchie von $A_i$ auf. (Regel 3.3.3.2).
4. Entferne alle Attribute mit maximalem Element ihrer Konzepthierarchie aus $A$.
5. Fasse gleiche Tupel zusammen.
6. Falls die Anzahl der Komplexe $\geq n$ ist, setze $k$ gleich der Maximalzahl verschiedener Attributwerte eines Attributs und gehe nach 3.
7. Versuche die gewonnen Regeln noch zu vereinfachen, verbinde die verbliebenen Komplexe zu einer Abdeckung und beende den Algorithmus.

Ein Beispiel für die Durchführung des Algorithmus findet sich in Han, Cai & Cercone (1992)

Die Aussagen, die mit diesem Algorithmus gewonnen werden können, sind von der Form: Wenn ein Beispiel in der Kategorie liegt, nehmen die und die Attribute die und die Werte an. Wichtig ist festzuhalten, daß es das Ziel des Algorithmus ist, Aussagen, die nicht falsch sind, in Form einer Abdeckung mit einer vorgegebenen Maximalzahl von Disjunktionen und einer vorgegebenen Maximalzahl von Konjunktionen in den Komplexen zu erzeugen. Die Bezeichnung “charakteristische Regeln” legt nahe zu vermuten, daß sich diese Regeln wie die charakteristische Funktion einer Menge ($Y \subset X,\ f : X \to \{0,1\},\ f(x) = \begin{cases} 1 & falls\ x \in Y \\ 0 & sonst \end{cases}$) umkehren lassen. Das ist aber nicht der Fall. Welche Eigenschaften einer Kategorie extrahiert werden und welche nicht, hängt vor allem von dem vorhandenen Hintergrundwissen ab. Hat die Konzepthierarchie einen kleinen Verzweigungsgrad, ist die Chance des zugehörigen Attributes, in der extrahierten Regel vorzukommen, hoch, hat sie einen hohen Verzweigungsgrad, ist sie eher gering. So können seltene Ausnahmen bereits zur Entfernung eines Attributs führen, auch wenn die große Mehrzahl der Werte sich auf eine kleine Menge beschränkt.

Um für eine charakteristische Regel eine quantitative Regel zu erzeugen, verwendet man das “vote” Attribut. Es gibt für die verbleibenden Komplexe an, auf wieviele Beispiele sie sich stützen. Bei der Konstruktion der Abdeckung kann man für jeden Komplex den entsprechenden Prozentwert angeben.

Um **Unterscheidungsregeln** zu lernen, müssen zwei Kategorien, zwischen denen unterschieden werden soll, angegeben werden. In beiden Kategorien wird dann simultan der Verallgemeinerungsalgorithmus durchgeführt. Dadurch brauchen die entstehenden Teilmengen der Beispielmenge, die jeweils die Kategorien enthalten, nicht mehr disjunkt zu sein. Deshalb werden solche Komplexe, die durch die Verallgemeinerungen in beiden Mengen vorkommen, markiert und nicht weiter berücksichtigt. Bleiben am Schluß in der ersten Menge unmarkierte Komplexe übrig, gibt es also Komplexe in der ersten Abdeckung, die nicht durch die zweite Abdeckung überdeckt werden, werden diese Komplexe als Unterscheidungsregeln ausgegeben.

Ein Beispiel ist in Han, Cai & Cercone (1992) angegeben.

Auch hier geht es nicht darum, die beste Unterscheidung zwischen zwei Kategorien zu finden, sondern darum, Mengen zu finden, die sich in einer vorgegebenen Form schreiben lassen und kein Beispiel enthalten, das in der entstehenden Obermenge der zweiten Kategorie liegt. In dem angegebenen Beispiel zeigt sich das besonders

darin, daß eine weitere Vereinfachung der gefundenen Regel durch Weglassen der Studienrichtung und der Benotung das selbe Ergebnis erzielen würde.

DBLearn ist in einer Erweiterung von SQL implementiert (siehe Han, Cai & Cercone (1992))

Während in DBLearn die zu extrahierenden Regeln nur durch zwei Parameter bestimmt werden könnten, können Nutzende im folgenden Verfahren erheblich genauer bestimmen, welche Art von Regeln sie extrahieren wollen.

Zu DBLearn existiert auch ein Video sowie die Beschreibung in der Liste der Softwareprodukte, die von G. Piatetsky-Shapiro (html) im WWW geführt wird. Die aktuelle Beschreibung und Links zu den Autoren finden sich am besten unter
http://info.gte.com/~kdd/siftware.html.

### 3.3.4 Assoziative Regeln (Klemettinen et al 1994)

Ziel des in (Klemettinen, Mannila, Ponkainen, Toivonen and Verkamo 1994) beschriebenen Verfahrens ist es, statistische Aussagen der Form "80 % der Studenten, die Kurse in Basic und Pascal belegt haben, belegen auch einen Kurs in C" zu erschließen und die interssantesten davon auszuwählen. Dazu wird folgende Definition getroffen:

***Definition 3.3.4.1 Assoziative Regel*** *Sei $D = \{d_1, ..., d_m\}$ eine Beispielmenge und $A = \{A_1, ..., A_n\}$, $A_i : D \to \{0, 1\}$ eine Menge von $n$ binären Attributen. Sei weiter $W \subset A$ und $B \in A \setminus W$, dann heißt der Ausdruck $W \Rightarrow B$ eine **assoziative Regel über** $D$. Sind weiter $\gamma, \sigma \in \mathbb{R}$, dann **erfüllt** $D$ **die assoziative Regel** $W \Rightarrow B$ zur **Basis** $\sigma$ und zum **Grad** $\gamma$, wenn*

$$| \left( \bigwedge_{A_i \in W} (A_i = 1) \right) \wedge (B = 1) | \geq m\sigma \tag{77}$$

*und*

$$\frac{| \left( \bigwedge_{A_i \in W} (A_i = 1) \right) \wedge (B = 1) |}{| \bigwedge_{A_i \in W} (A_i = 1) |} \geq \gamma \tag{78}$$

*gilt. Das wird in der Form*

$$W \Rightarrow B \ (\gamma, \sigma) \tag{79}$$

*ausgedrückt.*

*Der Wert $\sigma$ wird auch* support threshold *und der Wert $\gamma$* confidence threshold *genannt.*

Eine Beispielmenge erfüllt eine assoziative Regel also dann zur Basis $\sigma$, wenn der Anteil der Beispiele, bei denen in allen Attributen, die in der Regel genannt sind, eine $1$ steht, größer als $\sigma$ ist. Sie erfüllt sie zum Grad $\gamma$, wenn der Anteil der Beispiele, bei denen das Attribut $B$ gleich $1$ ist, unter denen, bei denen die Attribute aus $W$ gleich $1$ sind, größer ist als $\gamma$.

### 3.3.4.2 Beispiel

In einer Datenbank über Teilnahme an Lehrveranstaltungen im Fach Informatik könnte die assoziative Regel stehen:

`Einführung in UNIX ⇒ Programmieren in C (0.84, 0.34)`

Das würde bedeuten, daß 84% der Studierenden, die eine Einführung in UNIX belegt haben auch eine Einführung in C belegt haben, und daß insgesamt 34% der Studierenden beide Veranstaltungen belegt haben.

Sobald die Anzahl der Attribute größer wird, gibt es sehr viele solche assoziativen Regeln, auch wenn die Basis und der Grad groß gewählt werden.

**Abbildung 42: Anzahl der Regeln aus zwei Beispielsammlungen**

| | Beispiele | Attribute | Regeln | | | |
|---|---|---|---|---|---|---|
| | | | $\gamma = 0.00$ | $\gamma = 0.7$ | $\gamma = 0.9$ | $\gamma = 1.0$ |
| Veranstaltungsbelegung $\sigma = 0.01$ | 1066 | 112 | 2010 | 420 | 99 | |
| Computerversand $\sigma = 0.06$ | 524 | 894 | 4817 | 4468 | 3724 | 3548 |

*Die obere Zeile beschreibt die Beleglisten von Studierenden in Informatik. Attribute sind die verschiedenen Veranstaltungen. Die untere Zeile beschreibt Versandvorgänge in einem Computerversandhandel mit den verschiedenen Rechnerkomponenten als Attributen. Die mittlere Anzahl von belegten Attributen ist bei der Veranstaltungsbelegung gleich fünf, bei den Computerteilen gleich* $15.2$

In Abbildung 42 sind zwei Beispiele aus Klemettinen et al. (1994) angegeben. In beiden Fällen ist die Anzahl der Regeln hoch. Bei den Beispielen aus dem Computerversand ist sie allerdings noch erheblich größer als bei den Belegungen von Lehrveranstaltungen. Das kann zum einen dadurch erklärt werden, daß die Rechner eher in standardisierten Zusammenstellungen geliefert werden, zum anderen aber auch dadurch, daß die Anzahl der Attribute höher und die Anzahl der Beispiele niedriger ist. Zudem ist die mittlere Anzahl von belegten Attributen im zweiten Beispiel höher.

In beiden Fällen ist die Anzahl der Regeln zu groß, um sie noch sinnvoll "von Hand" sichten und auswerten zu können. Zudem gibt es Regeln, die sich bereits aus der Natur der Daten ergeben und daher nicht weiter interessant sind. So ist die Regel

`Design and Analysis of Algorithms (graduate course)⇒ Fundamentals of ADP (basic course) (0.97, 0.03)`

die besagt, daß 97% der Studierenden, die den Kurs "Design and Analysis of Algorithms" belegt haben, im Laufe ihres Studiums auch den Kurs "Fundamentals of automated data processing" belegt hatten, nicht sonderlich erstaunlich.

Zur Selektion von Regeln verwenden Klemettinen et al. (1994) Hintergrundwissen in Form von **Teilordnungen der Attribute**. Das ist formal ein anderes Vorgehen als bei DBLearn, inhaltlich steckt dahinter aber die gleiche Idee: Durch eine Strukturierung der Elemente, mit denen umgegangen wird, eine stufenweise Verallgemeinerung der gesuchten bzw. gemachten Aussagen zu erreichen. Der Grund für den formalen Unterschied liegt darin, daß bei der Selektion von assoziativen Regeln im Grunde ein Extraktionsverfahren "zweiter Ordung" durchgeführt wird.

***Definition 3.3.4.3 Template*** *Sei wieder $A = \{A_1, ..., A_n\}$, $A_i : D \rightarrow \{0, 1\}$ eine Menge von Attributen. Seien $W_1, ..., W_{k+1} \subset A$ Teilmengen der Attribute, auf denen eine Teilordung besteht. Ein Ausdruck $W_1, ..., W_k \Rightarrow W_{k+1}$ heißt (einfaches)* ***Template****. Eine assoziative Regel $B_1, ..., B_k \Rightarrow B_{k+1}$ wird von dem Template erfaßt, wenn, gegebenenfalls nach geeigneter Umordnung, für $i = 1, ..., k + 1$ gilt $B_i \in W_i$.*

*Die $W_i$ können durch Terme der Form $W_i+$ und $W_i*$ ersetzt werden, wobei $W_i+$ bedeutet, daß ein oder mehrere Attribute aus $W_i$ auftreten, und $W_i*$, daß Attribute aus $W_i$ auftreten können aber nicht müssen. In diesem Fall spricht man von einem Template mit Wildcards oder einfach von einem Template.*

Mit Hilfe dieser Templates können nun Mengen von Regeln bestimmt werden. Dazu kann zunächst ein positives Template angegeben werden. Das System zeigt dann alle Regeln, die von diesem Template erfaßt werden, an. Zusätzlich kann ein negatives Template angegeben werden. Die Regeln, die von diesem Template erfaßt werden, werden dann wieder von der Anzeige ausgeschlossen.

## 3.4 Fuzzy Sets

Im letzten Kapitel waren in Abschnitt 3.2.20 neben Kategorisierungen und überlappenden Einteilungen auch Methoden erwähnt worden, die mit anderen Mengenkonzepten arbeiten als die klassische Mengenlehre. Im folgenden soll als Beispiel einer solchen Methode eine kurze Einführung in die Ideen der Theorie der Fuzzy Sets (unscharfen Mengen) gegeben werden. Fuzzy Sets wurden 1965 von L. A. Zadeh (1965) eingeführt. Die folgende Darstellung orientiert sich an Bademer & Gottwald (1993).

Die Theorie der Fuzzy Sets stellt als eine Verallgemeinerung des Mengenkonzepts den “Grad der Mitgliedschaft in einer Menge” zur Verfügung. Für ein Element muß also nicht mehr eindeutig gesagt oder bestimmt werden, ob es in einer bestimmten Menge liegt oder nicht, sondern es kann ein Zahlenwert angegeben werden, der die Zugehörigkeit des Elements zu der Menge beschreibt. Diese Verallgemeinerung wird mit Hilfe einer Zugehörigkeitsfunktion definiert:

***Definition 3.4.1 Fuzzy Set*** *Sei $D$ eine Menge und*

$$m_X : D \rightarrow [0, 1] \qquad (80)$$

*eine Funktion auf dieser Menge. Dann heißt $m_X$ eine* ***Zugehörigkeitsfunktion*** (membership function) *der* ***unscharfen Menge*** *(*fuzzy set*) $X$ über dem Grundbereich $D$. Der* ***Grad der Zugehörigkeit*** *eines Elementes $d \in D$ zur Menge $X$ wird durch den Funktionswert*

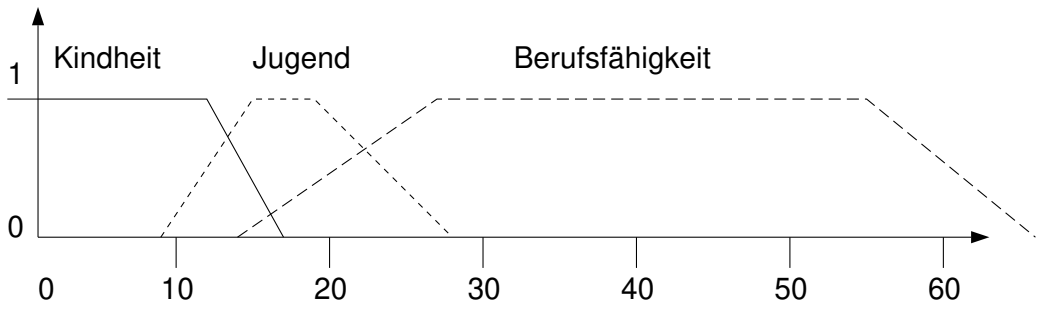
$$m_X(d) \in [0, 1] \qquad (81)$$

*gegeben. Um den Unterschied deutlich zu machen, werden herkömmliche Mengen in diesem Zusammenhang häufig auch als* ***scharfe Mengen*** *bezeichnet.*

Die herkömmliche Mengendefinition (oder die Definition einer scharfen Menge) läßt sich als Spezialfall der Definition einer unscharfen Menge schreiben. Dazu wählt man als Zugehörigkeitsfunktion einer Menge $U \subset D$ die charakteristische Funktion:

$$m_U(d) = \mathbf{1_U}(d) = \begin{cases} 1 & falls\ d \in U \\ 0 & sonst \end{cases} \tag{82}$$

**Abbildung 43: Beispiele von Fuzzy Sets als Beschreibung von Lebensaltern**




Zwei unscharfe Mengen sind genau dann **gleich**, wenn ihre Zugehörigkeitsfunktionen gleich sind. Die leere unscharfe Menge ist durch die Zugehörigkeitsfunktion

$$m_\emptyset \equiv 0 \tag{83}$$

gegeben, die Grundmenge $D$ durch

$$m_D \equiv 1 \tag{84}$$

Weiter können folgende Begriffe definiert werden:

***Definition 3.4.2 Träger, Kern, Schnitte und Höhe*** *Als **Träger** (*support*) einer unscharfen Menge $X$ über einer Grundmenge $D$ bezeichnet man die (scharfe) Teilmenge von $D$, auf der die Zugehörigkeitsfunktion ungleich $0$ ist:*

$$supp(X) = \{d \in D \mid m_X(d) > 0\} \tag{85}$$

*Als $\alpha$**–Schnitt** einer unscharfen Menge $X$ bezeichnet man die (scharfe) Teilmenge von $D$, auf der die Zugehörigkeitsfunktion größer als $\alpha$ ist:*

$$X^{>\alpha} = \{d \in D \mid m_X(d) > \alpha\} \tag{86}$$

*Als **scharfen** $\alpha$**–Schnitt** bezeichnet man die (scharfe) Teilmenge von $D$, auf der die Zugehörigkeitsfunktion größer oder gleich $\alpha$ ist:*

$$X^{\geq\alpha} = \{d \in D \mid m_X(d) \geq \alpha\} \tag{87}$$

*Der scharfe $1$–Schnitt heißt auch **Kern** der Menge $X$*

*Als **Höhe** einer unscharfen Menge bezeichnet man das Supremum der Zugehörigkeitsfunktion*

$$hgt(X) = \sup_{d \in D} \{m_X(d)\} \tag{88}$$

Zur weiteren Untersuchung der Eigenschaften unscharfer Mengen, bzw. zur weiteren Verallgemeinerung von Eigenschaften scharfer Mengen auf unscharfe Mengen kann man die folgende Aussage benutzen, die eine Beziehung zwischen scharfen und unscharfen Mengen herstellt.

