Algorithmenauswahl im KDD–Prozess

Zur Erlangung des akademischen Grades eines

Doktors der Wirtschaftswissenschaften

(Dr. rer. pol.)

von der Fakultät für
Wirtschaftswissenschaften
der Universität Fridericiana zu Karlsruhe

genehmigte

DISSERTATION

von

Diplom–Informatiker Guido Lindner

Tag der mündlichen Prüfung: 14.01.2005
Referent: Prof. Dr. R. Studer
Korreferenten: Prof. Dr. G. Nakhaeizadeh

Karlsruhe 2005

Meiner Frau Kerstin und unseren Kindern

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Teil I

Der KDD–Prozess: Theorie und Praxis

Kapitel 1

Einleitung

Die ständige Leistungssteigerung der Informationstechnologien in der Vergangenheit ermöglichte eine breite, effiziente und oft auch automatisierte Speicherung von Daten in allen Bereichen der Industrie und der öffentlichen Dienste. Schon Anfang der 90'er schätzte man, dass sich die weltweit vorhandene Datenmenge alle 20 Monate verdoppelt [Frawley u. a. 1991]. Heute kann man davon ausgehen, dass sich die Zeitspanne bis zur Verdopplung der Datenmenge eher verkleinert hat. Die meisten der enormen Datenmengen mit Millionen oder Milliarden von Informationen wurden ursprünglich für andere Aufgaben als die Verwendung für „Knowledge Discovery in Databases"(KDD) gesammelt. Heute hält dieses Wachstum an und scheint fast außer Kontrolle zu sein [Fayyad 1999]. Diese Datenbanken werden heute herangezogen, um sie nach neuen Mustern und Informationen zu durchsuchen.

Im kommerziellen Bereich geht man heute davon aus, dass solche Analysen entscheidende Wettbewerbsvorteile sichern. Allgemein wird diese Analyse von großen Datenmengen als iterativer Prozeß angesehen, in dem Mensch und Maschine ihre jeweiligen Stärken einbringen [Nakhaeizadeh u. a. 1998, Fayyad u. a. 1996, Brachman und Anand 1996]. Der Nobelpreisträger und früherer Chef der Bell Labs, Arno Penzias erklärte 1999 in einem Interview der Computerwoche[1] auf die Frage:

CW: What will be the killer apps in the corporation?

Penzias: Data Mining will become much more important. Your bank will know everything you've bought. Companies will throw away nothing they know about their customers because it will be so valuable. If you're not doing this, you're out of business.

Mittlerweile haben viele Unternehmen die Bedeutung und Notwendigkeit erkannt, ihre über die Jahre gesammelten Daten zu analysieren und den KDD–Prozess für ihre Anwendungen zu starten. Die verschiedenen Definitionen des Prozessmodelles für KDD werden im folgenden Kapitel 2 genauer erläutert.

[1] Das vollständige Interview ist unter http://www.computerworld.com/news/1999/story/0,11280,33391,00.html verfügbar.

Oft zitierte Beispiele für Anwendungen sind die Analyse von Supermarktdaten, Scannerdaten, Prozessdaten aus produktiven Systemen oder die Beurteilung von Kunden. Neben solchen kommerziellen Anwendungen gibt es auch wissenschaftliche Sensordaten, die in großen Datenbanken gespeichert werden und ohne maschinelle Auswertungen nicht mehr analysierbar wären.

Aktuell sieht die Gartner Group den Trend, dass Data Mining Verfahren in Anwendungen, wie zum Beispiel *Customer relationship management* (CRM) integriert werden. Nach der Einschätzung von Gartner macht sich zur Zeit eine gewisse Ernüchterung breit, da realisiert wird, dass KDD noch recht schwer umsetzbar ist und der Wissensentdeckungsprozess nicht einfach per Knopfdruck realisierbar ist [A. Linden 2002].

Für Gartner gibt es aktuell zwei Entwicklungen. Die meisten Unternehmen haben noch nicht verstanden, wo und ob Data Mining hilfreich ist oder nicht. Diese Unternehmen haben typischerweise nur beschränkt Daten zur Verfügung und haben somit auch keinen dringenden Bedarf Data Mining Verfahren einzusetzen. Diese Unternehmen kommen in der Regel mit einfacher *Business Intelligence* (BI) Software wie zum Beispiel *online analytical processing* (OLAP) aus.

Größere und fortschrittlichere Unternehmen in datenintensiven Industrien, wie die Telekommunikation, Versicherungen und Banken oder der Einzelhandel, haben viel mehr Möglichkeiten KDD einzusetzen. Hier wird auch der Bedarf an dieser Technologie erkannt.

Die vorliegende Arbeit unterteilt sich in drei Teile.

Teil I: Der erste Teil behandelt neben theoretischen Grundlagen KDD Anwendungen in der Praxis. Hier werden Anwendungen im Bereich des Qualitätsmanagements bei einem Automobilhersteller vorgestellt. Anhand der Erfahrungen aus dieser praktischen Arbeit und den anderen publizierten Arbeiten werden die verschiedenen Notwendigkeiten für eine Benutzerunterstützung für KDD innerhalb des Prozesses hergeleitet.

Teil II: Im zweiten Teil der Arbeit wird ein Ansatz entwickelt den KDD–Prozess in der Phase der Algorithmenselektion mittels einer Datencharakterisierung zu unterstützen. Algorithmenselektion ist nicht nur im Rahmen von KDD interessant. Im Maschinellen Lernen ist die Unterstützung bei der Auswahl von Algorithmen aktueller Forschungsschwerpunkt. Die erste relevante Arbeit war das STATLOG Projekt [Michie u. a. 1994]. Die Bedeutung des Themas wird weiterhin durch Workshops auf internationalen Konferenzen wie der ECML 1998 [Giraud-Carrier und Hilario 1998] und ICML 1999 [Giraud-Carrier und Pfahringer 1999] deutlich. Weiterhin ist die Unterstützung des Anwenders Forschungsthema des ESPRIT Projekt METAL(1999-2002).

Teil III: Der dritte Teil dieser Arbeit beschreibt dann einen CBR–Ansatz mit dem die Algorithmenauswahl im Rahmen des KDD–Prozesses unterstützt werden

kann. Hierzu wurde initial eine erste Fallbasis aufgebaut und auch der ML-Community zur Verfügung gestellt. Das Daten Charakeristik Tool aus Kapitel 6 wurde dem ESPRIT Projekt METAL zur Verfügung gestellt und kann unter der URL: http://www.metal-kdd.org bezogen werden.

Kapitel 2

KDD: Ein iterativer Prozess

2.1 Warum KDD?

„Knowledge Discovery in Databases“ (KDD) ist ein relativ junges Forschungsgebiet. Es zieht seine Motivation aus zwei wesentlichen Entwicklungen: Auf der einen Seite sind die Preise für Massendatenspeicherung in der Vergangenheit ins Bodenlose gefallen. Auch im Heimcomputer–Bereich sind heute Festplattengrößen Standard, die sich vor wenigen Jahren schlicht utopisch anhörten. Die durchschnittlichen Größen haben sich in den letzten 4 Jahren verzehnt- bis verzwanzigfacht. Was für die privat genutzten Computer gilt, gilt in entsprechendem Maßstab für die Datenhaltung im kommerziellen Bereich. Auf der anderen Seite liegen mehr und mehr Daten in digitaler Form vor. Zunehmend werden alltägliche Vorgänge durch oder unter Zuhilfenahme von Computern abgewickelt. Dadurch ist nahezu kein zusätzlicher Aufwand mit der zusätzlichen Speicherung der Daten solcher Transaktionen verbunden. Erwähnt seien hier etwa die Daten von Scannerkassen in Supermärkten oder die Speicherung von Telefongesprächsdaten im Mobilfunknetz, bei denen täglich Daten im Bereich von etlichen Mega- bis Gigabyte anfallen. Neue Medien wie das Internet sind weitere Quellen vielfältiger Daten. Es ist also für ein Unternehmen vergleichsweise einfach, gigantische Massen von Daten aufzuzeichnen und zu speichern. Große Hoffnung wird darauf gesetzt, dass wichtiges, geldwertes Wissen in den Daten schlummert, welches nur darauf wartet, entdeckt zu werden.

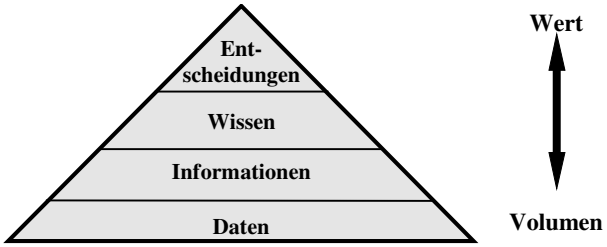

Abbildung 2.1: Verhältnis Volumen zu Nutzen

In den folgenden Abschnitten werden die Fragen, was ist KDD und was ist Data Mining, beantwortet. Hierzu werden verschiedene Definitionen aus dem noch recht jungen Forschungsgebiet vorgestellt. Die ersten Definitionen wurden Anfang der 90'ziger von [Frawley u. a. 1991] veröffentlicht.

Definition 2.1 (*knowledge discovery in databases)*
Gegeben seien eine Sprache $\mathcal{L}$, eine Menge von Fakten $\mathcal{F} \subseteq \mathcal{L}$, und ein Kriterium $\mathcal{C}$. KDD ist das Problem, eine Aussage $\mathcal{S} \in \mathcal{L}$ zu finden, so dass $\mathcal{S}$ eine Faktenmenge $\mathcal{F}_\mathcal{S} \subseteq \mathcal{F}$ beschreibt, $\mathcal{S} \neq \mathcal{F}_\mathcal{S}$, und $\mathcal{S}$ dem Kriterium $\mathcal{C}$ genügt.

Diese frühe Definition von KDD machte die Nähe und Ähnlichkeit zu den Zielsetzungen im maschinellen Lernen deutlich. Auch im maschinellem Lernen geht man von einer Sprache $\mathcal{L}_H$ zur Beschreibung der Hypothese und einer Menge von Beispielen (Fakten) aus. Das Ergebnis eines maschinellen Lernvorganges ist eine Hypothese, die den Kriterien des Benutzers genügt. Diese Hypothese entspricht der Aussage $\mathcal{S}$ aus der Definition von KDD. In diesem Sinne bezeichnen Matheus, Chan und Piatetsky–Shapiro KDD als eine Teilmenge des maschinellen Lernens [Matheus u. a. 1993].

Definition 2.2 (Lernen aus Beispielen)
Lernen aus Beispielen ist die Suche nach einer Beschreibung eines Begriffs $\mathcal{C}$, zu dem man eine endliche Anzahl an Beispielen gegeben hat.

Gegeben sei eine Menge von Beispielen in einer Sprache $\mathcal{L}_B$, eine Sprache $\mathcal{L}_H$ zur Beschreibung des zu lernenden Begriffs (Hypothesensprache) und ein Kriterium zur Akzeptanz einer Hypothese. Eine Aussage $H \in \mathcal{L}_H$, die dem Akzeptanzkriterium genügt, ist eine Beschreibung des Begriffs $\mathcal{C}$.

Aber nicht nur zum maschinellen Lernen gibt es im KDD enge Verknüpfungen. Maschinelles Lernen ist nur eine Technik Wissen oder Informationen aus Daten zu gewinnen. Zu den in KDD eingesetzten Techniken gehören genauso die klassische Statistik und auch Datenbanktechniken, ohne welche die Datenmengen nicht handhabbar wären, spielen im KDD eine grosse Rolle. In Abbildung 2.2 sind die wichtigsten Gebiete aufgeführt, in denen KDD Techniken benutzt werden. Aus diesem Grund gibt es auch die Meinung, dass KDD nur eine interdisziplinäre Anwendung der schon bekannten Forschungsgebiete ist und kein eigenes Forschungsgebiet darstellt. Im Gegensatz zu dieser Meinung erläutert David Hand in seinem Artikel [Hand 1999] zum Beispiel die unterschiedlichen Zielsetzungen von KDD und Statistik in der Anwendung und Nutzung der Verfahren.

Im Laufe der Zeit entwickelte sich zunehmend die Ansicht, dass KDD ein Prozess ist, der sich in verschiedene Phasen unterteilt und die bisherige Definition nur einen Teil des Prozesses abdeckt. Dieser Prozessgedanke macht auch deutlich, dass KDD viele Techniken aus anderen Forschungsgebieten nutzt.

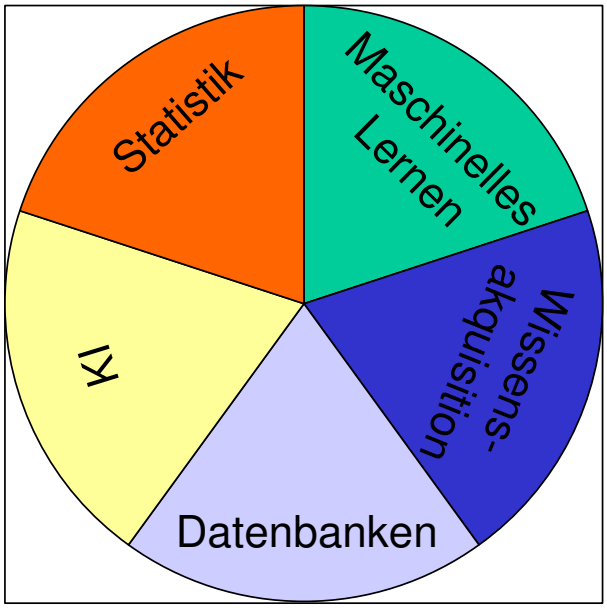

Abbildung 2.2: Gebiete im KDD

2.2 KDD als Prozess

Mitte der neunziger Jahre wurden dann die ersten Definitionen, die KDD als Prozess definierten, veröffentlicht. Im Laufe der Entwicklung dieses neuen Forschungsgebietes KDD entwickelte sich die Definition von KDD als Prozess, der zur Zeit eine Standardisierung durchläuft. Die erste anerkannte Definition für KDD, die den Prozessgedanken beinhaltete, wurde 1995 [Fayyad u. a. 1996] gemacht.

Definition 2.3 (*Knowledge Discovery in Databases*) *Knowledge Discovery in Databases (KDD) is the non–trivial process of identifying vailid, novel, potentially useful and ultimately understandable patterns in data.*

[Fayyad u. a. 1996] grenzt den Begriff Data Mining auch vom KDD–Prozess ab, während in deutschen Veröffentlichungen die beiden Begriffe oft synonym verwendet werden. In dieser Arbeit wird der Unterscheidung von Fayyad gefolgt.

Definition 2.4 (*Data Mining*) *Data Mining ist der Schritt im KDD–Prozess, die Anwendung von Data Mining Algorithmen mit dem Ziel eine Anzahl von besonderen Mustern aus den Daten zu erhalten.*

Data Mining ist also nur ein Schritt im KDD–Prozess, der die Anwendung von Data Mining Verfahren beinhaltet. Insgesamt umfaßt der KDD–Prozess nach

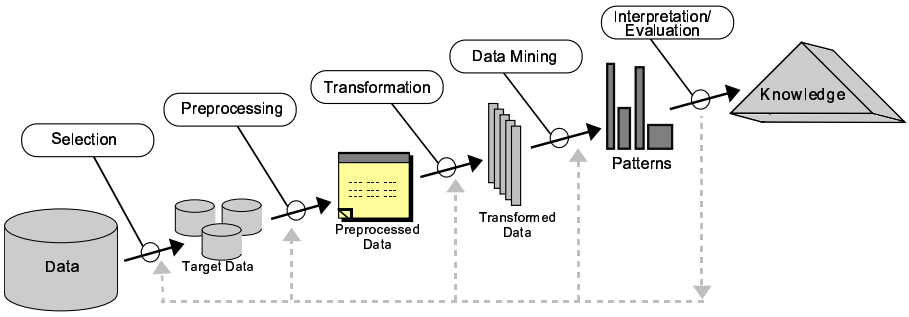

Abbildung 2.3: KDD–Prozess nach [Fayyad u. a. 1996]

[Fayyad u. a. 1996] neun Basisschritte, von denen die fünf wichtigsten in der Abbildung 2.3 dargestellt sind:

1. Anwendungsverständnis
2. Datenselektion
3. Datenbereinigung und Vorverarbeitung: z.B. Behandlung von fehlenden Werten
4. Datenreduktion und Projektion: Reduzierung der Dimension oder Transformationen zur Reduzierung der Anzahl Attribute.
5. Auswahl des Data Mining Ziels
6. Algorithmenauswahl
7. Anwendung des Data Mining Algorithmus
8. Interpretation und Bewertung der gefundenen Muster
9. Konsolidierung des entdeckten Wissens

2.3 Goal driven KDD–Prozess

Eine erste weiterführende Definition wurde von [Engels 1996] gemacht. Engels nimmt in seiner Definition die impliziten Annahmen die [Fayyad u. a. 1996] an den Prozess stellen auf.

1. Wer beurteilt, was neu und interessant ist an den Mustern, die entdeckt werden? Es muss also ein Orakel existieren, welches eine solche Bewertung durchführt.
2. Der Prozess muss mit einer Zielsetzung oder Motivation gestartet werden und bestimmt wesentlich die Schritte in den folgenden Prozessphasen.

Definition 2.5 (Goal Driven KDD–Prozess) *KDD ist der nicht triviale Prozess gültige, neue, potentiell nützliche und verständliche Muster in Daten zu identifizieren, gesteuert durch die Zielsetzung, die in der Problembeschreibung während der Anwendungsanalyse durch den Anwender bestimmt wird und die Bewertung der gefundenen Muster.*

Aus dieser Definition wird deutlich, dass es sich bei dem KDD–Prozess um einen iterativen Prozess handelt, der durch die Bewertungen des Anwendungsbesitzers (application owner) neu initiiert werden kann. [Brachman und Anand 1996] erklären deutlich die Bedeutung des Anwenders und des Data Mining Ingenieurs, die den Prozess steuern und dass ein Rückschritt zu vorherigen Phasen des Prozesses jeder Zeit möglich ist.

[Engels 1999] unterteilt den Prozess in fünf Phasen:

Problemanalyse: In dieser Phase wird das Anwendungsproblem (*business problem*) definiert. Welche Fragen möchte man beantworten? Welche Anforderungen werden an die Lösung gestellt?

Aufgabenanalyse: Während der Aufgabenanalyse wird eine Aufgabe definiert, die das in der Problemanalyse–Phase beschriebene Problem lösen kann. Diese definierte Aufgabe wird dann weiter unterteilt (verfeinert = *refinement*), wobei typischerweise ein erster Verfeinerungsschritt zu den zwei Phasen Modell–Entwicklung und Wissensgenerierung führt.

Entwicklung: Die Ergebnisse der Aufgabenanalyse–Phase bilden die Basis für die Definition der Entwicklungs–Phase. Das Ziel der Entwicklungs–Phase ist, das Geschäftsproblem zu lösen, welches in Zusammenarbeit mit dem Anwender beschrieben wurde.

Wissensgenerierung: Sind die Ziele der Anwendung und der Entwicklungs–Phase definiert, so ist es möglich den Wissensgenerierungsprozess zu definieren, der spezielle Informationen (Wissen), welche die Daten oder Beziehungen innerhalb dieser Daten beschreiben, als Ergebnis hat.

Dokumentation: Während der gesamten Anwendung, also der Definition des Problems, Finden einer Aufgabe, Zerlegen der Aufgabe ... ist es wichtig zu dokumentieren welche Entscheidungen zu welchen Lösungen geführt haben.

Wesentlicher Unterschied zu bisherigen Definitionen des KDD–Prozesses wird aus der Phasenaufteilung deutlich:
Der Prozess kann durchaus auch ein rekursives Verhalten aufweisen. Teilaufgaben aus der Aufgabenanalyse können wiederum ein eigenes KDD–Problem beschreiben [Engels u. a. 1997]. Ein weiteres KDD–Problem ergibt sich meistens aus der Datenaufbereitung. Ein Beispiel hierfür stellen wir in Kapitel 3.9 vor.

2.4 Das CRISP Modell

Das zur Zeit am meisten zitierte Prozessmodell geht aus einem von der Europäischen Union geförderten Projekt *Cross Industrie Standard Process (CRISP) for Data Mining* hervor [Chapman u. a. 1999]. Dieses Projekt wurde von einem Konsortium von Firmen aus verschiedenen Wirtschaftszweigen durchgeführt. Jeder der einzelnen Partner ist ein Branchenführer in seinem jeweiligen Industriezweig.

- NCR, Anbieter von Data Warehouse Lösungen,
- DaimlerChrysler, als eines der grössten europäischen Unternehmen,
- SPSS, Hersteller eines der marktführenden KDD–Tools und
- OHRA, eine der grössten niederländischen Versicherungen.

Ziel dieses Projektes war es, den Prozess zu standardisieren. Mittlerweile gibt es hierzu auch eine *special intrest group* (SIG CRISP), in der sich zahlreiche Firmen aus aller Welt registriert haben und die jährlich einen Workshop zum Erfahrungsaustausch veranstaltet. Dieses dokumentiert anschaulich die Akzeptanz und die Wichtigkeit der Thematik.

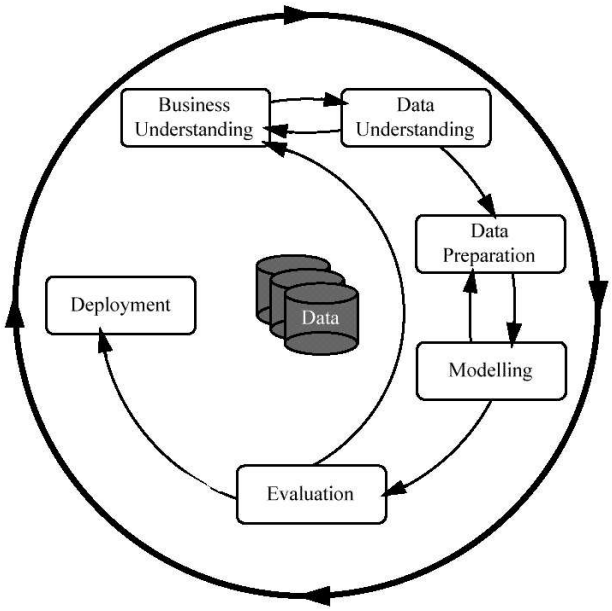

Abbildung 2.4: Phasen im CRISP Reference Model [Chapman u. a. 1999]

In Abbildung 2.4 sind die Phasen des CRISP–Modells aufgezeigt. Die Phasenaufteilung des KDD–Prozesses nach der CRISP Methodik unterscheidet sich nicht grundsätzlich von den Prozessdefinitionen nach [Fayyad u. a. 1996] oder [Engels 1996]. Die CRISP Methode beschreibt nicht nur die Phasen des Prozesses, sondern auch

verfeinerte Schritte. Der gesamte Prozess wird in CRISP durch ein vierstufiges hierarchisches Prozessmodell definiert. Die Idee der Zerlegung des Prozesses wurde auch in [Engels u. a. 1997] und [Engels 1999] vorgestellt.

1. Phasen
2. Generische Aufgaben
3. Spezifizierte Aufgaben
4. Instanziierte Prozessebene

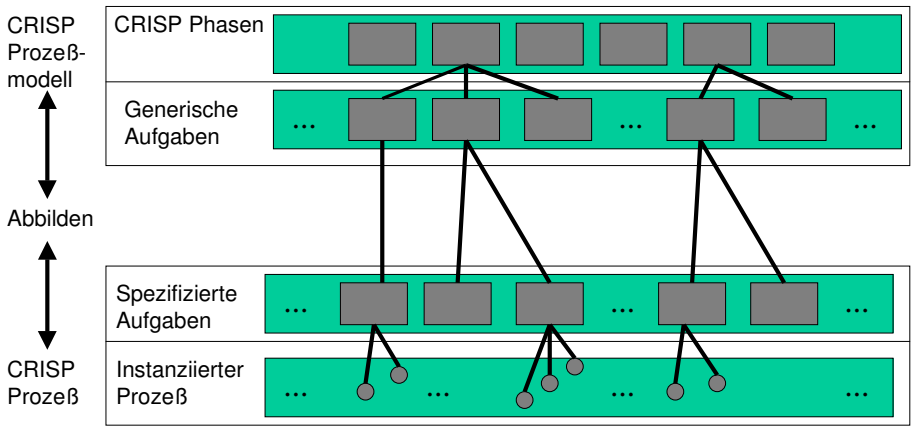

Abbildung 2.5: Vier Stufen Unterteilung der CRISP–Methode

In der ersten Ebene ist der KDD–Prozess in die schon genannten Phasen unterteilt. Jede Phase enthält auf der zweiten Ebene verschiedene *generische Aufgaben*, die allgemein genug definiert sind, um alle KDD–Situationen abzudecken. Die generische Ebene soll möglichst komplett den Prozess und alle möglichen KDD–Anwendungen abdecken. Die dritte Ebene, die Ebene *spezifizierte Aufgaben*, beschreibt wie die generischen Aufgaben in den jeweiligen Situationen ausgeführt werden sollen. So kann die generische Aufgabe Datenbereinigung (clean data) als spezifizierte Aufgabe der Bereinigung von fehlenden Werten (missing values) bei numerischen oder symbolischen Werten angesehen werden.

Die letzte Ebene der Prozessmethodik ist die Instanziierung des Prozesses mit Aktionen, Entscheidungen und Ergebnissen einer KDD–Anwendung.

In den folgenden Abschnitten werden die Phasen und generischen Aufgaben des Prozesses näher erläutert. Die anderen Ebenen des Prozessmodells werden von den Autoren auch nicht weiter spezifiziert, da die Vielfalt der Möglichkeiten speziell auf der untersten Ebene zu gross ist.

2.4.1 Phasen in CRISP

Die Phasen in CRISP beschreiben den Prozess auf einer sehr abstrakten Ebene. Die einzelnen Phasen werden wie folgt unterteilt:

Geschäftsverständnis (*Business Understanding*): Diese erste Phase hat zur Zielsetzung, die Projektinteressen und Anforderungen aus Sicht des Geschäftsbereichs zu verstehen und dieses in eine KDD–Aufgabendefinition zu überführen. Ebenso sollte eine erste grobe Vorgehensweise entwickelt werden.

Datenverständnis (*Data Understanding*): Die Datenverständnisphase hat zum Ziel mit den vorhandenen Daten vertraut zu werden, um erste Aussagen über die Datenqualität machen zu können. In dieser Phase beschränkt man sich meist auf das Datenmodell und einen ausgewählten Ausschnitt der zur Verfügung stehenden Daten. In dieser Phase können aber schon erste Entdeckungen oder Hypothesen über verborgenes Wissen aufgestellt werden.

Datenaufbereitung (*Data Preparation*): Die Phase der Datenaufbereitung beinhaltet alle Aktivitäten, die durchgeführt werden, um aus den gegebenen *Rohdaten* die Daten für die Wissensgenerierungsphase zu generieren. Innerhalb des Prozesses ist diese Phase oft ein Wiedereinstieg in einen neuen Iterationsschritt.

Wissensgenerierung (*Modelling*): In der Wissensgenerierungsphase werden die Verfahren ausgewählt und angewendet. Hierbei werden die Verfahren nicht nur einmal angewendet, sondern mehrmals, um die optimalen Parameter der Verfahren für das Problem zu ermitteln.

Ergebnisevaluierung (*Evaluation*): In dieser Phase werden die Ergebnisse evaluiert, die aus Sicht des Datenanalysten gute Ergebnisse darstellen. Bevor das Ergebnis in der Anwendung Verwendung findet, muss eine Überprüfung des Ergebnisses und des Entstehungsweges stattfinden, ob alle wichtigen Geschäftseinflüsse ausreichend berücksichtigt wurden.

Einsatz (*Deployment*): Der Einsatz des erzielten Ergebnisses kann von unterschiedlicher Komplexität sein und von der einfachen Berichterstattung bis zur Implementierung eines wiederholbaren KDD–Prozesses innerhalb eines Geschäftsvorgangs gehen.

Nach den ersten zwei Phasen sollte eine Beurteilung des geplanten Projektes durchgeführt werden. Der KDD–Prozess sollte nur weitergeführt werden, wenn auch ein Potential gesehen wird, dass die Projektinteressen und Anforderungen mit den gegebenen Daten lösbar erscheinen. Diese Beurteilung sollte von den KDD–Experten durchgeführt werden. Weiterhin können alle Phasen innerhalb des Prozesses mehrfach durchlaufen werden.

2.4.2 Generische Aufgaben

An dieser Stelle werden die einzelnen generischen Aufgaben aus CRISP den Phasen zugeordnet, aber nicht im Detail erklärt. Eine detaillierte Beschreibung der generischen Aufgaben findet man in [Chapman u. a. 1999] beschrieben. Im folgenden werden für die Wissensgenerierungsphase die generischen Aufgaben vorgestellt, da diese Phase im zweiten Teil der Arbeit im Mittelpunkt steht.

Auswahl der Wissensgenerierungstechnik: In diesem Schritt des Prozesses wird ein spezieller Algorithmus für die in der Geschäftsverständnisphase identifizierte KDD–Aufgabe ausgewählt.

Test Design: Bevor der ausgewählte Algorithmus angewendet werden kann, benötigt man einen Mechanismus um die Qualität und Gültigkeit der Ergebnisse zu testen. Für die KDD–Aufgabe, überwachte Klassifikation, wird im Allgemeinen die Fehlerrate der Klassifikation (*error rate*) als Qualitätsmaß benutzt. Hierzu muss die gegebene Datenmenge in eine Trainings- und eine Testmenge aufgeteilt werden. Das ausgewählte Verfahren wird dann auf der Trainingsmenge angewendet und das Ergebnis an der Testmenge evaluiert.

Wissensgenerierung: Anwendung des ausgewählten Verfahrens.

Bewertung des generierten Ergebnisses: In diesem Schritt findet eine erste Bewertung und Interpretation der erzielten Ergebnisse statt. Die Bewertung berücksichtigt soweit möglich schon die Geschäftsziele. Der Schwerpunkt liegt aber auf der technischen Beurteilung der Ergebnisse. Es wird versucht eine erste Rangfolge über die erzielten Ergebnisse aufzustellen, um diese in der Evaluierungsphase mit den Auftraggebern zu besprechen. Ergebnis dieser ersten Bewertung kann aber auch sein, ein Verfahren mit geänderten Parametern noch einmal anzuwenden oder die Wissensgenerierungsphase mit einem anderen Verfahren noch einmal durchzuführen.

2.4.3 Spezifizierte Aufgaben und Methoden–Instanziierung

Um für eine spezifizierte Aufgabe eine geeignete Methode auszuwählen, gibt es in der Regel drei Restriktionen.

1. Methodenverfügbarkeit: Nicht alle möglichen Methoden/Techniken stehen zur Verfügung.

2. Politische Einschränkungen: Aufgrund von Management Entscheidungen stehen nur bestimmte Verfahren zur Verfügung.

3. Sachzwänge: Die Daten lassen eine Anwendung von bestimmten Verfahren nicht zu oder das Wissen über die Methoden ist nicht ausreichend vorhanden.

2.5 Akteure im KDD–Prozess

In der Beschreibung des Prozesses sind schon einige Rollen genannt worden. Eine Person kann hierbei durchaus mehrere Rollen wahrnehmen. Die Rollen definieren sich aus dem Wissen und den Fähigkeiten, dass an den verschiedenen Stellen im Prozess benötigt wird. Die Tabelle 2.1 zeigt, in welchen Phasen welche Akteure benötigt werden.

Die zentrale Person in dem ganzen Prozess ist der Anwender (*Domain Expert, Application Owner*). Der Anwender besitzt das Anwendungswissen, aber in der Regel hat er keine Kenntnisse über den KDD–Prozess und die Data Mining Verfahren. Im Allgemeinen besteht zwischen dem Anwender und allen anderen Akteuren eine Kundenbeziehung, in der der Anwender Kunde ist und auch für die Leistung der anderen Akteure bezahlt. Neben den Anwendern gibt es im KDD–Prozess zwei weitere Akteure:

- Der Geschäftsanalyst (*business analyst*) analysiert mit dem Anwender seine Geschäftsinteressen und die Einsatzmöglichkeiten für KDD. Der Geschäftsanalyst tritt eigentlich nur in den ersten Phasen des Prozesses in Aktion, um die KDD–Ziele herauszuarbeiten.

- Der Datenanalyst (*data analyst*) ist der Experte, der sich mit der KDD Technologie auskennt und die eigentliche Anwendung der Verfahren zur Wissengenerierung durchführt. Das heißt, der Datenanalyst führt in Interaktion mit dem Anwender die Datenaufbereitung, Wissensgenerierung und Ergebnisevaluierung durch.

In der Praxis werden die Rollen des Geschäftsanalysten und des Datenanalysten meist von einer Person wahrgenommen, während die Einbindung des Auftraggebers/Anwenders oft vernachlässigt wird. Dies ist ein Problem, wie es auch im Softwareentwicklungsprozess beschrieben wird [Oestereich u. a. 1999].

Phase	Geschäftsanalyst	Datenanalyst	Anwender
Geschäftsverständnis	X		X
Datenverständnis		X	X
Datenaufbereitung		X	X
Wissensgenerierung		X	X
Ergebnisvalidierung		X	X
Einsatz		X	X

Tabelle 2.1: Akteure im KDD–Prozess

2.6 KDD–Aufgaben

Bisher wurde der KDD–Prozess in den Vordergrund gestellt, was aber sind die KDD–Probleme oder Aufgaben? [Fayyad u. a. 1996] und [Chapman u. a. 1999] identifizieren folgende Aufgabentypen:

Klassifikation : Klassifikation[1] setzt voraus, dass eine Menge von Objekten existiert (Klassen), die durch Attribute aus der Datenmenge beschrieben werden und zur Unterscheidung zwischen den einzelnen Klassen beitragen können. Die Bezeichner der Klassen sind symbolische[2] Werte. Die Klassenzugehörigkeit ist für jeden Datensatz aus der Datenmenge bekannt. Gesucht ist dann eine Funktion, die einen Datensatz einer Klasse zuweist. In der Statistik nennt man diesen Aufgabentyp auch überwachte Klassifikation.

Klassifikation ist eine der wichtigsten KDD Aufgaben, die in vielen praktischen Anwendungen auftaucht [Brodley und Smyth 1997]. Hinzu kommt, dass sich auch andere KDD–Aufgaben in eine Klassifikationsaufgabe überführen lassen [Nakhaeizadeh u. a. 1998]. So kann ein Vorhersageproblem in ein Klassifizierungsproblem, durch die Diskretisierung der Zielgrösse, transformiert werden. Eine Schnittmenge besteht zu den Problemen der Abhängigkeitsanalyse und Beschreibung, da viele der Klassifikationsmethoden auch Beziehungen zwischen Attributen und dem Klassenattribut beschreiben.

Vorhersage oder Regression: Wie schon bei der Klassifikation angedeutet, ist die Vorhersage sehr ähnlich zur Klassifikation. Im Gegensatz zur Klassifikation ist das Zielattribut hier kein diskreter symbolischer Wert, sondern ein numerischer kontinuierlicher Wert.

Clustering oder Segmentierung: Bei einem Clustering–Problem kennt man das Klassenattribut nicht. Ein Clustering–Problem hat das Ziel, die Daten in interessante und aussagekräftige Gruppen oder Klassen zu zerlegen. Je nach Verfahren wird die Anzahl der Klassen vorgegeben oder selbst bestimmt. In der Statistik nennt man diesen Problemtyp auch unüberwachte Klassifikation.

Beschreibung oder Zusammenfassung: Das Ziel hierbei ist, eine möglichst kurze und prägnante Beschreibung einer Teilmenge der Datenmenge zu finden. Im Vordergrund steht die Wissensakquisition. Einfaches Beispiel ist die Berechnung aller Mittelwerte und Standardabweichungen für alle Attribute. Aber auch die Anwendung von Klassifikationsmethoden, die ein interpretierbares Ergebnis liefern, kann hier eingesetzt werden.

Abhängigkeitsanalyse: Ziel der Abhängigkeitsanalyse ist es, ein Modell zu finden, das signifikante Abhängigkeiten zwischen Attributen beschreibt. Solche Abhängigkeiten können wiederum für die Vorhersage von Werten (numerisch

[1] *supervised classification*
[2] diskrete

oder symbolisch) benutzt werden. Im Allgemeinen liegt der Schwerpunkt bei diesem Problemtyp auf dem Verstehen der Zusammenhänge in den Daten.

Aus dieser Sammlung von KDD–Problemtypen wird klar, dass es keine eineindeutige Zuordnung der Verfahren zu den Problemtypen gibt. Vielmehr hängt die Auswahl der Verfahren von mehreren Faktoren ab, wie beispielsweise dem Geschäftsziel, den Daten und dem Problemtyp. Diese Problematik wird in Kapitel 5 genauer analysiert.

2.7 Zusammenfassung

In diesem Kapitel haben wir die ersten Definitionen für KDD kennengelernt, die eine hohe Übereinstimmung mit der Definition des Begriffs Lernen aus dem Bereich Maschinelles Lernen aufweist. Heute stimmt man im KDD allgemein überein, dass KDD ein Prozess ist. Die Anzahl der eingeteilten Phasen variiert in den verschiedenen Definitionen, die Phasen unterscheiden sich aber inhaltlich nur unwesentlich. Festzustellen bleibt dennoch, dass KDD sich aus den Bereichen Datenbanken, Statistik, Maschinelles Lernen, Wissensakquisition und KI zusammensetzt und aus allen Gebieten Techniken übernimmt, aber sich in den Zielsetzungen ihrer Forschung unterscheiden. So beschreibt zum Beispiel David Hand in seinem Artikel *Statistics and Data Mining: Intersecting Disciplines* im SIGKDD Exploration Journal [Hand 1999], die verbindenden und trennenden Elemente dieser beiden Disziplinen.

Das zur Zeit anerkannteste Prozessmodell ist das CRISP–DM, welches eine hierarchische Zerlegung des Prozesses beschreibt. Die Zerlegung des KDD–Prozesses in Teilaufgaben mit unterschiedlichen Abstraktionsgraden wurde 1997 von Engels, Lindner und Studer [Engels u. a. 1997] vorgestellt. Insbesondere wird die Aufgabenzerlegung des KDD–Prozesses in *Problem Solving Module (PSM)* und eine Abbildung auf *Reuseable Process Units (RPU)* im Rahmen des *User Guidance Module (UGM)* vorgestellt (vgl. Kapitel 5.2). Die generischen Aufgaben aus CRISP entsprechen somit den PSM's und die spezifizierten Aufgaben den RPU's. Im Detail wurde dieser Aspekt von Robert Engels in seiner Dissertation beschrieben [Engels 1999]. Dieser Teil des UGM Ansatzes wird von den Autoren *top-down* Ansatz genannt. Um aber nun zu einem instanziierten Prozess zu kommen, reicht es nicht, die RPU's oder spezifizierten Aufgaben über Zerlegung abzubilden. An dieser Stelle müssen die zur Verfügung stehenden Daten für die Abbildung auf einen instanziierten Prozess berücksichtigt werden. Innerhalb des UGM ist die Auswahl von Methoden zu den RPU's die *bottem up* Komponente des Ansatzes, die im zweiten Teil dieser Arbeit vorgestellt wird. In den folgenden Kapiteln werden aber erst einmal Anwendungen aus der Praxis vorgestellt, um zu zeigen wie komplex die Problematiken im KDD–Prozess sind.

Kapitel 3

KDD Anwendungen in der Praxis

3.1 KDD in der Praxis

In diesem Kapitel wird anhand einer konkreten Anwendung im Bereich Qualitätsmanagement der Produktbetreuung von Fahrzeugen aus eigener Erfahrung beschrieben. Die Anwendung verdeutlicht wie KDD–Projekte in der Praxis durchgeführt werden. In Kapitel 4 sind dann die Ergebnisse aus der Anwendung dargestellt. Aufgrund dieser Erfahrungen und Berichten aus realen Anwendungen, lassen sich Unterstützungspotentiale innerhalb des KDD–Prozesses identifizieren, die in Kapitel 5 dargestellt werden.

3.2 KDD im Produktbewährungsprozess

In den folgenden Abschnitten wird die KDD–Anwendung und der Produktbewährungsprozess vorgestellt. Nach der Analyse der Benutzeranforderungen und den Zielen der Anwender, die mit KDD erreicht werden sollen, werden die einzelnen Lösungswege, die auf Basis der Anforderungsanalyse (siehe Abschnitt 3.4) entwickelt wurden, vorgestellt.

Der Produktbewährungsprozess (**PBP**) ist ein wichtiger Bestandteil der Qualitätssicherung und Verbesserung der Produkte. Bevor auf die Einzelheiten des PBP eingegangen wird, werden im folgenden zuerst zentrale Begriffe aus dem Bereich Qualtitätsmanagement definiert.

Definition 3.1 (Ausfall:) *Die Beendigung der Funktionsfähigkeit einer materiellen Einheit (Produkt) im Rahmen der zugelassenen Beanspruchung bezeichnet man als Ausfall [Deutsche Gesellschaft für Qualität e.V. 1995].*

Definition 3.2 (Ausfallquote:) *Die Ausfallquote ist die temporäre Ausfallhäufigkeit dividiert durch die betrachtete Betriebsdauer [Deutsche Gesellschaft für Qualität e.V. 1995].*

Definition 3.3 (Schadensschwerpunkt:) *Ein Schadensschwerpunkt ist eine materielle Einheit, bei der die Ausfallquote über den definierten Produktionstoleranzen liegt.*

Definition 3.4 (Schadensschlüssel:) *Im Falle eines Ausfalls einer Einheit wird der Ausfall anhand eines Schadensschlüsselbuches vercodet. Ein Schadensschlüssel beschreibt eindeutig ein Bauteil.*

Abbildung 3.1 zeigt ein Beispiel aus dem Schadensschlüsselbuch.

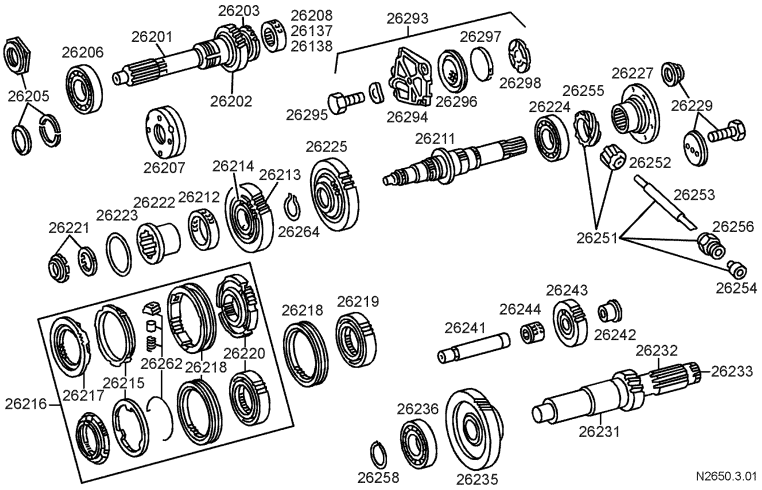

Abbildung 3.1: Schadensschlüssel

Trotz sorgfältigster Qualitätssicherung während des Produktentstehungsprozesses treten Ausfälle auf. Danach muss das Produkt in aller Regel instandgesetzt werden. Die Kosten für diese Instandsetzung gehen während der Gewährleistungszeit zu Lasten des Unternehmens. Darüber hinaus übernehmen Hersteller auch Kosten außerhalb des Gewährleistungszeitraums im Rahmen von Kulanzregelungen. Diese Übernahme der Kosten ist vor allem bei Markenherstellern mit einem hohen Qualitätsanspruch eine Imagefrage.

Wie schon gesagt, ist der PBP ein Element der Qualitätssicherung und der Sicherung der Kundenzufriedenheit. Der PBP beginnt mit dem Verlassen des Produkts aus dem Werk und der Übergabe an den Kunden. Der PBP setzt sich mit dem Produkt nach der Fertigung auseinander.

Die Hauptaufgaben des PBP sind:

- Beobachtung und Berichterstattung über die Qualität des Produktes im Feld beim Kunden
- Fehleranalyse wenn Beanstandungsquoten über den Produktionstoleranzen liegen.

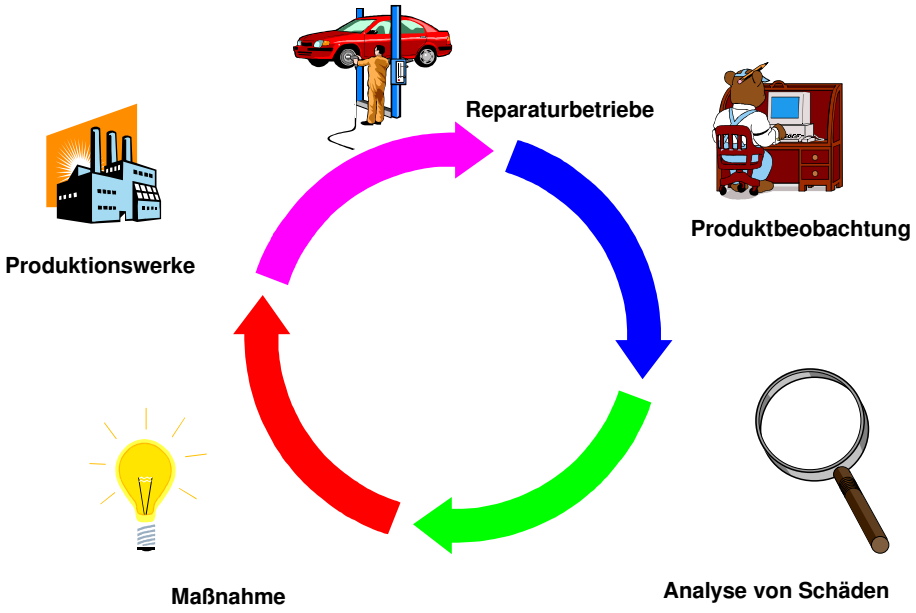

Abbildung 3.2: Produktbewährungsprozess

Die Ergebnisse aus dem Produktbewährungsprozess fließen im Rahmen des Qualitätsmanagements wieder in die Produktion und die Neuentwicklung der Produkte ein. In der Abbildung 3.2 ist dieser Prozess dargestellt.

3.3 Das Qualitätsinformationssystem

Für eine KDD–Aufgabe, wie sie zum Beispiel in der Anforderungsanalyse (Abschnitt 3.4] identifiziert wird, und somit die Phase des Geschäftsverstehens[1] beschreibt, benötigt man eine geeignete Datenbasis. Als Datenbasis stand für das Projekt ein Qualitätsinformationssystem zur Verfügung.

Dieses Qualitätsinformationssystem beinhaltet alle bekannten Daten zu einem produzierten Fahrzeug[2]. Diese Daten umfassen wie das Fahrzeug gebaut wurde und welche Beanstandungen und Reparaturen im Rahmen von Garantie und Kulanz an dem einzelnen Fahrzeug aufgetreten sind. Weiterhin sind eine Reihe von Funktionalitäten vorhanden, mit denen die aufgezeichneten Daten ausgewertet bzw. reportet werden können. Mit dem Informationssystem werden folgende Ziele verfolgt:

- Fehleranalyse, Eingrenzen von Fehlern
- Standardisierung der Berichterstattung
- Verbesserte, flexiblere Auswertungen

[1] Business understanding
[2] PKW, LKW oder Bus

- Verringerung der Zeiten und Kosten für Auswertungen
- Vereinheitlichung der Schadenskodierung

Die Anwender des Informationssystems sind vor allem Produktbetreuer im Rahmen des PBP, die jeweils auf einen bestimmten Bereich des Produkts spezialisiert und für dessen Qualität verantwortlich sind. Weitere Anwender kommen beispielsweise aus dem kaufmännischen Controlling oder der Entwicklung. Den Produktbetreuern als die Hauptanwender des Systems steht das Informationssystem in Form eines Mainframeprogramms zur Verfügung. Unter einer menügeführten Benutzeroberfläche stehen eine Vielzahl von Auswertungsmöglichkeiten bereit. Diese decken die häufig benötigten Standardanfragen ab. Dazu gehören zum Beispiel Hitlisten der häufigsten Fehler oder grafische Darstellungen der Entwicklung des Schadensverhaltens einzelner Teile oder Teilegruppen. Diese können auf vielfältige Weise parametrisiert und an spezielle Bedürfnisse angepaßt werden.

Im Informationssystem liegen alle Informationen über das Produkt, also das Fahrzeug, vor. Bei diesen kann zwischen Daten über den Bauzustand und die Produktbewährung unterschieden werden. Unter Bauzustand werden dabei alle Daten verstanden, die zum Zeitpunkt der Auslieferung über das Fahrzeug bekannt sind. Dazu gehören unter anderem:

- Ausstattungsdaten wie Baureihe, Motorisierung oder Extras
- Herkunftswerk des Fahrzeugs
- Vertriebsgebiet des Fahrzeugs
- Zeitpunkt der Produktion oder der Erstzulassung.

Die Produktbewährung wird in Form einer Beanstandungshistorie gespeichert. Für jeden Werkstattbesuch und jeden beanstandeten Mangel wird ein Datensatz angelegt. Folgende Informationen werden in einem solchen Datensatz aufgenommen:

- Betroffenes Bauteil
- Art des Schadens
- Kilometerstand
- Datum des Werkstattbesuchs
- Entstandene Kosten

Die drei Richtungen, aus denen die Daten des Informationssystems kommen, sind in Abbildung 3.3 angedeutet. Zunächst wird für jedes einzelne produzierte Fahrzeug vom Werk ein Datensatz angelegt. Der Vertrieb liefert Daten, die dem Auftrag

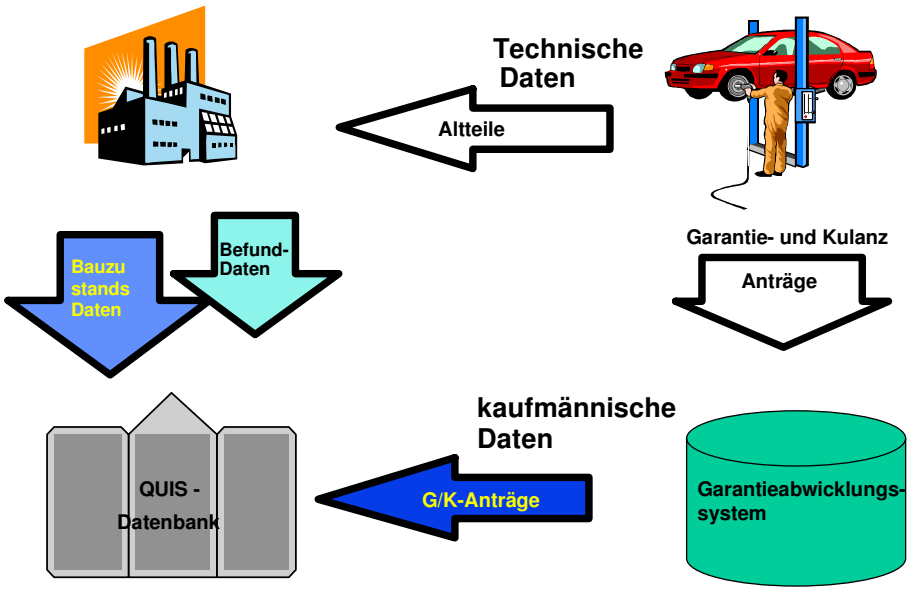

Abbildung 3.3: Quellen der Daten

entstammen. Dritte Quelle sind die Werkstätten, die die Beanstandungshistorie aufnehmen.

Diese Beanstandungsdatensätze werden stets angelegt, wenn die Vertragswerkstätten Anträge auf Erstattung der entstandenen Reparaturkosten stellen. Während der Garantiezeit werden alle Reparaturen vom Hersteller übernommen, und somit befinden sich alle Daten der Reparaturen aus dieser Zeitspanne in der Datenbank. Auch die regulären Wartungen, die während dieses Zeitraums auftreten, sind abgespeichert, allerdings nicht extra ausgezeichnet. Mängel, die nach der Garantiezeit aufgetreten sind, können ganz oder teilweise auf Kulanzbasis beseitigt werden. Sollen also Aussagen über die Produktbewährung nach der Garantiezeit aus den Beanstandungsdaten gemacht werden, so ist zu beachten, dass nicht mehr alle Schäden gemeldet werden. Insbesondere liegt kein Datensatz vor, wenn Reparaturen nicht in den Vertragswerkstätten durchgeführt werden.

Um auf die Produktqualität Einfluß zu nehmen, können verschiedene Maßnahmen ergriffen werden. Zum einen kann das Produkt, dass sich bereits im Einsatz befindet, verändert werden. Es kann sich als sinnvoll erweisen, sogenannte verdeckte Rückrufaktionen durchzuführen, indem bei den regulären Inspektionen schadensverursachende Teile präventiv ausgetauscht werden, ehe der Fehler zutage tritt und eventuell teure Folgeschäden nach sich zieht. Häufiger jedoch wird eine Auffälligkeit im Fehlerverhalten zu einer Veränderung in der Produktion führen, in der Hoffnung, die Fehlerursache damit auszuschalten oder einzudämmen. Solche Änderungen können an der Konstruktion vorgenommen werden, es können andere Materialien eingesetzt werden oder der Produktionsprozess kann verändert werden.

3.4 Anforderungen an die KDD–Anwendungen

In den vorangegangen Abschnitten 3.2 und 3.3 wurde die Anwendungsdomain vorgestellt. Zur Durchführung einer KDD–Anwendung benötigt man aber eine klare Identifizierung der Ziele, die mit dem Projekt erreicht werden sollen. Es müssen also die Aufgaben entsprechend der generischen Aufgabe aus der *Business Understanding* Phase gelöst werden. Zu Beginn des Projektes wurde im Unternehmensbereich eine Anforderungsanalyse durchgeführt, um die Zielsetzungen für ein KDD–Projekt zu identifizieren [Becker 1995].

3.4.1 Erkennen von ungünstigen Bauzuständen

Bei der Erkennung von ungünstigen Bauzuständen geht es um das Erkennen von Zusammenhängen, bzw. Abhängigkeiten zwischen Bauzuständen und einem bekannten Schadensschwerpunkt. Durch eine solche Analyse verspricht man sich Hinweise auf die Schadensursache. Weiterhin kann hierdurch die Gruppe der betroffenen Fahrzeuge für eventuelle Kundendienstmaßnahmen, die meist sehr kostenintensiv sind, eingeschränkt werden.

3.4.2 Optimierung des Frühwarnsystems

Diese globale Anforderung von Seiten des Unternehmensbereiches unterteilt sich in zwei Themenschwerpunkte:

1. Ressourcenbedarf
2. Frühere Prognosen

Punkt 1 bezieht sich auf die Verbesserung der Auswertungen in QUIS im Sinne von kürzeren Wartezeiten und geringeren Ressourcenbedarf.

Punkt 2 hingegen hat die Zielsetzung, die Erkennung von zukünftigen Schadensschwerpunkten in dem Informationssystem zu optimieren.

3.4.3 Erkennen von zeitgleichen Anstiegen

Da in einem Fahrzeug alle Teile in einem funktionalen und räumlichen Zusammenhang stehen, können Beanstandungen auch durch die Werkstätten unterschiedlich vercodet werden. Dies geschieht in der Regel bei der Entstehung von neuen Schadensschwerpunkten. So kann sich ein neuer Schadensschwerpunkt in der Beanstandungsquote auf mehrere abhängige Teile verteilen. Die Anforderung beinhaltet das möglichst frühe Erkennen von Teilegruppen, auf die sich ein neuer Schadensschwerpunkt verteilen könnte.

3.4.4 Erkennen von Kostenveränderungen

Veränderungen der Kostenstruktur einer Beanstandung können als Indikator für eine Veränderung der Schadensursache dienen. Eine solche Veränderung der Schadensursache weißt auf einen neuen Schadensschwerpunkt hin.

3.4.5 Erkennen von saisonalen Einflüssen

Aus den Erfahrungen der Anwender ist bekannt, dass es saisonale Schwankungen der Beanstandungszahlen für einzelne Beanstandungsteile gibt. Diese periodischen Entwicklungen werden von den Produktbetreuern als normale und schwer beeinflußbare Entwicklungen angesehen. Aus diesem Grunde sollen solche Ereignisse nicht als Themen mit Handlungsbedarf in einem Frühwarnsystem identifiziert werden. Diese Fragestellung gliedert sich somit in zwei Teilanforderungen.

1. Identifizierung von periodischen An- und Abstiegen von Beanstandungen oder Kosten in historischen Daten.

2. Neutralisierung der saisonalen Ereignisse.

3.4.6 Prognose von Schadensschwerpunkten

Bei der Prognose von Schadensschwerpunkten ist es das Ziel von frühen Betriebszeiten/Ausfallverhalten auf spätere Betriebszeiten/Ausfallverhalten zu schließen. Diese Prognosen können dann mit den geplanten Qualitätszielen verglichen werden. Eine solche Prognose kann insbesondere bei der Beobachtung von Maßnahmen bei bekannten Schadensschwerpunkten hilfreich sein, da man dann die Wirkung einer Maßnahme früher beurteilen kann. Das Hauptziel ist aber die Überwachung aller Fahrzeuge im Feld um neue Schadensschwerpunkte so früh wie möglich zu erkennen.

3.4.7 Ziele des Projektes

Neben der Anforderungsanalyse ist für den Erfolg eines Projektes und somit auch eines KDD–Projektes wesentlich, dass man sich in dieser frühen Phase des Projektes darüber einigt, wann das Projekt erfolgreich abgeschlossen ist.

In den vorangegangenen Abschnitten wurde das Ergebnis der Anforderungsanalyse vorgestellt. Innerhalb des Projektes sollte zu drei der Themen ein KDD–Lösungsansatz entwickelt werden und das Potential von KDD–Techniken aufgezeigt werden.

Aus der Anforderungsanalyse ergeben sich zwei Arten von Problemen in dem Anwendungsbereich:

1. Der erste Problemtyp bezieht sich auf die Analyse von Schadensschwerpunkten: Wie kann der Prozess der Ursachenfindung unterstützt werden? Gibt es einen Bauzustand bei dem der Schaden häufiger auftritt?

2. Der zweite Problemtyp bezieht sich auf die Verbesserung der Produktbeobachtung mit dem Ziel, früher auf Schadensschwerpunkte reagieren zu können. Hierunter fallen die Anforderungen

   - Optimierung der Auswertungsmöglichkeiten
   - Erkennen von zeitgleichen Anstiegen
   - Erkennen von Kostenveränderungen
   - Erkennen von saisonalen Einflüssen
   - Prognose von Schadensschwerpunkten

Mit beiden Problemtypen deckt man zwei zentrale Phasen des Produktbewährungsprozesses, die Produktbeobachtung und die Analyse von Schadensschwerpunkten, ab (vgl. Abbildung 3.2).

3.5 Problemtyp I: Auswahl der Methoden zur Analyse

Nachdem die Ziele und Aufgaben identifiziert wurden und die Datenbasis betrachtet wurde, stellt sich die Frage, welche Verfahren für den ersten Aufgabentyp geeignet sind. Im allgemeinen wird die Auswahl der Verfahren durch das *Know How* des Datenanalysten bestimmt. Doch bevor man das Verfahren auswählen kann, muss man den KDD–Problemtyp bestimmen, in Abschnitt 2.6 wurden diese vorgestellt. Die Frage, die in dieser Anwendung beantwortet werden soll, ist welche Kombination von Bauzuständen wurden in den Fahrzeugen mit dem Schadensschwerpunkt X verbaut.

Dieses Problem läßt sich als ein Klassifikationsproblem mit zwei Klassen definieren:

1. Fahrzeuge mit dem Ausfall

2. Fahrzeuge ohne den Ausfall

Wichtig für den Klassifikator ist noch, dass er eine Beschreibung der Klassen liefert. In dieser Anwendung steht der Wissensakquisitionsaspekt im Vordergrund und nicht unbedingt die Klassifikationsgüte des Klassifikators. Die Beschreibung des Ausfalls ist hier von Interesse, nicht die Vorhersage.

Die Anwendungsfrage läßt sich auch auf den Problemtyp der Abhängigkeitsanalyse abbilden. Wie hängen die Ausfälle von den Bauzuständen der betroffenen Fahrzeuge ab?

Die KDD–Aufgabe ist somit auf zwei Problemtypen übertragbar, was die Anzahl der möglichen Algorithmen vergrössert, die theoretisch angewendet werden können. Für beide Problemtypen stehen eine Vielzahl von Algorithmen zur Verfügung.

In der Praxis wird die Auswahl der Algorithmen durch das *Know How* der Datenanalysten und der aktuell verfügbaren Algorithmen bestimmt, so dass so mindestens schon eine Vorauswahl getroffen wird. Diese Algorithmen werden dann auf ihre Anwendbarkeit getestet. Dieser Testprozess zur Auswahl der Algorithmen ist sehr zeitintensiv und durchläuft im allgemeinen mehrere Iterationen, bis man sich für das *geeignetste* Verfahren entscheidet.

Im folgenden werden die, in dieser Anwendung berücksichtigten Gruppen von Verfahren beschrieben, und ihre Anwendung an Beispielen gezeigt. In darauf folgenden Abschnitten wird dann über die konkreten Anwendungen auf die Fragestellung berichtet.

3.6 Klassifikationsverfahren

Der Begriff der Klassifikation ist von verschiedenen Autoren unterschiedlich definiert. So versteht Gordon in seinem Buch [Gordon 1999] Klassifikation, als das Finden einer Klasseneinteilung auf einer Datenmenge in der die Klassen unbekannt sind. Im KDD ist die Data Mining Aufgabe Klassifikation nach [Fayyad u. a. 1996], das Lernen einer Menge von Funktionen, die ein Element der Datenmenge einer der vordefinierten Klassen zuordnet. Die Klasseneinteilung ist also bekannt. Klassifikation, wie sie Gordon beschreibt, wird im KDD unter der Data Mining Aufgabe Clustering verstanden (vergleiche auch 2.6).

Klassifikationsverfahren lassen sich durch verschiedene Kriterien beschreiben. In erster Linie unterscheidet man zwischen den statistischen Verfahren und Verfahren aus dem Bereich des induktiven maschinellen Lernen.

Die hier beschriebenen Verfahren sind alle induktive Verfahren, die auf einem Ausschnitt der zur Verfügung stehenden Daten (Trainingsmenge) angewendet werden, um einen Klassifikator zu erhalten. Man spricht auch in diesem Zusammenhang von trainieren. Der Klassifikator wird dann auf die bisher noch nicht genutzten Daten (Testmenge) angewandt und die Klassifikationsgüte des Klassifkators getestet.

3.6.1 Statistische Verfahren der Klassifikation

Voraussetzung für die statistischen Verfahren ist, dass alle Attribute numerisch sind. Da die Daten in KDD diese Bedingung im allgemeinen nicht erfüllen, müssen die Daten in geeigneter Form in stetig–numerische Werte transformiert werden [Kauderer und Nakhaeizadeh 1998].

3.6.1.1 Diskriminanzanalyse

Die Diskriminanzanalyse (DA) ist ein klassisches Verfahren aus der Statistik, von dem es inzwischen viele Versionen gibt. Ausführliche Beschreibungen findet man in fast jedem Buch über statistische multivariate Analyse so zum Beispiel in [Hartung und Elpelt 1995]. An dieser Stelle soll nur das Grundprinzip der DA kurz erklärt werden, da im Rahmen der Datenbeschreibung im zweiten Teil dieser Arbeit noch genauer auf die DA eingegangen wird (siehe Kapitel 6).

Ziel der Diskriminanzanalyse ist es, eine Diskriminanzfunktion zu finden, d.h. eine geeignete Linearkombination der diskriminierenden Variablen, die am stärksten zwischen den q Klassen trennt. Da es sich im allgemeinen im KDD um mehr als zwei Klassen handelt, zwischen denen unterschieden werden soll, benötigt man auch mehrere Funktionen.

An einem kleinen Beispiel mit nur zwei Klassen, aus [Nakhaeizadeh u. a. 1998], soll im folgenden die Grundidee und Vorgehensweise der Diskriminanzanalyse dargestellt werden. Eine Bank möchte zwischen kreditwürdigen (KW) und nicht kreditwürdigen (NKW) Kunden unterscheiden. X1 und X2 seien zwei Merkmale der Kunden z .B . Einkommen und Alter. In Abbildung 3.4 ist dargestellt, dass aufgrund eines Merkmales zwischen den beiden Klassen KW und NKW nicht unterschieden werden kann. Die Projektion der Merkmalswerte auf die Achsen X1 oder X2 erlauben keine Differenzierung, da die Überschneidungsbereiche der zwei Klassen zu groß ist. Die DA versucht eine neue Achse zu bestimmen. Die neue Achse ist eine Kombination der Merkmale Einkommen und Alter. Die Funktion, die diese neue Achse beschreibt, heißt Diskriminanzfunktion. Diese geschätzte Diskriminanzfunktion ermöglicht nicht nur die Klassifikation neuer Kunden, sondern zeigt auch welche Merkmale zur Trennung der Klassen beitragen.

3.6.1.2 K–Nächste-Nachbarn

Das KNN–Verfahren ist ein Verfahren das auf Ähnlichkeitsberechnungen basiert. Ziel ist es Analogien zu anderen Objekten zu bestimmen. In unserem Beispiel Klassifikation von Bankkunden bestimmt man die K ähnlichsten Kunden. Der neue Kunde wird dann der Klasse zugeordnet, die die Majorität der ähnlichen Kunden hat. Die verschiedenen Methoden der Ähnlichkeitsberechnung werden in Abschnitt 3.11 beschrieben.

3.6.2 ML–Verfahren der Klassifikation

Die bisher beschriebenen Verfahren können nur numerische Werte verarbeiten. Die Verfahren aus dem maschinelle Lernen sind traditionell auch in der Lage symbolische Werte zu verarbeiten. Die hier im folgenden beschriebenen Gruppen von Verfahren gehören alle zu den induktiven Lernverfahren. Viele dieser Verfahren können nur symbolische Werte verarbeiten.

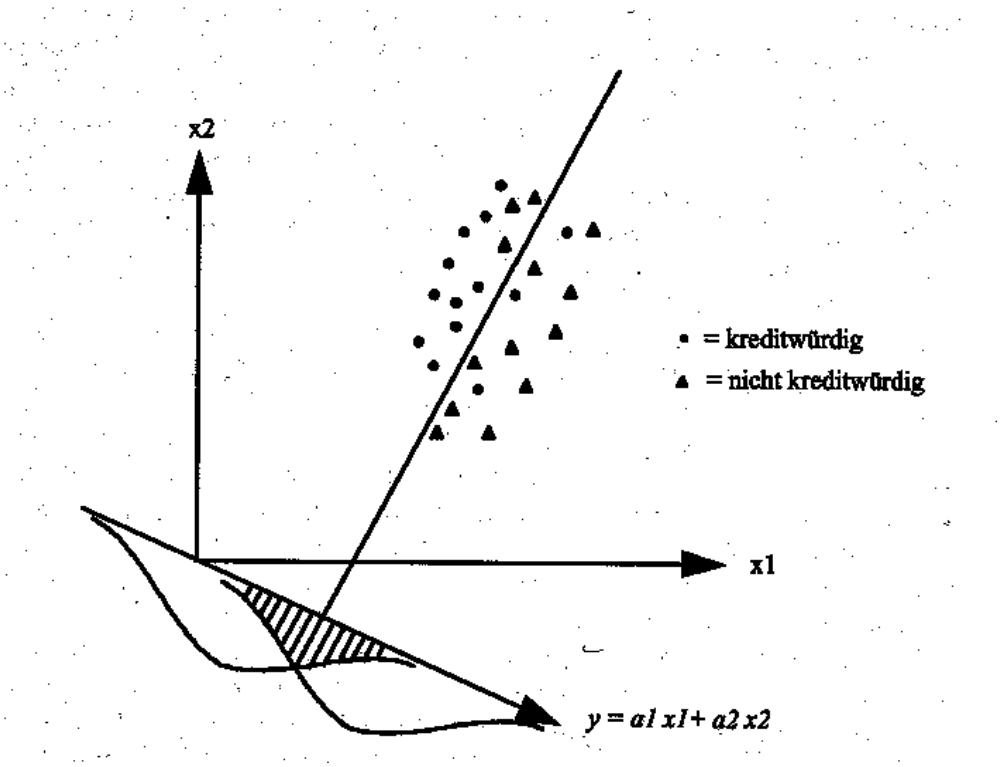

Abbildung 3.4: Diskriminanzfunktion im Fall von zwei Klassen aus [Nakhaeizadeh u. a. 1998]

3.6.2.1 Entscheidungsbäume

Grundlagen zu Entscheidungsbäumen Entscheidungsbäume sind die bekannteste und am meisten eingesetzte Data Mining Methode aus dem maschinellen Lernen um eine Begriffsbeschreibung und einen Klassifikator zu erhalten. Eine ausführliche Einführung findet man in [Mitchell 1997]. In dieser Arbeit soll an dieser Stelle das Grundprinzip dieser Gruppe von Verfahren beschrieben werden, zu der auch die populären Verfahren wie ID3 [Quinlan 1986] und C45 [Quinlan 1993] gehören. Diese Entscheidungsbaumlernverfahren suchen in einem aussagenlogischen Hypothesenraum einen Entscheidungsbaum mit dem Ziel möglichst einen Baum mit wenig Ebenen zu erhalten.

Ein Entscheidungsbaum besteht aus Knoten, die durch gerichtete Kanten verbunden sind. Jeder Knoten beinhaltet einen Test, der auf eines der Merkmale prüft. Für jedes Ergebnis aus dem Test führt eine Kante wieder zu einem Knoten oder einem Blatt. Ein Blatt beinhaltet die Klasse zu dem der entsprechende Datensatz gehört. In unserem Beispiel über die Kreditwürdigkeit der Bankkunden könnte ein Entscheidungsbaum wie in Abbildung 3.5 aussehen. Um einen Entscheidungsbaum darzustellen, wurde das Beispiel um drei Attribute erweitert, X3 Erwerbstätig, X4

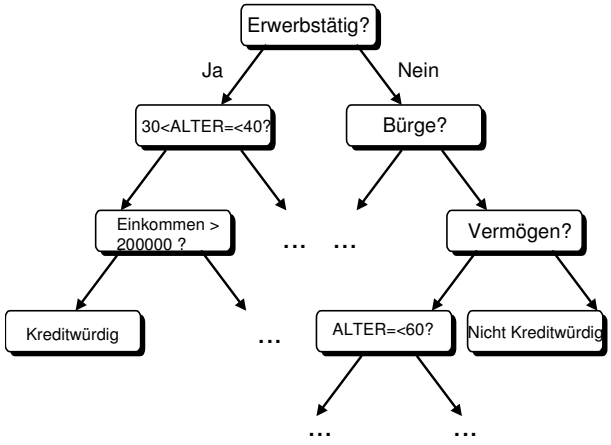

Abbildung 3.5: Vereinfachter Entscheidungsbaum

Bürgen und X5 Vermögen.

Entscheidungsbäume können auch leicht in eine Menge von *wenn* - *dann* Regeln transformiert werden, die die Lesbarkeit für Anwender verbessert.

Bevor der Basisalgorithmus für Entscheidungsbaumlerner beschrieben wird, soll verdeutlicht werden, für welche Probleme Entscheidungsbäume allgemein geeignet sind (vergleiche auch [Mitchell 1997]).

- Beispiele sind in Form von Attribut–Werte–Paaren repräsentiert. Die Instanzen sind durch eine feste Menge von Attributen (z.B. Erwerbstätig) und ihren Werten (z.B. Ja) dargestellt. Die Attribut–Werte Paare müssen hierbei nicht binär sein. Die einfachste Lernaufgabe für Entscheidungsbaumlerner ist, wenn jedes Attribut nur eine kleine Menge von Werten annehmen kann, zum Beispiel für ein Attribut Temperatur wären {Heiß, Mild, Kalt} mögliche Werte. Durch Erweiterung des Basisalgorithmus durch Diskretisierungsmethoden können auch Attribute mit numerischen Werten verarbeitet werden.

- Das Klassenattribut muss symbolisch sein. Das Beispiel in Abbildung 3.5 ist eine boolesche Klassifikation. Entscheidungsbaumlernern können aber ohne weiteres auch mehr Klassenprobleme lösen.

- Eine Disjunkte Beschreibung der Klassen sollte erwartet werden. Wie vorher schon beschrieben, sind Entscheidungsbäume disjunkte Ausdrücke.

- Die Trainingsdaten können Fehler enthalten. Entscheidungsbaumlerner sind robust gegen Fehler, egal ob in den Klassenattributen oder den beschreibenden Attributen.

- Die Trainingsdaten können fehlende Werte (*missing values*) enthalten.

Basisalgorithmus Die meisten der entwickelten Algorithmen für Entscheidungsbaumlerner sind Varianten eines Grundgerüstes von Algorithmen, der mit einer top–down, greedy Suche durch den Raum der möglichen Entscheidungsbäume (Hypothesenraum) den Entscheidungsbaum aufbaut. Diese Verfahren werden auch TDIDT[3]–Verfahren genannt.

Aufbau eines Entscheidungsbaums

A: Attributliste
C: Menge von Beispielen
generiere_baum(A,C):

1. Auswahl eines Attributes a aus der Attributliste, Knoten mit Namen a, Menge von Beispielen C

2. Reduktion der Attributliste A um a;
   Für alle Attributwerte von a:
   Kante mit dem i-ten Wert beschriften;
   alle Beispiele aus C mit dem i-ten Wert entfernen und in C_i eintragen;
   C_i als neuen Knoten unter die Kante setzen;
   für alle Beispielmengen C_j:
   wenn $C_j = \emptyset$ Anhalten! keine Generalisierung möglich
   wenn alle Beispiele aus C_j derselben Klasse angehören ist C_j ein Blatt
   wenn alle Beispielmengen Blätter sind Anhalten!
   sonst mit den Knoten wieder zu 1 gehen.

An dem Basisalgorithmus kann man leicht erkennen, dass der Baum nicht tiefer als die Anzahl der Attribute werden kann, da in jedem Rekursionsschritt ein Attribut aus der Attributliste gestrichen wird. Die Qualität des Lernergebnisses hängt von der Auswahl der Attribute für die Knoten ab. Bei ID3 und C4.5 basiert die Auswahl auf dem Informationsgewinn (*information gain* [Quinlan 1986, Quinlan 1993]) der einzelnen Attribute. Neben dem Informationsgewinn können auch andere informationstheoretische Maße, wie der *Gini Index* [Breiman u. a. 1984], zur Attributauswahl genutzt werden. Aber auch das Relevanzmaß [Baim 1988] und das χ^2–Maß, sowie viele andere Maße sind anwendbar. [Borgelt und Kruse 1998] haben eine Auswahl von verschiedenen Maßen zur Attributauswahl für die Induktion von Entscheidungsbäumen untersucht. Eine klare Überlegenheit eines Maßes ließ sich aus den experimentellen Ergebnissen nicht herleiten. Die Qualität des gelernten Klassifikators ist vielmehr auch vom Datensatz abhängig. Borgelt und Kruse empfehlen daher,

[3]top-down induction of decision tree

in Abhängigkeit der besonderen Eigenschaften der Datensätze mehrere Maße auszuprobieren um einen möglichst guten Entscheidungsbaum zu erhalten.

Erweiterungen zum Basisalgorithmus Wie zuvor schon erwähnt sind in den verschiedenen Entscheidungsbaumlernern Erweiterungen zum Basisalgorithmus implementiert. An dieser Stelle wird nicht auf die Implementierung eingegangen, sondern es werden die grundsätzlichen Möglichkeiten besprochen. Eine ausführlichere Beschreibung findet man in [Mitchell 1997]. Der Einsatz von verschiedenen Attributauswahlmethoden wurde schon im vorherigen Abschnitt beschrieben.

Der simple Basisalgorithmus, wie er in Abschnitt 3.6.2.1 eingeführt wurde, neigt zum *Overfitting*. Ein Modell[4] *overfits* die Trainingsmenge, wenn es ein anderes Modell gibt, das eine höhere Fehlerrate auf der Trainingsmenge hat und eine kleinere Fehlerrate auf der Testmenge.

Um dieses *Overfitting* zu vermeiden, wird in vielen Entscheidungsbaumlernern der konstruierte Baum anschließend gestutzt (*pruning*), d.h. Entscheidungsknoten die nur geringen Anteil an der Klassifikationsgüte haben, werden wieder entfernt. Für das *Pruning* gibt es zwei Methoden die in [Mitchell 1997] beschrieben werden:

1. *reduce-error pruning* [Quinlan 1987] und
2. *rule post-pruning*, eingesetzt in C4.5 [Quinlan 1993]

Der Basisalgorithmus ist bisher auf symbolische Attribute beschränkt, für das Zielattribut sowie für die beschreibenden Attribute. Die letztere Restriktion kann relativ leicht durch dynamische Definition von neuen diskreten Attributen aus den numerischen Attributen aufgehoben werden. Die Schwierigkeit hierbei ist, die optimale Diskretisierung zu finden.

3.6.2.2 Künstliche neuronale Netze

Künstliche neuronale Netze (NN) sind eine allgemeine, praktikable Methode zum Lernen aus Beispielen von

- reellwertigen Funktionen,
- diskreten Funktionen und
- vektorwertigen Funktionen.

Neben dem Lernen aus Beispielen können bestimmte NN auch zum Clustern, d.h. zum Finden einer Klassenstruktur in Datensätzen, eingesetzt werden. Auf diesen Aspekt wird aber im Folgenden nicht weiter eingegangen. Die vielleicht relevanteste Methode sind die *Self Organising Maps*, die in [Kohonen 1997] beschrieben sind.

[4] Entscheidungsbaum

NN sind dynamische Systeme, die aus einer Anzahl von einfachen Verarbeitungselementen bestehen, die parallel und unabhängig von einander arbeiten können. Diese Verarbeitungselemente werden Neuron genannt. Ein NN besteht aus einer oder mehreren Gruppen von Neuronen (Units) und gewichteten Verbindungen, die die einzelnen Neuronen miteinander verbinden. Abbildung 3.6 zeigt ein typisches Neuron.

Ein Neuron kann sich in einem aktiven oder passiven Zustand befinden. Seinen Zustand gibt es über die gewichteten Verbindungen an andere Neuronen weiter. Es bestimmt seinen Zustand aus den Eingaben. Überschreitet das Eingabesignal von den aktiven Eingaben einen gewissen Schwellwert, wird das Neuron selbst aktiv.

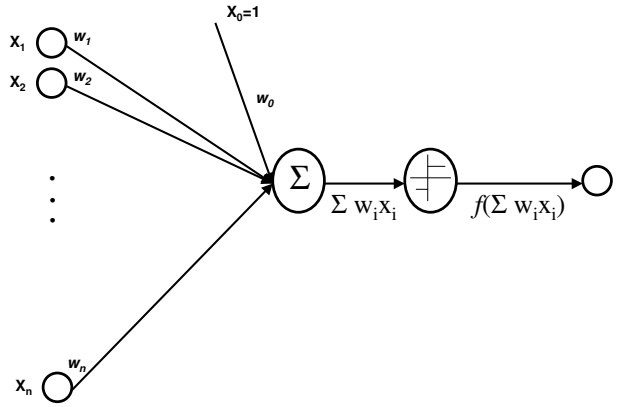

Abbildung 3.6: Neuron mit Eingaben,

Das Perceptron ist die einfachste Form der NN. Das Perzeptron ist definiert durch die Aktivierungsfunktion [Mitchell 1997]

$$f_{perceptron}(x) = \begin{cases} 1 & wenn \sum_{i=1}^{n} w_i x_i > w_0 \\ -1 & sonst \end{cases} \tag{3.1}$$

Das Perzeptron definiert eine Hyperebene durch den Hypothesenraum. Mittels dieser Hyperebene werden die Beispiele in zwei Klassen geteilt. Ein Perzeptron ist also ein binärer Klassifikator. Komplexere Probleme kann man nur durch Verwendung von mehreren Neuronen lösen. In Abbildung 3.7 ist der Aufbau eines neuronalen Netzes dargestellt. Diese Netze nennt man auch Multilayer Netzwerke. Diese NN sind schichtweise aufgebaut und Verbindungen bestehen immer nur zwischen Neuronen einer Schicht zur nächsten Schicht[5]. Um auch nichtlineare separierbare Probleme zu lösen bedarf es noch zwei Änderungen. Erstens die Verwendung von reellwertigen Eingaben und zweitens statt eines Schwellwertes und einer binären Aktivierung wird eine stetige Aktivierung erlaubt, die aus der Abbildung der Netzeingaben auf eine

[5]feedforward Netze

sigmoide Transferfunktion hervorgeht.

$$f_{sigmoid}(x) = \frac{1}{1 + e^x} \tag{3.2}$$

Numerische Attribute werden auf der Eingabeebene (*Input Layer*) durch ein Neuron abgebildet. Dagegen benötigt man im allgemeinen für symbolische Attribute mit u Werten u Neuronen auf der Eingabeebene.

Das Trainieren eines Multilayer Netzes hat folgende Phasen:

- Wahl einer Netzwerk Topologie, die sich vielleicht auch später ändert.
- Teilung der Trainingsdaten in eine Trainingsmenge und Validierungsmenge, um ein Overfitting des Lernlaufs zu verhindern.
- Solange bis ein definiertes Haltekriterium erfüllt ist, werden folgende Aktivitäten durchgeführt:
  - Inkrementelles Anlegen der Trainingsbeispiele an die Eingabeebene
  - Berechnung des Fehlermaßes als Basis für die Änderung der Gewichtungen
  - Anpassung der Gewichte im Netz

Die Anpassung der Gewichte geschieht nach dem Backpropagation Algorithmus. Hierbei werden von der Ausgabeebene (*Output Layer*) beginnend der Fehler dieser Schicht bestimmt und eine Korrektur der Gewichte vorgenommen. Die Fehler werden also im Netz zurückpropagiert, bis jede Schicht ihren Fehler verkleinert hat. Bei Multilayer Netzwerken kann nicht garantiert werden, dass beim Lernprozess ein globales Fehlerminimum erreicht wird. Zahlreiche Anwendungen haben aber gezeigt, dass mit diesen Netzen sehr gute Ergebnisse erzielt werden können.

Aus dem oben beschriebenen Verfahren ergeben sich drei offene Punkte, die bisher noch nicht diskutiert wurden:

1. **Die Auswahl der Topologie:** Mit der Auswahl der Topologie des Netzwerkes definiert man die Lösungsfähigkeit des Netzes. Mitchell beschreibt in [Mitchell 1997, S. 105] auf für welche Funktionsklassen welche Art von Netzwerktyp notwendig ist.
   - **Boolesche Funktionen:** Jede boolesche Funktion kann durch ein zweischichtiges Netzwerk[6] abgebildet werden, obgleich im ungünstigsten Fall die Anzahl der Neuronen in der Zwischenebene *Hidden Layer* in Verhältnis zu den Eingabeneuronen exponentiell wächst.

[6] Die Eingabeebene wird nicht mitgezählt.

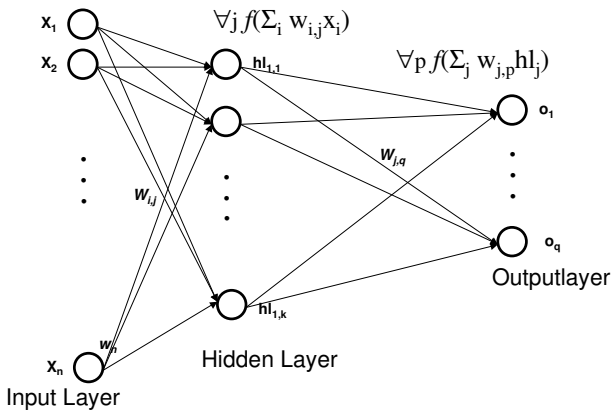

Abbildung 3.7: Layout eines Multilayer Netzwerk

- **Stetige Funktionen:** Jede beschränkte stetige Funktion kann mit einem beliebig kleinen Fehler durch ein zweischichtiges Netz approximiert werden [Cybenko 1989]. Das Theorem basiert auf der Anwendung von Sigmoid-Neuronen in der Zwischenebene und Neuronen mit einer linearen Aktivierungsfunktion in der Ausgabeebene.
- **Beliebige Funktionen**: Mit einem dreischichtigen Netzwerk mit zwei Zwischenebenen mit sigmioden Neuronen kann jede Funktion mit beliebiger Genauigkeit approximiert werden [Cybenko 1988].

2. **Das Fehlermaß zur Änderung der Gewichtungen:** Basis für die Änderung der Gewichte ist der Vergleich der generierten Ausgabe (A) mit der richtigen Ausgabe (T).

$$w_{ij} \leftarrow w_{ij} + \Delta w_{ij},\ wobei + \Delta w_{ij} = \eta(T - A)x_i \tag{3.3}$$

In der Formel 3.3 steht Δw_{ij} für die Veränderung der Gewichte nach der Fehlerbestimmung.

Will man aber stetige Funktionen lernen, so berechnet sich der Fehler und die Veränderung der Gewichte etwas anders. Backpropagation ist dann nichts anderes als ein Gradientenverfahren, durch das die Parameter (Gewichte) der Zielfunktion iterativ approximiert werden, so dass der Fehler D minimiert wird.

$$D = \frac{1}{2}\sum_{l=1}^{n}(t_l - a_l)^2 \tag{3.4}$$

Hieraus ergibt sich dann Δw_{ij} wie folgt:

$$\Delta w_{ij} = \eta \sum_{l=1}^{n}(t_l - a_l)x_{li} \tag{3.5}$$

Hierbei steht n für die Anzahl der Beispiele, η für die Lernrate des Netzes und x_{li} für den Wert des Attributes x_i in dem l-ten Beispiel. Die Lernrate η ist die Schrittgröße während der Gradientenanpassung. Wenn die Schrittgröße zu groß gewählt wird, besteht das Risiko, dass man das globale Minimum der Fehlerrate überspringt. Das nennt man *overstepping*. Andererseits, wenn die Schrittgröße klein gewählt wird, kann die Lernzeit inakzeptabel lang werden. Eine gängige Lösung dieses Problems ist die Verknüpfung der Iteration und der Schrittgröße.

3. **Die Wahl des Abbruchkriteriums:** Die Wahl des Abbruchkriteriums ist ein weiterer wichtiger Faktor beim Trainieren von NN. Durch zuwenig Iterationen kann es sein, dass keine ausreichende Fehlerreduzierung erreicht werden kann, während zu viele Iterationen zum *Overfitting* der Trainingsdaten führen. Im folgenden werden verschiedene Abbruchkriterien vorgestellt, die beim Trainieren von NN eingesetzt werden.

   - Abruch, wenn keine ausreichende Verbesserung des Lernergebnisses mehr erzielt wird
   - Abbruch nach einer festgelegten Anzahl von Iterationen
   - Abbruch nach einer festgelegten Zeit
   - Abbruch, wenn ein ausreichend gutes Modell gelernt wurde bzgl. der Fehlerrate

Geeignete KDD–Probleme für Neuronale Netze Allgemein sind Neuronale Netze nach [Mitchell 1997] für KDD–Probleme mit folgenden Eigenschaften geeignet.

- Beispiele sind durch viele Attribut–Wertepaare gegeben.
- Die Zielfunktion kann durch diskrete Werte, reelle Werte oder einen Vektor von mehreren diskreten und reellen Werten definiert sein.
- Die Trainingsbeispiele enthalten vielleicht Fehler. NN sind robust gegen Rauschen (*noise*) in den Daten.
- Lange Trainingszeiten sind in der Anwendung akzeptabel. Längere Trainingszeiten für NN, als zum Beispiel für Entscheidungsbaumlerner, sind typisch. Die Differenz kann zwischen wenigen Sekunden und mehreren Stunden liegen.
- Eine schnelle Evaluierung und Anwendung der Zielfunktion ist in der Anwendung gefordert. NN sind schnelle Klassifikatoren.
- Die gelernte Zielfunktion muß nicht durch Menschen interpretierbar sein. Die Gewichte zwischen den einzelnen Neuronen sind schwer interpretierbar, der Einfluß von einzelnen Attributen auf das Klassifikationsergebniss ist schwer herauszufinden.

3.6.2.3 Induktive logische Programmierung

Muggleton beschreibt das Gebiet der induktiven logischen Programmierung[7] als die Schnittmenge zwischen logischer Programmierung und maschinellem Lernen [Muggleton 1990]. Die Zielsetzung des induktiven logischen Programmierens kommt aus dem induktiven maschinellen Lernen (vergleiche auch 3.6.2.1). Die zwei zentralen Punkte im ILP sind die Verwendung von Hintergrundwissen und dass das Lernergebnis in einer eingeschränkten Prädikatenlogik 1. Stufe vorliegt [Lavrač und Džeroski 1994].

Diese beiden Erweiterungen zum attributorientierten induktiven maschinellem Lernen werden im Folgenden kurz erläutert.

Hintergrundwissen: Lernen mit Hintergrundwissen entspricht der natürlichen Lernsituation. Mit bereits bekanntem Wissen und aktuellen Beispielen für einen Sachverhalt wird versucht, eine Theorie zu diesem Sachverhalt zu finden. Für die einzelnen Verfahren ist es deshalb von zentraler Bedeutung, in welcher Form und in welchem Umfang Hintergrundwissen zur Verfügung steht. Der Hypothesenraum wird durch die Hypothesensprache, das Hintergrundwissen und die Beispiele beschrieben. Vom Hintergrundwissen ist es im Allgemeinen abhängig, wie leicht oder gut ein Begriff zu lernen ist. Durch das Hintergrundwissen kann eine kürzere und verständlichere Beschreibung des zu lernenden Begriffes gefunden werden. Hintergrundwissen ist aber nicht in jedem Fall nützlich, zu umfangreiches vergrößert auch den Hypothesenraum und damit den Suchraum für die Verfahren. In vielen ILP–Systemen ist die Repräsentation des Hintergrundwissen auf Grundfakten beschränkt [Lavrač und Džeroski 1994], wie zum Beispiel in dem Verfahren FOIL [Quinlan 1990].

Hypothesensprache: Eingeschränkte Prädikatenlogik als Hypothesensprache zu verwenden hat einige Vorteile. Neben der direkten Interpretierbarkeit des Lernergebnisses durch ein System, ist die Ausdrucksstärke im Vergleich zu Standardverfahren des maschinellen Lernens, wie den TDIDT-Verfahren, ein Argument für induktive logische Programmierung. ILP-Verfahren sind in der Lage Relationen auszudrücken. Mit dem Repräsentationsformalismus können auch die Techniken aus dem Bereich der logischen Programmierung übernommen werden [Muggleton und Raedt 1993].

Aufgrund dieser Vorteile würden sich ILP–Verfahren geradezu als Data Mining Verfahren aufdrängen, aber neben diesen Vorteilen gibt es auch einen wesentlichen Nachteil in der Performance der Verfahren und der Art der Repräsentation des Hintergrundwissen in Form von Grundfakten im Speicher. Die Verfahren müssen, um als Data Mining Verfahren einsetzbar zu sein, direkt auf relationalen Datenbanken arbeiten können. Als erstes Verfahren wurde RDT [Kietz und Wrobel 1992] in einer Variante RDT/DB [Lindner und Morik 1995] um einen Hypothesentest direkt auf

[7] In dieser Arbeit steht induktive logische Programmierung immer für den Forschungsbereich und die Abkürzung ILP für die technische Betrachtung des Bereiches.

relationale Datenbanksysteme erweitert. In der Praxis finden ILP–Verfahren bisher allerdings wenig Anwendung.

3.7 Abhängigkeitsanalyse

3.7.1 Probabilistische und possibilistische Netze

Die folgende Einführung wurde von Borgelt, Kruse und Lindner in [Borgelt u. a. 1998] veröffentlicht. Neben der theoretischen Einführung wird dort auch auf die zuvor beschriebene Anwendung im PBP eingegangen.

Das Lernen eines probabilistischen oder possibilistischen Schlussfolgerungsnetzes besteht darin, eine gegebene mehrdimensionale Wahrscheinlichkeits- oder Possibilitätsverteilung in Verteilungen auf Unterräume zu zerlegen. Die zu zerlegende Verteilung ist dabei jedoch nicht direkt gegeben, sondern es steht nur eine Datenbank von Beispielen zur Verfügung. Diese wird benutzt, um (bedingte) relative Häufigkeiten auszuzählen, aus denen die (bedingten oder marginalen) Wahrscheinlichkeiten und Possibilitätsgrade geschätzt werden.

Ein Algorithmus zum Lernen von Schlussfolgerungsnetzen aus Daten besteht immer aus zwei Teilen: einem Bewertungsmaß und einer Suchmethode. Mit Hilfe des Bewertungsmaßes wird die Güte einer gegebenen Zerlegung (eines gegebenen Hypergraphen) eingeschätzt, während die Suchmethode bestimmt, welche Zerlegungen (welche Hypergraphen) überhaupt betrachtet werden. Oft kann das Bewertungsmaß auch benutzt werden, um die Suche zu steuern, da es gewöhnlich das Ziel ist, seinen Wert zu maximieren (oder zu minimieren).

Es gibt eine Vielzahl von Bewertungsmaßen, sowohl für das Lernen probabilistischer, als auch für das Lernen possibilistischer Netzwerke.

Hier sollen nicht alle im Detail besprochen werden. Eine detailliertere Beschreibung findet man in [Borgelt und Kruse 1997a, Borgelt und Kruse 1997b]. Im Folgenden wird daher nur eine Auswahl weiterbetrachtet. Alle aufgeführten Maße haben die wünschenswerte Eigenschaft, dass sie sich lokal, d.h. auf Teilnetzen bzw. einzelnen Hyperkanten berechnen lassen. Die Gesamtbewertung wird aus diesen Einzelbewertungen zusammengesetzt.

Probabilistische Maße

- χ^2-Maß
- Informationsgewinn/wechselseitige Information (information gain/mutual information) [Quinlan 1986, Quinlan 1993]
- (symmetrisches) Informationsgewinnverhältnis [Quinlan 1986, Quinlan 1993]
- Gini-Index [Breiman u. a. 1984]
- symmetrischer Gini-Index [Zhou und Dillon 1991]

- minimale Beschreibungslänge mit relativer oder absoluter Häufigkeitscodierung [Rissannen 1983, Kononenko 1995]
- stochastische Komplexität [Krichevsky und Trofimov 1983, Rissannen 1995]
- g-Funktion (ein Bayessches Maß) [Cooper und Herskovits 1992]

Possibilistische Maße

- d_{χ^2}, abgeleitet vom χ^2-Maß [Borgelt und Kruse 1997a, Borgelt und Kruse 1997b]
- d_{mi}, abgeleitet von wechselseitiger Information [Borgelt und Kruse 1997a, Borgelt und Kruse 1997b]
- Spezifizitätsgewinn (specificity gain) [Borgelt u. a. 1996]
- (symmetr.) Spezifizitätsgewinnverhältnis [Borgelt u. a. 1996]

Alle der oben genannten Maße lassen sich in Verbindung mit einer Vielzahl von Suchmethoden verwenden. Die beiden am häufigsten verwendeten Methoden sind die Bestimmung eines optimalen spannenden Baumes [Chow und Liu 1968], die gleichzeitig auch die älteste ist, sowie die gierige (greedy) Elternauswahl [Cooper und Herskovits 1992] (K2-Algorithmus). Im Prinzip lassen sich beliebige heuristische Suchverfahren, wie z.B. simuliertes Ausglühen (simulated annealing), genetische Algorithmen etc., nutzen.

Die oben genannten Verfahren und Suchmethoden nach Cooper und Herskovits wurden von Christian Borgelt im Rahmen einer Kooperation mit der Otto von Guerike Universität Magdeburg und der DaimlerChrysler Forschung implementiert. Diese Implementierung INES (Induktion von NetzwErkStrukturen) wurde dann für den ersten Problemtyp eingesetzt und evaluiert.

3.7.2 Assoziationsregeln

3.7.2.1 Boolische Assoziationsregeln

1993 führten Agrawal, Imielinski und Swami eine Klasse von Regelmäßigkeiten, Assoziationsregeln (*assoziation rules*) ein [Agrawal u. a. 1993]. Assoziationsregeln beschreiben eine funktionale Abhängigkeit zwischen Attributwerten.

Formal ist das Problem nach [Agrawal u. a. 1993, Agrawal u. a. 1996] folgendermaßen definiert.

Definition 3.5 (Assoziationsregeln:) *Sei $\mathcal{I} = \{i_1, i_2, \ldots, i_n\}$ eine Menge von Literalen, genannt Ereignissen (*item*). Sei $\mathcal{D}$ eine Menge von Transaktionen, wobei jede Transaktion $\mathcal{T}$ eine Ereignismenge ist, so dass $\mathcal{T} \subseteq \mathcal{I}$ ist. Das heißt, $\mathcal{I} = \{i_1, i_2, \ldots, i_n\}$ ist eine Menge von Binärattributen $\{0, 1\}$. Jede Transaktion ist mit einem eindeutigen Schlüssel verknüpft, genannt $\mathcal{TID}$. Eine Menge heißt Elementmenge wenn $\mathcal{X} \subset \mathcal{I}$ ist. Eine Transaktion $\mathcal{T}$ enthält eine Elementmenge $\mathcal{X}$, wenn $\mathcal{X} \subseteq \mathcal{T}$ ist. Eine Assoziationsregel ist eine Implikation der Art $\mathcal{X} \Rightarrow \mathcal{Y}$, für $\mathcal{X} \subset \mathcal{I}$, $\mathcal{Y} \subset \mathcal{I}$ und $\mathcal{X} \cap \mathcal{Y} = \emptyset$.*

Definition 3.6 (Konfidenz (*confidence*)) *Die Regel $\mathcal{X} \Rightarrow \mathcal{Y}$ gilt in der Transaktionsmenge $\mathcal{D}$ mit der Konfidenz c, wenn in c Prozent der Transaktionen aus $\mathcal{D}$, die $\mathcal{X}$ enthalten auch $\mathcal{Y}$ enthalten ist.*

Definition 3.7 (Abdeckung (*support*)) *Die Regel $\mathcal{X} \Rightarrow \mathcal{Y}$ gilt in der Transaktionsmenge $\mathcal{D}$ mit der Abdeckung s, wenn in s Prozent der Transaktionen aus $\mathcal{D}$, die $\mathcal{X} \cup \mathcal{Y}$ enthalten ist.*

Eine typische Anwendung von Assoziationsregeln ist die Analyse von Warenkörben aus Supermärkten. Hier sind Regeln der Art, dass 34% der Kunden, die Fisch kaufen, kaufen auch Weißwein. 34% ist in dem Beispiel die Konfidenz der Regel. Von allen, bezogen auf die Gesamtanzahl der Kunden, trifft die Regel für 5% der Kunden zu, dies ist die Abdeckung der Regel.

3.7.2.2 Generalisierte Assoziationsregeln

Ein Nachteil von booleschen Assoziationsregeln ist, dass sie auf Regeln zwischen den Elementen beschränkt sind. Durch das Ausnutzen von Begriffshierarchien, Taxonomien, die eine Halbordnung über die Elemente definieren, wie zum Beispiel eine ist–Teil–von–Hierarchie, können sogenannte generalisierte Assoziationsregeln gefunden werden [Srikant und Agrawal 1996a]. Solche Taxonomien hängen jeweils von der Anwendungsdomäne ab. In der Anwendungsdomäne Produktbewährung ist das Schadensschlüsselbuch zum Beispiel eine solche Begriffstaxonomie, da die einzelnen Teile wieder zu funktionellen Einheiten zusammengefasst werden. Ein weiters effizientes Verfahren (Prutax) zur Erkennung von generalisierten Assoziationsregeln stammt von [Hipp u. a. 1998].

3.7.2.3 Quantitative Assoziationsregeln

Die bisher vorgestellten Assoziationsregeln berücksichtigen nicht die quantitativen Aspekte. Attribute in realen Anwendungen sind im allgemeinen komplexer als vom Typ Boolean. Diese Attribute enthalten quantitative oder symbolische Werte. Symbolische Attributtypen lassen sich leicht auf boolesche Attribute abbilden. Bei quantitativen Attributen ist dies schwieriger, hier muss eine Partitionierung in Intervalle erfolgen, die quantitativen Attribute werden diskretisiert. Quantitative Assoziationsregeln wurden erstmals von [Srikant und Agrawal 1996a] vorgestellt. Ein effizientes Verfahren (Q2) zur Erkennung solcher Assoziationsregeln wurde 1998 von Büchter und Wirth vorgestellt [Büchter und Wirth 1998].

3.7.2.4 Sequentielle Muster

In vielen Anwendungen müssen auch die zeitlichen Zusammenhänge berücksichtigt werden. Zum Beispiel könnte es von Interesse sein, ob eine Reparatur an einem Teil einen späteren Ausfall impliziert. Um zeitliche Aspekte berücksichtigen zu können, muss jede Transaktion zusätzlich eine Zeitmarke enthalten.

Definition 3.8 (Sequenz) *Eine Sequenz $\mathcal{S}$ ist eine endliche Folge von Transaktionen, die aufsteigend nach ihren Zeitmarken sortiert ist.*

$$\mathcal{S} = \{\mathcal{T}_1, \ldots, \mathcal{T}_n\} \tag{3.6}$$

In der Anwendungsdomäne Produktbewährung können die einzelnen Werkstattaufenthalte Transaktionen darstellen.

Ein sequentielles Muster ist nun eine Folge von Elementen

$$\mathcal{M} = \{i_1, i_2, \ldots, i_k\}. \tag{3.7}$$

Das Ereigniss i_i fand zeitlich vor dem Ereignis i_{i+1} statt. Die weiteren zeitliche Bedingungen für ein Muster sind

- $\triangle_p$ ist die maximale Länge einer Teilsequenz,
- $\triangle_{min}$ ist die minimale zeitliche Distanz zwischen zwei aufeinanderfolgenden Elementen und
- $\triangle_{max}$ die maximale Distanz.

Der Basisalgorithmus (GSP) wurde von [Srikant und Agrawal 1996b] entwickelt, während die Erweiterung (eGSP)von [Klenk 1997] implementiert und im Projekt verifiziert wurde.

3.8 Einsatzkonzepte der Verfahren für den Problemtyp I

3.8.1 Klassifikation

Klassifikationsverfahren sind in dieser Anwendung grundsätzlich einsetzbar. Einfach betrachtet handelt es sich um ein Zwei–Klassenproblem. Von Interesse ist in diesem Fall aber nicht die Vorhersagegüte, sondern die Beschreibung der Klassen, bzw. die Regeln, die gelernt werden. In der Praxis haben sich die Verfahren wie C4.5, die eingesetzt wurden, als nicht so leicht anwendbar erwiesen.

Problem: Im Allgemeinen liegt die Beanstandungsquote der Fahrzeuge bei wenigen Prozent. Dies hat zur Folge, dass der einfachste Klassifikator per Default auf „keine Beanstandung“klassifiziert. Es ist also notwendig, das Verhältnis der Klassen zu verändern, um eine Klassenbeschreibung zu erhalten.

Tests und Vorstudien führten die Entscheidungsbaumlerner zu nicht befriedigenden Ergebnissen, während Neuronale Netze aufgrund der fehlenden Interpretierbarkeit des Modells nicht berücksichtigt wurden.

3.8.2 Einsatzmöglichkeiten von INES

Einen Überblick zum Einsatz von INES zur Analyse im PBP findet man beschrieben von Borgelt, Kruse und Lindner in [Borgelt u. a. 1998].

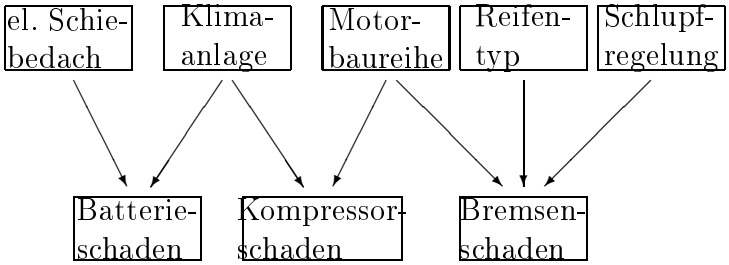

Abbildung 3.8: Ein Ausschnitt eines fiktiven zweischichtigen Netzes, das die Abhängigkeiten zwischen Schäden/Fehlern (untere Schicht) und Bauzustandsmerkmalen (obere Schicht) beschreibt.

(fiktive) Häufigkeit von Batterieschäden		Klimaanlage mit	Klimaanlage ohne
elektrisches Schiebedach	mit	9 %	3 %
	ohne	3 %	2 %

Abbildung 3.9: Ein fiktives Teilnetz, das die Abhängigkeit eines Batterieschadens vom Vorhandensein eines elektrischen Schiebedaches und einer Klimaanlage beschreibt.

Die verfolgte Idee ist sehr einfach. Da man an Ursachen von Fehlern interessiert ist, wird ein zweischichtiges probabilistisches Netzwerk gelernt, dessen obere Schicht diejenigen Attribute enthält, die den Bauzustand eines Fahrzeugs beschreiben, während die Attribute in der unteren Schicht mögliche Schäden oder Fehler wiedergeben. Abbildung 3.8 zeigt ein mögliches zweischichtiges Netzwerk, Abbildung 3.9 die Häufigkeitsverteilung, die zu seinem ersten Teilnetz gehört. Da in diesem Beispiel die Batterieschadensrate für Fahrzeuge mit Klimaanlage und elektrischem Schiebedach deutlich höher ist als für solche mit keinem oder nur einem dieser Ausstattungsmerkmale, kann man vermuten, dass der durch sie hervorgerufene erhöhte Stromverbrauch zu häufigeren Batterieausfällen führt.

Für diese Methode wurden zwei Einsatzmöglichkeiten angedacht, mit denen die Produktbetreuer unterstützt werden können.

1. Bauzustandsabhängigkeiten

2. Beanstandungsabhängigkeiten

Neben dem Finden von Bauzustandsabhängigkeiten auf Basis der Verkaufscode, wie sie in QUIS hinterlegt sind, ist bei der Suche nach der Schadensursache die erste Frage, welche Fahrzeuge betroffen sind. Hier bezieht man in den ersten Untersuchungen nicht die Verkaufscode ein, sondern bezieht sich auf charakteristische Baumerkmale, wie Motor- und Getriebeausführungen, Baureihenvarianten usw. .

Eine weitere Fragestellung, die auch aus Sicht der Produktbetreuer interessant ist, ist die Frage nach Abhängigkeiten zwischen einzelnen Werkstattaufenthalten. Treten Beanstandungen in Kombination auf? Mit dem bisher beschriebenen Ansatz erhält man allerdings keine Aussage über die zeitlichen Zusammenhänge. Sind die Beanstandungen direkt aufeinanderfolgend? Wieviel Zeit liegt zwischen den einzelnen Beanstandungen? Solche Zeitreihen–Untersuchungen wurden mit eGSP (siehe Abschnitt 3.8.3 und 4.3) untersucht, während für die beiden ersten Fragestellungen das Programm INES eingesetzt wurde.

Mit der Auswahl von INES als Werkzeug wurde die Methode noch nicht eindeutig festgelegt. Die in Abschnitt 3.7.1 aufgezählten Bewertungsmaße sind alle potentiell geeignet, das gewünschte Ergebnis zu erzielen [Borgelt und Kruse 1998]. Auch für diese spezielle Anwendung konnten in einer experimentellen Studie keines der oben aufgeführten Bewertungsmaße als immer geeignet identifiziert werden [Zintz 1996].

3.8.3 Einsatz von sequentiellen Mustern

Wie schon in Abschnitt 3.8.2 aufgezeigt, sind auch weitere Fragestellungen, neben der Frage nach ungünstigen Bauzuständen, für den Bereich der Produktbetreuung von Interesse. Für die Anwendung im PBP bot sich die erweiterte Implementierung von sequentiellen Mustern von Klenk an [Klenk 1997]. Diese Realisierung von sequentiellen Mustern (GSP) nach [Srikant und Agrawal 1996b] wurde von Klenk um die Ausnutzung von Taxonomien erweitert und wird um folgenden eGSP genannt. Das Schadensschlüsselbuch zur Vercodung der Schäden enthält eine hierarchische Struktur in Form einer *ist–Teil–von* Beziehung, die mit eGSP berücksichtigt wird. Mit der Ausnutzung der Taxonomie kann auch die Problematik der Fehlvercodung aus der Anforderungsanalyse (vgl. Abschnitt 3.4) berücksichtigt werden. Durch die Berücksichtigung von funktionellen Einheiten, wie sie im Schadensschlüsselbuch beschrieben sind, können Falschverschlüsselungen in der Analyse berücksichtigt werden. Bei Falschverschlüsselungen kann man die Annahme treffen, dass das verursachende Teil zumindest zu der gleichen Funktionseinheit gehört.

In diesem Zusammenhang können zum Beispiel auch Fragen des Controllings von Interesse sein, z.B. wie gut ist die Reparatur an einer Funktionseinheit. Oder aus einer technischen Perspektive, wie im PBP, ist die Frage interessant, ob eine bestimmte Reparatur, die Wahrscheinlichkeit für einen Fehler erhöht.

In der Tabelle 3.1 und 3.2 sind Ergebnisse aus der Analyse dargestellt, die für die oben genannten Fragestellungen für eine Baureihe durchgeführt wurden. Für diese Untersuchung standen zwei Produktionsjahre und sechs Jahre Werkstatthistorie zur Verfügung. Tabelle 3.1 zeigt Ergebnisse zu der Frage nach Wiederholungsschä-

den, d. h. ob ein Kunde mit dem *gleichen* Schaden noch einmal in die Werkstatt kommt. Um auszuschließen, dass es sich bei der zweiten Beanstandung um denselben Schaden handelt, wurde das Verfahren so parametrisiert, dass mindestens 30 Tage zwischen den Werkstattaufenthalten liegen mussten.

Aggregatgruppe	tritt auf mit	dann nochmal mit	Abstand Tage (min 30)	Tkm
401	5,2 %	11,7 %	140	15
301	2,5 %	12,1 %	150	18
103	4,3 %	10,0 %	143	29
101	5,5 %	9,3 %	164	15

Tabelle 3.1: eGSP: Wiederholungsschäden

Die Wahrscheinlichkeit, dass ein Schaden an dem Teil nochmals auftritt ist hier jeweils wesentlich höher als zu erwarten. Wenn man davon ausgeht, dass ein Schadensteil nach Reparatur *so gut wie neu* ist, darf die Wahrscheinlichkeit höchstens die gleiche sein wie die Grundwahrscheinlichkeit für die Beanstandung.

Desweiteren wurden Ergebnisse für die zweite Fragestellung gefunden (siehe Tabelle 3.2:

Aggregatgruppe	tritt auf mit	wird fortgesetzt von	mit der Wahrsch.	erwartete max. Wahrsch.
601	3,6 %	301	11,0 %	4,5 %

Tabelle 3.2: eGSP Ergebnisse: Abhängigkeiten zwischen Beanstandungen

Sind die Beanstandungen unabhängig von einander, dann darf die Fortsetzungswahrscheinlichkeit maximal so hoch sein, wie die Auftrittswahrscheinlichkeit des fortsetzenden Ereignisses alleine.

In dem in Tabelle 3.2 dargestellten Ergebnis ist die Fortsetzungswahrscheinlichkeit $11,0\%$, während die Grundwahrscheinlichkeit für das Eintreten des Folgeereignisses nur $4,5\%$ beträgt.

Diese und ähnliche Fragestellungen wurden bei weiteren Fahrzeugtypen untersucht, in der die zeitliche Beziehung zwischen einzelnen Beanstandungen untersucht wurde. Die Ergebnisse sind im Abschnitt 4.3 dargestellt und auch unter [Lindner und Hipp 1999] veröffentlicht.

3.9 Trendprognose für den Produktbewährungsprozess

Neben der Frage nach ungünstigen Bauzuständen wurden in der Anforderungsanalyse (siehe Abschnitt 3.4) weitere Anforderungen identifiziert. Diese Anforderungen beziehen sich auf die Verbesserung der Produktbeobachtung mit dem Ziel früher auf Schadensschwerpunkte reagieren zu können. In den folgenden Abschnitten wird der hierzu entwickelte Ansatz, die potentiell einsetzbaren Verfahren und die Auswahl der Verfahren beschrieben.

Der hierzu entwickelte Ansatz wurde erstmals von [Lindner und Klose 1997] und [Lindner und Studer 1999b] mit weiteren Experimenten und einem implementierten Prototyp, der in das kommerzielle KDD–Tool Clementine™ [8] integrierte wurde, vorgestellt. Zum Abschluss des Projektes wurden Mitarbeiter der Fachabteilungen (Produktbetreuer) in der Methode an dem Tool geschult.

Der Ansatz zur Trendprognose ist durch die Arbeit von [Legner u. a. 1996] motiviert. Diese Studie hatte als Hauptergebnis, dass die Beanstandungskurven für die unterschiedlichen Zeitpunkte des Fahrzeugalters dieselbe Form haben. Diese Beobachtung zeigt, dass Probleme (z. B. steigende Trends) schon früher erkannt werden können. Dieses Ergebnis führte zu einer Änderung der Strategie der Prognose. Anstatt eine Vorhersage für Fahrzeuge einer Altersgruppe (z. B. 6 Monate)[9] zu machen, werden nun die zukünftigen Garantie- und Kulanzkosten für die Fahrzeuge einer Produktionsperiode[10] vorhergesagt und diese Prognose für die Kalkulation der gesamten Garantie- und Kulanzkosten benutzt.

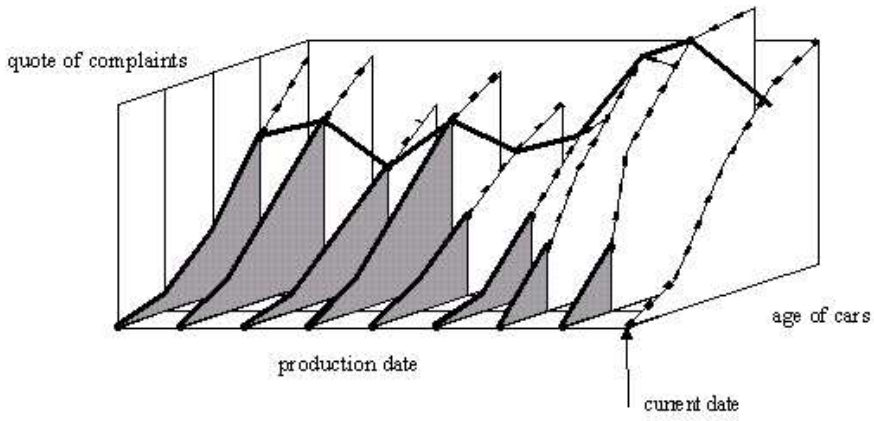

Abbildung 3.10: Ansatz nach Legner/etal

Der wesentliche Unterschied zwischen dem Ansatz zur Trendprognose von [Lindner und Klose 1997, Lindner und Studer 1999a] und der Arbeit von

[8] Zum Zeitpunkt der Implementierung gab es eine strategische Entscheidung für dieses Tool bei der DaimlerChrysler AG

[9] Dies entspricht einer „horizontalen“Vorhersage in der Abbildung 3.10

[10] Dies entspricht einer „vertikalen“Vorhersage in der Abbildung 3.10

[Legner u. a. 1996] ist, dass kumulierte Kosten für einzelne Baureihen berechnet werden. Der entwickelte Trendprognoseansatz hat aber zum Ziel, Vorhersagen über die Entwicklung der Beanstandungen von einzelnen Teilen der Fahrzeuge, bzw. über sinnvolle Gruppen von Teilen (siehe auch Abschnitt 3.10), möglichst zu einem sehr frühen Zeitpunkt zu machen.

Der zweite wesentliche Unterschied ist, dass in unserem Ansatz die Veränderungen während der Lebenszeit und dem Produktionszeitpunkt nicht separat, sondern die Kombination der horizontalen und vertikalen Richtungen gleichzeitig auszunutzen, wie es in der Abbildung 3.11 skizziert ist. Die Idee ist, alarmierende Trends durch die schon gegebenen Muster aus den frühen Daten abzuleiten.

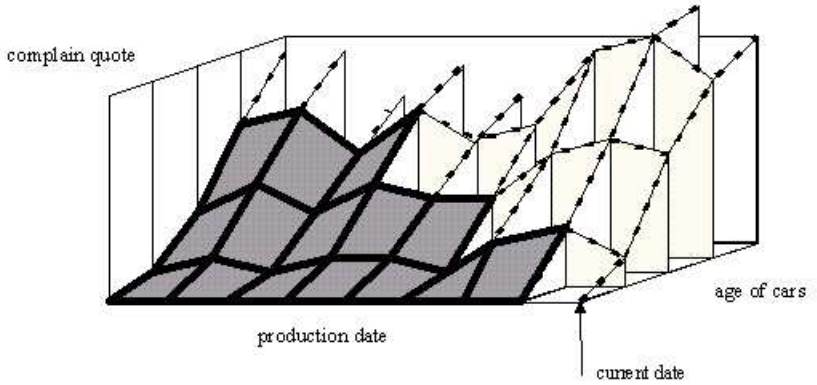

Abbildung 3.11: Kombination der horizontalen und vertikalen Richtung

Im Rahmen unserer Arbeit mussten wir allerdings in der Phase des Datenverständnises feststellen, dass ausgehend von mehreren hundert Megabyte Daten nach einer Aufsplittung auf die Baureihen und Schäden zuwenig Daten für die Data Mining Verfahren übrig blieben, um gesicherte Vorhersagen machen zu können. Ein Auszug über eine PKW–Baureihe über 2 Jahre enthält Daten über 54000 Fahrzeuge. Von den über 5000 möglichen Beanstandungen kamen nur noch ca. 2600 Schäden vor, von denen 600 einen Anteil von mehr als 0,1% hatten. Die Abbildung 3.12 zeigt, wie sich die Schäden auf die einzelnen Komponenten verteilen. Außer diesen aus den Daten motivierten Gründen, gibt es folgende Gründe aus der Anwendung für eine Clusterbildung [11].

- Die Gruppenbildung von sinnvollen Clustern reduziert die Komplexität und erhöht die Interpretierbarkeit für den Experten.
- Die Gruppenbildung kann das Problem der Falschvercodung in den Werkstätten reduzieren. Wenn ein Teil fälschlich als schadensverursachendes Teil vercodet wurde, so steht das wirklich verursachende Teil in örtlicher oder funktioneller Beziehung zu dem vercodeten Teil, d.h. es ist im gleichen Cluster.

[11] Gruppenbildung

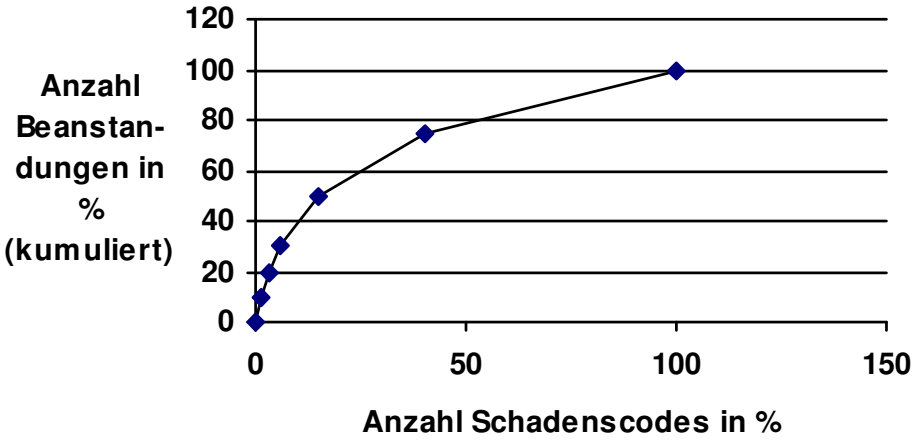

Abbildung 3.12: Verteilung der Schäden

- Die Veränderung der Clusterstruktur kann ein Indikator für eine neue Art von Schaden sein.

- Mittels der Cluster ist es möglich auch Gruppen von Teilen zu beobachten.

Aus diesem Grund wurde eine Clusteranalyse als Pre-Processing Schritt eingeführt. Ein solch komplexer Vorgang definiert für sich eine eigene Data Mining Aufgabe. Dieser Sachverhalt wurde in der Betrachtung des KDD–Prozesses in Abschnitt 2 schon beschrieben. An diesem Beispiel aus der Praxis kann man sehen, wie komplex sich eine Data Mining Aufgabe gestalten kann.

Der Ansatz zur Trendprognose besteht nun aus in Abbildung 3.13 dargestellten Teilaufgaben.

Aus diesem Ansatz heraus ergeben sich zwei Gruppen von Verfahren, aus denen je ein geeignetes Verfahren gewählt werden muss.

1. ein Clusterverfahren

2. ein Verfahren zur Vorhersage

In den nächsten Abschnitten werden nun die beiden Gruppen von Verfahren beschrieben und welche Methoden für die gegebene Aufgabe ausgewählt wurden. Neben dem Punkt, welche Methode ausgewählt wurde, steht die Frage wie wird die Methode in der Praxis ausgewählt werden, im Mittelpunkt.

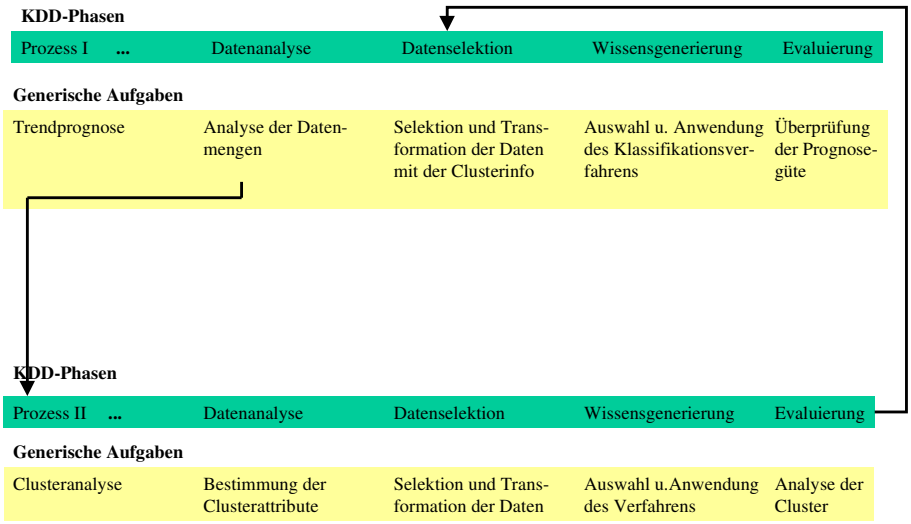

Abbildung 3.13: Aufgabenzerlegung im Ansatz der Trendprognose

3.10 Clusteranalyse

3.10.1 Arten von Clusterverfahren

In der Clusteranalyse[12] wird versucht, eine Menge von Objekten in einem vorher definierten Sinne zu gruppieren. Im maschinellen Lernen, wie es in Abbildung 3.14 skizziert ist, wird das Clustering zur Gruppe der nicht überwachten Lernverfahren gezählt.

Im Gegensatz zu den überwachten Lernverfahren (vgl. Abschnitt 3.6) bestehen bei den unüberwachten Lernverfahren die Lernaufgaben immer nur aus einer Menge von Eingabedaten ohne einer bekannten Klassenzugehörigkeit. Die Clusteranalyse stellt das Pendant zur Klassifikation im Bereich des nicht überwachten Lernens dar. Hier sind die Ausgabeklassen zuvor nicht bekannt und sollen vom Verfahren gefunden werden. Nach der Festlegung der Klassen durch das Verfahren, sollen weitere, noch unbekannte Eingabemuster in die gefundenen Klassen eingeordnet werden können. In der Literatur ist für Clusteranalyse daher auch der Name „automatische Klassifikation“ geläufig.

Die zu clusternden Objekte liegen üblicherweise in einem hochdimensionalen Raum, der durch die Attribute aufgespannt wird. Die Attributausprägungen bestimmen die Positionen der Objekte in diesem Raum. Von dieser Betrachtungsweise ausgehend, werden die Attribute des Objekts auch als Merkmalsvektor bezeichnet.

[12]Die Begriffe Clustering und Clusteranalyse werden in dieser Arbeit synonym verwendet

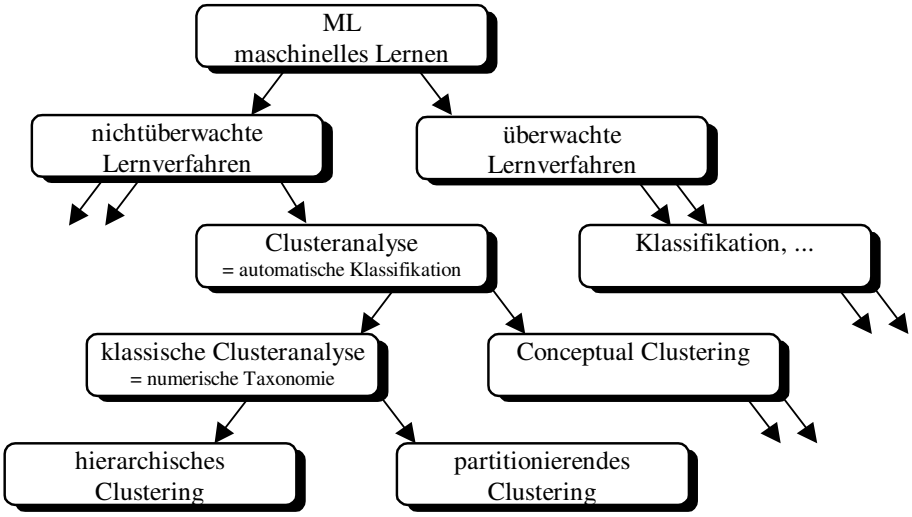

Abbildung 3.14: Clustering im maschinellen Lernen

Mit der Clusteranalyse werden im Allgemeinen zwei Zielsetzungen verfolgt:

1. Finden einer beschreibenden oder vereinfachenden Zusammenfassung. Hierzu wird eine Abstandsfunktion definiert, über die eine Ähnlichkeit zwischen zwei Objekten bestimmt werden kann. Ziel der Clusteranalyse ist es, im Sinne der Abstandsfunktion ähnliche Objekte in gleiche Cluster und unähnliche in verschiedene Cluster einzuordnen. Das Ergebnis ist dann eine Einteilung der Objekte in Cluster.

2. Bei der zweiten Motivation wird von der Vorstellung ausgegangen, dass die Daten aus Zufallsexperimenten einer multivarianten Mischverteilung stammen oder einfach den unterschiedlichen Klassen verschiedene Verteilungen zugrunde liegen. Diese Mischverteilung setzt sich aus einer - in der Regel unbekannten - Anzahl von einzelnen Verteilungen zusammen. Jede Verteilung spiegelt die Ausprägungen einer Klasse wider. Die Objekte sind Stichproben der einzelnen Verteilungen und werden daher auch Beobachtungen genannt. Mit Hilfe der Clusteranalyse wird versucht, die Anzahl und die Daten der einzelnen Verteilungen zu bestimmen und die Beobachtungen den gefundenen Klassen zuzuordnen. Wesentliches Ergebnis ist dabei weniger die Einteilung der Objekte, als vielmehr die Beschreibung der Klassen.

Bei der Clusteranalyse wird zwischen den klassischen Verfahren, die auch als numerische Taxonomie bezeichnet werden, und neueren Ansätzen des Conceptual Clusterings unterschieden. Die klassischen Methoden werden seit den 60er Jahren in der Statistik erforscht und beschränken sich darauf, die Objekte in Gruppen zu vereinen. Im Gegensatz dazu versuchen die Verfahren des Conceptual Clusterings, Beschreibungen für die Gruppen zu finden. Sie spielen im Rahmen des Data Mining eine

größere Rolle, da die Beschreibung der Datenstrukturen eine Form der Wissensextraktion darstellt. Diese gefundenen Beschreibungen, auch Konzepte genannt, haben meist die Form von Hyperquadern im Eingaberaum, die durch die Konjunktion auf den Attributen entstehen. Als bekannte Verfahren wären hier beispielhaft ECOBWEB [Reich 1994], UNIMEM [Lebowitz 1986] oder COBWEB [Fisher 1987] zu nennen. Da diese Verfahren aber für die Anwendung nicht von Bedeutung sind (siehe Abschnitt 3.12) soll hier auch nicht weiter darauf eingegangen werden. Eine Beschreibung des Conceptual Clusterings findet man in [Gordon 1999].

Die klassischen Verfahren werden weiter nach ihrer Arbeitsweise unterteilt,

1. in hierarchische und
2. partitionierende Methoden.

Die partitionierenden Methoden setzen voraus, dass die Anzahl k der Klassen a priori bekannt ist. Es wird dann eine Partition, also disjunkte Einteilung der Objekte in k Klassen bestimmt. Die hierarchischen Methoden arbeiten sukzessiv, indem in jedem Schritt je nach Vorgehensweise zwei Cluster vereinigt werden oder einer aufgespaltet wird. Die sogenannten *agglomerativen* hierarchischen Clustermethoden beginnen mit der trivialen Einteilung der n Objekte in n Cluster mit einem Objekt. Mit jedem Schritt werden je zwei Cluster vereinigt, bis schließlich nur ein einziger Cluster übrig bleibt, der alle Objekte enthält. Die Methoden, die mit einem n-elementigen Cluster beginnen und diesen schrittweise bis zur trivialen Einteilung in n Einzelobjekte aufspalten, heißen *divisiv*. Wegen der geringeren praktischen Bedeutung der divisiven Algorithmen wird jedoch nur auf die agglomerativen eingegangen. Beide Varianten bilden im Gegensatz zu den partitionierenden Methoden nicht nur eine einzige Partition, sondern in Form einer Hierarchie eine Folge von geschachtelten Partitionen. Bei Bedarf kann im Nachhinein aus der gefundenen Hierarchie durch den Schnitt auf einer Ebene wieder eine Partition gewonnen werden.

Die partitionierenden Clusteringverfahren arbeiten zwar auch mit Heuristiken, versuchen jedoch nur eine Partition statt des Partitionierungsprozesses zu optimieren und liefern so häufig ausgewogenere Gruppierungen. Daher werden sie oft eingesetzt, wenn das Ziel in der Bildung lediglich einer Partition besteht. Um mit dem Problem der vorher festzulegenden, aber unbekannten Klassenanzahl umgehen zu können, werden getrennte Clusteranalysen für ein Intervall erwarteter Klassenanzahlen durchgeführt. Die Stabilität und Qualität der Ergebnisse wird dann anhand verschiedener Kriterien überprüft und die beste Partition ausgewählt. Eine andere Vorgehensweise ist das Clustering mittels hierarchischer Methoden zur Festlegung einer Klassenanzahl und die anschließende Optimierung der erhaltenen Gruppierungen mit einem partitionierenden Verfahren.

Die Verfahren der Clusteranalyse lassen sich wie zuvor beschrieben anhand von verschiedenen Kriterien unterteilen. In der Tabelle 3.3 sind die Unterscheidungsmerkmale noch einmal zusammengefasst (vgl. auch [Grimmer und Mucha 1998]).

Gruppierungsform	hierarchisch	vs.	partitionierend
Gruppierungsprozess bei hierarch. Verfahren	agglomerativ	vs.	divisiv
Zuordnungsprinzip	disjunkt exhaustiv	vs. vs.	nicht disjunkt nicht exhaustiv

Tabelle 3.3: Unterscheidungskriterien in der Clusteranalyse

3.10.2 Clusteringalgorithmen

Im Folgenden werden die Grundprinzipien der partitionierenden und hierarchischen Clusteringalgorithmen erläutert. Neben der grundlegenden Vorgehensweise werden die verschiedenen Varianten und die Auswirkungen dieser Varianten auf das Clusteringergebnis angesprochen.

3.10.2.1 Partitionierendes Clustern

Die partitionierenden Clusterverfahren suchen jeweils nur eine Partitionierung der Daten in eine feste Anzahl von Klassen. Es gibt im wesentlichen zwei Varianten, die Minimaldistanzmethode und die Radiusrestriktion.

Minimaldistanzmethode Die Minimaldistanzmethode und ihre Varianten liefern für eine feste Anzahl von Klassen meistens eine bessere Gruppierung als die hierarchischen Methoden oder auch die Radiusrestriktion. Dies ist im Wesentlichen darauf zurückzuführen, dass in den letzteren eine einmal vollzogene Fusion nicht mehr rückgängig gemacht werden kann, während bei den erst genannten Methoden im Verlauf des Verfahrens die Objekte ihre Klassenzugehörigkeit wechseln können.

Die bekanntesten Verfahren sind die Minimaldistanzmethode sowie das Austauschverfahren. Sie unterscheiden sich in dem zu optimierenden Kriterium. Beide Verfahren gehen von einer Startpartition, also einer initialen Einteilung der Objekte in die Klassen, aus. Diese kann einer Voranalyse entstammen, etwa einem hierarchischen Clustering oder einer statistischen Hauptkomponentenanalyse mit Einteilung der Hauptkomponente in Intervalle. Auch eine rein zufällige Wahl der Startklassifikation ist möglich. Allerdings ergeben unterschiedliche Anfangspartitionen auch unterschiedliche Klassifikationsergebnisse. Das liegt daran, dass ausgehend von der anfänglichen Partition ein lokales Optimum des Klassifikationskriteriums angestrebt wird.

Das Verfahren beginnt mit der Bestimmung eines Repräsentanten für jede Klasse der Startpartition. Danach werden die folgenden Schritte solange iteriert, bis ein Abbruchkriterium erreicht wird. Abbruchkriterien können die Anzahl der Iterationszyklen oder die Angabe einer geforderten Mindeständerung pro Durchlauf sein.

- In jedem Durchgang wird jedes der Objekte betrachtet und der bezüglich des gewählten Kriteriums optimalen Klasse zugeordnet.
- Nach der Zuordnung werden die Klassenrepräsentanten aktualisiert. Als Repräsentanten werden üblicherweise die Zentroide der Klassen verwendet.

Beim Minimaldistanzverfahren wird das Objekt in diejenige Klasse eingeordnet, zu der es den kleinsten euklidischen Abstand hat. Beim Austauschverfahren wird geprüft, ob sich die Summe der klasseninternen Varianzen durch den Wechsel verkleinert. Wenn dies der Fall ist, wird der Wechsel durchgeführt. Bei beiden Verfahren wird durch einen Wechsel die Summe der klasseninternen Varianzen verkleinert und so ein lokales Minimum dieser Größe angestrebt, allerdings minimiert das Minimaldistanzkriterium diesen Wert nur indirekt.

Methode der Radiusrestriktion Diese Methode ist die schnellste der hier vorgestellten. Sie benötigt lediglich einen Durchlauf durch die Daten. Es wird nicht direkt eine Klassenanzahl festgelegt, sondern ein Maximalradius R für die Clustergröße. Vor Beginn des Durchlaufs wird die erste Klasse mit der ersten Beobachtung initiiert. Diese wird auch Kernobjekt der Klasse genannt. Es werden dann die Beobachtungen 2 bis n durchlaufen und jeweils geprüft, zu welcher Klasse sie den kleinsten Abstand hat. Wenn diese Distanz kleiner als der Grenzradius R ist, wird sie in diese Klasse eingeordnet. Anderenfalls bildet die Beobachtung das Kernobjekt einer neuen Klasse.

Vorteil dieses Verfahrens ist, dass für die Herleitung kein euklidischer Vektorraum vorausgesetzt wird. Es können so beliebige Distanzen verwendet werden. Problematisch ist die Wahl des Radius. Ein weiterer Nachteil ist die Abhängigkeit des Ergebnisses von der Reihenfolge der Beobachtungen. Da bei diesem Verfahren ebenfalls keine nachträglichen Klassenwechsel zugelassen werden, sind die Clusterergebnisse schlechter als die der anderen partitionierenden Verfahren.

3.10.2.2 Hierarchisches Clustern

Das hierarchische agglomerative Clustern basiert auf der sukzessiven Zusammenfassung der Zeilen der Datenmatrix. Häufig wird für eine effiziente Durchführung der Analyse eine Distanzmatrix aufgestellt, die die paarweisen Distanzen aller Objekte enthält. Wichtiger Punkt ist die Verallgemeinerung des Abstandsmaßes auf Abstände zwischen Clustern. Die Algorithmen unterscheiden sich in der Art und Weise, wie diese erweiterte Abstandsbestimmung geschieht.

Der eigentliche Algorithmus besteht aus drei Phasen:

1. Bestimmung der zwei Klassen i und j mit dem kleinsten Abstand. Sind die Abstände in einer Distanzmatrix gespeichert, entspricht Schritt 1 der Suche nach dem kleinsten Element der Matrix. Eventuell werden zusätzlich die Massen der Cluster an dieser Stelle berücksichtigt (Ward'sches Verfahren).

2. Vereinigung der beiden in 1 gefundenen Klassen i und j zu einer Klasse k*. Dabei fällt eine Zeile und eine Spalte der Distanzmatrix weg. Berechnung der neuen Masse als Summe der beiden Einzelmassen.

3. Aktualisierung der Distanzmatrix mit den Abständen zwischen jeweils der neuen, vereinigten Klasse k* und allen anderen Klassen k.

Im Abschnitten 3.11 werden mögliche Methoden zur Berechnung von Abständen und Klassenvereinigung beschrieben.

3.10.3 Stabilitätsuntersuchungen

Ein großes Problem der Clusterverfahren besteht darin, dass diese auch dann eine Partition oder Hierarchie finden, wenn den Daten gar keine Klassenstruktur zugrunde liegt. Durch geeignete Tests wird daher versucht, das Klassifikationsergebnis zu verifizieren. Ein Teilbereich der Verifikation prüft die Korrektheit der Klassenanzahl.

In der hierarchischen Clusteranalyse erfolgt keine a priori Festlegung auf eine bestimmte Klassenanzahl. Im Agglomerationsprozess werden alle Klassenanzahlen von n Klassen zu je einem Objekt bis zu einer Trivialklasse mit allen Objekten durchlaufen. Es kann das Distanzniveau, auf dem die Cluster vereinigt werden, während des Agglomerationsprozesses im Dendrogramm beobachtet werden. Ungünstige Zusammenfassungen zeigen sich durch große Sprünge im Vereinigungsniveau.

Bei der partitionierenden Clusteranalyse besteht das Verfahren zur Bestimmung der Klassenanzahl darin, für ein Intervall verschiedener möglicher Klassenanzahlen getrennte Analysen durchzuführen. Für die Klassenanzahlen kann das minimierte Clusteringkriterium betrachtet werden. Die Kurve wird dann auf Unregelmäßigkeiten geprüft. Im Gegensatz zu den beiden genannten Möglichkeiten, die auf der Beobachtung eines Kurvenverlaufs beruhen, besteht eine andere Möglichkeit darin, Hypothesen über die Struktur aufzustellen und zu versuchen, diese in statistischen Tests zu widerlegen. Mögliche Hypothesen sind etwa die Nullhypothese *Den Daten liegt keine Klassenstruktur zugrunde* oder eine Hypothese *Den Daten liegen k Klassen zugrunde*. Für die Tests solcher Hypothesen können entweder die zum Clustern verwendeten oder neue Daten benutzt werden.

Wenn der Schwerpunkt der Clusteranalyse auf die Bildung einer deskriptiven Zusammenfassung der Daten gesetzt wird, und das Ziel nicht in erster Linie die Enthüllung der in den Daten enthaltenen Strukturen ist, spielt auch die Validierung eine kleinere Rolle. In solchen Fällen kann eine Vorgehensweise auch darin bestehen, die Clusteranalyse mit leicht veränderten Werten oder Abstandsfunktionen zu wiederholen und zu überprüfen, ob und wie stark sich die Ergebnisse verändern. Die Bewertung der Übereinstimmung der einzelnen Läufe kann mit Hilfe von Koinzidenzmatrizen geschehen. Bei dieser Vorgehensweise in Verbindung mit einem hierarchischen Clustering mit flexibler Strategie kann auch deren Parameter leicht modifiziert werden, um zu überprüfen, wie zufällig oder stabil die Klassifikation ist.

Zusätzlich zu den angesprochenen Verfahren ist es immer angebracht, eine Interpretation der Ergebnisse vorzunehmen und diese mit den intuitiv erwarteten Strukturen zu vergleichen.

3.11 Ähnlichkeitsberechnung

3.11.1 Abstandsberechnung zwischen Objekten

Basis für die hierarchischen Verfahren ist die Bestimmung der Ähnlichkeiten zwischen den Objekten. Da die Berechnung von Ähnlichkeiten im Teil III von zentraler Bedeutung ist, wird an dieser Stelle schon einmal detailliert auf dieses Thema eingegangen.

Zur Bestimmung der Ähnlichkeit bzw. Unähnlichkeit zwischen Objekten werden Abbildungen definiert, die jeweils zwei Objekten der Grundmenge einen reellen Wert zuordnet, der deren Beziehung quantifiziert. Entsprechend heißen die Abbildungen Ähnlichkeits- bzw. Unähnlichkeitsmaße. Ähnlichkeits- und Unähnlichkeitsmaße können ineinander überführt werden, erstere sind jedoch in der Clusteranalyse verbreiteter.