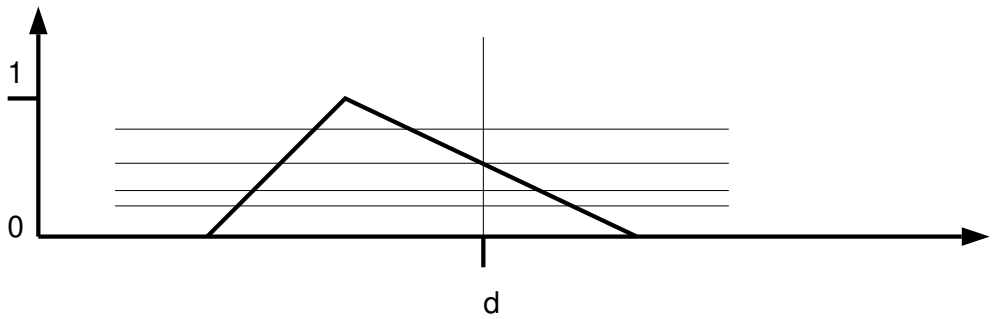
### 3.4.3 Aussage:

*Eine unscharfe Menge ist eindeutig durch ihre $\alpha$–Schnitte bestimmt. D. h. wenn für jedes $\alpha \in [0,1]$ die Menge $X^{>\alpha}$ bekannt ist, läßt sich daraus die Zugehörigkeitsfunktion rekonstruieren:*

$$m_X(d) = \sup_{\alpha \in [0,1]} \alpha \cdot m_{X^{>\alpha}}(d) \tag{89}$$

Das läßt sich folgendermaßen einsehen: $m_{X^{>\alpha}}$ ist die charakteristische Funktion des $\alpha$–Schnitts. Sie ist also solange gleich $1$, wie $d$ in $X^{>\alpha}$ liegt. Das ist aber genau dann der Fall, wenn $m_X(d) \leq \alpha$ gilt. Der größte Wert in dem Supremum ergibt sich also gerade für $\alpha = m_X(d)$

**Abbildung 44: Rekonstruktion des Wertes der Zugehörigkeitsfunktion aus einem $\alpha$–Schnitt.**




Mit dieser Äquivalenz lassen sich Eigenschaften und Operationen von scharfen Mengen auf unscharfe Mengen übertragen. Geht man von der Teilmengenrelation für scharfe Mengen aus, kann man fordern, daß die $\alpha$–Schnitte einer Teilmengenrelation unscharfer Mengen die Teilmengenrelation für scharfe Schnitte erfüllen sollen:

$$Y \subset X \Leftrightarrow Y^{\geq \alpha} \subset X^{\geq \alpha} \; \forall \alpha \in [0,1] \tag{90}$$

Aus dieser Forderung ergeben sich die Folgerungen:
Sei $d \in D$ mit $m_Y(d) \neq m_X(d)$. Setze $\alpha = m_Y(d)$.
Dann gilt $d \in Y^{\geq \alpha} \Rightarrow d \in X^{\geq \alpha} \Rightarrow m_X(d) \geq \alpha$ also $m_X(d) \geq m_Y(d) \; \forall d \in D$.

Zusammen gilt

$$Y^{\geq\alpha} \subset X^{\geq\alpha}\ \forall\alpha \in [0,1] \Leftrightarrow m_Y(d) \leq m_X(d)\ \forall d \in D \tag{91}$$

Man kann also für unscharfe Mengen die Teilmengenrelation durch diese Äquivalenz definieren.
Es folgt dann unmittelbar für diese Teilmengenrelation:

$$\emptyset \subset X \subset D \tag{92}$$

$$Y \subset X,\ X \subset Y \Rightarrow Y = X \tag{93}$$

sowie

$$Y \subset X,\ X \subset W\ \Rightarrow\ Y \subset W \tag{94}$$

Wie bei den scharfen Mengen gibt es auch bei unscharfen Mengen verschiedene sinnvolle Maße für die Größe einer Menge, je nachdem ob der Träger endlich, abzählbar oder meßbar ist. Bei endlichem Träger kann man analog zur Anzahl der Elemente einer scharfen Menge die Summe der Werte der Zugehörigkeitsfunktion verwenden.

$$\sum_{d \in supp(X)} m_X(d) \tag{95}$$

Bei abzählbarem Träger kann diese Summe (anders als bei der scharfen Menge) endlich sein oder nicht. Bei meßbarem, aber nicht endlichem Träger kann man das Integral über die Zugehörigkeitsfunktion als Maß für die Größe der Menge verwenden.

$$\int_{supp(X)} m_X(x)dx \tag{96}$$

Interessant sind dabei vor allem relative Größenangaben, also Angaben darüber, welchen Anteil der Größe der Grundmenge eine Teilmenge einnimmt. Diese relativen Größenangaben können für die verschiedenen Größenmaße berechnet werden.

### *Definition 3.4.4 Vereinigung, Durchschnitt und Komplement*

*Seien $X$ und $Y$ unscharfe Mengen über einer Grundmenge $D$.*

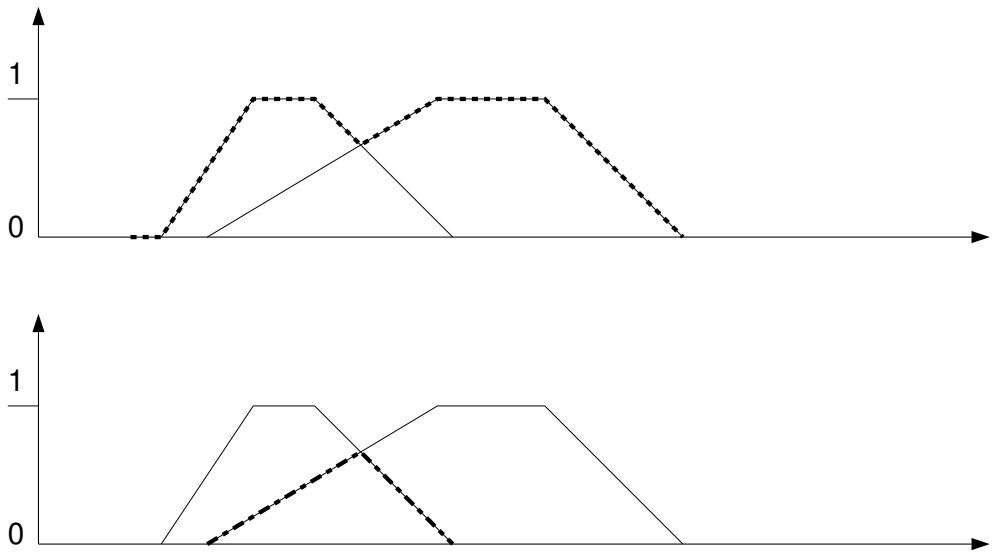
*Die **Vereinigung** zweier unscharfer Mengen ist durch das punktweise Maximum ihrer Zugehörigkeitsfunktionen bestimmt:*

$$X \cup Y := \max\{m_X(d), m_Y(d)\}\ \forall d \in D \tag{97}$$

*Der **Durchschnitt** zweier unscharfer Mengen ist als punktweises Minimum ihrer Zugehörigkeitsfunktionen definiert:*

$$X \cap Y := \min\{m_X(d), m_Y(d)\}\ \forall d \in D \tag{98}$$

**Abbildung 45: Vereinigung und Durchschnitt von unscharfen Mengen**




*Die Komplementbildung kann in der Form*

$$X^c := D \setminus X := m_{X^c} = 1 - m_X(d)\ \forall d \in D \qquad \textit{(99)}$$

*definiert werden.*

Man sieht, daß sich diese Definitionen bei der Verwendung von scharfen Mengen, also von charakteristischen Funktionen, mit den herkömmlichen Definitionen von Vereinigung, Durchschnitt und Komplement decken.

Aus den entsprechenden Eigenschaften der Maximumsfunktion ergeben sich auch sofort

- die Kommutativität:

$$Y \cup X = X \cup Y \qquad (100)$$

- die Assoziativität:

$$(Y \cup X) \cup W = X \cup (Y \cup W) \qquad (101)$$

- die Idempotenz:

$$X \cup X = X \qquad (102)$$

- und die Monotonie:

$$Y \subset X \Rightarrow Y \cup W \subset X \cup W \qquad (103)$$

Diese Eigenschaften lassen sich auch für den Durchschnitt zeigen. Entsprechend lassen sich auch Distributivgesetzte und deMorgansche Regeln auf unscharfe Mengen übertragen. Schließlich gilt

$$(Y \cup X)^{>\alpha} = Y^{>\alpha} \cup X^{>\alpha} \tag{104}$$

und

$$(Y \cap X)^{>\alpha} = Y^{>\alpha} \cap X^{>\alpha} \tag{105}$$

Neben dieser "kanonischen" Definition von Vereinigung und Durchschnittsbildung gibt es noch weitere Möglichkeiten, solche Operationen zu definieren. Darauf soll hier aber nicht eingegangen werden. Es soll im folgenden aber noch angedeutet werden, wie Fuzzy Methoden zum unscharfen Schließen bzw. unscharfen Kategorisieren verwendet werden können.

Dazu definieren wir unscharfe **Relationen** analog zu scharfen Relationen als Teilmengen der entsprechenden kartesischen Produkte:

Eine (scharfe) Relation zwischen zwei Attributen $A_1 : D \to R_1$ und $A_2 : D \to R_2$ kann als Teilmenge des kartesischen Produkts der Wertemenge der Attribute geschrieben werden.

$$T \subset R_1 \times R_2 = \{(x, y) \mid x \in R_1, y \in R_2\} \tag{106}$$

Die Gleichheit zweier Attribute könnte als Relation z. B. in der Form

$$T = \{(x, y) \mid x = y\} \tag{107}$$

geschrieben werden.

Kategorisierungsregeln, wie sie z. B. beim AQ-Algorithmus beschrieben wurden, können auch als Relationen dargestellt werden. Im folgenden beschränken wir uns der Einfachheit halber auf Kategorisierungen, die durch ein Attribut $A_2$ beschrieben sind, und sich nur auf ein Attribut $A_1$ stützen. Wie beschrieben, läßt sich ja jede Kategorisierung durch die geeignete Wahl bzw. Konstruktion von $A_1$ auf diese Aufgabe reduzieren. Sie ist dann eine Funktion, die sich als Regel in der Form

$$\texttt{If } A_1 = x \texttt{ then } A_2 = y$$

schreiben läßt und als Relation in der Form

$$\{(x, y) \mid A_1 = x, A_2 = y\} \tag{108}$$

Die charakteristische Funktion der Gleichheitsrelation auf der Menge der Paare ergibt sich z. B. für $R_1 = R_2 = \{0, 1, 2\}$ in der Form:

$$T = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{109}$$

Schreibt man ein Element aus $R$ auch als seine charakteristische Funktion, also z. B. $1 \in R$ als $x = (0, 1, 0)^t$, so ergibt sich die Kategorisierung als Multiplikation der Matrix $T$ mit dem Vektor $x$. Man kann sie in der Form

$$\texttt{If } A_1 = x \texttt{ then } A_2 = T \cdot x$$

schreiben.

In dieser Weise werden Kategorisierungen beschrieben, solange die Matrix eine Permutationsmatrix ist, d. h. solange in jeder Zeile und in jeder Spalte genau eine $1$ steht. Allgemeiner kann man die IF-THEN Regel als eine Verknüpfung der charakteristischen Funktion des Wertes von $A_1$ mit der Relation betrachten.

Die Darstellung der Relation durch die charakteristische Funktion ermöglicht die Verallgemeinerung der Relation zu einer unscharfen Relation und damit die Definition einer unscharfen Kategorisierung. Während bei der scharfen Relation nur $0$ und $1$ als Beschreibungen der Zugehörigkeit zulässig sind, sind bei der unscharfen Relation Zugehörigkeiten aus dem Intervall $[0, 1]$ erlaubt, z. B:

$$m_T = \begin{pmatrix} 1 & 0.3 & 0.1 \\ 0 & 1 & 0.3 \\ 0 & 0 & 1 \end{pmatrix} \tag{110}$$

Eine unscharfe IF-THEN-Regel kann nun mit Hilfe einer unscharfen Relation folgendermaßen geschrieben werden:

`If` $A_1 = X$ `then` $A_2 = Y$

wobei $X$ und $Y$ unscharfe Mengen sind und die Zugehörigkeitsfunktion von $Y$ folgendermaßen definiert ist:

$$m_Y(d) = \sup_{x \in D} \{\min(m_X(x), m_T(x, d))\} \tag{111}$$

Diese Definition lehnt sich an das Produkt zwischen dem Vektor und der Matrix an. Die Multiplikation wurde dabei durch das Minimum ersetzt und die Addition durch die Supremumsbildung.

Das Ergebnis dieser unscharfen Kategorisierung ist eine unscharfe Menge, also eine Zugehörigkeitsfunktion.

## Literaturverzeichnis


Bademer, H., & Gottwald, S. (1993). *Einführung in Fuzzy-Methoden*, 4. überarbeitete Auflage ed. Akademie Verlag: Berlin.

Carter, C., & Catlett, J. (1987). Assessing credit caord applications using machine learning. *IEEE Expert Fall 1987*, 71-79.

Fuhr, N. (1995). Information retrieval. Skriptum zur Vorlesung im SS 93, Universität Dortmund, http://ls6–www.informatik.uni-dortmund.de/teaching/courses/ir.

Han, J., Cai, Y., & Cercone, N. (1992). Knowledge discovery in databases: An attribute-oriented approach. In *Proceedings of the 18th VLDB Conference* (1992), 547–558.

Holsheimer, M., & Siebes, A. P. J. M. (1994). Data mining: the search for knowledge in databases. Technical report, Centrum voor Wiskunde en Informatica, ftp://ftp.cwi.nl/pub/CWIreports/AA/CS-R9406.ps.Z.

Klemettinen, M., Mannila, H., Ronkainen, P., Toivonen, H., & Verkamo, I. (1994). Finding interesting rules from large sets of discovered association rules. In *Third International Copnference on Information and Knowledge Management* (1994), N. R. Adam, B. K. Bhargava, & Y. Yesh, Eds., 401–407.

Michalski, R. S. *Machine Learning*. Los Altos, CA: Morgan Kaufmann Publishers, 1983, ch. 4: A Theory and Methodology of Inductive Learning, 83-133.

Michalski, R. S., Mozetic, I., Hong, J., & Lavrac, N. (1986). The multi-purpose incremental learning system aq15 and its testing application to three medical domains. In *Proceedings of the 5th national conference on Artificial Intellignece* (Philadelphia, 1986), 1041-1045.

Piatetsky-Shapiro, G. Knowledge discovery mine. WWW-HTML Page: http://info.gte.com/~kdd/index.html.

Zadeh, L. A. (1965). Fuzzy sets. *Information and Control 8*, 338–353.

# Teil 4 Knowledge Discovery und IR Systeme

Im letzten Teil der Vorlesung sollen Zusammenhänge zwischen Knowledge Discovery und Information Retrieval untersucht werden. Diese beiden Gebiete haben sich jeweils eher isoliert entwickelt und wenig von den Methoden und Ergebnissen des jeweils anderen zur Kenntnis genommen. Nimmt man aber die Definition von KDD, wie sie in Abschnitt 1.5 definiert worden war, so gibt es diverse Methoden und Verfahren im Information Retrieval, die unter diese Definition fallen, auch wenn sie selten als KDD Verfahren bezeichnet werden. Im folgenden sollen solche Ansätze vorgestellt werden. Dazu beginnen wir zunächst mit der Vorstellung verschiedener theoretischer Modelle des Information Retrieval, in deren Rahmen dann die Extraktionsverfahren eingebettet werden können.

## 4.1 Modelle des Information Retrieval

In Teil 2 war Information Retrieval als ein System aus Mengen, Transformationen und Attributen dargestellt worden. Dabei wurden Verfahren, mit denen bestimmte Teilmengen der Dokumentenmenge bestimmt werden können, vorgestellt. Es wurde aber kein ausgearbeitetes Modell der Informationssuche konstruiert. Das soll zu Beginn dieses Teiles getan werden. Zunächst soll das probabilistische Modell vorgestellt werden, wie es z. B. in Fuhr (1995) zu finden ist.

### 4.1.1 Probabilistisches Modell

In der Einleitung war darauf hingewiesen worden, daß Information Retrieval im Gegensatz zu Faktenretrieval immer nur nach relativ besten Lösungen oder Antworten suchen kann und im allgemeinen keine eindeutige beste Lösung existiert. Daher liegt es nahe, IR mit einem probabilistischen Ansatz zu modellieren.

Die diesem Ansatz zugrundeliegende Frage kann folgendermaßen formuliert werden:

> *Wie groß ist die Wahrscheinlichkeit, daß ein Dokument $d$ für eine Anfrage $q$ als relevant eingeschätzt wird.*

Hätte man diese Wahrscheinlichkeit für jede Anfrage und für alle Dokumente, könnte man alle Dokumente danach sortieren und die mit der größten Wahrscheinlichkeit ausgeben. Da diese Wahrscheinlichkeit aber nicht unmittelbar zugänglich ist, kann man versuchen, sie zu schätzen. Dazu müssen die Dokumente und die Anfragen weiter untersucht werden und entsprechende Vereinfachungen und Unabhängigkeitsannahmen gemacht werden.

Dazu zunächst ein paar Begriffe aus der Wahrscheinlichkeitsrechnung. Wir beschränken uns auf endliche Wahrscheinlichkeitsräume.

***Definition 4.1.1.1 Wahrscheinlichkeitsraum*** *Sei $G$ eine endliche Menge und $\mathfrak{P}(G)$ die Potenzmenge, d. h. die Menge der Teilmengen von $G$. Eine Funktion*

$$p : \mathfrak{P}(G) \to [0, 1] \tag{112}$$

*heißt **Wahrscheinlichkeitsfunktion**, wenn die beiden folgenden Bedingungen erfüllt sind:*

$$p(G) = 1 \tag{113}$$

$$p(X_1 \cup ... \cup X_m) = p(X_1) + ... + p(X_m) \tag{114}$$

*für alle paarweise disjunkten Teilmengen $X_1, ..., X_m \subset G$*

*$G$ heißt **Grundraum** und jede Teilmenge $X \subset G$ ein **Ereignis**. Die einelementigen Teilmengen heißen **elementare Ereignisse** oder **Elementarereignisse**. Die Menge $\bar{X} = X^c = G \setminus X$ heißt **komplementäres Ereignis** von $X$*

Aus der Definition folgt unmittelbar

$$p(\emptyset) = 0 \tag{115}$$

denn $1 = p(G) = p(G \cup \emptyset) = p(G) + p(\emptyset) = 1 + p(\emptyset)$

### 4.1.1.2 Beispiel: Würfel

Betrachten wir als Grundraum die Menge $G = \{1, ..., 6\}$ der Augenzahlen eines Würfels. Ein Würfel ist fair, wenn jedes elementare Ereignis, also das Fallen einer Zahl, die Wahrscheinlichkeit $p(\{x\}) = \frac{1}{6}$ hat. Das Ereignis $X = \{1, 3, 5\} \subset G$ ist dadurch definiert, daß eine der Augenzahlen der Teilmenge gewürfelt wird, d. h. daß eine ungerade Zahl fällt. Die Wahrscheinlichkeit des Ereignisses ergibt sich aus $p(X) = p(\{1, 3, 5\}) = p(\{1\} \cup \{3\} \cup \{5\}) = p(\{1\}) + p(\{3\}) + p(\{5\}) = \frac{3}{6} = \frac{1}{2}$

Man sieht unmittelbar, daß sich dieser Sachverhalt auch mit Attributen und Bedingungen schreiben läßt: Als Attribut $A_1 : G \to G$ wählt man die Augenzahl. Dann ist $X = (A_1 \in \{1, 3, 5\})$ die Menge der Würfe mit ungerader Augenzahl. Die Wahrscheinlichkeit kann dann auch so geschrieben werden: $p(X) = p((A_1 \in \{1, 3, 5\})) = \frac{1}{2}$. Das Attribut $A_1$ wird dann auch als **Zufallsvariable** bezeichnet, und in der Formel wird die innere Klammer häufig weggelassen.

Würde man nicht nur mit einem Würfel, sondern mit zweien, einem roten und einem blauen, würfeln, wäre der Grundraum entsprechend komplizierter. Er bestünde aus allen Paaren aus den Zahlen eins bis sechs: $G = \{1, ..., 6\}^2 = \{(x, y) \mid x, y \in \{1, ..., 6\}\}$. Bei fairen Würfeln sollte jedes Paar – also jedes Elementarereignis – gleich wahrscheinlich sein. Bei 36 Paaren ergibt sich also eine Wahrscheinlichkeit von $\frac{1}{36}$

Wählt man als Attribut oder Zufallsvariable $A_r : G \to \{1, ..., 6\}$ die Augenzahl des roten Würfels und als $A_b : G \to \{1, ..., 6\}$ die des blauen, ergibt sich z. B. als Wahrscheinlichkeit für das Ereignis "der rote Würfel zeigt eine Eins" mit $p(A_r = 1) = p\left(\bigcup_{i=1}^{6} ((A_r = 1) \wedge (A_b = i))\right) = \sum_{i=1}^{6} p((A_r = 1) \wedge (A_b = i)) = \frac{6}{36} = \frac{1}{6}$, wie erwartet die Wahrscheinlichkeit, mit einem Würfel eine Eins zu werfen. In diesem Fall verteilen sich die Wahrscheinlichkeiten der Augenzahlen des blauen Würfels gleichmäßig auf die Paare mit einer "roten Eins". Die Wahrscheinlichkeit jedes Paares ergibt sich so als ein Sechstel der Wahrscheinlichkeit der "roten Eins" also als $\frac{1}{6} \cdot \frac{1}{6} = \frac{1}{36}$. Umgekehrt ist

die Wahrscheinlichkeit jedes einzelnen Paares unter den Paaren mit einer “roten Eins” gerade $\frac{1}{6}$.
Diese Sachverhalte lassen sich folgendermaßen verallgemeinern:

***Definition 4.1.1.3 Bedingte Wahrscheinlichkeit und Unabhängigkeit***

*Seien $X, Y \subset G$ Ereignisse eines Wahrscheinlichkeitsraumes mit der Wahrscheinlichkeitsfunktion $p$. Dann heißt der Ausdruck*

$$p(X \mid Y) = \frac{p(X \cap Y)}{p(Y)} \tag{116}$$

***bedingte Wahrscheinlichkeit** von $X$ unter der **Bedingung** $Y$.*

*Die Ereignisse heißen **unabhängig**, wenn*

$$p(X \cap Y) = p(X) \cdot p(Y) \tag{117}$$

*gilt.*

*Zwei Zufallsvariablen $A_1 : G \to R_1$ und $A_2 : G \to R_2$ heißen unabhängig, wenn für beliebige Mengen $X_1 \subset R_1$ und $X_2 \subset R_2$ gilt*

$$p((A_1 \in X_1) \cap (A_2 \in X_2)) = p((A_1 \in X_1)) \cdot p((A_2 \in X_2)) \tag{118}$$

Für unabhängige Ergeignisse $X, Y \subset G$ gilt:

$$p(X \mid Y) = \frac{p(X \cap Y)}{p(Y)} = \frac{p(X) \cdot p(Y)}{p(Y)} = p(X) \tag{119}$$

Die bedingte Wahrscheinlichkeit ist also gerade gleich der normalen Wahrscheinlichkeit. D. h. die Wahrscheinlichkeit von $X$ wird durch das Auftreten des Ereignisses $Y$ nicht beeinflußt.
Weiter ergibt sich aus der Definiton der bedingten Wahrscheinlichkeit

$$\begin{aligned} p(X \mid Y) \cdot p(Y) &= p(X \cap Y) \\ p(Y \mid X) \cdot p(X) &= p(Y \cap X) \end{aligned} \tag{120}$$

und damit

$$p(X \mid Y) \cdot p(Y) = p(Y \mid X) \cdot p(X) \tag{121}$$

bzw.

$$p(X \mid Y) = \frac{p(Y \mid X) \cdot p(X)}{p(Y)} \tag{122}$$

eine einfache Form der **Bayes’schen Formel**.
Die Produktformel für die Wahrscheinlichkeit unabhängiger Ereignisse und die Bayes’sche Formel werden im folgenden verwendet, um mit vereinfachenden Annahmen die gesuchte Wahrscheinlichkeit für die Beurteilung der Relevanz eines Dokuments herzuleiten. Sie kann jetzt als bedingte Wahrscheinlichkeit geschrieben werden. Dazu verwendet Fuhr (1995) das Ereignis $R$ “Ein Dokument wird als relevant eingeschätzt” und schreibt $P(R \mid q, d)$, also die bedingte Wahrscheinlichkeit dafür, daß eine Relevanz angegeben wird, unter der Bedingung, daß die Anfrage $q$ und das Dokument $d$ vorliegen.

Der zugrundeliegende Grundraum ist das kartesische Produkt $G = D \times Q \times \{R, NR\}$, wobei $Q$ die Menge aller Anfragen und $R$ bzw. $NR$ die Beurteilung als relevant bzw. nicht relevant bedeuten. Die Schreibweise der Bedingung mit einem Komma entspricht dem Durchschnitt der Mengen, die durch geeignet definierte Attribute mit diesen Werten beschrieben werden können. Sei $A_r : G \to \{R, NR\}$ das Attribut, das angibt, ob ein Element aus dem Grundraum als relevant beurteilt wurde oder nicht, und $A_q : G \to Q$ das Attribut, das die verwendete Anfrage beschreibt, dann ist die Bedingung $R, q = ((A_r = R) \cap (A_q = q))$, also die Menge der Elementarereignisse, bei denen ein Dokument für die Anfrage $q$ als relevant beurteilt wurde.

Um die bedingte Wahrscheinlichkeit $P(R \mid q, d)$ abschätzen zu können, geht man zunächst zu einfacheren Repräsentationen von Dokument und Anfrage über. Dazu sei $T = \{t_1, ..., t_N\}$ die Menge der Terme, die in einer Dokumentensammlung $D$ vorkommen. Eine Anfrage $q \subset T$ wird als die Menge von Termen dargestellt und ein Dokument $d$ ebenfalls als $d \subset T$ oder als Vektor von Attributen bzw. Zufallsvariablen $A_1, ..., A_N$ mit

$$A_i(d) = \begin{cases} 1 & \text{falls } t_i \in d \\ 0 & \text{sonst} \end{cases} \tag{123}$$

Die Vektorschreibweise besagt, daß die durch die Bedingung an die einzelnen Attribute beschriebenen Mengen konjunktiv verknüpft werden. Um ein einzelnes Dokument zu beschreiben, kann abkürzend auch nur der Vektor mit den Bedingungswerten angegeben werden. Dieses Dokument wird dann durch $\vec{d} = (d_1, ..., d_N) \in \{0, 1\}^N$ bezeichnet.

Eine weitere Darstellungsmöglichkeit für die Chance, daß ein Ereigniss auftritt, ist der Quotient der Wahrscheinlichkeit mit der Wahrscheinlichkeit des Komplementärereignisses, die **Quote** des Ereignisses (*Odds*).

$$O(X) = \frac{p(X)}{p(\bar{X})} = \frac{p(X)}{1 - p(X)} \tag{124}$$

Die Quote ist $< 1$ für Wahrscheinlichkeiten $< 0.5$ und $> 1$ für Wahrscheinlichkeiten $> 0.5$. Sie ist streng monoton, liefert also dieselbe Rangreihe für Ereignisse wie die Wahrscheinlichkeit. Sie erlaubt aber manchmal einfacheres Rechnen.

Statt der gesuchten beingten Wahrscheinlichkeit wird nun ihre Quote

$$O(R \mid q, d) = \frac{p(R \mid q, d)}{p(\bar{R} \mid q, d)} \tag{125}$$

angenähert. Dabei werden einige Annahmen gemacht werden, die i. a. in der Realität nicht haltbar sein dürften.

Zunächst gilt

$$\begin{aligned} P(R \mid q, d) &= \frac{p(R \cap q \cap d)}{p(q \cap d)} \\ &= \frac{p(d \mid R \cap q) \cdot p(R \cap q)}{p(d \mid q) \cdot p(q)} \\ &= \frac{p(d \mid R \cap q) \cdot p(R \mid q)}{p(d \mid q)} \end{aligned} \tag{126}$$

Es folgt also

$$O(R \mid q, d) = \frac{p(R \mid q, d)}{p(\bar{R} \mid q, d)} = \frac{\frac{p(d|R \cap q) \cdot p(R|q)}{p(d|q)}}{\frac{p(d|\bar{R} \cap q) \cdot p(\bar{R}|q)}{p(d|q)}} = \frac{p(R \mid q)}{p(\bar{R} \mid q)} \cdot \frac{p(d \mid R \cap q)}{p(d \mid \bar{R} \cap q)} \tag{127}$$

$p(d \mid R, q)$ ist die bedingte Wahrscheinlichkeit eines Dokuments $d$ unter der Bedingung, daß es zur Anfrage $q$ als relevant beurteilt wird.

Um diese Wahrscheinlichkeit zu schätzen, müßten zu allen Anfragen Dokumente mit Relevanzbeurteilungen vorliegen. Da dieser Aufwand selten zu leisten ist, wird die Berechnung im nächsten Schritt auf die Terme heruntergebrochen. Dazu wird die sehr starke (und unrealistische) Annahme gemacht, daß das Auftreten von Termen in Dokumenten unabhängig ist. So ergibt sich (mit $I = \{1, ..., N\}$)

$$\begin{aligned} p(d \mid R, q) = p\Big(\vec{d} \mid R, q\Big) &= p\left(\bigwedge_{i \in I} (A_i = d_i) \mid R, q\right) \\ &= \prod_{i \in I} p((A_i = d_i) \mid R, q) = \prod_{i \in I} p(d_i \mid R, q) \end{aligned} \tag{128}$$

und für die Quote

$$O(R \mid q, d) = \frac{p(R \mid q)}{p\big(\bar{R} \mid q\big)} \cdot \frac{p(d \mid R, q)}{p\big(d \mid \bar{R}, q\big)} = O(R \mid q) \cdot \prod_{i \in I} \frac{p((A_i = d_i) \mid R, q)}{p\big((A_i = d_i) \mid \bar{R}, q\big)} \tag{129}$$

Man kann die Annahme etwas abschwächen, indem man direkt annimmt, daß

$$\frac{p(d \mid R, q)}{p\big(d \mid \bar{R}, q\big)} = \prod_{i \in I} \frac{p((A_i = d_i) \mid R, q)}{p\big((A_i = d_i) \mid \bar{R}, q\big)} \tag{130}$$

(die sog. *linked dependency assumption*) gilt. Allerdings ist diese Annahme wesentlich schwieriger zu interpretieren.

Zur weiteren Vereinfachung wird angenommen, daß für alle $t_i \in T \setminus q$ gilt, daß $p(A_i = d_i \mid R, q) = p\big(A_i = d_i \mid \bar{R}, q\big)$ ist.