Definition 3.9 (Abstandsmaß:) *Seien a, b Objekte aus einer Grundmenge G. Formal ist eine Distanzfunktion d eine Abbildung aus* $GxG \in \mathcal{R}^+$*, die folgende Bedingungen erfüllt:*

- *Reflexivität:* $d(a, a) = 0$
- *Symmetrie:* $d(a, b) = d(b, a)$
- *Nichtnegativität:* $d(a, b) \geq 0$

Gilt darüber hinaus für drei Objekte a, b, c die Dreiecksungleichung $d(a, b)+d(b, c) \geq d(a, c)$*, so wird das Maß auch Metrik genannt. Metriken haben Eigenschaften, die für einige Berechnungen oder Interpretationen von Vorteil sind.*

Da die Attribute, mit deren Hilfe die Objekte charakterisiert werden, unterschiedlicher Art sein können, müssen geeignete Funktionen definiert werden. In den folgenden Abschnitten werden verschiedene Möglichkeiten für die unterschiedlichen Arten von Attributen beschrieben. Neben diesen sind in der Literatur auch weitere Maße bekannt, so dass die vorgestellten Maße einen Ausschnitt betrachten mit einem Fokus auf die Anwendung.

3.11.1.1 Numerische Attribute

Numerische Attribute sind Attribute, deren Wertebereich aus Zahlen besteht und bei denen Rechenoperationen wie Differenz- oder Quotientenbildung interpretierbare Größen ergeben. Dabei spielt es keine Rolle, ob es sich um reellwertige oder ganzzahlige Werte handelt. In der Literatur werden für numerische Attribute auch die Begriffe quantitaiv oder metrisch benutzt.

Für Merkmalsvektoren $x_i = (x_{i1}, x_{i2}, ..., x_{in})$ von numerischen Attributen wird eine Klasse von Abständen durch die Minkowski-Metrik definiert:

$$d_{ij}^{(p)} = \sqrt[p]{\sum_{k=1}^{n} w_k \left| x_{ik} - x_{jk} \right|^p} \, , p \in \mathcal{N} \tag{3.8}$$

Die w_k sind dabei Variablengewichte, mit denen der Einfluß einzelner Attribute auf das Clustering bestimmt werden kann. Wenn nicht anders angegeben, wird im folgenden $w_k = 1$ angenommen. Die Minkowski-Metrik legt je nach Parameter p ein verschiedenes Verhalten an den Tag. Die Unterschiede liegen in der Gewichtung der einzelnen Differenzen in den Dimensionen des Merkmalsraumes.

Die bekanntesten Sonderfälle sind $p = 1$, $p = 2$ und $p \to \infty$. Für $p = 2$ erhält man den euklidischen Abstand, der – zumeist quadriert verwendet – von großer Bedeutung in der Clusteranalyse ist. Wenn die Dimensionen in den gleichen Skalen vorliegen, kann der euklidische Abstand als Abstand im umgangssprachlichen Sinn interpretiert werden.

Die sich aus $p = 1$ ergebende Metrik wird als City-Block-Metrik oder auch Manhattan-Metrik bezeichnet. Sie berechnet sich als Summe der Abstände in den einzelnen Dimensionen. Für $p \to \infty$ ergibt sich die Maximum-Distanz. Sie nimmt den Wert der größten Differenz $|x_{ik} - x_{jk}|$ zwischen den k-ten Attributen der zwei Objekte an. Daher kommt der ebenfalls gebräuchliche Name Dominanzmetrik. Ausreißer im Sinne extremer Werte in einer Dimension spielen mit wachsendem p eine größere Rolle, bei der City-Block-Metrik fallen Ausreißer am wenigsten ins Gewicht. Abbildung 3.15 zeigt die drei Sonderfälle im Überblick. Die Linien geben jeweils alle Punkte im $\mathcal{R}^2$ an, die den Abstand 1 zum Ursprung in den jeweiligen Metriken haben.

Alle von der Minkowski-Metrik abgeleiteten Distanzen sind von Verschiebungen der Datenpunkte unabhängig. Der euklidische Abstand ist darüber hinaus von Drehungen und Spiegelungen unabhängig. Jedoch sind sie alle von der Skalierung in den Werten, wie sie etwa mit der Änderung der Größeneinheit verbunden sind, abhängig.

Wenn dies nicht erwünscht ist, können die Werte vorher normiert werden. Eine andere Möglichkeit besteht in der Verwendung eines von der Skalierung unabhängigen Maßes. Hier bietet sich zum Beispiel das Kosinusmaß an, welches den Kosinus des Winkels zwischen zwei Objekten, vom Koordinatenursprung aus gemessen, bestimmt:

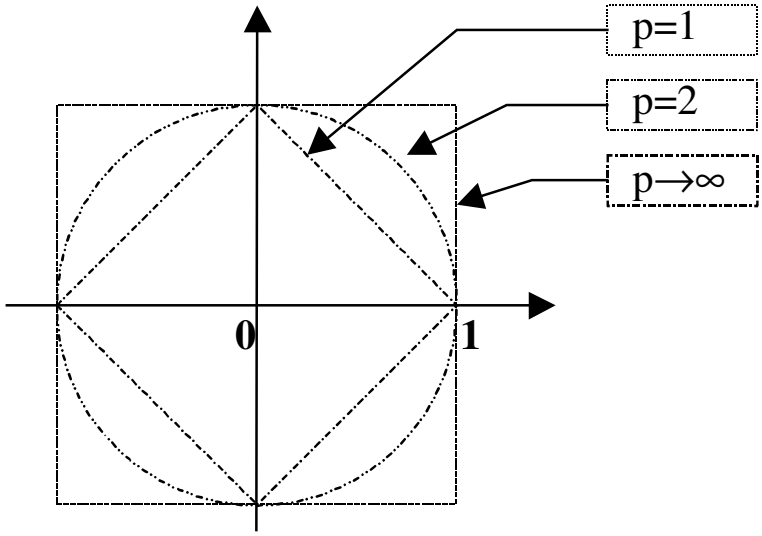

Abbildung 3.15: Minkowski Metrik

$$s_{ij} = \frac{\sum_{k=1}^{n} x_{ik}x_{jk}}{\sqrt{\sum_{k=1}^{n} x_{ik}^2 \sum_{k=1}^{n} x_k^2}} \tag{3.9}$$

Bei diesem Maß zählt nur die Richtung der Vektoren, so dass eine Streckung oder Stauchung eines Vektors keine Veränderung des Abstandsmaßes mit sich bringt. Dies kann für Attribute sinnvoll sein, deren Verhältnis zueinander entscheidender ist als ihre Absolutwerte.

Ein anderes Maß ist der Korrelationskoeffizient:

$$s_{ij} = \frac{\sum_{k=1}^{n} (x_{ik} - x_{i\cdot})(x_{jk} - x_{j\cdot})}{\sqrt{\sum_{k=1}^{n} (x_{ik}^2 - x_{i\cdot}) \sum_{k=1}^{n} (x_{jk}^2 - x_{j\cdot})}}, \; mit \; x_{i\cdot} = \frac{1}{n} \sum_{k=1}^{n} x_{ik} \tag{3.10}$$

Dieses gibt in der Statistik einen Anhaltspunkt über die Abhängigkeit von Variablen. In der Clusteranalyse ist neben der Zusammenfassung der Objekte auch die Zusammenfassung der Attribute möglich. Für diesen Fall ist die Eignung des Korrelationskoeffizienten offensichtlich. Wenn es aber für den Vergleich von Objekten verwendet werden soll, ist dieses Maß schwer interpretierbar. Bei verschiedenartigen Attributen ist zum Beispiel die Mittelwertbildung über die Attribute, wie sie zur Berechnung dieses Maßes geschieht, kaum zu rechtfertigen. Daher wurde dieses Maß zur Abstandsbestimmung von Objekten kritisiert und ist sicher mit Vorsicht zu verwenden [Bacher 1996]. Allerdings werden in [Gordon 1999] Studien zitiert, die

Anwendungsfälle aufzeigen, in denen mit dem Korrelationskoeffizienten gute Resultate erzielt wurden. Das Ergebnis fasst Gordon intuitiv so zusammen, dass dieses Maß bei Objekten Sinn macht, bei denen die durch die Attribute beschriebene Form wichtiger ist als die Größe. Auch weist er darauf hin, dass der Korrelationskoeffizient, wie das Kosinusmaß, den Winkel zwischen zwei Vektoren bestimmt. Allerdings verwendet es als Bezugspunkt nicht den Koordinatenursprung, sondern den Mittelwert der Daten. Daraus folgt, dass nach anfänglich erfolgter Normierung der einzelnen Attribute auf den Mittelwert 0 die beiden Maße äquivalent sind.

Um noch die Nichtnegativität zu erfüllen müssen das Kosinus- und das Korrelationsmaß noch transformiert werden. Beide liegen in einem Wertebereich von -1 bis 1; mit $s^*_{ij} = (1 + s_{ij})/2$ liegen diese zwischen 0 und 1. Beide Maße sind Ähnlichkeitsmaße und müssen für die Clusteranalyse in Abstandsmaße überführt werden. Übliche Transformationen für diesen Zweck sind $d = 1 - |s|$ oder $d = \frac{1}{|s|} - 1$.

3.11.1.2 Nicht-quantitative Attribute

Attribute, die nicht quantitativ sind, können ordinal, symbolisch oder binär sein. Die Werte symbolischer Attribute enthalten keine Information über eine Ähnlichkeit, wenn kein Kontext bekannt ist. Typische Beispiele sind zum Beispiel Familienstand oder Haarfarbe einer Person. Ordinale Attribute sind symbolische Attribute, auf denen jedoch eine Ordnung definiert ist. Binäre Attribute sind symbolische Attribute mit genau zwei Ausprägungen. Mit diesen Attributen kann nicht gerechnet werden wie mit numerischen. Der allgemeinste Weg ist, für Attribute mit endlichem Wertebereich eine Matrix (auch Abstandstabelle genannt) aufzustellen, in der für jede Kombination der Attributwerte ein Distanzkoeffizient angegeben ist. Beim Sonderfall binärer Attribute ist dies kein Problem, für große Wertebereiche ist dieser Weg jedoch nicht praktikabel. Ein alternatives, sehr einfaches Maß besteht darin, lediglich auf Übereinstimmung der Werte zu prüfen. Dies würde einer Matrix entsprechen, die in der Hauptdiagonalen den Wert 0 und sonst überall den gleichen Wert – etwa 1 – als Abstand enthält.

Die bei ordinalen Attributen in Form der Ordnung gegebene zusätzliche Information kann verwendet werden, um einen Rang zu bestimmen. Dieser ist als die Position des Attributwertes in einer sortierten Anordnung aller vorkommenden Attributwerte definiert. Der Rang kann dann zur Abstandsberechnung dienen. Die Distanz könnte sich etwa als Differenz der Ränge zweier Attributwerte berechnen. Bei Objekten mit mehreren ordinalen Attributen bietet sich die City-Block-Metrik auf den Rängen an.

Die Verwendung von Rängen kann auch für quantitative Attribute interessant sein. Zwar geht bei der vereinfachten Betrachtung numerischer als ordinale Attribute Information verloren, dafür ist ein auf Rängen definiertes Abstandsmaß unempfindlicher gegenüber Ausreißern in den Daten.

3.11.1.3 Gemischte Attribute

Häufig werden die in einer Clusteranalyse betrachteten Objekte anhand verschiedenartiger Attribute beschrieben. Die oben beschriebenen Verfahren gehen aber stets von einem Vektor gleichartiger Variablen aus. Die Verschiedenartigkeit kann sich in gemischten Attributtypen oder in der Kombination von Attributen mit unterschiedlicher Bedeutung zeigen. Es entsteht also das Problem, mit solchen Attributen umgehen zu können.

Gordon [Gordon 1999] schlägt 3 Lösungen vor, von denen er von einer - der getrennten Analyse für einzelne Typen mit anschließender Kombination der Ergebnisse - gleich abrät. Eine Lösung für unterschiedliche Attributtypen sieht er in der Überführung der Typen in einen gemeinsamen Typ. Wenn die Attributtypen nach Informationsgehalt geordnet werden, ergibt sich folgende Ordnung:

NUMERISCH → ORDINAL → NOMINAL → BINÄR

Dabei lassen sich die Attribute unter Informationsverlust in die jeweils weiter rechts stehenden Attribute umwandeln. Der gemeinsame Typ, in den alle Attribute überführt werden müssen, ist der am weitesten rechts stehende. Wegen des damit verbundenen Informationsverlusts ist dieser Ansatz nur dann praktikabel, wenn die Mehrzahl der Attribute ohnehin im informationsärmsten Typ gegeben sind.

Eine weitere, in der Praxis häufig eingesetzte Lösung ist die getrennte Festlegung eines Distanzmaßes für jede Variable oder für Gruppen von Variablen. Die totale Distanz berechnet sich dann als gewichtete Summe, bzw. Mittelwert der Einzelabstände. Es kann sich als sinnvoll erweisen, die einzelnen Differenzen auf ein festes Intervall, also etwa [0, 1] zu normieren, so dass die möglichen Beiträge der Einzeldifferenzen zum Gesamtabstand gleich sind. Mucha schlägt die Verwendung einer verallgemeinerten City-Block-Metrik vor [Mucha 1992]. Da sich diese als Summe der Einzeldifferenzen berechnet, kann sie auch auf ordinale und binäre Attribute ausgeweitet werden. Mit entsprechender Normierung ergibt sich folgende Formel:

$$d_{ij} = \frac{1}{n} \sum_{k=1}^{n} \frac{|x_{ik} - x_{jk}|}{u_k} \tag{3.11}$$

mit $u_k = max(x_k) - min(x_k)$ (Spannweite der Variable k)

3.11.1.4 Gewichtung

Ausgehend von der Schreibweise der zu clusternden Daten in einer Matrix mit den Objekten in den Zeilen und den Variablen in den Spalten kann zwischen Spalten- und Zeilengewichtung unterschieden werden.

Spaltengewichtung Die Spaltengewichtung, also die Gewichtung der Variablen, kann das Ergebnis einer Klassifikation stark verändern. Entsprechende Beachtung

hat dieses Problem auch in der Forschung gefunden. Einige Autoren lehnen eine Gewichtung gänzlich ab. Argumente sind, dass eine Festlegung von Gewichten schwierig zu rechtfertigen sei. Wenn der Experte die Gewichte solange verschiebt, bis ihm das Klassifikationsergebnis gefällt, liege der Verdacht der Unwissenschaftlichkeit nahe. Allerdings ist auch bei einer formalen Gleichgewichtung, also etwa alle $w_k = 1$ im gewichteten euklidischen Abstand, eine tatsächliche Gewichtung nicht ausgeschlossen. Schon die Wahl der verwendeten Attribute ist subjektiv und entspricht einer Gewichtung der nicht verwendeten mit $w_k = 0$. Wenn linear stark korrellierte Variablen verwendet werden, fallen diese stärker ins Gewicht als andere. Und nicht zuletzt entspricht die Wahl der Einheit einer Skalierung und damit einer Gewichtung der Variablen.

Es stellt sich also eher die Frage, wie über die explizite Gewichtung die Analyse objektiviert werden kann. Eine Möglichkeit der gezielten Gewichtung zur Objektivierung besteht zum Beispiel in der Normierung der Standardabweichung oder des Mittelwertes auf 1.

Neben der Objektivierung kann aber auch Vorwissen über tatsächlich für eine Unterscheidung wichtigere Attribute verwendet und in Form von Spaltengewichten eingebracht werden. Wenn dies geplant ist, ist es sinnvoll, zunächst die Attribute zu normieren und anschließend zu gewichten. Üblich ist hierfür etwa eine Gewichtung mit der Standardabweichung. Die Quotientenbildung mit der Standardabweichung löst Probleme, die sich aus unterschiedlichen Skalierungen ergeben. Die so normierten Attribute können anschließend entsprechend ihrer vom Experten vermuteten Wichtigkeit mit individuellen Gewichten versehen werden.

Falls Gewichte verwendet werden, sollte man sich der Probleme bewusst sein, die daraus erwachsen können und entsprechend sorgfältig sollte die Gewichtung vorgenommen werden.

Zeilengewichtung Die Gewichte der Zeilen werden als Massen bezeichnet. Häufig handelt es sich um natürliche Zahlen, die die Anzahl der Objekte angeben, die durch eine Zeile repräsentiert werden. Das eine Zeile mehrere Objekte darstellt, ist das Ergebnis der Aggregation mehrerer Zeilen. Es kann jedoch auch von Anfang an eine Zeile eine Zusammenfassung mehrerer Objekte sein und mit einer entsprechenden Masse versehen werden. Die Massen werden in einigen Clusteralgorithmen verwendet, um die Varianz oder den Schwerpunkt eines Clusters zu bestimmen. Auf diese Weise beeinflussen die Zeilengewichte das Clusteringergebnis. Das Ward'sche Minimalvarianzverfahren tendiert zum Beispiel dazu, massenmäßig homogene Cluster zu generieren. Werden keine Massen angegeben, bzw. diese auf 1 gesetzt, werden bevorzugt Cluster gebildet, die aus ähnlich vielen Zeilen bestehen. Wenn die Datenzeilen mehr als jeweils ein und insbesondere unterschiedlich viele Objekte repräsentieren, ist im letzteren Fall die Homogenität in Bezug auf die Anzahl Objekte nicht gewährleistet. In solchen Fällen ist zu überlegen, welches Clusterverhalten die in den Daten enthaltenen Strukturen besser erkennt, bzw. welche Art von Clustern als Lösung gewünscht werden.

Auch wenn dies der häufigste Fall ist, müssen die Massen nicht für die Darstellung der Anzahl der Objekte in einer Zeile verwendet werden. Sie werden ebenfalls eingesetzt, um Zeilen mit fehlenden oder weniger glaubwürdigen Werten niedriger zu gewichten. Ziel ist es, dass derartige Zeilen das Clusterergebnis weniger beeinflussen als Zeilen mit gesicherten Werten. In diesen Fällen sind für die Massen auch reelle Werte möglich. Wegen der inhärenten Subjektivität solcher Gewichtungen ist diese Einsatzmöglichkeit jedoch nicht unproblematisch.

3.11.2 Abstandsberechnung bei Klassenvereinigung

Es gibt verschiedene Möglichkeiten die Distanzbestimmung zwischen den Clustern durchzuführen. Die Berechnung der neuen Distanzen kann bei allen Varianten aus den bisherigen Distanzwerten erfolgen, so dass nach initialer Berechnung der Distanzmatrix nicht mehr auf die Attributwerte zugegriffen werden muss. In die Berechnungsvorschrift zur Abstandsberechnung eines Clusters k von zwei zu vereinigenden Clustern i und j können die Abstände $d_{k,i}$, $d_{k,j}$ und $d_{i,j}$, sowie die Massen u_k, u_i und u_j der drei Cluster einfliessen. Die Cluster i und j bilden die neue Klasse $k*$.

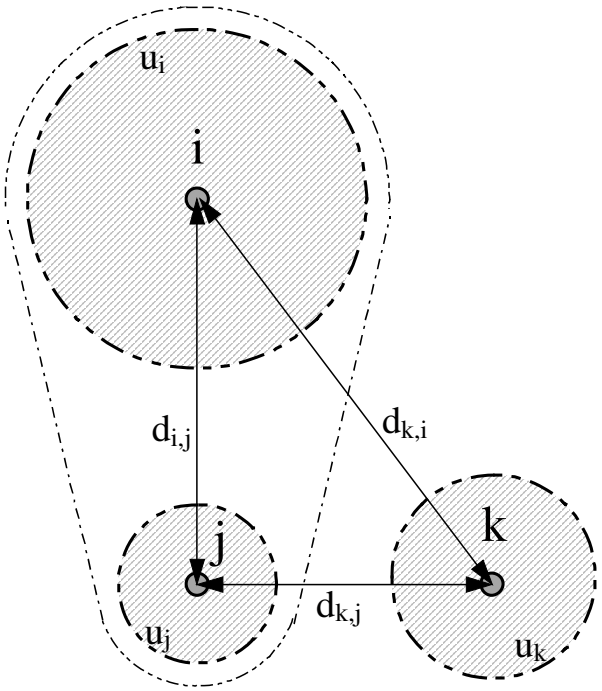

Abbildung 3.16: Einflußfaktoren bei Klassenvereinigung

Einige der bekanntesten Methoden sind

- Single Linkage,

- Complete Linkage und
- (Simple) Average Linkage,
- Zentroid-Methode,
- Minimalvarianzverfahren nach Ward,
- sowie die flexible Strategie nach Lance und Williams.

Single und Complete Linkage stellen zwei Extreme dar, während die anderen Methoden in ihrem Verhalten zwischen diesen beiden liegen. Generell ist es schwierig, allgemeingültige Aussagen über das Verhalten der verschiedenen Agglomerationsstrategien zu machen, da dieses stark mit der Struktur der Daten variiert. Ein paar Anhaltspunkte werden im Folgenden beschrieben.

- Berücksichtigung von Massen:
  Ein wesentlicher Unterschied zwischen den einzelnen Verfahren liegt darin, ob und in welcher Weise die Massen der Cluster einbezogen werden. Einige Verfahren, wie Single, Complete oder Simple Average Linkage, berücksichtigen keine Massen. Bei den Verfahren, die die Massen mit in die Distanzbestimmung einbeziehen, wird die neue Klassendistanz stärker vom schwereren der vereinigten Cluster bestimmt, was durchaus plausibel ist und homogenere Cluster fördert. Das Minimalvarianzverfahren nach Ward verwendet die Massen zusätzlich bei der Auswahl der zu vereinigenden Cluster aus der Distanzmatrix.

- Kettenbildung:
  Single Linkage verwendet als Klassenabstand immer die kürzeste Entfernung zwischen den Objekten zweier Klassen. Bei der Vereinigung der Cluster i und j ist die neue Distanz $d_{\{i,j\},k}$ zu einem dritten Cluster k gerade min $(d_{i,k}, d_{j,k})$. Dadurch neigt das Clustern zur Kettenbildung, sowie zum Isolieren von Ausreißern. Dies kann zum Beispiel in der Muster- oder Zeichenerkennung so gewollt sein, zumal auch unterschiedlich große und sehr heterogene Cluster gut erkannt werden. Für die Entdeckung sphärischer Cluster, wie sie bei der Modellvorstellung überlagerter multivariat normalverteilter Populationen erwartet werden können, ist Single Linkage jedoch nicht geeignet. Insbesondere sich überlappende Cluster oder solche, die durch *Brücken* aus einzelnen Punkten verbunden sind, werden nicht erkannt.

- Kompakte Cluster:
  Im Gegensatz dazu steht das Complete Linkage, das den Diameter, also die Distanz der am weitesten entfernten Punkte zweier Klassen als Klassendistanz übernimmt und dadurch sehr kompakte Cluster bildet. Einzelpunkte bleiben jedoch auch lange isoliert. Im Gegensatz zu Single Linkage gilt hier $d_{\{i,j\},k} = max(d_{i,k}, d_{j,k})$. Average Linkage nimmt den Mittelwert aller paarweisen Distanzen der Objekte der beiden Klassen, das entspricht dem mit

den Massen gewichteten Mittelwert der Abstände $d_{k,i}$ und $d_{k,j}$. Ein hierarchisches Clustern auf dieser Basis tendiert zur Bildung von sphärischen Clustern. Schwierigkeiten bestehen bei der Erkennung anders geformter Cluster. Beim Simple Average Linkage werden die Clustermassen wie erwähnt ignoriert. Die Zentroid-Methode bestimmt die Schwerpunkte aller Objekte der Cluster und verwendet dann die Distanz dieser Repräsentanten. Für die Zentroidbestimmung wird von einem euklidischen Vektorraum ausgegangen. Beim Ward'schen Verfahren wird ebenfalls von einem euklidischen Vektorraum ausgegangen. Aus den Klassendistanzen und -massen werden die mit einer Fusion verbundenen Varianzzuwächse hergeleitet. Es werden dann aus der Distanzmatrix jeweils die beiden Klassen bestimmt und vereinigt, deren Fusion mit dem geringsten Varianzzuwachs verbunden ist. Diese Vorgehensweise erzeugt recht homogene Cluster. Insbesondere werden bevorzugt kleine Cluster, also solche mit niedrigen Massen, mit großen Clustern vereinigt, wodurch auch einzeln liegende Punkte einbezogen werden. Wegen der Homogenität der Cluster wird dieses Verfahren sehr häufig eingesetzt. Es erzeugt ähnliche Cluster wie die partitionierenden Verfahren. Die gewichtete flexible Strategie nach Lance und Williams berechnet die Klassendistanzen nach folgender Formel:

$$d_{k',k} = d_{\{i,j\},k} = \frac{1-b}{u_i + u_j}\left(u_i d_{i,k} + u_j d_{j,k}\right) + b d_{i,j}, b \in (-1,1) \tag{3.12}$$

Sie bietet die Möglichkeit, über den Parameter b einen weiten Bereich von Verhalten abzudecken. Insbesondere kann durch die geeignete Wahl das Verhalten der anderen angegebenen Methoden angenähert werden (vergleiche [Gordon 1999, Seite 79]). Ein Standardwert für das Entdecken homogener Cluster ist b = -0,15. Die Möglichkeit, den Parameter leicht zu ändern, ist im Zusammenhang mit der Stabilitätsuntersuchung nützlich. Bei der ungewichteten flexiblen Strategie werden die Gewichte u_i und u_j gleich 1 gesetzt.

3.11.3 Einbindung von Hintergrundwissen

Ein verbreitetes Problem des maschinellen Lernens ist die Integration des bereits vorhandenen (Experten-)Wissens. Die durch das Einbringen von Hintergrundwissen mögliche Einschränkung des Suchraums kann zum einen die Effizienz eines Algorithmus erhöhen, zum anderen kann die Beschränkung der möglichen Ausgaben die Korrektheit sichern und die Interpretierbarkeit verbessern. Die subjektive Einschränkung ist allerdings mit der Gefahr verbunden, unerwartete, aber interessante Ergebnisse zu verwerfen.

Bei der Clusteranalyse kann Vorwissen darin bestehen, dass gewisse Klassenzugehörigkeiten schon bekannt sind oder zumindest vermutet werden. Eine andere Form von Vorwissen ist, dass Informationen über die Daten bekannt sind, die nicht in den Attributen enthalten sind und nicht problemlos integriert werden können.

Ein Beispiel[13] ist die Aufgabe der automatischen Einteilung eines Gebietes in Bo-

[13]Entnommen aus [Gordon 1999] Beispiel 2

dengruppen. Dazu wurde das Gebiet in Planquadrate eingeteilt, für die jeweils Bodenproben entnommen wurden. Jede Bodenprobe wird durch verschiedene Attribute beschrieben. Zu den aufgenommenen numerischen Attributen gehören die einzelnen Mineralstoffgehalte, während eines der binären Attribute zum Beispiel angibt, ob die Bodenprobe Regenwürmer enthält oder nicht. Bodenproben ähnlicher Zusammensetzung dürfen jedoch nur dann zusammengefasst werden, wenn die zugehörigen Planquadrate benachbart sind. Das Problem ist es, die Nachbarschaftsbeziehungen der Bodenpunkte mit zu verwenden.

[Gordon 1999] schlägt für die Lösung dieser Aufgabe einen graphentheoretischen Ansatz vor. Ein anderer Ansatz zur Integration von Hintergrundwissen arbeitet mit speziellen Distanzfunktionen (siehe Abschnitt 3.11.3.1).

Eine einfache, eher intuitive Möglichkeit besteht im gezielten Anpassen der Gewichte. Die Gewichtung der Variablen spielt eine entscheidende Rolle für die Bildung der Cluster. Wenn also bekannt ist, dass einige Variablen einen größeren Einfluß auf die Trennung der Cluster haben als andere, so kann dies in der Clusteranalyse durch stärkere Gewichtung berücksichtigt werden (siehe auch 3.11.1.4).

3.11.3.1 Spezielle Distanzfunktion

[Mucha 1992] schlägt einen anderen Ansatz vor. Er definiert eine Distanzfunktion, die auch die Zusatzinformationen mit einbezieht. Dabei konstruiert er den speziellen Abstand d als Summe eines der bekannten Abstandsmaße d und einer Funktion f, die den euklidischen Abstand $d*$ der Zusatzinformationen in eine zusätzliche Distanz abbildet: $d = d + f(d*)$. Aus der so erhaltenen Distanz wird die Distanzmatrix bestimmt, auf der dann der Clusteralgorithmus arbeitet.

Das Beispiel mit den Bodenproben könnte also so gelöst werden, dass die Koordinaten der Entnahmestellen als zusätzliche Daten angegeben werden. Anhand des euklidischen Abstandes dieser Koordinaten kann festgestellt werden, ob die dazugehörigen Bodenproben an benachbarten Punkten entnommen wurden. Bei größeren Abständen nimmt die Funktion f sehr große Werte an und verhindert dadurch die Zusammenfassung der nicht-benachbarten Objekte.

Vorteil dieser Vorgehensweise ist, dass nach der Bestimmung der Distanzmatrix die Standardalgorithmen verwendet werden können und daher vorhandene Programme weitestgehend unverändert eingesetzt werden können. Nachteil ist, dass durch die künstliche Distanzerhöhung die tatsächlichen Distanzen verfälscht werden und darunter die Klassifikationsergebnisse potentiell leiden. Daher sollte bei Verwendung dieses Ansatzes die Aufgabe des Hintergrundwissen nicht wie hier im ausschließen unerlaubter Vereinigungen, sondern lediglich in einer moderaten Beeinflussung des Agglomerationsprozesses bestehen.

Eine andere Variante besteht darin, in einem hierarchischen Standardverfahren nur solche Agglomerationen zu erlauben, die nicht den Nebenbedingungen widersprechen. Es wird also nur unter den Elementen der Distanzmatrix, deren dazugehörige

Cluster vereinigt werden dürfen, das kleinste gesucht und die Agglomeration mit diesem durchgeführt.

3.12 Auswahl der Cluster–Verfahren

In der Anwendungsdomäne des Produktbewährungsprozesses ist nicht die Beschreibung der Klassen von Interesse, sondern lediglich die Gruppierung der einzelnen Schadenscode. Die Schadenscode sind eine feste Anzahl von Individuen, für die zunächst einmalig eine Einteilung stattfinden soll. Aus zwei Gründen versprechen die Verfahren des Conceptual Clusterings für diesen Anwendungszweck keine Vorteile gegenüber den klassischen Clusteranalyseverfahren. Wie sich im Laufe der Datenanalyse gezeigt hat, sind die für die Ähnlichkeitsbeziehungen verwendeten Attribute für sich genommen nicht für eine interpretierbare Beschreibung der Objekte geeignet. Eine Aufstellung von Konzepten anhand ihrer Werte liefert den Experten kein neues, verständliches Wissen. Daneben ist die Festlegung auf beschreibende Hyperquader eine einschränkende Annahme über die Lage der Cluster, die bei Verwendung der klassischen Verfahren nicht notwendig ist.

Im Einzelnen lassen sich folgende Vor- und Nachteile für die hierarchischen oder partitionierenden Methoden identifizieren:

- Wenn die Bildung einer Hierarchie das Ziel der Clusteranalyse ist oder wenn die Reihenfolge der Klassenagglomerationen von Bedeutung ist, müssen hierarchische Methoden zum Einsatz kommen.

- Bei den hierarchischen Methoden müssen keine Annahmen über die Klassenanzahl gemacht werden. Auch wenn nur eine Partition der Daten gewonnen werden soll, kann die Hierarchie zur Voranalyse verwendet werden, um daraus eine Klassenanzahl abzuleiten.

- Der Nachteil der hierarchischen gegenüber den partitionierenden Verfahren besteht darin, dass in jedem Agglomerationsschritt stets die im Sinne einer Kostenfunktion billigsten Cluster (ähnlichsten) vereinigt werden. Es handelt sich hier also um einen Greedy-Algorithmus. Daraus folgt, dass man nicht sicher sein kann das globale Optimum zu finden.

Wie oben schon beschrieben bringen Conceptual Clusterverfahren keine Vorteile, so dass die Wahl zwischen einem partitionierenden oder hierarchischen Verfahren getroffen werden muss. Gegen ein Partitionsverfahren spricht, dass in den Daten eigentlich keine Klassenstruktur existiert. Ziel ist es vielmehr, möglichst kleine Klassen mit ca. 5-6 Teilen zu haben, damit diese Gruppen für die Anwender interpretierbar bleiben und auch eine Aussagekraft haben. Da die Ausfallraten der Teile im Cluster kumuliert werden sollen, machen größere Cluster keinen Sinn, da sich Veränderungen in den Ausfallraten wieder ausgleichen können.

3.12.1 Auswahl der Clusterattribute

Um sinnvolle Cluster bilden zu können, wurde versucht Gruppen von Teilen zu bilden, die funktional von einander abhängen oder benachbarte Aggregate sind (lokale Abhängigkeiten). Hierzu wurden vier potentielle Kriterien zum Clustern der Schadensteile mit den Experten ausgewählt:

ähnliche Kosten:

Ähnliche Kosten können ein Indikator für eine Fehlvercodung der Schadensursache sein. Diesem Kriterium liegt die Annahme zugrunde, dass die Werkstätten mit einer großen Wahrscheinlichkeit ihre wahren Kosten in Rechnung stellen.

Es wurden verschiedene Kombinationen der Kostenattribute oder abgeleiteter Werte betrachtet. Als möglicherweise interessant und für einen Schaden spezifisch werden dabei die entstandenen Materialkosten, sowie das Verhältnis der Lohn- zu den Gesamtkosten eingeschätzt. Wenn diese für die einzelnen Beanstandungen gegeneinander aufgetragen werden, liegen die Schadensgruppen, wenn auch teilweise überlagert, in relativ eng umgrenzten Teilgebieten.

Ausgehend von dieser Beobachtung läßt sich die Vermutung, dass die Kosten der Beanstandungen eine Aussagekraft für eine sinnvolle Gruppierung besitzen, grafisch untermauern (siehe Abbildung 3.17).

In den folgenden Abschnitten werden exemplarisch zwei Funktionsgruppen betrachtet, um die Auswahl der Attribute zu veranschaulichen. Dies sind die Betriebsbremsen und die Einspritzpumpe. Es werden jeweils die Gruppierungen für die Teile dieser Gruppen interpretiert und für die unterschiedlichen Kriterien verglichen. An der Einspritzpumpe gibt es für Beanstandungen am Motorlauf getrennte Schadensschlüssel für „allgemeine Beanstandungen“, „Beanstandungen am kalten Motor“, bzw. „Beanstandungen am warmen Motor“. Diese könnten intuitiv zusammengefasst werden, da sie einen typischen Fall von mehrdeutiger Codierung darstellen. Weiterhin werden bei den Beanstandungen die Einspritzleitungen für den ersten bis fünften Zylinder unterschieden. Auch bei diesen kann ein sehr ähnliches Verhalten erwartet werden, so dass eine Zusammenfassung wünschenswert ist. In der Abbildung 3.17 sind die entsprechenden Gruppen dargestellt. Für jede Gruppe sind die Materialkosten, bzw. das Lohn- zu Gesamtverhältnis der einzelnen Beanstandungen gemittelt und dann aufgetragen worden.

Als Distanzmaß haben wir hierfür das euklidische Maß gewählt. Diese Kriterien reichen nicht aus, um bei einer Clusteranalyse gerade diese Gruppen zu finden. Die Punkte deren Gruppierung sinnvoll erscheint liegen aber eng beieinander. Ein ergänzender Einsatz dieses Kriteriums neben den anderen erscheint daher sinnvoll.

Ausfallverhalten über die Lebenszeit:

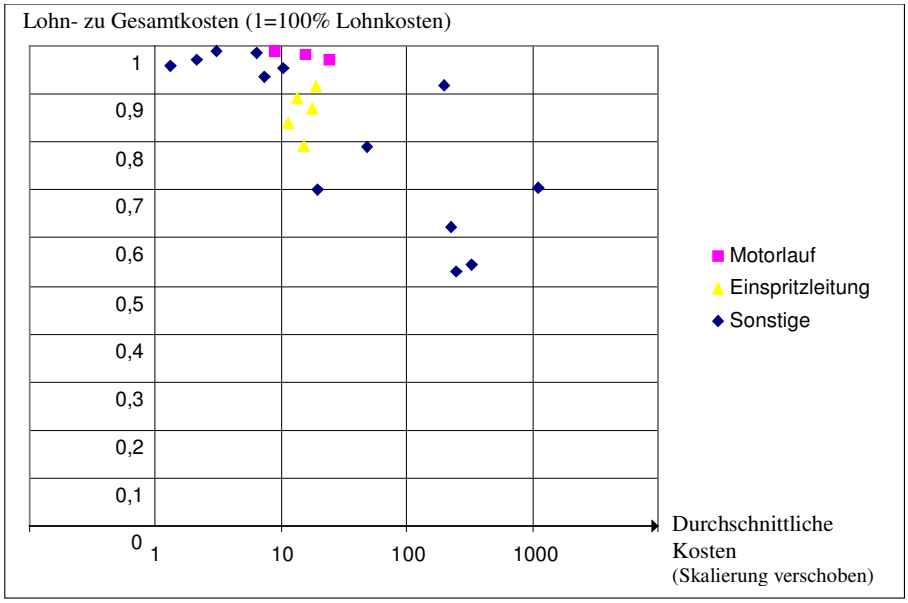

Abbildung 3.17: Kosten als Unterscheidungskriterium

Beobachtungen des Schadensverhaltens der verschiedenen Teile im ersten Jahr zeigten, dass man das Ausfallverhalten einfach beschreiben kann. Das heißt, man kann beobachten, dass bestimmte Teile eher zu Beginn ihrer Lebenszeit[14] ausfielen, während andere erst gegen Ende ihrer Lebenszeit die höheren Ausfallraten zu verzeichnen hatten. Um dieses Verhalten in einem Distanzmaß zu repräsentieren, wurde das Kosinusmaß gewählt. Es ist unabhängig von der Skalierung und vergleicht mehr die Form, die durch die Datenpunkte beschrieben wird. Als Laufzeitverhalten wurde die Einsatzdauer in Tagen seit der Erstzulassung verwendet. In der Abbildung 3.18 ist das normierte Ausfallverhalten zum Ende der einjährigen Garantiezeit für die Betriebsbremse dargestellt.

Von oben konvexe Kurven zeigen dabei solche Teile, deren Ausfallwahrscheinlichkeit anfangs hoch ist und dann abnimmt. Von oben konkave Kurven deuten auf ein Spätausfallverhalten der entsprechenden Teile hin.

Auffällig ist die erstaunlich gute paarweise Überlagerung der Kurven. Interessanterweise weisen die zu den sich überlagernden Kurven gehörenden Teile nachvollziehbare Ähnlichkeiten auf. So haben die vorderen und hinteren Bremsscheiben ein recht gut übereinstimmendes Verhalten. Das gleiche gilt für die vorderen und hinteren Bremsklötze. Die befragten Experten interpretieren dies als funktionale Ähnlichkeit, die zu einem ähnlichen Abnutzungsverhalten führt. Anders verhält es sich mit Ähnlichkeit zwischen der Unterdruckpumpe und deren Antrieb. Hier kann es sich auch um einen Fall einer Fehlcodierung durch eine mehrdeutige Verschlüsselungsmöglichkeit handeln. Die Annahme, die dieser Ähnlichkeitsbeziehung zugrunde liegt, ist, dass eine bestimmte, „wah-

[14] sogenannte Frühausfälle

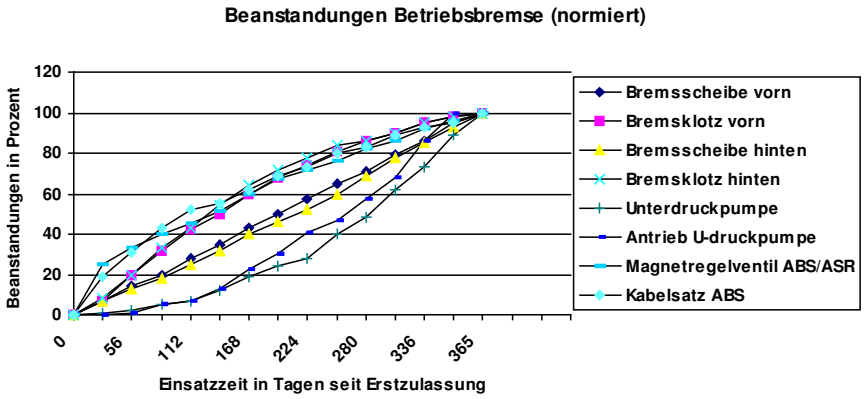

Abbildung 3.18: Ausfälle über die Laufzeit (normiert)

re“ Schadensursache eine spezifische Form hat, also etwa sehr früh oder spät, oder gerade gleich verteilt auftritt. Die daraus resultierenden Schäden und damit funktional abhängigen Teile, bzw. die codierten Beanstandungen legen dann das gleiche Verhalten an den Tag. Von diesem Kriterium kann also sowohl eine Aussage über funktionale Abhängigkeiten, als auch über Fehlcodierungen erwartet werden. Bei anderen Teilen, etwa der Einspritzpumpe, kann ein entsprechendes Verhalten beobachtet werden. Allerdings gibt es genauso Gegenbeispiele, in denen die Ähnlichkeit der Kurven einer Interpretation entbehrt. Dies liegt sicher daran, dass zwar aus funktionalen Abhängigkeiten ähnliche Kurven folgen, die Umkehrung aber nicht zwingend ist. Die Ähnlichkeit der Kurven ist also nur eine notwendige Bedingung für funktionale Abhängigkeit. Darüber hinaus können hier keine „harten“ Regeln erwartet werden, da alle Folgerungen immer nur mit einer gewissen Wahrscheinlichkeit gelten.

Zeitgleiche Anstiege von Ausfallquoten:

Die Beanstandungsraten wurden über die Produktionszeit aufgetragen. Wenn die Beanstandungsrate zweier Teile oder Gruppen sich simultan verändert, so ist dies vielleicht ein Indikator für eine funktionale oder lokale Abhängigkeit. Hintergrund für die Auswahl dieses Kriteriums ist weiterhin, dass die Werkstätten beim Aufkommen eines neuen Schadensschwerpunkt unterschiedliche Teile als das schadensverursachende Teil vercoden. So kommt es erst zu einem relativ leichten Anstieg mehrerer Teile, bis das wahre schadensverursachende Teil durch die Befundung bestimmt ist. Ab diesem Zeitpunkt steigt dann nur noch die Beanstandungsquote für dieses Teil, da die Werkstätten ab diesem Zeitpunkt auf dieses Teil vercoden. Ab diesem Zeitpunkt ist die Kostenübernahme für die Werkstätten geregelt.

Es bietet sich an, als Ähnlichkeitsmaß die Korrelation zu verwenden. Da hier die Attribute alle die gleichen Werte, nämlich die Beanstandungsquoten repräsentieren, ist die Korrelation problemlos interpretierbar. Im Gegensatz zum Ausfallverhalten der Teile ist hier nicht die Form des Anstiegs von 0 auf

100%, sondern eher das tendenzielle, gemeinsame Schwanken der Quoten um den Mittelwert das ausschlaggebende Kriterium. Dies wird im Gegensatz zum Kosinus-Maß von der Korrelation berücksichtigt. Da eher ein Formvergleich als ein Lagevergleich im Raum durchgeführt wird, bieten sich andere Maße nicht an.

Schadensschlüsselbuch als Hintergrundwissen:

Als weiteres Kriterium haben wir die Struktur des Schadensschlüsselbuches, da es das Expertenwissen über die lokalen und funktionalen Abhängigkeiten repräsentiert. Allerdings ist die verwendete Hierarchie nicht homogen und die Größe der Gruppen ist für unseren Ansatz nicht geeignet, so dass sie nur als zusätzliche Information genutzt werden kann. Wir haben ein einfaches Distanzmaß definiert. Die Distanz ist gleich 0 für Teile aus der gleichen Konstruktions- und Funktionsgruppe, ist nur die Konstruktions- oder Funktionsgruppe unterschiedlich, so ist die Distanz größer 0. Mit einem solchen Maß kann auf einfache Weise die Struktur des Schadensschlüsselbuches berücksichtigt werden.

3.12.2 Auswahl der Cluster–Strategie und Gewichtung

Die Auswahl der Cluster–Strategie und die Gewichtung der einzelnen Kriterien wurde experimentell untersucht [Klose 1997]. Die drei Kriterien für die Ähnlichkeit und Abhängigkeit lieferten in den Voruntersuchungen, auch von den Experten als brauchbar eingestufte Einteilungen. Mit allen drei Kriterien wurden gleiche oder ähnliche Einteilungen gefunden, so dass es gerechtfertigt erscheint, ein gemeinsames Maß aus diesen drei zusammenzusetzen. Durch das Zusammensetzen soll die Sicherheit gegenüber den Einzelkriterien gesteigert werden.

Die Hierarchie des Schadensschlüsselbuches wurde als zusätzliche Komponente zum Distanzmaß addiert. Die Skalierung wurde so gewählt, dass diese zusätzliche Distanz nur einen moderaten, lenkenden Einfluss auf die Clusterbildung haben kann. Explizit wurde zu der Summe der jeweils auf Standardabweichung 1 normierten drei Distanzen 0,25 für unterschiedliche Funktions- und 1,0 für unterschiedliche Konstruktionsgruppen addiert. Als Agglomerationsverfahren wurde die gewichtete flexible Strategie verwendet. Das Ziel dieser Clustervariante ist es zu ermöglichen, dass Teile bei sehr ähnlichem Verhalten aus verschiedenen Funktions- oder Konstruktionsgruppen zusammengefasst werden können, bei mäßiger Ähnlichkeit jedoch die Einhaltung der bestehenden Hierarchie präferiert wird.

Für den praktischen Einsatz zeigte sich die gewichtete flexible Strategie nach Lance und Williams mit einem Parameterwert von $-0,15$ als Agglomerationstrategie einsetzbar.

3.13 Verfahren und Auswahl zur Prognose

Zur Erkennung der Schadensmuster sind verschiedene Verfahren aus dem Bereich des maschinellen Lernens verfügbar. Die erste Frage die sich aber stellt, ist die Frage nach der Anzahl der Modelle. Folgende Ansätze wären denkbar:

1. ein Modell pro Schadensteilgruppe und Baureihe
2. ein Modell pro Schadensteil oder Gruppe
3. ein Modell pro Baureihe
4. ein Modell für alle Baureihen und Schadensteile

Die Modelle werden gespeichert und bei einer Prognose für eben diese Schadensteilgruppe/Baureihe verwendet. Bei den ersten beiden Varianten werden viele Modelle benötigt. Das bedeutet einen großen Aufwand, sowohl für den Aufbau als auch für die Speicherung der vielen getrennten Prognosen. In den vorherigen Abschnitten wurde schon öfters die Schadenshäufigkeit angesprochen. Einzelne Schadensteile oder -gruppen können aufgrund der Datenbasis nicht ausreichend trainiert werden, um daraus ein zum Generalisieren fähiges Modell zu erhalten. Bei der vierten Variante ist zu befürchten, dass das Ergebnis ein zu generelles Modell zur Prognose ist. Diese Einschätzung wurde von den Produktbetreuern geteilt. Ein Modell das für LKW und PKW ausreicht war nicht vorstellbar. Aus diesem Grund wurde die Variante drei im Projekt umgesetzt.

3.13.1 Verfahren zur Prognose

Im Rahmen des Projektes wurden folgende Klassen von Verfahren bzw. repräsentante Vertreter dieser Klassen zur Prognose der Ausfalltendenz betrachtet:

- Regression
- Entscheidungsbaumlerner
- Neuronale Netze

Innerhalb des Projektes wurde das Tool Clementine™ um verschiedene Komponenten erweitert, während die anderen Verfahren alle von dem Tool zur Verfügung gestellt werden.

- Clusterknoten, mit den benötigten Verfahren[15]
- Sourceknoten, zum Daten einlesen von Host–Selektionen aus der Datenquelle

[15]Realisiert von Udo Grimmer, Mitarbeiter bei der Daimler Chrysler AG

3.13.1.1 Auswahl der Prognose–Verfahren

In unserer Machbarkeitsstudie haben wir folgende Verfahren getestet:

- lineare Regression
- Regellerner C4.5, als Repräsentant für TDIDT Lerner.
- Neuronale Netze (Multi Percepton)

Bevor wir auf die Ergebnisse der experimentellen Untersuchung eingehen, sollen an dieser Stelle die Eingabevariablen und Zielvariablen vorgestellt werden.

3.13.1.2 Eingabeattribute

Für die Prognose ist eine Festlegung der erklärenden Variablen und der Zielvariablen zu treffen. Für die erklärenden Variablen wird hier eine Beschränkung auf die Beanstandungsquoten vorgenommen. Es werden die relativen Beanstandungsquoten verwendet, also die Beanstandungen an dem Teil in Relation zur Gesamtzahl der Beanstandungen.

Es bleibt die Frage zu klären, welche Zeiträume der Beanstandungsquoten zum Lernen verwendet werden sollen. Abbildung 3.19 zeigt die zweidimensionale Betrachtung des Aufkommens der Beanstandungsquoten. Diese Art der Darstellung verdeutlicht die zeitlichen Strukturen des Datenmaterials. Jeder Punkt in der Abbildung steht für eine Beanstandungsquote eines bestimmten Produktionsmonats und eines bestimmten Fahrzeugalters. Jeden Monat kann für jede Spalte ein neuer Punkt in Y-Richtung hinzugefügt werden, da die Fahrzeuge um einen Monat gealtert sind. Zusätzlich wurden neue Fahrzeuge produziert, und es kann ein neuer Punkt auf der X-Achse eingezeichnet werden. Die neuen Punkte liegen auf der gestrichelten, schrägen Linie. Links unterhalb dieser Linie liegen die bereits bekannten Daten. Die Linie wandert jeden Monat um einen Schritt nach rechts.

Da die Daten hier im Nachhinein betrachtet werden, sind auch alle grau angedeuteten Punkte rechts oberhalb der Linie bereits bekannt und können für die Herleitung der Zielvariablen verwendet werden, wie dies im nächsten Kapitel erläutert wird. Den Lernverfahren dürfen sie selbstverständlich nicht als Eingangsvariablen präsentiert werden.

Die Eingangsvariablen werden aus einem trapezförmigen Ausschnitt der Daten verwendet, wie es im Bild durch die dunkelgraue Unterlegung angedeutet ist. Die Größe des Trapezes kann frei gewählt werden.

Zu klein darf der Zeitraum aber auch nicht sein, denn es sollen ja immerhin Trends gefunden werden können. Bei der Wahl eines Zeitraumes von 3 Monaten steht die gleiche Datenmenge zur Verfügung, die vom 3-Monats-Trend des Frühwarnsystems verwendet wird. Dieses wurde übernommen, damit eine Vergleichbarkeit zu den herkömmlichen Methoden gegeben ist. Um dem Lernverfahren mehr Daten über das

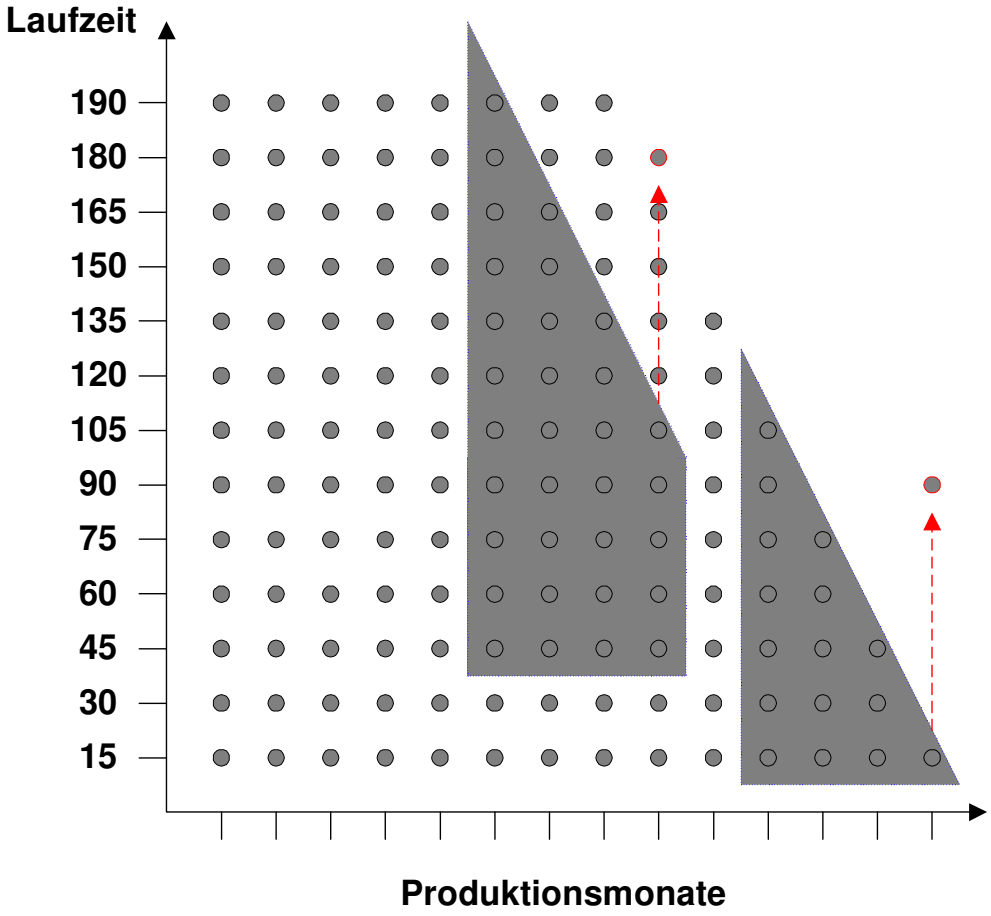

Abbildung 3.19: Zweidimensionale Betrachtung der Beanstandungsquoten

Vergangenheitsverhalten der Teile zu geben, werden zwei weitere Monate hinzugenommen.

In der Richtung des Fahrzeugalters wird die Anzahl der Zeitpunkte durch die Höhe des Trapezes bestimmt. Wenn vom jüngsten Produktionsmonat zum Beispiel zwei Monate Laufzeit verwendet werden, sind vom fünftletzten die Beanstandungsquoten von maximal 7 Monaten bereits bekannt. Da die Beanstandungen über die Lebenszeit kumuliert betrachtet werden, kann die Y-Richtung auch feiner als monatsweise aufgespalten werden. Durch die Aufsummierung verändern sich die Monatswerte nicht, und die zusätzlichen, beispielsweise 14-tägigen Werte stellen Zwischenwerte dar. Speziell für die späteren Zeitpunkte sind die Quoten, die nur zwei Wochen auseinander liegen, allerdings hochkorrelliert.

3.13.2 Zielvariablen

Nach der Festlegung möglicher Eingangsvariablen für die Prognose muss nun eine Zielvariable definiert werden. Aus der Anwendungsdomäne ist vorgegeben, dass möglichst früh auf einen Zeitpunkt geschlossen werden soll, zu dem die Trends be-

reits erkennbar und stabil sind. Als Zeitraum, nach welchem die Trends als relativ stabil angesehen werden, wurden 5 Monate gewählt. Bei längeren Zeiträumen treten eventuell Spätausfallschäden auf, die zu sehr frühen Zeitpunkten schlicht noch nicht in den Daten enthalten sind und die dieses System als Hilfe zur expliziten Früherkennung nicht finden können muss.

Auf den 5-Monatswerten werden verschiedene Werte berechnet, die prognostiziert werden sollen. Bei diesen Werten ist zwischen den numerischen und den symbolischen zu unterscheiden. Die numerischen Werte sollen dabei mit linearer Regression und neuronalen Netzen prognostiziert werden, während für die symbolischen Werte neben dem neuronalen Netz der Regellerner C4.5 eingesetzt wird. Folgende Werte werden als Zielvariable verwendet:

- Die Beanstandungsquote nach 5 Monaten
- Der Trend über 3 Monate nach der Berechnung des konventionellen Frühwarnsystems (unter Verwendung der 5-Monats-Werte):

$$Z_3(t) = \frac{\frac{1}{2}(R_t + R_{t-1}) - R_{t-2}}{R_{t-2}} * 100 \tag{3.13}$$

  wobei R_t die Beanstandungsquote zum Monat t ist.

- Nach Abstimmung mit den Experten wird der Trend durch die Angabe von Grenzwerten in drei Intervalle mit diskreten Klassen fallend, konstant oder steigend unterteilt. Die Grenzwerte werden so gewählt, dass etwa die beiden sechstel mit dem größten positiven, bzw. negativen Trend als steigend, bzw. fallend klassifiziert werden.

3.13.3 Ergebnis der experimentellen Untersuchung

Die wesentliche Erkenntnis, die aus den Prognosen über das Datenmaterial gewonnen wurde, besteht darin, dass dieses über die starken linearen Abhängigkeiten hinaus nur wenig Strukturen enthält, die eine wesentliche Verbesserung der Prognose erlauben würden. Die vorhandenen Abweichungen von den linearen Zusammenhängen sind zu einem Großteil als Rauschen zu interpretieren und nicht, wie zunächst erhofft, als erklärende Größe.

Trotzdem konnten alle Prognoseergebnisse insbesondere durch die Verwendung beider Vorhersagerichtungen verbessert werden. Das Potential der Betrachtungsweise der Daten ist also gegeben. Immerhin konnten zum Beispiel die besten neuronalen Netze für das im Sinne des Frühwarnsystems wichtige Kriterium der Korrektheit, die den Anteil der falsch alarmierten unter den auffälligen Teilen angibt, Werte um die 80% Sicherheit erreicht werden. Dabei wurden über 65% der tatsächlich auffälligen Teile gefunden. Durch die Hinzunahme der ebenfalls prognostizierten Konfidenzen lassen sich unter den angewarnten Teilen die fast sicheren Prognosen bestimmen.

Die Frage, welche Verfahren für die Prognose am besten geeignet sind, lässt sich nicht eindeutig beurteilen. Vom Standpunkt der Interpretierbarkeit nehmen sich die Verfahren für diesen Einsatzzweck nicht viel. Da die Hinzunahme weiterer bereits bekannter Datenpunkte die Ergebnisse fast immer verbessert hat, sind in der Regressionsgleichung oder den Entscheidungsbäumen so viele Variable, dass deren Bedeutung nicht mehr erfaßt werden kann. Das neuronale Netz ist durch seine inhärente Art der Wissensspeicherung ebenfalls nicht durch den Menschen interpretierbar.

Bei der Betrachtung der Geschwindigkeit sind die Unterschiede deutlicher. Die Regressionsgleichung kann mit Abstand am schnellsten bestimmt werden, C4.5 ist bei den betrachteten Größenordnungen der Datensätze ebenfalls sehr schnell (< 1 Minute). Das neuronale Netz ist hier mit Abstand am langsamsten. Erste relativ stabile Ergebnisse sind nach wenigen Minuten verfügbar, soll jedoch ein großes Netz eventuell mit Pruning trainiert werden, so sind auch nach Stunden noch leichte Verbesserungen möglich. Darüber hinaus kann bei den neuronalen Netzen nicht garantiert werden, dass sie ein globales Minimum finden. Dieses hat das Vergleichen der Läufe erschwert. Insbesondere durch zufällige Initialisierungen der Gewichte können auch die Ergebnisse mehrerer Läufe voneinander abweichen. Bei den durchgeführten Prognosen hat dies mitunter dazu geführt, dass Läufe schlechtere Ergebnisse ergeben haben, als erwartet wurde, bei einer Wiederholung die Ergebnisse jedoch besser waren. Dies ist auch darauf zurückzuführen, dass das Training der Netze aus Zeitgründen selten länger als 10 bis 15 Minuten gehalten wurde. Speziell bei den kleineren Netzen mit weniger Eingabevariablen treten nach dieser Zeitspanne kaum noch Änderungen in der Fehlerrate auf. Es kann jedoch nicht ausgeschlossen werden, dass mit anderen Netzwerkstrukturen und vor allem längeren Lernphasen die Ergebnisse noch verbessert werden können. Dies gehört sicher zu den großen Nachteilen der neuronalen Netze.

Zu den Leistungen der Verfahren lässt sich sagen, dass bei so eindeutigen linearen Zusammenhängen, wie sie bei der Prognose der kumulierten Quoten vorliegen, die lineare Regression den besten Kompromiss zwischen Leistung und Aufwand darstellt. Der nicht unbeachtliche Mehraufwand, den ein neuronales Netz mit sich bringt, kann hier kaum gerechtfertigt werden. Anders ist der Fall bei den Trends gelagert. Hier kann das neuronale Netz eher überzeugen. Ein Vorteil ist insbesondere, dass über die zugrunde liegenden Abhängigkeiten keine Annahmen gemacht werden müssen. Zwar lässt sich durch die angesprochene Hinzunahme der Kehrwerte einiger Punkte als Eingangsgrößen der Lernprozess etwas beschleunigen, bessere Endergebnisse werden aber dadurch nicht erzielt. Bei der Klassifikation schneidet das neuronale Netz etwas besser ab als C4.5. Allerdings ist die Generalisierungsfähigkeit des Entscheidungsbaumlerners schlechter, neigt also trotz Pruning in dieser Domäne zum Overfitting. Dies liegt wahrscheinlich daran, dass die funktionalen Abhängigkeiten zwischen den Eingangsattributen nur ungenügend und nur über Treppen von Entscheidungen als Entscheidungsbaum modelliert werden können. Da hier keine symbolischen Attribute auftreten, kann C4.5 seine möglicherweise vorhandene Überlegenheit auf diesem Gebiet nicht zum Einsatz bringen. Zusammenfassend ist zu sagen, dass die neuronalen Netze für die Prognose der Trends und der dazugehörigen Klassen von den

betrachteten Verfahren das geeignetste ist. Die Probleme der langen Trainingszeit spielen speziell in dieser Domäne keine dominierende Rolle, da lediglich ein Netz initial trainiert wird, welches allgemeingültige Abhängigkeiten lernen soll. Der Schluss von der ersten auf die zweite Jahreshälfte machte keine Probleme.

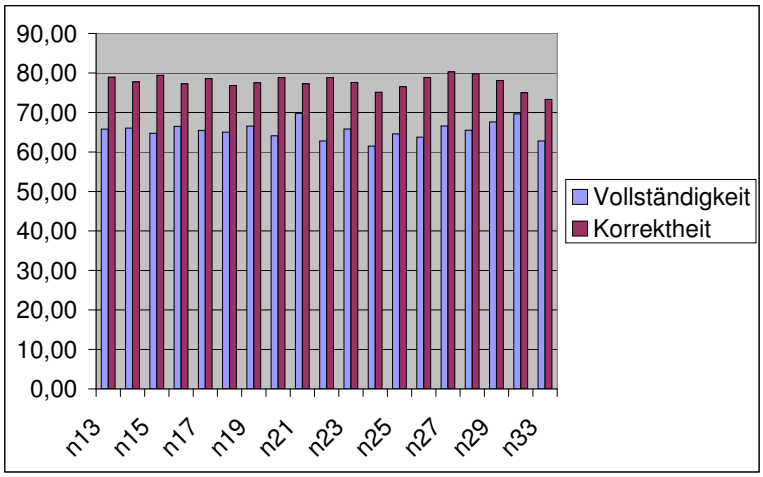

Abbildung 3.20: Vollständigkeit und Korrektheit der verschiedenen Netze für eine Baureihe

Da die gesuchten Abhängigkeiten jedoch sehr genereller Art und nicht auf Zeitpunkte innerhalb eines Jahres oder auf bestimmte Teile spezialisiert waren, ist eine „universelle“ Verwendbarkeit gegeben.

Die Prognose hat vom Einsatz des Clusterings indirekt profitiert. So konnten durch die Gruppierung größere Trainingsdatensätze generiert werden, die auch wechselseitig zur Validierung verwendet werden konnten.

Nach Rücksprache mit den Experten wurden die weiteren Aktivitäten auf die Trendprognose fokussiert.

Auf den vorherigen Ergebnissen aufbauend wurden weitere Tests auf aktuellen Baureihen durchgeführt und verschiedene Netzkonfigurationen ausprobiert. In der Abbildung 3.20 und 3.21 sind die Vollständigkeits- und die Korrektheitswerte der gelernten neuronalen Netze für die Prognose der drei Monatstrends dargestellt.

Die Vollständigkeit beschreibt, wieviel Prozent der auffälligen Teile, mit steigendem oder fallendem Trend, erkannt werden. Die Korrektheit hingegen beschreibt, wieviele der als auffällig klassifizierten Trends richtig erkannt wurden.

Aus den oben beschriebenen Tests hat sich ergeben, dass die Zeit wenig Einfluss auf die Güte der Ergebnisse hat, so dass man die Clusterbildung und das Trainieren der Netze nur alle sechs Monate wiederholen muss.

Weiterhin konnte beobachtet werden, dass die Verteilung der Trends in den Daten den größten Einfuss auf die Erkennungsleistung bei der Prognose der Drei-

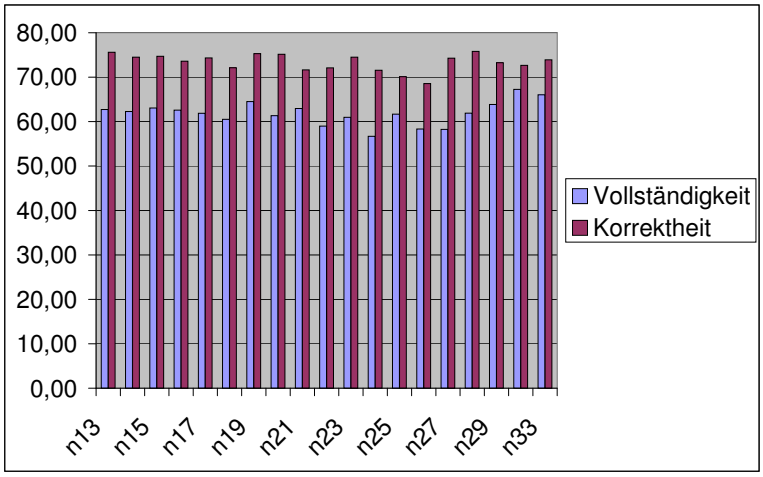

Abbildung 3.21: Vollständigkeit und Korrektheit der verschiedenen Netze für eine weitere Baureihe

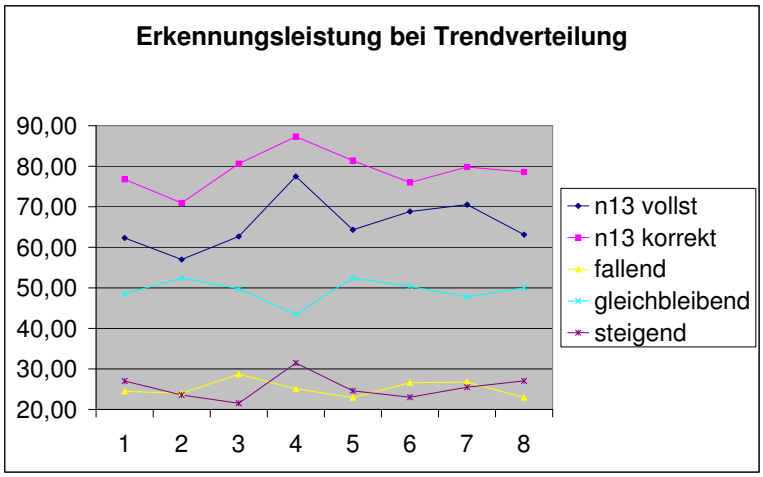

Abbildung 3.22: Erkennungsleistung bei Trendverteilung einer Baureihe

Monatskurve hat. Das heißt, wenn die Menge der auffälligen Teile größer ist, steigt die Klassifikationsgüte des Systems (siehe Abbildung 3.22).

3.14 Zuverlässigkeitsanalyse und Analyse von Lebensdauerdaten

Bei der Trendprognose von Ausfallraten könnte man auch von einer Schätzung der Lebenserwartung des Bauteils sprechen. Das wohl bekannteste Verfahren, im Rahmen der Qualitätssicherung auch oft eingesetzte Verfahren, ist das Schätzen der Weibull-Verteilung. Alternativ zu den bisher vorgestellten Ansätzen wurde dieser statistische Ansatz auf seine Einsetzbarkeit geprüft.

Die Analyse von Lebensdauer- oder auch Ausfalldaten ist ein Teilbereich der Statistik, der besonders in den Bereichen der Biomedizin und der Ingenieurwissenschaften sehr an Bedeutung gewonnen hat. Die Einsatzmöglichkeiten reichen dabei von Untersuchungen der Lebensdauer produzierter Teile oder Systeme bis zu Vergleichsstudien von Medikamenten, deren Wirksamkeit ermittelt werden soll. Eine sehr wichtige Methode zur Untersuchung solcher Daten sind parametrische stochastische Modelle. Mit Hilfe von bestimmten Wahrscheinlichkeitsverteilungen wird versucht, das Ausfallverhalten der zu untersuchenden Teile über ihre Lebensdauer zu modellieren. Dabei wird die Betriebszeit, die Anzahl korrekt durchgeführter Operationen oder auch die Anzahl geleisteter Kilometer als Zufallsvariable betrachtet, die von vielen Einflüssen abhängig sein kann, wie etwa dem verwendeten Material oder der Herstellungsart. Für die Modellierung müssen häufig mehrere unbekannte Parameter der Verteilung anhand der vorliegenden Daten ermittelt werden. Sobald dies erfolgreich geschehen ist, ist man in der Lage, Aussagen über den weiteren Verlauf des Ausfallverhaltens zu treffen. Auf diesem Wege lassen sich Qualitätsprobleme entdecken.

Für die Analyse von Lebensdauerdaten existieren viele potentielle Modelle, wobei die Motivation häufig eher empirisch als theoretisch belegt ist. Entweder beschreibt das gewählte Modell die vorhandenen Daten auf Anhieb, oder frühere Untersuchungen haben gezeigt, dass dieses Modell gute Ergebnisse für ähnliche Daten ergeben hat, oder aber Kenntnisse über den zugrundeliegenden Fehlerprozess lassen das Modell plausibel erscheinen, etc.. Manchmal wird ein Modell als unangemessen betrachtet, weil die Hazard-Funktion die falsche Ausprägung vorweist, obwohl sie die Daten gut repräsentiert. Ein solcher Fall ist die Log-Normalverteilung, die ab einem gewissen Zeitpunkt eine fallende Hazard-Funktion besitzt und daher als Modell ausscheidet, obwohl die vorhandenen Daten eventuell gut angenähert werden.

Häufig werden die Verteilungen aber auch nach der Verfügbarkeit von statistischen Methoden und ihrer mathematischen Handhabbarkeit ausgewählt. Dies war einer der Gründe, weshalb die Exponentialverteilung lange das dominierende Modell in der Analyse von Ausfalldaten war. Es handelt sich um ein verhältnismäßig einfaches Modell, das eine Reihe guter statistischer Eigenschaften vorzuweisen hatte. Ähnlich ist der Erfolg der Weibullverteilung zu betrachten, für die mit der Zeit mehr und mehr gute Schätzverfahren entwickelt worden sind. Die Weibullverteilung wird aufgrund ihrer größeren Flexibilität öfter der Exponentialverteilung vorgezogen. Ein weiterer Grund für den zunehmenden Einsatz der Weibullverteilung ist die Verfügbarkeit leistungsstarker Computer, die zur Berechnung herangezogen werden können.