D. h. man nimmt an, daß für alle Terme, die nicht in der Anfrage genannt werden, die Wahrscheinlichkeit, daß sie in einem relevanten Dokument auftreten, genauso groß ist wie die, daß sie in einem nicht relevanten Dokument auftreten. Um diese vereinfachende Annahme auszunutzen, spaltet man das Produkt auf:

$$\begin{aligned} O(R \mid q, d) = O(R \mid q) \cdot & \prod_{\{i \in I \mid t_i \in q \cap d\}} \frac{p((A_i = d_i) \mid R, q)}{p\big((A_i = d_i) \mid \bar{R}, q\big)} \\ \cdot & \prod_{\{i \in I \mid t_i \in q \setminus d\}} \frac{p((A_i = d_i) \mid R, q)}{p\big((A_i = d_i) \mid \bar{R}, q\big)} \\ \cdot & \prod_{\{i \in I \mid t_i \notin q\}} \frac{p((A_i = d_i) \mid R, q)}{p\big((A_i = d_i) \mid \bar{R}, q\big)} \end{aligned} \tag{131}$$

Der letzte Faktor dieses Produkts ist aufgrund der Vereinfachung gleich Eins, kann also weggelassen werden. Setzt man $r_i = p(A_i = 1 \mid R, q)$ und $n_i = p\big(A_i = 1 \mid \bar{R}, q\big)$, so kann man schreiben

$$O(R \mid q, d) = O(R \mid q) \cdot \prod_{\{i \in I \mid t_i \in q \cap d\}} \frac{r_i}{n_i} \cdot \prod_{\{i \in I \mid t_i \in q \setminus d\}} \frac{1 - r_i}{1 - n_i} \tag{132}$$

Da es meistens darum geht, verschiedene Dokumente in eine Rangreihe bezüglich einer Anfrage zu bringen, interessieren vor allem solche Faktoren des Produktes, die

sich bei verschiedenen Dokumenten ändern. Um diese zu isolieren, kann man folgende Umformung vornehmen

$$O(R \mid q,d) = O(R \mid q) \cdot \prod_{\{i \in I \mid t_i \in q \cap d\}} \frac{r_i}{n_i} \cdot \prod_{\{i \in I \mid t_i \in q \setminus d\}} \frac{1-r_i}{1-n_i} \cdot \prod_{\{i \in I \mid t_i \in q \cap d\}} \frac{(1-r_i)(1-n_i)}{(1-n_i)(1-r_i)} \tag{133}$$

$$O(R \mid q,d) = O(R \mid q) \cdot \prod_{\{i \in I \mid t_i \in q \cap d\}} \frac{r_i(1-n_i)}{n_i(1-r_i)} \cdot \prod_{\{i \in I \mid t_i \in q\}} \frac{1-r_i}{1-n_i} \tag{134}$$

Berechnet man die Werte für mehrere Dokumente $d$, so ist lediglich der mittlere Faktor dieses Produkts noch von den verwendeten Dokumenten abhängig, also für die Bildung einer Rangreihe relevant. Zur einfacheren Berechnung kann man auf diesen Faktor noch einen Logarithmus anwenden, der als streng monotone Funktion ja die Rangreihe nicht verändert. So erhält man als Kenngröße eines Dokuments $d$ den **Retrievalstatuswert** (*retrieval status value*):

$$\sum_{\{i \in I \mid t_i \in q \cap d\}} \log \frac{r_i(1-n_i)}{n_i(1-r_i)} = \sum_{\{i \in I \mid t_i \in q \cap d\}} \left( \log \frac{r_i}{n_i} + \log \frac{(1-n_i)}{(1-r_i)} \right) \tag{135}$$

Um das Verfahren anzuwenden, müssen Werte für $r_i$ und $n_i$ geschätzt werden. $r_i$ ist die Wahrscheinlichkeit, daß der Term $t_i$ in einem für die Anfrage $q$ relevanten Dokument vorkommt, und $n_i$ ist die Wahrscheinlichkeit, daß der Term $t_i$ in einem für die Anfrage $q$ nicht relevanten Dokument vorkommt.

Um die Werte zu schätzen, kann eine Menge von Dokumenten verwendet werden, für die bekannt ist, ob sie für eine Anfrage relevant sind oder nicht. Diese Information kann z. B. durch **Relevance Feedback** Methoden ermittelt werden: Nutzenden werden eine Reihe von Dokumenten gezeigt, und sie geben an, welche sie für ihren Informationsbedarf für relevant halten. Als Schätzung für $r_i$ kann man die Anzahl $rel_i$ der relevanten Dokumente, die den Term $t_i$ enthalten, durch die Gesamtzahl $rel$ der relevanten Dokumente teilen:

$$\widetilde{r_i} = \frac{rel_i}{rel} \tag{136}$$

Als Schätzung für $n_i$ kann man die Anzahl $nrel_i$ der Dokumente, die nicht als relevant eingeschätzt wurden und den Term $t_i$ enthalten durch die Gesamtzahl $nrel$ der nichtrelevanten Dokumente teilen

$$\widetilde{n_i} = \frac{nrel_i}{nrel} \tag{137}$$

Bei dieser Schätzung ergeben sich besonders bei kleinen Dokumentmengen Probleme, wenn eine der relativen Häufigkeiten Eins oder Null ist. In diesen Fällen ist entwerder der Bruch oder der Logarithmus nicht definiert. Ist das nicht der Fall, ergibt sich für den Retrievalstatus der Schätzwert:

$$\begin{aligned} &\sum_{\{i \in I \mid t_i \in q \cap d\}} \log \frac{\frac{rel_i}{rel}\left(1 - \frac{nrel_i}{nrel}\right)}{\frac{nrel_i}{nrel}\left(1 - \frac{rel_i}{rel}\right)} \\ &= \sum_{\{i \in I \mid t_i \in q \cap d\}} \log \frac{rel_i(nrel - nrel_i)}{nrel_i(rel - rel_i)} \\ &= \sum_{\{i \in I \mid t_i \in q \cap d\}} \left( \log \frac{rel_i}{(rel - rel_i)} - \log \frac{nrel_i}{(nrel - nrel_i)} \right) \end{aligned} \tag{138}$$

d. h. ein Term $t_i$ trägt positiv zu der Summe bei, wenn seine Quote, berechnet mit der relativen Häufigkeit, in relevanten Dokumenten größer ist als in nicht relevanten Dokumenten. Weiter folgt, daß ein neues Dokument, das viele Terme enthält, die überproportional häufig in relevanten Dokumenten aufgetreten sind, einen hohen Statuswert erhält.

**Abbildung 46: Beispiele mit Relevanzangaben zur Schätzung des Retrievalstatuswertes zu einer Anfrage $q = \{t1, ..., t6\}$**

| | t1 | t2 | t3 | t4 | t5 | t6 | R |
|---|---|---|---|---|---|---|---|
| d1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 |
| d2 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| d3 | 0 | 0 | 1 | 0 | 1 | 1 | 1 |
| d4 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| d5 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| d6 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| d7 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
| d8 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| rel_i | 2 | 1 | 3 | 3 | 3 | 2 | rel=4 |
| nrel_i | 1 | 1 | 2 | 1 | 2 | 1 | nrel=4 |
| exp(sv_i) | 3 | 1 | 3 | 9 | 3 | 3 | |

*Die Tabelle zeigt das Auftreten der Terme $t1, ..., t6$ in den Dokumenten $d1, ..., d8$ und unter R die Einschätzung ihrer Relevanz zu der Anfrage $q$. exp(sv_i) ist der Wert des Quotienten, dessen Logarithmus in die Summe eingeht.*

**Abbildung 47: Neue Dokumente und ihr Retrievalstatuswert**

| | t1 | t2 | t3 | t4 | t5 | t6 | Retrievalstatus |
|---|---|---|---|---|---|---|---|
| d9 | 1 | 1 | 0 | 0 | 0 | 1 | log(3) +log(1) +log(3) =.95 |
| d10 | 0 | 1 | 0 | 1 | 0 | 0 | log(1) +log(9) =.95 |
| d11 | 1 | 0 | 1 | 1 | 1 | 1 | 4* log(3) +log(9) =2.86 |
| d12 | 0 | 0 | 1 | 0 | 1 | 0 | log(3) +log(3) =.95 |

*Für neue Dokumente d9,...,d12 werden die Retrievelstatuswerte aufgrund der Dokumenten aus Abbildung 46 für die einzelnen Terme ermittelten Summanden berechnet.*

Bei diesem Verfahren muß zu jeder Anfrage zunächst eine Menge von Dokumenten auf ihre Relevanz bezüglich der Anfrage beurteilt werden, damit anschließend zusätzliche Dokumente bewertet werden können. Das ist immer noch ein aufwendiges Verfahren.

Eine Möglichkeit dieses Verfahren zu verallgemeinern besteht darin, über verschiedene Anfragen zu mitteln, d. h. solche Terme hoch zu gewichten, die in vielen Dokumenten viel zu der Summe des Statuswertes beitragen.

Eine andere Möglichkeit wurde im sog. **Darmstädter Indexierungs Ansatz** (*Darmstadt Indexing Approach*) DIA gewählt. Dort wurde zusätzlich zu den Attributen, die das Auftreten von Termen angeben, die Art und Weise, wie ein Term in einem Dokument auftritt, erhoben. Das kann z. B. die Anzahl des Auftretens im Dokument, oder die Stelle des Auftretens (Titel oder Abstract) oder eine Kombination solcher Angaben sein. Allgemein kann man das als eine **Relevanzbeschreibung** (*relevance description*) formulieren, ein Attribut $A_i : D \rightarrow R_i$, in das auch weitere Informationen über den Term und das Dokument einfließen können, wie die Länge des Dokuments oder die Häufigkeit des Terms in der Dokumentensammlung. Werden nur allgemeine Eigenschaften eines Terms, wie seine Häufigkeit in der Sammlung etc. verwendet, kann eine Relevanzbeschreibung auch als Funktion des Dokuments $d \in D$ und des Terms $t_i \in T$ aufgefaßt werden, also eine Abbildung $x : D \times T \rightarrow \bigcup_{i \in I} R_i$ mit $x(d, t_i) = A_i(d)$.

Man kann die verschiedenen Werte, die in der Relevanzbeschreibung verwendet werden, auch als separate Attribute auffassen, die dann z. B. die Häufigkeit, mit der ein Term im Dokument auftritt, oder den Ort an dem er auftritt, oder die Anzahl der Dokumente in der Sammlung in denen er auftritt, beschreiben. Sie können in unterschiedlichem Maße zum Retrievalstatus eines Dokuments beitragen. Nimmt man in einem weiteren Schritt an, daß diese Gewichtung des Einflußes nur von einigen globalen Größen des Terms abhängt, die aus der Dokumentensammlung gewonnen werden können, kann man den Einfluß der verallgemeinerten Attribute durch Machine Learning Verfahren bestimmen.

**Abbildung 48: Elemente einer Relevanzbeschreibung nach Fuhr (1995)**

| |
|---|
| Häufigkeit von $t_i$ in $d$ |
| Maximale Häufigkeit eines Terms in $d$ |
| Inverse Dokumenthäufigkeit von $t_i$ in der Sammlung. $(\#(d \in D \mid t_i \in d))^{-1}$ |
| Anzahl der Terme im Dokument $\log(\#d)$ |
| Auftreten von $t_i$ im Titel des Dokuments |

Neben der Verringerung der Anzahl der zu bearbeitenden Attribute erhöht sich die Anzahl der Trainingbeispiele, da jeder Term, der in einer Anfrage und einem für die Anfrage bewerteten Dokument vorkommt, ein Trainingsbeispiel erzeugt. Beispiele für Attribute sind in Abbildung 48 angegeben.

Im Rahmen des Darmstädter Ansatzes wurden verschiedene Verfahren zur Bestimmung einer Indexierungsfunktion verwendet, darunter:

- Der ID3 Algorithmus
- Ein Maximum Spanning Tree Ansatz, der paarweise Abhängigkeiten unter den Komponenten zuläßt
- Logistische Regression
- Eine Methode der kleinsten Fehlerquadrate auf einem Raum von Polynomen

Zusammenfassend kann man sagen, daß bei den besprochenen probabilistischen Verfahren Abhängigkeiten zwischen Dokumenten lediglich in Form der Inversen Dokumenthäufigkeit und durch die (kleinen) Beispielmengen mit Relevanzbeurteilungen in Spiel kommen.

Die meisten bisher besprochenen Retrievalverfahren stützen sich auf das gleichzeitige Auftreten von Termen in einer Anfrage und in einem Dokument. Dabei bleibt es

weitgehend den Anfragenden und ihrer Kenntnis über das Gebiet, aus dem die Anfrage stammt, überlassen, welche Terme sie verwenden. Ansätze, die die Wahl der Terme auf ein vorgegebenes Vokabular einschränken, sind in Abschnitt 2.2.3 über Thesauri beschrieben worden. Ein Thesaurus gibt – neben einem vorgeschriebenen Vokabular – auch Beziehungen zwischen den Termen des Vokabulars an, wie z. B, daß ein Term ein Oberbegriff eines anderen Terms ist. Die Konzepthierarchien, die im Abschnitt 3.3.2 über KDD Verfahren mit externem Wissen eingeführt wurden, bilden einen Teil der Beziehungen ab, die in einem Thesaurus dargestellt werden können. Sie waren als eine Methode, externes Wissen in einem KDD System zu nutzen, verwendet worden. Entsprechend können Thesauri für IR Systeme verwendet werden, um Anfragen zu verallgemeinern, indem man zu Oberbegriffen übergeht oder Synonyme hinzufügt. Dieses Vorgehen kann auch als Formel geschrieben werden:

```
If (x=PC) then (x=Computer)
```

```
If (x=Workstation) then (x=computer)
```

Allgemeiner kann man den gesamten Vorgang des Information Retrieval als eine Form des logischen Schließens oder der Inferenz betrachten. Diese Sichtweise liegt dem Modell von van Rijsbergen (1986) zugrunde:

## 4.1.2 Logische Modelle des Information Retrieval

Die Basis des logischen Modells des Information Retrievals ist die mit der künstlichen Intelligenz verwandte Annahme, daß Anfragen und Dokumente durch logische Formeln repräsentiert und durch einen Inferenzprozeß zueinander in Beziehung gesetzt werden können. Ein Dokument wird als zu einer Anfrage relevant angesehen, wenn das Retrieval System die Anfrage aus den Formeln und Aussagen, die das Dokument repräsentieren, ableiten kann. Dabei kann der Ableitungsprozeß prinzipiell komplizierter sein als lediglich die Überprüfung des Vorkommens von gleichen Termen in Dokument und Anfrage. Insbesondere kann zusätzliches Wissen in Form von logischen Formeln verwendet werden, das aus einer Wissensbasis des Systems stammt oder von den Nutzenden eingegeben bzw. abgefragt wird.

Ein solches System ist sehr mächtig. Daher ist es zunächst auch sehr anfällig für Inkonsistenzen in der Wissensbasis bzw. in den Anfragen und Dokumenten. Nun sind Dokumentensammlungen – wie schon öfter erwähnt – in aller Regel nicht konsistent, und auch bei Regeln, die von Nutzenden eingegeben werden, kann man i. a. nicht von Konsistenz ausgehen.

### 4.1.2.1 Imaging

Zur Lösung dieses Problems kann man eine probabilistische Inferenz einführen, indem man die Wahrscheinlichkeit $p(d \rightarrow q)$, daß eine Folgerung gilt – hier: daß die Anfrage $q$ aus dem Dokument $d$ gefolgert werden kann –, abschätzt. Dazu kann man zunächst jedes Dokument als eine "**mögliche Welt**" (*possible World*) betrachten, also als eine Menge von Aussagen mit zugehörigen Wahrheitswerten (und hoffen, daß diese Welt konsistent ist). Das Prinzip der unsicheren oder probabilistischen Inferenz ist nun folgendes: Es seien zwei Aussagen $x$ und $y$ gegeben. Als Maß der Unsicherheit der Aussage $y \rightarrow x$ bezüglich einer gegebenen Datenbasis kann der Umfang der kleinsten Informationsmenge gewählt werden, die zu der Datenbasis hinzugefügt werden muß, damit die Aussage $y \rightarrow x$ gültig ist.

Aus dieser eher vagen Beschreibung leitet van Rijsbergen schrittweise konkreter werdend einen Algorithums ab, mit dem Dokumente in eine Rangreihe bezüglich einer

Anfrage gebracht werden können. Dazu benötigt man eine Ähnlichkeitsfunktion zwischen den "möglichen Welten" $w \in W$ und eine Wahrscheinlichkeitsfunktion $p$ auf dem Grundraum $W$.

Im Laufe dieser Ableitung wird sich zeigen, daß sich das System dabei im wesentlichen auf ein gewichtetes Vektorraummodell reduziert, bei dem aus dem Korpus gewonnene Ähnlichkeiten zwischen Termen bei der Gewichtung berücksichtigt werden.

Seien nun wieder $x$ und $y$ zwei Aussagen und $w \in W$ eine "mögliche Welt". Mit

$$\sigma(w, y) \tag{139}$$

bezeichnen wir die Welt, die am ähnlichsten zu $w$ ist und in der $y$ gilt. Wir sagen dann, daß $y \rightarrow x$ gilt, wenn $x$ in $\sigma(w, y)$ gilt. Bezeichne

$$\tau(w, x) = \begin{cases} 1 & \text{falls } x \text{ in } w \text{ gilt} \\ 0 & \text{sonst} \end{cases} \tag{140}$$

die Indikatorfunktion dafür, daß eine Aussage in einer Welt gilt, dann folgt:

$$\tau(w, y \rightarrow x) = \tau(\sigma(w, y), x) \tag{141}$$

Die Wahrscheinlichkeit $p(y \rightarrow x)$ wird nun folgendermaßen definiert:

$$P(y \rightarrow x) = \sum_{w \in W} p(w)\tau(w, y \rightarrow x) = \sum_{w \in W} p(w)\tau(\sigma(w, y), x) \tag{142}$$

d. h. um die globale Wahrscheinlichkeit des Schlusses $y \rightarrow x$ zu bestimmen, wird zu jeder Welt $w$ die ähnlichste Welt gesucht, in der die Aussage $y$ gilt, und nachgesehen, ob dort auch $x$ gilt. Ist das der Fall, wird die Wahrscheinlichkeit der Welt $w$ zur globalen Wahrscheinlichkeit addiert. Der Übergang zur ähnlichsten Welt, in der $y$ gültig ist, wird **Imaging** genannt. Durch Imaging und den Mittelungsprozeß soll die Informationsmenge geschätzt werden, die im Mittel fehlt, damit $y \rightarrow x$ gültig ist.

Die Anwendung dieser Theorie mit Dokumenten als "mögliche Welten" ist wenig erfolgversprechend, da ja dann auch die Aussage $y$ ein Dokument ist. Nimmt man an, daß ein Dokument in einem anderen Dokument nur dann "gültig" ist, wenn es eine Teilmenge des letzteren ist, könnten beim Imaging nur solche Dokumente verwendet werden, bei denen das eine im anderen enthalten ist. Solche Dokumente sind aber vermutlich sehr selten.

Crestani und van Rijsbergen (1995) wählen deshalb eine Menge $T$ von Termen als "mögliche Welten". Die Wahrscheinlichkeit der Inferenz von einem Dokument $d$ auf eine Anfrage $q$ ergibt sich dann als

$$p(d \rightarrow q) = \sum_{t \in T} p(t)\tau(\sigma(d, t), q) \tag{143}$$

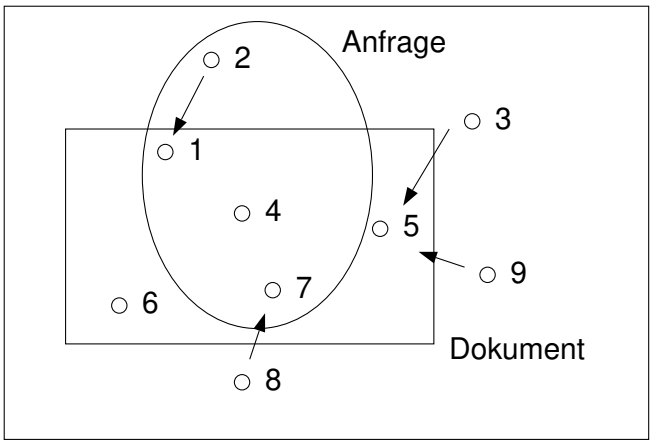
Inhaltlich kann ein Term im einfachsten Fall als durch die Menge der Dokumente repräsentiert angesehen werden, in denen er auftritt. D. h ein Dokument bzw. eine Anfrage ist in einem Term "gültig", wenn der Term darin auftaucht. Hat man zusätzlich eine Ähnlichkeitsfunktion zwischen Termen, so wird beim Imaging die Wahrscheinlichkeit eines Terms, der nicht in einem Dokument auftaucht, auf den nächstgelegenen Term aus dem Dokument übertragen.

Die globale Wahrscheinlichkeit $p(d \rightarrow q)$ für ein Dokument $d \subset T$ und eine Anfrage $q \subset T$ kann folgendermaßen berechnet werden:

- Zunächst wird für jeden Term $t \in T \setminus d$ der Term $t' \in d$ bestimmt, der am nächsten an $t$ liegt.
- Dann wird die Wahrscheinlichkeit $p(t)$ auf die von $t'$ addiert.
- Schließlich werden diese Summen über alle Terme aus der Anfrage $q$ aufaddiert.

Diese Summe wird als Wahrscheinlichkeit, daß sich die Anfrage $q$ aus dem Dokument $d$ folgern läßt, interpretiert. Die Dokumente werden in der Rangreihe, die sich aus diesen Werten ergibt, ausgegeben.

**Abbildung 49: Imaging**




*Die Punkte stellen die Terme $t_1, ..., t_9$ dar. Die Terme der Anfrage liegen innerhalb der Ellipse, die des Dokuments innerhalb des (inneren) Rechtecks. Geht man von einer Gleichverteilung der Wahrscheinlichkeiten aus, erhält die dargestellte Konstellation die Wahrscheinlichkeit $\frac{5}{9}$*

Entscheidend für diese Bewertung sind die Wahrscheinlichkeiten der Terme und das Ähnlichkeitsmaß zwischen den Termen. Da sie lediglich an einer Rangordnung der Dokumente interessiert sind, wählen Crestani und van Rijsbergen (1995) anstelle einer Wahrscheinlichkeit das IDF-Maß

$$IDF(t_i) = -\log\left(\frac{n_i}{N}\right) \tag{144}$$

wobei $n_i$ die Anzahl der Dokumente in der Sammlung angibt, in denen der Term $t_i$ vorkommt, und $N$ die Gesamtzahl der Dokumente. (IDF steht auch hier für Inverse Document Frequency. Das Maß unterscheidet sich aber von dem IDF Maß, das im Darmstädter Indexierungs-Ansatz verwendet wurde. Lediglich qualitativ verhalten sich die Maße gleich: für kleine seltene Terme sind sie groß, für häufige klein. )

Als Ähnlichkeitsmaß wählen sie das EMIM Maß:

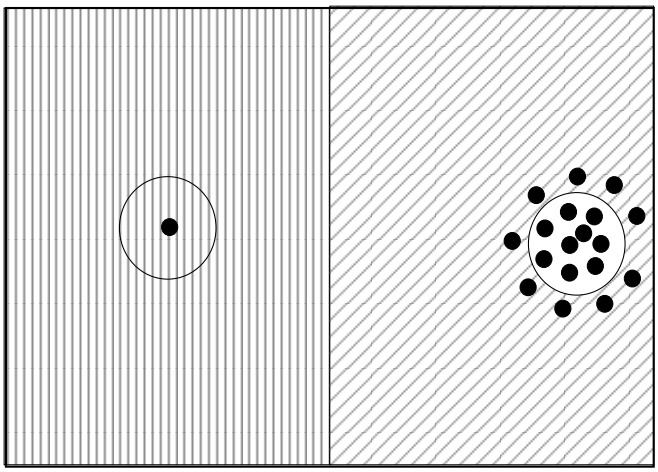
$$EMIM(t_i, t_j) = p(t_i, t_j)\log\left(\frac{p(t_i, t_j)}{p(t_i)\cdot p(t_j)}\right) \tag{145}$$

Dabei bedeutet $p(t_i)$ bzw. $p(t_j)$ die Wahrscheinlichkeit, daß der Term $t_i$ bzw. der Term $t_j$ gültig sind, also in einem Dokument auftreten, und $p(t_i, t_j)$ die Wahrscheinlichkeit, daß $t_i$ und $t_j$ zusammen gültig sind, also gemeinsam in einem Dokument auftreten.

Mit den Daten aus der Cranfield Collection wurden Versuche gemacht, wobei zwei Läufe verglichen wurden, einer mit Imaging und einer, in dem lediglich die IDF Werte als Gewichtungen verwendet wurden. Die Ergebnisse zeigen leicht bessere Ergebnisse für Imaging. Der Unterschied ist aber statistisch nicht signifikant.

Während bei dem hier vorgestellten Ansatz des Imaging die Dokumente wieder nur als Mengen von Termen behandelt wurden, können bei stärker strukturierten Dokumenten logische Verfahren besser zum Tragen kommen. Eine dieser Methoden ist die Abduktion.

**Abbildung 50: Probleme des Imaging**



*Die Abbildung zeigt schematisch eines der Probleme des Imaging Ansatzes. Die schwarzen Punkte stellen die Terme eines Dokuments dar. Die übrigen Terme der Termmenge seien gleichmäßig über die Fläche verteilt. Durch das Imaging werden die Wahrscheinlichkeiten aller Terme aus dem linken, senkrecht gestreiften Bereich auf den einzelnen linken Dokumentterm übertragen. Die Wahrscheinlichkeiten der Terme im rechten, schräg gestreiften Bereich werden auf die Dokumentterme im äußeren Ring verteilt. Lediglich die Wahrscheinlichkeiten der Terme im kleinen weißen Kreis werden auf die darin enthaltenen Terme übertragen. Betrachtet man die beiden Kreise als zwei verschiedene Anfragen, so erhält das Dokument bezüglich der Anfrage, die von dem rechten Kreis charakterisiert wird, lediglich ein Gewicht, das proportional zur Fläche des weißen Kreises ist, während die Anfrage, die vom linken Kreis symbolisiert wird, ein Gewicht erhält, das proportional zu der linken, senkrecht gestreiften Fläche und damit um ein Vielfaches größer ist.*

#### 4.1.2.2 Bayes'sche Inferenznetze

Ein bayes'sches Inferenznetz kann als ein gerichteter, azyklischer Graph definiert werden, dessen Knoten Aussagen und dessen Kanten Abhängigkeiten zwischen Aussagen darstellen. Die Knoten können Wahrscheinlichkeitswerte zwischen $0$ und $1$ annehem. Diese Wahrscheinlichkeiten dienen als Ausgangswerte zur Berechnung der Wahrscheinlichkeiten in den Knoten, auf die die Kanten zeigen. Dazu besitzt jeder Knoten eine Funktion oder Tabelle, die aus der Wahrscheinlichkeit der Knoten, die auf ihn zeigen, eine neue Wahrscheinlichkeit berechnet. In einfachsten Fall, wenn ein Knoten nur einen anderen Knoten hat, der auf ihn zeigt, kann das die einfache bedingte Wahrscheinlichkeit sein.

Zur Anwendung im Information Retrieval führen Turtle und Croft (1990, 1991) ein zweigeteiltes Inferenznetz ein. Der eine Teil, das *Document Network* (**Dokumentennetz**) besteht aus drei Schichten von Knoten,

1. der Dokumentenschicht
2. der Textrepräsentationsschicht und
3. der Konzeptrepräsentationsschicht.

Die Kanten des Netzes können nur zwischen den Schichten bestehen und nur von Dokumentknoten zu Textrepräsentationsknoten und von Textrepräsentationsknoten zu Konzeptrepräsentationsknoten führen. Dieser Teil des Netzes wird nur von den Dokumenten einer Dokumentensammlung bestimmt.

Der zweite Teil des Netzes, das Query Network (Anfragenetz) kann ebenfalls aus mehreren Schichten bestehen, wobei die erste Schicht mit der Konzeptrepräsentationsschicht verbunden ist und die letzte Schicht nur aus einem Knoten besteht, dessen Wahrscheinlichkeitswert als Wahrscheinlichkeit der mit dem Netz berechneten Inferenz interpretiert wird. Dieser Teil des Netzes ist nur von der jeweiligen Anfrage abhängig und nicht von den Dokumenten.

Um mit dem Netz eine Inferenz zu berechnen, werden die Werte der Dokumente, deren Relevanz zu einer Anfrage berechnet werden soll auf den Wert $1$ gesetzt, die Knoten der anderen Dokumente erhalten den Wert $0$. Nun werden bei der Textrepräsentationsschicht beginnend sukkzessive für die Knoten jeder Schicht die Wahrscheinlichkeitswerte berechnet, bis schließlich im letzten Knoten des Infereznetzes die Relevanz abgelesen wird.

**4.1.2.2.a Die Dokumentenschicht** In der ersten Schicht des Netzes, werden die verschiedenen Dokumente der Sammlung repräsentiert. Um die Dokumente in eine Rangreihe bezüglich einer Anfrage zu bringen, werden die Werte der Dokumentknoten einzeln auf Eins gesetzt und der Wahrscheinlichkeitswert der Inferenz berechnet. Es können aber auch mehrere Knoten auf Eins gesetzt werden, dann wird die Relevanz der entsprechenden Dokumentenmenge berechnet.

**4.1.2.2.b Die Textrepräsentationsschicht** Die Knoten dieser Schicht stehen für verschiedene Textteile oder Typen. Dabei kann ein Dokument mehrere Textrepräsentationsknoten besitzten. So könenn z. B. verschieden Teile eines Dokuments, wie – Titel oder Abstrakt in verschiedenen Knoten gespeichert sein. Bei größeren Dokumenten können auch verschiedene Kapitel oder Teile in verschiedenen Knoten gespeichert werden. Bei multimedialen Dokumenten können die verschiedenen Modalitäten wie Bilder, Audio– oder Viedeodaten verschiedenen Knoten zugewiesen werden. Auch Kommentare, Indexterme oder Stichwörter von Reviewern oder anderen Nutzern können verschiedenen Knoten zugeordnet werden.

**4.1.2.2.c Die Konzeptrepräsentationsschicht** Die Knoten dieser Schicht repräsentieren abstrakte Konzepte. Ihre Aktivierungen werden aus denen der Textrepräsentationsknoten berechnet. Sie stellen die Schnittstelle zum Anfragenetz dar. Ihre Definition wird i. a. schwierig sein.

Die drei Schichten des Dokumentennetzes stellen also eine zunehmende Abstrahierung der Dokumentinhalte von der jeweiligen Oberflächenform, in der sie in den Dokumenten vorliegen, dar. Neben der Definition der entsprechenden Knoten stellt insbesondere die Bestimmung der Übergangsfunktionen für die Knoten ein Problem dar. Sie müssen einfach genug sein, um auch für große Dokumentensammlungen berechnet werden zu können, andererseit hängt von ihnen der Erfolg der Methode ab.

Das Anfragenetz ist in der Theorie von Turtle und Croft einfacher strukturiert. Zwischen den Konzepten und dem Bewertungsknoten sehen sie eine Schicht mit verschiedenen Anfragetypen vor, die verschiedene Konzepte zusammenfassen.

Aus diesem theoretischen Modell leiten Turtle und Croft (1991) ein weiter vereinfachtes Modell her, das in dem Retrievalsystem INQUERY implementiert wurde. Dazu

- verzichten sie auf die Textrepräsentationsschicht,
- leiten die Konzepte unmittelbar aus den Termen, die in den Dokumenten vorkommen her,
- verzichten auf die Anfrageschicht im Anfragenetz

- Verwenden anstelle der Wahrscheinlichkeiten allgemeinere Werte, die aus einem IDF Maß und anderen Häufigkeitsmaßen der Terme gewonnen werden und
- gehen von einer Unabhängigkeitsannahme bei der Berechnung der Aktivierungswerte der Knoten aus.