Vorstudien haben gezeigt, dass eine genauere Betrachtung der Weibullanalyse empfehlenswert ist. Allerdings sind die bisher verwendeten Werkzeuge zur Durchführung einer Fallstudie weniger geeignet, da sie schlecht an die Aufgabenstellung angepasst werden können. Daher wurde ein Prototyp entwickelt, der die theoretischen Verfahren zur Berechnung eines Weibullmodells implementiert. Dabei soll es möglich sein, zu einer gegebenen Datenbasis die zugehörige Weibullverteilung bzw. deren Parameter mittels der Rank Regression und des Maximum Likelihood Verfahrens zu ermitteln. Auf der Basis dieser Schätzer wird dann eine Prognose für das zukünftige Ausfallverhalten der untersuchten Bauteile berechnet. Dabei sind sowohl Punkt- als auch Trendprognosen möglich. Um auch eine Güte der Prognosen abzuschätzen, erfolgt zusätzlich auch ein Vergleich mit den wahren Ausfalldaten, die bis zu einem bestimmten Punkt bereits vorlagen. Damit eine möglichst große Portabilität gewährleistet werden konnte, wurde die Implementierung in *C* angefertigt, so dass dadurch auch eine spätere Einbindung in das Data Mining Tool Clementine™ erleichtert wird.

Im Folgenden soll anhand einer Fallstudie gezeigt werden, inwieweit sich die Weibullverteilung für die Prognose von Ausfällen im Automobilbereich eignet. Dabei wurde im Gegensatz zu den Vorstudien sowohl eine größere Anzahl an Bauteilen als auch ein längerer Beobachtungszeitraum untersucht. Die Datenbasis bildete ein Auszug aus der Datenbank einer bestimmten Baureihe über einen Beobachtungszeitraum von zwei Jahren. Dieser Auszug wurde weiterhin mittels dem Data Mining Tool Clementine™ so aufbereitet, dass nur noch Bauteile zur Untersuchung standen, die eine Ausfallquote von mindestens 0.1% über diesen Zeitraum aufwiesen. Nach dieser Vorverarbeitung standen noch 209 Schadensschlüssel zur weiteren Analyse bereit. Die Ausfalldaten lagen für jedes Bauteil in Form von Intervalldaten über einen Zeitraum von jeweils 15 Tagen vor.

3.14.1 Berechnung der Prognosen mit Weibull

Auf der Grundlage der oben beschriebenen Datenbasis wurde für jedes Bauteil bezüglich seines Betriebsalters fortlaufend das zugehörige Weibullmodell berechnet. Für den Produktbetreuer und die Herstellerfirma sind bei der Auswertung von Ausfalldaten besonders die Quartalsausfallkurven interessant, d. h. das kumulierte Ausfallverhalten nach einem Zeitraum von jeweils drei Monaten (siehe auch Abschnitt 3.9). Daraus ergeben sich in dieser Analyse die Ausfallkurven nach 3, 6, 9, 12, 15 und 18 Monaten, für die eine Trendberechnung möglich ist. Somit erfolgte die Berechnung nicht nach jedem 15-Tages-Intervall, sondern immer genau dann, wenn der Abstand zur nächsten Quartalskurve noch 75 Tage, also $2\frac{1}{2}$ Monate, betrug. Dieser Wert ist zum einen dadurch begründet, dass die möglichen Ausfälle der kommenden drei Monate natürlich so früh wie möglich bekannt sein sollten, was eine spätere Prognose uninteressant machen würde. Zum anderen kann natürlich auch nicht zu früh prognostiziert werden, da zu wenig Daten vorliegen und der Verlauf zu unsicher wäre. Dies macht sich gerade bei der Prognose der 3-Monatskurve bemerkbar. Bei einem Prognosehorizont von 75 Tagen sind immerhin schon die Ausfalldaten der

ersten 15 Tage des betrachteten Produktionsmonats verfügbar.

Wie bereits in den Vorstudien empfohlen, wurden für die Berechnungen zusätzlich die Ausfalldaten der drei vorangegangenen Produktionsmonate mit einbezogen, damit eine ausreichende Datenmenge garantiert war. Die Berechnungen pro Zeitpunkt und Schadensteil beinhalten:

1. Schätzung der Weibullparameter anhand der vorliegenden Ausfalldaten,
2. Berechnung einer Punktprognose für die kommende Quartalskurve,
3. Berechnung eines Trends anhand der eben errechneten Punktprognose und der Punktprognose des letzten Produktionsmonats bei zwei Punkten bzw. der letzten beiden Produktionsmonate bei drei Punkten und
4. Vergleich mit den Realdaten.

Da beim Schätzen der Parameter zusätzlich auch Intervalle geschätzt wurden, bestand die Möglichkeit auch Intervalle für die Punktprognosen zu berechnen. Auf diese Möglichkeit wurde hier aus mehreren Gründen verzichtet. Zum einen sollte ein Vergleich zwischen Maximum Likelihood und Rank Regression ermöglicht werden. Da beide Verfahren aber unterschiedliche Methoden zur Intervallschätzung benutzen, wäre die Vergleichbarkeit nicht im vollen Umfang gegeben gewesen. Zum anderen hängt die Größe des berechneten Intervalls auch von der Menge der Ausfälle ab. Dies führt bei geringer Ausfallmenge zu sehr kleinen Intervallen, die oft nicht einmal ein ganzes Bauteil enthalten. Somit wäre eine Treffermarkierung auf dieser Basis nicht aussagekräftig genug bezüglich der Güte der Prognose. Aus diesen Gründen wurde ein festes Intervall um die Punktprognose als Trefferintervall gewertet.

3.14.2 Ergebnisse

Die Analysen erfolgten über eine Grundgesamtheit von 209 Schadensschlüsseln. Zusätzlich zu diesen „globalen“ Auswertungen wurden auch drei einzelne Bauteile getrennt dazu untersucht, um die Ergebnisse an konkreten Beispielen zu verdeutlichen.

3.14.2.1 Punktprognosen

Für die Punktprognosen wurde ein Intervall von ± 6 Ausfällen um den eigentlichen Punktschätzer als Treffer und somit als korrekt prognostizierter Wert betrachtet. Im Folgenden sollen zuerst die Ergebnisse des Maximum Likelihood und des Median Rank Regression Verfahrens bezüglich der Analyse von 209 Schadensschlüsseln beschrieben werden.

Die Datenbasis beschreibt die Ausfälle, die in einem Zeitraum von zwei Jahren aufgetreten sind. Somit stehen theoretisch 24 Produktionsmonate für die Analyse zur

Verfügung. Allerdings wurden zur Datenunterstützung, wie in den Vorstudien empfohlen, zusätzlich für jeden Produktionsmonat seine drei vorangegangenen Produktionsmonate in die Analyse miteinbezogen. Daher stehen effektiv nur 21 Produktionsmonate für die nähere Untersuchung zur Verfügung.

Weiterhin wurde keine durchgehend fortlaufende Analyse durchgeführt, sondern feste Zeitpunkte im Betriebsleben einer Monatsproduktion gewählt, für die eine Prognose hinsichtlich der nächst folgenden Quartalskurve berechnet worden ist. Eine zusätzliche Einschränkung bildete die Verfügbarkeit der Vergleichsdaten, die ebenfalls diesem Datenauszug von zwei Jahren entnommen wurde. Zum Ende des aufgezeichneten Ausfallverhaltens eines jeden Produktionsmonats hätten durchaus noch weitere Prognosen angestellt werden können. Diese hätten jedoch Datenpunkte beschrieben, die nicht mehr im untersuchten Datenauszug vorhanden gewesen wären, und somit hätte kein Vergleich erfolgen können.

Aus diesen Bedingungen folgt letztendlich eine Gesamtanzahl von 14630 Punktprognosen, die berechnet worden sind, d. h. pro Schadensschlüssel 70 Punktprognosen. Die Aufteilung der Prognosen auf die berechneten Ausfallkurven und die Prognoseintervalle[16] ist in Tabelle 3.4 dargestellt.

	Maximum Likelihood						
	Gesamtprognosen	0-10	10-25	25-50	50-75	75-100	>=100
3-Monatskurve	3781	1875	1288	459	74	33	52
6-Monatskurve	3323	1462	1303	366	99	48	45
9-Monatskurve	2713	1122	1076	339	73	54	49
12-Monatskurve	2088	747	860	305	91	43	42
15-Monatskurve	1462	543	575	225	65	20	34
18-Monatskurve	834	534	210	60	8	10	12
21-Monatskurve	208	165	34	6	0	3	0
	Median Rank Regression						
	Gesamtprognosen	0-10	10-25	25-50	50-75	75-100	>=100
3-Monatskurve	3810	1895	1294	461	75	33	52
6-Monatskurve	3331	1467	1304	367	100	48	45
9-Monatskurve	2717	1126	1076	339	73	54	49
12-Monatskurve	2090	749	860	305	91	43	42
15-Monatskurve	1463	544	575	225	65	20	34
18-Monatskurve	836	536	210	60	8	10	12
21-Monatskurve	209	166	34	6	0	3	0

Tabelle 3.4: Globale Aufteilung der Punktprognosen

In dieser Tabelle wird beispielsweise dargestellt, wieviele Prognosen auf die Berechnung der 3-Monatskurven fiel, und in welchen Größenordnungen sich die Prognosen bewegten. Gleiches gilt für die übrigen Monatskurven. Das globale Trefferergebnis der berechneten Punktprognosen ist in Tabelle 3.5 beschrieben.

Der Ausschuss betrifft die Zeitpunkte, zu denen keine Analyse möglich war. Dies kann mehrere Gründe haben. Zum einen können trotz Datenunterstützung zu wenig Datenpunkte verfügbar sein, so dass das verwendete Verfahren nicht in der Lage ist, das Weibullmodell zu berechnen. Zum anderen können innerhalb der Modellberech-

[16] Die Spalten drei bis acht beinhalten die Anzahl der berechneten Prognosen auf einem Ausfallniveau. Die Spaltenüberschriften geben die Grenzen der Niveaus in absoluten Ausfällen an.

	Maximum Likelihood		Median Rank Regression	
	Anzahl	Prozentual	Anzahl	Prozentual
Ausschuß	221	1.51%	174	1.19%
Treffer	8584	58.67%	6653	45.48%
Fehler	5825	39.81%	7803	53.34%
zu hoch	4105	70.47%	6510	83.43%
zu niedrig	1720	29.53%	1293	16.57%

Tabelle 3.5: Globales Ergebnis der Punktprognosen

nung ungültige Parameterwerte auftauchen, so dass das Modell verworfen werden muss und keine weitere Analyse möglich ist.

Die fehlgeschlagenen Prognosen wurden zusätzlich in zu hoch und zu niedrig angesetzte Prognosewerte unterteilt. Die prozentualen Werte beziehen sich dabei auf die Fehlprognosen als Grundgesamtheit. Aus dieser Aufschlüsselung ist zu erkennen, dass eine fehlgeschlagene Prognose eher zu hoch schätzt als zu niedrig. Dies führt oft auch dazu, dass die Ausfallkurven eines einzelnen Schadensschlüssels trotz fehlgeschlagener Punktprognosen von der Form auf einem höheren Niveau sehr gut nachgebildet werden können. Auf dieses Phänomen wird in den Einzelanalysen noch näher eingegangen.

Ein erstes Ergebnis, das an dieser Stelle bereits angegeben werden kann, ist die bessere Treffsicherheit eines mit der Maximum Likelihood Methode berechneten Weibullmodells. Bei der Median Rank Regression Methode ist zwar der Ausschuss geringer, aber auch die Treffsicherheit. Daher eignet sich die Maximum Likelihood Methode trotz des verhältnismäßig hohen erforderlichen Rechenaufwands eher zur Modellberechnung als die Median Rank Regression Methode. Zusätzliche Analysen anderer Baureihen lieferten ähnliche Ergebnisse, die diese Aussage bekräftigen.

Zusätzlich zu dieser globalen Analyse wurden auch Berechnungen durchgeführt, die sich auf nur einen einzelnen Zeitpunkt im Betriebsleben der betrachteten Bauteile festlegen. Dadurch erfolgt die Schätzung einer speziellen Ausfallkurve, beispielsweise der nach drei Monaten Laufzeit. Dabei stellte sich heraus, dass die Prognosen der frühen Ausfallkurven eine höhere Trefferquote aufwiesen, als die der späteren. Dies ist zu einem hohen Grad auf das fest gewählte Trefferintervall zurückzuführen. Zu diesem Zeitpunkt existieren noch relativ wenig Ausfälle, die aufgrund dieses Intervalls gut getroffen werden. Tabelle 3.6 beschreibt die Verhältnisse der fehlgeschlagenen Prognosen relativ zu den für die jeweilige Ausfallkurve durchgeführten Prognosen.

	Maximum Likelihood		Median Rank Regression	
	Anzahl	Prozentual	Anzahl	Prozentual
3-Monatskurve	1119	29.59%	1560	40.94%
6-Monatskurve	1326	39.90%	1708	51.28%
9-Monatskurve	1144	42.17%	1571	57.82%
12-Monatskurve	888	42.53%	1234	59.04%
15-Monatskurve	691	47.26%	898	61.38%
18-Monatskurve	518	62.11%	648	77.51%
21-Monatskurve	139	66.83%	184	88.04%

Tabelle 3.6: Globales Aufteilung der fehlerhaften Punktprognosen

Weiterhin wurde beobachtet, dass das Verhältnis zwischen zu hohen und zu niedri-

gen Prognosen mit fortschreitender Betriebszeit zugunsten der zu hoch angesetzten Prognosen zunimmt. Beispielsweise sind die Fehlprognosen der 21-Monatskurve zu 99% zu hoch geschätzt.

Trotz der zu hoch geschätzten Punktprognosen lässt sich beobachten, dass oft mit zunehmender Betriebszeit der Verlauf der Ausfallkurve gut nachgebildet werden kann. Im Folgenden soll dies an drei Beispielen verdeutlicht werden. Dazu wurden drei Bauteile ausgewählt, deren Ausfallkurven bis zur 12-Monatskurve in den Abbildungen 3.23 bis 3.25 dargestellt sind:

1. Dichtring Antriebskegelrad
2. Abdichtring Achsgehäuse
3. Elektrischer Fensterheber Vordertür

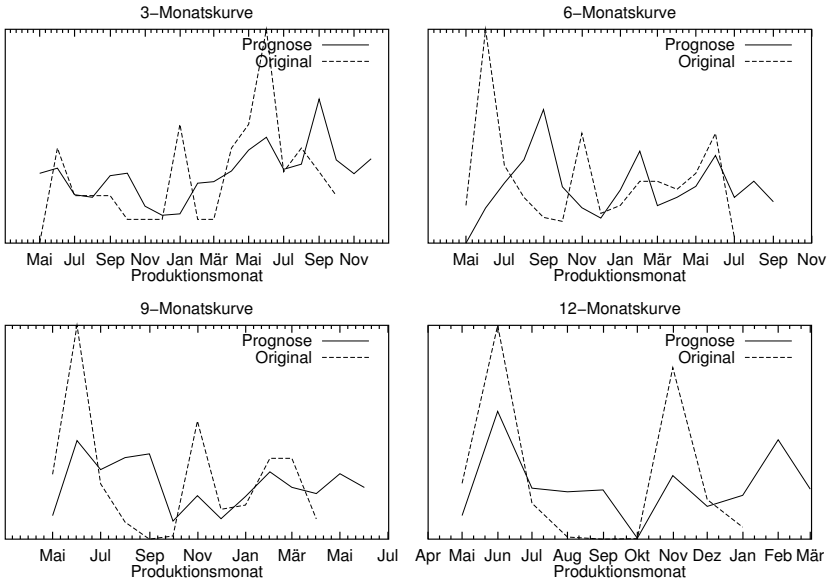

Abbildung 3.23: Ausfallkurven Dichtring Antriebskegelrad

Bei allen drei Bauteilen erkennt man, dass die Prognosen der 3-Monatskurven selten den Originalverlauf treffen. Allerdings existieren bis zu diesem Zeitpunkt nur wenig Ausfälle, so dass trotz des schlechten Verlaufs und aufgrund des fest gewählten Trefferintervalls eine sehr gute Trefferquote erreicht wird (siehe Tabelle 3.7).

	Dichtring		Abdichtring		Fensterheber	
	ML	MRR	ML	MRR	ML	MRR
Ausschuss	0.00%	0.00%	15.79%	15.79%	0.00%	0.00%
Treffer	100.00%	100.00%	84.21%	84.21%	47.37%	10.53%
Fehler	0.00%	0.00%	0.00%	0.00%	52.63%	89.47%
zu hoch	0.00%	0.00%	0.00%	0.00%	70.00%	82.35%
zu niedrig	0.00%	0.00%	0.00%	0.00%	30.00%	17.65%

Tabelle 3.7: Trefferquoten für die 3-Monatskurve

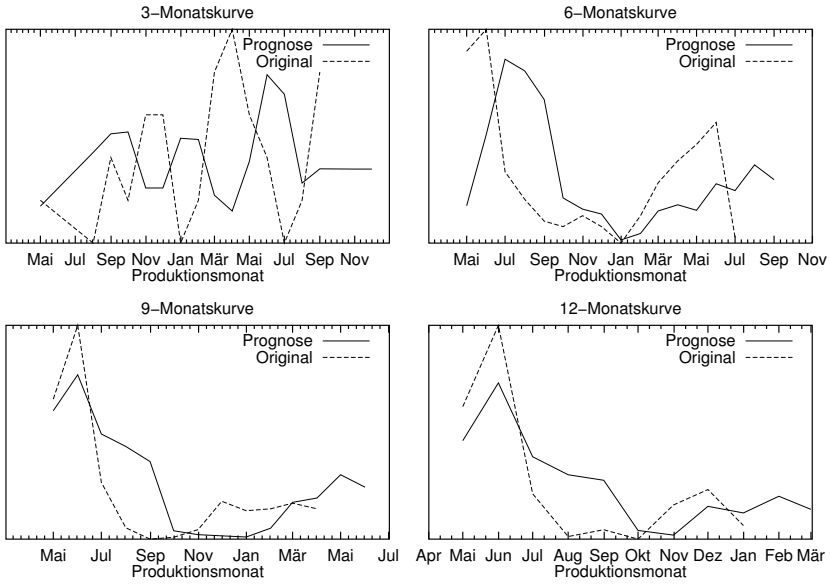

Abbildung 3.24: Ausfallkurven Abdichtring Achsgehäuse

Eine Ausnahme bei diesen Bauteilen ist der Fensterheber, der entgegen den anderen beiden Bauteilen eine besonders geringe Trefferquote aufweist. Beim Betrachten der entsprechenden Ausfallkurven erkennt man, dass der Grund hierfür das oszillierende Ausfallverhalten des Fensterhebers ist. Gerade bei der 3-Monatskurve wird die Trägheit des Weibullmodells ersichtlich. Prognostizierte Anstiege erscheinen häufig erst kurz nach dem originalen Anstieg.

Mit fortlaufender Betriebszeit passen sich die prognostizierten Ausfallkurven immer besser der originalen Ausfallkurve an, was hauptsächlich auf die größer werdende Datenbasis zurückzuführen ist. Besonders die Kurve des Fensterhebers erreicht eine nahezu optimale Anpassung, obwohl die Trefferquoten dieses Bauteils mit zunehmender Lebensdauer nicht wesentlich besser werden. Der Dichtring Antriebskegelrad zeigt auch in der 12-Monatskurve noch große Schwankungen, die zwar vom Verlauf her mit modelliert werden, aber meist auf niedrigerem Niveau. Ähnliches gilt auch für den Abdichtring Achsgehäuse.

Bei einer isolierten Betrachtung der Trefferquoten, die das Weibullmodell erreicht, sind die Ergebnisse wenig befriedigend. Man würde sich viel mehr eine höhere Trefferquote wünschen. Wenn allerdings zusätzlich der Verlauf der originalen und der prognostizierten Ausfallkurven mit in die Bewertung einbezogen wird, so erkennt man, dass die Weibullanalyse in einem Großteil der Fälle den realistischen Daten ziemlich nahe kommt.

Zusätzlich zu diesen Ergebnissen ist zu erwähnen, dass die Datenbasis zur Berechnung des Modells nicht optimal ist. Dieses ist durch die Aufgabenstellung begründet. Zum einen sind gerade in der Frühphase wenig Daten vorhanden, so dass zusätzlich vorangegangene Produktionsmonate mit in die Analyse einbezogen werden müssen. Dies hat den bereits mehrfach angedeuteten Nachteil der Verfälschung des Ausfall-

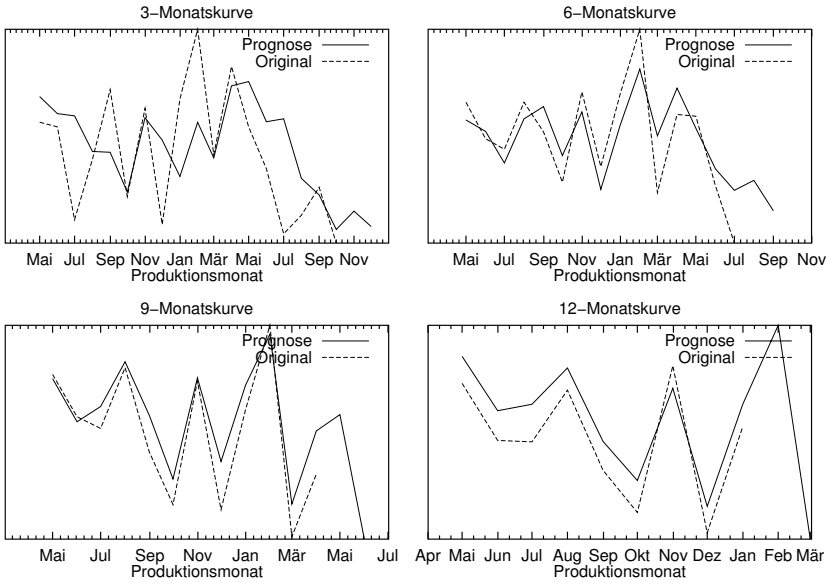

Abbildung 3.25: Ausfallkurven Elektrischer Fensterheber Vordertür

verhaltens. Weiterhin beinhalten die Daten absolute Ausfallzahlen innerhalb eines Kalenderzeitraums. Dabei wird nicht berücksichtigt, wieviele Fahrzeuge sich bereits im Einsatz befinden. Vielmehr wird davon ausgegangen, dass alle Fahrzeuge von Beginn der Analyse an berücksichtigt werden. Der Vorteil dieser Betrachtung ist die Möglichkeit einer Prognose für einen bestimmten Kalenderzeitpunkt, beispielsweise der Ausfallquote für das dritte Quartal eines Jahres. Bei der Betrachtung relativer Daten, d. h. Ausfalldaten relativ zur Erstzulassung eines Fahrzeugs, können solche Prognosen nicht ohne weiteres berechnet werden. Die Ausfallquoten nach Betriebszeit eines Bauteils sollten zwar höhere Trefferquoten aufweisen, aber für eine Projektion auf einen Kalenderzeitraum müssen zusätzliche Annahmen gemacht werden. Diese betreffen hauptsächlich die Zeit zwischen Produktion und Erstzulassung.

3.14.2.2 Trendprognosen mit Weibull

Mittels der oben beschriebenen Punktprognosen wurden zusätzlich Trendprognosen errechnet, die ein steigendes, gleichbleibendes oder fallendes Ausfallverhalten beschreiben sollen. Wie bereits in Abschnitt 3.9 beschrieben, konnten zur Trendprognose zwei oder drei Stützpunkte verwendet werden. In beiden Fällen wird mittels dieser Punkte eine prozentuale Abweichung errechnet. Sollte auf der Basis von zwei Punkten geschätzt werden, so wird die prozentuale Abweichung des zweiten Punktes relativ zum ersten Punkt berechnet. Bei der Betrachtung von drei Punkten erfolgt eine Mittelung des zweiten und dritten Punktes. Der Trend ergibt sich schließlich aus der prozentualen Abweichung dieses neuen Punktes relativ zum ersten Punkt. Die Methode der Neuronalen Netze aus den früheren Untersuchungen benutzt ebenfalls letzteres Verfahren. Da die Weibullanalyse hauptsächlich mit diesem Modell verglichen werden soll, wird an dieser Stelle ein größeres Gewicht auf die Ergebnisse

der Trendprognose, basierend auf drei Punkten, gelegt.

Der wichtigste Schritt ist die Definition der Schwellwerte, die über die Trendklassifizierung entscheiden. Die vom Hersteller verfolgte Strategie legt die Annahme zugrunde, dass ca. 50% aller Trends gleichbleibend sind. Aus dieser Annahme heraus wurden die Schwellwerte empirisch ermittelt. Für die Analysen des Weibullmodells ergab sich ein Schwellwert von 0.2, also eine Abweichung von mehr als ±20% wird als Trend bezeichnet.

Wie bereits erwähnt, basieren die Trendprognosen auf den Ergebnissen der Punktprognosen. Da jedoch mindestens drei Werte zur Bestimmung eines Trends vorhanden sein müssen, kommt es zu einer Verzögerung bis zur ersten Trendberechnung. Dies schlägt sich in einer geringeren Anzahl berechneter Trends im Vergleich zur Anzahl der Punktprognosen nieder.

In Tabelle 3.8 werden die Ergebnisse der Trendprognosen unterteilt nach verwendetem Berechnungsverfahren und zugehöriger Ausfallkurve dargestellt. Es werden hier nur die Ergebnisse der ersten drei Ausfallkurven präsentiert, da diese von besonderem Interesse im Produktbewährungsprozess sind.

	3-Monatskurve		6-Monatskurve		9-Monatskurve	
ungewichtet	ML	MRR	ML	MRR	ML	MRR
Ausschuss	5.21%	4.51%	0.72%	0.44%	0.17%	0.00%
Korrekt gesamt	32.97%	31.76%	55.95%	54.17%	69.38%	67.42%
Korrekt steigend	14.29%	17.83%	25.23%	25.24%	25.27%	26.19%
Korrekt gleichbl.	71.34%	63.81%	51.07%	51.10%	50.85%	50.00%
Korrekt fallend	14.37%	18.36%	23.70%	23.66%	23.89%	23.80%
Fehler gesamt	61.82%	63.72%	43.34%	45.39%	30.45%	32.58%
Falsch + statt =	9.89%	11.97%	14.20%	13.48%	15.71%	16.82%
Falsch + statt -	11.93%	13.90%	4.02%	4.74%	1.29%	2.00%
Falsch = statt +	30.02%	26.44%	30.91%	29.44%	30.43%	27.37%
Falsch = statt -	29.61%	24.90%	33.68%	32.45%	35.29%	33.91%
Falsch - statt +	9.30%	10.65%	3.15%	4.37%	0.86%	1.87%
Falsch - statt =	9.25%	12.14%	14.04%	15.51%	16.43%	18.02%
	3-Monatskurve		6-Monatskurve		9-Monatskurve	
gewichtet	ML	MRR	ML	MRR	ML	MRR
Ausschuss	8.89%	4.51%	1.06%	0.44%	0.22%	0.00%
Korrekt gesamt	32.88%	32.97%	59.33%	57.04%	72.51%	68.29%
Korrekt steigend	11.68%	14.80%	27.19%	28.34%	27.47%	28.03%
Korrekt gleichbl.	78.52%	73.64%	45.73%	44.64%	47.21%	46.43%
Korrekt fallend	9.80%	11.56%	27.07%	27.02%	25.31%	25.54%
Fehler gesamt	58.23%	62.52%	39.61%	42.52%	27.23%	31.71%
Falsch + statt =	7.13%	9.10%	17.52%	17.77%	19.97%	20.71%
Falsch + statt -	8.28%	10.94%	2.93%	3.54%	0.96%	1.51%
Falsch = statt +	35.63%	31.17%	29.68%	26.21%	25.40%	23.59%
Falsch = statt -	36.34%	31.84%	30.97%	30.06%	33.23%	31.00%
Falsch - statt +	5.54%	7.89%	2.24%	4.02%	0.64%	1.78%
Falsch - statt =	7.08%	9.06%	16.65%	18.41%	19.81%	21.40%

Tabelle 3.8: Ergebnis der Trendprognosen

Die 3-Monatskurve weist noch eine hohe Rate gleichbleibender Trends auf, die mit der 6- und 9-Monatskurve abfällt. Diese hohe Erkennungsrate gleichbleibender Trends im Vergleich zu den steigenden bzw. fallenden Trends ist auf eine Erkenntnis aus den Punktprognosen zurückzuführen. Dort wurde festgestellt, dass besonders bei der Prognose der 3-Monatskurve das Modell eine starke Trägheit aufweist. Folglich werden Änderungen im Ausfallverhalten erst "spät" nach modelliert, so dass

falsche Trends erkannt werden. Weiterhin wird anfangs nur ein geringer Anteil der Trends korrekt erkannt, der aber bei den späteren Kurven stark zunimmt. Demgegenüber verringert sich mit zunehmender Betriebsdauer die Anzahl falsch klassifizierter Trends.

Desweiteren sind die Quoten der prognostizierten Trends, die das genaue Gegenteil des originalen Trends beschreiben, also beispielsweise steigend statt fallend, sehr gering. Nur bei der 3-Monatskurve liegen sie mit ca. 21.2% (Maximum Likelihood) bzw. 24.5% (Median Rank Regression) im ungewichteten Fall recht hoch. Mit zusätzlicher Gewichtung liegt diese Quote nur noch bei ca. 13.8% (Maximum Likelihood) bzw. 18.8% (Rank Regression). Diese Erkenntnis ist ebenfalls auf die Trägheit des Modells im Bereich der 3-Monatskurve zurückzuführen. Wenn folglich für spätere Ausfallkurven ein steigender Trend prognostiziert wird, so ist er im Fehlerfall meist gleichbleibend, selten aber fallend. Gleiches gilt auch für fallend prognostizierte Trends.

Man erkennt anhand dieser Tabellen, dass mit Gewichtung der Daten eine leichte Verbesserung im Erkennen der Trends erfolgt. Gleichzeitig hat dies aber auch einen höheren Ausschuss bei der Maximum Likelihood Methode zur Folge. Allerdings sind hohe Ausschussquoten meist nur bei den frühen Ausfallkurven zu beobachten. Mit fortlaufender Dauer fallen diese unter 1%.

Verglichen mit den Punktprognosen werden die Trendprognosen mit zunehmender Betriebszeit der betrachteten Bauteile besser. Während bei den frühen Ausfallkurven noch Fehlerraten von über 55% auftreten, so nehmen diese mit fortlaufender Dauer immer weiter ab. Bei der 18-Monatskurve beträgt der Fehler schließlich weniger als 6%. Tabelle 3.9 gibt detaillierten Aufschluss über die einzelnen Fehlerraten relativ zu der Anzahl berechneter Prognosen. Dabei ist anzumerken, dass durchgängig eine leichte Verbesserung der Prognosen erreicht werden kann, indem die Daten gewichtet werden.

	ungewichtet		gewichtet	
	ML	MRR	ML	MRR
3-Monatskurve	58.31%	59.66%	57.26%	58.53%
6-Monatskurve	38.16%	39.87%	35.03%	37.35%
9-Monatskurve	25.80%	27.57%	23.09%	26.83%
12-Monatskurve	19.68%	20.62%	17.94%	20.62%
15-Monatskurve	14.36%	15.79%	12.55%	14.15%
18-Monatskurve	5.52%	7.78%	4.81%	6.22%

Tabelle 3.9: Globale Aufteilung der fehlerhaften Trendprognosen

3.14.3 Vergleich der Neuronale Netze vs. Weibull–Analyse

Wie bereits in Abschnitt 3.9 beschrieben, wird die Güte des Modells in Bezug auf die Trendprognose mit den Quotienten q_1 (Vollständigkeit) und q_2 (Sicherheit) bewertet. In Tabelle 3.10 wird das Weibullmodell direkt mit dem Neuronalen Netz aus den früheren Untersuchungen verglichen. Der Auftraggeber legt bei den Trendprognosen

besonderen Wert auf die frühen Ausfallkurven. Daher werden im Folgenden nur die 3-, 6- und 9-Monatskurve zum Vergleich herangezogen.

Man erkennt natürlich schnell, dass das Weibullmodell dabei schlechter abschneidet als das Neuronale Netz. Das Weibullmodell wird erst mit zunehmender Betriebszeit besser. Dazu ist anzumerken, dass auch ein Neuronales Netz bei den späteren Ausfallkurven bessere Ergebnisse erzielt. Somit erzielt das Weibullmodell in keinem Fall bessere Ergebnisse als das Neuronale Netz.

Weibull	ML		MRR	
	q_1	q_2	q_1	q_2
3-Monatskurve	0.1590	0.2747	0.1920	0.2705
6-Monatskurve	0.4681	0.6408	0.4511	0.6050
9-Monatskurve	0.6227	0.7656	0.6136	0.7277
Weibull gewichtet	**ML**		**MRR**	
	q_1	q_2	q_1	q_2
3-Monatskurve	0.1239	0.3022	0.1452	0.2731
6-Monatskurve	0.5525	0.6738	0.5378	0.6294
9-Monatskurve	0.7000	0.7726	0.6659	0.7176
Neuronales Netz				
	q_1	q_2		
3-Monatskurve	0.6033	0.7113		
6-Monatskurve	0.7550	0.8360		
9-Monatskurve	0.8200	0.8400		

Tabelle 3.10: Vergleich Neuronales Netz - Weibullmodell

Besonders die Werte für die frühen Ausfallkurven liefern schlechte Ergebnisse. Diese Tatsache ist größtenteils auf die Trägheit des Weibullmodells in dieser Phase zurückzuführen, wie oben bereits angedeutet wurde. Auf den 3-Monatskurven kommt es häufig zu einem oszillierenden Ausfallverhalten. Durch die Trägheit werden diese sehr kurzfristigen Trends nur schlecht erkannt. Das Neuronale Netz kann sich diesem Umstand anscheinend besser anpassen.

Zusammengefasst liefert das Neuronale Netz in jedem Fall die besseren Ergebnisse. Die Trendprognosen des Weibullmodells sind im Vergleich zum Neuronalen Netz gerade für die frühen Ausfallkurven als schlecht zu bezeichnen. Zwar verbessern sich die Trendprognosen mit zunehmender Betriebszeit der untersuchten Bauteile, aber sie bleiben dennoch unter den Ergebnissen des Neuronalen Netzes, die sich ebenfalls verbessern. Ähnliches spiegelt sich in den Punktprognosen wieder. Anhand der Punktprognosen wurde auf grafischem Wege die Anpassung der prognostizierten Ausfallkurve an die wahre Ausfallkurve ermittelt. Auch dabei wurden bei den frühen Kurven schlechtere Ergebnisse als bei späteren erzielt.

Kapitel 4

Ergebnisse zur KDD–Anwendung im Produktbewährungsprozess

Bevor auf die Ergebnisse der Anwendungen eingegangen wird, beschreibt Abschnitt 4.1 wie für die beiden Fragestellungen im Produktbewährungsprozesses die Datenverarbeitungsschritte des KDD–Prozesses realisiert wurden. In den folgenden Abschnitten werden Ergebnisse, die ich während meiner Arbeit bei einem Automobilhersteller erzielt habe, vorgestellt. Diese realen Fälle dienten der Evaluierung der zuvor beschriebenen, entwickelten Methoden und Vorgehensweisen, sowie der Unterstützung der Produktbetreuer bei ihrer täglichen Arbeit.

In dem vorherigen Kapitel 3 wurde die Anwendung des Produktbewährungsprozesses vorgestellt bis einschließlich der Algorithmenauswahl für die beiden Aufgaben der Analyse von Schadensschwerpunkten und der Trendprognose. In den folgenden Abschnitten werden, bezogen auf die KDD–Phasen, die Wissensgenerierungs- und Evaluierungsphasen des Projektes beschrieben. Abschließend wird ein Ausblick bzgl. des Einsatzes im täglichen Geschäft beim Anwender gegeben.

4.1 Implementierung der Anwendungen

Die in Kapitel 3 beschriebenen Methoden wurden in das kommerzielle KDD–Tool Clementine™ im Rahmen einer Kooperation des Herstellers ISL, der später von SPSS übernommen wurde, und des Auftraggebers implementiert. Die Auswahl des Tools war eine strategische Unternehmensentscheidung, die nicht aus der Anwendung heraus getroffen wurde. Das Tool wurde für die Anwendung um folgende Programmteile erweitert, die in den folgenden Abschnitten beschrieben werden.

4.1.1 Implementierung KDD im Analyseprozess

Zur Analyse von Schadensursachen, basierend auf der Kombination von Bauzuständen der Fahrzeuge, wurden folgende Teile der Datenverarbeitungsschritte standar-

disiert und automatisiert:

- Datenselektion in QUIS
- Datenvorverarbeitung und Wissensgenerierung

4.1.1.1 Datenselektion

Da das Qualitätsinformationssystem (QUIS) ein produktives System ist, war kein direkter Zugriff aus dem KDD–Tool auf die Daten möglich. Es mussten also die Auswerte- und Abfragemethoden von QUIS benutzt werden. Für die KDD–Anwendungen im PBP wurde die *Wahlfreie Selektion* des Systems verwendet.

Drei Datensätze mussten als Datenbasis für die Analyse von Schäden selektiert werden, da ein Joint dieser Daten im QUIS nicht möglich war.

1. Beanstandungsdaten: Hier wurde aus QUIS selektiert, mit welchen Beanstandungen, die zu untersuchende Fahrzeuggruppe in den Werkstätten war. Die Datenbank enthält je Beanstandung einen Datensatz.
2. Fahrzeugdaten: Mit dieser Selektion wurden die Basisdaten, z.B. Getriebeart, Motor, Karosserieform, Anzahl der Achsen usw. der Fahrzeuge selektiert.
3. Ausstattungsdaten: Neben den Fahrzeugdaten sind für die Anwendung die Ausstattungsmerkmale der Fahrzeuge von Interesse. Ein Fahrzeug hat im Durchschnitt 40 verschiedene Ausstattungsmerkmale. Nutzfahrzeuge haben sogar noch mehr. Pro Ausstattungsmerkmal und Fahrzeug ist ein Datensatz hinterlegt. Bei einer durchschnittlichen PKW–Jahresproduktion einer Baureihe von über 50.000 Fahrzeugen kommt man schnell auf über 200.000 Datensätze.

Für diese Selektionen wurden Abfragen mit einem vordefinierten Datensatzaufbau definiert, die für die einzelnen Anwendungen nur noch parametrisiert werden müssen. Die wichtigsten Parameter sind:

- der Fahrzeugtyp,
- Beanstandungszeitraum und
- Produktionszeitraum.

4.1.1.2 Vorverarbeitung und Wissensgenerierung

Für die Wissensentdeckung wurde das Tool INES [Borgelt u. a. 1998] in das KDD–Tool Clementine™ integriert.

Somit war es möglich ein Standardprogramm in Clementine™ mittels visueller Programmierung (Stream) zu erstellen. Innerhalb dieses Streams wurden die Beanstandungs- und Ausstattungsdaten transformiert, so dass je Fahrzeug nur ein Datensatz vorliegt. Mittels dieses Streams lassen sich unabhängig vom zeitlichen Auftreten der Beanstandungen Abhängigkeiten zwischen dem Schadensschwerpunkt und den anderen Beanstandungen untersuchen. Weiterhin ist eine Untersuchung nach Abhängigkeiten zwischen dem Schadensschwerpunkt und der Bauart so wie den Ausstattungsmerkmalen möglich.

4.1.2 Implementierung der Trendprognose

Für die ausgewählte Vorgehensweise zur Trendprognose wie sie in Abschnitt 3.9 beschrieben wurde, wurde Clementine™ um die folgenden Komponenten ergänzt:

- Datenselektionsprogramm zur Aufbereitung der QUIS Daten
- Clusterprogramm

4.1.2.1 Datenselektion für die Trendprognose

Um die kumulierten Beanstandungsquoten zu erhalten, müssen die Beanstandungsdaten aus QUIS transformiert werden. Aber auch für die Clusterberechnung müssen diese Daten vor der Anwendung der Verfahren transformiert werden. Basis hierfür ist eine Selektion aller Fahrzeuge mit ihren Beanstandungen und Werkstattdaten die benötigt werden. Hierzu wurde ein neuer Clementine™ Knoten implementiert, der im Folgenden als QUIS-Knoten bezeichnet wird. Der *QUIS Knoten* transformiert, gesteuert über Parameter, die Daten für die Clusterberechnung oder für die Trendprognose.

4.1.2.2 Clusterprogramm

Im Rahmen des CITRUS Projektes [Wirth u. a. 1997] wurde von Udo Grimmer und Joachim Mucha [Grimmer und Mucha 1998] Clementine™ um eine umfangreiche Clusteranalyse erweitert, mit der auch die speziellen Anforderungen und Methoden aus dem Ansatz der Trendprognose umsetzbar sind.

4.2 KDD als Analysehilfe – Anwendungsbeispiele

Die ersten beiden Analysen waren rückwirkende Bearbeitungen der Problemstellung, um die Anwendbarkeit der Methoden nachzuweisen. Die weiteren Pilotanwendungen sind jeweils aktuelle Fragestellungen aus der Praxis, die parallel zu den Analysen der Produktbetreuer durchgeführt wurden. Abschließend wird das Fazit aus der Anwendung der Experten aus dem Produktbewährungsbereich dargestellt.

4.2.1 Ringfederbruch bei schweren Nutzfahrzeugen

Die erste Anwendung zur Evaluierung des Ansatzes war ein Ringfederbruch bei schweren LKW's. Da es sich um einen Fall handelte, mit dem die Anwendbarkeit nachgewiesen werden sollte, war zum Zeitpunkt der Analyse mit INES die Lösung des Problems bekannt.

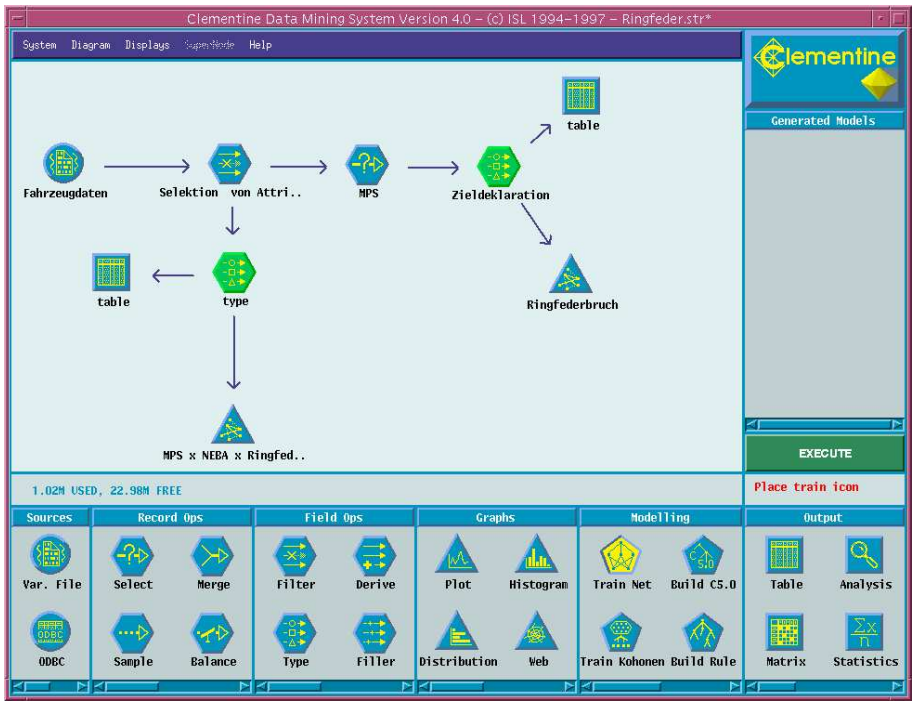

Abbildung 4.1: Abhängigkeitsanalyse in Clementine™

Von den Experten im Nutzfahrzeugbereich wurde eine Menge von Bauzuständen[1] definiert, die bei der Suche nach der Lösung berücksichtigt werden sollten. Die Kombination der Merkmale definiert den möglichen Lösungsraum. Weiterhin lag die Information vor, dass nur Fahrzeuge mit MPS[2] betroffen waren.

[1] Beschreibungsmerkmale für die Fahrzeuge, Ausstattungsmerkmale und Bauvarianten
[2] Mechanisch Pneumatische Schaltung

Die zu untersuchende Basis beinhaltete 6026 Fahrzeuge, die mit 25 Attributen/Bauzuständen beschrieben wurden.

Der Produktbetreuer kann in QUIS immer nur eine Hypothese verifizieren. Das heißt, der Erfolg seiner Arbeit hängt von seinem Fachwissen und einem Anteil Glück ab. Um nur alle Zweier–Kombinationen zu überprüfen, müsste der Produktbetreuer 325 Auswertungen gegen die Datenbank laufen lassen. Bei einer Auswertezeit von einer halben Stunde pro Anfrage kommt man schnell auf wirtschaftlich relevante Zeiten. In dem im Projekt erarbeiteten Ansatz werden alle Fahrzeuge mit ihren ausgewählten Bauzuständen in einer Auswertung gegen das Informationssystem selektiert. Dieser Auszug ist die Datenbasis für die Auswertungen in dem KDD–Tool. Für diese erste Anwendung wurde noch nicht die in Abschnitt 4.1.1 beschriebene Stream Implementierung verwendet.

In der Abbildung 4.1 ist der dazu in Clementine™ aufgebaute Stream dargestellt, während Abbildung 4.4 eine graphische Ergebnisdarstellung zeigt. Die Stärke der Linien beschreibt die Stärke der Abhängigkeit. Die textuelle Ausgabe mit den entsprechenden quantitativen Ergebnissen ist für den Anwender wesentlich informativer und lässt auch eine Beurteilung durch den Produktbetreuer zu.

```
prob(Ringfederbruch|NEBA)
  { Y: { Y: 20.91/  344, N: 79.09/ 1301 ( 1645) },
    N: { Y:  9.35/   49, N: 90.65/  475 (  524) } };
```

Abbildung 4.2: Ergebnis Ringfederbruch ohne MPS

Der Ringfederbruch tritt verstärkt bei Fahrzeugen mit einem Nebenantrieb (NEBA) auf. Das Ergebnis ist folgendermaßen zu lesen. Fahrzeuge mit einem Nebenantrieb werden zu 20.91% (absolut 344 Fahrzeuge) von 1645 Fahrzeugen, die diese Eigenschaft haben, mit einem Ringfederbruch beanstandet. Während nur 9.35% (49) Fahrzeuge beanstandet werden, die keinen Nebenantrieb haben. Hieraus folgt dass ca. 86% der beanstandeten Fahrzeuge einen Nebenantrieb haben (vgl. Abbildung 4.2).

```
prob(Ringfederbruch|MPS,NEBA) =
  { Y: { Y: { Y: 20.91/  344, N: 79.09/ 1301 ( 1645) },
         N: { Y:  9.35/   49, N: 90.65/  475 (  524) } },
    N: { Y: { Y:  0.30/    6, N: 99.70/ 2009 ( 2015) },
         N: { Y:  0.05/    1, N: 99.95/ 1841 ( 1842) } } };
```

Abbildung 4.3: Ergebnis Ringfederbruch mit MPS

Auch wenn man die Analyse nicht nur auf der Menge der MPS Fahrzeuge durchführt, findet das System den Zusammenhang von MPS und Nebenantrieb zu der Beanstandung. Dieser Nachweis war in der Pilotanwendung sehr wichtig, da dieses Ergebnis den Produktbetreuern das Potential des Verfahrens verdeutlichte (vgl. Abbildung 4.3).

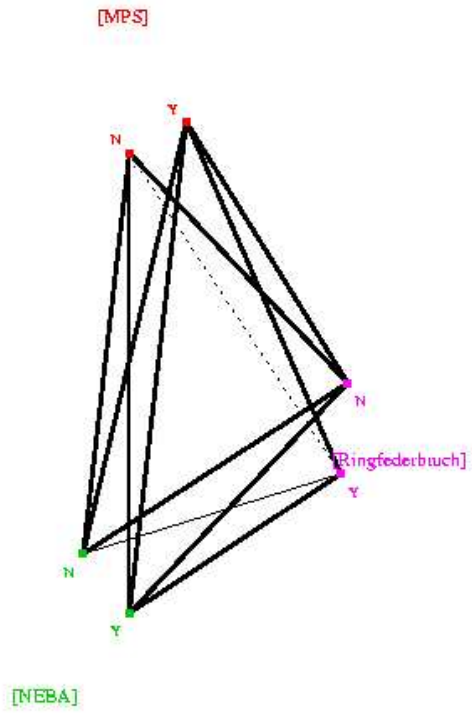

Abbildung 4.4: Graphische Darstellung der Abhängigkeiten zur Schadensursache

Ergebnisbewertung: Die gefundene Abhängigkeit ist die, die als schadensverursachender Zusammenhang von Seiten der Produktbetreuung ermittelt wurde. Mittels der Abhängigkeitsanalyse können Produktbetreuer unterstützt werden. Im Gegensatz zu der bisherigen Vorgehensweise, bei der der Produktbetreuer jede Hypothese über eine Abhängigkeit mittels einer Datenbankabfrage verifizieren muss, ist mit INES eine zielgerichtete Suche innerhalb der festgelegten Merkmalsmenge möglich. Der zeitliche Aufwand kann durch den Einsatz des Verfahrens erheblich reduziert werden. Für den hier untersuchten Schadensschwerpunkt wurden ohne das Verfahren mehrere Wochen benötigt, um auf die Lösung zu kommen.

4.2.2 Niveauregulierung

Die zweite Pilotanwendung wurde ebenfalls im Nutzfahrzeugbereich durchgeführt. Wie bei der ersten Anwendung sollte in dieser Pilotanwendung nochmals das Lö-

sungspotential des Ansatzes unter Beweis gestellt werden.

4.2.2.1 Problemstellung

Zum Zeitpunkt der Analyse gab es vermehrt Beanstandungen der Scheinwerfereinstellungen bei einer LKW-Baureihe. Ziel war es herauszufinden, ob von dieser Beanstandung eine bestimmte Gruppe von Fahrzeugen betroffen ist und wie sich diese Gruppe von Fahrzeugen über die Bauzustände beschreiben lässt. Für dieses Problem wurde keine Einschränkung der Bauzustände gemacht, so dass der gesamte Hypothesenraum (alle Kombinationen waren möglich) zu untersuchen war.

Im Hinblick auf eine weitere Nutzung der Methode im Unternehmensbereich wurde in Zusammenarbeit mit den Experten aus dem Nutzfahrzeugbereich eine *Standardauswertung* aufgebaut, wie in Abschnitt 4.1 beschrieben, die es den Anwendern aus dem Unternehmensbereich ermöglicht, Analysen mit INES selbständig durchzuführen.

4.2.2.2 Lösungen

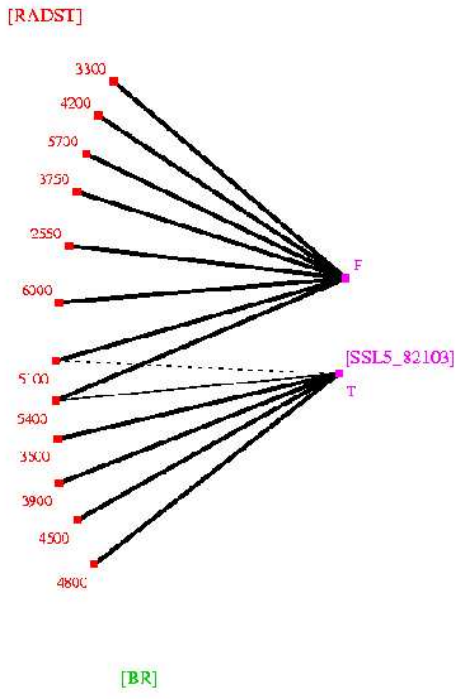

Abbildung 4.5: Abhängigkeitsanalyse in Clementine™

Das Ergebnis der Abhängigkeitsanalyse zeigte eine eindeutige Abhängigkeit zwischen den Baureihenvarianten und den Radständen zum Schadensschwerpunkt (vgl. Abbildung 4.6). Dieser Zusammenhang war der Gleiche, der von der Produktbetreuung

gefunden worden war und auf die Ursache, dass die Kalibrierung der Niveauregulierung bei Fahrzeugen mit bestimmten Radständen nicht stimmte, hinwies. In Abbildung 4.5 ist dieser Zusammenhang noch einmal graphisch dargestellt, hier wurde zur besseren Darstellung die Abhängigkeit zur Baureihe weggelassen.

```
prob(SSL5_82103|BR,RADST) =
  { BR1: { 3750: {                  F: 100.0/    3 (    3) },
           4200: {                  F: 100.0/   41 (   41) },
           4500: { T:  1.84/   39, F: 98.16/ 2076 ( 2115) },
           4800: { T:  1.90/   27, F: 98.10/ 1396 ( 1423) },
           5100: { T:  1.75/    3, F: 98.25/  168 (  171) },
           5400: { T:  1.15/   10, F: 98.85/  858 (  868) },
           5700: { T:  1.41/    1, F: 98.59/   70 (   71) },
           6000: {                  F: 100.0/   12 (   12) },
           6300: {                  F: 100.0/    1 (    1) } },
    BR2: { 3300: {                  F: 100.0/  271 (  271) },
           3600: { T:  2.33/    2, F: 97.67/   84 (   86) },
           3900: { T:  0.34/    1, F: 99.66/  294 (  295) },
           4200: {                  F: 100.0/    7 (    7) },
           4500: {                  F: 100.0/   18 (   18) },
           4800: {                  F: 100.0/   13 (   13) },
           5100: {                  F: 100.0/    2 (    2) } },
    BR3: { 3300: {                  F: 100.0/   12 (   12) },
           3600: {                  F: 100.0/    1 (    1) },
           4200: {                  F: 100.0/   17 (   17) },
           5100: {                  F: 100.0/    2 (    2) } },
    BR4: { 2550: {                  F: 100.0/    3 (    3) },
           3300: {                  F: 100.0/   54 (   54) },
           3600: { T:  3.60/  152, F: 96.40/ 4075 ( 4227) },
           3900: { T:  3.13/   37, F: 96.87/ 1146 ( 1183) } } };
```

Abbildung 4.6: Abhängigkeit Radstand zur Niveauregelierung

Ergebnisbewertung: Der ermittelte Zusammenhang war nicht das verursachende Schadensteil wie in der ersten Analyse der Nebenantrieb. Durch die erkannte Abhängigkeit zwischen den Radständen und dem Schaden, den die Experten in ihren Analysen auch entdeckt hatten, haben die Experten auf einen Fehler in der Kalibrierung der Niveauregulierung der Hinterachse geschlossen. Auch mit dieser zweiten Pilotanwendung konnte gezeigt werden, dass mittels der Abhängigkeitsanalyse die Produktbetreuer unterstützt werden können. Darüber hinaus stand nach dieser Anwendung der Abteilung eine Standardauswertung zur Verfügung, die die Produktbetreuer selbständig durchführen können.

Die in den folgenden Abschnitten beschriebenen Anwendungen wurden parallel zu den Aktivitäten der Produktbetreuer durchgeführt.

Bei diesen Anwendungen war nicht sichergestellt, dass die Ursachen oder der Hinweis auf die Ursache auch in den Daten enthalten ist. Hier gilt der Grundsatz „wo nichts ist, kann man auch nichts finden“. Die Information muss in den Daten stecken, wie in den ersten beiden Anwendungen.

4.2.2.3 Ölverbrauch Kleinbus/Transporter

Dieser Pilotanwendungsfall wurde parallel zu den Analysen der Produktbetreuer und der Konstruktion durchgeführt. Bei einem Kleinbus/Transporter trat bei einer Motorbaureihe ein nicht erklärbarer Ölverbrauch auf. Ein solcher Ölverbrauch ist kritisch, da ein Motorschaden die direkte Folge eines solchen Schadens sein kann.

In einer ersten Aufgabenstellung wurden von den Produktbetreuern aus dem Bereich Nutzfahrzeuge eine Menge von Ausstattungsmerkmalen benannt, die nach Abhängigkeiten zum Schaden untersucht werden sollten. Hier wurden keine Abhängigkeiten mit INES gefunden. Auch von den Produktbetreuern konnte kein Zusammenhang zwischen den Ausstattungsmerkmalen und dem Schadensbild bestimmt werden.

Als Schadensursache wurde die Qualität des Motorenöls von Seiten der Konstruktion und Produktbetreuung bestimmt. Ein solcher Zusammenhang ist nicht in den QUIS–Daten repräsentiert und kann folglich nicht gefunden werden.

Ein paar Monate später wurde das Thema ein zweites Mal als Pilotanwendung in das Projekt aufgenommen, da die Maßnahme nicht den gewünschten Erfolg hatte. Diesmal sollte die Analyse ohne Einschränkung auf die Ausstattungsmerkmale durchgeführt werden.

Von dem System wurde ein Zusammenhang zwischen der Abgasrückführung und dem Schaden gefunden. Die gefundene Abhängigkeit war mit einer der von den Experten untersuchten identisch.

Auch in dieser Anwendung konnten mit INES von den Experten nachvollziehbare Ergebnisse erzielt werden. Die eigentliche Ursache stand nach Ansicht der Experten, wie sich später herausstellte, nicht in Zusammenhang zu einer Kombination von Ausstattungsmerkmalen. Die Schadensursache wurde durch konstruktive Maßnahmen behoben.

4.2.3 Bremsenrubbeln Vorderachse

Dieser Fall war die erste Anwendung für den Bereich PKW. Bei einer PKW Baureihe traten vermehrt Beanstandungen über Rubbeln an der Vorderachse auf. Von Seiten der Produktbetreuer wurde eine Analyse in zwei Schritten gefordert.

Die erste Analyse wurde über die Fahrzeugmerkmale und eine zweite über die Ausstattungsmerkmale durchgeführt.

In der ersten Untersuchung wurde folgender Zusammenhang gefunden:

```
prob(Schadensfall_VA|Baumuster) =
  { PKW024: { T:  1.66/  102, F: 98.34/ 6041 ( 6143) },
    PKW025: {                 F: 100.0/   11 (   11) },
    PKW026: { T:  0.23/   12, F: 99.77/ 5127 ( 5139) },
    PKW028: { T:  2.75/  459, F: 97.25/16233 (16692) },
    PKW029: { T:  0.45/    7, F: 99.55/ 1548 ( 1555) },
    PKW078: { T:  0.51/  151, F: 99.49/29545 (29696) },
    PKW080: { T:  0.33/   28, F: 99.67/ 8358 ( 8386) },
    PKW082: {                 F: 100.0/    6 (    6) },
    PKW083: { T:  0.77/   24, F: 99.23/ 3101 ( 3125) },
    PKW085: { T:  0.15/    1, F: 99.85/  676 (  677) },
    PKW086: { T:  0.23/    6, F: 99.77/ 2563 ( 2569) },
    PKW089: { T:  0.54/    4, F: 99.46/  735 (  739) },
    PKW128: { T:  1.46/  215, F: 98.54/14482 (14697) },
    PKW188: { T:  0.95/   68, F: 99.05/ 7056 ( 7124) } };
/* log2(g)/N: -0.0849023 */
```

Abbildung 4.7: Ergebnis Vorderachse

Aus der gefundenen Abhängigkeit geht für den Experten hervor, dass die Schadenshäufigkeit bei Fahrzeugen mit größeren (betroffene Baumuster) Motoren höher ist, da das Baumuster im wesentlichen über die Motorvariante definiert ist. Dieser Zusammenhang bestärkte die Produktbetreuer in ihrer Vermutung, dass die Fahrweise einer bestimmten Kundengruppe die Ursache für die Höhe der Schadensquote sein könnte (siehe Abbildung 4.7).

In einem zweiten Schritt haben wir nach Zusammenhängen zwischen den Ausstattungsmerkmalen und den Beanstandungen gesucht. Insgesamt werden über 300 Sonderausstattungen verbaut. Ausstattungen, die weniger als 0,01% verbaut wurden, wurden nicht berücksichtigt.

```
prob(Schadensfall_VA|CODE_ID_957,CODE_ID_772) =
  { T: { T: { T:  9.40/  127, F: 90.60/ 1224 ( 1351) },
         F: {                 F: 100.0/    1 (    1) } },
    F: { T: { T:  2.10/   16, F: 97.90/  747 (  763) },
         F: { T:  0.99/  930, F: 99.01/93456 (94386) } } };
/* weighted information gain ratio: 0.340933 */

957 = AMG Technik Paket 772 = AMG Optik Paket
```

Abbildung 4.8: Abhängigkeit Beanstandung Vorderachse

Ergebnis der Analyse ist, dass AMG–Fahrzeuge wesentlich öfter beanstandet werden als Fahrzeuge ohne diese Ausstattung (vgl. Abbildung 4.8). Die Wahrscheinlichkeit ist zehnmal so hoch, dass ein solches Fahrzeug mit der Beanstandung

in die Werkstatt kommt. Dieser Zusammenhang bestätigte das Ergebnis und die Vermutung der Experten aus der ersten Analyse. Nimmt man die oben genannten Ausstattungsmerkmale aus der Betrachtung raus, so werden auch keine weiteren Abhängigkeiten gefunden. Das Ergebnis bestätigt die Vermutungen der Experten und schließt andere Zusammenhänge aus.

Ergebnisbewertung: Von den Experten war diese Aussage nachvollziehbar, da der Schaden auch durch ein Fahrverhalten dieser Kunden erklärt werden kann. Auch konnte von den Experten keine andere Ursache festgestellt werden. Die erkannten Abhängigkeiten waren somit eine der vermuteten Hypothesen und wären von den Experten ebenfalls untersucht worden. Mittels unserer Methode können wir den Produktbetreuer durch eine zielgerichtete Suche unterstützen, und auch nicht offensichtliche Zusammenhänge werden mit unserer Methode untersucht, da ein wesentlich größerer Hypothesenraum untersucht wird.

4.2.4 Hartes Bremspedal

Baureihen übergreifend gab es Beanstandungen der Bremsfunktion bei Diesel Fahrzeugen in ca. 30 Fällen. Da dieses Thema sicherheitsrelevant ist, wurde hierzu eine Task–Force eingesetzt, die nach einer Schadensursache sucht. Die beanstandeten Fahrzeuge stammen aus zwei Jahresproduktionen. Da die Grundgesamtmenge somit über zwei Jahresproduktionen geht und nur 30 Fahrzeuge beanstandet wurden, war es nicht ratsam das Verfahren über die Gesamtmenge der Fahrzeuge anzuwenden, da die Schadensquote einfach zu klein ist. Aus diesem Grund wurde eine Stichprobe zu den beanstandeten Fahrzeugen gezogen.

Die Stichprobe wurde ausgehend von den 30 bekannten Fahrzeugen gebildet. Zu diesen Fahrzeugen wurden alle weiteren Fahrzeuge selektiert, deren Fahrzeugidentifizierungsnummer (FIN) jeweils in dem Intervall [bek. FIN WAAAAA12345678900 - WAAAAA12345679999] lag.

```
prob(Schadensfall_T|CODE_ID_472,CODE_ID_581) =
  { T: { T: {                 F: 100.0/   20 (   20) },
         F: { T: 28.57/    2, F: 71.43/    5 (    7) } },
    F: { T: { T:  1.31/   15, F: 98.69/ 1128 ( 1143) },
         F: { T:  0.29/   12, F: 99.71/ 4117 ( 4129) } } };

CODE_ID_472 = ESP CODE_ID_581 = Klimatisierungsautomatik
```

Abbildung 4.9: Ergebnis hartes Bremspedal

Auf der Stichprobe konnte festgestellt werden, dass Fahrzeuge mit einer Klimatisierungssteuerung und ohne ESP eine wesentlich höhere Ausfallwahrscheinlichkeit

haben (vgl. Abbildung 4.9).

Ergebnisbewertung: Das Ergebnis wurde von den Experten und Auftraggebern der Pilotanwendung als Zufall interpretiert, da zwar die prozentualen Differenzen groß sind aber die absoluten Zahlen ähnlich groß sind. Leider konnte der Experte nicht überzeugt werden, den gefundenen Zusammenhang genauer zu untersuchen. Die oben beschriebene Anwendung ist kein typisches Problem aus der Produktbetreuung. Diese Anwendung stellt ein Extremproblem dar, für das in der Produktbetreuung auch keine Lösungsmöglichkeit gefunden wurde.

4.2.5 Fazit aus den Anwendungen

Der Nutzfahrzeugbereich beurteilte in einer Stellungnahme die Data Mining Aktivitäten zu den Pilotanwendungen mit folgenden Aussagen.

Zielsetzung der Pilotanwendungen war die Klärung der folgenden Fragen:

- Werden bekannte Zusammenhänge bzw. Einflüsse erkannt?
- Gibt es weitere, noch unbekannte Zusammenhänge?
- Wie hoch ist der zeitliche Aufwand für die Datenaufbereitung und die Datenanalyse?

Nach den bisherigen Beurteilungen aller Beteiligten wurden die bekannten Zusammenhänge auch von INES klar erkannt. Neue und unbekannte Zusammenhänge zu erkennen, ist sehr schwierig. Hierzu muss man berücksichtigen, dass die Experten natürlich über ein beträchtliches Wissen über ihren Aufgabenbereich besitzen. Die Frage, die sich eigentlich hier stellt, ist die Frage nach dem Potential für neue unbekannte Abhängigkeiten. Die erkannten Abhängigkeiten waren im allgemeinen von den Experten nachvollziehbar und erklärbar, aber sicherlich nicht immer in ihrer Ausprägung bekannt.

Neben dieser methodischen Unterstützung ergibt sich ein weiterer wirtschaftlicher Nutzen durch den Einsatz dieser Analysemethode. Ein breiterer Einsatz von INES könnte auch die Zahl der QUIS–Auswertungen reduzieren und zwar in zweifacher Weise.

1. Um Hypothesen bezüglich Abhängigkeiten zwischen Bauzustand und Schaden zu verifizieren, müssen immer neue Auswertungen angestartet werden. Mit unserem Ansatz selektiert man einmal die Daten und analysiert lokal das Problem.
2. Wenn man die Fahrzeugdaten und Ausstattungsmerkmale zentral aus QUIS selektiert, kann man weitere Auswertungen sparen.

Ein weiterer Vorteil der entwickelten Methode ist die Zeitersparnis innerhalb des Analyseprozesses. Bei einer *konventionellen* Analyse müssen die von den Produktbetreuern aufgestellten Hypothesen einzeln durch Datenbankabfragen verifiziert werden. Im Gegensatz dazu kann man mit dem hier vorgestellten Ansatz innerhalb des Systems bei der Suche nach Abhängigkeiten unterstützt werden und erhält Abhängigkeiten je nach Komplexität in 1 bis 30 Minuten.

Zusammenfassend kann gesagt werden, dass mit der entwickelten Methode für den Unternehmensbereich eine neue Analysetechnik zur Verfügung gestellt wird, die man im Nutzfahrzeugbereich als kurzfristig einsetzbar einstuft. Der Prototyp steht den Bereichen zur Verfügung.

4.3 Anwendungsfall: Elektrikschäden

Neben den zuvor beschriebenen Analysen wurden die Experten an weiteren Problemen aktiv unterstützt. Das folgende Problem gehört nicht zu den in den vorherigen Abschnitten beschriebenen Problemtypen. Das zu untersuchende Problem war wesentlich schwieriger zu untersuchen, da es sich nicht um eine einzelne Ursache, sondern eine Kombination von Einflüssen für eine Gruppe von Schäden handelt.

In einer LKW Baureihe traten überproportional viele Beanstandungen an den elektrischen Leitungen auf. Die Ursache für die Beanstandungen war nicht bekannt. Mit Hilfe der bisher eingesetzten Verfahren und anderen, sollte die Analyse der Experten unterstützt werden, nachdem diese das Problem schon mehrere Monate untersucht hatten. Nicht alle elektrischen Leitungen können einem elektrischen System zu geordnet werden, so dass sich die gemeine elektrische Kabelbeanstandung aus einer Vielzahl von Ursachen zusammensetzen kann.

Prinzipiell kann sich eine Schadensquote aus drei Komponenten zusammensetzen:

1. dem Fehler selbst,
2. Falschverschlüsselungen,
3. Kundenwunsch, Arbeitswerte-Beschaffung.

Die ersten beiden Punkte wurden schon in den vorherigen Abschnitten behandelt und erklärt. Der dritte Punkt beinhaltet das Controlling der Werkstätten.

Von Seiten der Produktbetreuer waren folgende Fragestellungen von besonderem Interesse:

- Gibt es Abhängigkeiten von Leitungsbeanstandungen zu anderen elektrischen Systemen?
- Gibt es eine Abhängigkeit vom Schaden zu einem Baumuster?
- Tritt der Schaden in einzelnen Werkstätten überproportional häufig auf?

- Welche Beanstandungen treten in Kombination mit anderen auf?

Ein Verfahren, das solche Zusammenhänge der letzten Frage finden kann, auch Assoziationsregeln genannt, ist **apriori** [Agrawal und Srikant 1994], das ebenfalls in dem von uns verwendeten KDD–Tool zur Verfügung steht.

Assoziationsregeln: $X \Rightarrow Y$

Aus X folgt Y mit einer Sicherheit Z. Ein Beispiel

Temperaturgeber $\Rightarrow$ Leitungssatz Retarder (1233:5.9%,0.143)

Die Grundgesamtheit bei dieser Untersuchung besteht aus ca. 20800 Fahrzeugen. Der Temperaturgeber wurde 438 mal beanstandet, während die Regel für 176 Fahrzeuge $(1233 * 0,143)$ gilt. Aus diesem Zusammenhang ergibt sich, dass ca. **40%** aller Fahrzeuge, bei denen der Temperaturgeber beanstandet wurde, auch der Leitungssatz des Retarder, beanstandet wurden.

Motiviert durch diese Ergebnisse wurden weiterführende Untersuchungen zu allen oben aufgeführten Fragestellungen durchgeführt. Abweichend von den in Kapitel 3.8 ausgewählten Verfahren wurden für einzelne Fragestellungen andere Varianten der Verfahren eingesetzt. An den entsprechenden Stellen wird darauf noch im Detail eingegangen.

Die Ergebnisse der Untersuchung wurden auch dem Lenkungsausschuss der zuständigen *Task Force*, dem fachlichen Kontrollgremium, vorgestellt, sowie auf der ICSC 1999 veröffentlicht [Lindner und Hipp 1999].