Dadurch kann das System mit Hilfe einer invertierten Liste mit Wahrscheinlichkeitswerten implementiert werden und ist trotzdem in der Lage sehr große Dokumentsammlungen zu verwalten. Es reduziert sich allerdings auch im wesentlichen auf ein gewichtetes Vektorraummodell. Über diese Struktur hinaus soll auch die Integration von Abhängigkeiten zwischen Dokumenten und zwischen Termen möglich sein, wie das allerding realisiert wurde ist nicht ersichtlich.

#### 4.1.2.3 Abduktives Retrieval

Im Abschnitt 3.1.1 waren als verschiedene Formen des Lernens deduktives, induktives und abduktives Lernen eingeführt worden. Abduktives Lernen war als Hypothesenbildung über mögliche Ursachen beschrieben worden.

Im Information Retrieval sind Anfragen in der Regel unscharf und vage. Sie können oft auf verschiedene Weisen interpretiert werden. Insbesondere wenn die Dokumente stärker strukturiert sind, ist eine solche Interpretation notwendig. Hier können abduktive Verfahren eingesetzt werden, um verschiedene Interpretationen einer Anfrage zu generieren und Nutzenden zur Auswahl anzubieten. In einem Dialog über die möglichen Interpretationen können Anfragen präzisiert werden.

Als Modellansatz wird angenommen, daß bei Nutzenden ein Informationsbedarf besteht, als dessen Folge eine Anfrage formuliert wird. Der abduktive Retrievaldialog schließt von der Folge – nämlich der eingegebenen Anfrage – auf die möglichen Ursachen – nämlich den Informationsbedarf. (Dabei übersetzt er gleichzeitig den Informationsbedarf der Nutzenden in die interne Repräsentation der Dokumente.)

Als Beispiel soll hier kurz das am GMD - IPSI entwickelte abduktive Retrievalsystem MIRACLE vorgestellt werden. Es arbeitet z. B. auf einer Sammlung von Biographien von Künstlern, die in einer SGML-ähnlichen Notation vorliegen. Aus den Texten der Biographien können mit Hilfe der Tagstruktur Attribute extrahiert werden, wie z. B. Geburtsort und –datum, Sterbeort und –datum eines Künstlers, wichtige Austellungen und Lebensdaten.

Eine Anfrage nach “Pariser Künstlern der Jahrhundertwende” z. B. läßt eine ganze Reihe von möglichen Interpretationen, d. h. Rückschlüssen auf den tatsächlichen Informationsbedarf der oder des Anfragenden zu. Z. B kann ein “Pariser Künstler” ein Künstler sein, der in Paris geboren ist, einer, der in Paris gelebt hat, oder einer, dessen Werke vor allem in Paris zu finden sind. Die Zeitangabe “Jahrhundertwende” muß auch interpretiert werden. Während es noch naheliegt, einen Zeitraum vom Ende des 19. Jahrhunderts bis zum Anfang des 20. Jahrhunderts anzunehmen, wird es schon erheblich unkarer, wie lang der Zeitraum anzusetzen ist.

Eine mögliche Interpretation der Anfrage wäre auch die, daß Künstler gesucht werden, die gerade zu dieser Zeit in Paris waren, während sie sonst eventuell eher anderen Orten zugeordnet werden, daß also die Zeitangabe als eine Modifikation der Ortsangabe gemeint war.

Um diese Interpretationen zu leisten, muß das IR System über eine Regelbasis verfügen, die die Domäne der Dokumente beschreibt. So sollte in einer solchen Basis z. B. stehen, daß eine Person, die ein Künstler oder eine Künstlerin ist, ein Geburtsdatum hat, daß sie entweder einen Todestag hat oder noch lebt, und daß der Todestag zeitlich

nach dem Geburtstag liegt. Es kann aber natürlich auch vorkommen, daß ein Geburtstag nicht (genau) bekannt ist, oder daß zwei Daten in verschiedenen Zeitrechnungen angegeben sind. Auch solche Fälle müssen durch die Regeln der Regelbasis abgedeckt werden. Mit den Regeln dieser Basis werden die Anfragen interpretiert. Die Konstruktion einer solchen Regelbasis ist entsprechend aufwendig. Für neue Themengebiete müssen jeweils die gebietsspezifischen Regeln neu entwickelt werden.

## 4.2 Korpusbasierte Verfahren

Im probabilistischen Ansatz und bei der Implementierung des Inferenznetzes wurde angenommen, daß das Auftreten von Termen in der Sprache unabhängig ist; d. h. es wurde angenommen, daß die Wahrscheinlichkeit, zwei Terme im selben Dokument zu finden, nur von deren globalen Auftretenswahrscheinlichkeiten abhängt und nicht von ihrer jeweiligen "Bedeutung". Diese Annahme scheint im allgemeinen recht unrealistisch (vergleiche Abbildung 51). Sie wurde gemacht, um Berechnungs- bzw. Schätzmöglichkeiten für andere Wahrscheinlichkeiten zu haben; ohne sie hätten keine Werte für die in den jeweiligen Ansätzen gesuchten Wahrscheinlichkeiten geschätzt werden können. Allerdings können bei starken Vereinfachungen die Schätzwerte unrealistisch werden (Cooper 1991).

### 4.2.1 Der assoziative Ansatz

Ein erster Schritt, die Unabhängigkeitsannahmen zu vermeiden, sind Modelle, die mit Kookurrenzdaten arbeiten. Unter **Kookurrenzdaten** versteht man dabei Häufigkeitsdaten über das gemeinsame Auftreten von Termen. Diese Daten werden aber i. a. nicht genutzt um die Wahrscheinlichkeiten aus dem Modell des probabilistischen Retrievals oder bei Inferenznetzen zu berechnen, sondern um Ähnlichkeitsmaße zwischen Termen zu berechnen (wie sie schon beim Imaging verwendet wurden).

Die diesem Ansatz zugrundeliegende Assoziationstheorie läßt sich bereits bei Aristoteles finden. Der englische Psychologe William James formulierte sie 1890 folgendermaßen:

> *"objects once experienced together tend to become associated in the imagination, so that when any one of them is thought of, the others are likely to be thought of also"*

(James, 1890, Band 1, Seite 561)

Mit dieser einfachen Annahme kann man versuchen, aus großen Textkorpora Assoziationen zwischen Termen zu berechnen und sie zur Darstellung von Begriffen oder Konzepten zu verwenden, wie das bereits beim Imaging angenommen wurde: Dort wurden Terme durch die Dokumente charakterisiert, in denen sie auftraten.

Diese Methode kann auch als anderer und allgemeinerer Ansatz zu den Überlegungen aus Abschnitt 2.2.1 zur Grundformenreduktion aufgefaßt werden. Dort war zwischen einem Pattern Recognition Ansatz, der Terme als Zeichenketten behandelt, und einem computerlinguistischen Ansatz, der Terme als Formen eines Wortes ansieht, unterschieden worden. Im letzten Fall wird ein Wort als "Oberflächenform" eines Begriffs oder Konzepts aufgefaßt. In diesem Zusammenhang waren bereits die Probleme der Synonymie und Polysemie angesprochen worden, also die Tatsache, daß zwei verschiedene

**Abbildung 51: Aus dem Lob– und dem Brown-Korpus mit Kookurrenzdaten berechnete Assoziationen zu drei Termen**

| tax: | fruit: | sin: |
|---|---|---|
| income: 71.81 | eggs: 56.69 | crime: 107.11 |
| fiscal: 66.96 | meat: 56.69 | doctrine: 98.31 |
| taxes: 61.99 | foods: 55.09 | morality: 92.00 |
| profits: 56.67 | fresh: 54.99 | adam: 87.57 |
| revenue: 56.35 | seed: 52.16 | christ: 57.57 |
| sales: 42.85 | sugar: 42.99 | jesus: 54.83 |
| reduction: 41.33 | milk: 40.12 | suffering: 52.18 |
| file: 39.18 | meal: 32.20 | flesh: 50.89 |
| paying: 35.93 | tree: 31.55 | original: 47.89 |
| payments: 32.81 | believes: 31.42 | born: 45.49 |
| collection: 31.20 | soft: 31.30 | burden: 39.32 |
| towns: 30.29 | tea: 30.09 | consequently: 36.19 |
| estimated: 29.64 | expenditure: 29.98 | heaven: 36.19 |
| finance: 29.06 | wine: 29.64 | god: 35.91 |
| net: 28.81 | fish: 26.98 | creation: 34.78 |
| uniform: 27.23 | breakfast: 26.89 | requires: 34.35 |
| corporation: 26.70 | containing: 26.08 | grace: 30.84 |
| purchase: 26.37 | eat: 25.82 | death: 30.78 |
| spending: 25.13 | referred: 25.08 | moral: 30.70 |
| excess: 24.21 | parks: 23.29 | darkness: 30.16 |

*Assoziationen unter den Termen, die auf den Häufigkeitsrangplätzen 100-3100 in einem Korpus, der aus Brown-Korpus und Lob-Korpus zusammengesetzt war, auf die Terme "tax", "fruit" und "sin" gefunden wurden. Zur Berechnung der Assozationen wurden Kookurrenzen auf Satzebene erhoben und der Quotient aus der relativen Häufigkeit des gemeinsamen Auftretens und dem Produkt der relativen Häufigkeiten der einzelnen Terme verwendet.*

*Die Annahme, daß das Auftreten von Termen unabhängig sei, scheint wenig berechtigt. Sonst müßten die angegebenen Assoziationen alle nahe bei Eins liegen.*

Wörter das selbe Objekt bezeichnen können, bzw. daß ein Wort zwei verschiedene Objekte bezeichnen kann. Ersetzt man "Objekt" durch "Begriff" oder "Konzept" sieht man, daß dieses Problem viel allgemeiner besteht, daß einem Begriff nämlich häufig mehrere Wörter zugeordnet werden können, die ihn charakterisieren, bzw. ein Wort zur Charakterisierung von verschiedenen Begriffen verwendet werden kann.

Der assoziative Ansatz stellt eine einfache Modellierung dieser Beobachtung dar: er charakterisiert die "Bedeutung" eines Terms durch dessen Beziehungen zu anderen Termen, konkret durch eine Menge von Gewichten, die die Ähnlichkeiten zu den anderen Termen eines Vokabulars beschreiben. Angewandt auf das Information Retrieval, versucht der assoziative Ansatz durch ein Ähnlichkeitsmaß zwischen Termen, einer Indexierung oder einer Anfrage weitere Terme hinzuzufügen, die im allgemeinen zur Charakterisierung eines Begriffes oder Konzepts verwendet werden, in dem Dokument bzw. der Anfrage aber nicht auftauchen.

### 4.2.2 Kookurrenzverfahren

Untersuchungen, die mit Kookurrenzdaten arbeiten, wurden bereits Anfang der 60er Jahre durchgeführt (Giuliano & Jones, 1963). Seitdem ist das Thema immer wieder aufgegriffen, häufig aber als nicht erfolgreich bewertet worden (Salton & Buckley 1988, Peat & Willett 1991). Ein grundsätzliches Problem des Ansatzes ist die Aufwendigkeit in bezug auf Rechenkapazität und Speicherplatz sowie die Notwendigkeit, für die Bestimmung der Kookurrenzdaten große Korpora zur Verfügung zu haben, um brauchbare Schätzungen für die Wahrscheinlichkeiten des gemeinsamen Auftretens von Termen zu gewinnen (Gale & Church, 1990).

Formal lassen sich Kookurrenzmodelle auf mehrere Arten herleiten. Zunächst greifen wir auf die Definition 2.3.1.1 des Vektorraumodells zurück. Dort waren die Dokumente einer Sammlung $D = \{d_1, ..., d_m\}$ als Vektoren $w_i = (w_{i,1}, ..., w_{i,n})$ über den Termen $T = \{t_1, ..., t_n\}$ oder den Attributen $A = \{A_1, ..., A_n\}$ definiert worden, wobei der Eintrag $w_{i,j}$ in einem Dokumentvektor die Wichtigkeit des Terms $t_j$ im Dokument $d_i$, bzw. den Wert des Attributes $A_j(d_i)$ angab. Diese Vektoren kann man als Zeilen einer **Term-Dokument-Matrix**

$$W = \{w_{i,j}\}_{i=1,...,m;\ j=1,...,n} \tag{146}$$

schreiben. Diese Matrix oder Relation enthält also alle Informationen einer Dokumentensammlung, die sich mit der Repräsentation durch Terme bzw. Attribute darstellen lassen. Auf sie können formal die Data Mining Verfahren angewendet werden, die im Teil 3 besprochen wurden.

#### 4.2.2.1 Ein Machine Learning Ansatz

Eine solche Anwendung ist in Apte, Damerau und Weiss (1994) beschrieben. Hier ging es darum, eingehende Meldungen einer Nachrichtenagentur nach verschiedenen Sachgebieten zu verteilen. Dazu wurden die Meldungen mit einem regelbasierten Verfahren nach verschiedenen Kategorien aufgeteilt. Allerdings wurden dazu lediglich zwischen 30 und 200 Terme, also ein sehr kleines Vokabular, berücksichtigt. Die Anzahl der Kategorien war mit maximal 47 ebenfalls relativ klein. Sie mag für diesen Zweck angemessen sein, für ein Information Retrieval System wäre eine solche Kategorisierung allerdings viel zu grob (bei einer Sammlung von einer Million Dokumenten wären das im Durchschnitt über 20 000 Dokumente pro Kategorie).

Würden mehr Kategorien verwendet, müßten vermutlich auch mehr Terme verwendet werden, um die Kategorien noch trennen zu können, zum anderen müßte ein erheblich größeres Trainingsset verwendet werden. Dann würden sich vermutlich auch Inkonsistenzen im Trainingsset stärker bemerkbar machen.

Die direkte Anwendung von Machine Learning Verfahren auf den Term-Dokument-Matrizen wie in dieser Studie sind allerdings die Ausnahme. Im allgemeinen werden Term-Dokument-Matrizen dazu verwendet, Ähnlichkeiten zu berechnen. Für Ähnlichkeiten zwischen Dokumenten wurde das im Abschnitt 2.3.1.1 über Vektorraummodelle beschrieben. Aus diesen Verfahren lassen sich nicht unmittelbar Regelmäßigkeiten der Dokumentensammlung ablesen. Verfahren zur Berechnung von Ähnlichkeiten zwischen Termen können aber als Methoden der Regelextraktion gesehen werden: ihre Ergebnisse geben Auskunft über die Verwendung von Termen in den Dokumenten der Sammlung und können in eine für Nutzende verständliche Form gebracht werden.

#### 4.2.2.2 Term-Term-Matrizen

Um Ähnlichkeiten zwischen Termen zu bestimmen, kann man die $n \times n$–Matrix $W^t \cdot W$ berechnen. Der Eintrag $\tilde{w}_{i,j}$ in dieser Matrix ist dann das Skalarprodukt der $i$–ten und

der $j$–ten Spalte der Matrix. Die $i$–te Spalte ist ein Vektor, der angibt, wie wichtig der Term $t_i$ für die einzelnen Dokumente $d_1, ..., d_m$ ist, bzw. welchen Wert das Attribut $A_i$ in den einzelnen Dokumenten $d_1, ..., d_m$ annimmt.

Im Bool'schen Fall sind das jeweils binäre Vektoren, die angeben, ob der Term $t_i$ im Dokument $d_k$ vorkommt ($w_{k,i} = 1$) oder nicht ($w_{k,i} = 0$). Der Eintrag $\tilde{w}_{i,j}$ in der Matrix $W^t \cdot W$ gibt in diesem Fall nach der Definition des Skalarprodukts also gerade die Anzahl der Dokumente an, in denen die Terme $t_i$ und $t_j$ gemeinsam vorkommen. In den allgemeineren Fällen von gewichteten Termen, bzw. Attributen gibt der entsprechende Eintrag einen Ähnlichkeitswert an. Es können auch andere Ähnlichkeitsmaße verwendet werden, wie sie in Abschnitt 2.3.3 besprochen wurden. Die dort für die Dokumente diskutierten Eigenschaften der einzelnen Maße lassen sich auch auf die Terme übertragen.