4.3.1 Datenverständnis und Transformation

Mit den verantwortlichen Produktbetreuern wurde eine geeignete Datenbasis der zu untersuchenden Fahrzeuge bestimmt. Ausgewählt wurden die Stückzahlträger der Baureihen die ca. 80% (14864 Fahrzeuge) der Jahresproduktion (18580 Fahrzeuge) ausmachen und somit von Exoten und Spezialfahrzeugen bereinigt ist. Von diesen Fahrzeugen wurden 11007 Fahrzeuge innerhalb des ersten Jahres beanstandet. Von diesen Fahrzeugen hatten 7945 Fahrzeuge mindestens eine Beanstandung an einem elektrischen System oder einem Leitungssatz, der keinem der Systeme zugeordnet werden kann. Mit der Einführung des Modells hatte man eigentlich eine Reduzierung der Schadensquoten erwartet.

Zum Verständnis der Daten ist in Abbildung 4.11 die Verteilung der Schäden dargestellt. Leitungssatzsysteme sind Beanstandungen an elektrischen Systemen die nicht im Detail untersucht wurden, während elektrische Leitungen allgemein Beanstandungen an Leitungssätzen sind, die keinem elektrischen System zugeordnet werden konnten. Explizit sollten in der Untersuchung die in Abbildung 4.11 aufgeführten Systeme betrachtet werden. Die Auswahl dieser Systeme wurde von den Produktbetreuern und Experten vorgenommen.

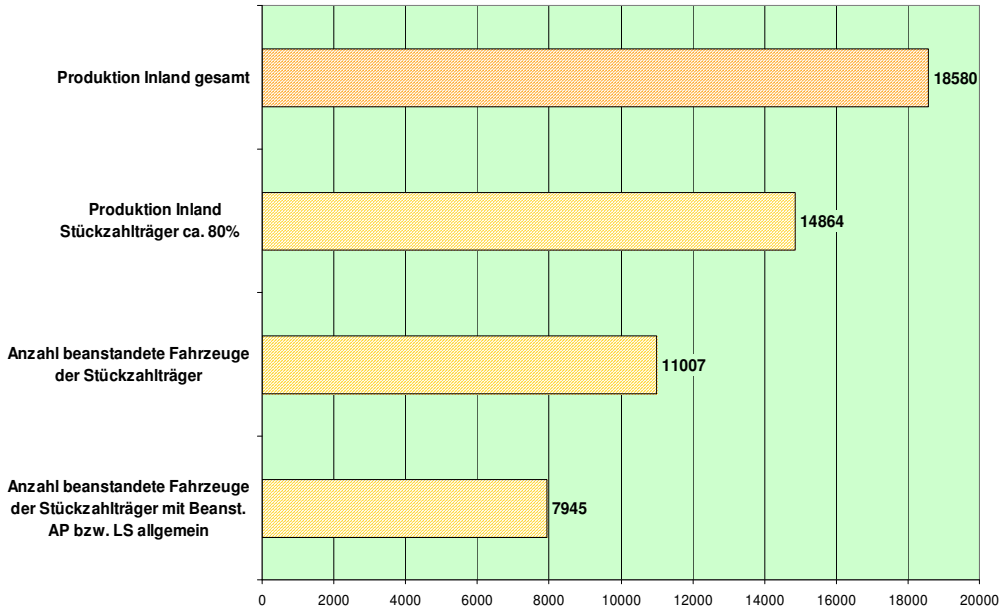

Abbildung 4.10: Datenbasis Kabelschäden

Auch hier, wie in allen anderen Anwendungen zum PBP, ist QUIS die Datenbasis. Im ersten Schritt wurden die Daten aus verschiedenen Datensichten selektiert. Aber nicht alle gewünschten Informationen waren für die gestellte Aufgabe explizit verfügbar. Weitere Attribute wurden aus den selektierten Daten abgeleitet, z. B. „Anzahl der Tage vom Tag der ersten Beanstandung bis zum zweiten Werkstattaufenthalt“. Die selektierten und abgeleiteten Attribute wurden in einer Tabelle zusammengefasst.

Die Aufgabenstellung lässt sich unter der generellen Data Mining–Aufgabe, der Abhängigkeitsanalyse, zusammenfassen. Aus der Vielzahl der verfügbaren Verfahren wählten wir INES, apriori [Agrawal und Srikant 1994] und ein Verfahren, das sequentielle Muster (sequential patterns) nach [Srikant und Agrawal 1995] in einer Implementierung von [Klenk 1997] erkennt, aus. Das Problem bei diesen Verfahren ist, dass jedes der Verfahren ein eigenes Eingabeformat für die Daten braucht, so dass viel Aufwand in die Datenvorverarbeitung investiert werden musste.

4.3.2 Wissensentdeckung

Probabilistische Netze (INES) (siehe Abschnitt 3.7.1) wurden für die Frage nach Abhängigkeiten zwischen Beanstandungen und Werkstätten eingesetzt. Hier ist der zeitliche Aspekt nicht relevant.

Das Auffällige an den gefundenen Zusammenhängen ist, dass es Werkstätten gibt, bei bei denen bis zu drei von vier Fahrzeugen an Elektrikschaden repariert worden. Die

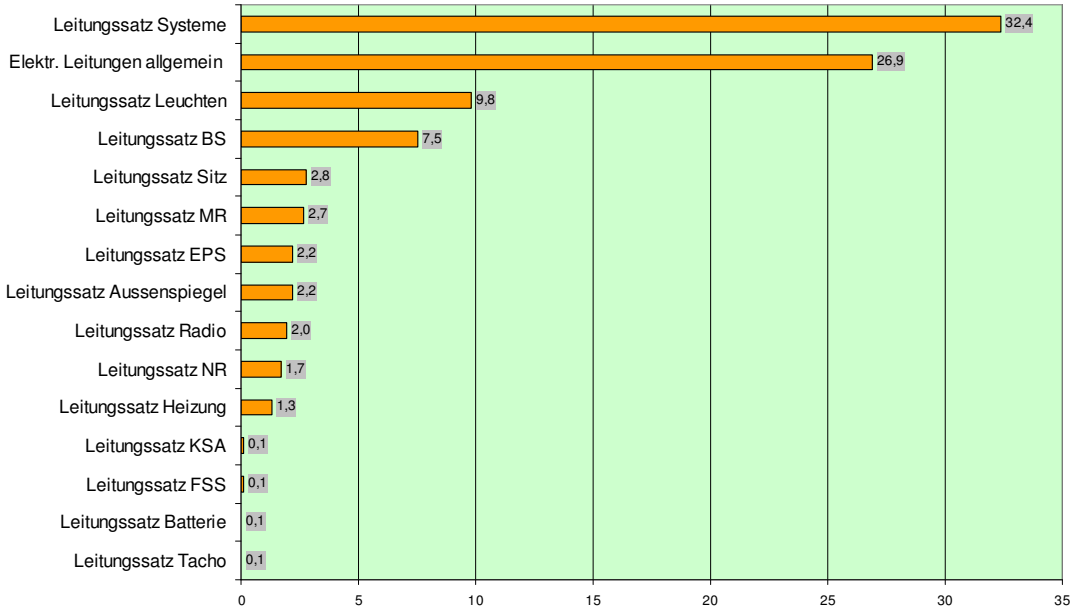

Abbildung 4.11: Schadensverteilung Elektrik

`prob(LS|Reparaturbetrieb_23400,Reparaturbetrieb_24110)`= T: F: Y: 69.33/ 52, F: 30.67/ 23 (75) , F: T: Y: 75.00/ 33, F: 25.00/ 11 (44) , F: Y: 35.52/ 3792, F: 64.48/ 6884 (10676) ;

Abbildung 4.12: Werkstattabhängigkeiten

durchschnittliche Beanstandung dürfte aber nur bei $35,52\%$ liegen (vgl. Abbildung 4.12).

Eine Schwäche von booleschen Assoziationsregeln ist, dass sie beschränkt sind auf die Entdeckung von Abhängigkeiten zwischen Elementen einer Generalisierungsebene [Srikant und Agrawal 1995]. Speziell in der PBP–Anwendung sind Zusammenhänge auf einer höheren Generalisierungsebene interessant. So können im PBP nicht nur Abhängigkeiten zwischen einzelnen Teilen, sondern auch zu Funktionsgruppen relevant sein oder gar der gesuchte Hinweis zur Schadensursache sein. Zusätzlich ist im PBP der zeitliche Aspekt relevant, impliziert eine Beanstandung einen Folgefehler? Dieser zeitliche Zusammenhang ist in vielen Anwendungsfragen relevant. Solche sequentiellen Muster (*seqential pattern*) sind eine Erweiterung der Assoziationsregeln. Die Implementierung eGSP (*extended generalised sequential patterns*) [Klenk 1997] unterstützt beide Eigenschaften, die Generalisierung durch Taxonomien und die Erkennung von sequentiellen Mustern. Dieses Verfahren wurde für die Fragen nach Abhängigkeiten zu anderen Systemen oder den dazugehörigen Systemen eingesetzt. Von besonderem Interesse waren hierbei folgende Fragen :

- Um wieviel Prozent reduziert sich die Beanstandung an den Leitungssätzen,

wenn der Fehler an einem System gegen 0 reduziert werden kann?

- Welche Systeme und Leitungssätze werden bei einem Werkstattaufenthalt beanstandet?

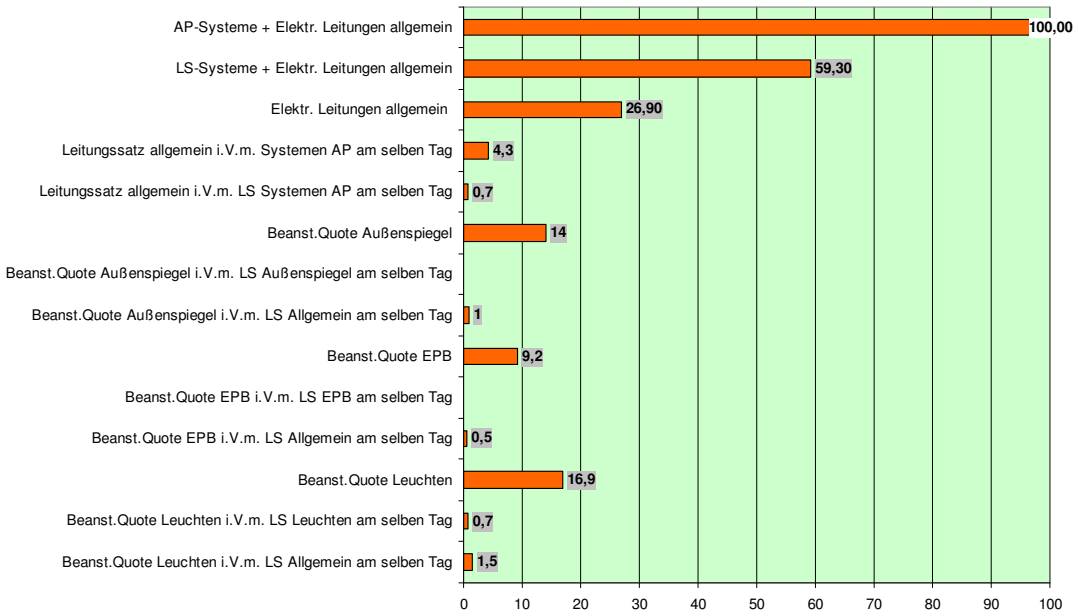

Abbildung 4.13: Abhängigkeiten Leitungssätze und Systeme

59, 3% aller Fahrzeuge, bei denen elektrische Systeme oder Leitungssätze beanstandet wurden, wurden an den Leitungssätzen der Systeme (LS-Systeme) oder an den allgemeinen Leitungssätzen beanstandet. Bei nur 4, 3% wurden der allgemeine Leitungssatz und die ausgewählten elektrischen Systeme (AP–Systeme[3]) am selben Tag beanstandet.

Aus diesem Grunde wurden Beanstandungen, die innerhalb von zwei Wochen an einem Fahrzeug auftraten, als ein Werkstattaufenthalt bewertet. Hintergrund ist die Annahme, dass wenn ein Fahrzeug nach so kurzer Zeit wieder in der Werkstatt steht, der ursprüngliche Schaden nicht behoben wurde.

In den Abbildungen 4.14 und 4.15 sind zwei Ergebnisse exemplarisch dargestellt. Abbildung 4.14 zeigt, welche Leitungssätze zusammen mit dem Aussenspiegel beanstandet wurden. Bemerkenswert war das Ergebnis, dass der Leitungssatz für die Spiegel ohne die neu getroffene Annahme gar nicht beanstandet bzw. ausgewechselt wurde. Erst bei einem weiteren Werkstattaufenthalt wurde der Leitungssatz für den

[3]AP = Arbeitspakete (elektrische Systeme) die von der Task Force untersucht werden. LS = Leitungssatz, EPB = elektrische pneumatische Bremse

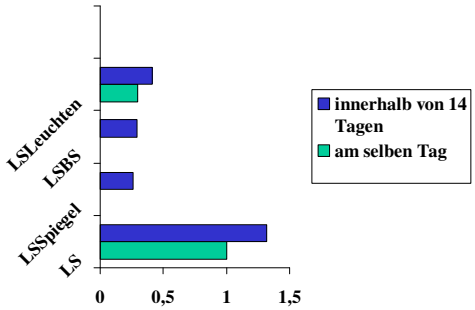

Abbildung 4.14: Abhängigkeiten Aussenspiegel zu Leitungssätzen

Spiegel beanstandet. Könnte man die Schadensquote für den Aussenspiegel auf 0 reduzieren, so würde man alleine bei den allgemeinen Leitungssätzen eine Reduzierung um $5,4\%$ erreichen.

Ein ähnliches Bild zeigte sich bei den Bremssystemen. Auch hier wird von den Werkstätten erst ein allgemeiner Leitungsschaden vercodet. Erst wenn das Fahrzeug wiederholt in die Werkstatt kommt, wird auch der zum beanstandeten System gehörende Leitungssatz beanstandet.

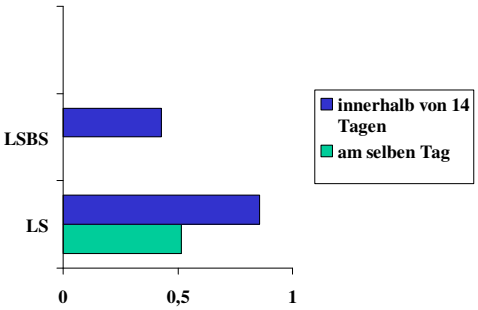

Abbildung 4.15: Abhängigkeiten vom Bremssystem zu Leitungssätzen

Diese Ergebnisse machten deutlich, dass das Problem sehr vielschichtig ist. Eine klare Ursache konnte nicht festgestellt werden, das war aber auch nicht Ziel dieser Untersuchungen. Die gefundenen Ergebnisse gaben klare Antworten auf die gestellten Fragen und bestätigten, dass es für die Beanstandungen an den Leitungssätzen viele Ursachen gibt.

4.4 Ergebnisse zur Trendprognose

In den folgenden Abschnitten werden einzelne Prognosen zu speziellen Themen, die für die Experten aus gegebenen Anlässen von besonderem Interesse waren, vorgestellt. Die Gesamtgüte der Modelle wurde schon im Abschnitt 3.9 beschrieben.

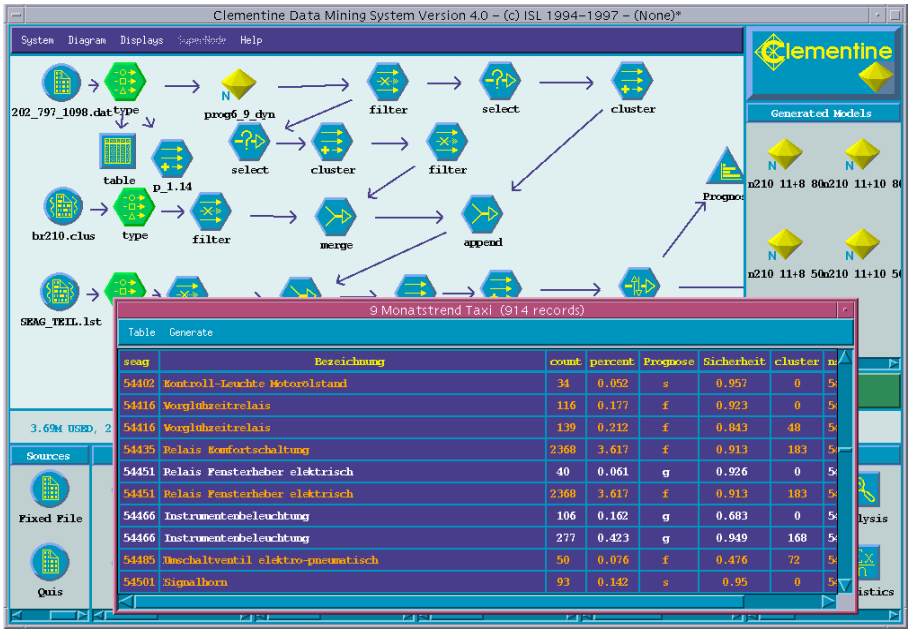

Abbildung 4.16: Implementierung in Clementine™

4.4.1 Fensterheber PKW

Der Fensterheberknopf bei einer PKW–Baureihe bricht. Im März wurde hierzu eine Maßnahme durchgeführt. Der zuständige Produktbetreuer ist nun natürlich daran interessiert, so früh wie möglich eine Aussage über die Wirkung der Maßnahme zu bekommen. Anfang Juli wurde hierzu eine erste Trendprognose für die Drei– und Sechs–Monatskurve der Mai bzw. der Februar Produktion gegeben. Die Sechs–Monatsprognose für Februar ist in dieser Anwendung sicher nicht so interessant, da die Maßnahme erst im März eingeführt wurde. Für die Drei–Monatskurve haben wir einen gleichbleibenden Trend prognostiziert. Im darauf folgenden Monat haben wir noch einmal eine Prognose der Sechs–Monatskurve für den Produktionsmonat März durchgeführt. Der prognostizierte Trend für den Schaden ist gleichbleibend mit einer Sicherheit von 86%. Das heißt, die rel. Beanstandungsquote für März wird sich auf einem ähnlichen Niveau bewegen, wie die Quote von Januar. Der von uns berechnete Trend bezieht sich auf die Steigung zwischen dem zuletzt bekannten Wert und dem zu prognostizierenden Monat. Bezogen auf die Quote der beanstandeten Fahrzeuge pro Produktionsmonat ist die oben gemachte Aussage übertragbar. Abbildung 4.17 zeigt den Verlauf der Beanstandungen.

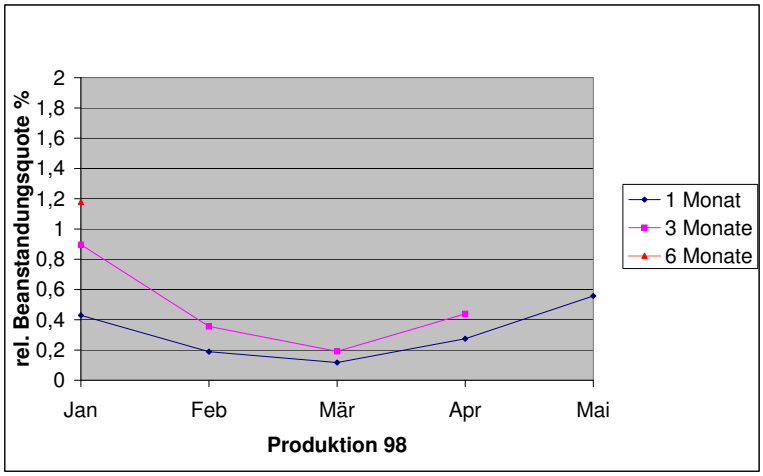

Abbildung 4.17: Beanstandungen Fensterheber

In Abbildung 4.18 sind die tatsächlichen und prognostizierten Monatskurven gegenübergestellt. Für den Fensterheber lassen sich sehr gut Quoten vorhersagen. Wie im Abschnitt 3.9 ausgeführt, geben wir in der Implementierung die Quote und eine Trendklassifikation als Ergebnis aus. Bei solch frühzeitigen Prognosen mit Beanstandungen von real unter 10 Fahrzeugen/Teilen, wie sie in dieser Problemstellung gefordert ist, kann es schnell zu großen Abweichungen kommen. Aus diesem Grund hatte man sich für die Vorhersage von Trends entschlossen. In Abbildung 4.16 ist der implementierte Stream (Programm) mit einer Ausgabe der Prognose dargestellt. Auffällige Trends werden rot dargestellt.

4.4.2 Undichtes Hinterachsenmittelstück

Eine ähnliche Aufgabe wurde für die Undichtheit des Hinterachsenmittelstücks einer PKW–Baureihe gestellt. Das Schadensverhalten bei dieser Beanstandung wird von den Experten wie folgt unterschieden:

1. Frühausfälle, die durch Montagefehler entstehen.
2. Spätausfälle, die aufgrund von Materialfehlern auftreten.

Die zu beobachtende Maßnahme wurde Ende März eingeführt und zielte auf die Frühausfälle. Es sollte nun mit der Trendprognose beobachtet werden, ob die Maßnahme greift. Aus dieser Anforderung heraus wurde der Ansatz der Trendprognose auf eine Prognose der Sechs–Monatskurve auf Basis der Drei–Monatswerte transferiert. Die Prognosen zur Drei–Monatskurve tendieren gegen Null.

Die Trendprognose für Schadensschwerpunkte, einzelne Teile (5-stelliger Schadensschlüssel) oder Gruppen von Teilen, basiert auf den relativen Beanstandungsquoten in dem Betrachtungszeitraum. In der Abbildung 4.20 sind die bisher bekannten rel. Beanstandungsquoten für das Schadensteil und Trendprognosen für die Sechs– und

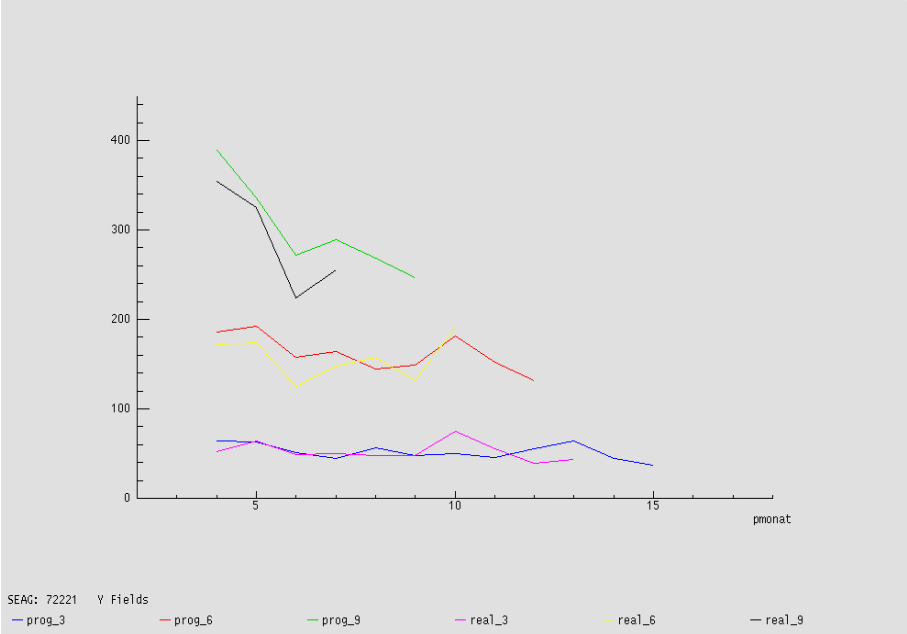

Abbildung 4.18: Prognose Fensterheber

Neun–Monatskurven dargestellt. Da der Trend nicht im Kurvenverlauf darstellbar ist, ist dieser durch Pfeile kenntlich gemacht. Der prognostizierte Trend für die März–Produktion sagt für die Beanstandungsquoten nach 6 und 9 Monaten einen fallenden Trend vorher. Die Beanstandungsquoten bleiben danach auf dem Niveau. Mit dieser Aussage konnten die Experten drei Monate früher die Wirkung der Maßnahme beurteilen. Dieses bedeutet einen hohen wirtschaftlichen Vorteil. Hätte die Maßnahme nicht gewirkt oder gar das Gegenteil bewirkt, so würden bis zu 3 Monate länger nach den Vorgaben die Fahrzeuge produziert.

4.4.3 Bruch der Riemenscheibe bei AMG Fahrzeugen

Eine weitere Anfrage zur Bestimmung einer Trendaussage kam von den Experten aus dem Bereich Produktbetreuung Motoren. Bei Fahrzeugen zweier Baumuster kommt es zu einem Speichenbruch in der Riemenscheibe. Die Experten wünschten eine Trendprognose für die Sechs–Monatskurve zu bekommen.

Der prognostizierte Trend für den Schaden ist **gleichbleibend** mit einer Sicherheit von 86%. Das heißt, die relative Beanstandungsquote für den Produktionsmonat März wird sich auf einem ähnlichen Niveau bewegen, wie die Quote von Januar. Der von uns berechnete Trend bezieht sich auf die Steigung zwischen den zuletzt bekannten Werten und dem zu prognostizierenden Monat. Bezogen auf die Quote der beanstandeten Fahrzeuge pro Produktionsmonat ist die oben gemachte Aussage übertragbar.

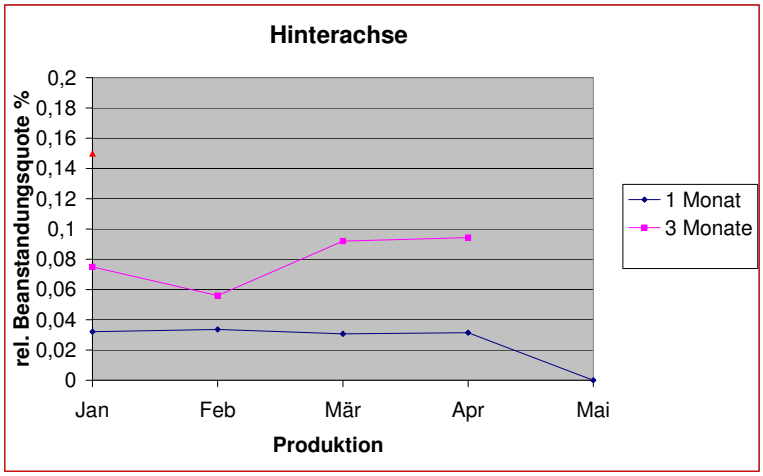

Abbildung 4.19: Schadensverlauf Hinterachsenmittelstück

Die oben getroffene Aussage ist mit Vorsicht zu behandeln, da die Datengrundlage für den Schaden sehr gering ist. Im Schnitt traten ein bis zwei Schäden pro Produktionsmonat auf. Aufgrund des bisherigen Schadensverlaufs ist die obige Aussage gemacht worden. Weiterhin ist durch die Reduzierung der Grundgesamtheit auf Fahrzeuge mit diesen zwei Baumustern (ca. 4000 Fahrzeuge[4]) der relative Anteil am Schadensaufkommen höher, was auch eine leichte Verzerrung zur Folge haben kann.

4.4.4 Fazit aus den Anwendungen

Die Experimente, die ja über reale Daten durchgeführt wurden, zeigen gute Erkennungsleistungen der Trends bis hin zu den Beanstandungsquoten. Die Leistungsmerkmale aus den Untersuchungen bei der Methodenauswahl konnten in den weiteren Untersuchungen bestätigt werden. Aber auch die neuen Analysemethoden sind auf positive Resonanz besonders im Nutzfahrzeugbereich gestoßen.

4.5 Übergabe in die Linie

Die in den vorherigen Abschnitten gezeigten Ergebnisse konnten bis zu einem gewissen Grad in den Analyseprozess des PBP integriert werden. Andere Verfahren wurden prototypisch eingesetzt um das Potential dieser Verfahren für den Geschäftsprozess aufzuzeigen. Die Trendprognose wurde ebenso in einem Prototyp realisiert. Der Projektauftrag beinhaltete keine Überführung in einen Produktionsbetrieb, d.h. keine Integration in die Rechenzentren oder genutzte Software. Trotzdem und vor allem auf Grund der positiven Ergebnisse haben einzelne Gruppen in der Linie die Prototypen übernommen.

[4]Gesamte Jahresproduktion

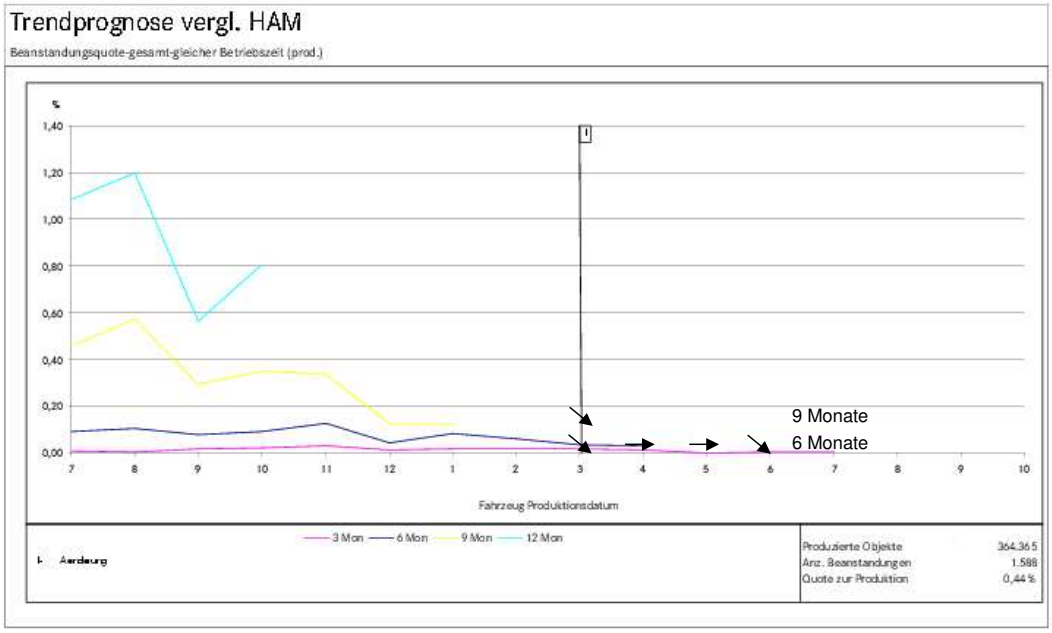

Abbildung 4.20: Trendprognose Hinterachsenmittelstück

4.5.1 Analysemethoden

Der Prototyp mit den Standardauswertungen aus QUIS wurde dem Nutzfahrzeugbereich übergeben. Hierzu wurden an einem Arbeitsplatz Clementine™ und alle entwickelten Knoten und Programme installiert. Darüber hinaus nahm fürs erste ein Mitarbeiter an einer Schulung teil und wurde anschließend noch in der Handhabung des Prototypen geschult.

4.5.2 Trendprognose

Die Trendprognose wurde vom PKW Bereich übernommen. Auch hier wurde ein Arbeitsplatz komplett ausgestattet und ein Mitarbeiter über mehrere Wochen in der Handhabung unterstützend betreut. Darüber hinaus sollten die Trendprognosen in den vierteljährlich erstellten Quartalsberichten berücksichtigt werden.

Innerhalb des Quartalsberichtes werden für die einzelnen Themenschwerpunkte Trendaussagen gemacht, die von allen beteiligten Experten an einem runden Tisch getroffen werden. Ziel ist es, mit Hilfe der Trendprognose die Trendaussagen der Experten zu stützen oder die Trendaussage als Entscheidungshilfe anzubieten.

Teil II

Auswahl von Data Mining Verfahren

Kapitel 5

Benutzerunterstützung im KDD–Prozess

5.1 Konzept der Benutzerunterstützung im KDD

Aus den in den Kapiteln 3 und 4 beschriebenen Anwendungen wurde deutlich, dass KDD kein trivialer Prozess ist. Der Erfolg des Projektes oder der Analyse hängt von den Fähigkeiten und den Erfahrungen des Miner's (Anwenders) ab. Aus diesem Grund besteht ein großer Bedarf an Unterstützung im KDD–Prozess. Nur eine gute Qualität in der Unterstützung im KDD–Prozess kann der Garant für ein gutes Ergebnis sein. In diesem Kapitel werden die einzelnen kritischen Entscheidungspunkte in den KDD–Phasen identifiziert und mögliche Ansätze zur Benutzerunterstützung beschrieben. Die Benutzerunterstützung für einen der kritischen Entscheidungspunkte, der Algorithmenauswahl ist dann Schwerpunkt der vorliegenden Arbeit.

Einen ersten Ansatz der Benutzerunterstützung im KDD–Prozess wurde von Engels, Lindner und Studer 1997 auf der dritten Internationalen Konferenz für Knowledge Discovery und Data Mining vorgestellt [Engels u. a. 1997]. Basis dieses Ansatzes ist eine top down– und bottom up–Betrachtung im Benutzerunterstützungskonzept, die innerhalb des UGM (User Guidance Module) Ansatzes vorgestellt wurde.

Die top down–Komponente des UGM–Ansatzes beinhaltet eine Aufgabenzerlegung in die einzelnen Phasen und generischen Aufgaben, wie sie auch in CRIPS vorgestellt wurde. In diesem Abschnitt werden die Ideen des UGM's vorgestellt, die Robert Engels in seiner Dissertation [Engels 1999] zur top down–Komponente erläutert. Die vorliegende Arbeit hat nun den Schwerpunkt, den KDD–Prozess bottom up zu unterstützen. In diesem Teil II wird nun eine Lösung für eine bottom up–Unterstützung für die Algorithmenselektion im KDD–Prozess vorgestellt. Die Umsetzung der Lösungsidee wird in Teil III dieser Arbeit beschrieben.

5.2 User Guidance Module Ansatz

Der User Guidance Module (UGM) Ansatz ist eigentlich ein genereller Ansatz zur Benutzerunterstützung. Der UGM Ansatz spezifiziert spezielle Arten von Aufgaben durch die Definition eines Prozessmodells für diese Art von Aufgabe. Dieses Prozessmodell kann dann durch eine Menge von einfachen Aufgaben (*simple tasks*) und den dazu gehörigen Techniken instanziiert werden.

Eine solche Benutzerunterstützung hat zum Ziel, bei der Unterstützung im KDD–Prozess den Anwender in die Lage zu versetzen, sein Problem und sein Ziel in dem System zu spezifizieren. Hierzu müssen alle Eigenschaften des Anwenderproblems erfasst werden. Dieses beinhaltet die funktionalen und nicht funktionalen Anforderungen. Diese müssen von dem Anwender durch geeignete Dialoge erfasst werden. Einige nicht funktionale Anforderungen lassen sich auch automatisch aus den Daten herleiten. Ziel ist es in dieser Phase, eine Problembeschreibung zu erhalten. Im nächsten Schritt wird auf Basis der Problembeschreibung versucht, das Problem mittels Techniken der Aufgabenzerlegung in einzelne Aufgaben zu unterteilen, für die Teillösungen oder Lösungen für einfache Aufgaben (*simple task*) gefunden werden können.

Vorherige Erfahrungen, wiederverwendbare Teillösungen, einzelne Aufgaben und eine Beschreibung der Daten (Daten Charakteristik) sind alles Elemente die nach dem UGM–Ansatz gespeichert werden sollten. Hierzu sind verschiedene Repositories vorgesehen, mit denen dann eine Verknüpfung zu den Aufgabenbeschreibungen besteht.

Insgesamt sieht der Ansatz vier Repositories vor:

1. Repository mit Gesamtprozesslösungen und wiederverwendbaren Prozessteilen (*Reusable Process Units* RPU)
2. Repository mit einzelnen Aufgaben
3. Repository mit Beschreibungen zu möglichen Verfahren
4. Repository mit einer Beschreibung der Daten zu den Problemen

5.2.1 Repository mit Gesamtprozesslösungen und RPU's

Gesamtprozesslösungen beinhalten eine Beschreibung des gesamten Prozesses und des Projektes, von der initialen Projektbeschreibung bis zur endgültigen Lösung und den erzielten Ergebnissen. Im Einzelnen enthält ein solches Repository:

- eine Problembeschreibung,
- eine Aufgabenzerlegung, die zeigt wie die ursprüngliche Aufgabe heruntergebrochen wurde bis zu einzelnen Aufgaben (*simple Tasks*),

- die Reihenfolge der einzelnen Aufgaben,
- die Beschreibung der Daten
- und die Projektergebnisse
- sowie Anmerkungen zum Vorgehen.

Das Gleiche gilt für die wiederverwendbaren Teilprozesse (RPU's) mit folgenden Unterschieden:

- RPU's repräsentieren immer nur eine Teillösung
- RPU's werden durch eine Menge von Eigenschaften beschrieben, die sich von denen der Problembeschreibung unterscheiden (vgl. [Engels 1999]).
- RPU's sind anwendungsunabhängig, bezogen darauf, dass sie nicht notwendigerweise an eine spezielle Datenmenge und Projektergebnisse gebunden sind, wie die Gesamtprozessbeschreibung. Basierend auf der Beschreibung der Funktionalität sind RPU's anwendbar in Situationen mit ähnlichen Problemen und Daten.

5.2.2 *Simple Task* Repository

Simple Tasks sind die kleinsten Einheiten des Aufgabenzerlegungsprozesses.

Definition 5.1 (Simple Task:) *Ein Simple Task beschreibt im UGM Kontext eine Aufgabe auf dem niedrigsten Abstraktionslevel der Aufgabenzerlegung. Eine Simple Task braucht nicht weiterzerlegt zu werden, da sie so spezialisiert ist, dass die Aufgabe auf eine Methode abgebildet werden kann.*

Simple Tasks können mit den Folgerungen (*inferences*), wie sie in CommonKADS ([Schreiber u. a. 1993, Breuker und van de Velde 1994]) und MIKE [Angele u. a. 1996] beschrieben sind, verglichen werden. Folgerungsschritte in diesen Ansätzen beschreiben nicht zerlegbare Aufgaben. Im UGM–Ansatz sind *simple tasks*–Aufgaben auf der untersten Ebene der Aufgabenzerlegung, denen eine Menge von Methoden zugeordnet werden kann. Bei den Folgerungsschritten unterscheidet man 16 Basisfolgerungen, die ausreichen um komplexe Aufgaben zu lösen [Engels 1999]. *Simple Tasks* im UGM–Ansatz sind nicht das gleiche wie diese Basisinferenzen. Dieses ergibt sich aus der Definition und dem Rahmen des UGM–Ansatzes, in dem weitere praktische Anforderungen, wie nicht funktionale Anforderungen, berücksichtigt werden müssen.

5.2.3 Methodenbeschreibung

Die zerlegten Aufgaben müssen auf Methoden abgebildet werden. Eine der Annahmen im UGM–Ansatz ist, dass eine Menge von Methoden zur Datentransformation, Visualisierung, Erkennung und Modellierung vorhanden ist. Von diesen verfügbaren Methoden müssen geeignete adäquate Methoden für eine gegebene Problemstellung ausgewählt werden. Im Rahmen des UGM–Ansatzes werden die Methoden im folgenden Sinne beschrieben.

Anwendungsanforderungen: Bedingungen, die die Methode im Kontext des KDD–Prozesses erfüllen muss. Diese Bedingungen sind nicht nur funktional, sondern bestehen viel mehr aus nicht funktionalen pragmatischen Anforderungen des Unternehmens.

Ergebnisanforderungen: Diese beschreiben die Ergebnisse nach Anwendung der Methoden, unter der Vorraussetzung, dass die Anwendungsanforderungen eingehalten werden. Z.B. kann ein operationales Modell als Ergebnis gefordert sein.

Initiale Parameter: Initiale Instantiierung der Methode in Bezug zum Kontext zur Aufgabe.

Adaptionsregeln: Iterationen im Prozess können zu Änderungen der bestimmten Aufgabe führen. Adaptionsregeln sollen helfen, die neuen Parameter zu bestimmen.

In der vorliegenden Arbeit wird der Abbildungsprozess von Methoden mit einem Fokus auf Klassifikationsmethoden betrachtet.

5.2.4 Repository für Datenbeschreibungen

Der UGM–Ansatz besteht aus verschiedenen Dimensionen aus der Aufgabenzerlegung, aus der Problemstellung und den Zielen in Form von funktionalen und nicht funktionalen Restriktionen. Eine weitere Dimension ist die Beschreibung der Daten. Durch die gegebenen Daten gibt es weitere Einschränkungen bzgl. möglicher Lösungen. Bei der Benutzerunterstützung im UGM–Ansatz ist die Datenbeschreibung der zentrale Punkt in der *bottom up*–Unterstützung. Im Gegensatz dazu ist die Aufgabenzerlegung die *top down*–Komponente des Ansatzes. Im Rahmen des UGM–Ansatzes wird die Datenbeschreibung Datencharakteristik genannt.

Die Datencharakteristik lässt sich in folgende Kategorien unterteilen:

- Welche Art von Daten liegen vor bzgl. Dimensionen, Ausprägungen, Datenformate usw.?
- Datenbeschreibungen über Attributbeziehungen, Informationsgehalte und Informationsmaße in Verbindung zum Problemtyp.

Eine genaue Beschreibung der Datencharakteristik erfolgt im Kapitel 6.

5.3 Entscheidungsunterstützung im KDD– Prozess

5.3.1 Benutzerunterstützung bei kommerziellen Anbietern

In den folgenden Abschnitten werden die KDD–Systeme der *großen* Anbieter bezüglich ihrer Unterstützung im KDD–Prozess beschrieben. Ausgewählt wurden folgende Systeme:

1. Clementine von SPSS Inc.
2. Enterprise Miner von SAS Inc.
3. Intelligent Miner von IBM
4. MineSet von Silicon Graphics

5.3.1.1 Clementine

Clementine wurde von SPSS Inc. entwickelt und war eines der ersten kommerziellen KDD–Tools. Clementines visuelle Programmierung ermöglicht eine Visualisierung des KDD– Prozesses. Hierzu werden entsprechend der Prozessphaseneinteilung der obersten Ebene für

- den Datenzugriff,
- die Transformation,
- die Expolation,
- die Modellierung,
- die Evaluierung und
- die Anwendung der generierten Modelle

Methoden und Verfahren den Anwendern zur Verfügung gestellt.

Auf der Basis der Arbeit von Robert Engels und dem CRISP-DM (Esprit-Projekt), an dem die SPSS als Partner beteiligt war, ist eine zusätzliche Toolunterstützung zur Organisation und Dokumentation von Projekten integriert worden.

Trotzdem liegt heute noch in dem System der Schwerpunkt in der Bereitstellung von Methoden zum KDD–Prozess. Eine Unterstützung bei der Auswahl von geeigneten Methoden zu einer gegebenen Problemstellung mit entsprechenden Daten ist nicht Bestandteil des Systems.

5.3.1.2 Enterprise Miner

Der SAS–Enterprise Miner definiert seine eigene Methode *SEMMA*. SEMMA steht für:

- **S***ample*,
- **E***xplore*,
- **M***odify*,
- **M***odel* und
- **A***ssess*.

Dies ist nur eine Umsetzung der möglichen Schritte im KDD–Prozess in eine firmenspezifische Marketingsicht, eine eigene Methode anzubieten. Diese logische *Superstruktur* [Sang 2002] bietet Anwendern einen strukturierten Weg zur Konzepterstellung, Erstellung und Evaluierung von KDD–Projekten.

Die Methoden sind somit unterteilt nach der SEMMA Struktur, der der Anwender folgen soll. Welche Verfahren für sein Problem geeignet sind, geht hierbei nicht hervor. Hier kommt es wieder auf das Wissen des Anwenders an. Im SAS–Enterprise Miner wird der KDD–Prozess somit auf der obersten Ebene unterstützt und für jede Phase des Prozesses eine Menge von Methoden angeboten, die leicht wie in Clementine mit drag und drop in einem Verlaufsprozess angewendet werden können.

5.3.1.3 MineSet

MineSet ist ein generelles Datenanalysetool, das Datenzugriff, analytisches Data Mining und Datenvisualisierung in einem integrativen Umfeld und die Phasen des KDD–Prozesses nach [Fayyad u. a. 1996] unterstützt.

MineSet wird von Sillicon Graphics Inc. angeboten und besitzt daher umfangreiche Datenvisualisierungsmethoden. Die analytischen Data Mining–Methoden basieren auf der frei verfügbaren MLC++ Library [Kohavi und Sommerfield 2002], die eine Vielzahl von Methoden aus dem Bereich des maschinellen Lernens enthält.

Neben dem Ensemble von Datenzugriffskomponenten, Transformationen, analytischen Data Mining– und Visualisierungs–Methoden durch ein Benutzer–Interface, bietet MineSet eine Historifizierung und die Speicherung und spätere Ausführbarkeit von Modellen.

MineSet unterstützt wie die anderen Systeme den KDD–Prozess und den Anwender durch eine Menge von Methoden für die verschiedenen Phasen des Prozesses.

5.3.1.4 Intelligent Miner

Der Intelligent Miner von IBM bietet eine Menge von Methoden zur Unterstützung aller Phasen des KDD- Prozesses. Durch die enge Anbindung an DB2™ und weil Intelligent Miner Bestandteil der DB2–Produktfamilie ist, hat dieser eine offene SQL– Schnittstelle als Datenzugriffskomponente. Als Data Mining Verfahren bietet das System Verfahren zum Clustering, zur Assoziationsanalyse, sequenzielle Muster, Klassifikationsalgorithmen, Vorhersagealgorithmen und Zeitreihenanalyse. Darüber hinaus sind anpassbare Visualisierungsmethoden Bestandteil des Systems.

Der Intelligent Miner führt den Anwender durch die Phasen des Prozesses, bietet aber keine Unterstützung bei der Auswahl der geeigneten Methoden für die gegebene Problemstellung.

5.3.1.5 Zusammenfassung

Stand heute bietet keiner der großen betrachteten Hersteller eine Unterstützung bei der Methodenauswahl im KDD–Prozess. Alle hier vorgestellten Systeme unterstützen den KDD–Prozess auf der obersten Ebene, indem dem Anwender eine Auswahl von Methoden für jede Phase angeboten wird.

Wann und für welche Problemstellung welche Methode geeignet ist, wird nicht von Seiten der Systeme unterstützt. Die Auswahl der Methoden wird von den Anbietern auch als ein Beratungsgeschäft gesehen und stellt somit auch ein einheitliches Geschäftsmodell der Anbieter dar.

5.4 Problem der Methodenauswahl

Es gibt keine Methode, die für jedes Problem über alle Anwendungsgebiete die anderen im Ergebnis übertrifft. Der theoretische Beweis ist bekannt als das *No Free Lunch Theorem* [Wolpert 1994], [Schaffer 1994]. Somit ist es wichtig, für ein gegebenes Problem eine geeignete Methode zu finden. Die Praxis hat gezeigt, dass die Auswahl nicht trivial ist.

Methodenauswahl spielt ebenfalls eine Rolle in den Datenanalyse–Paradigmen der Statistik (vgl. [Hand 1994a], [Hand 1994b]). In der Vergangenheit wurde im maschinellen Lernen mit dem MLT Projekt und seinem Consultant System [Consortium 1993], sowie dem STATLOG Projekt [Michie u. a. 1994] versucht, die Methodenauswahl zu unterstützen. Diese Projekte haben die Performance einer festen Menge von Algorithmen über mehre Datensätze verglichen.

Einen ähnlichen Ansatz für die Methodenauswahl findet man in [Kohavi u. a. 1997], [Kohavi und Sommerfield 2002]. Mit MLC++ wird der Ansatz verfolgt, dass die am besten geeignetste Methode durch Ausprobieren ermittelt werden muss. Hierzu stellt MLC++ eine Sammlung von Werkzeugen in einer einheitlichen Umgebung zur Verfügung und erlaubt somit den Vergleich der Methoden, um die am besten

geeignete Methode zu der gegebenen Problemstellung und den gegebenen Daten zu finden. Hierzu muss allerdings jedes potentiell geeignete Verfahren ausprobiert werden, was einen hohen Arbeitsaufwand bedeutet.

Das EU-Projekt METAL (META Learning 1999-2002) verfolgte einen fast gleichen Ansatz wie er in dieser Arbeit beschrieben wird und von dem Autor auf der PKDD 1999 [Lindner und Studer 1999a] und auf der Konferenz der Gesellschaft für Klassifikation [Lindner und Studer 2000] sowie im *Handbook of Data Mining and Knowledge Discovery* beschrieben ist [Lindner u. a. 2002] vorgestellt wurde. Der Autor dieser Arbeit war zu Beginn des Projektes an dem Projekt beteiligt und auch die in Kapitel 6 vorgestellte Datencharakteristik mit dem entsprechenden Tool DCT wird von METAL genutzt. Seit Anfang 2003 kann DCT im Rahmen einer unterstützten Methodenauswahl von den Webseiten des Projektes heruntergeladen werden (www.metal-kdd.org). In Kapitel 7 wird der Ansatz von AST (**A***lgorithm* **S***election* **T***ool)* genauer beschrieben und mit dem von METAL in Kapitel 9.2 verglichen.

Der Grundansatz lässt sich wie folgt beschreiben:

1. Für die gegebenen Daten wird die Datencharakteristik berechnet.
2. Aus einer Erfahrungsdatenbank wird der *ähnlichste* Datensatz berechnet.
3. Anzeige der Datencharakteristik des gefundenen Datensatzes mit dem Ergebnis, das mit den jeweiligen angewendeten Methoden erzielt wurde.

Aufgrund der Komplexität und Vielzahl von Methoden wird im folgenden der Fokus auf die Gruppe der Klassifikationsverfahren gelegt. Basis für diese Vorgehensweise ist die Datencharakteristik, die im folgenden Kapitel 6 beschrieben wird.

Kapitel 6

Charakterisierung von Datensätzen

Mit der Charakterisierung von Datensätzen soll eine sinnvolle Beschreibung erstellt werden, die es ermöglicht, festzustellen bei welchen beschreibenden Merkmalen die einzelnen Verfahren die besten Ergebnisse liefern.

6.1 Einleitung

In diesem Kapitel werden zunächst die zu berechnenden Maße für die Beschreibung der Daten, die Datencharakteristik, vorgestellt. Dabei wird zwischen statistischen und informationstheoretischen Maßen unterschieden.

Die statistischen Maße beziehen sich auf die numerischen Attribute und ermöglichen Aussagen über Lage- und Streuungsparameter der einzelnen Attribute. Daneben sind auch Maße für alle numerischen Attribute als Ganzes definiert, so das Abhängigkeiten zwischen den Attributen sowie Tests auf multivariate Eigenschaften berücksichtigt werden können.

Bei den informationstheoretischen Maßen steht neben den Häufigkeitsverteilungen die Entropie im Mittelpunkt der Betrachtungen. Es werden verschiedene auf der Entropie basierende Maße berechnet, die Aussagen über den Informationsgehalt bzw. den Informationsgewinn der einzelnen Attribute ermöglichen.

Die hier vorgestellten Maße sind in dem *Datencharakteristik Tool* (DCT) implementiert, dass auch dem Projet METAL zur Verfügung gestellt wurde und im Data Mining Advisor (vgl. Abschnitt9.2.1) verwendet wurde.

6.2 Statistische Maße

Ist man an einer einfachen und kompakten Charakterisierung eines Datentupels interessiert, so kann dies durch Angabe von Lage- oder Streuungsparametern geschehen. Lageparameter geben über die "mittlere Lage" der Werte eines Attributes

Auskunft. Streuungsparameter beschreiben die Variabilität der Werte, also den Bereich, auf dem die Werte verteilt sind. Erst gemeinsam betrachtet bieten sie nützliche Informationen über die Verteilung eines Attributs.

In diesem Abschnitt werden Datentupel der Form $x_1, \ldots, x_n \in \mathbb{R}^p$ betrachtet, wobei p die Anzahl der Attribute symbolisiert. Man kann in diesem Zusammenhang auch von n unabhängigen Realisierungen eines p-dimensionalen Zufallsvektors X sprechen.

6.2.1 Lageparameter

Allgemein besitzt ein Lagemaß $\mu(x_1, \ldots, x_n)$ folgende Eigenschaft

$$\mu(x_1 + b, \ldots, x_n + b) \quad = \quad \mu(x_1, \ldots, x_n) + b, \qquad \text{mit } b \in \mathbb{R}. \tag{6.1}$$

Mit anderen Worten: Verschiebt man die Daten um $b \in \mathbb{R}$, so ändert sich das Lagemaß ebenfalls um diesen Wert. Neben dieser Eigenschaft ist es wichtig zu wissen, dass nicht alle Lagemaße robust sind, d. h. die Anwesenheit von Ausreißern [1] verzerrt teilweise die Ergebnisse erheblich. Eine ausführliche Betrachtung der Ausreißerproblematik findet man in [Barnett und Lewis 1994]. Nähere Informationen zu robusten statistischen Verfahren findet man in [Huber 1977] oder [Rey 1993]. Im Folgenden werden die implementierten Lagemaße vorgestellt.

6.2.1.1 Arithmetisches Mittel

Das arithmetische Mittel ist wohl das bekannteste Lagemaß. Es wird häufig zur Schätzung des Erwartungswertes einer Verteilung verwendet und ist wie folgt definiert

$$\bar{x} \quad = \quad \frac{1}{n} \sum_{j=1}^{n} x_j. \tag{6.2}$$

Leider ist das arithmetische Mittel nicht sonderlich robust, weshalb oft das folgende α-getrimmte Mittel gebildet wird.

6.2.1.2 α-getrimmtes Mittel

Das α-getrimmte Mittel von $x_1 \leq x_2 \leq \ldots \leq x_n$ ist wie folgt definiert:

[1] Als Ausreißer bezeichnet man Werte, die besonders groß oder besonders klein sind. Sie können z. B. durch systematische Fehler beim Messen der Daten entstehen.

$$t_\alpha = \frac{1}{n-2\lfloor n\alpha\rfloor}\sum_{j=\lfloor n\alpha\rfloor+1}^{n-\lfloor n\alpha\rfloor} x_j \quad \text{mit } 0 \le \alpha < \frac{1}{2}. \tag{6.3}$$

Für $(n\alpha) \le 1$ ist das α-getrimmte Mittel gleich dem Mittelwert und für $(n\alpha) \ge \frac{(n-1)}{2}$ gleich dem Median[2].

6.2.1.3 Empirische Quantile

Gegeben sei eine sortierte Folge von Datenpunkten $x_1 \le x_2 \le \ldots \le x_n$ und p eine reelle Zahl aus $[0, 1)$. Dann beschreibt (6.4) das empirische p-Quantil von $x_1 \le x_2 \le \ldots \le x_n$.

$$q_p(x_1, x_2, ...x_n) = \begin{cases} x_{(\lfloor np\rfloor+1)} & \text{falls } np \notin \mathbb{N} \\ \frac{1}{2}(x_{np} + x_{np+1}) & \text{falls } np \in \mathbb{N} \end{cases} \tag{6.4}$$

Das p-Quantil besitzt die Eigenschaft, dass mindestens np Werte unterhalb und mindestens $n(1-p)$ der Werte oberhalb liegen. Besondere Bedeutung hat das 0.5-Quantil, das auch als Median bezeichnet wird. Daneben werden häufig die Quantile $x_{0.25}$ und $x_{0.75}$ verwendet, die unteres bzw. oberes Quartil genannt werden. Bleibt anzumerken, dass empirische Quantile unempfindlich gegenüber Ausreißern sind.

6.2.2 Streuungsparameter

Im Gegensatz zu Lagemaßen ändern sich Streuungsmaße bei Verschiebung nicht. (Vgl. Gleichung (6.1).) Es gilt

$$\sigma(x_1 + b, \ldots, x_n + b) = \sigma(x_1, \ldots, x_n), \quad \text{mit } b \in \mathbb{R}. \tag{6.5}$$

6.2.2.1 Varianz

Das bekannteste und am häufigsten verwendete Streuungsmaß ist die Varianz, die ein Maß für die Streuung um den Schwerpunkt ist. Sie ist folgendermaßen definiert

$$\sigma^2(x_1, \ldots, x_n) = \frac{1}{n}\sum_{j=1}^{n}(x_j - \bar{x})^2. \tag{6.6}$$

[2]Der Median wird im Anschluss im Zusammenhang mit den empirischen Quantilen definiert.

6.2.2.2 Standardabweichung

Die Standardabweichung ist als die positive Wurzel der Varianz definiert. Problematisch bei der Standardabweichung ist die Ausreißeranfälligkeit.

$$\sigma(x_1, \ldots, x_n) = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(x_j - \bar{x})^2}. \tag{6.7}$$

6.2.2.3 Quartilsabstand

Der Quartilsabstand ist als die Differenz aus dem oberen und unteren Quartil definiert. Er gibt an, wie groß die Spannweite der mittleren 50% der Daten ist.

$$QA(x_1, x_2, \ldots, x_n) = q_{\frac{3}{4}}(x_1, x_2, \ldots, x_n) - q_{\frac{1}{2}}(x_1, x_2, \ldots, x_n). \tag{6.8}$$

6.2.2.4 Medianabweichung

Die Medianabweichung ist ein Maß für die Streuung einer Wertereihe, ausgedrückt durch die Abweichung vom Median. Das Maß hat den Vorteil, dass es bis zu 50% Ausreißer toleriert.

$$\begin{aligned} M(x_1, x_2, ..., x_n) &= q_{\frac{1}{2}}(y_1, y_2, ..., y_n), \\ \text{mit} \quad & y_j = \mid x_j - q_{\frac{1}{2}}(x_1, x_2, \ldots, x_n) \mid . \end{aligned} \tag{6.9}$$

6.2.3 Kovarianz und Korrelation

Bevor im Rahmen der Diskriminanzanalyse auf multiple Korrelationskoeffizienten eingegangen wird, sollen an dieser Stelle die Begriffe Kovarianz bzw. Kovarianzmatrix und Korrelation eingeführt werden. Die Kovarianzmatrix spielt im Rahmen der Arbeit eine wichtige Rolle, z. B. bei der Generierung der Multinormalverteilung.

In (6.11) wird eine dreidimensionale Kovarianzmatrix[3] definiert. Dabei bezeichnet $\mathbf{X}$ einen Zufallsvektor, X_1, X_2, X_3 sind Zufallsvariablen, σ_i bezeichnet die Varianz von X_i und ϱ_{ij} entspricht dem Korrelationskoeffizienten von X_i und X_j.

$$\mathbf{X} = \begin{pmatrix} X_1 \\ X_2 \\ X_3 \end{pmatrix} \tag{6.10}$$

[3]Eine ähnliche Definition der Kovarianzmatrix für den zweidimensionalen Fall findet man in [Hartung u. a. 1995].

$$\begin{aligned}\Sigma &= \begin{pmatrix} VarX_1 & Cov(X_1,X_2) & Cov(X_1,X_3) \\ Cov(X_2,X_1) & VarX_2 & Cov(X_2,X_3) \\ Cov(X_3,X_1) & Cov(X_3,X_2) & VarX_3 \end{pmatrix} \\ &= \begin{pmatrix} \sigma_1^2 & \sigma_1\sigma_2\varrho_{12} & \sigma_1\sigma_3\varrho_{13} \\ \sigma_2\sigma_1\varrho_{12} & \sigma_2^2 & \sigma_2\sigma_3\varrho_{23} \\ \sigma_3\sigma_1\varrho_{13} & \sigma_3\sigma_2\varrho_{23} & \sigma_3^2 \end{pmatrix}\end{aligned} \tag{6.11}$$

Der sich hieraus ergebende Standard-Korrelationskoeffizient[4] ist nicht invariant gegenüber Transformationen. Es wäre also wünschenswert, wenn man eine entsprechend robustere Kenngröße für die Korrelation zwischen den Attributen hätte.

6.2.4 Diskriminanzanalyse

Abgesehen von der Einführung der Kovarianzmatrix wurden bislang nur Maße angegeben, die sich auf ein Attribut beziehen. Ein Verfahren, welches sich mit der Erkennung von Abhängigkeiten zwischen den einzelnen Attributen einer Datenbank von Fallbeispielen befasst, ist die Diskriminanzanalyse. Werden Abhängigkeiten erkannt, kann die Komplexität des Problems entsprechend reduziert werden, sowie die entsprechende Anzahl der in Frage kommenden Klassifikationsalgorithmen. [Theusinger 1998] und [Zintz 1998] geben folgende Voraussetzungen für die Anwendbarkeit der Diskriminanzanalyse an:

1. Lineare Unabhängigkeit der (numerischen) Attribute
2. Multivariate Normalverteilung
3. Gleichheit der Kovarianzmatrizen für alle Klassen

Der Test auf lineare Unabhängigkeit geschieht mit Hilfe des Multiplen Korrelationskoeffizienten. Dabei wird für jedes Attribut untersucht, ob es von den restlichen Attributen linear abhängig ist. Für den Test auf Multivariate Normalverteilung wird der BHEP-Test [Henze und Wagner 1991] eingesetzt. Die Gleichheit der Kovarianzmatrizen wird mit Hilfe der M-Statistik getestet.

6.2.4.1 Der Multiple Korrelationskoeffizient

Der Multiple Korrelationskoeffizient lässt sich für mehr als drei Variablen am Besten in der Matrix Notation beschreiben.

Sei $\mathbf{X} = (X_1, X_2, \ldots, X_p)'$ ein Vektor und $\Sigma > 0$. Nun sind $\mathbf{X}$ und Σ definiert als

[4] Eine Definition findet man z. B. in [Hartung u. a. 1995]. Hier steht der Zusammenhang zwischen Standard-Korrelationskoeffiezent und Kovarianzmatrix im Vordergrund.

$$\mathbf{X} = \begin{pmatrix} X_1 \\ \mathbf{X_2} \end{pmatrix} \qquad \Sigma = \begin{pmatrix} \sigma_{11} & \sigma'_{12} \\ \sigma_{12} & \Sigma_{22} \end{pmatrix} \tag{6.12}$$

$\mathbf{X_2} = (X_2, \ldots, X_p)'$, $Var(X_1) = \sigma_{11}$, $Cov(\mathbf{X_2}) = \Sigma_{22}$, Σ_{22} hat die Dimension $(p-1) \times (p-1)$ und $\sigma 12$ ist ein Vektor $(p-1) \times 1$ mit Kovarianzen zwischen X_1 und jeder Variable in $\mathbf{X_2}$.

Der Multiple Korrelationskoeffizient wird definiert als

$$R = \sqrt{\frac{\sigma'_{12}\Sigma_{22}^{-1}\sigma_{12}}{\sigma_{11}}}, \qquad \text{wobei} \quad \Sigma = \begin{pmatrix} \sigma_{11} & \sigma'_{12} \\ \sigma_{12} & \Sigma_{22} \end{pmatrix} \tag{6.13}$$

die Kovarianzmatrix darstellt.

Der Multiple Korrelationskoeffizient kann nur Werte im Intervall $[-1, 1]$ annehmen. Er ist ein Maß für die lineare Abhängigkeit zwischen numerischen Attributen und kann, wie anfangs erwähnt, zur Reduzierung des Datenvolumens durch Eliminierung von linear abhängigen Attributen herangezogen werden.

6.2.4.2 Test auf Multivariate Normalverteilung (BHEP-Test)

Sei d die Anzahl der Attribute und $\beta \geq 0$ ein Glättungsparameter. Die Teststatistik wird wie folgt definiert

$$T_{(n,\beta)} = (4 \cdot \mathbf{1}\{S_n \text{ist singulär}\} + W_{n,\beta} \cdot \mathbf{1}\{S_n \text{ist nicht singulär}\}) \tag{6.14}$$

wobei

$$\begin{aligned} W_{n,\beta} &= \frac{1}{n^2}\sum_{j=1}^{n}\sum_{k=1}^{n} exp\left(-\frac{\beta^2}{2}\|Y_j - Y_k\|^2\right) \\ &\quad -2(1+\beta^2)^{-\frac{p}{2}}\frac{1}{n}\sum_{j=1}^{n} exp\left(-\frac{\beta^2\|Y_j\|^2}{2(1+\beta^2)}\right) + (1+2\beta^2)^{-\frac{d}{2}}, \\ Y_k &= S_n^{-\frac{1}{2}}(X_k - \bar{X_n}), \\ \bar{X_n} &= \frac{1}{n}\sum_{j=1}^{n} X_j, \\ S_n &= \frac{1}{n}\sum (X_j - \bar{X_n})(X_j - \bar{X_n})^t \end{aligned}$$

Als approximativer kritischer Wert für einen Test zum Niveau α wird

$$q_{\beta,p}(\alpha) = \mu_{\beta,p}\left(1 + \frac{\sigma^2_{\beta,p}}{\mu^2_{\beta,p}}\right)^{-\frac{1}{2}} exp\left(\Phi^{-1}(1-\alpha)\sqrt{\ln\left(1 + \frac{\sigma_{\beta,p}}{\mu_{\beta,p}}\right)}\right) \tag{6.15}$$

definiert, wobei

$$\begin{aligned}\mu_{\beta,p} &= 1-\gamma^{-\frac{p}{2}}\left[1+\frac{p\beta^2}{\gamma}+\frac{p(p+2)\beta^4}{2\gamma^2}\right], \quad \gamma = 1+2\beta^2 \\ \sigma^2_{\beta,p} &= 2(1+4\beta^2)^{-\frac{p}{2}}+2\gamma^{-p}\left[1+\frac{2p\beta^4}{\gamma^2}+\frac{3p(p+2)\beta^8}{4\gamma^4}\right] \\ &\quad -4\delta^{-\frac{p}{2}}\left[1+\frac{3p\beta^4}{2\delta}+\frac{p(p+2)\beta^8}{2\delta^2}\right], \quad \delta = 1+4\beta^2+3\beta^4\end{aligned}$$

$q_{\beta,d}(\alpha)$ bezeichnet das $(1-\alpha)$-Quantil der logarithmischen Normalverteilung.

6.2.4.3 M-Statistik

$$M = \gamma \cdot \sum_{i=1}^{q}(n_i-1)\ln|S_i^{-1}S| \tag{6.16}$$

wobei

$$\gamma = 1-\frac{2p^2+3p-1}{6(p+1)(q-1)}\left\{\sum_{i=1}^{q}\frac{1}{n_i-1}-\frac{1}{n-q}\right\} \tag{6.17}$$

und S_i und S die erwartungstreuen Schätzer der i-ten Stichproben-Kovarianzmatrix bzw. der gemeinsamen Kovarianzmatrix sind. Gilt M $\sim \chi^2_{p(p+1)(q-1)/2}$, so weisen die Klassen eine gleiche Kovarianzstruktur auf.[5]

6.2.4.4 Sd-Ratio

Sd-Ratio ist ebenfalls ein Maß, das Aussagen über die Kovarianzstruktur der Klassen macht. Es ist wie folgt definiert:

$$Sd-Ratio = \exp\frac{M}{(p\cdot\sum_{i=1}^{q}(n_i-1))} \tag{6.18}$$

wobei M das Ergebnis der M-Statistik darstellt.

Sd-Ratio ist echt größer Eins, falls die Kovarianzmatrizen unterschiedlich sind, und ist gleich Eins genau dann wenn die M-Statistik Null ist, d. h. die Kovarianzmatrizen die gleiche Struktur haben.

6.2.5 V-Statistik

Seien $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_r$ die positiven Eigenwerte von $W^{-1}B$. Folgende Maßzahlen sind für die V-Statistik relevant:[6]

[5] Die Approximation ist gut, falls $n_i \gg 20$ für alle i, p und $q \ll n_i$ sind für alle n_i.

[6] Einzelheiten zur V-Statistik können in [Theusinger 1998] nachgelesen werden.

6.2.5.1 Wilk's Lambda

$$\Lambda = \prod_{j=1}^{r} \frac{1}{1+\lambda_r} \tag{6.19}$$

Wilk's Lambda macht Aussagen über die Signifikanz der r Diskriminanzfunktionen. Ist der Wert von Wilk's Lambda größer als ein kritischer Wert einer χ^2-Verteilung mit $p \cdot (q-1)$ Freiheitsgraden, so folgt daraus, dass alle r Diskriminanzfunktionen signifikant sind. Ist diese Bedingung nicht erfüllt, muss die Signifikanz der einzelnen Diskriminanzfunktionen mit Hilfe der Bartlett's V-Statistik getestet werden.

6.2.5.2 Bartlett's V-Statistik

$$V = [(n-1) - \frac{1}{2}(p+q)] \sum_{j=1}^{r} \ln(1+\lambda_j), \tag{6.20}$$

wobei p die Anzahl der numerischen Attribute des Datensatzes ist und n die Anzahl der Datentupel und q die Anzahl der Klassen darstellt.

6.2.5.3 Funktion für j-ten Eigenwert

Die Signifikanz des j-ten Eigenwertes kann wie folgt bestimmt werden:

$$V_j = [(n-1) - \frac{1}{2}(p+q)] \ln(1+\lambda_j) \sim \chi^2_{p+q-2j} \tag{6.21}$$

Sind nicht alle r Diskriminanzfunktionen signifikant, müssen sukzessive Tests durchgeführt werden.

6.2.5.4 Test auf Signifikanz der n-ten Diskriminanzfunktion

$$V - \sum_{j=1}^{n} V_j \tag{6.22}$$

$V - V_1$ kann interpretiert werden als die residuale Diskriminierung nach Anwendung der 1. Diskriminanzfunktion. Die Teststatistiken mit dazugehörigen Freiheitsgraden sind in Tabelle 6.2.5.4 zusammengefasst.[7]

Sobald $V - \sum_{k=1}^{j} V_k$ kleiner ist als der kritische Wert der entsprechenden χ^2-Verteilung (siehe Tabelle 6.2.5.4), folgt, dass nur die ersten k Diskriminanzfunktionen signifikant zum angegebenen Signifikanzniveau sind.

[7] Eine detaillierte Beschreibung erfolgt in [Theusinger 1998].

Diskriminanzfunkt.	Statistik	Freiheitsgrade
1. Diskfunk.	$V - V_1$	$(p-1)\cdot(q-2)$
1. & 2. Diskfunkt.	$V - V_1 - V_2$	$(p-2)\cdot(q-3)$
$\vdots$	$\vdots$	$\vdots$

Tabelle 6.1: Teststatistiken

6.3 Informationstheoretische Maße

Bei der Analyse der symbolischen Attribute ist lediglich die Verteilung der Attribute von Bedeutung. Im Gegensatz zu den numerischen Attributen ist es hier nicht möglich, Beziehungen zwischen allen Attributen in Form von Maßen festzustellen. Sämtliche Maße beziehen sich daher immer nur auf einzelne Attribute.

Im Folgenden bezeichnet B ein symbolisches Attribut mit k Ausprägungen und $A = (A_1, \ldots, A_s)$ die Menge der symbolischen Attribute des Datensatzes.

Als einfache Maße werden vom DCT das Minimum, das Maximum und der Durchschnitt der Ausprägungen aller symbolischen Attribute ausgegeben.

Zentrales Maß zur Bestimmung der Eigenschaften symbolischer Attribute ist die Entropie, die hier als ein Maß zur Bestimmung des Informationsgehaltes eines Attributes benutzt wird. Sie ist ein Maß für die Abweichung der Verteilung eines symbolischen Attributes von der Gleichverteilung. Die vorgestellten Entropie-Maße können als ein Analogon zu einem Streuungsmaß für numerische Attribute aufgefasst werden.

6.3.1 Attributentropie

Bezeichnet X eine diskrete Zufallsvariable mit k Ausprägungen, so ist die Entropie von X definiert als:

$$H(X) := -\sum_{i=1}^{k} \pi_i \log_2(\pi_i) \tag{6.23}$$

wobei $\pi_i \quad (1 \leq i \leq k)$ die Wahrscheinlichkeit dafür bezeichnet, dass X den i-ten Wert annimmt ($\sum_{i=1}^{k} \pi_i = 1$).

Da ein Attribut als diskrete Zufallsvariable aufgefasst werden kann, gilt sinngemäß für die Entropie eines Attributes:

$$H_B = H(B) := -\sum_{i=1}^{k} \pi_i \log_2(\pi_i) \tag{6.24}$$

Es gilt, dass

$$0 \leq H(B) \leq \log_2(k) \tag{6.25}$$

Daraus läßt sich ableiten, dass

- $H(B)$ maximal ist, falls alle Werte gleich verteilt sind
- wenn B nur einen Wert annimmt, so hat $H(B)$ den Wert Null.

Damit gibt die Attributentropie an, wie viel ein Attribut zur Unterscheidung zwischen den Klassen beiträgt. Nimmt ein Attribut nur einen einzigen Wert an, so enthält es keine Information und trägt nicht zur Trennung zwischen den Klassen bei.

Vom DCT wird die **mittlere Entropie** aller symbolischen Attribute, die folgendermaßen definiert ist

$$\bar{H} := \frac{1}{s} \sum_{i=1}^{s} H_{A_i} \tag{6.26}$$

ausgegeben. Sie ist ein Maß für die gesamte in den Attributen enthaltene Information.

6.3.2 Klassenentropie

Die Entropie des Klassen- oder Zielattributes C wird auch als Klassenentropie bezeichnet. Analog zur allgemeinen Entropiedefinition ist die Klassenentropie:

$$H_C = H(C) := -\sum_{i=1}^{k} \pi_i \log_2(\pi_i) \tag{6.27}$$

Die Klassenentropie gibt an, wie viel Information notwendig ist, um eine Klasse zu spezifizieren. D.h. die Klassenentropie ist interpretierbar als die Anzahl der binären Fragen, die notwendig sind um zwischen zwei Klassen zu unterscheiden. Es ist somit möglich die Komplexität abzuschätzen; $H(C)$ gibt die Untergrenze für die Anzahl notwendiger Fragen an.

Die Klassenentropie wird ebenfalls vom DCT berechnet und ausgegeben.

Bisher wurden Attribute und Klassen getrennt betrachtet. Durch die gemeinsame Entropie können das Zielattribut und ein weiteres Attribut in Beziehung gesetzt werden.

6.3.3 Gemeinsame Entropie

Die gemeinsame Entropie (engl. joint entropy) $H(C, B)$ zweier Attribute C und B ist definiert durch

$$H_{CB} = H(C, B) := -\sum_{ij} \pi_{ij} \log_2(\pi_{ij}) \tag{6.28}$$

Sie ist ein Maß für die gesamte Entropie des gemeinsamen Systems der Attribute, d. h. aller Kombinationen (C,B). Dabei bezeichnet π_{ij} die gemeinsame Wahrscheinlichkeit für die Klasse i und den j-ten Wert von B.

Das vom DCT ausgegebene Maß ist ein Mittelwert der gemeinsamen Entropien aller symbolischen Attribute des Datensatzes.

6.3.4 Mutual Information

Der *mutual information* Wert eines Attributes und des Zielattributes ergibt sich als Differenz von

$$I_{mut}(C, B) := H_C - H_B + H_{CB} \tag{6.29}$$

Der Informationsgewinn gibt die Ersparnis in der Anzahl der Fragen an, falls Attribut B bekannt ist. Sind B und C unabhängig, so besteht zwischen ihnen kein Zusammenhang, und die gemeinsame Information ist Null.

Dem so ermittelten Wert sollte man aber eher kritisch gegenüberstehen. Hat ein Attribut viele Ausprägungen, so hat es üblicherweise auch einen hohen Informationsgewinn. Man stelle sich z. B. in einem Datensatz eine laufende Nummer vor. Kennt man diese laufende Nummer, so kann man auch auf die Klasse schließen, der Informationsgewinn ist sehr hoch. Allerdings wird kein Lernverfahren sehr gute Lernergebnisse aus diesem Sachverhalt liefern, da jede Nummer nur einmal vorkommt. Für neue Tupel mit neuen Nummern sind keine Aussagen möglich.

Wie auch schon bei der Attributentropie wird vom DCT der **mittlere mutual information Wert**

$$\bar{I}_{mut}(C, A) := \frac{1}{s} \sum_{i=1}^{s} I_{mut}(C, A) \tag{6.30}$$

errechnet und ausgegeben. Hierdurch wird ein Überblick gegeben, wie viel brauchbare Information über die Klassen in den Attributen enthalten ist.

Die Information, die benötigt wird um die Klasse zu bestimmen, ist H_C. Wenn das Klassifikationsproblem gelöst werden soll, muss diese Information aus den Attributen gewonnen werden. Es ist durchaus möglich, dass die brauchbare Information aller Attribute zusammen, $I_{mut}(C, A)$, größer ist als die Summe der einzelnen Informationen $I_{mut}(C, A_1) + \ldots + I_{mut}(C, A_s)$. Falls alle Attribute linear unabhängig sind, gilt Gleichheit:

$$I_{mut}(C, A) = I_{mut}(C, A_1) + \ldots + I_{mut}(C, A_s)$$

In diesem Fall trägt jedes Attribut, unabhängig von den anderen, brauchbare Information zur Klassifikation bei. Man kann bestimmen, wie viel Attribute im Mittel benötigt werden, indem man das Verhältnis der Klassenentropie zum Mittleren Informationsgewinn bestimmt.

6.3.5 Benötigte Attribute

Das Verhältnis

$$EN.attr := \frac{H(C)}{\bar{I}_{mut}(C, A)} \tag{6.31}$$

ist ein Anhaltspunkt für die Anzahl der zur Klassifikation benötigten Attribute. Es heißt daher “Equivalent Number of Attributes”

Auch wenn die lineare Unabhängigkeit der Attribute nicht gegeben ist, scheint es sinnvoll, dieses Maß zu berechnen. Es wird ebenfalls vom DCT berechnet und ausgegeben.

Eventuell trägt nur ein geringer Teil der in den Attributen enthaltenen Information zur Klassifikation bei. Wenn $\bar{I}_{mut}(C, A)$ ein Maß dafür ist, wie viel brauchbare Information über die Klassen in den Attributen enthalten ist, so gibt die Differenz $\bar{H}_A - \bar{I}_{mut}(C, A)$ den Anteil nicht brauchbarer Informationen an.

6.3.6 Rauschfaktor

Das Verhältnis von nützlicher und nicht brauchbarer Information

$$NS.ratio := \frac{\bar{H}_A - \bar{I}_{mut}(C, A)}{\bar{I}_{mut}} \tag{6.32}$$

wird als Rauschfaktor (engl. Signal Noise Ratio) bezeichnet. Große Werte für diesen Faktor zeigen an, dass der Datensatz hinsichtlich der Klassifikation viele irrelevante Informationen enthält. Auch dieses Maß wird vom DCT für den Datensatz berechnet und ausgegeben.

Weitere informationstheoretische Maße, die ebenfalls in DCT berechnet werden, können als Alternative zum mutual information Wert genutzt werden. Diese Maße sind der *Gini-Index* [Breiman u. a. 1984], das *Relevanzmaß* [Baim 1988] und die *g-Funktion* [Cooper und Herskovits 1992].

6.4 Einbindung von DCT

Die unmittelbare Verfügbarkeit des DCT innerhalb des Data Mining Prozesses spricht für die Einbindung in Clementine™ . Dadurch ist es möglich, einen Datensatz zu analysieren und seine Datencharakteristik zu bestimmen, ohne die Benutzerumgebung zu verlassen. Aufgrund der ermittelten Charakteristik können Veränderungen an der Verarbeitung des Datensatzes vorgenommen[8] und die Auswirkungen wiederum analysiert werden.

Das eigentliche Programm DCT ist in der Programmiersprache "C" implementiert. Da Clementine™ eine Schnittstelle zur Einbindung externer Programme zur Verfügung stellt, kann das Programm ohne Änderungen verwendet werden. Im Rahmen der vorliegenden Arbeit wurde die Einbindung des DCT in Clementine™ von einem `Terminal Node` in einen `Process Node` geändert. Dadurch ist es möglich, die vom DCT für den gesamten Datensatz berechneten Maße in Clementine™ weiter zu verarbeiten. Die im Stream weitergegebenen Daten entsprechen den in Tabelle 6.2 beschriebenen. Die Integration von DCT in Clementine™ war sehr nützlich für den Aufbau einer Benutzerunterstützung (vgl. Kapitel 8).

[8] Zum Beispiel können linear abhängige Attribute gestrichen oder mehrere Attribute miteinander kombiniert werden.

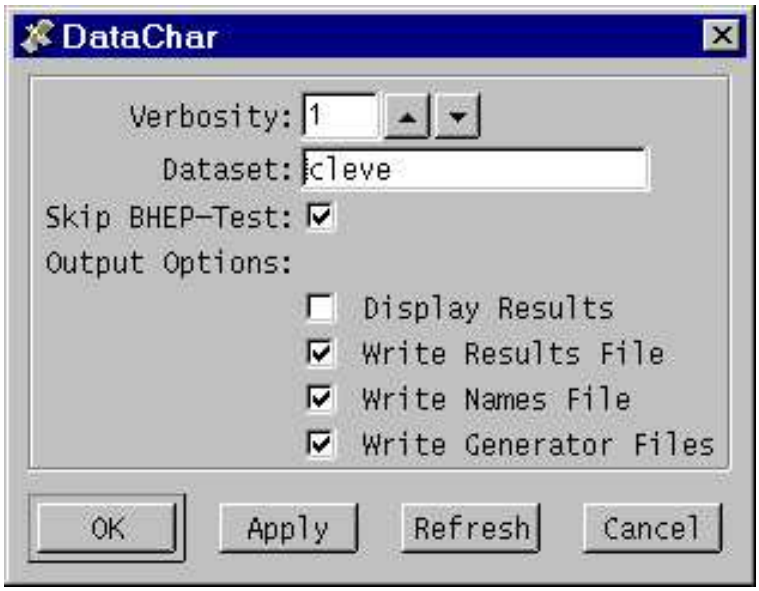

Abbildung 6.1: Data Characteristics Tool (DCT)

Abbildung 6.1 zeigt das Fenster zum Einstellen der Parameter des DCT in Clementine™ .

Verbosity: Durch einen Wert zwischen 1 und 8 kann die Fülle der Informationen, die vom DCT ausgegeben werden, eingestellt werden. Diese Option hat nur Auswirkungen, wenn auch `Display Results` aktiviert wurde. In den meisten Fällen reicht ein Level von 1, bei einem Level von 8 werden alle möglichen Informationen einschließlich der eingelesenen Daten angezeigt.

Dataset: Hier kann der Name des Datensatzes eingegeben werden. Standardmäßig wird der Name des ersten Knotens im Stream verwendet. Der hier eingestellte Bezeichner des Datensatzes wird vom DCT als Dateiname für die zu speichernden Informationen verwendet[9].

Skip BHEP-Test: Da der BHEP-Test schon bei relativ kleinen Datensätzen zu sehr langen Laufzeiten[10] führen kann und meist die Resultate der Diskriminanzanalyse verwendet werden ohne dass der BHEP-Test erfolgreich war, ist ein Schalter implementiert worden, um den BHEP-Test abzuschalten.

Write Results File: Wenn diese Option aktiviert ist, wird eine Datei mit der Endung ".res" erzeugt, in der die Maße für den Datensatz entsprechend Tabelle 6.2 abgespeichert werden.

[9] Die erzeugten Dateien werden in das Verzeichnis geschrieben, das von Clementine™ als aktuelles Verzeichnis gesetzt ist. Der unter Dataset angegebene Name dient als Wurzel für den Dateinamen, entsprechend der Bedeutung der Datei werden unterschiedliche Endungen verwendet.

[10] Der BHEP-Test hat eine Komplexität von $O(n^3)$, mit n = Anzahl der Tupel. Außerdem hängt die Laufzeit von der Anzahl der numerischen Attribute ab.

Write Names File: Für den analysierten Datensatz wird bei Aktivierung dieser Option ein `Names-File` entsprechend des C4.5-Formates (mit der Endung ".names"), wie es unter anderem auch von MLC++ verwendet wird, erzeugt. Mit dem so erzeugten Names-File ist es möglich, den Datensatz mit den von MLC++ unterstützten Data Mining Verfahren zu untersuchen.

Write Generator Files: Wie schon erwähnt, kann das DCT auch dazu verwendet werden, um die Regeln für einen Datengenerator zu erzeugen (siehe Kapitel 10). Mit Hilfe dieser Regeln ist es möglich, einen ähnlichen Datensatz zu generieren. Für weitere Informationen und zum Format der drei Dateien mit den Endungen ".covar", ".domain" und ".rules" sei auf [Kleiner 1998] verwiesen.

6.5 Zusammenfassung

In diesem Kapitel wurden eine ganze Reihe von Maßen vorgestellt, die auch in Form des DCT zur Berechnung der Charakteristiken einer Datenbank von Fallbeispielen zur Verfügung stehen.

Es bleibt festzuhalten, dass sowohl für symbolische als auch für numerische Attribute adäquate Maße definiert wurden. Zusammen beschreiben sie die Charakteristiken einer Datenmenge. In den folgenden Kapiteln wird beschrieben, wie diese Maße dazu genutzt werden, die Methodenauswahl zu unterstützen. Weiterhin sind diese Maße die Basis für den Generator zur Generierung von Datensätzen in Kapitel 10. In diesem Kapitel werden auch die Auswirkungen von Veränderungen der Charakteristiken auf den Lernerfolg von Data Mining–Verfahren untersucht.

Zur Vertiefung der angegebenen Maße, die hier nur knapp definiert sind, kann u. a. auf [Hartung u. a. 1995] und [Brandt 1981] zurückgegriffen werden. Für die angegeben informationstheoretischen Maße sei auf [Borgelt und Kruse 1998] verwiesen.

DCT wurde auch dem EU Projekt METAL zur Verfügung gestellt und ist die Basis für den dort entwickelten DM-Advisor. Ein Vergleich der Ansätze und Ergebnisse wird in Kapitel 9.2 beschrieben.

	einfache Maße	
	Nr_{Attr}	Anzahl der Attribute
	Nr_{sym}	Anzahl der symbolischen Attribute
	Nr_{num}	Anzahl der numerischen Attribute
	Nr_{Val}	Anzahl der Tupel
	Nr_{Class}	Anzahl der Klassen
	Acc_{def}	Wahrscheinlichkeit der häufigsten Klasse
	$Sdev_{Class}$	Standardabweichung der Klassenverteilung
	$MissVal$	relative Häufigkeit fehlender Werte
	$Tupel_{MissVal}$	relative Häufigkeit von Tupeln mit fehlenden Werten
	Maße für numerische Attribute	
	$Skew_{mean}$	Mittelwert der Schiefe aller numerischen Attribute
	$Kurt_{mean}$	Mittelwert der Wölbung aller numerischen Attribute
	Nr_{outl}	Anzahl der Attribute mit Ausreißern
BHEP-Test	Res_{BHEP}	Ergebnis des BHEP-Testes
	$CritVal_{0.1}$	kritischer Wert für $\alpha = 0.1$
	$CritVal_{0.05}$	kritischer Wert für $\alpha = 0.05$
	$CritVal_{0.01}$	kritischer Wert für $\alpha = 0.01$
M-Statistik	Res_M	Ergebnis der M-Statistik
	$Res_{\chi^2 M}$	Ergebnis der zugehörigen χ^2-Verteilung
	SD_{Ratio}	Wert von SD-Ratio
V-Statistik	$Fract1$	Wert für Fract1
	$Cancor1$	Wert für Cancor1
	Λ_{Wilk}	Wert für Wilk's Lambda
	Res_V	Ergebnis von Bartlett's V-Statistik
	$Res_{\chi^2 V}$	Ergebnis der zugehörigen χ^2-Verteilung
	Nr_{Disc}	Anzahl der signifikanten Diskriminanzfunktionen
	Maße für symbolische Attribute	
	$Nr_{sym_{min}}$	minimale Anzahl von Ausprägungen eines symbolischen Attributes
	$Nr_{sym_{max}}$	maximale Anzahl von Ausprägungen eines symbolischen Attributes
	$Nr_{sym_{av}}$	durchschnittliche Anzahl von Ausprägungen eines symbolischen Attributes
	H_{Class}	Klassenentropie
	H_{Attr}	mittlere Entropie aller symbolischen Attribute
	I_{gain}	mittlerer Informationsgewinn aller symbolischen Attribute
	H_{joint}	Mittelwert der gemeinsamen Entropien aller symbolischen Attribute
	EN_{Attr}	Anzahl der benötigten Attribute
	NS_{Ratio}	Signal-Rausch-Abstand

Tabelle 6.2: Von DCT berechnete Maße

Teil III

Ansatz zur Unterstützung der Algorithmenauswahl

Kapitel 7

Architektur von AST

Der hier beschriebene Ansatz, sowie die Ergebnisse sind auch in dem *Handbook of Data Mining and Knowledge Discovery* [Lindner u. a. 2002], auf einem Workshop der ICML 1999 [Lindner und Studer 1999a] und auf der 23. Konferenz der Gesellschaft für Klassifikation [Lindner und Studer 2000] veröffentlicht worden.

7.1 Einleitung

Wie im Kapitel 5 ausgeführt wurde, ist die Auswahl eines geeigneten wenn nicht gar des für die spezielle Anwendung besten Data Mining Verfahrens nicht trivial und erfordert einige Erfahrungen auf diesem Gebiet. Auch das Ausprobieren aller verfügbaren Verfahren, um auf diesem Wege das Beste zu ermitteln, ist nicht immer möglich. Bei den für die vorliegende Arbeit durchgeführten Versuchen ergaben sich Laufzeiten der einzelnen Verfahren von weniger als einer Sekunde bis zu mehreren Tagen, eine Zeitspanne, die nicht immer zur Verfügung steht. Wünschenswert wäre es also, bei Kenntnis des Datensatzes durch ein geeignetes Verfahren bestimmen zu können, welche Ergebnisse die einzelnen Algorithmen erreichen werden, ohne sie alle ausprobieren zu müssen.

Die erreichbare Fehlerrate ist aber nicht das einzige Kriterium, das bei der Auswahl eines Verfahrens berücksichtigt werden muss. Aus dem festgelegten Ziel der Anwendung können sich Restriktionen ergeben, die einzelne Verfahren geeignet erscheinen und andere ganz ausscheiden lassen. Wenn es z. B. darauf ankommt zu erkennen, warum ein Tupel einer bestimmten Klasse zugeordnet wird, werden sicherlich Verfahren, die Regeln erzeugen, bevorzugt werden, während neuronale Netze oder Nearest Neighbor Verfahren nicht geeignet sind. Eventuell ist auch die Zeit begrenzt, die zum Erstellen eines Modells oder für die Klassifikation der Tupel eines Datensatzes bei Anwendung eines einmal ermittelten Modells zur Verfügung steht. Aber auch die konkrete Verfügbarkeit einzelner Verfahren beim Anwender kann ein Kriterium für deren Einsatz sein.

In der Vergangenheit hat man wie in STATLOG [Michie u. a. 1994] und im MLP Projekt CONSULTANT [Consortium 1993] versucht, durch Lernen von Regeln den

Auswahlprozess von Methoden zu unterstützen (Meta-Lernen). Auch hier lag der Fokus auf Klassifikationsaufgaben wie in der vorliegenden Arbeit auch. Eine solche Regelbasis hat zwei wesentliche Nachteile:

1. Wartbarkeit: Es werden neue Methoden entwickelt. In einem solchen Fall muss die Regelbasis neu erstellt werden und die neue Methode auf die Referenzdatensätze angewendet werden.

2. Starre Regeln: Die gelernten Regeln haben starre bzw. harte Regelgrenzen. In Abbildung 7.1 ist ein Beispiel dargestellt. Hier ist anzuzweifeln, dass diese gute Empfehlungen für neue reale Datensätze geben können. Voraussetzung hierfür wäre, dass die Referenzdatensätze eine gute Stichprobe der realen Welt sind. Aktuell stehen aber nur wenige reale Anwendungen aus Unternehmen zur Verfügung, da diese wichtige Informationen und Daten enthalten, von denen sich die Unternehmen Wettbewerbsvorteile versprechen. Aus diesem Grund stehen zu wenige solcher Datensätze zur Verfügung um alle Verfahren auf diesen Daten testen zu können.

```
NrAttr =< 5   -> Appl
NrAttr > 5
        Noise =< 4.218  -> Appl
        Noise > 4.218
                NrAttr =< 13.5
                        NrNum / NrAttr =< 0.395  -> Non-Appl
                        NrNum / NrAttr > 0.395
                                NrOut / NrNum =< 0.405  -> Appl
                                NrOut / NrNum > 0.405  -> Non-Appl
                NrAttr > 13.5  -> Non-Appl
```

Abbildung 7.1: Beispiel für Regel oder Entscheidungsbäume zur Methodenauswahl

Aufgrund der beschriebenen Problematik wird in dieser Arbeit ein Ansatz zur Methodenauswahl vorgestellt, der der Idee folgt, die Erfahrungen aus der Anwendung der Methoden, die für die Experten bei der Auswahl der Methoden so hilfreich sind, in einer Wissensbasis zu speichern und den Anwendern zur Verfügung zu stellen. Die Auswahl einer geeigneten Methode wird durch die Bestimmung der ähnlichsten Anwendung unterstützt. Neben den beschreibenden Merkmalen der Anwendung und der Datencharakteristik erhält der Anwender die Information, welche Ergebnisse erzielt werden konnten.

In Abbildung 7.2 ist das Vorgehen, wie eine Unterstützung der Anwender in der Methodenauswahl aussieht, dargestellt.

Aus dem UGM–Ansatz (vgl. Kapitel 5) heraus gibt es zwei Einflussfaktoren für die Methodenauswahl:

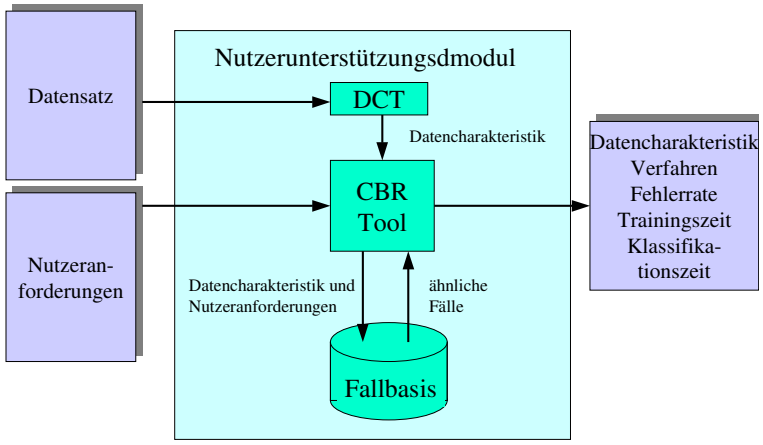

Abbildung 7.2: CBR-Ansatz zur Unterstützung der Methodenauswahl

1. die Datencharakteristik

2. die Anwendungsrestriktionen

Dieses sind die Eingabefaktoren zur Methodenauswahl in AST (**A***lgorithm* **S***election* **T***ool*). Zu den Datensätzen wird mittels DCT die Datencharakteristik berechnet. AST ist die Umsetzung des hier vorgestellten Ansatzes und wird in Kapitel 8 beschrieben. Auf Basis der Datencharakteristik und den Anwendungsrestriktionen wird nach vergleichbaren Anwendungen gesucht. Grundidee ist also aus den Erfahrungen zu lernen und diese zur Methodenauswahl zu nutzen. Aus Erfahrungen neue Lösungsansätze zu entwickeln ist auch die Kernidee des fallbasierten Schließens (*Case-Based Reasoning (CBR)*). In den folgenden Abschnitten wird das Prinzip des fallbasierten Schließens erklärt und auf die Vorteile für die Methodenauswahl eingegangen.

7.2 Einführung in CBR

Fallbasiertes Schließen [1] (*engl. Case-Based Reasoning, CBR*) ist ein Gebiet der künstlichen Intelligenz. Es folgt dem natürlicher Ansatz zur Modellierung des menschlichen Denkens und zum Bau von intelligenten Systemen.

[1] Im Folgenden sollen fallbasiertes Schließen und die gebräuchliche englische Abkürzung CBR synonym verwendet werden.

> Case-Based Reasoning can mean adapting old solutions to meet new demands, using old cases to explain new situations, using old cases to critique new solutions, or reasoning from precedents to interpret a new situation or create an equitable solution to a new problem.
>
> ⋮
>
> Case-Based Reasoning is both [...] the ways people use cases to solve problems and the ways we can make machines use them.
>
> [Kolodner 1993]

CBR hat seine Wurzeln nicht nur in der Informatik, sondern auch in anderen Wissenschaften, wie Kognitionswissenschaften und Mathematik (Statistik), zu denen enge Verflechtungen bestehen. Aber insbesondere zu anderen Gebieten der Informatik, wie wissensbasierte Systeme, Datenbanken, maschinelles Lernen, Mustererkennung und anderen bestehen zahlreiche Berührungspunkte. CBR ist somit wie KDD aus einer Kombination von Gebieten entstanden (vgl. Kapitel 2). Von der obigen Abgrenzung von CBR ist aus der Sicht der Informatik neben der Erforschung des Umganges mit Erfahrungswissen hauptsächlich der Bau von intelligenten Systemen von Interesse. Abbildung 7.3 zeigt ein einfaches Modell eines CBR–Systems gegenüber einem klassischen wissensbasierten System in Abbildung 7.4.

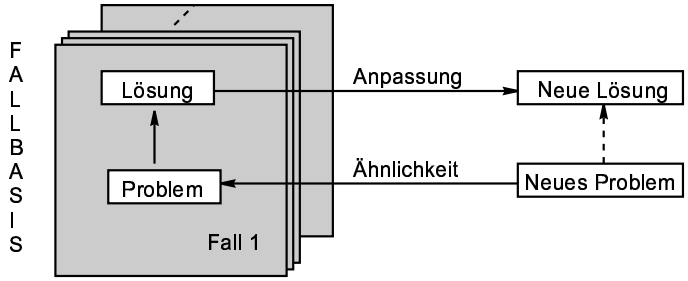

Abbildung 7.3: Einfaches Modell eines fallbasierten Systems

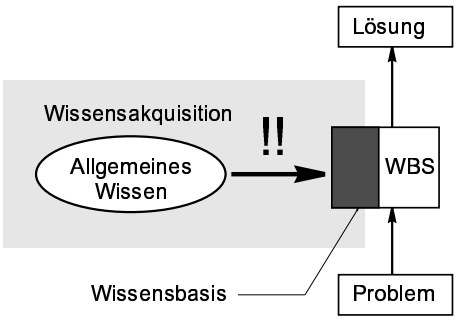

Abbildung 7.4: Klassisches wissensbasiertes System

Die Motivation für diesen neuen Ansatz resultiert hauptsächlich aus der Erfahrung, wie Menschen üblicherweise Probleme lösen. Im täglichen Leben setzen Menschen

sehr häufig *Fälle* zum Problemlösen ein. Ein Arzt erinnert sich zum Beispiel an die Krankengeschichte eines anderen Patienten, ein Jurist argumentiert mit einem Präzedenzfall, ein Verkäufer schildert den erfolgreichen Einsatz des Produktes bei einem anderen Kunden. Aber auch ein Mathematiker versucht, bekannte Lösungsmethoden oder Beweise auf neue Probleme zu übertragen, und sehr oft erfolgt Planung auf der Grundlage bereits realisierter Projekte. Neben diesem Erfahrungswissen spielt ein allgemeines Fachwissen bei der Problemlösung ebenfalls eine wichtige Rolle: Wann ist die Krankengeschichte eines anderen Patienten als Vergleichsbasis geeignet, welche Gesetze, die sich seit dem Präzedenzfall geändert haben, sind für den vorliegenden Fall von Belang, was ist bei der neu zu planenden Situation anders als bei den vorangegangenen und in welcher Weise muss von alten Methoden abgewichen werden, welche Veränderungen mussten an der Beweisführung vorgenommen werden, damit das neue Problem gelöst werden konnte und sind eventuell neue allgemeingültige Verfahren entwickelt worden, die es lohnt, zu bewahren.

Die zentrale Grundidee des CBR lässt sich so beschreiben:

- Gemachte Erfahrungen (Fälle, engl. Cases) werden gespeichert.
- Zum Lösen einer neuen Aufgabe werden
  - ähnliche Erfahrungen aus dem Speicher abgerufen,
  - die Erfahrungen im Kontext der neuen Situation ganz, teilweise oder modifiziert wiederverwendet,
  - die dabei neu gewonnenen Erfahrungen wieder gespeichert.

In der Domäne der Methodenauswahl lässt sich der CBR–Ansatz wie folgt umsetzen:

- Die gemachten Erfahrungen in Anwendungsdomänen werden gespeichert. Ein solcher Fall setzt sich aus den Anwendungsanforderungen, der Datencharakteristik und den erzielten Ergebnissen zusammen.
- Im Rahmen einer Methodenauswahl müssen folgende Schritte gemacht werden:
  1. Es werden die ähnlichsten Fälle bestimmt.
  2. Aus dieser Menge der Erfahrungen bestimmt der Anwender auf Basis der weiteren Anwendungsrestriktionen, welche Methode angewendet werden soll. Ein Ausprobieren aller Methoden entfällt und kann auf eine kleine ausgewählte Menge reduziert werden.
  3. Die vorgeschlagenen Methoden werden in der Anwendungsdomäne angewendet.
  4. Die hierbei neu gewonnenen Erfahrungen werden wieder in der Fallbasis gespeichert.

In Kapitel 8 wird die Umsetzung dieses Ansatzes im Detail beschrieben.

Gleichzeitig verspricht der Ansatz, einige der Schwierigkeiten, die die künstliche Intelligenz besonders beim Entwurf von Expertensystemen in der Vergangenheit hatte, beseitigen zu können.

Nach [Leake 1996] haben sich fünf Hauptprobleme herauskristallisiert, für die CBR bessere Lösungen erwarten lässt:

Wissensaquisition: Ein klassisches Problem traditioneller wissensbasierter Systeme ist der hohe Aufwand bei der Gewinnung des notwendigen Domänenwissens, das meist in Form von Regeln dargestellt wird. Es kann sehr schwierig sein, diese Regeln zu erkennen und umzusetzen, oder die Anzahl der notwendigen Regeln kann so groß sein, dass sie sich nicht verwalten lassen.

Fallbasierte Systeme erfordern sehr wenig allgemeines Wissen. Da sie von kompletten Situationen ausgehen, ist es nicht erforderlich, die einzelnen Fälle zu zerlegen und allgemeine Regeln aufzustellen. Das notwendige Fallwissen[2] lässt sich oft sehr einfach beschaffen oder liegt bereits vor.

Für die Methodenauswahl trifft dieser Punkt zu. In dieser Domäne ist es sehr schwer das Domänenwissen zu erfassen oder sogar in Regeln darzustellen. Die Problematik wurde in den Praxiskapiteln schon beschrieben. Durch das Erfassen der Gesamtsituation wird das Wissen implizit erfasst.

Wartung des Wissens: Das Aufstellen einer Wissensbasis ist nur der erste Schritt einer erfolgreichen Anwendung. Oft wird das Problem erst im Laufe der Entwicklung ausreichend verstanden oder das Umfeld für den Einsatz der Anwendung ändert sich. Regelbasen sind dabei nur mit großem Aufwand zu warten. Es existieren viele Abhängigkeiten zwischen den Regeln, die Regeln sind aufgrund der Repräsentation oft nur dem Experten verständlich, und die Auswirkungen von Änderungen sind nur sehr schwer vorherzusagen. Daher können regelbasierte Systeme meist nur von einem KI-Experten gewartet werden.

Demgegenüber sind fallbasierte Systeme sehr viel einfacher zu warten. Die abgespeicherten Fälle sind in aller Regel unabhängig voneinander und auch für den Laien[3] verständlich. Die Wartung der Wissensbasis kann durch Hinzunahme oder Löschen von Fällen erfolgen, wenn sich herausstellt, dass die ursprüngliche Fallbasis den Anforderungen nicht genügt. Das kann auch vom Benutzer der Anwendung durchgeführt werden, ein spezieller Experte ist dazu meist nicht notwendig.

Die Domäne der Methodenauswahl ist ein sehr komplexes und nicht komplett verstandenes Wissensgebiet. Die Pflege einer solchen Wissensbasis durch Regeln ist aufgrund der Problematik in der Wissensakquisition nicht denkbar. In

[2] Das Wissen ist in fallbasierten Systemen unter anderem implizit in den abgespeicherten Fällen enthalten (siehe 7.4).

[3] Gemeint ist hier ein Laie auf dem Gebiet der künstlichen Intelligenz, nicht ein Laie auf dem Gebiet der Domäne der Anwendung.

der Methodenauswahl mit CBR erfolgt die Wartung durch die Hinzunahme von kompletten weiteren Fällen. Ein Löschen von Fällen in dieser Domäne ist nicht notwendig, da jeder Fall zur Erweiterung des Wissens beiträgt.

Effizienz beim Problemlösen: Viele Probleme des täglichen Lebens, wie z.B. das Problem der Planung, sind NP-vollständig und erfordern einen sehr hohen Aufwand, wenn sie immer wieder von Grund auf gelöst werden.

Die Wiederverwendung früherer Lösungen hilft, die Effizienz bei der Problemlösung zu erhöhen. Auch das Abändern bestehender Lösungen ist oft einfacher und damit effizienter als das Berechnen einer neuen Lösung. Da fallbasierte Systeme genauso gut fehlgeschlagene Versuche wie erfolgreiche Lösungen speichern können, sind sie auch in der Lage, vor möglichen Gefahren zu warnen.

Im Rahmen der Methodenauswahl ist das Ausprobieren aller potentiellen Methoden mit einem nicht realisierbaren Aufwand verbunden. Selbst eine Hilfe bzw. eine Reduzierung der potentiell anwendbaren Methoden wäre eine wesentliche Verbesserung.

Qualität der Lösung: Oft werden die Problembereiche einer Domäne nicht vollständig verstanden. Selbst Experten sind daher nicht in der Lage, allgemeingültige Regeln anzugeben, um gute Lösungen zu finden. Werden in einer derartigen Domäne regelbasierte Systeme verwendet, so werden die Regeln also nicht perfekt sein, und insbesondere im Randbereich der Domäne sinkt die Qualität der Lösung extrem ab.

In solchen Situationen können Lösungen, die durch fallbasierte Systeme vorgeschlagen werden, besser sein. Sie gehen von den vorliegenden Verhältnissen aus und haben gute Lösungen gespeichert. Wenn nur geringe Anpassungen notwendig sind, wird das meist keine großen Auswirkungen auf die Qualität der Lösung haben.

Die Domäne der Methodenauswahl ist nicht vollständig verstanden. Da selbst Experten dazu raten, die Methoden auszuprobieren, liegt in einem CBR–Ansatz zur Methodenauswahl ein hohes Potential der Qualitätsverbesserung.

Benutzerakzeptanz: Eine Anwendung ist nur dann etwas wert, wenn die Ergebnisse vom Nutzer akzeptiert werden. Der Inferenzprozess regelbasierter Systeme ist oft recht kompliziert, so dass viel Aufwand getrieben werden muss, um dem Nutzer zu erklären, wie die Lösung gefunden wurde.

Fallbasierte Systeme besitzen eine hohe Erklärungsfähigkeit. Der ausgewählte Fall und eventuelle Lösungsanpassungen können dem Nutzer präsentiert werden. Der Vergleich dieses präsentierten Falles mit einer gegebenen Situation ist meist viel intuitiver als das Nachvollziehen des Inferenzmechanismus eines regelbasierten Systems.

Natürlich ist CBR nicht die Lösung für alle Probleme. Abbildung 7.5 zeigt das Anwendungsspektrum, in dem fallbasierte Systeme erfolgreich eingesetzt werden können und auch in der Praxis schon eingesetzt werden.

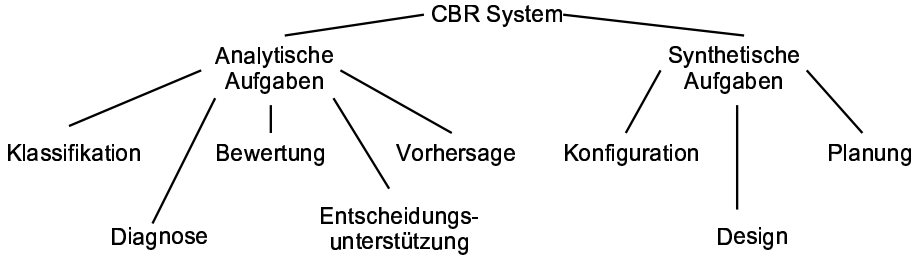

Abbildung 7.5: Typische Aufgabenklassen für fallbasierte Systeme

Die erfolgreiche Anwendung von CBR hängt davon ab, wie gut es gelingt, vorliegende Fälle zu beschreiben, für ein gegebenes Problem den ähnlichsten Fall zu finden und eventuell die Lösung an die aktuelle Situation anzupassen. Dabei hat sich in der Praxis ein Zyklus bei der Anwendung von CBR bewährt, der in [Aamodt 1995] beschrieben wurde.

7.3 Der CBR–Zyklus

Abbildung 7.6 zeigt den Ablauf der Anwendung eines fallbasierten Systems. Ausgehend von einem Problem wird ein neuer Fall (New Case) erzeugt. Zu diesem neuen Fall wird aus einer Menge von früheren, abgespeicherten Fällen (Previous Cases) der ähnlichste Fall (Retrieved Case) ermittelt (RETRIEVE) und als Lösung des Problems (Suggested Solution) vorgeschlagen. Dabei kann eventuell eine Anpassung (REUSE) an das Problem erforderlich sein. Aus dem neuen Fall wird unter Zuhilfenahme der abgespeicherten Fälle eine Lösung (Solved Case) erzeugt. Der Begriff Fall ist hier von zentraler Bedeutung und wird im folgenden Abschnitt 7.4 definiert.

Bis hierher wird der Zyklus[4] von jedem CBR–System durchlaufen. Für ein einzelnes Problem lässt sich so bereits ein Lösungsvorschlag ermitteln. Mit dem zweiten Teil des Zyklus ist aber eine Verbesserung der Qualität des CBR–Systems möglich. Die gefundene Lösung wird einer Überprüfung (REVISE) unterzogen und nötigenfalls korrigiert. Es entsteht eine verbesserte Lösung (Tested/Repaired Case). Die bei der Überprüfung und eventuellen Korrektur gemachten Erfahrungen können bewahrt (RETAIN) und die korrigierte Lösung als neuer Fall (Lerned Case) im System gespeichert werden. Bevor im folgenden auf die einzelnen Verarbeitungsschritte näher eingegangen wird, soll die Art der Wissensspeicherung in fallbasierten Systemen erläutert werden und eine Bestimmung des Begriffes *Fall* erfolgen.

7.4 Der Begriff *Fall*

In [Kolodner und Leake 1996] wird der Begriff *Fall* folgendermaßen definiert:

[4]Streng genommen ist es erst eine lineare Abfolge und noch kein Zyklus.

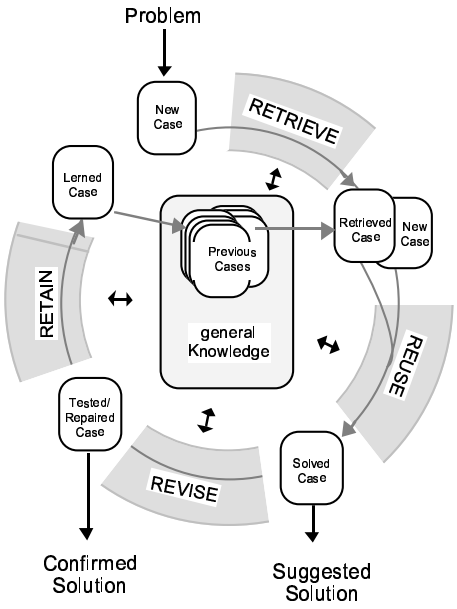

Abbildung 7.6: CBR–Zyklus (entnommen [Aamodt und Plaza 1994])

> *A* **case** *is a contextualized piece of knowledge representing an experience that teaches a lesson fundamental to achieving the goals of the reasoner.*

Diese Definition enthält einige Begriffe, die der näheren Erklärung bedürfen. Fälle können in vielfältiger Form auftreten. Allgemein ist ein Fall eine Situation, die ein Problem und eine Lösung beschreibt[5]. Aber nicht jede Situation ist ein brauchbarer Fall. Entscheidendes Kriterium ist, ob der Fall nützliches Wissen enthält, nützlich zum Erreichen eines Zieles. Während Menschen eine breite Palette von Zielen verfolgen und schon geringe Unterschiede über die Nützlichkeit des Wissens entscheiden können, ist die Lage für Maschinen einfacher. Ein System der künstlichen Intelligenz besitzt in der Regel eine sehr begrenzte Anzahl von Zielen, und damit lässt sich auch besser entscheiden, ob das Wissen für das jeweilige System nützlich ist oder nicht. Zusammenfassend werden folgende Anforderungen an einen Fall gestellt[6]:

- Ein Fall enthält in einem bestimmten Kontext spezifisches Wissen auf einem operationalen Niveau.
- Ein Fall kann in vielfältiger Form auftreten. Er kann einen kurzen oder längeren Zeitraum umfassen, und er verbindet Lösungen mit Problemen, oder Ergebnisse mit Situationen, oder beides.
- Ein Fall enthält Erfahrung, die helfen kann, ein Ziel oder eine Gruppe von Zielen in Zukunft einfacher zu erreichen. Die Erfahrung kann auf bisher nicht berücksichtigte Probleme hinweisen und somit vor möglichen Fehlern warnen.

[5] vergleiche [Bergmann 1998b, Kolodner 1996]
[6] vergleiche [Kolodner 1993, Kolodner 1996, Kolodner und Leake 1996]

Wenn auch das in den Fällen enthaltene Wissen den Hauptanteil des in einem fallbasierten Systems enthaltenen Wissens ausmachen soll, so ist Wissen doch in vielfältigerer Form vorhanden. Auch in der Repräsentation (z. B. welche Merkmale verwendet werden), der Ähnlichkeitsbeurteilung und der Adaption ist Wissen enthalten. Dadurch ergibt sich eine hohe Flexibilität fallbasierter Systeme, im Prinzip kann jeder der genannten Bereiche das gesamte Wissen aufnehmen, wenn auch der Schwerpunkt auf dem Fallwissen liegt. Als weiterer Vorteil von CBR–Systemen kommt hinzu, dass das Wissen der Fallbasis, der Ähnlichkeitsbeurteilung und der Adaption auch durch Verfahren des maschinellen Lernens gelernt werden kann.

7.5 Fallrepräsentation

In CBR–Systemen können verschiedene Repräsentationsformen verwendet werden. Allen gemeinsam ist aber, dass in einem Fall das Wissen einer bestimmten früheren Situation gespeichert werden muss. Der Gegenstand und die Aufgabe des CBR–Systems bestimmen:

- das im Fall enthaltene Wissen und dessen Strukturierung
  - Informationen über ein Problem oder eine Situation
  - Informationen über die Lösung
  - eventuell Güteinformationen
- den Umfang des Wissens in einem Fall (seine Größe)
  - vollständige Situationen oder nur Fragmente
  - detaillierte oder abstrakte Beschreibung
- den Wissensrepräsentationsformalismus
  - Attribut-Wert-Paare
  - Semantische Netze
  - Objektorientierte Modelle
  - Graphen, Bäume
  - Prädikatenlogik
  - ⋮

Die Problem- oder Situationsinformation beschreibt ein gelöstes Problem oder eine Situation, die analysiert wurde. Dabei muss die Situationsbeschreibung alle notwendigen Informationen enthalten, um zu entscheiden, ob der Fall in einer neuen Situation anwendbar ist. Wieviel Information dazu nötig ist, hängt vom Lösungsteil ab. Situationsbeschreibungen können das Ziel der Problemlösung (Zielbeschreibungen), Einschränkungen der Lösung (Constraints) und alles, was sonst noch wichtig ist (Merkmale der Situation), enthalten.

Die Lösungsinformation enthält alle Informationen, die die Lösung des Problems hinreichend genau beschreiben. Weiterhin sind Informationen enthalten, die bei der Anpassung der Lösung helfen. Mögliche Komponenten der Lösungsinformation sind die Lösung selbst (z. B. Klassenbeschreibung, Plan, usw.), ein Lösungsweg (eine Folge von Schritten, mit denen das Problem gelöst werden kann), eine Rechtfertigung für eine getroffene Entscheidung bei der Problemlösung (z. B. Auswahl eines Problemlöseschrittes) oder Lösungsschritte, die versuchsweise durchgeführt wurden und fehlgeschlagen sind.

In der Güteinformation kann eine Rückkopplung mit der realen Welt, z. B. aus der Revise-Phase, abgespeichert sein, die besagt wie gut die Lösung für das Problem war. Die Güteinformation kann Informationen darüber enthalten, ob die Lösung erfolgreich oder fehlerhaft war, welcher Aufwand für die Lösung notwendig war, und welche Kosten entstanden sind, sowie welche Akzeptanz die Lösung beim Nutzer gefunden hat.

Die Benutzerunterstützung bei der Auswahl eines Data Mining Verfahrens ist eine Klassifikation. Für eine solche Aufgabenstellung sind die beiden Repräsentationsformen Attribut-Wert-Paare und die objektorientierte Repräsentation gut geeignet. Daher soll von den einzelnen Repräsentationsformalismen hier nur auf diese beiden eingegangen werden. Die anderen Formen werden hauptsächlich für Planungsaufgaben verwendet.

7.5.1 Attribut-Wert-Repräsentationen

Ein Fall wird durch Attribut-Wert-Paare repräsentiert. Die Menge der Attribute kann dabei für alle Fälle fest vorgegeben sein oder von Fall zu Fall variieren. Jedem Attribut ist ein Wertebereich (Typ) zugeordnet:

- Attribute: $A_1, \ldots, A_n$
- Wertebereiche (Typen): $T_1, \ldots, T_n$
- Werte (Attributausprägungen): $a_1 \in T_1, \ldots, a_n \in T_n$
- Bei fest vorgegebener Attributmenge:
  Fall ist ein Vektor: $F = (a_1, \ldots, a_n) \in T_1 \times \ldots \times T_n$
  zur Repräsentation von unbekannten Attributwerten Einführung eines speziellen Symbols: $''unknown''$
  damit: $a_i \in T_i \cup \{unknown\}$
- Bei variabler Attributmenge:
  Fall ist eine Menge: $F = \{A_1 = a_1, \ldots, A_n = a_n\}$
  mit $a_1 \in T_1, \ldots, a_n \in T_n$
  Attribute, die nicht vorkommen, sind unbekannt[7]

[7] In vielen CBR–Systemen wird zwischen unbekannten (unknown) und nicht definierten (undefined) Attributwerten unterschieden.

Die Attribute der Situationsbeschreibung sollten möglichst so gewählt werden, dass sie unabhängige Merkmale repräsentieren, was aber nicht immer möglich ist. Zur Vollständigkeit der Beschreibung ist zu sagen, dass nur aufgrund der Attribute entscheidbar sein muss, ob ein Fall und eine neue Situation ähnlich sind, d. h. ob der Fall anwendbar ist. Gleichzeitig sollte die Menge der Attribute aber minimal sein, d. h. nur solche Attribute aufgenommen werden, die auch tatsächlich zur Ähnlichkeitsbeurteilung beitragen. Die Attribute der Lösungsbeschreibung müssen alle Informationen enthalten, die zur Reproduktion der Lösung notwendig sind. Jedes gegebenenfalls anzupassende Lösungsmerkmal muss durch ein eigenes Attribut repräsentiert werden. Die Wahl der jeweiligen Typen wird durch die Art der erforderlichen Ähnlichkeitsberechnung bzw. der Lösungsanpassung bestimmt. Im Wesentlichen werden folgende Typen verwendet:

- **Symbole** werden bei einer kleinen Anzahl fester Ausprägungen verwendet und zeichnen sich durch einfache Ähnlichkeitsberechnung und Lösungsanpassung aus.
- **Integer/Real**[8] werden für Mess- oder andere Zahlenwerte verwendet. Auch bei ihnen ist die Ähnlichkeitsberechnung und die Lösungsanpassung einfach.
- **Text** ist geeignet für unstrukturierte Informationen und sehr flexibel. Dafür ist die Ähnlichkeitsberechnung sehr schwierig und eine Lösungsanpassung nicht möglich

Zusammenfassend kann man sagen, dass Attribut-Wert-Repräsentationen folgende Vor- und Nachteile besitzen:

- Vorteile:
  - einfache, leicht zu verstehende Repräsentation
  - einfache und effiziente Ähnlichkeitsbestimmung
  - einfache Speicherung (z. B. in Datenbanken)
  - effizientes Retrieval möglich
- Nachteile:
  - keine strukturelle und relationale Information repräsentierbar
  - keine Reihenfolgeinformation (Aktionsfolge) repräsentierbar

Damit sind Attribut-Wert-Paare geeignet für analytische Aufgaben, insbesondere für die Klassifikation und für große Fallbasen mit relativ wenigen Merkmalen.

[8] In vielen Systemen werden Integer ähnlich wie Symbole verwendet, z. B. als Aufzählungstyp mit fester Ausprägung.

7.5.2 Objektorientierte Repräsentation

Einige der eben beschriebenen Nachteile kann man durch eine objektorientierte Repräsentation vermeiden. Zusammengehörige Attribute werden zu Objekten zusammengefasst. Jedes Objekt wird durch eine feste Menge von Attributen beschrieben und stellt eine eigenständige Einheit dar, die einer Objektklasse angehört. Die Objektklassen sind durch eine Vererbungshierarchie angeordnet, bei der jede Unterklasse die Eigenschaften der Oberklasse erbt. Ein Fall wird durch eine Menge von Objekten beschrieben. Zwischen den Objekten eines Falles bestehen Beziehungen (Relationen). Dabei haben zwei Beziehungen besondere Bedeutung erlangt, für die in Abbildung 7.7 je ein Beispiel angegeben ist:

- **Taxonomische Relationen** (a-kind-of) drücken Abstraktions- bzw. Verfeinerungsbeziehungen zwischen den Objekten der Domäne aus.
- **Kompositionelle Relationen** (is-part-of) drücken die Zusammensetzung eines Objektes aus Teilobjekten aus.

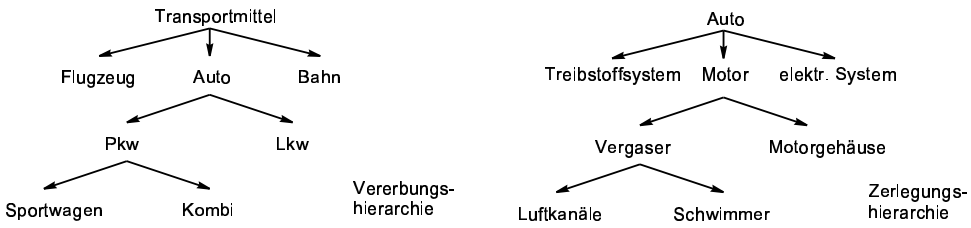

Abbildung 7.7: Beispiel für taxonomische und kompositionelle Relation

Für objektorientierte Repräsentationen kann man folgende Vor- und Nachteile anführen:

- Vorteile
  - strukturiertere und "natürlichere" Fallrepräsentation
  - strukturelle und relationale Informationen sind repräsentierbar
  - kompaktere Speicherungen als mit Attribut-Wert-Paaren möglich
  - Strukturinformation kann für die Ähnlichkeitsberechnung und die Lösungsanpassung genutzt werden.
- Nachteile
  - Ähnlichkeitsberechnung und Retrieval aufwendiger als bei Attribut-Wert-Paaren
  - keine Reihenfolgeinformation (Aktionsfolge) repräsentierbar.

Aus den genannten Eigenschaften folgt, dass diese Repräsentationsform für analytische Aufgaben, insbesondere bei komplexen Fallstrukturen und für synthetische Aufgaben im Bereich von Konfiguration und Design geeignet ist. Für Aufgaben der Planung ist auch diese Repräsentationsform ungeeignet.

Für Planungsaufgaben werden Graphen oder prädikatenlogische Repräsentationsformen verwendet, auf die in dieser Arbeit aber nicht eingegangen wird.

7.6 Ähnlichkeitsbestimmung

Ein zentraler Begriff beim fallbasierten Schließen ist die Ähnlichkeit, die zwischen zwei Problemsituationen A und A' besteht. Die Auswahl der Fälle im Retrieve-Schritt erfolgt auf der Grundlage der Ähnlichkeit zu einem neuen Fall, der Problemsituation. Formale Berechnungen (Ähnlichkeitsmaße) sind relativ leicht zu implementieren, sie reichen aber oft nicht aus, die Realität ausreichend zu modellieren. Ein menschlicher Experte benötigt meist auch ein umfangreiches Fachwissen, um seine Erfahrungen (erinnerte Fälle) richtig zu beurteilen. An diesem Problem wird gegenwärtig weltweit gearbeitet.

Es soll zunächst kurz die inhaltliche Bedeutung der Ähnlichkeit betrachtet werden, bevor auf die Berechnung von Ähnlichkeit eingegangen wird.

7.6.1 Inhaltliche Bedeutung der Ähnlichkeit

Ähnlichkeit bezieht sich immer auf einen speziellen **Aspekt** oder einen bestimmten Zweck, absolute Ähnlichkeit gibt es nicht[9]. Zum Beispiel sind sich im Renngeschehen zwei Autos ähnlich, wenn sie die gleiche Höchstgeschwindigkeit erreichen, während für einen Käufer zwei Autos ähnlich sein können, die ungefähr gleich viel kosten. Damit steht Ähnlichkeit auch immer im Zusammenhang mit Abstraktion, die Abstraktion wählt einen bestimmten Aspekt aus (Ähnlichkeit in Bezug auf was, z. B. Höchstgeschwindigkeit).

Ähnlichkeit ist **nicht notwendigerweise transitiv**. Das sei an zwei Beispielen erläutert:

- Ähnlichkeit zwischen verschiedenen Aspekten:
  - Berlin und London sind ähnlich, weil beide Hauptstädte eines Landes sind.
  - Berlin und Greifswald sind ähnlich, weil sie in Deutschland liegen.
  - Aber: London und Greifswald sind nicht ähnlich.
- Eigenschaft nicht transitiv übertragbar:

[9] vergleiche [Kolodner und Leake 1996]

- 10 EURO ist ähnlich zu 12 EURO
- 12 EURO ist ähnlich zu 14 EURO
- ...
- 1000 EURO ist ähnlich zu 1002 EURO
- Aber: 10 EURO ist nicht ähnlich zu 1002 EURO

Ähnlichkeit ist **nicht notwendigerweise symmetrisch**. Zum Beispiel bedeutet die Aussage "das Schiff bewegt sich wie eine Schnecke", dass das Schiff in Bezug auf eine bestimmte Eigenschaft (hier z. B. Geschwindigkeit) ähnlich einer Schnecke (Prototyp) ist. Die umgekehrte Aussage "die Schnecke bewegt sich wie ein Schiff" ist nicht typisch, das Schiff ist kein Prototyp für Geschwindigkeit.

Der Zweck der Ähnlichkeit beim fallbasierten Schließen ist die Auswahl einer Lösung, die sich leicht auf das aktuelle Problem übertragen lässt. Damit wird Ähnlichkeit gleichgesetzt mit **Nützlichkeit**. Es ist aber zu beachten, dass Nützlichkeit ein *posteriori* Kriterium ist, über die Nützlichkeit kann man in der Regel erst nach der Problemlösung entscheiden. Die Ähnlichkeit auf einer Problemstellung ist ein *apriori* Kriterium, über die Ähnlichkeit muss vor der Problemlösung entschieden werden. Ziel muss es daher sein, dass die Ähnlichkeit die Nützlichkeit möglichst gut approximiert.

7.6.2 Berechnung der Ähnlichkeit

Die Idee ist, Ähnlichkeit numerisch zu modellieren.

Definition 7.1 (Ähnlichkeitsmaß:) *Ein Ähnlichkeitsmaß auf einer Menge M ist eine reellwertige Funktion $sim : M^2 \to [0, 1]$ mit*

1. $sim(x, x) = 1 \qquad \forall x \in M \qquad$ *(Reflexivität)*
2. $sim(x, y) = sim(y, x) \quad \forall x, y \in M \quad$ *(Symmetrie)*

Damit ist neben einer ordinalen Information auch eine quantitative Aussage über den Grad der Ähnlichkeit möglich. Oft wird anstelle des Ähnlichkeitsmaßes ein Distanzmaß bzw. eine Distanzfunktion verwendet. Wenn man alle Elemente in eine mathematische Struktur, den metrischen Raum, einbettet, kann man die Ähnlichkeit zweier Objekte mit einem geometrischen Abstand identifizieren. Die Struktur enthält eine Menge von Objekten X , für die eine Metrik d existiert:

1. $d(x, y) = 1 \iff x = y$
2. $d(x, y) = d(y, x) \qquad \forall x, y \in X$
3. $d(x, z) \leq d(x, y) + d(y, z) \quad \forall x, y, z \in X$

Eine Unterklasse der Metriken entspricht genau den Ähnlichkeitsfunktionen, für die sich das Distanzmaß in ein kompatibles Ähnlichkeitsmaß transformieren lässt.[10]

In der Praxis werden einige Ähnlichkeits- bzw. Distanzmaße recht häufig verwendet, die hier vorgestellt werden sollen.

Für **binäre Merkmale** wird meist der Hammingabstand benutzt:

$$\begin{aligned} x &= (x_1, \ldots, x_n) \qquad x_i \in \{0,1\} \\ y &= (y_1, \ldots, y_n) \qquad y_i \in \{0,1\} \\ H(x,y) &= n - \sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} (1-x_i)(1-y_i) \end{aligned} \tag{7.1}$$

Von dieser einfachen Form existieren verschiedene Abwandlungen, um z. B. einzelne Attribute unterschiedlich zu gewichten oder die Gleichheit bzw. die Unterschiedlichkeit von Attributen mehr zu betonen.

Bei **reellwertigen Attributen** kann man den Hammingabstand zur City-Block-Metrik verallgemeinern, andere Abstandsmaße sind der euklidische und der gewichtete euklidische Abstand.

Die bisher beschriebenen Maße setzen voraus, dass alle Attribute den gleichen Typ haben. Das ist aber oft nicht der Fall. Um bei **beliebigen Attributtypen** die Ähnlichkeit zu berechnen, wird für jeden Typ ein eigenes Ähnlichkeitsmaß definiert. Damit wird für jedes Attribut die lokale Ähnlichkeit berechnet und anschließend mit einer Amalgierungsfunktion die globale Ähnlichkeit zwischen zwei Fällen berechnet.

- **Lokale Ähnlichkeit** $sim_{Ai}(x_i, y_i) : T_i \times T_i \to [0,1]$
- **Globale Ähnlichkeit**
  - $sim(x,y) = F(sim_{A1}(x_1, y_1), \ldots, sim_{An}(x_n, y_n))$
  - $F : [0,1]^n \to [0,1]$ ist eine Amalgierungsfunktion
  - Für F wird üblicherweise gefordert
    - F ist monoton in jedem Argument
    - $F(0, \ldots, 0) = 0$
    - $F(1, \ldots, 1) = 1$

Einige Beispiele der für F verwendeten Funktionen sind:

$$\text{Gewichtetes Mittel} \qquad F(s_1, \ldots, s_n) = \sum_{i=1}^{n} w_i s_i \quad \text{mit} \quad \sum_{i=1}^{n} w_i = 1 \tag{7.2}$$

[10] Das hier vorgestellte Ähnlichkeitsmaß kann in vielen mathematischen und naturwissenschaftlichen Anwendungen verwendet werden. Insbesondere aus der weiter vorn beschriebenen inhaltlichen Bedeutung der Ähnlichkeit lässt sich die Eigenschaft der Symmetrie in vielen Anwendungen nicht halten, so dass CBR–Systeme oft unsymmetrische Ähnlichkeitsmaße anbieten.

$$\text{Verallgemeinerung} \qquad F(s_1,\ldots,s_n) = \sqrt[\alpha]{\sum_{i=1}^{n} w_i(s_i^\alpha)} \tag{7.3}$$

$$\text{mit} \quad \sum_{i=1}^{n} w_i = 1 \quad \text{und} \quad \alpha \in \mathbb{R}^+$$

$$\text{Maximum} \qquad F(s_1,\ldots,s_n) = max(s_1,\ldots,s_n) \tag{7.4}$$

$$\text{Minimum} \qquad F(s_1,\ldots,s_n) = min(s_1,\ldots,s_n) \tag{7.5}$$

$$\text{k-Maximum} \qquad F(s_1,\ldots,s_n) = s_{ik} \text{ mit } s_{i1} \geq s_{i2} \geq \ldots \geq s_{in} \tag{7.6}$$

$$\text{k-Minimum} \qquad F(s_1,\ldots,s_n) = s_{ik} \text{ mit } s_{i1} \leq s_{i2} \leq \ldots \leq s_{in} \tag{7.7}$$

Um die **lokale Ähnlichkeit** der Attribute zu berechnen, sind für die einzelnen Attributtypen folgende Ähnlichkeitsmaße gebräuchlich:

Ungeordnete Symbole: Für ungeordnete Symbole werden häufig Ähnlichkeitstabellen verwendet, in denen die Ähnlichkeit zwischen zwei Werten eingetragen wird.
$sim_A(x,y) = s[x,y]$ Symboltyp $T_A = \{v_1,\ldots,v_k\}$

$s[x,y]$	v_1	v_2	$\ldots$	v_k
v_1	$s[1,1]$	$s[1,2]$		$s[1,k]$
v_2	$s[2,1]$	$s[2,2]$		$s[2,k]$
$\ldots$				
v_k	$s[k,1]$	$s[k,2]$		$s[k,k]$

Bei reflexiven Ähnlichkeitsmaßen sind die Einträge in der Diagonalen gleich 1, und bei symmetrischen Ähnlichkeitsmaßen entspricht die obere Dreiecksmatrix der unteren.

Integer / Real: Hier wird die Ähnlichkeit häufig auf der Grundlage der Differenz beider Werte berechnet, für
linear skalierte Wertebereiche: $sim_A(x,y) = f(x,y)$ und für
exponentiell skalierte Wertebereiche: $sim_A(x,y) = f(log(x), log(y))$
Für f wird in der Regel gefordert:

- $F : \mathbb{R} \rightarrow [0,1]$ oder $\mathbb{Z} \rightarrow [0,1]$
- $f(0) = 1$ (Reflexivität)
- $f(x)$: monoton fallend für $x > 0$ und monoton steigend für $x < 0$

Einige Beispiele sind in Abbildung 7.8 dargestellt.

Eine oft verwendete Funktion ist:
$$sim(x,y) = \frac{|x-y|}{v_{max}-v_{min}} \quad \text{für} \quad x,y \in [v_{min}, v_{max}]$$
In vielen CBR–Systemen können Integer auch als Aufzählungstyp ähnlich wie geordnete Symbole verwendet werden.

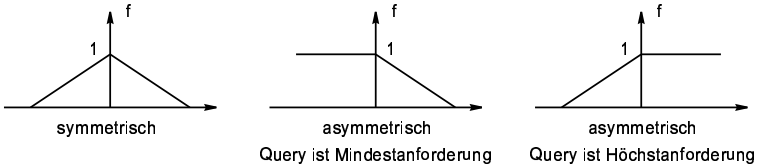

Abbildung 7.8: Lokale Ähnlichkeitsmaße für reellwertige Attribute

Geordnete Symbole: Bei geordneten Symbolen wird zusätzlich zum Wertebereich eine Ordnung angegeben, z. B. für die qualitativen Werte

$\{klein, mittel, gross\}$ die Ordnung $klein < mittel < gross$.

Es können dann dieselben Ähnlichkeitsmaße wie für Integer verwendet werden, indem jedem Symbol ein Integerwert derart zugeordnet wird, so dass die Ordnung erhalten bleibt, für unser Beispiel wäre folgende Zuordnung möglich:

$$\begin{aligned} klein &\rightarrow 1 \\ mittel &\rightarrow 2 \\ gross &\rightarrow 3 \end{aligned}$$

Taxonomien: Wenn sich die betrachteten Objekte in einer Taxonomie oder einem Begriffverbund anordnen lassen, ähnlich wie in Abbildung 7.7 dargestellt, so kann man die Beziehungen in dieser Baumstruktur auch für die Berechnung der Ähnlichkeit benutzen. Dazu werden jedem inneren Knoten Ähnlichkeitswerte derart zugewiesen, dass die Ähnlichkeitswerte für Nachfolgeknoten immer größer werden, wie in Abbildung 7.9 dargestellt.

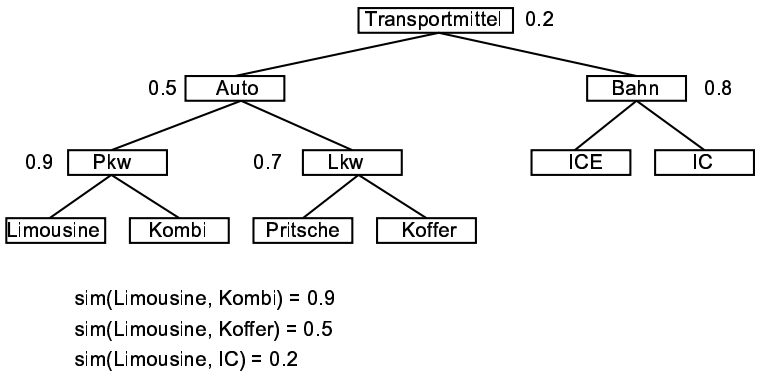

Abbildung 7.9: Lokale Ähnlichkeitsmaße einer Taxonomie

Die Ähnlichkeit zwischen zwei Blattknoten berechnet sich durch den Ähnlichkeitswert des tiefsten gemeinsamen Vorgängers. Je nachdem welche Be-

deutung die inneren Knoten besitzen[11], kommen verschiedene Arten der Ähnlichkeitsberechnung zum Einsatz. Eine detaillierte Darstellung kann in [Bergmann 1998a] nachgelesen werden.

7.6.2.1 Unbekannte Attributausprägungen

Wenn ein Attributwert nicht bekannt ist, kann das unterschiedliche Auswirkungen auf die Ähnlichkeit haben. Man unterscheidet drei Strategien, die zum Einsatz kommen:

Optimistische Strategie: Es wird angenommen, dass der unbekannte Wert vermutlich für die Ähnlichkeit spricht, und man drückt dies durch eine lokale Ähnlichkeit von 1 aus.

Pessimistische Strategie: Es wird angenommen, dass der unbekannte Wert vermutlich gegen die Ähnlichkeit spricht, und man drückt dies durch eine lokale Ähnlichkeit von 0 aus.

Strategie des Erwartungswertes: Man nimmt den Erwartungswert für die lokale Ähnlichkeit auf der Grundlage der bekannten Informationen an.

7.6.2.2 Objektorientierte Repräsentationen

Die bisherigen Betrachtungen zur Ähnlichkeitsberechnung beziehen sich auf Attribut-Wert-Paare. Für objektorientierte Repräsentationen kann man das derart erweitern, dass Objekte wie Attribut-Wert-Paare betrachtet werden und die Ähnlichkeit damit zu einer Eigenschaft der Objektklasse wird:

- Auf der Slotebene (Vergleich einzelner Attributwerte) wird ein lokales Ähnlichkeitsmaß verwendet.
- Auf der Klassenebene (Vergleich von Objekten) erfolgt die Bestimmung der Objektähnlichkeit durch Amalgierung
  - der lokalen Ähnlichkeiten von getypten Slots und
  - der Objektähnlichkeiten der Unterobjekte bei relationalen Slots.

In [Bergmann und Stahl 1998] sind weiterführende Betrachtungen zur objektorientierten Repräsentation und der damit zusammenhängenden Ähnlichkeitsberechnung enthalten.

[11] Ein innerer Knoten steht für die Menge der darunter liegenden Blattknoten. Die Bedeutung der inneren Knoten für einen gespeicherten Fall und für das aktuelle Problem kann sein:

Unbekannt Es liegt ein Element der Menge vor, welcher Ausprägung ist aber unbekannt.

Beliebig Es ist egal, welches Element der Menge betrachtet wird.

7.6.3 Schlussbemerkungen

Die Ähnlichkeit ist ein zentraler Aspekt bei der Entwicklung fallbasierter Systeme. Es existieren eine Reihe von domänenspezifischen Verfahren zur Berechnung der Ähnlichkeit. Hier konnten nur allgemeine Ansätze erläutert werden. Eine weitere Eigenschaft ist z. B., dass die Ähnlichkeit zwischen dem Problem im abgespeicherten Fall und dem aktuellen Problem von der Lösung im abgespeicherten Fall abhängig sein kann. Je nach Lösung können einzelne Attribute unterschiedliche Relevanz für die Ähnlichkeit besitzen.

7.7 Anpassung der Lösung

Weil eine alte Situation sich niemals vollständig mit einer neuen deckt, ist es erforderlich, dass ein CBR–System in der Lage ist, die alte Lösung so anzupassen, dass sie in der neuen Situation angewendet werden kann. Die Anpassung kann dabei vom einfachen Austausch einer Komponente der Lösung durch eine andere bis hin zu einer komplexen Veränderung der Struktur der Lösung reichen. Es werden zwei wesentliche Arten der Lösungsanpassung unterschieden:

Transformationsbasierte Lösungsanpassung Bei der transformationsbasierten Lösungsanpassung erfolgt die Anpassung durch eine feste Menge von Adaptionsregeln und Adaptionsoperatoren. Bei der *substitutional Adaption* werden einzelne Teile der Lösung oder einzelne Parameter verändert, eingefügt oder gelöscht. Die *structural Adaption* reorganisiert die Lösung durch Einfügen oder Löschen von Objekten.