Die Ähnlichkeitswerte zwischen den Termen können in einer $n \times n$–Matrix gespeichert werden. Sie wird **Term-Term-Matrix** genannt. Sind die Ähnlichkeiten symmetrisch, ist auch die Term-Term-Matrix symmetrisch.

#### 4.2.2.3 Anwendung im IR

Mit den Ähnlichkeitsmaßen lassen sich verschiedene Schritte des Information Retrievals unterstützen.

- Ähnlichkeitsmaße können eingesetzt werden, um einen **themenspezifischen assoziativen Thesaurus** zu generieren. Dazu können zu jedem Term die anderen Terme bezüglich ihrer Ähnlichkeit in eine Rangreihe gebracht werden. Als assoziativ verwandte Terme können dann alle Terme, deren Ähnlichkeit eine Schwelle überschreitet, angenommen werden. Eine andere Wahl wäre eine feste Anzahl von Termen mit den höchsten Gewichten.
  Dieser Thesaurus könnte Indexierenden und Anfragenden zur Verfügung gestellt werden, um weitere zu einer Anfrage verwandte Terme ausgesucht werden können.
- Die Ähnlichkeiten können auch direkt beim **Indexierungsprozeß** verwendet werden, um eine Menge von Termen, die von Hand oder mit anderen Verfahren ermittelt wurde, um weitere Terme zu erweitern.
- Analog können die Ähnlichkeiten genutzt werden, um die Terme einer **Anfrage** mit weiteren ähnlichen Termen zu **ergänzen** (sog. **Query Expansion**), und die Anfrage entsprechen breiter zu machen.

Die beiden letztgenannten Möglichkeiten der automatischen Verwendung von ähnlichen Termen haben in diversen Untersuchungen nicht zu signifikanten Verbesserungen der Retrievalergebnisse geführt (vergl. Peat und Willet 1991). Allerdings lassen sich dafür durchaus auch mögliche Gründe in den jeweiligen Untersuchungen finden.

#### 4.2.2.4 Häufigkeit der Terme

Im Abschnitt 2.3.4 über Gewichtungsmethoden und beim probabilistischen Retrieval waren IDF-Maße eingeführt worden, die seltenen Termen höhere Gewichte zuordnen. Diese Gewichtung wurde vorgenommen, weil seltenen Termen eine höhere Diskriminationsfähigkeit zugeschrieben wird (vergleiche Abbildung 19). Peat und Willett (1991) führen die Mißerfolge der automatischen Expansion darauf zurück, daß dabei häufige und damit wenig nützliche Terme gefunden werden. Sie haben diesen Effekt für drei Maße, das Cosinusmaß

$$COSINE(X,Y) = \frac{H(X,Y)}{\sqrt{H(X)H(Y)}} \tag{147}$$

das Dice-Maß

$$DICE(X,Y) = \frac{2 \cdot H(X,Y)}{H(X) + H(Y)} \tag{148}$$

und das Tanimoto-Maß

$$TANIMOTO(X,Y) = \frac{H(X,Y)}{H(X) + H(Y) - H(X,Y)} \tag{149}$$

untersucht. Dabei bezeichnet $H(X)$ die Anzahl der Dokumente, in denen der Term $X$ auftritt, und $H(X,Y)$ die Anzahl der Dokumente, in denen die Terme $X$ und $Y$ zusammen vorkommen.

Betrachtet man die Formeln aus statistischer Sicht, zeigt sich, daß sie häufige Terme begünstigen: Während die angegebenen Maße Ähnlichkeiten oder Winkel zwischen Vektoren messen, kann man auch überlegen, ob die Häufigkeit, mit der zwei Terme zusammen auftreten, zufällig, überzufällig oder unterzufällig ist.

Falls $p(X)$ die Wahrscheinlichkeit des Auftretens des Terms $X$ in einem Dokument und $p(X\&Y)$ die Wahrscheinlichkeit des gemeinsamen Auftretens der Terme $X$ und $Y$ in einem Dokument bezeichnen, gilt nach der Definition der statistischen Unabhängigkeit bei zufälligem gemeinsamen Auftreten der Terme

$$p(X\&Y) = p(X) \cdot p(Y) \tag{150}$$

Der Quotient

$$\frac{p(X\&Y)}{p(X) \cdot p(Y)} \tag{151}$$

ist also genau dann kleiner als $1$, wenn die Terme unterzufällig häufig zusammen auftreten, er ist gleich $1$, wenn sie genau zufällig häufig zusammen auftreten, und er ist größer als $1$, wenn sie überzufällig häufig zusammen auftreten.

Ersetzt man die Wahrscheinlichkeiten durch relative Häufigkeiten, erhält man die Formel

$$U(X,Y) = A \cdot \frac{H(X\&Y)}{H(X) \cdot H(Y)} \tag{152}$$

wobei $A$ die Anzahl der Dokumente bezeichnet, die bei der Berechnung der relativen Häufigkeit herangezogen wurden, und als konstanter Faktor in den folgenden Überlegungen ignoriert werden kann.

Vergleicht man diesen Quotienten mit den Ähnlichkeitsmaßen, so zeigt sich, daß alle drei Maße häufige Terme stärker begünstigen als dieser. Der $COSINE(X,Y)$ z. B. unterscheidet sich von $U(X,Y)$ durch den Faktor $F_c = \sqrt{H(X)H(Y)}$:

$$\begin{aligned} COSINE(X,Y) &= \frac{H(X,Y)}{\sqrt{H(X)H(Y)}} \\ &= \frac{H(X,Y)}{H(X)H(Y)} \cdot \sqrt{H(X)H(Y)} \\ &= U(X,Y) \cdot \sqrt{H(X)H(Y)} \end{aligned} \tag{153}$$

Je häufiger ein Term ist, desto größer ist also der Faktor $F_c$.

Für die beiden anderen Maße lassen sich ähnliche Effekte zeigen:

Um einen Faktor $F_D$ für das Dice–Maß zu berechen setzen wir

$$f \cdot DICE(X,Y) = U(X,Y) \tag{154}$$

mit $f \in \mathbb{R}$. Dann folgt

$$f = \frac{H(X,Y)}{H(X) \cdot H(Y)} \cdot \frac{2(H(X)+H(Y))}{H(X,Y)} = 2\frac{H(X)+H(Y)}{H(X,Y)} \tag{155}$$

und

$$f = 2\left(\frac{1}{H(Y)} + \frac{1}{H(X)}\right) \tag{156}$$

Mit $F_D = \frac{1}{f}$ erhalten wir schließlich

$$DICE(X,Y) = F_D \cdot U(X,Y) \tag{157}$$

Auch dieser Faktor $F_D$ wächst mit zunehmender Häufigkeit der Terme $X$ und $Y$.
Für das Tanimoto-Maß schließlich führt

$$f \cdot TANIMOTO(X,Y) = U(X,Y) \tag{158}$$

zu

$$f = \frac{H(X,Y)}{H(X) \cdot H(Y)} \cdot \frac{H(X)+H(Y)-H(X,Y)}{H(X,Y)} \tag{159}$$

und

$$f = \frac{H(X)+H(Y)-H(X,Y)}{H(X) \cdot H(Y)} = \frac{1}{H(Y)} + \frac{1}{H(X)} - \frac{H(X,Y)}{H(X) \cdot H(Y)} \tag{160}$$

Da $H(X) \geq H(X,Y)$ und $H(Y) \geq H(X,Y)$ gilt, folgt

$$\frac{1}{H(Y)} + \frac{1}{H(X)} \geq \frac{H(X,Y)}{H(X) \cdot H(Y)} \tag{161}$$

und damit

$$0 \leq f \leq \frac{1}{H(Y)} + \frac{1}{H(X)} =: g \tag{162}$$

Schließlich können wir $F_T = \frac{1}{g}$ setzen und erhalten

$$TANIMOTO(X,Y) = \frac{1}{f} \cdot U(X,Y) \geq F_T \cdot U(X,Y) \tag{163}$$

Auch der Faktor $F_T$ wächst also mit zunehmender Häufigkeit der beteiligten Terme und begünstigt damit häufige Terme.

#### 4.2.2.5 Expansion von Termen oder Anfragen

In vielen Untersuchungen zur Query Expansion werden zu den einzelnen Termen einer Anfrage jeweils weitere Terme hinzugefügt (Rijsbergen, Harper & Porter 1981, Peat & Willett 1991, Grefenstette 1992). Dieses Verfahren ignoriert den Kontext, in dem ein Term benutzt wird. Es ist aber zu vermuten, daß sich gerade bei häufigen Termen die Bedeutung aus dem Kontext erschließt. Das kann auch heißen, daß auf einen seltenen spezifischen Term nur geschlossen werden kann, wenn die gesamte Anfrage berücksichtigt wird.

Die Charakterisierung der Bedeutung von Termen als Vektoren mit Ähnlichkeiten zu den anderen Termen des Vokabulars erlaubt es, den Einfluß aller Terme einer Anfrage zusammenzufassen. Dazu können die Vektoren addiert werden. Diese einfache lineare Kombination ist sicherlich nur eine sehr einfache Modellierung des Kontexts, es ist mit diesem Modell aber möglich, daß sich mehrere schwache Einflüsse von verschiedenen, häufigen Termen auf einen seltenen Term addieren und dieser Term dadurch in die Anfrage aufgenommen wird, auch wenn er bei der Ergänzung der einzelnen Terme nicht aufgenommen worden wäre. Entsprechend ist es auch möglich, daß durch negative Einflüsse ein Term nicht aufgenommen wird, der bei der Ergänzung der einzelnen Terme aufgenommen worden wäre.

#### 4.2.2.6 Größe der Dokumentensammlung

Die Größe der Dokumentensammlungen, aufgrund derer Ähnlichkeitsmaße in den von Peat und Willett (1991) untersuchten Studien berechnet wurden, variiert von unter 100 (Giuliano & Jones 1963) bis zu ca. 27 000. Das hat zum einen technische Gründe, zum anderen wurden Sammlungen verwendet, zu denen Relevanzeinschätzungen vorliegen. Diese Anforderungen zwingen dazu, die Größe der Sammlungen zu beschränken. Es hat sich aber gezeigt, daß die Güte der Simulation sprachlicher Leistungen aus Kookurrenzdaten stark von der Größe der zugrundeliegenden Korpora abhängt (Rapp 1991, Rapp 1993, Wettler, Rapp & Ferber 1993).

#### 4.2.2.7 Eine Untersuchung

In einer Untersuchung von Ferber, Wettler und Rapp (1995) wurden aus Dokumenten der Datenbank PsycLIT Kookurrenzdaten erhoben und mit Formel 152 Ähnlichkeiten zwischen den Termen eines Vokabulars aus 872 Termen berechnet. Das Vokabular stammte aus 94 Protokollen von Anfragen an die Zentralstelle für psychologische Information und Dokumentation in Trier. Diese Protokolle enthielten schriftliche, natürlichsprachliche Anfragen und die Datenbanksuchen, die die professionellen Recherchierenden der Zentralstelle dazu durchgeführt hatten. Ziel der Untersuchung war es, aus den schriftlichen Anfragen mit den Ähnlichkeitsmaßen die Terme vorherzusagen, die die Recherchierenden in den Anfragen verwendet hatten. Zusätzlich wurde die Anzahl der Dokumente, aus denen die Kookurrenzdaten erhoben wurden, systematisch zwischen 20 000 und 246 889 variiert.

Zur Vorhersage der Terme wurde zunächst zu jeder Anfrage ein Anfragevektor konstruiert, indem die Werte der Terme auf die Häufigkeit ihres Auftretens in der schriftlichen Anfrage gesetzt wurden. Dieser Vektor wurde dann mit der Term-Term-Matrix mit den Ähnlichkeiten multipliziert und die Terme nach ihren Werten in dem resultierenden Vektor in eine Rangreihe gebracht.

Zur Auswertung wurden bezüglich dieser Rangreihe die mittleren Rangplätze der Terme ermittelt,

- die sowohl in der Anfrage als auch in der Recherche vorkamen (IN-IN),

- die nur in der Anfrage, nicht aber in der Recherche vorkamen (IN-OUT), und
- die nur in der Recherche, nicht aber in der Anfrage vorkamen, die also vom Recherchierenden neu gewählt wurden (OUT-IN).

Zusätzlich wurde ein Overlapmaß verwendet, das die Überschneidung zwischen den Termen, die die Recherchierenden verwendet hatten, und der gleichen Anzahl von Termen an der Spitze der Rangreihe mißt. Ein ganz ähnliches Maß wurde in einer Studie von Sarcevic und Kantor (1988) dazu verwendet, die Übereinstimmung der Wortwahl zwischen verschiedenen Recherchierenden zu messen. Sie erhielten – mit anderem Material – eine mittlere Übereistimmung von $0.27$

**Abbildung 52: Ergebnisse der Studie zur Simulationen der Wortwahl (Ferber, Wettler & Rapp 1995)**

---

**Eichstichprobe:**

| Anzahl Dokumente | IN-IN | OUT-IN | IN-OUT | OVL |
|---|---|---|---|---|
| 246,889 | 18.5 | 155.6 | 194.4 | 0.39 |
| 136.887 | 18.4 | 159.8 | 186.8 | 0.38 |
| 75,000 | 19.4 | 165.9 | 165.2 | 0.40 |
| 40,000 | 21.8 | 183.5 | 147.8 | 0.38 |
| 20,000 | 21.3 | 186.3 | 164.4 | 0.38 |

**Teststichprobe:**

| Anzahl Dokumente | IN-IN | OUT-IN | IN-OUT | OVL |
|---|---|---|---|---|
| 246,889 | 18.9 | 172.2 | 203.7 | 0.41 |
| 136.887 | 18.5 | 174.7 | 196.3 | 0.39 |
| 75,000 | 18.5 | 197.7 | 170.2 | 0.40 |
| 40,000 | 21.5 | 211.6 | 150.8 | 0.37 |
| 20,000 | 20.1 | 228.8 | 178.1 | 0.37 |

---

*Die erste Spalte gibt die Größe der zugrundeliegenden Dokumentsammlung an, die zweite den mittleren Rangplatz (bei 872 mö glichen Rangplätzen) der IN-IN Terme, die dritte den der OUT-IN Terme, die vierte den der IN-OUT Terme und die letzte den Overlap.*

Die Ergebnisse dieser Studie sind in Abbildung 52 angegeben. Sie zeigen gerade bei den neu gewählten Termen eine Abhängigkeit der Ergebnisse von der Größe der zugrundegelegten Dokumentensammlung. Außerdem zeigen sie, daß der Overlap nicht schlechter ist als der (etwas anders berechnete) Overlap bei den Experimenten von Saracevic und Kantor (1988).

### 4.2.2.8 Komplexere Kookurrenzverfahren

Andere Untersuchungen, die mit Kookurrenzdaten arbeiten, verwenden kompliziertere Attribute als nur das Auftreten von Termen in Dokumenten. Grefenstette (1992) und Ruge (1992) verwenden eine grobe syntaktische Analyse zur Bestimmung der Kookurrenz. Dabei werden nur Wörter in bestimmten syntaktischen Positionen berücksichtigt.

In einem ähnlichen Ansatz verwenden Jing und Croft (1994) Satzteile (“phrases”), die einer bestimmten Regel genügen. Eine einfache Regel für einen Satzteil (noun-phrase)

wäre z. B. $\{N, NN, NNN\}$, d. h., daß ein einzelnes Hauptwort der Regel genügt, oder auch zwei oder drei unmittelbar hintereinander stehende Hauptwörter (im englischen enspricht diese Regel zum Teil unserer Kompositionsregel für Substantive: Immigration officer = Grenzbeamter). Eine andere Regel wäre $\{JNN, JNNN\}$, wobei $J$ für ein Adjektiv steht.