Generative Lösungsanpassung Die generative Lösungsanpassung geht von der Annahme aus, dass es einen Problemlöser gibt, der in der Lage ist, jedes Problem auch ohne Fälle zu lösen, d. h. der Problemlöser besitzt Domänenwissen und einen Inferenzmechanismus. Die Fälle werden dazu verwendet,

- um die Lösung schneller zu finden,
- um eine Lösung zu bekommen, die ähnlich zu einer bekannten Lösung ist oder
- um eine Lösung durch möglichst wenige Änderungen einer bestehenden Lösung zu erhalten.

Die generative Lösungsanpassung kombiniert damit Fallwissen mit einem Problemlöser. Eine Anwendung kann so aussehen, dass Teile des Lösungsweges, die für das aktuelle Problem anwendbar sind, übertragen und gegebenenfalls Teillösungen von verschiedenen Fällen kombiniert werden. Der Problemlöser wird dazu benutzt, um "Lücken" in der Lösung auszufüllen.

Tabelle 7.1 stellt einige Eigenschaften der generativen und der transformationsbasierten Lösungsanpassung gegenüber.

generativ	**transformationsbasiert**
• Problemlösewissen ist erforderlich • Fälle müssen den Lösungsweg beinhalten	• kein Problemlösewissen erforderlich • Wissen über die Anpassung von Lösungen ist erforderlich • Erfahrungen brauchen den Lösungsweg nicht zu enthalten
• Falls die Problemlösewissen korrekt ist, ist die Korrektheit der Lösung garantiert	• Korrektheit der Lösung ist nicht garantiert
• In der Regel: Erfahrungen führen zur Effizienzsteigerung	• In der Regel: Erfahrungen führen zur Kompetenzsteigerung

Tabelle 7.1: transformationsbasiert vs. generativ

7.8 Lernphase

In der Lernphase ist es möglich, neues Wissen in das System einzubringen. Dabei werden hauptsächlich zwei Ziele verfolgt:

- Die Verbesserung der Kompetenz des Systems.
- Die Verbesserung bzw. Bewahrung der Effizienz.

Die einfachste Form des Lernens ist sicherlich das Einbringen neuer Fälle in die Fallbasis. Ein fallbasiertes System kann sich effizienter an alte Fälle erinnern und diese anpassen als die Lösung eines Problemes neu zu generieren. Wenn ein Problem auf eine neue Art gelöst wurde, wenn eine neue Methode der Lösungsanpassung verwendet wurde oder wenn zur Lösung des Problemes die Lösungen mehrerer Fälle kombiniert wurden, ist es daher sinnvoll, diese neue Lösung in das System aufzunehmen. Dadurch kann die Fallbasis im Laufe der Zeit stark anwachsen, so dass es erforderlich wird, sie zu reorganisieren und eventuell überflüssige Fälle zu entfernen. Die Kompetenz des fallbasierten Systems kann aber auch verbessert werden, indem die Ähnlichkeitsbeurteilung und die Lösungsanpassung verbessert werden. Eine Form der Verbesserung der Ähnlichkeitsbeurteilung ist das Lernen von Attributgewichten. Dadurch wird die Relevanz einzelner Attribute für die Lösung verändert, was z. B. durch fallspezifische Gewichte möglich ist.

7.9 Zusammenfassung

Fallbasiertes Schließen zeichnet sich hauptsächlich dadurch aus, dass alte Lösungen so auf ein neues Problem angepasst werden, dass alte Fälle zur Auseinandersetzung mit neuen Situationen benutzt werden oder dass alte Fälle zur Beurteilung neuer Lösungen herangezogen werden. Ein CBR–System lernt im Laufe seiner Benutzung dazu. Es kann durch das Abspeichern der Erfahrungen, die während der Problemlösung gewonnen wurden, immer effizienter und kompetenter werden. Im Unterschied zu anderen Methoden der künstlichen Intelligenz betrachtet das fallbasierte Schließen die Problemlösung als einen Prozess des Erinnerns an einen bzw. wenige konkrete Fälle und ermittelt seine Entscheidung durch den Vergleich der neuen Situation mit einer alten.

Die Qualität eines CBR–Systems hängt ab von der Erfahrung, die es gespeichert hat, von seiner Fähigkeit, eine neue Situation in den Kategorien der alten, abgespeicherten Erfahrungen zu verstehen und von seinem Geschick bei der Anpassung der Lösung.

Fallbasiertes Schließen kann in einem weiten Bereich täglicher Anwendungen eingesetzt werden, von wissensintensiven Problemen mit komplexer Lösungsanpassung bis hin zu Problemen, die weniger Wissen erfordern und wo die abgespeicherten Fälle das einzige Wissen darstellen. CBR–Systeme haben Vorteile, wenn es darum geht, schnell eine Lösung zu finden, wenn die Domäne nicht vollständig erforscht ist, oder wenn algorithmische Lösungen nicht bekannt sind. Nachteile ergeben sich hauptsächlich dann, wenn die Fälle schlecht genutzt werden, z. B. wenn zu sehr auf den Lösungsvorschlag vertraut wird ohne ihn zu bewerten. Fallbasiertes Schließen ist ein natürlicher Weg für Menschen, um Probleme zu lösen und auch eine erfolgversprechende Möglichkeit für Maschinen.

Aufgrund der Vorteile und der speziellen Eigenschaften der Methodenauswahl ist der Einsatz von CBR zur Methodenauswahl vielversprechend.

Kapitel 8

Realisierung von AST

Um den in Kapitel 7 beschriebenen Ansatz umzusetzen, müssen die verschiedenen bisher beschriebenen Aspekte der Benutzerunterstützung im Prozess der Methodenauswahl in ein CBR-Modell abgebildet und eine initiale Fallbasis aufgebaut werden. In den folgenden Abschnitten wird beschrieben, wann eine Methode als anwendbar gilt, wie die Nutzeranforderungen, Methoden und Datencharakteristiken abgebildete werden können und wie eine initiale Fallbasis aufgebaut wurde. Vorher werden aber die eingesetzten Werkzeuge beschrieben. Clementine™ und MLC++ wurden für den Aufbau der Fallbasis benutzt, während CBR Works als Basis für AST eingesetzt wurde.

8.1 CBR–Works

Als Werkzeug zur Entwicklung fallbasierter Anwendungen stand *CBR–Works for Professionals* der Firma TECINNO GmbH zur Verfügung. CBR–Works ist auf mehreren Betriebssystem-Plattformen implementiert und stellt eine ganze Produktfamilie dar. Neben dem hier verwendeten Entwicklungswerkzeug gibt es die Möglichkeit, Client-Server-Anwendungen zu entwickeln, und es existieren Schnittstellen für Intranet- und Internet–Anwendungen.

Mit CBR–Works ist es möglich, fallbasierte Anwendungen zu entwickeln, die eine objektorientierte Repräsentation verwenden. Dem Entwickler werden vordefinierte Typen und Ähnlichkeitsmaße zur Verfügung gestellt, von denen eigene Erweiterungen abgeleitet werden können. Sowohl die verwendeten Klassen (*Concepts*), als auch die Typen können von einer Datenbank übernommen werden. Dafür und zum direkten Import der Fallbasis steht eine ODBC–Schnittstelle zur Verfügung. Der Nutzer der fertigen Anwendung hat die Möglichkeit, bei der Berechnung der Ähnlichkeit einzelne Attribute entsprechend seinen Anforderungen zu gewichten oder Filter zur Einschränkung der Suche zu definieren. Zur Integration in bestehende Produkte kann eine eigene Sprache (*Case Query Language* CQL) verwendet werden.

CBR–Works wird mit einer grafischen Benutzeroberfläche bedient, die dem Entwickler zwei Editoren zum Erstellen des Modells zur Verfügung stellt. Mit dem ersten

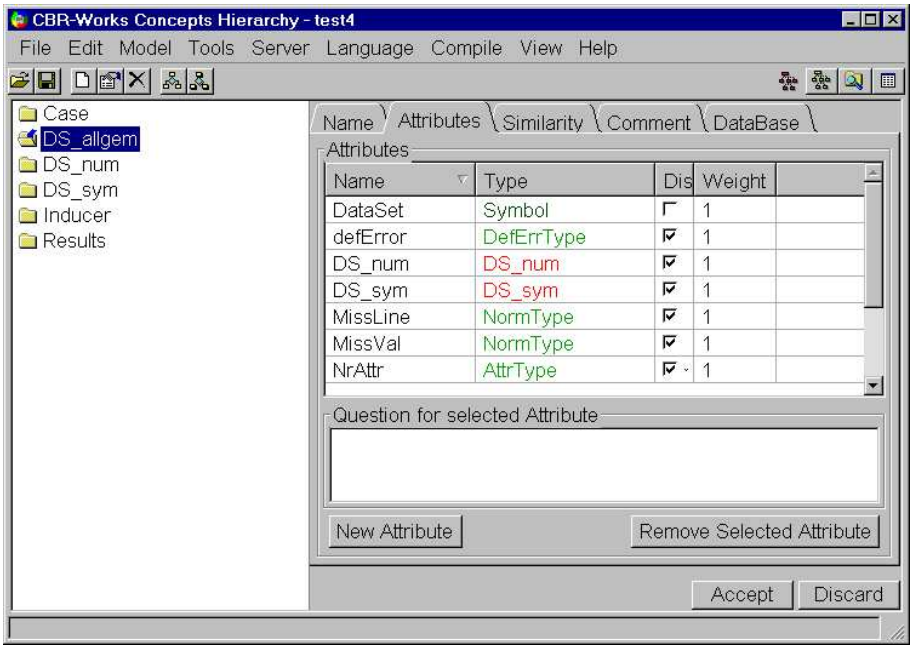

Abbildung 8.1: Editor für Klassen

Editor (Abbildung 8.1) werden die Klassen (*Concepts*) und die Attribute (*Slots*) mit ihren Typen bearbeitet. Dabei ist es sowohl möglich, eine Klassenhierarchie mit Vererbung der Slots aufzubauen, als auch Klassen als Slot einer anderen Klasse zu verwenden. In diesem Editor wird auch eingestellt, wie die globale Ähnlichkeit aus den lokalen Ähnlichkeiten berechnet wird. Weiterhin können die einzelnen Klassen mit den Tabellen einer Datenbank verknüpft werden. Dadurch wird es möglich, die Fallbasis aus einer Datenbank zu laden.

In der Abbildung 8.1 ist auch schon der Aufbau eines Falls ersichtlich. Ein Fall setzt sich zusammen aus einer allgemeinen Beschreibung des Datensatzes, der Charakterisitik der numerischen Werte und der symbolischen Werte. Weiterhin ist in der Klasse *Inducer* die in diesem Fall genutzte Methode hinterlegt, sowie in *Results* die erzielten Ergebnisse mit der Methode. Die Details der Modellierung werden in Abschnitt 8.6 beschrieben.

Zur Bearbeitung der Typen und der mit ihnen verbundenen Ähnlichkeitsmaße dient ein zweiter Editor (Abbildung 8.2). Hier können von den vorgegebenen Typen neue abgeleitet und diesen abgeleiteten Typen eigene Ähnlichkeitsmaße zugewiesen werden. Für die Berechnung der lokalen Ähnlichkeiten stehen, abhängig vom Typ, verschiedene Möglichkeiten zur Verfügung, die in weiten Grenzen vom Entwickler abgewandelt werden können.

In der Abbildung 8.2 ist auch die Taxonomie über die Methoden im Ausschnitt dargestellt, die auch im Abschnitt 8.6 im Detail beschrieben wird. In dem verwendeten CBR–Tool kann für jedes Attribut ein Typ und die Berechnung der Ähnlichkeit definiert werden.

Ein weiterer Editor dient der Bearbeitung der Fallbasis (Abbildung 8.3). Hier kön-

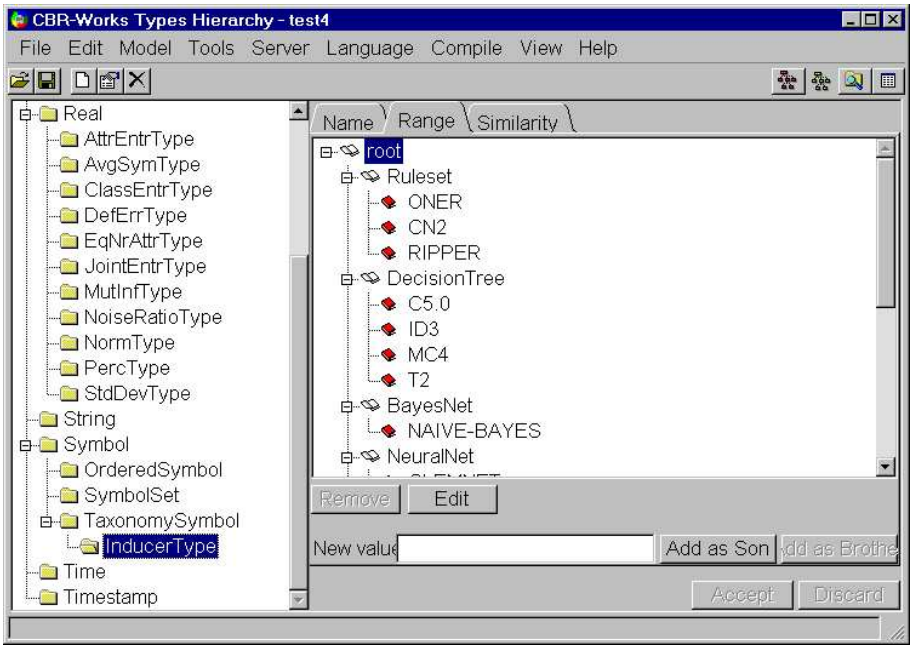

Abbildung 8.2: Editor für Typen und Ähnlichkeitsmaße

nen neue Fälle eingegeben oder die Werte der abgespeicherten Fälle verändert werden. Abhängig vom Typ des zu verändernden Wertes werden verschiedene Editoren aufgerufen, so wird bei Aufzählungs- oder Taxonomietypen eine Liste der verfügbaren Werte oder bei Typen, die Werte aus einem Intervall annehmen können, der Minimal- und Maximalwert angezeigt sowie jeweils ein Auswahl- bzw. Eingabefeld. Um die abgespeicherten Fälle zu editieren, müssen sie zuvor in einen Puffer (*Case Buffer*) kopiert werden. Dadurch wird ein versehentliches Verändern der Fallbasis verhindert. Bei einer umfangreichen Fallbasis bietet es sich allerdings an, die Fälle aus einer Datenbank zu übernehmen und durch eine entsprechende Vorverarbeitung der Fälle sicherzustellen, dass korrekte Werte eingelesen werden.

Ein eigenständiger Dialog (siehe Abbildung 8.4) ermöglicht das eigentliche *Retrieval*. Dabei kann der neue Fall[1] entweder durch direkte Eingabe der Werte in eine Tabelle oder durch die Beantwortung aufeinanderfolgender Fragen zu den Werten spezifiziert werden. Die Reihenfolge der Fragen kann von dem zu erwartenden Informationsgewinn gesteuert werden. Die Editoren für die einzelnen Werte sind denen für die Eingabe der Fallbasis ähnlich. Zusätzlich kann für jedes Attribut die Wichtigkeit (*Very High, High, Medium, Low, Very Low*) und ein Filter (*No Filter, Equal, Not Equal, Less, Less or Equal, Greater, Greater or Equal*) festgelegt werden. Nach der Abfrage der Fallbasis wird eine Liste der ähnlichsten Fälle[2], geordnet nach der berechneten Ähnlichkeit, ausgegeben. Da immer nur zwei Fälle angezeigt werden, ist eine Schalterleiste zur Navigation durch die Liste vorhanden.

[1] Im folgenden Text wird der neue Fall auch als Abfrage bezeichnet.

[2] In den generellen Einstellungen kann die Anzahl der ermittelten ähnlichen Fälle (Standard: 10, Maximum: 30) eingestellt werden.

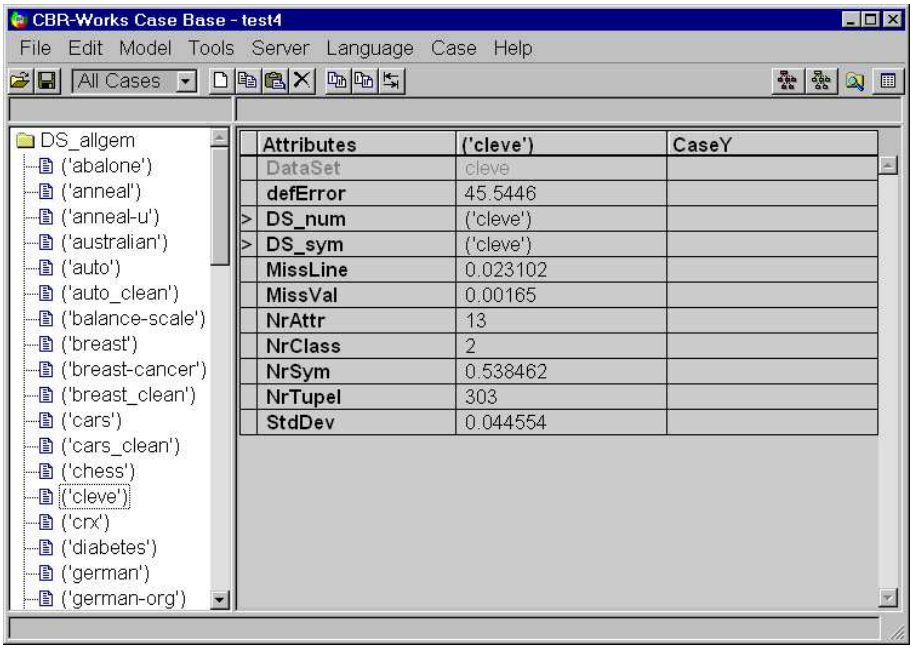

Abbildung 8.3: Editor für die Fallbasis

Ähnlichkeitsberechnung in CBR–Works Die Art der Berechnung der globalen Ähnlichkeit kann für jede Klasse getrennt eingestellt werden. Dabei stehen folgende Möglichkeiten zur Berechnung der Ähnlichkeit von zwei Instanzen einer Klasse zur Verfügung:

Average: Von den lokalen Ähnlichkeiten der Attribute, die zur Berechnung der Ähnlichkeit beitragen, wird der Durchschnitt (gewichtetes Mittel) berechnet.

Minimum: Die niedrigste lokale Ähnlichkeit zwischen zwei Attributen bestimmt die Ähnlichkeit der Instanzen.

Maximum: Die höchste lokale Ähnlichkeit zwischen zwei Attributen bestimmt die Ähnlichkeit der Instanzen.

Euclidean: Die Ähnlichkeit der Instanzen wird geometrisch interpretiert ("Abstand" zwischen zwei Instanzen).

Wenn eine Klasse als Slot einer übergeordneten Klasse fungiert (*Part of Beziehung*), so wird die Ähnlichkeit zwischen den Instanzen dieser Klasse entsprechend der eingestellten Art berechnet und der so ermittelte Wert ist ein lokales Ähnlichkeitsmaß der übergeordneten Klasse.

Die Berechnung der lokalen Ähnlichkeit kann, abhängig vom Typ, mit verschiedenen Optionen gesteuert werden. Für fast alle Typen lassen sich, neben der Standardmethode, folgende generelle Methoden einstellen:

Tabelle: Für aufzählbare Typen kann die Ähnlichkeit zwischen je zwei Werten in einer Tabelle explizit festgelegt werden.

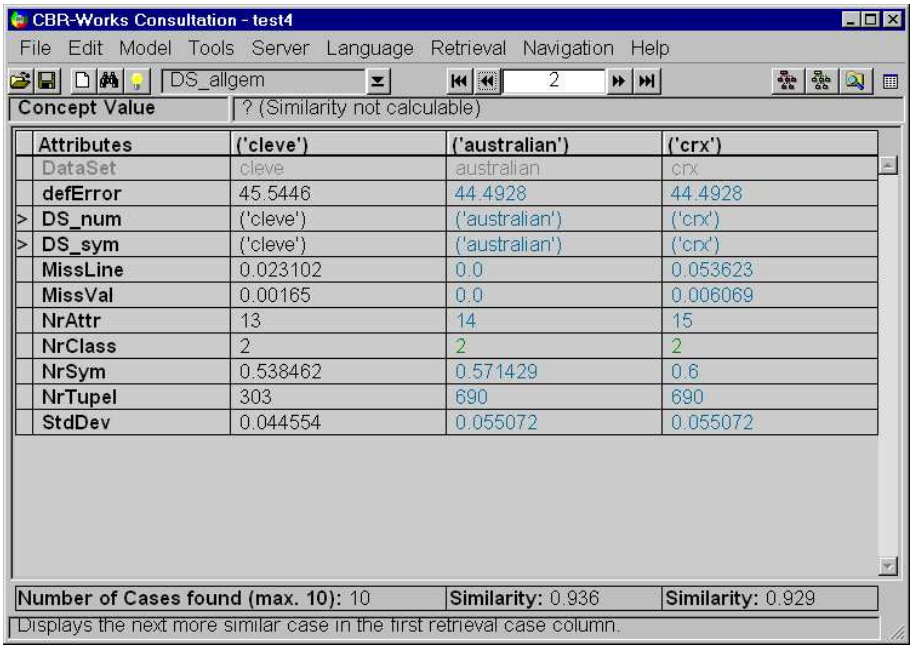

Abbildung 8.4: Standardansicht für Vergleichsanfragen

Programm: Für komplexere Ähnlichkeiten ist es möglich, in der Programmiersprache Smalltalk einen Algorithmus zur Berechnung anzugeben.

Symmetrie: Außer im Programm-Modus kann für jedes Ähnlichkeitsmaß eingestellt werden, ob es symmetrisch oder asymmetrisch sein soll. Davon abhängig sind zwei unterschiedliche Distanzen definiert:

$$\begin{array}{ll} symmetrisch & |(case\,value) - (query\,value)| \\ asymmetrisch & (case\,value) - (query\,value) \end{array}$$

Alle Datentypen enthalten zwei besondere Werte (*special value*) für unbekannte (*unknown*) bzw. nicht definierte (*undefined*) Daten. Bei der Berechnung der Ähnlichkeit nehmen sie eine Sonderstellung ein:

***undefined*:** Tritt in der Abfrage oder im abgespeicherten Fall mindestens ein Wert auf der nicht definiert ist, so wird der Slot bei der Ähnlichkeitsberechnung nicht berücksichtigt[3].

***unknown*:** Tritt in der Abfrage oder im abgespeicherten Fall mindestens ein Wert auf der nicht bekannt ist, und sind beide Werte definiert[4], so wird die Ähnlichkeit für diesen Slot auf 0 gesetzt.

[3]Für diesen Slot wird keine Ähnlichkeit berechnet und der Quotient zur Durchschnittsberechnung für die globale Ähnlichkeit wird um eins verringert.

[4]In dem Fall, dass einer der Werte undefiniert und der andere unbekannt ist, wird der Slot bei der Ähnlichkeitsberechnung nicht berücksichtigt.

Von den lokalen Ähnlichkeitsmaßen sollen hier nur solche vorgestellt werden, die bei den in dieser Arbeit verwendeten Typen benutzt werden.

Für **numerische Typen** kann zwischen einem Standardmodus und einem erweiterten Modus gewählt werden, die sich in der Anzahl der bereitgestellten Optionen unterscheiden. Im Standardmodus kann die Ähnlichkeit entweder mit einer polynominalen Funktion mit zwischen 0 und 10 einstellbarem Gradienten oder mit einer Schrittfunktion mit hartem oder weichem Übergang (signoide Funktion) berechnet werden. Außerdem kann mit einer *Perfect*-Option eingestellt werden, dass die Ähnlichkeit den Wert 1.0 annimmt, wenn der Wert des gespeicherten Falles größer (*more is perfect*) bzw. kleiner (*less is perfect*) als der Wert der Abfrage ist. Mit den zusätzlich im erweiterten Modus angebotenen Möglichkeiten lassen sich Ähnlichkeitspunkte[5] festlegen und die Skalierung von linear auf logarithmisch[6] ändern.

Bei **geordneten symbolischen Typen** wird jeder Ausprägung eine Ordnungszahl zugeordnet. Damit stehen im Standard- und im erweiterten Modus ähnliche Funktionen wie für numerische Typen zur Verfügung. Zusätzlich kann die Skalierung durch eine explizite Zuordnung der Ordnungszahlen beeinflusst werden. Dabei gilt es zu beachten, dass die Ordnungszahlen nur so zugeordnet werden können, dass die ursprüngliche Ordnung erhalten bleibt.

Innerhalb einer **Taxonomie** ist die Zuordnung der einzelnen Werte bereits ein Ausdruck der Ähnlichkeit. Im Standardmodus lässt sich festlegen, ob innere Knoten erlaubt sind oder nicht und wenn ja, welche Semantik ihnen zugeordnet wird[7]. Im erweiterten Modus kann diese Semantik detaillierter definiert werden.

Hier konnte nur ein grober Überblick über die Möglichkeiten von CBR–Works gegeben werden. Eine umfassende Beschreibung ist in [CBR-Works 1998] enthalten.

8.2 MLC++

MLC++ steht für Machine Learning library in C++. MLC++ ist ein System von Hilfsmitteln zur Anwendung von verschiedenen Data Mining Algorithmen sowie eine Klassenbibliothek zur Entwicklung neuer Algorithmen. Das MLC++ -Projekt begann 1993 an der Stanford Universität und ist Public Domain Software. Seit 1995 wird das Projekt bei Silicon Graphics weiterentwickelt und liegt in einer Version 2.0 vor. Für Forschungszwecke sind die Quelltexte nach wie vor verfügbar. Eine Beschreibung von MLC++ findet sich in [Kohavi u. a. 1997, Kohavi und Sommerfield 2002], sowie eine umfangreiche Vergleichsstudie der einzelnen Algorithmen.

Der Vorteil von MLC++ besteht darin, dass eine Vielzahl von Algorithmen unterstützt wird, die entweder direkt implementiert sind, oder für die ein externer Aufruf

[5] Ein Ähnlichkeitspunkt ordnet einer Distanz einen Ähnlichkeitswert zu.

[6] Für die Berechnung der Ähnlichkeit wird nicht der Wert selbst sondern der Logarithmus zur Basis 10 verwendet. Diese Möglichkeit steht nur zur Verfügung, wenn der Wertebereich entweder positiv (ausschließlich Null) oder negativ ist.

[7] vergleiche Abschnitt 7.6.2

vorgesehen ist. Dadurch kann der Anwender einen Datensatz mit den unterstützten Algorithmen ausprobieren, ohne sich um die verschiedenen Datenformate oder Aufrufoptionen kümmern zu müssen.

Die folgende Aufstellung gibt einen kurzen Überblick über einige der zur Verfügung stehenden Hilfsmittel:

Inducer: Der Inducer ist ein Tool, das einen vorgegebenen Algorithmus[8] auf einem Datensatz anwendet und folgende Ausgaben erzeugt:

- Anzahl der Tupel im Trainingsset
- Anzahl der Tupel im Testset sowie die Anzahl der davon im Trainingsset enthaltenen (*seen*) und neuen (*unseen*) Tupel
- Anzahl der korrekt und der nicht korrekt klassifizierten Tupel des Testset
- Fehlerrate der neuen Tupel (*Generalisation error*) sowie der bereits im Trainingsset enthaltenen Tupel (*Memorization error*)
- Fehlerrate auf dem Testset insgesamt sowie eine auf früheren Tests beruhende Standardabweichung für die Fehlerrate

Die einzelnen Verfahren lassen sich durch verschiedene Parameter steuern. Außerdem ist es möglich, Vorverarbeitungsschritte wie die Diskretisierung numerischer Attribute oder die Auswahl eines Subsets[9] durchführen zu lassen. Der Inducer passt das Dateiformat des Datensatzes an den verwendeten Algorithmus an und führt auch notwendige Anpassungen wie Diskretisierung numerischer Attribute automatisch durch.

PerfEst: PerfEst ist ein Tool, mit dem die zu erwartende Fehlerrate eines Verfahrens für einen Datensatz bestimmt werden kann.

Info: Mit Hilfe von Info kann man grundlegende statistische Aussagen des verwendeten Datensatzes erhalten. Die Maße entsprechen weitgehend den vom DCT erzeugten einfachen Maßen.

Project: Um ein Subset eines Datensatzes zu erzeugen, kann das Tool Project verwendet werden. Die im Subset enthaltenen Attribute werden vom Nutzer erfragt, alle anderen Attribute werden entfernt. Dabei passt das Tool sowohl den gesamten Datensatz als auch Test- und Trainingsset sowie die Names-Datei automatisch an.

Discretize: Um numerische Attribute zu diskretisieren, stellt **Discretize** verschiedene Methoden zur Verfügung. Auch hier werden, wie schon bei Project, alle zum Datensatz gehörenden Dateien angepasst.

[8] In der Dokumentation zu MLC++ wird ein Algorithmus auch als *Inducer* bezeichnet.

[9] Einzelne oder mehrere Attribute des Datensatzes werden nicht berücksichtigt. Dadurch kann man z. B. bei Nearest Neighbour Verfahren irrelevante Attribute entfernen.

Conv: Einzelne Verfahren können nur numerische Attribute verarbeiten. Das Tool Conv erlaubt es, symbolische Attribute in eine numerische Darstellung zu konvertieren.

GenCVFiles: Um die für die verschiedenen Arten der Validierung notwendige Zerlegung des Datensatzes in Teilmengen zu ermöglichen, wird das Tool GenCVFiles angeboten.

MLC++ ist mehr ein System für die Forschung oder den Entwickler als für einen Endanwender. Für die meisten Algorithmen lässt sich das erzeugte Modell nicht abspeichern, so dass sich zwar ein Datensatz mit verschiedenen Algorithmen untersuchen lässt, die Ergebnisse aber nicht unmittelbar für eine Anwendung nutzbar sind.

Um die von den einzelnen Algorithmen zum Erstellen des Modells und für die Klassifikation der Tupel des Testsets benötigten Zeiten gesondert erfassen zu können, wurden sowohl im Inducer als auch in den einzelnen Verfahren Funktionen zur Protokollierung der aktuellen Rechnerzeit sowie zur Ausgabe der Zeiten durch den Inducer implementiert.

8.3 Nutzerunterstützung

Eine Teilaufgabe des UGM ist es, für ein bereits analysiertes Anwenderproblem und ein daraus abgeleitetes Data Mining Ziel die geeigneten Algorithmen zu bestimmen. Welcher Algorithmus unter den vom Anwender geforderten Bedingungen (zur Realisierung des Data Mining Zieles) für den betreffenden Datensatz der beste ist, ist Gegenstand der Forschung und, für den Bereich der Klassifikationsalgorithmen, Thema der vorliegenden Arbeit. Geht man von der Gesamtmenge der Klassifikationsalgorithmen aus, so wird die Menge der geeigneten Algorithmen als Durchschnittsmenge der Menge der anwendbaren Algorithmen[10] und der Menge der Algorithmen, die die Anforderungen des Nutzers erfüllen, bestimmt (siehe Abbildung 8.5).

8.4 Nutzeranforderungen

Abhängig vom zu realisierenden Data Mining Ziel gibt es bestimmte Anforderungen an den Algorithmus. Das wichtigste Kriterium ist sicherlich die zu erwartende Fehlerrate[11], die die Anwendbarkeit eines Algorithmus bestimmt. Die weiteren Kriterien sollen als Nutzeranforderungen bezeichnet werden. Darunter fallen:

[10] Zum Begriff Anwendbarkeit eines Algorithmus siehe Kapitel 8.5

[11] Oft wird auch die Klassifikationsgenauigkeit verwendet. Beide Werte lassen sich aber leicht ineinander überführen. Während die Fehlerrate den Anteil der falsch klassifizierten Tupel angibt, ist die Klassifikationsgenauigkeit der Anteil der richtig klassifizierten Tupel. Es gilt also: Fehlerrate + Klassifikationsgenauigkeit = 1.

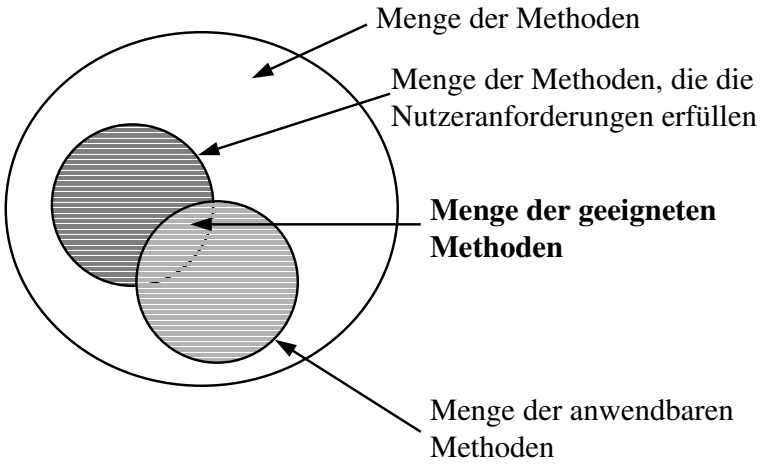

Abbildung 8.5: Zusammenhang der Begriffe *anwendbar* und *geeignet*

Erstellung eines Modells: In bestimmtem Fällen ist der Anwender an einem Modell[12] interessiert, in anderen Fällen ist allein das Ergebnis der Klassifikation interessant. Ob ein Modell erstellt wird oder nicht, ist eine Eigenschaft des verwendeten Algorithmus. Wird ein Modell erstellt, sind weitere Aspekte von Bedeutung:

Interpretierbarkeit: Soll vom Anwender nachvollzogen werden können, wie die Zuordnung eines Tupels zu einer Klasse erfolgt, wird ein interpretierbares Modell benötigt. Gleichzeitig ist der Anwender in der Regel an einem leicht interpretierbaren Modell interessiert. In diesem Sinne stellen z. B. Regelmengen und Entscheidungsbäume interpretierbare Modelle dar, wobei Regelmengen gemeinhin als leichter interpretierbar gelten. Das von einem neuronalen Netz erzeugte Modell gilt als nicht interpretierbar.

Kompaktheit des Modells: Dieses Kriterium wird in der vorliegenden Arbeit nicht zur Auswahl eines Algorithmus verwendet, soll hier aber trotzdem erwähnt werden, da es oft als Eigenschaft eines Modells angegeben wird. Dabei gibt es unterschiedliche Definitionen und Interpretationen, allgemein stellt aber z. B.. eine Regelmenge eine kompaktere Darstellung als ein Entscheidungsbaum dar, wenn man letzteren als eine Menge von Regeln interpretiert.

Trainingszeit: In vielen Anwendungsfällen müssen immer wiederkehrende Klassifikationen nach gleichen Kriterien durchgeführt werden, wobei die

[12] Als Modell soll hier eine Abstraktion von den einzelnen Tupeln des Trainingsset verstanden werden. Ein einfaches Abspeichern der Tupel wie bei Instance Based Verfahren üblich, stellt kein Modell dar, die Topologie und die Gewichte eines neuronalen Netzes dagegen beschreiben ein Modell.

Klassifikationsgenauigkeit entscheidend ist. Die Trainingszeit für das Modell spielt dabei eine untergeordnete Rolle, da ein einmal trainiertes Modell immer wieder verwendet werden kann. Dem stehen zeitkritische Anwendungen gegenüber, die eine kurze Trainingszeit erfordern, dafür aber in Kauf nehmen, nicht die beste Klassifikationsgenauigkeit zu erreichen, d. h., mit einem Algorithmus mit längerer Trainingszeit wäre eventuell eine höhere Klassifikationsgenauigkeit erreichbar[13].

In der vorliegenden Arbeit werden auch Trainingszeiten für Algorithmen angegeben, die kein Modell erzeugen. Meist handelt es sich dabei um Instance Based Verfahren. Um zur Klassifikation eines Tupels schnellen Zugriff auf die Tupel des Testsets zu haben, werden diese oft indiziert und in einer speziellen Struktur abgespeichert. Die dazu benötigte Zeit wird als Trainingszeit bezeichnet.

Klassifikationszeit: Die Zeit, die zur Klassifikation der Tupel des Testsets benötigt wird, wird als Klassifikationszeit (oder Testzeit) bezeichnet. Ähnlich wie schon bei der Trainingszeit ausgeführt, kann der Anwender unterschiedliche Anforderungen an die zur Klassifikation benötigte Zeit stellen. Dabei hängen Trainingszeit, Klassifikationszeit und Klassifikationsgenauigkeit aber nicht notwendig zusammen.

Speicherbedarf: Auch der Speicherbedarf wird in dieser Arbeit nicht zur Auswahl eines Klassifikationsverfahrens verwendet. Mit Speicherbedarf kann sowohl der Bedarf an Hauptspeicher als auch der Festplattenbedarf gemeint sein. Eigentlich sollten Speicherprobleme heute nicht mehr vorkommen. In einigen speziellen Fällen kann es aber notwendig sein, auf den Speicherbedarf zu achten, da oft große Datenmengen im Hauptspeicher gehalten werden, um die Laufzeit zu verringern.

8.5 Anwendbarkeit eines Algorithmus

Die Anwendbarkeit eines Algorithmus ergibt sich aus der Charakteristik des zu untersuchenden Datensatzes und wird wesentlich durch die erreichbare Fehlerrate bestimmt. Die Fehlerrate alleine sagt dabei aber nicht allzu viel aus. Eine Fehlerrate von z. B. 90% mag für einen Datensatz hervorragend sein, für einen anderen Datensatz kann das aber inakzeptabel sein. Um die einzelnen Algorithmen untereinander vergleichen zu können, bedarf es einer Normalisierung der erreichten Ergebnisse. In [Gama und Brazdil 1995] werden drei Methoden dazu vorgestellt:

[13] Soll der Lösungsraum eines Klassifikationsverfahrens vollständig durchsucht werden, ergibt sich eine hohe Komplexität und damit oft eine unakzeptable Laufzeit. Fast alle Verfahren verwenden daher eine Heuristik, um den Lösungsraum einzuschränken. Oft bedeutet eine längere Laufzeit eine vollständigere Suche und damit eine bessere Klassifikationsgenauigkeit, nicht einen schlechter implementierten Algorithmus.

1. Methode: Für diese Methode ist Voraussetzung, dass der Datensatz mit allen Algorithmen getestet wurde. Ausgehend von der Fehlerrate ER des besten Algorithmus und der Anzahl der Tupel des verwendeten Testsets NT wird ein Fehlerbereich EM (*error margin*) berechnet.

$$EM = \sqrt{\frac{ER \cdot (100 - ER)}{NT}} \tag{8.1}$$

Für jeden Algorithmus kann jetzt der Abstand vom besten Algorithmus in Fehlerbeichen berechnet werden[14].

2. Methode: Die zweite Methode ist der ersten sehr ähnlich, nur dass hier die Fehlerrate ER des Default-Algorithmus[15] als Ausgangsbasis dient. Daraus wird ebenfalls ein Fehlerbereich EM nach Formel 8.1 und für jeden Algorithmus der Abstand[16] zu ER berechnet.

3. Methode: Auch zur Normalisierung nach dieser Methode müssen zuerst alle Algorithmen mit dem Datensatz getestet werden. Anschließend wird von den erreichten Fehlerraten der Durchschnitt und die Standardabweichung berechnet. Die einzelnen Fehlerraten werden normalisiert, indem der Durchschnittswert subtrahiert und die Differenz durch die Standardabweichung dividiert wird.

Für die Berechnung der Anwendbarkeit eines Algorithmus besitzen alle drei Methoden ihre Vor- und Nachteile. Die 3. Methode hat den Vorteil der klaren Interpretierbarkeit der normalisierten Fehlerrate bezüglich des Mittelwertes. Aber das ist auch gleichzeitig ihr Nachteil[17]. Um die einzelnen Algorithmen als anwendbar oder nicht anwendbar einzustufen, muss eine Abweichung vom Mittelwert festgelegt werden, die für alle Datensätze gleich ist. Algorithmen, deren Fehlerrate unter dieser Schranke liegt, gelten als anwendbar, die anderen als nicht anwendbar.

Der Datensatz "monk1-bin"[18] beispielsweise wird von fast allen Algorithmen mit einer Fehlerrate größer 10% klassifiziert, von drei Algorithmen aber mit 0%. Wenn man für die Anwendbarkeit eines Algorithmus eine Schranke von -1.5 festlegt, würden in diesem Fall nur diese drei Algorithmen als anwendbar eingestuft, was sicherlich akzeptabel ist. Andererseits erreichen auf dem Datensatz "soybean-small"[19] alle Algorithmen, außer OC1 und OneR, eine Fehlerrate von 0.0%. Hier würde bei einer Schranke von -0.4 kein Algorithmus mehr als anwendbar eingestuft.

Die zweite Methode hat den Vorteil der einfachen Berechenbarkeit. Für jeden neu hinzukommenden Algorithmus kann man die Anwendbarkeit sofort entscheiden. Alle

[14] Wie viele Fehlerbereiche der Algorithmus schlechter ist als der beste Algorithmus.

[15] Alle Tupel werden der im Trainingsset am häufigsten vorkommenden Klasse zugeordnet.

[16] Wie viele Fehlerbereiche ist der Algorithmus besser als die Default-Fehlerrate.

[17] Darüber hinaus hat diese Methode den Nachteil, dass alle Berechnungen bei einem neu hinzukommenden Algorithmus wiederholt werden müssen.

[18] monk1-bin ist ein frei zugänglicher Datensatz aus dem UCI–Repository (siehe auch Abschnitt 8.6.1)

[19] soybean-small ist ein frei zugänglicher Datensatz aus dem UCI–Repository (siehe auch Abschnitt 8.6.1)

Algorithmen, deren Fehlerrate kleiner ist als

$$BD = ER - k \cdot EM \quad (k \in \mathbb{N})$$

werden als anwendbar eingestuft. Die Nachteile dieser Methode liegen darin, dass für alle Datensätze ein einheitliches k gewählt werden muss und in der Abhängigkeit von der Default-Fehlerrate. Bei einem Datensatz mit 5 Klassen, die gleichverteilt sind, beträgt die Default-Fehlerrate $ER = 80\%$. Bei 100 Tupeln im Testset beträgt $EM = 4$. Bei einem Datensatz mit nur 2 Klassen und sonst gleichen Voraussetzungen ergeben sich $ER = 50\%$ und $EM = 5$. Angenommen, die erreichten Fehlerraten reichen bei beiden Datensätzen von 10% bis 40%. Während im ersten Fall bei $k = 9$ alle Algorithmen anwendbar wären, würden im zweiten Fall alle Algorithmen als nicht anwendbar eingestuft.

Die erste Methode hat den Nachteil der Abhängigkeit vom besten Algorithmus. Wird ein Algorithmus dazu genommen und schneidet er am besten ab[20], so sind alle Berechnungen zu wiederholen. Die Einstufung als anwendbar erfolgt analog der zweiten Methode, wenn die Fehlerrate eines Algorithmus unter

$$BD = ER + k \cdot EM \quad (k \in \mathbb{N})$$

liegt. Bei geringen Fehlerraten (nahe 0%) kann man mit dieser Methode recht gute Ergebnisse erzielen. Bei größeren Fehlerraten werden bei gleichem k tendenziell mehr Algorithmen als anwendbar eingestuft, da der Fehlerbereich größer wird[21]. Da aber dadurch die Menge der anwendbaren Algorithmen nur erweitert werden kann, führt das zwar zu vermehrtem Aufwand beim Nutzer, es können aber keine potentiellen Lösungen verloren gehen. Ein Problem bei dieser Methode tritt auf, sobald ein Algorithmus eine Fehlerrate von 0% hat, damit werden ausschließlich Algorithmen mit einer Fehlerrate von 0% als anwendbar eingestuft, solche mit z. B. 0.1% schon nicht mehr.

Trotz der genannten Einschränkungen wird in der vorliegenden Arbeit die erste Methode zur Normalisierung der erreichten Ergebnisse benutzt. Um die Anwendbarkeit eines Algorithmus zu bestimmen, kam ein mittlerer Faktor von $k = 4$ zum Einsatz. Die Wahl der Größe von k basierte auf experimentellen Analysen und den oben beschriebenen Eigenschaften. Um auch dann, wenn mindestens ein Algorithmus eine Fehlerrate von 0% erreicht hat, einen etwas größeren Fehlerbereich zu verwenden, wurde in diesen Fällen eine beste Fehlerrate von 1.0% zur Berechnung verwendet.

Allgemein ist der Ansatz der mit AST verfolgt wird offen. Der Anwender kann auf Basis der Erfahrungen und Kenntnisse über das Anwendungsgebiet entscheiden.

[20] Die Wahrscheinlichkeit dafür ist aufgrund der Weiterentwicklung der Algorithmen recht groß.

[21] Ein Beispiel dafür ist der Datensatz monk1-corrupt, bei dem alle Algorithmen als anwendbar eingestuft werden, obwohl die Fehlerrate von $24,31\%$ bis $37,5\%$ reicht

8.6 Vorgehen

Die Idee besteht darin, aus einer Menge von bereits untersuchten Datensätzen den oder die ähnlichsten auszuwählen und dem Anwender die Algorithmen zur Anwendung für seinen eigenen Datensatz vorzuschlagen, die auf den ähnlichen Datensätzen die besten Ergebnisse erreicht haben und die Benutzeranforderungen erfüllen.

Um diese Idee zu realisieren, sind folgende Schritte zu durchzuführen:

1. Aufbau einer Fallbasis.
2. Bestimmung eines Ähnlichkeitsmaßes.
3. Charakterisierung der Verfahren hinsichtlich der Nutzeranforderungen.

8.6.1 Aufbau einer Fallbasis

8.6.1.1 Datensätze

Um in der Fallbasis für möglichst viele Anwendungsfälle einen ähnlichen Fall, das bedeutet konkret einen ähnlichen Datensatz und ein anwendbares Verfahren, finden zu können, sollten Datensätze aus unterschiedlichen Bereichen und mit unterschiedlichen Charakteristiken enthalten sein. Andererseits stößt man bei der Beschaffung geeigneter Datensätze auf einige Schwierigkeiten. Daten aus realen Anwendungen wurden oft unter großen Aufwendungen zusammengetragen oder enthalten Informationen, die der Eigentümer nicht weitergeben will. Daher wurde zum Erstellen der Fallbasis hauptsächlich auf die öffentlichen Datensätze des UCI[22]-Repository zurückgegriffen. Die meisten dieser Datensätze wurden bereits in anderen Untersuchungen verwendet und liegen in einer Form vor, die von MLC++ unmittelbar verarbeitet werden kann. Mit dem Quellcode von MLC++ werden einige der Datensätze mitgeliefert, die zur Untersuchung der unterstützten Verfahren in verschiedenen Punkten variiert wurden[23]. Zu jedem der Datensätze steht eine Names-Datei zur Verfügung, in der die Herkunft des Datensatzes und seine bisherige Verwendung beschrieben wird. Von den zusätzlichen Datensätzen sei der Datensatz *quisclas* erwähnt, der Daten zur Trendprognose von Ausfallquoten enthält, also ein Beispiel einer realen Anwendung (vgl. Abschnitt 3.9).

Soweit vorhanden, wurde die Einteilung eines Datensatzes in Trainings- und Testset beibehalten. Wenn keine Unterteilung vorhanden war, wurde das mit MLC++ mitgelieferte Tool *GenCVFiles* verwendet, um den Datensatz im Verhältnis 2 : 1 in Trainings- und Testset zu zerlegen.

Die Berechnung der Datencharakteristik erfolgte mit dem in Clementine™ eingebundenen DCT. Für jeden Datensatz wurde ein Stream aufgebaut, die Datencharakteristik berechnet und für alle Datensätze in einer Datei abgespeichert. Bei vorhandenen

[22] University of California, Irvine

[23] Ein Beispiel dafür ist der künstlich erzeugte Datensatz *monk*, der in vielfältigen Abwandlungen vorliegt.

linear abhängigen Attributen, angezeigt durch den multiplen Korrelationskoeffizienten, wurden diese eliminiert. Dabei sind für die weitere Verarbeitung jeweils sowohl der ursprüngliche Datensatz als auch der mit dem Tool *Project* bereinigte Datensatz[24] verwendet worden. Das Verhalten der einzelnen Algorithmen bezüglich dieser Datensätze wurde im Rahmen dieser Arbeit aber nicht weiter untersucht. Die Charakteristiken der Datensätze wurden in einer Datei abgespeichert und sind im Anhang in Tabelle A.2 wiedergegeben.

8.6.1.2 Algorithmen

Ebenso wie bei der Auswahl der Datensätze wurde bei den verwendeten Algorithmen eine möglichst große Vielfalt angestrebt. Aber auch hier sind recht enge Grenzen gesetzt. Einige Verfahren, die z. B. in [Michie u. a. 1994] verwendet wurden, sind nur kommerziell verfügbar, so dass auf deren Verwendung verzichtet werden musste. Soweit möglich, wurden alle Verfahren mit allen Datensätzen getestet. Dabei sind an den Datensätzen, bis auf geringe Ausnahmen, keine Änderungen vorgenommen worden, um sie für einen Algorithmus speziell aufzubereiten. Wenn ein Verfahren aufgrund der Eigenschaften des Datensatzes nicht anwendbar ist, wurde diese Kombination als fehlgeschlagener Versuch ebenfalls in die Fallbasis aufgenommen. Das betrifft insbesondere Datensätze, die fehlende Werte enthalten[25] und Datensätze mit mehr als zwei Klassen[26]. Da Winnow und Perceptron nur numerische Attribute verarbeiten können, wurden symbolische Attribute vorher mit dem in MLC++ vorhandenen Tool *Conv* konvertiert[27]. Für alle anderen Verfahren wurden die internen Umwandlungen, soweit notwendig, verwendet. Einige Kombinationen von Datensätzen und Verfahren konnten aus Komplexitätsgründen nicht vollständig getestet werden. Entweder war der Speicherbedarf zu groß, so dass die Verarbeitung abbrach, oder die Verfahren wurden explizit gestoppt, da selbst nach mehreren Tagen noch kein Ergebnis erreicht wurde. Es gab aber auch Abbrüche, ohne dass die Ursache ermittelt werden konnte[28]. Alle Verfahren arbeiteten mit den Standardeinstellungen, irgendwelche Optimierungen oder Anpassungen wurden nicht vorgenommen. Für

[24] Der Datensatz, aus dem linear abhängige Attribute entfernt wurden, erhielt denselben Namen wie der ursprüngliche Datensatz, erweitert um den Zusatz "_clean".

[25] Der Algorithmus Pebls ist nicht in der Lage, fehlende Werte zu verarbeiten. Eine Ausnahme ist die Verarbeitung von Datensätzen mit fehlenden numerischen Werten durch die in Clementine™ implementierten neuronalen Netze. Da in Clementine™ selbst mehrere Möglichkeiten vorhanden sind, auf fehlende Werte zu reagieren, wurden die betroffenen Datensätze trotzdem verarbeitet (Tupel mit fehlenden Werten wurden gestrichen).

[26] Wie schon mehrfach erwähnt, können Winnow und Perceptron diese Datensätze nicht verarbeiten.

[27] Entsprechend der Anzahl der Ausprägungen eines symbolischen Attributes wird dieses durch numerische Attribute ersetzt. Dabei wird jeder Ausprägung ein Attribut zugeordnet, das beim Auftreten dieses Wertes mit 1 belegt wird, alle anderen Attribute werden mit 0 belegt. Dieses Verfahren wird bei neuronalen Netzen, so auch in Clementine™ , häufig verwendet.

[28] Einige der in MLC++ bereitgestellten Verfahren wie Aha-IB, T2, OC1 oder LazyDT sind noch in der Entwicklung oder arbeiten aus anderen Gründen nicht sehr stabil.

die neuronalen Netze von Clementine™ wurde die Methode *dynamic*[29] als Abbruchkriterium verwendet. In der Fallbasis wird mit "MLPN" das Multilayer-Perceptron und mit "RBFN" das Radiale Basisfunktionen Netz bezeichnet. Der Algorithmus Aha-IB stellt die Instance Based Verfahren der Serie IB1-4 bereit, die sich durch entsprechende Parameter auswählen lassen. In der Fallbasis sind alle vier Algorithmen[30], bezeichnet mit "IB1" bis "IB4", enthalten. Eine Übersicht der verwendeten Methoden sind in der Tabelle 8.2 dargestellt.

8.6.2 Bestimmung eines Ähnlichkeitsmaßes

Wie im Abschnitt 8.1 beschrieben wurde, basiert die Berechnung der Ähnlichkeit für numerische Werte in CBR-Works auf der Formel von Seite 155. Damit wird deutlich, dass das Intervall aus dem die Werte stammen können, möglichst genau bestimmt werden muss, wenn das Ähnlichkeitsmaß eine hohe Aussagekraft haben soll. Wenn z. B. alle Werte aus dem Intervall $[0, 10]$ stammen, als Wertebereich für das Attribut und damit auch als Intervall für die Berechnung der Ähnlichkeit aber die reellen Zahlen (in CBR-Works $\pm 1 \cdot 10^{30}$) angegeben wird, so werden alle Werte die Ähnlichkeit von 1 haben. Wenn als Wertebereich aber das tatsächlich vorkommende Intervall angegeben wird, wird auch die Ähnlichkeit Werte zwischen 0 und 1 annehmen. Aus diesem Grunde sind normierte Attribute recht gut geeignet, da für sie definitiv der Wertebereich angegeben werden kann. Bei nicht normierten Attributen oder wenn der Wertebereich nicht begrenzt ist, kann als Wertebereich das Intervall genommen werden, in dem die tatsächlich vorkommenden Werte liegen. Dabei geht man aber das Risiko ein, dass bei einer zu untersuchenden Situation Werte außerhalb dieses Bereiches auftreten.

Die Beschreibung eines Datensatzes wurde in CBR-Works in objektorientierter Repräsentation implementiert. Dabei wurde die Einteilung in einfache, statistische und informationstheoretische Maße zugrunde gelegt. Aus den oben genannten Gründen erfolgte eine Einschränkung und teilweise eine Transformierung der vom DCT berechneten Maße, die aber sonst den im Kapitel 6 beschriebenen Maßen entsprechen. Die Änderungen betreffen folgende Maße:

Sym: Anstelle der absoluten Anzahl der numerischen und symbolischen Attribute wird das Verhältnis von symbolischen Attributen zur Anzahl der Attribute insgesamt (Sym = NrSym / NrAttr) verwendet.

defError: Da von den einzelnen Algorithmen die Fehler- und nicht die Klassifikationsrate ausgegeben wird, wird auch für die Beschreibung des Datensatzes die Fehlerrate in Prozent (defError = (1 - defAcc) * 100%) angegeben.

[29] Sobald sich keine Verbesserungen mehr ergeben, wird das Training abgebrochen, eine genauere Beschreibung findet sich in [Clementine™1998].

[30] Die Ergebnisse der Algorithmen sind meist ähnlich, insbesondere zwischen IB-1 und IB-2 sowie IB-3 und IB-4. Um eine möglichst große Vielfalt an Verfahren zu untersuchen, wurden alle vier Algorithmen in die Fallbasis aufgenommen. Eine weitere Untersuchung über die Unterschiede der Verfahren erfolgt im Rahmen dieser Arbeit nicht.

Outl: In der Beschreibung des Datensatzes wird nicht die absolute Anzahl der Attribute mit Ausreißern verwendet, sondern das Verhältnis zur Anzahl numerischer Attribute (Outl = NrOutl / NrNum).

DiscFct: Die Anzahl der signifikanten Diskriminanzfunktionen kann als Dimension eines Subraumes des untersuchten Raumes gedeutet werden, der tatsächlich zur Unterscheidung der Klassen beiträgt. Daher wird für die Beschreibung des Datensatzes nicht die absolute Anzahl der signifikanten Diskriminanzfunktionen verwendet sondern deren Verhältnis zur Anzahl der numerischen Attribute (DiscFct = NrDisc / NrNum).

SpanSym: Anstelle der beiden Werte für die minimale und maximale Ausprägung symbolischer Attribute wird die Spannweite (SpanSym = SymMax - SymMin) der Ausprägungen verwendet.

Tabelle 8.1 gibt die Objektstruktur, die Typen und Gewichtungen der verwendeten Maße wieder. Bei einer Gewichtung von 0 wird das Attribut für die Ähnlichkeitsberechnung nicht berücksichtigt. Mit der in Tabelle 8.1 wurden bisher die besten Ergebnisse erzielt. Fast alle Attribute gehen mit der gleichen Gewichtung in die Berechnung ein. Nur drei Attributte wurden aufgrund der bekannten Erfahrungen höher gewichtet. Das ist zum einen die Anzahl der Attribute *NrAttr* im Datensatz und die Anzahl der Klassen *NrClass*. Beide Attributte sind starke Indikatoren für die Komplexität der Anwendung. Als drittes Attribut wird *Sym* etwas stärker in der Berechnung der globalen Ähnlichkeit berücksichtigt, da der Anteil an symbolischen Attributten im Datensatz oft ein Kriterium für die Anwendbarkeit einer Methode ist.

Wenn der Wertebereich nicht durch das verwendete Maß eingeschränkt war, wurde er anhand der tatsächlich auftretenden Werte festgelegt. Abbildung 8.6 verdeutlicht die Vererbungsstruktur der verwendeten Typen.

Die Berechnung der lokalen Ähnlichkeit verwendet, bis auf eine Ausnahme, bei allen Slots die Standardeinstellung. Für die Anzahl der Tupel wird der Logarithmus der Werte zur Berechnung der Ähnlichkeit benutzt. Die globale Ähnlichkeit wird als Durchschnitt aller lokalen Ähnlichkeiten berechnet.

Objekt	Slot	Typ	Wertebereich	Gewichtung
DataChar	DataSet	Symbol		0
	DS_allgem	DC_allgem		1
	DS_num	DC_num		1
	DS_sym	DC_sym		1
DC_allgem	DataSet	Symbol		0
	NrAttr	AttrType	1 ... 30	2
	Sym	NormType	0.0 ... 1.0	4
	NrTupel	TupelType	1 ... 10000	1
	NrClass	ClassType	1 ... 10	2
	defError	DefErrorType	0.0 ... 100.0	1
	StdDev	StdDevType	0.0 ... 0.5	1
	MissVal	NormType	0.0 ... 1.0	1
	MissLine	NormType	0.0 ... 1.0	1
DC_num	DataSet	Symbol		0
	Outl	NormType	0.0 ... 1.0	1
	Fract	NormType	0.0 ... 1.0	1
	Cancor	NormType	0.0 ... 1.0	1
	WLambda	NormType	0.0 ... 1.0	1
	DiscFct	NormType	0.0 ... 1.0	1
DC_sym	DataSet	Symbol		0
	SpanSym	SpanSymType	1 ... 10	1
	AvgSym	AvgSymType	0.0 ... 10.0	1
	ClassEntr	ClassEntrType	0.0 ... 4.0	1
	AttrEntr	AttrEntrType	0.0 ... 2.0	1
	MutInf	MutInfType	0.0 ... 1.0	1
	JointEntr	JointEntrType	0.0 ... 5.0	1
	EqNrAttr	EqNrAttrType	0.0 ... 30.0	1
	NoiseRatio	NoiseRatioType	0.0 ... 100.0	1

Tabelle 8.1: Beschreibung des Datensatzes in CBR-Works

Anmerkung: Eine Gewichtung von 0 bedeutet, dass der Slot nicht zur Berechnung der Ähnlichkeit verwendet wird.

8.6.3 Charakterisierung der Verfahren hinsichtlich der Nutzeranforderungen

Von den im Abschnitt 8.4 besprochenen Nutzerforderungen sind in der Arbeit berücksichtigt:

- Verfahren
- Modell
- Trainingszeit
- Klassifikationszeit (Testzeit)

Dabei werden diese Anforderungen als Eigenschaften der Algorithmen verstanden, d. h. eine Aussage über die zu erwartende Trainingszeit wird nicht in Abhängigkeit vom zu untersuchenden Datensatz sondern nur in Abhängigkeit vom verwendeten Algorithmus gegeben. Ein unmittelbarer Zusammenhang zur zu erwartenden Laufzeit für das Training oder die Klassifikation zu einzelnen Eigenschaften, z. B.. Anzahl der Tupel und Anzahl der Attribute, eines Datensatzes konnte nicht ermittelt werden. Da alle Algorithmen Heuristiken verwenden, ist die zu erwartende Laufzeit offensichtlich von mehreren Faktoren abhängig. Daher wurde auf eine Vorhersage der Laufzeit für den zu untersuchenden Datensatz verzichtet. Bei der Ausgabe der ähnlichsten Fälle werden aber die für die jeweilige Datensatz-Algorithmus-Kombination benötigten Zeiten mit angezeigt.

Um die einzelnen Algorithmen bezüglich der Trainings- und Testzeiten zu klassifizieren, wurde das in Clementine™ enthaltene KMeans-Verfahren verwendet. Dazu wurden jeweils die durchschnittlich und die maximal benötigte Zeit[31] der Algorithmen für alle Datensätze[32] als Eingabe zum Clustern verwendet. Die Trainingszeiten wurden in fünf und die Testzeiten in drei Klassen eingeteilt, die mit *slow*, *moderate* und *fast* bzw. bei den Trainingszeiten zusätzlich mit *very slow* und *very fast* interpretiert wurden. Die Tabellen A.1 und A.2 im Anhang zeigen die Clusterstrukturen.

Ebenfalls für jeden Algorithmus wurde eine Eigenschaft *Modell* festgelegt, die beschreibt, ob ein Modell erzeugt wird und wenn ja, ob es interpretierbar ist oder nicht. Diese Eigenschaft wurde nicht an die praktische Realisierung des Algorithmus geknüpft sondern an das verwendete Verfahren[33]. Tabelle 8.2 zeigt die verwendeten Algorithmen und die ihnen zugeordneten Eigenschaften.

[31] Da die tatsächlichen Zeiten zu keiner befriedigenden Clusterstruktur führten, wurde der natürliche Logarithmus des Durchschnitts und des Maximums verwendet, was sehr kompakte Cluster ergab.

[32] Es wurden alle die Datensätze berücksichtigt, auf denen alle Algorithmen außer Winnow, Perceptron und Pebls zu einem Ergebnis kamen. Dadurch fallen besonders die größeren Datensätze aus der Berechnung, da bei ihnen vermehrt Probleme auftraten.

[33] Einige Verfahren, die eigentlich ein Modell erzeugen, sind in MLC++ so implementiert, dass das Modell nicht ausgegeben wird und damit auch nicht abgespeichert werden kann.

Verfahren	Modelltyp	Trainingszeit	Klassifikationszeit
C5.0	interpretable	very fast	fast
CN2	interpretable	fast	fast
IB	no	very fast	slow
IB1	no	moderate	slow
IB2	no	moderate	slow
IB3	no	fast	moderate
IB4	no	fast	moderate
ID3	interpretable	very fast	fast
LAZYDT	interpretable	very fast	slow
MC4	interpretable	very fast	fast
MLPN	non interpretable	very slow	moderate
NAIVE-BAYES	no	very fast	fast
NBTREE	no	slow	fast
OC1	interpretable	slow	fast
ONER	interpretable	moderate	fast
PEBLS	no	very fast	moderate
PERCEPTRON	non interpretable	very fast	fast
RBFN	non interpretable	very slow	slow
RIPPER	interpretable	moderate	fast
T2	interpretable	very slow	fast
WINNOW	non interpretable	very fast	fast

Tabelle 8.2: Eigenschaften der Klassifikationsverfahren

Die Verfahren sind in CBR-Works als Instanzen eines Objekts *Inducer* abgebildet, wobei die Slots den Spalten der Tabelle 8.2 entsprechen. Die Verfahren selbst sind als Taxonomie entsprechend Abbildung 8.6 angelegt. Die anderen Slots verwenden geordnete Symbole als Typen, wobei die Ähnlichkeitsmaße so eingestellt sind, dass die Ähnlichkeit 1 beträgt, wenn der ausgewählte Fall mehr als die Nutzeranforderungen realisiert[34]. Alle Slots besitzen eine einfache Gewichtung.

Die Verbindung von Verfahren und Datensatz geschieht in CBR-Works durch das Objekt *Case*, dessen Aufbau Tabelle 8.3 verdeutlicht. Die Ursache der hohen Gewichtung für die Datencharakteristik wird im nächsten Abschnitt beschrieben. Die Trainings- und Testzeit sind nur zur Information und gehen nicht in die Ähnlichkeitsberechnung ein.

[34]Niemand wird wohl ernsthaft ein langsames Verfahren verlangen, wenn er auch ein schnelles bekommen kann, das gleich gut oder besser ist. Ähnlich verhält es sich mit dem Modelltyp.

Objekt	Slot	Typ	Wertebereich	Gewichtung
Case	DataSet	DataChar		10
	Error	PercType	0.0 . . . 100.0	1
	Method	Inducer		1
	TrainTime	String		0
	TestTime	String		0

Tabelle 8.3: Fallstruktur in CBR-Works

Anmerkung: Eine Gewichtung von 0 bedeutet, dass der Slot nicht zur Berechnung der Ähnlichkeit verwendet wird.

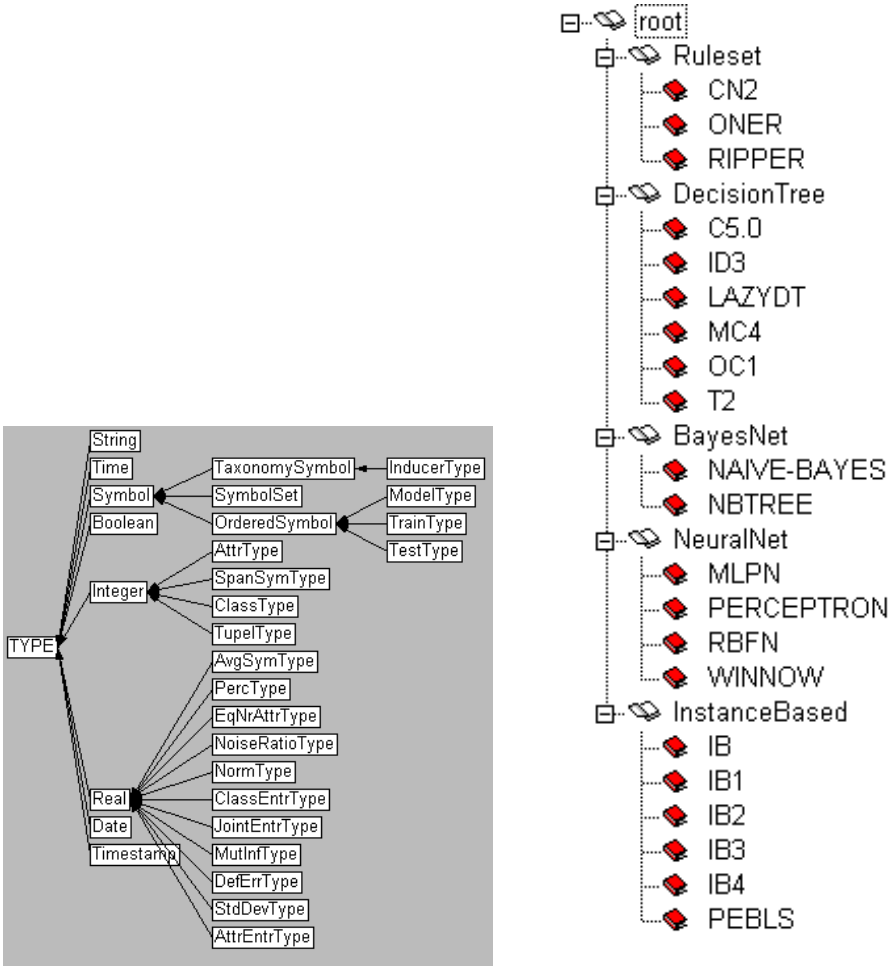

Abbildung 8.6: Vererbungsstruktur der verwendeten Typen und Taxonomie der Verfahren

Kapitel 9

Evaluierung des Systems

9.1 Ergebnisse mit AST

Für ein Verfahren mit so vielen Einflussfaktoren ergibt sich die Frage nach der Validierung der Ergebnisse. In anderen Untersuchungen, z.B. [Zintz 1998], wurde ein Datensatz aus der Menge zur Ermittlung der Regeln herausgenommen und zur Bewertung der Ergebnissse benutzt. Bei diesem Verfahren spielt der Zufall sicherlich eine beträchtliche Rolle. Andererseits ist in der vorliegenden Arbeit die Anzahl der Parameter, die das Endergebnis beeinflussen, so groß, dass eine vollständige Validierung sehr aufwendig ist. Als Endergebnis soll hier das Resultat folgenden Prozesses verstanden werden:

- Der Anwender hat einen Datensatz und ein bestimmtes Ziel für das Data Mining. Aus diesem Ziel ergeben sich die in dieser Arbeit verwendeten Nutzeranforderungen.
- Für den Datensatz wird mit Hilfe des DCT die Datencharakteristik bestimmt und als Abfrage formatiert[1].
- Die Beschreibung des Datensatzes und die Nutzeranforderungen werden in CBR-Works als Abfrage eingegeben[2]. Von CBR-Works wird eine Liste mit den ähnlichsten Fällen erzeugt.
- Diese Fälle werden dem Anwender zur Entscheidungshilfe angeboten.

[1] Um mit Hilfe des DCT die Datencharakteristik des Datensatzes zu berechnen, diese in die für die Beschreibung des Datensatzes in CBR-Works verwendeten Maße umzuwandeln und mittels ODBC in einer Datenbank abzuspeichern, steht in Clementine™ ein vorbereiteter Stream zur Verfügung.

[2] Um nicht alle Maße per Hand eingeben zu müssen, kann die zuvor in einer Datenbank abgespeicherte Beschreibung des Datensatzes mittels ODBC als neuer Fall in die Datenbasis aufgenommen und als Abfrage definiert werden. Die Werte für die Nutzeranforderungen müssen selbst eingegeben werden.

Lässt man die Nutzeranforderungen und die zu erreichende Fehlerrate als undefiniert, gehen sie nicht in die Ähnlichkeitsberechnung ein und es wird nur die Datencharakteristik verwendet. Folgerichtig erhält man eine Liste ähnlichster Fälle, die für den ähnlichsten Datensatz alle verwendeten Verfahren auflistet. Um diese Verfahren nach der erreichten Fehlerrate zu sortieren, kann eine Fehlerrate von 0 für die Abfrage eingestellt werden. Damit jetzt nicht Fälle als ähnlich eingestuft werden, die zwar eine geringe Fehlerrate erreicht haben, deren Datencharakteristik aber zur Abfrage nicht sehr ähnlich ist, wurde die recht hohe Gewichtung für die Datencharakteristik eingestellt[3].

Für die Evaluierung von AST wurde folgende Fragestellung untersucht:

Kann von einer ähnlichen Datencharakteristik auch auf ein ähnliches Verhalten der Methoden und der Ergebnisse der Methoden geschlossen werden.