Zwischen den Termen und den Satzteilen, die in einem Satz gefunden werden, werden Assoziationen gebildet, indem die Auftretenshäufigkeiten miteinander multipliziert werden. Diese Assoziationen werden zu einem assoziativen Thesaurus zusammengefaßt. Um dieses Verfahren anzuwenden, muß für die einzelnen Terme bestimmt werden, um was für eine Wortart es sich handelt. Experimentelle Studien haben für einige Dokumentensammlungen bei der Verwendung des assoziativen Thesaurus zur Query Expansion Verbesserungen der Ergebnisse gegenüber dem Retrieval ohne Expansion gezeigt (Jing und Croft 1994).

### 4.2.3 Latent Semantic Indexing

Ein weiteres Verfahren, Strukturen in Dokumentsammlungen zu finden, ist die "Latent Semantic Indexing" genannte Methode, die z. B. von Deerwester, Dumais, Furnas, Landauer und Harshman (1990) beschrieben wurde. Allerdings werden diese Strukturen nur sehr bedingt zugänglich gemacht.

Das Verfahren setzt auf der Term-Dokument-Matrix $W$, wie sie in Abschnitt 4.2.2 beschrieben wurde, auf. Der durch diese Matrix beschriebene hochdimensionale Vektorraum wird nun auf einen Vektoraum mit erhebliche kleinerer Dimension projeziert, der durch ein der Faktorenanalyse ähnliches Verfahren bestimmt wird. Dazu wird die Matrix $W$ mit einem "singular value decomposition" Verfahren in die Form

$$W = TSD' \tag{164}$$

gebracht, wobei $T$ und $D$ orthonormale Spalten haben und $S$ eine Diagonalmatrix ist, mit entlang der Hauptdiagonalen absteigenden, nicht negativen Einträgen. Der niedrigdimensionale Vektorraum wird nun dadurch erzeugt, daß in der mittleren Matrix $S$ lediglich die ersten $r$ Zeilen beibehalten, und alle anderen Diagonalelemente auf $0$ gesetzt werden. Dadurch, daß die Elemente in der Hauptdiagonalen der Größe nach geordnet vorliegen, sollen so die wesentlichen Eigenschaften erhalten bleiben.

Experimente mit der Methode zeigen, daß sich bei der MED (vergl. Abbildung 21) Dokumentensammlung Verbesserungen im Precision-Recall-Diagramm ergeben, bei der CISI Sammlung ergeben sich dagegen keine Verbesserungen. Die Autoren vermuten, daß das an der Zusammenstellung der Sammlungen liegt. CISI scheint erheblich homogener zu sein als MED.

## 4.3 Schlußbemerkung

In dieser Vorlesung sollte sich gezeigt haben, daß sich Information Retrieval und Data Mining zwar unter einem gemeinsamen formalen Modell darstellen lassen, daß die Interaktion zwischen den beiden Disziplinen aber sowohl in der Entwicklung als auch in den Methoden bisher nicht sehr intensiv war. Das hat zum einen mit den unterschiedlichen Schwerpunkten in der Zielsetzung und Entwicklung zu tun, wie sie in Abbildung 4 dargestellt sind, zum anderen mit der "Herkunft" der beiden Disziplinen.

Es ist aber zu erwarten, daß sich mit zunehmender Größe und Vielfalt von Faktendatenbanken einerseits und der zunehmenden Verbreitung von strukturierten Dokumentrepräsentationen andererseits die beiden Gebiete aufeinander zu entwickeln werden. Zudem wird durch die Anforderungen an multimediales Retrieval der Einsatz von komplexeren Attributen auf Datenbankobjekte zunehmen und damit Verfahren notwendig machen, die auf solche nichttextuellen Daten eingerichtet sind.

## Literaturverzeichnis


Apte, C., Damerau, F., & Weiss, S. M. (1994). Towards language independent automated learning of text categorization models. In *Prodeedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval* (Dublin, 1994).

Cooper, W. (1991). Some inconsistencies and misnomers in probabilistic information retrieval. In *Proceedings of the 14th Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval SIGIR '91* (1991), 57-61.

Crestani, F., & Van Rijsbergen, C. J. (1995). Information retrieval by logical imaging. *Journal of Documentation 51*(1), 3-17.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science 41*(6), 391-407.

Ferber, R., Wettler, M., & Rapp, R. (1995). An associative model of word selection in the generation of search queries. *Journal of the American Society for Information Science (JASIS) 46*(9), 685-699.

Fuhr, N. (1995). Information retrieval. Skriptum zur Vorlesung im SS 93, Universität Dortmund, http://ls6–www.informatik.uni-dortmund.de/teaching/courses/ir.

Gale, W. A., & Church, K. W. (1990). Poor estimates of context are worse than none. In *DARPA Speech and Natural Language Workshop* (Hidden Valley, PA, 1990), 283-287.

Giuliano, V. E., & Jones, P. E. (1963). Linear associative information retrieval. In *Vistas in Information Handling*, P. W. Howerton & D. C. Weeks, Eds., vol. 1. Spartan Books, Washington D. C., Washington, D.C., ch. 2, 30-54.

Grefenstette, G. (1992). Use of syntactic context to produce term association lists for text retrieval. In *Proceedings of the Fifteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (1992), ACM SIGIR, 89-97.

James, W. (1890). *The Principles of Psychology*. New York: Holt, Reprinted New York: Dover Publications, 1950.

Jing, Y., & Croft, W. B. (1994). An association thesaurus for information retrieval. In *Proceedings of the RIAO 94* (1994), vol. 1, 146-160.

Peat, H. J., & Willett, P. (1991). The limitations of term co-occurrence data for query expansion in document retrieval systems. *Journal of the American Society for Information Science 42*(5), 378-383.

Rapp, R. (1991). Ein statistisches Modell für die maschinelle Sprachverarbeitung. *Grundlagenstudien aus Kybernetik und Geisteswissenschaft 32*(4), 163-176.

Rapp, R. (1993). A statistical model to verify verbal material. In *Sprache-Kommunikation-Informatik. Akten des 26. Linguistischen Kolloquiums.* (1993), J. Darski & Z. Vetulani, Eds., 543-548.

Rijsbergen, C. J. v., Harper, D. J., & Porter, H. F. (1981). The selection of good serach terms. *Information Processing and Management 17*, 77-91.

Ruge, G. (1992). Experiments on linguistically-based term associations. *Information Processing and Management 28*(3), 317-332.

Salton, G., & Buckley, C. (1988). On the use of spreading activation methods in automatic information retrieval. In *Proceedings of the eleventh Annual International Conference on Research and Development in Information Retrieval* (1988), ACM, 147-160.

Saracevic, T., & Kantor, P. (1988). A study of information seeking and retrieving. III searchers, searches and overlap. *Journal of the American Society for Information Science 3*(39), 197–216.

Turtle, H., & Croft, W. B. (1990). Inference networks for document retrieval. In *Proceedings of the 13th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (1990), 1-24.

Turtle, H., & Croft, W. B. (1991). Efficient probabilistic inference for text retrieval. In *Proceedings of the RIAO'91* (1991), ACM, 644-661.

Van Rijsbergen, C. J. (1986). A non-classical logic for information retrieval. *The Computer Journal 29*(6), 481-485.

Wettler, M., Rapp, R., & Ferber, R. (1993). Freie Assoziationen und Kontiguitäten von Wörtern in Texten. *Zeitschrift für Psychologie 201*, 99-108.

# Stichwortverzeichnis


A

Abdeckung . . . . . 57, 69
abduktives Lernen . . . . . 44
adding condition rule . . . . . 74
Ähnlichkeitsfunktion . . . . . 25
Ähnlichkeitsschranke . . . . . 36
allgemeiner Thesaurus . . . . . 23
Ambiguität . . . . . 20
Anfragevektor . . . . . 25
AQ . . . . . 69
AQ15 . . . . . 69
Assoziationstheorie . . . . . 104
Assoziative Regel . . . . . 80
assoziative Regel . . . . . 80
atomare Bedingung . . . . . 59
Attribut . . . . . 12
Attribut-Wert-Paare . . . . . 12

B

background knowledge . . . . . 75
Basis . . . . . 80
Bayes'sche Formel . . . . . 92
Bayes'sche Inferenznetz . . . . . 101
Beam-Search . . . . . 68
bedingte Wahrscheinlichkeit . . . . . 92
Bedingung . . . . . 59, 92
beschränkter Stern . . . . . 71
Bottom-up Methode . . . . . 65

C

characteristic rules . . . . . 77
charakteristische Regeln . . . . . 77
classification accuracy . . . . . 57
closing interval rule . . . . . 74
Clusterverfahren . . . . . 48
computerlinguistische Ansatz . . . . . 16
concept . . . . . 76
confidence threshold . . . . . 80
constraint . . . . . 71
Cosinusmaß . . . . . 107
Cover . . . . . 69
Coverage . . . . . 57

D

Darmstadt Indexing Approach . . . . . 97
Darmstädter Indexierungs Ansatz . . . . . 97
Data Mining . . . . . 48
Data Mining (DM) . . . . . 5
Data Mining Algorithmus . . . . . 48
decision list . . . . . 62
deduktives Lernen . . . . . 44
DIA . . . . . 97
Dice-Maß . . . . . 108
discriminant rules . . . . . 77
Disjunktion . . . . . 59
Document Network . . . . . 101
Document Test Collections . . . . . 39
Document-Source-Methode . . . . . 39
Dokument . . . . . 10
Dokumentennetz . . . . . 101
Dokumentvektor . . . . . 25
dropping condition rule . . . . . 74
Durchschnitt . . . . . 85

E

Effektivität . . . . . 33
Effizienz . . . . . 33
Einflußfaktoren . . . . . 33
Einschränkung . . . . . 71
elementare Bedingung . . . . . 59
elementare Ereignisse . . . . . 91
Elementarereignis . . . . . 91
Entscheidungsliste . . . . . 62
Entwicklung . . . . . 42
Ereignis . . . . . 91
Erfahrung . . . . . 42
Evaluationsverfahren . . . . . 32
extending reference rule . . . . . 74
externes Wissen . . . . . 75

F

Faktendatenbanken . . . . . 46
Feedbackmethoden . . . . . 30
Fenster . . . . . 56
Fenstertechnik . . . . . 56
formale Grundform . . . . . 17
Frageerweiterung . . . . . 39
Frames . . . . . 21
fuzzy set . . . . . 82
Fuzzy Sets . . . . . 69

G

Gedächtnis . . . . . 42
Genauigkeit . . . . . 35
Generalisierung . . . . . 43, 65
Grad . . . . . 80
Grad der Zugehörigkeit . . . . . 82
Grundbereich . . . . . 82
Grundformreduktion . . . . . 16
Grundraum . . . . . 91

H

Hamming Distance . . . . . 70
Hill Climbing . . . . . 66
Hintergrundwissen . . . . . 75
Höhe . . . . . 84

I

IDF . . . . . 32
Imaging . . . . . 99
induktives Lernen . . . . . 44
inkonsistent . . . . . 55
Inkonsistente Trainingssets . . . . . 55
inkrementell lernen . . . . . 57
Inkrementelles Lernen . . . . . 57
Intervallskala . . . . . 54
Intervallskalenniveau . . . . . 54
Inverted Document Frequency . . . . . 32
invertierten Liste . . . . . 13
IR . . . . . 4

K

Kategorisierung . . . . . 47
KDD . . . . . 5, 46
kennen . . . . . 42
Kern . . . . . 83
KI . . . . . 43
Klassifikationen . . . . . 21
Knowledge Discovery in Databases (KDD) . 5
Komplement . . . . . 59
komplementäres Ereignis . . . . . 91
Komplex . . . . . 69
Konjunktion . . . . . 59
konjunktiver Normalform . . . . . 61
können . . . . . 42
konsistent . . . . . 51
konsistente Beispielsammlung . . . . . 52
Konsistenz . . . . . 52
kontrolliertes Vokabular . . . . . 13, 23
Konzept . . . . . 76
Konzepthierarchie . . . . . 76
Kookurrenzdaten . . . . . 104
Kookurrenzmodelle . . . . . 106
Kookurrenzverfahren . . . . . 106
Korpusbasierte Verfahren . . . . . 104
künstliche Intelligenz . . . . . 43

L

Latent Semantic Indexing . . . . . 112
leere unscharfe Menge . . . . . 83
Lemmatisierungen . . . . . 16
Lernen . . . . . 42
Lernenforschung . . . . . 42
lexikographische Grundform . . . . . 17
linked dependency assumption . . . . . 94

M

Machine Learning (ML) . . . . . 5
Makrobewertung . . . . . 38
maximalem Element . . . . . 75
membership function . . . . . 82
Mikrobewertung . . . . . 38
mögliche Welt . . . . . 98

N

Nachrichtenagentur . . . . . 106
Nichtüberwachtes Lernen . . . . . 48
Nominalskala . . . . . 54
nutzerorientiert . . . . . 38

O

Oberflächenform . . . . . 104
Oberflächenstruktur . . . . . 21
Odds . . . . . 93
Ordinalskala . . . . . 54

P

partieller Stern . . . . . 71
Polysemie . . . . . 20
possible World . . . . . 98
Präzision . . . . . 35
Precision . . . . . 35
Precision-Recall-Diagramm . . . . . 36
predicted attributes . . . . . 48
predicting attributes . . . . . 48
preprocessing . . . . . 55
probabilistische Inferenz . . . . . 98
Produktionsregel . . . . . 60
Psychologie . . . . . 42

Q

Quantitative Regeln . . . . . 77
quantitative rules . . . . . 77
Query Expansion . . . . . 107
Queryvektor . . . . . 25
Quote . . . . . 93

R

Rangreihe . . . . . 36
Rationale Skala . . . . . 54
Recall . . . . . 35
Regelextraktion . . . . . 5
Reifung . . . . . 42
Relationen . . . . . 87
relevance description . . . . . 97

Relevance Feedback . . . 95
Relevanz . . . 34
Relevanzbeschreibung . . . 97
Repräsentationen . . . 10
repräsentative Stichporbe . . . 38
retrieval status value . . . 95
Retrievalstatuswert . . . 95
Ripple-down Regelmenge . . . 63

S

scharfe Menge . . . 82
Schnitt . . . 83
Schwelle . . . 26, 36
Schwellenfunktion . . . 26
Schwellenwert . . . 6
Schwellwertfunktion . . . 55
Selektor . . . 69
semantische Netze . . . 21
Skalenniveaus . . . 53
Skripts . . . 21
SMART . . . 32
Stammform nach linguistischen Prinzipien . 17
Stammformreduktion . . . 16
Stern . . . 71
supervised learning . . . 48
support . . . 83
support threshold . . . 80
Synonymie . . . 21
system oriented . . . 38
systemorientiert . . . 38

T

Tanimoto-Maß . . . 108
teilgeordnete Menge . . . 75
Teilordnung . . . 75
Template . . . 82
Term . . . 12, 14
Term-Dokument-Matrix . . . 106
Term-Term-Matrix . . . 107
Term-Term-Matrizen . . . 106
Testkollektionen . . . 39
Testset . . . 47
Textdokumente . . . 12
themenspezifischer assoziativer Thesaurus . . . 107
Thesauri in IR-Systemen . . . 23
Top-down Methode . . . 65
Träger . . . 83
Trainingsset . . . 47
Tupel . . . 46
turning conjunction into disjunction rule . . 74

U

überwachtes Lernen . . . 48
unabhängig . . . 92
Unabhängigkeit . . . 92
undefined . . . 55
unscharfe Menge . . . 82
unscharfen Schließen . . . 87
Unterscheidungsregeln . . . 77
Unvollständige Beispiele . . . 55
user-oriented . . . 38

V

Vereinigung . . . 85
Verfugung . . . 18
Vergleichbarkeit von Inhalten . . . 16
Verhaltensdispositionen . . . 42
vollständige Suche . . . 68
Vollständigkeit . . . 35
Volltextdatenbanken . . . 10
Vorbehandlung . . . 55
vorhersagende Attribute . . . 48
vorherzusagende Attribute . . . 48

W

Wahrscheinlichkeitsfunktion . . . 91
Wahrscheinlichkeitsraum . . . 91
Wertebereich . . . 55
wissen . . . 42
Wissensextraktion . . . 5

Z

Zerlegung . . . 47
Zufallsvariable . . . 91
Zugehörigkeitsfunktion . . . 82
zugelassenen Deskriptoren . . . 23
zusammengesetzte Bedingung . . . 59