Um das zu überprüfen, wurde als Fallbasis die Datencharakteristik der verwendeten Datensätze benutzt und zu jedem Datensatz wurden die drei ähnlichsten Datensätze bestimmt. Die drei besten Algorithmen dieser ähnlichsten Datensätze wurden auf Anwendbarkeit bezüglich des ursprünglichen Datensatzes untersucht. Die genauen Ergebnisse dieser Vergleiche findet man in den Tabellen 9.2, 9.3 und 9.4, unterteilt in Datensätze, die sowohl numerische als auch symbolische Attribute besitzen, sowie solche, die ausschliesslich numerische bzw. symbolische Attribute besitzen. Dabei enthält die erste Spalte den Namen des Datensatzes, die Spalten mit simX die Ähnlichkeit des Datensatzes zum 1., 2. und 3. ähnlichsten Datensatz der Fallbasis und die Spalten mit apX.Y die Einstufung des besten, zweit- und drittbesten Algorithmus auf dem Datensatz X bezüglich des untersuchten Datensatzes. In der vorletzten und letzten Zeile sind jeweils die Prozentzahlen der anwendbaren Algorithmen angegeben.

Dabei ergeben sich unterschiedliche Ergebnisse, wie Tabelle 9.1 zeigt.

Datensatz	ähnlichster		zweitähnlichster		drittähnlichster	
Algorithmus	bester	∅	bester	∅	bester	∅
mixed	85.71%	82.54%	66.67%	77.78%	61.90%	58.73%
numeric	86.21%	73.56%	72.41%	72.41%	65.52%	67.82%
symbolic	67.74%	60.22%	67.74%	62.37%	48.39%	48.39%
all	79.01%	70.78%	69.14%	69.96%	58.02%	58.02%

Tabelle 9.1: Anwendbarkeit der Algorithmen

Wenn jeweils der beste Algorithmus auf dem ähnlichsten Datensatz zugrunde gelegt wird, so ist dieser auf Datensätzen mit numerischen Attributen in mehr als 85% der Fälle anwendbar, bei Datensätzen mit ausschließlich symbolischen Attributen aber nur in etwas weniger als 68% der Fälle.

[3]Vergleiche Abschnitt 8.6

Bei den Datensätzen “segment”, “sonar” und “soybean-small” ließen sich nicht alle Werte der Datencharakteristik berechnen (siehe Tabelle A.2). Da die fehlenden Werte als “unknown” in die Berechnung der Ähnlichkeit bei den numerischen Attributen eingehen, wurden für diese Datensätze solche Datensätze als ähnlichste ermittelt, die nur symbolische Attribute enthalten. Trotzdem sind alle 9 in diesem Test vorgeschlagenen Algorithmen anwendbar.

Nur der beste Algorithmus auf den ähnlichsten Datensätzen ist im Durchschnitt besser als der zweit- und drittbeste Algorithmus. Bei den zweit- und drittähnlichsten Datensätzen sind oft die zweit- und drittbesten Algorithmen auf dem zu untersuchenden Datensatz besser.

Das Ergebnis zeigt, dass die Verfahren alle eine gewisse Güte haben, aber es gibt dennoch immer ein signifikant besseres Verfahren. Mit dem CBR–Ansatz steht der Gedanke der Entscheidungsunterstützung im Vordergrund. Die Evaluierung einer Vielzahl von Verfahren soll damit vermieden werden.

Generell liegt der Prozentsatz der Anwendbarkeit des besten Algorithmus bei Datensätzen mit ausschliesslich numerischen Attributen immer über dem bei Datensätzen, die auch symbolische Attribute enthalten.

Wie kommt es zu den unterschiedlichen Ergebnissen zwischen den Datensätzen? Die Ergebnisse lassen den Schluss zu, dass die Datencharakteristik für symbolische Attribute nicht dieselbe Güte haben, wie die Datencharakteristik der numerischen Attribute. Im Abschnitt 11 wird dieser Punkt diskutiert und mit Ergebnissen aus anderen Arbeiten verglichen.

DataSet	sim1	ap1.1	ap1.2	ap1.3	sim2	ap2.1	ap2.2	ap2.3	sim3	ap3.1	ap3.2	ap3.3
abalone	0.872	a	na	na	0.865	na	na	a	0.861	na	na	a
adult	0.932	na	a	na	0.926	a	na	a	0.857	na	na	na
anneal	0.835	a	a	na	0.727	a	a	a	0.703	na	na	na
anneal-u	0.835	a	a	a	0.801	na	a	a	0.791	a	a	a
australian	0.991	a	a	a	0.932	a	a	na	0.923	a	na	a
auto	0.953	a	a	a	0.846	a	a	a	0.830	a	na	a
auto_clean	0.953	a	a	a	0.867	a	na	a	0.815	na	na	na
cars	0.911	a	a	a	0.906	na	a	a	0.888	a	na	na
cars_clean	0.872	a	a	a	0.778	a	a	a	0.772	a	na	na
cleve	0.923	a	a	a	0.920	a	a	a	0.907	a	a	a
crx	0.991	a	a	a	0.926	a	a	na	0.920	a	a	a
flag-language	0.922	a	a	a	0.815	a	a	a	0.798	a	a	a
flag-religion	0.922	a	a	a	0.867	a	a	a	0.846	a	a	a
german	0.875	a	a	a	0.873	a	a	a	0.872	a	a	a
german-org	0.885	a	a	a	0.883	a	a	a	0.882	a	a	na
hepatitis	0.907	na	na	a	0.878	na	na	a	0.877	na	na	a
horse-colic	0.874	a	na	a	0.867	a	a	a	0.861	a	a	a
hypothyroid	0.991	a	a	a	0.883	na	a	a	0.839	na	na	a
hypothyroid_clean	0.991	a	a	a	0.885	na	a	a	0.841	na	na	a
labor-neg	0.897	na	na	na	0.877	na	na	a	0.830	na	na	a
solar	0.890	a	a	a	0.859	a	a	a	0.823	a	a	a
avg		85.71%	80.95%	80.95%		66.67%	76.19%	90.48%		61.90%	42.86%	71.43%
all		82.54%				77.78%				58.73%		

Tabelle 9.2: Anwendbarkeit der besten Verfahren ähnlicher Datensätze für Datensätze mit numerischen und symbolischen Attributen

DataSet	sim1	ap1.1	ap1.2	ap1.3	sim2	ap2.1	ap2.2	ap2.3	sim3	ap3.1	ap3.2	ap3.3
balance-scale	0.928	na	a	na	0.911	na	na	na	0.910	na	a	na
breast	0.996	a	a	a	0.925	a	a	a	0.905	a	a	a
breast_clean	0.996	a	a	a	0.923	a	a	a	0.908	a	a	a
ecoli	0.896	a	a	a	0.892	a	a	a	0.872	a	a	a
glass	0.884	a	a	a	0.868	a	a	a	0.853	a	a	a
glass2	0.966	a	na	na	0.950	na	a	na	0.930	a	a	a
glass2_clean	0.966	a	a	a	0.966	a	a	a	0.964	a	a	a
heart	0.928	a	a	a	0.925	a	a	a	0.925	a	a	a
ionosphere	0.530	a	na	na	0.530	a	a	a	0.529	a	a	a
iris	0.978	a	a	a	0.928	a	a	a	0.899	a	a	a
iris_clean	0.978	a	a	a	0.910	a	a	a	0.884	a	a	a
letter	0.996	a	na	na	0.892	na	na	na	0.861	na	na	na
letter_clean	0.996	a	na	na	0.896	na	na	na	0.865	na	na	na
liver	0.964	a	a	a	0.947	a	a	a	0.930	a	a	a
page-blocks	0.974	a	a	a	0.883	a	a	a	0.858	a	a	na
page-blocks_clean	0.974	a	a	a	0.866	a	a	a	0.853	a	a	na
pima	0.966	a	na	na	0.950	a	a	a	0.947	a	a	a
quisclas	0.887	na	a	na	0.878	a	na	na	0.866	na	na	na
satimage	0.900	a	a	a	0.865	na	na	a	0.839	na	a	na
satimage_clean	0.922	na	na	a	0.900	a	a	a	0.883	na	a	na
segment	0.611	a	a	a	0.608	a	a	a	0.577	a	a	a
sonar	0.636	a	a	a	0.565	a	a	a	0.531	a	a	a
soybean-small	0.525	a	a	a	0.498	a	a	a	0.491	a	a	a
vehicle	0.922	a	a	na	0.922	na	a	a	0.922	na	na	a
vehicle_clean	0.912	a	a	a	0.905	na	na	a	0.897	a	na	na
waveform-21	0.922	na	na	na	0.915	a	na	na	0.905	na	na	na
waveform-40	0.915	a	a	na	0.851	na	na	na	0.850	na	na	na
wdbc	0.902	a	a	a	0.900	a	a	a	0.879	na	a	a
wine	0.922	a	na	na	0.899	a	a	na	0.892	a	a	a
avg		86.21%	72.41%	62.07%		72.41%	72.41%	72.41%		65.52%	75.86%	62.07%
all		73.56%				72.41%				67.82%		

Tabelle 9.3: Anwendbarkeit der besten Verfahren ähnlicher Datensätze für Datensätze mit ausschließlich numerischen Attributen

DataSet	sim1	ap1.1	ap1.2	ap1.3	sim2	ap2.1	ap2.2	ap2.3	sim3	ap3.1	ap3.2	ap3.3
breast-cancer	0.890	a	a	a	0.890	a	a	a	0.890	a	a	a
car	0.920	na	na	na	0.915	na	na	na	0.915	na	na	na
chess	0.887	a	na	na	0.880	a	na	na	0.879	na	na	a
dna	0.823	na	a	na	0.823	na	na	a	0.819	a	na	a
led24	0.884	na	a	na	0.798	na	a	a	0.794	a	a	na
led7	0.884	na	na	na	0.841	na	na	na	0.823	na	na	na
lenses	0.914	a	a	a	0.908	a	a	a	0.908	a	a	a
monk1	1.000	a	a	na	1.000	a	a	na	0.961	na	na	na
monk1-bin	0.968	a	a	na	0.965	a	a	na	0.963	a	na	a
monk1-corrupt	0.913	a	a	a	0.913	a	a	a	0.913	a	a	a
monk1-cross	0.934	na	na	a	0.934	na	na	a	0.934	na	na	a
monk1-full	1.000	a	a	na	1.000	a	a	na	0.961	na	na	na
monk1-local	0.968	a	a	a	0.960	a	na	a	0.956	a	a	na
monk1-org	1.000	a	a	na	1.000	a	a	na	0.961	na	na	na
monk2	0.956	na	a	na	0.946	a	a	na	0.929	a	a	na
monk2-bin	0.976	a	na	na	0.951	na	na	a	0.949	na	a	na
monk2-local	0.976	a	na	na	0.937	na	na	na	0.929	a	na	na
monk3	1.000	a	a	a	1.000	a	a	a	0.961	na	na	a
monk3-full	1.000	a	a	a	1.000	a	a	a	0.961	na	na	a
monk3-local	0.960	a	na	a	0.951	na	a	na	0.928	a	na	na
monk3-org	1.000	a	a	a	1.000	a	a	a	0.961	na	na	a
mushroom	0.875	a	a	na	0.874	a	na	a	0.861	a	a	a
mux6	0.970	a	na	a	0.965	a	a	a	0.956	a	a	na
nursery	0.920	na	na	na	0.890	na	na	na	0.890	na	na	na
parity5+5	0.869	na	a	a	0.862	a	na	a	0.853	na	a	a
soybean-large	0.806	na	a	na	0.785	a	a	a	0.750	na	a	a
threeof9	0.970	a	a	na	0.963	a	a	a	0.954	a	a	na
tic-tac-toe	0.956	a	na	na	0.949	a	na	a	0.929	na	a	a
titanic	0.931	a	a	a	0.931	a	a	a	0.931	a	a	a
vote	0.915	a	a	a	0.914	a	a	a	0.910	a	a	na
zoo	0.890	na	a	a	0.848	na	na	na	0.841	na	na	na
avg		67.74%	67.74%	45.16%		67.74%	58.06%	61.29%		48.39%	48.39%	48.39%
all		60.22%				62.37%				48.39%		

Tabelle 9.4: Anwendbarkeit der besten Verfahren ähnlicher Datensätze für Datensätze mit ausschließlich symbolischen Attributen

9.2 Vergleich mit METAL

9.2.1 METAL

Das EU–Projekt METAL[4], *A Meta–Learning Assistant for Providing User Support in Machine Learning and Data Mining* wurde 1999 gestartet. Die beteiligten Partner[5] waren:

- Universität Wien, ÖFAI[6]
- Universität Bristol
- Universität Genf
- DaimlerChrysler AG
- Integral Solutions Ltd., London (UK) (bis 2000)
- Dialogis Software & Services GmbH, (Germany) (ab 2001)

Ziel des Projektes war es, eine Unterstützung für Anwender von maschinellen Lernverfahren und Data Mining Verfahren zu entwickeln. Innerhalb des Projektes hat man sich auf Klassifikationsmethoden und Regressionsverfahren beschränkt.

Das Projekt endete im Herbst 2002 und seit dem Frühjahr 2003 ist das Ergebnis in Form des DM Advisor unter www.metal-kdd.org verfügbar.

9.2.2 DM Advisor

Zusammengefasst werden in DM Advisor die Informationen bzgl. der Performance von Methoden auf Datensätzen, die ähnlich zu einem neuen Datensatz sind genutzt, um die erwartete Performance zu schätzen und anschließend eine Rangfolge zu erstellen. Hierzu werden, wie in AST eine Datencharakterisik basierend auf DCT berechnet, über diese Charakteristik wird die Ähnlichkeit eines neuen Datensatzes bestimmt. Für diesen Ansatz wird die *k-Nearest Neighbor* Methode genutzt. Die Performance der Methoden wird nicht nur über die Klassifikationsgenauigkeit definiert, sondern die Geschwindigkeit wird auch berücksichtigt und der Anwender hat die Kontrolle, welches Kriterium stärker berücksichtigt wird. Eine gute Erläuterung findet man auch in [Brazdil u. a. 2003].

[4] Esprit LTR Projekt METAL 26.357

[5] Der Autor war im Rahmen seiner Tätigkeit bei der DaimlerChrysler AG an dem Projekt beteiligt

[6] Österreichisches Forschungsinstitut für Künstliche Intelligenz

9.2.2.1 Ähnlichkeit zwischen Datensätzen in DM Advisor

Basis für die Beschreibung der Datensätze ist auch hier DCT. Die Anzahl der Maße in DCT ist relativ groß im Verhältnis zu den zur Verfügung stehenden Beispielen (Datensätzen). Erschwerend kommt die Tatsache hinzu, dass *nearest Neighbor* Methoden sensitiv auf irrelevante Attribute reagieren [Mitchell 1997].

Im Rahmen des Projektes METAL wurde deshalb a priori eine kleine Teilmenge von Maßen ausgewählt, von denen man annimmt, dass sie Informationen über die Performance Eigenschaften der Methoden beinhalten. Die folgenden Maße wurden ausgewählt:

- Anzahl von Beispielen
- Anteil von symbolischen Attributen
- Anteil von fehlenden Werten (Missing Values)
- Anteil von Attributen mit Ausreißern
- Klassenentropy
- Durchschnittliche *mutual information* der Klassen und Attribute
- 1. Kanonische Korrelation

Im Rahmen von AST wurden mehr Maße von DCT berücksichtigt. Betrachtet man die Ergebnisse der Evaluierung von AST, so ist eine Schlussfolgerung, dass zu viele Maße berücksichtigt werden, bestimmt nicht zulässig. Wie in Abschnitt 9.1 beschrieben, sollte die Datencharakteristik in Bezug auf die symbolischen Attribute verbessert werden.

9.2.3 Unterstützte Methoden

In Rahmen von METAL wurden zehn Methoden ausgewählt, die im Prozess der Entscheidungsunterstützung angeboten werden.

1. c5.0 (tree)
2. c5.0 (boost)
3. c5.0 (rule)
4. ripper
5. Ltree
6. IB1

7. NB
8. MLP
9. RBFN
10. LD

Im Vergleich hierzu wurden in AST 21 Methoden berücksichtigt (vgl. Kapitel 7).

9.2.4 DM Advisor versus AST

Der DM Advisor und AST haben dieselbe Zielsetzung. Beide wollen den Schritt der Methodenauswahl im KDD Prozess unterstützen und beide geben auf Basis einer Datencharakteristik eine Empfehlung. Beide Ansätze verfolgen trotzdem unterschiedliche Philosophien. AST ist Teil des UGM–Ansatzes, in dem der Anwender die zentrale Rolle spielt. Als Teil des UGM–Ansatzes wird mit AST auch ein genereller Ansatz der Benutzerunterstützung verfolgt, in dem nicht nur die Klassifikationsgüte und Performance der Methoden berücksichtigt werden. Mit AST können auch die nicht funktionalen Anforderungen der Anwendung berücksichtigt werden.

Der DM Advisor unterscheidet sich in einem weiteren Punkt vom AST–Ansatz. Während in AST *nur* die ähnlichsten Fälle mit ihren Ergebnissen angezeigt werden, kann der Anwender im DM Advisor verschiedene Rankingmethoden auswählen. Innerhalb dieser Rankingmethoden kann der Anwender angeben, ob die Performance der Methode oder der Klassifikationsgüte stärker berücksichtigt werden sollen.

Eine solche Option könnte man in AST bei der Berechnung der globalen Ähnlichkeit berücksichtigen. Dieses ist aber zur Zeit in AST nicht vorgesehen.

9.3 CBR Ansatz im Projekt MiningMart

Die Grundidee des Ansatzes im Projekt MiningMart[7] [Morik und Scholz 2003] ist es, die *Best Practice* für die Preprozess Datenaufbereitung zur Benutzerunterstützung wiederzufinden, die von Experten entwickelt wurden. Die Daten werden hierzu in einer Meta-Sprache beschrieben und in Anwendungsgebiete unterteilt. MiningMart Anwender wählen einen Fall und wenden die korrespondierenden Transformationen und Methodenketten in ihre Anwendung an.

In dem Projekt wurden folgende Ziele erreicht:

- MiningMart hat eine Metasprache zur Beschreibung der Daten und Operatoren entwickelt.
- MiningMart hat erste Fälle der Fallbasis erarbeitet.

[7] http://mmart.cs.uni-dortmund.de/research/index.html

- MiningMart stellt seine Fallbasis im Internet zur Verfügung.

Stand heute sind 3 Fälle in der Fallbasis aus den Breichen Betrugserkennung und Telekommunikation verfügbar. Zur Beschreibung der Daten werden auch hier einfache statistische Maße benutzt. Interessant ist hier die Unterstützung vor der Methodenauswahl, durch eine Folge von Operationen. Auch hier wurde erkannt, dass nur eine Unterstützung über die Erfahrung ein sinnvoller Weg sein kann.

Kapitel 10

Datengenerator

In den vorherigen Kapiteln wurde schon auf das Problem der Verfügbarkeit von Datensätzen eingegangen. Da reale Datensätze im Allgemeinen nicht frei zugänglich sind, wird in den folgenden Abschnitten ein Datengenerator beschrieben, der zu einer gegebenen Datencharakteristik aus DCT einen Datensatz generiert, der dieser Charakteritik entspricht.

10.1 Generierung einer Datenbank von Fallbeispielen

In diesem Abschnitt werden die verschiedenen Arten zur Generierung von Datenbanken von Fallbeispielen aufgezeigt. Hierbei wird zuerst die Generierung symbolischer Attribute besprochen. Im Anschluss wird die Generierung numerischer Attribute mit uni- und multivariater Verteilung und Verfahren zur Schätzung der Dichtefunktion vorgestellt. Dann wird die Generierung mit Hilfe von Regeln betrachtet. Danach wird noch ein Ansatz beschrieben, der die Spezifikation von symbolischen und numerischen Attributen erlaubt.

10.1.1 Generierung symbolischer Attribute

In diesem Abschnitt soll die Generierung symbolischer Attribute vorgestellt werden. Dazu wird zunächst geklärt, was man konkret unter symbolischen Attributen versteht und welche Beschreibungsmöglichkeiten es gibt. Im Anschluss wird die Generierung einzelner symbolischer Attribute besprochen, bevor Konzepte zur Generierung symbolischer Attribute mit gewissen Abhängigkeiten vorgestellt werden.

Ein symbolisches Attribut wird durch die Menge der möglichen Ausprägungen beschrieben. Auf dieser Menge ist nicht zwingend eine Ordnung definiert. Bezogen auf eine Datenbank von Fallbeispielen entspricht ein symbolisches Attribut einem Spaltenvektor. Ein Attribut lässt sich auch über die Häufigkeitsverteilung angeben, d.h. die Anzahl, mit der jede Attributausprägung in den Daten vorkommt.

Ein einzelnes Attribut lässt sich bei gegebener Häufigkeitsverteilung leicht mit Hilfe einer entsprechenden Multinomial-Verteilung generieren.

Wenn man nun mehrere Attribute gleichzeitig betrachtet, kann man über gemeinsame Häufigkeitsverteilungen die Abhängigkeiten der Attribute untereinander beschreiben. Für die Generierung ist dies allerdings nicht besonders praktisch, da man schon bei mehr als zwei symbolischen Attributen eine sehr hohe Zahl an Parametern benötigen würde. Man braucht also eine kompaktere Darstellungsweise zur Modellierung der Abhängigkeiten der Attribute einer Datenbank und ein effizientes Verfahren zur Generierung. Eine mögliche Lösung stellt die Verwendung von Regeln dar. Mit Hilfe von Regeln ist es sehr einfach, Abhängigkeiten zwischen Attributen vorzugeben. Wie solche Regeln aussehen können und wie man sie effizient für die Generierung einsetzen kann, wird in Abschnitt 10.1.3 betrachtet.

Im folgenden Abschnitt wird auf die Generierung numerischer Attribute eingegangen.

10.1.2 Generierung numerischer Attribute

Es soll zunächst erläutert werden, was man unter einem numerischen Attribut versteht und welche Beschreibungsmöglichkeiten es gibt. Im Anschluss werden die Generierungsmöglichkeiten sowohl für den univariaten als auch für den multivariaten Fall angegeben. Bezogen auf eine Evaluierung wird am Schluss noch allgemein auf Methoden der Dichteschätzung eingegangen.

Ein numerisches Attribut entspricht einer Zufallsvariablen. In der Regel ist ein numerisches Attribut entweder Element der ganzen oder der reellen Zahlen. Im folgenden soll dieser Zusammenhang mit den Begriffen *integer* für ein Attribut $X \in \mathbb{Z}$ bzw. *continous* für $X \in \mathbb{R}$ beschrieben werden. Am besten lassen sich numerische Attribute über die zugrunde liegende Verteilung beschreiben. Für den univariaten Fall stehen bereits entsprechende Verfahren zur Verfügung.

In der Regel benötigt man die entsprechenden Verteilungsparameter und im multivariaten Fall eventuell die Kovarianzmatrix. Da aber reale Datenbanken von Fallbeispielen oft nur eine kleine Auswahl eines Modells repräsentieren, ist gerade das Finden einer geeigneten Verteilung problematisch.

Hat man eine Datenbank von Fallbeispielen mit ausschließlich numerischen Attributen gegeben, kann man versuchen die Dichtefunktionen der einzelnen Attribute zu schätzen. Dazu hat man verschiedene Möglichkeiten, [Silverman 1986] beschreibt u. a. die folgenden Verfahren

- Histogramme,
- Naive Schätzer,
- Kernschätzer
- und *Nearest Neighbor* Methoden.

Bei Histogrammen wird die Häufigkeitsverteilung des auf n äquidistante Intervalle aufgeteilten Raumes graphisch dargestellt. Man hat so die Möglichkeit, die grobe Struktur des numerischen Attributs zu betrachten. Dies kann auch bezogen auf eine Klassifikation erfolgen. Anzumerken ist, dass die Anzahl der "Körbe" die Form des Histogramms verzerren kann.

Der Naive Schätzer wird in (10.1) angegeben. Konkret wird die Anzahl der Beispiele gezählt, die in das Intervall $(x-h, x+h)$ fallen. Dabei wird $h \in \mathbb{R}$ nahe Null gewählt. Naive Schätzer sind zur Visualisierung gedacht. Als nachteilig kann es empfunden werden, dass man, wie bei Histogrammen, eine Sprungfunktion erhält.

$$\begin{aligned} \hat{f}(x) &= \frac{1}{n}\sum_{i=1}^{n}\frac{1}{h}w\left(\frac{x-X_i}{h}\right) \\ &= \frac{1}{2hn} \quad \text{mit } w(x) = \frac{1}{2} \text{ für } |x| < 1 \text{ sonst } 0. \end{aligned} \tag{10.1}$$

Kernschätzer stellen eine Verallgemeinerung der Naiven Schätzer dar, da lediglich die Gewichtungsfunktion $w(x)$ durch eine sogenannte Kern-Funktion ersetzt wird.

$$\int_{-\infty}^{\infty} K(x)dx = 1. \tag{10.2}$$

Auf diese Weise erhält man eine geglättete Kurve, was unter Umständen das Erkennen einer speziellen Dichtefunktion erleichtert. Kernschätzer sind laut [Silverman 1986] die am häufigsten eingesetzten Verfahren zur Dichteschätzung.

Abschließend sollen nun noch kurz *Nearest Neighbor* Methoden zur Dichteschätzung betrachtet werden. Dabei wird die lokale Dichte der Daten bei der Glättung berücksichtigt. In (10.3) wird ein entsprechender Schätzer angegeben. $d_k(x)$ bezeichnet die Distanz der einzelnen Datenpunkte.

$$\hat{f}(x) = \frac{k}{2nd_k(x)}. \tag{10.3}$$

In [Silverman 1986] werden aber nicht nur Verfahren zur Dichteschätzung für den univariaten Fall vorgestellt. Es wird ebenso eine Kernschätzer-Methode für den multivariaten Fall vorgestellt. Es würde allerdings zu weit führen, hier darauf näher einzugehen.

Es bleibt festzuhalten, dass bereits alle notwendigen Verfahren verfügbar sind, um sowohl univariate als auch spezielle multivariate Verteilungen zu generieren. Mit DCT wurden in Abschnitt 6.2 Maße definiert, die auch zur Evaluierung der vorgestellten Verfahren eingesetzt werden.

10.1.3 Generierung mit Hilfe von Regeln

Als eine mögliche Form der Beschreibung einer Datenbank von Fallbeispielen sollen im Folgenden Regeln betrachtet werden. Dabei soll der Generator von Gabor Melli als Beispiel für die Generierung auf Basis von Regeln dienen. Im Anschluss wird die Verwendung von Assoziationsregeln zum Aufbau einer Regelbasis diskutiert. Die Konsequenz könnte eine Modellierung der Daten auf rein symbolischer Ebene sein. Aus diesem Grund werden im Anschluss Diskretisierungsverfahren vorgestellt, um die Erfolgsaussichten eines rein symbolischen Ansatzes besser beurteilen zu können, bevor auf den für den Generator verwendeten Ansatz eingegangen wird.

10.1.3.1 *Synthetic Classification Data Sets* (SCDS)

SCDS scheint der einzige frei verfügbare Datengenerator zu sein. Zu diesem Schluss kommt man, wenn man sich auf die Suche nach Implementierungen auf diesem Gebiet macht. Man kann diesen Generator über [Melli 1996] abrufen. Im Folgenden soll SCDS vorgestellt und analysiert werden.

Es handelt sich um ein C-Programm, bei dem über eine ganze Reihe von Parametern angegeben werden kann, wie die zu generierenden Daten beschaffen sein sollen. Konkret wird aufgrund der Parameter eine Regelbasis in konjunktiver Normalform aufgestellt, die dann im zweiten Schritt zur Generierung der gewünschten Anzahl von Tupeln benutzt wird.

In Abbildung B.1 werden die Aufrufparameter beschrieben. Es soll hier nicht auf alle Parameter im Einzelnen eingegangen werden, da die prinzipiellen Möglichkeiten auch erläutert werden können, ohne die Bedeutung jedes Schalters zu kennen. Bevor näher auf die Beschaffenheit der Regeln eingegangen wird, sollen noch ein paar grundlegende Eigenheiten von SCDS angesprochen werden, die den Einsatzbereich von SCDS einschränken.

Aufgrund der Implementierung ist SCDS auf maximal 32 Attribute beschränkt, was im Normalfall als nicht sehr gravierend eingestuft werden kann. Nachteilig ist allerdings die Tatsache, dass numerische Attribute nicht berücksichtigt werden können und somit auch nicht generiert werden können. Als Folge müsste man symbolische Attribute generieren und diese im Anschluss geeignet in numerische Attribute umwandeln. Darüber hinaus wäre es schöner, wenn die Attributausprägungen direkt beeinflussbar wären. Bei SCDS ist es bislang so, dass als Ausprägungen der Attribute natürliche Zahlen verwendet werden. Ein direktes Angeben der zulässigen Attributausprägungen nur über Parameter wäre allerdings nicht praktikabel. Dieser Punkt ist aber nur bedingt kritisch, da man ohne großen Aufwand die generierten Daten entsprechend transformieren kann.

Was hier mehr oder weniger als Nachteil dargestellt wurde, muss auf der anderen Seite nicht unbedingt so gewertet werden, da man mit SCDS einen Generator bekommt, der ohne großen Aufwand Daten erzeugt, deren Eigenschaften man einigermaßen beeinflussen kann. Für die meisten Fälle sollte SCDS durchaus praktikabel sein. Wenn

man allerdings "zielgenau" spezielle Charakteristiken treffen möchte, die sich in der Regel auch auf numerische Attribute beziehen, dann sind die Möglichkeiten, die man mit SCDS bekommt, unbefriedigend. Für einen eigenen Ansatz muss man überlegen, wie man die Einschränkungen von SCDS überwinden kann.

Abschließend soll noch auf die Regelbasis von SCDS eingegangen werden. Wie oben bereits erwähnt, werden Regeln konjunktiver Normalform verwendet. Außerdem dient SCDS zur Generierung von Klassifikationsdaten, jede Regel bezieht sich als Konsequenz also auf eine Klasse. Dabei kann es auch mehrere Regeln pro Klasse geben. Die einzelnen Regeln beschreiben somit Attributabhängigkeiten bezogen auf eine Klasse.

Jede Regel besteht aus mehreren Konjunktionen, den sogenannten Termen, die jeweils die Ausprägungen eines Attributs für die betrachtete Regel beschreiben. Ein Term besteht wiederum aus mehreren Disjunktionen, die die zulässigen Ausprägungen für dieses Attribut festlegen.

Ist die Regelbasis aufgrund der eingestellten Parameter "zufällig" aufgestellt worden, wird abhängig von der Aktivierung eine Regel ausgewählt, die dann zur Generierung des aktuellen Tupels genutzt wird. Im zweiten Schritt werden die einzelnen Attribute durchgegangen. Wenn die Regel für dieses Attribut etwas definiert, wird gemäß der Regel die entsprechende Ausprägung gewählt, sonst wird zufällig eine aus allen möglichen Ausprägungen ermittelt. Diese Prozedur wird solange fortgesetzt, bis die gewünschte Anzahl Objekte generiert worden ist.

Man erhält mit SCDS eine einfache Möglichkeit zur Generierung von Klassifikationsdaten, von kleineren Einschränkungen einmal abgesehen. Im folgenden Abschnitt wird in einem kleinen Exkurs auf die Verwendung von Assoziationsregeln eingegangen. Assoziationsregeln werden in diesem Zusammenhang untersucht, um u.a. die Möglichkeiten der automatischen Erzeugung einer Regelbasis zu evaluieren.

10.1.3.2 Verwendung von Assoziationsregeln

Um eine bestehende Datenbank nachbilden zu können, ist es notwendig neben der Struktur der Daten auch die Abhängigkeiten der Attribute untereinander zu berücksichtigen. Deshalb erscheint es auf den ersten Blick sinnvoll, sich aus den zu einer Datenbank berechneten Assoziationsregeln eine Regelbasis aufzubauen. Bevor dieser Gedanke weiter verfolgt wird, sollen Assoziationsregeln kurz erläutert werden.

Am häufigsten werden Assoziationsregeln zur Verbesserung der Absatzpolitik eingesetzt. D.h., man beschäftigt sich z.B. mit Fragestellungen, welche Produkte häufig in Kombination gekauft werden. Dabei werden alle Regeln berechnet, die einen vom Benutzer vorgegebenen minimalen *support* und eine minimale *confidence* übersteigen. Der *support* beschreibt dabei den Grad der Unterstützung der Aussage durch die gegebene Datenbank von Fallbeispielen. Unter der *confidence* versteht man den Anteil der Daten die tatsächlich eine bestimmte Kombination aufweisen. Dieser Zusammenhang soll an einem einfachen Beispiel verdeutlicht werden:

- "30% aller Kunden, die Bier kaufen, kaufen auch Windeln."
- "2% aller Käufe enthalten sowohl Bier als auch Windeln."

Der erste Fall beschreibt die *confidence*, dass 30% derjenigen, die Bier kaufen auch Windeln kaufen. Auf die gesamte Datenbank bezogen, sind es allerdings nur 2% aller Kunden, die wirklich Bier und Windeln in Kombination kaufen.

Das momentan wohl bekannteste Verfahren zur Berechnung von Assoziationsregeln ist *apriori* von Agrawal, das im Rahmen der Arbeit unter Clementine™ eingesetzt wurde. Für eine detaillierte Betrachtung von Assoziationsregeln soll an dieser Stelle auf [Agrawal u. a. 1996] verwiesen werden. Im Folgenden soll nun konkret auf den Aufbau einer Regelbasis unter Verwendung von Assoziationsregeln eingegangen werden.

Wie kann man nun Assoziationsregeln dazu verwenden, eine Regelbasis aufzubauen, um damit eine Datenbank von Fallbeispielen zu generieren? Wenn man Klassifikationsdaten erzeugen möchte, kann man sich alle Assoziationsregeln bezogen auf die Ausprägungen des Klassenattributs berechnen lassen. Die sicherste Methode ist es, die Regeln mit *confidence* Eins, abhängig vom *support* zu verwenden, da die so gefundenen Abhängigkeiten zu 100% in der Datenbank vorhanden sind. In der Praxis erweist sich dieses Vorgehen allerdings als nicht sonderlich geeignet, da man so die Daten nur sehr grob beschreiben kann. Es kann auch passieren, dass bezogen auf das Klassenattribut nur wenig Assoziationsregeln gefunden werden. In diesen Fällen ist es nicht möglich, direkt eine adäquate Regelbasis aus den Assoziationsregeln zu extrahieren. Eine weitere Schwierigkeit besteht in der Anzahl der gefundenen Assoziationsregeln, die exponentiell von der Anzahl der Attribute abhängt.

Hinzukommt, dass Assoziationsregeln lediglich auf symbolischen Werten arbeiten. Man müsste also alle numerischen Attribute diskretisieren, um diesen Ansatz verwenden zu können.

Neben der Diskretisierung muss man sich ebenfalls auch Gedanken über die Rücktransformation der diskretisierten Daten machen. Insgesamt erweist sich die Verwendung von Assoziationsregeln zum Aufbau einer Regelbasis im Allgemeinen als unbefriedigend.

10.1.3.3 Ansatz des Generators

Der umgesetzte Ansatz basiert wie SCDS auf einer Regelbasis. Diese wird aber nicht durch Parameter angenähert und anschließend erzeugt, sondern direkt in Form einer Datei angegeben.

Der Vorteil dieser Methode besteht darin, dass man genauer spezifizieren kann, welche Attributkombinationen bezogen auf eine Klassifikation auftreten sollen. Der Nachteil dabei ist, dass man im Vorfeld die Regeln explizit spezifizieren muss. Neben Klassifikation berücksichtigt der Ansatz auch generelle Abhängigkeiten zwischen Attributen, wodurch sich die Form der Regeln geringfügig ändert. Man muss sich

allerdings im Vorfeld entscheiden, ob man Klassifikationsdaten oder allgemeine relationale Daten erzeugen möchte.

Die Struktur der Datenbank wird auch nicht mehr direkt über Parameter gesteuert, sondern ebenfalls aus einer Datei eingelesen. Wie im Fall der Regeldatei, erlaubt dies eine zielgenauere Generierung.

Die Überlegungen *apriori* zur Gewinnung einer Regelbasis zu verwenden, haben gezeigt, dass dieser Ansatz nur in Einzelfällen verwendbar ist. Außerdem erscheint eine rein symbolische Betrachtung aufgrund des relativ hohen Informationsverlusts durch Diskretisierung und anschließende Rückumwandlung der diskretisierten Attribute als nicht praktikabel.

Deshalb verfolgt der gewählte Ansatz die Zielsetzung einer offenen Regelstruktur, die sowohl Definitionen für symbolische als auch für numerische Attribute umfasst. Zu diesem Zweck wird unter Umständen die Angabe einer Kovarianzdatei erforderlich.

10.2 Aufbau des Datengenerators

10.2.1 Das Konzept des Generators

Wie in Kapitel 10.1.3.3 bereits erläutert, basiert der Generator auf einem kombinierten Regelkonzept, d.h. es wird eine Regelbasis verwendet, die sowohl für symbolische als auch für numerische Attribute Angaben über Verteilungen und Abhängigkeiten erlaubt.

Die Regelbasis wird in Form der *Regeldatei* angegeben. In Abschnitt 10.2.1.2 wird der mögliche Aufbau der *Regeldatei* detailliert beschrieben. Neben den Regeln hat man aber auch die Möglichkeit die Struktur der Daten vorzugeben. Dies geschieht über die *Domaindatei*, die Angaben über die einzelnen Attribute enthält. Unter der Struktur wird allgemein die Art der Ausprägungen der einzelnen Attribute verstanden. In Abschnitt 10.2.1.1 wird hierauf noch genauer eingegangen. Unter anderem werden die für den Generator zulässigen Eingabemöglichkeiten definiert.

Um auch komplexe Abhängigkeiten numerischer Attribute umsetzen zu können, benötigt man z.T. die Angabe der Kovarianzmatrix und der Mittelwerte für die relevanten Attribute. Die Übergabe dieser Informationen findet durch die *Kovarianzdatei* statt. In Abschnitt 10.2.1.3 wird genauer auf den Aufbau der *Kovarianzdatei* eingegangen. Konkret wird die *Kovarianzdatei* zur Generierung der Multinormalverteilung und zur Generierung abhängiger Zufallszahlen verwendet.

Im Anschluss an die Vorstellung der einzelnen Eingabedateien und ihrer Formate, wird in Abschnitt 10.2.2 die Verwendung des Generators beschrieben. Dabei sollen an einem Beispiel die Möglichkeiten veranschaulicht werden.

10.2.1.1 *Domaindatei*

Über die *Domaindatei* wird die Struktur der Datenbank definiert[1]. Unter Struktur versteht man die Definition der einzelnen Attribute. In Abschnitt 10.1.1 und Abschnitt 10.1.2 wurden die beiden Attributarten bereits vorgestellt. Neben symbolischen und numerischen kann man auch noch *ordinale* Attribute betrachten, also symbolische Attribute auf denen eine Ordnung definiert ist. Dieser Datentyp ist theoretisch im Generator vorgesehen, ist allerdings nicht konkret implementiert worden, da kein adäquates Verfahren zur Verfügung stand.

Die folgende Aufzählung beschreibt die zulässigen Attributarten:

- symbolische Attribute:
  - mit/ohne *default*-Verteilung
- numerische Attribute:
  - diskret (*integer*)
    mit/ohne Bereich und *default*-Verteilung
  - kontinuierlich (*continuous*)
    mit/ohne Bereich und *default*-Verteilung

Die *Domaindatei* stellt eine Erweiterung des "names"-Formats von C4.5 dar. Es besteht somit aus einer Menge von Attributdefinitionen, wobei die erste Attributdefinition im Klassifikationsfall als Definition des Klassenattributs eine gewisse Sonderstellung einnimmt. Zunächst sollen "normale" Attributdefinitionen vorgestellt werden, später wird auf den Klassifikationsfall eingegangen.

Eine Attributdefinition beinhaltet neben dem Attributnamen bei symbolischen Attributen eine Aufzählung der einzelnen Attributausprägungen. Diese können mit einer *default*-Aktivierung aus $(0, 1)$ versehen sein.

Bei numerischen Attributen folgt nach dem Attributnamen das Schlüsselwort *integer* oder *continuous*. Anschließend wird die Bereichsdefinition und/oder eine *default*-Verteilung angegeben.

Unter einer *default*-Verteilung wird allgemein die Verteilung verstanden, die der Generator verwenden soll, wenn in der aktuellen Regel für das angegebene Attribut nichts spezifiziert ist. Dies gilt sowohl für symbolische als auch für numerische Attribute.

Bei der *default*-Verteilung für ein numerisches Attribut kann es sich um Zahlwerte, Verteilungen oder *missing values* handeln. Die Verteilungen werden über einen Buchstaben für die entsprechende Verteilung und die zugehörigen Parameter angegeben. In Tabelle 10.1 und 10.2 sind alle möglichen univariaten Verteilungen

[1] Im Rahmen des Beispiels in Abschnitt 10.2.2 wird in Abbildung 10.2 eine mögliche *Domaindatei* angegeben.

aufgeführt. Desweiteren kann, wie im Fall der symbolischen Attribute, eine *default*-Aktivierung für die einzelnen Werte der *default*-Verteilung über die Angabe von Werten aus $(0, 1)$ angegeben werden. Es ist allerdings nicht notwendig, für alle Werte die Aktivierung anzugeben, da die "Restaktivierung"[2] auf die Werte, für die keine Aktivierung angegeben wurde, gleichmäßig aufgeteilt wird.

Code	Name der Verteilung	Parameter
B	Binomialverteilung	$n \in \mathbb{N},\ p \in [0,1]$
P	Poisson-Verteilung	$\lambda \in \mathbb{R}^+$
G	Geometrische Verteilung	$p \in [0,1]$
H	Hypergeometrische Verteilung	$N, k, n \in \mathbb{N}$
p	Pascal-Verteilung	$p \in [0,1],\ k \in \mathbb{N}$

Tabelle 10.1: Diskrete univariate Verteilungen

Code	Name der Verteilung	Parameter
N	Normalverteilung	$\mu \in \mathbb{R},\ \sigma \in \mathbb{R}^+$
L	Lognormalverteilung	$\mu \in \mathbb{R},\ \sigma \in \mathbb{R}^+$
C	Cauchy-Verteilung	$\mu \in \mathbb{R},\ \sigma \in \mathbb{R}^+$
U	Gleichverteilung	$min, max \in \mathbb{R}$
E	Exponentialverteilung	$\lambda \in \mathbb{R}^+$
W	Weibull-Verteilung	$\eta, \sigma \in \mathbb{R}^+$
e	Erlang-Verteilung	$\lambda \in \mathbb{R}^+,\ k \in \mathbb{Z}$
g	Gammaverteilung	$\lambda, \eta \in \mathbb{R}^+$
b	Betaverteilung	$a, b \in \mathbb{R}^+$
l	Logistische Verteilung	$a, b \in \mathbb{R}^+$

Tabelle 10.2: Stetige univariate Verteilungen

Abschließend soll jetzt noch die Besonderheit des Klassenattributs angesprochen werden. Es handelt sich beim Klassenattribut immer um ein symbolisches Attribut. Es gelten also alle Angaben, die weiter oben zu symbolischen Attributen gemacht wurden. Die einzige Ausnahme besteht darin, dass die Angabe des Namens inklusive Doppelpunkt weggelassen werden kann. In diesem Fall wird "*class*" als *default*-Name für das Klassenattribut gesetzt.

Mit den oben beschriebenen Möglichkeiten ist man in der Lage, die Struktur komplexer Datenbanken von Fallbeispielen zu beschreiben. Im nächsten Abschnitt soll die *Regeldatei* genauer betrachtet werden[3].

[2] Also die Differenz aus Eins und der Summe der angegebenen Aktivierungen.

[3] Im Rahmen des Beispiels in Abschnitt 10.2.2 wird in Abbildung 10.3 eine mögliche *Regeldatei* angegeben.

10.2.1.2 *Regeldatei*

Die *Regeldatei* kann als Erweiterung der Regelstruktur von Gabor Mellis SCDS verstanden werden. Zumindest der grundsätzliche Aufbau ist ähnlich. Die Möglichkeiten der *Regeldatei* sind allerdings weitreichender. Man ist zum einen nicht nur auf Klassifikationsdaten beschränkt und zum anderen können auch konkrete Verteilungen für numerische Attribute vorgegeben werden.

Im einzelnen besteht eine *Regeldatei* aus einer Menge von Regeln, die Abhängigkeiten der Attribute untereinander oder bezogen auf das Klassenattribut definieren.

Eine Regel beginnt mit der Regelaktivierung, wobei die Summe aller Regelaktivierungen 100 % ergeben muss. Danach steht im Klassifikationsfall die Ausprägung des Klassenattributs, auf das sich die Regel bezieht. Anschließend kann eine Menge von Attributtermen angegeben werden. Diese beinhalten neben dem Attributnamen eine Menge möglicher Ausprägungen, die für diese Regel erlaubt sein sollen. Optional kann für numerische Attribute nach dem Attributnamen auch eine Bereichsdefinition angegeben werden.

Man hat auch hier die Möglichkeit, eine Aktivierung anzugeben. Die zulässigen Werte hängen vom Attributtyp ab. Bei symbolischen Attributen sind lediglich einzelne Attributausprägungen, ggf. mit Aktivierung, zulässig. Bei numerischen Attributen können sowohl Zahlwerte als auch Verteilungen angegeben werden. In beiden Fällen können natürlich auch *missing values* angegeben werden.

10.2.1.3 *Kovarianzdatei*

Die *Kovarianzdatei* wird zur Generierung der Multinormal-Verteilung und zur Generierung abhängiger Zufallszahlen benötigt[4]. Sie besteht aus den Namen der numerischen Attribute, für die die Mittelwerte und die Kovarianzmatrix angegeben werden.

Im nachfolgenden Abschnitt wird nun auf die Verwendung des Generators eingegangen und ein konkretes Beispiel mit Angabe der entsprechenden Dateien betrachtet.

10.2.2 Verwendung des Generators

In diesem Abschnitt soll auf die Verwendung des Generators eingegangen werden. In Abbildung 10.1 werden zunächst die möglichen Parameter dargestellt, wie man sie auf der Shellebene angeben kann.

Die Angabe einer *Domaindatei* und einer *Regeldatei* sind Grundvoraussetzungen für den Einsatz des Generators. Alle anderen Schalter sind optional. Über den Schalter "-v" kann man sich beispielsweise ausführliche Informationen zu den eingelesenen

[4]Im Rahmen des Beispiels in Abschnitt 10.2.2 wird in Abbildung 10.4 eine mögliche *Kovarianzdatei* angegeben.

```
Version 5.20                                        07/16/1998

SYNTAX: generator [-hvcsp] -D 'domain-file' -R 'rule-file'
                  [-C 'covar-file'] [-d] [-O #objects]
                  [-f 'output-file']

        h:      help (this report) [default]
        v:      verbose report [false]
        c:      build classification data [false]
        s:      print Attributenames [false]
        p:      pseudo randomness [false]
                use '?' to code missing values

        D:      name of the domain file
        R:      name of the rule file

        C:      name of the covariance and mean file
        d:      use dependant random numbers
        O:      number of objects
        f:      output filename [stdout]
```

Abbildung 10.1: Beschreibung der Aufrufparameter des Generators

Strukturen und der Generierung anzeigen lassen. Ein wichtiger Schalter ist das "-c" über den gesteuert wird, ob Klassifikationsdaten erzeugt werden sollen. (Vgl. hierzu Abschnitt 10.2.1.2.) Bei jedem Lauf mit Schalter "-p" werden dieselben Zufallszahlen verwendet. Ohne den Schalter werden bei jedem Lauf jeweils andere Zufallszahlen verwendet. Man erhält also die Möglichkeit, reproduzierbare Daten zu erzeugen.

Möchte man eine Multinormal-Verteilung für bestimmte numerische Attribute generieren, muss man über die Option "-C" eine entsprechende *Kovarianzdatei* angeben. Um abhängig generierte Zufallszahlen zu verwenden, muss neben der *Kovarianzdatei* noch der Schalter "-d" aktiviert werden. Der Generator versucht dann, auf Basis der Kovarianzdatei erzeugte abhängige Zufallszahlenvektoren zur Generierung der in der *Regeldatei* bzw. *Domaindatei* angegebenen Verteilungen zu verwenden.

Nachdem exemplarisch einige Einstellmöglichkeiten des Generators vorgestellt sind, soll anhand eines Beispiels die Funktionsweise des Generators veranschaulicht werden.

10.2.2.1 Beispiel

Abbildung 10.2 gibt zunächst eine mögliche *Domaindatei* an.

In diesem Beispiel sollen Klassifikationsdaten generiert werden. In der *Domaindatei* darf deshalb in der ersten Attributdefinition der Name des Klassenattributs fehlen. Neben dem Klassenattribut werden sechs weitere Attribute definiert. Das Beispiel

```
# ------------------------------------------------------------ #
#  domain file example
#
#    "classification"
#
# ------------------------------------------------------------ #
# class attribute
yes, no, maybe

# predicting attributes
attribute1:  red, green, blue
attribute2:  continuous [3.678, 96]
attribute3:  continuous [-20,50] {[N,0,1],?;0.03}
  # remark: attribute3 is defined as a real range from -20 to 50
  #         has a default distribution consisting of a normal
  #         distribution with mean 0, standard deviation 1
  #         and an implicit activation of 0.97,
  #         and a missing value encoded  by ? with an
  #         activation of 0.03
attribute4:  value1;0.8, value2, value3, value4, value5;0.0001
attribute5:  integer
attribute6:  t,f;0.7
# ------------------------------------------------------------ #
#  End of domain file example
# ------------------------------------------------------------ #
```

Abbildung 10.2: Beispiel einer *Domaindatei*.

ist nicht darauf ausgelegt, inhaltlich besonders wertvolle Daten zu erzeugen. Es geht vielmehr darum, die Möglichkeiten der Datenerzeugung zu veranschaulichen.

In Abbildung 10.3 wird eine entsprechende *Regeldatei* dargestellt, die drei Regeln mit unterschiedlichen Aktivierungen für je ein Klassenattribut angibt. Bei genauerem Hinsehen stellt man fest, dass mehrere Regeln für die Klasse *yes*, aber keine Regel für die Klasse *no* angegeben sind, was in dieser Form zulässig ist.

```
# ------------------------------------------------------------- #
#  rule file example
#
#    "classification"
#
# ------------------------------------------------------------- #

# rule 1:
(75.37%)  yes  <-  attribute1 = red & \
                   attribute3 = (42;0.9,0.815,-12) & \
                   attribute6 = t

# rule 2:
(4.63%)  maybe <-  attribute5 = (1,5,23,18)

# rule 3:
(20%)    yes   <-  attribute1 = (blue;0.1,green;0.7,?) & attribute5 = 42

# ------------------------------------------------------------- #
#  End of rule file example
# ------------------------------------------------------------- #
```

Abbildung 10.3: Beispiel einer *Regeldatei*.

Es ist auch nicht für alle Attribute in jeder Regel etwas angegeben. Der Generator greift in solchen Fällen auf die *default*-Verteilungen aus der *Domaindatei* zurück. Des weiteren sind nur sporadisch Aktivierungen für einzelne Attributwerte angegeben, auch dies ist zulässig. In diesem Fall wird die "Restaktivierung" gleichmäßig auf die übrigen Attributwerte aufgeteilt.

In Abbildung 10.4 wird eine *Kovarianzdatei* angegeben. Sie gibt in diesem Fall für zwei der drei numerischen Attribute die Mittelwerte und die Kovarianzmatrix vor. Die angegebene *Kovarianzdatei* soll in einem der folgenden Beispielaufrufe zur Generierung einer Multinormal-Verteilung für die Attribute *attribute2* und *attribute3* benutzt werden.

```
# ------------------------------------------------------------- #
#  covariance file example
# ------------------------------------------------------------- #

# relevant attribute names:
attribute2, attribute3

# means:
-2, 0

# covariance matrix:
1.00, 0.63
0.63, 1.00


# ------------------------------------------------------------- #
#  End of covariance file example
# ------------------------------------------------------------- #
```

Abbildung 10.4: Beispiel einer *Kovarianzdatei*.

Nachdem nun die einzelnen Dateien vorgestellt wurden, sollen nun entsprechende Generatoraufrufe aufgezeigt werden. Es sollen zwei Aufrufe dargestellt werden: Zum einen ein Aufruf ohne Benutzung der *Kovarianzdatei* und zum anderen ein Aufruf unter Verwendung der *Kovarianzdatei* zur Generierung einer Multinormal-Verteilung.

generator -cp -D test.domain -R test.rules -O 10000

generator -cp -D test.domain -R test.rules -O 10000 -C test.covar

10.3 Evaluierung des Datengenerators

In diesem Kapitel wird die Evaluierung des Generators durchgeführt. Dabei wird zunächst die Generierung einzelner Attribute untersucht. Dies geschieht im Zusammenhang mit der Generierung von synthetischen Datenbanken von Fallbeispielen. Speziell werden Untersuchungen bezogen auf mögliche Verteilungen numerischer Attribute angesprochen. Eine genauere Untersuchung symbolischer Attribute wird im Rahmen der Reproduktion bestehender Datenbanken durchgeführt.

In Abschnitt 10.3.2 wird die Reproduktion bestehender Datenbanken von Fallbeispielen untersucht. Dabei wird zunächst auf die Verwendung der Ergebnisse des DCT eingegangen. Anschließend wird kurz die Reproduktion der Struktur einer Datenbank beleuchtet. Desweiteren werden symbolische und numerische Attribute betrachtet und es wird kurz die Problematik gemischter Verteilungen angesprochen.

Die Abschnitte 10.3.3.1, 10.3.3.2 und 10.3.3.3 zeigen an konkreten Datenbanken die Reproduktionsmöglichkeiten. Dabei wurde eine rein symbolische, eine rein numerische und eine gemischte[5] Datenbank ausgewählt. Es werden jeweils auch die Ergebnisse des DCT für das Original und die Reproduktion auszugsweise besprochen und angegeben.

Den Abschluss dieses Kapitels bildet eine Zusammenfassung, in der noch einmal die Ergebnisse der Evaluierung dargestellt werden.

10.3.1 Generierung synthetischer Datenbanken von Fallbeispielen

In diesem Abschnitt wird kurz auf die Generierung rein synthetischer Datenbanken von Fallbeispielen eingegangen. Dabei wird betrachtet, wie gut spezielle univariate Verteilungen getroffen werden können.[6]

10.3.1.1 Generierung symbolischer Attribute

Die Generierung symbolischer Attribute kann entweder durch die Regelbasis oder über eine *default*-Verteilung in der *Domaindatei* gesteuert werden. In Abschnitt 10.3.2 wird ausführlich auf die Reproduktion einer rein symbolischen Datenbank von Fallbeispielen eingegangen.

10.3.1.2 Generierung numerischer Attribute

Grundsätzlich ist festzuhalten, dass mit steigender Anzahl der Beispiele auch die Qualität der generierten Verteilung zunimmt. Verteilungen mit einer großen Spannweite benötigen dabei u. U. mehrere 10000 Beispiele, um die gewünschte Verteilung ausreichend gut zu repräsentieren.

Hinweisen muss man noch auf die implementierte Gleichverteilung, die bei großer Varianz schlechte Resultate liefert. Dieses Phänomen lässt sich durch eine in diesen Extremfällen nicht ausreichende Rechengenauigkeit erklären. Die Gleichverteilung auf dem Intervall (a, b) basiert auf einer auf $(0, 1)$ gleichverteilten Zufallsvariablen. Bei der Transformation wird eine "sehr große" Zahl $(a + b)$ mit einer Zahl aus $(0, 1)$ multipliziert, dabei können nicht alle Stellen berücksichtigt werden, so dass das Ergebnis ab einer bestimmten Nachkommastelle abgeschnitten wird. Die Folge ist, dass man lediglich an einigen diskreten Stellen Datenpunkte erhält. Schlimmer ist die Tatsache, dass der Bereich um Null unterrepräsentiert ist. Tritt dieses Phänomen auf, kann man sich helfen, indem man den Bereich in mehrere Gleichverteilungen mit entsprechend kleinerer Varianz aufteilt.

[5] Also eine Datenbank von Fallbeispielen sowohl mit symbolischen als auch numerischen Attributen.

[6] Abhängigkeiten zwischen verschiedenen Attributen werden in Abschnitt 10.3.2 behandelt.

10.3.2 Reproduktion bestehender Datenbanken von Fallbeispielen

10.3.2.1 Verwendung der Ergebnisse des DCT

Für die Auswertungen wurde DCT verwendet. Zu Testzwecken wurde die Generierung eines *Data-Dictionaries*, einer *Regeldatei* und einer *Kovarianzdatei* in den DCT integriert. Die Dateien können über die Option "-m" oder "-M" erzeugt werden[7].

Möchte man die oben genannten Dateien dazu verwenden, den Generator aufzurufen, muss ggf. die Reihenfolge der Attributdefinitionen im *Data-Dictionary* angepasst werden, da der DCT die Attributdefinitionen getrennt für symbolische und numerische Attribute ausgibt.

Das *Data-Dictionary* enthält bereits *default*-Verteilungen für die symbolischen Attribute, die allerdings auskommentiert sind. Für die numerischen Attribute ist jeweils die Wertemenge und der Wertebereich angegeben. Die Wertemenge ist für reellwertige Attribute mit *continuous* und für ganzzahlige Attribute mit *integer* gekennzeichnet. Eine *default*-Verteilung für die numerischen Attribute wird nicht angegeben, da eine konkrete Verteilung der numerischen Attribute nicht direkt aus den Daten abgelesen werden kann.[8]

Die vom DCT erzeugte Regeldatei enthält eine Regel je Klasse. Jede Regel besteht aus den Häufigkeitsverteilungen für alle symbolischen Attribute, bezogen auf die jeweilige Klasse. Die numerischen Attribute werden hierbei nicht berücksichtigt.

Für die numerischen Attribute wird die Kovarianzdatei erzeugt. Diese enthält die Attributnamen aller numerischen Attribute, ihre Mittelwerte und die Kovarianzmatrix. Diese Datei kann dazu verwendet werden, um entweder eine Multinormalverteilung der numerischen Attribute zu generieren oder um abhängige Zufallszahlen zur Generierung der angegebenen Verteilungen zu verwenden.[9] Entsprechende Generatoraufrufe wurden in Abschnitt 10.2.2 vorgestellt.

10.3.2.2 Struktur der Datenmenge

Unter der Struktur einer Datenmenge von Beispielen versteht man u.a. Informationen zur Anzahl der Attribute und zur Art der jeweiligen Ausprägungen. Diese Informationen lassen sich direkt über die *Domaindatei* an den Generator weitergeben. Diese grundlegenden Informationen einer Datenbank von Fallbeispielen können bei der Reproduktion leicht getroffen werden, da die Strukturinformationen im Generator als Basis für die Generierung der Daten genutzt werden. Folglich können auch nur Daten generiert werden, die diesen Strukturen genügen.

[7] Bei Verwendung des DCT in Clementine™ entspricht die Option "-m" dem Schalter *Save Measures*.

[8] Eine Ausnahme bildet die Kovarianzdatei, die das direkte Generieren einer Multinormalverteilung ermöglicht.

[9] Vgl. hierzu Abschnitt 10.2.1.3.

10.3.2.3 Betrachtung symbolischer Attribute

Das Generieren symbolischer Attribute inklusive gewisser Abhängigkeiten stellt prinzipiell für den Generator kein großes Problem dar. Die *Domaindatei* in Kombination mit der *Regeldatei* liefert gute Möglichkeiten zur Generierung symbolischer Attribute. Wie weiter unten noch am Beispiel der *nursery*-Datenbank nachgewiesen wird, lassen sich spezielle Häufigkeitsverteilungen in Kombination mit den berechneten informationstheoretischen Maßen nahezu exakt nachbilden. Einzige Voraussetzung ist eine ausreichend hohe Anzahl Beispiele. Dabei hängt die benötigte Anzahl Beispiele von der Komplexität der Datenbank ab.

Man muss allerdings dazu sagen, dass lediglich die Abhängigkeiten bezogen auf die Klassifikation betrachtet wurden. Um eine Datenbank wirklich exakt zu reproduzieren, wird es notwendig auch besondere Abhängigkeiten zwischen einzelnen Attributen untereinander zu berücksichtigen. Auf diesen Punkt wird weiter unten näher eingegangen.

Mit Hilfe des DCT lässt sich die Generierung einer rein symbolischen Datenbank von Fallbeispielen fast vollständig automatisieren. Um die informationstheoretischen Maße und die Häufigkeitsverteilungen bezogen auf die gesamte Datenbank und die Klassifikation vorzugeben, reichen die Informationen aus der *Domain*- und der *Regeldatei* aus. Existieren starke Abhängigkeiten zwischen einzelnen Attributen, die sich nicht auf die Klassifikation beziehen, können diese aber auch berücksichtigt werden. Dazu muss man die betroffenen Regeln der *Regeldatei* vervielfältigen, die Abhängigkeiten der Attribute integrieren und die Aktivierung der Regeln entsprechend anpassen.

Wie kann man solche Abhängigkeiten entdecken? Eine Möglichkeit bietet die Verwendung von Assoziationsregeln[10]. Auf diese Weise kann man Attributkombinationen auffinden, die für die Beschreibung der Datenbank wichtig sind. Dieser Schritt lässt sich allerdings nicht ohne weiteres automatisieren, da in der Regel eine Fülle von Assoziationsregeln berechnet werden.

10.3.2.4 Betrachtung numerischer Attribute

Die exakte Generierung numerischer Attribute erfordert sehr viel mehr Aufwand. In Abschnitt 10.1.2 wurden bereits Verfahren zur Dichteschätzung vorgestellt. Desweiteren wurde dort am Beispiel von Histogrammen eine spezielle graphische Möglichkeit der Dichteschätzung veranschaulicht. Dies ist eine Möglichkeit, die Verteilung eines numerischen Attributs zu ermitteln.

Im Folgenden soll zunächst auf gemischte Verteilungen eingegangen werden, um die Problematik des Auffindens der zugrundeliegenden Verteilung aufzuzeigen.

[10] Vgl. Abschnitt 10.1.3.2.

10.3.2.5 Gemischte Verteilungen

An einem einfachen Beispiel soll die Bedeutung von gemischten Verteilungen veranschaulicht werden. Abbildung 10.5 zeigt die Überlagerung zweier Normalverteilungen, die jeweils eine bestimmte Klasse eines Attributs repräsentieren sollen.

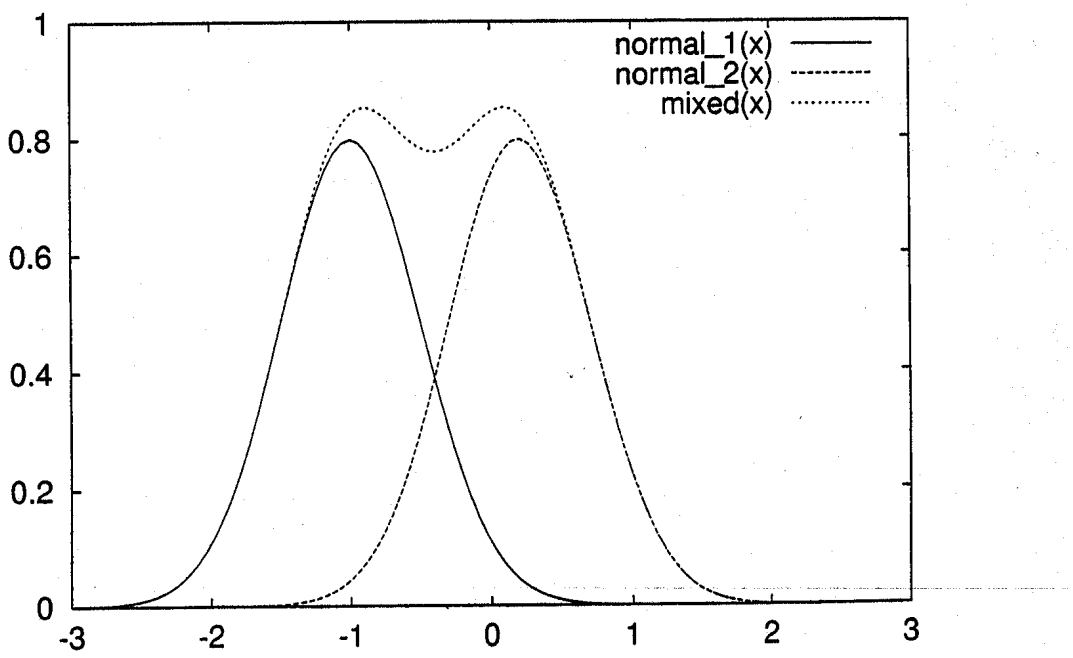

Abbildung 10.5: Überlagerung von zwei Normalverteilungen

Es ergibt sich eine nahezu unimodale Verteilung. Solange dieser Effekt bezogen auf die Klassifikation auftritt und die Anzahl der Beispiele ausreicht, um die Verteilungen bezogen auf die Klassen zu erkennen, ist die Generierung relativ einfach. Tritt eine solche gemischte Verteilung bezogen auf ein drittes Attribut auf, ist die Modellierung dem Zufall überlassen.

Aber auch bezogen auf die Klassifikation können komplexe gemischte Verteilungen, wie man sie in realen Datenbanken häufig findet, Probleme bereiten. In

[Lindsay 1995] und [Cornell 1990] findet man genauere Betrachtungen zu *Mixture* Problemen im Allgemeinen.

Insgesamt kann man festhalten, dass man auf die Kombination mehrerer Verteilungen angewiesen ist, wenn man aufgrund eines meist relativ kleinen Samples numerische Attribute möglichst exakt nachbilden möchte. Erschwerend kommt hinzu, dass, selbst wenn man die zugrundeliegende Verteilung kennt, eine zu geringe Anzahl Beispiele das zielgenaue Erreichen der berechneten Maße vereiteln kann.

In den nachfolgenden Abschnitten sollen drei Datenbanken von Fallbeispielen reproduziert werden. In Tabelle 10.3 sind einige grundlegende Angaben zu den verwendeten Datenbanken dargestellt. Es wurden im Rahmen der Arbeit noch andere Datenbanken aus dem UCI-Repository [Merz und Murphy 1996] und Datenbanken der Daimler Benz AG untersucht. Es sollen hier jedoch nur exemplarisch die erzielten Ergebnisse vorgestellt werden.

Name	Attribute	symbolisch	numerisch	Klassen	Beispiele
nursery	9	8	0	5	12960
breastLoss	11	0	10	2	699
crx	16	9	6	2	653

Tabelle 10.3: Angaben zu den reproduzierten Datenbanken

10.3.3 Beispiel-Datenbanken

10.3.3.1 *nursery*-Datenbank

Im folgenden soll eine rein symbolische Datenbank von Fallbeispielen reproduziert werden. Die Wahl fiel dabei auf die *nursery*-Datenbank aus dem UCI-Repository [Merz und Murphy 1996]. Es handelt sich hierbei um eine medizinische Datenbank, die aus acht symbolischen Attributen und einem Klassenattribut mit fünf Ausprägungen besteht.

Die Anzahl der Beispiele ist mit 12960 verhältnismäßig groß. Dieser Umstand erleichtert die Reproduktion, da die zugrundeliegende Häufigkeitsverteilung deutlicher ausgeprägt ist. Mit Hilfe des DCT wurden eine *Domaindatei* und eine *Regeldatei* erzeugt, die die jeweiligen Häufigkeitsverteilungen beschreiben. Dies gilt sowohl bezogen auf die gesamte Datenbank, als auch auf die einzelnen Klassen.

Erzeugt man die gleiche Anzahl Beispiele wie beim Original mit dem Generator, ergibt sich die gleiche Häufigkeitsverteilung der einzelnen Attribute bezogen auf die gesamte Datenbank und auf die Klassen. Mit Ausnahme der Klasse *very_rec* sind alle Häufigkeiten exakt gleich oder bis auf maximal 1% vom gewünschten Ergebnis entfernt. Eine Erklärung für die größere Abweichung der Klasse *very_rec* ist die mit 2.5% sehr geringe Auftrittshäufigkeit dieser Klasse.

Die informationstheoretischen Maße (vgl. Abschnitt 6.3) können, auf zwei Nachkommastellen gerundet, exakt nachgebildet werden. Als einzige Ausnahme sind das Relevanzmaß des Attributs *has_nurse* mit 0.309 anstatt 0.326 und das Relevanzmaß des Attributs *social* mit 0.145 anstatt 0.102 zu nennen. Abbildung B.3 zeigt die vom DCT berechneten informationstheoretischen Maße für die Original *nursery*-Datenbank. In Abbildung B.4 werden die gleichen Maße für die reproduzierte *nursery*-Datenbank angegeben.

Dieses Beispiel zeigt, dass es ohne größeren Aufwand möglich ist, rein symbolische Datenbanken von Fallbeispielen zu reproduzieren; unter der Bedingung, dass die Anzahl der Beispiele ausreichend groß ist.

Eine Einschränkung besteht allerdings in der Tatsache, dass lediglich Abhängigkeiten bezogen auf die Klassifikation und die gesamte Datenbank berücksichtigt wurden. Um eine symbolische Datenbank von Fallbeispielen "vollständig" zu reproduzieren, müsste man die Regeln für die einzelnen Klassen so aufsplitten, dass wichtige Abhängigkeiten unter den Attributen mit berücksichtigt werden. Solche Abhängigkeiten kann man z.B. mit Hilfe von apriori (Vgl. Abschnitt 10.1.3.2) ermitteln.

10.3.3.2 *breastLoss*-Datenbank

Die *breastLoss*-Datenbank ist ebenfalls dem UCI-Repository entnommen. Es handelt sich wiederum um eine medizinische Datenbank, bei der Daten von Brustkrebspatienten gesammelt wurden. Die Attribute beschreiben den Grad von bestimmten Befunden. Eine Ausnahme bildet die *Sample_code_number*, die jedem Beispiel eine eindeutige Identifikationsnummer zuordnet. Konkret liegen die Werte der *Sample_code_number* zwischen 63375 und 13454352. Wie in den anderen Fällen handelt es sich auch hier um eine Klassifikationsdatenbank, bei der in diesem Fall zwei Klassen unterschieden werden.

Aufgrund der hohen Varianz und aufgrund der Tatsache, dass es sich bei der *Sample_code_number* um eine Identifikationsnummer handelt, wurde dieses Attribut aus der zu reproduzierenden Datenbank entfernt. Hätte man die *Sample_code_number* mit berücksichtigt, hätte dies zu einer erheblichen Verfälschung der multivariaten Maße geführt.

Am Beispiel der *breastLoss*-Datenbank soll nun die Reproduktion einer rein numerischen Datenbank veranschaulicht werden. Zunächst wurde die Struktur der Datenbank nachgebildet und Verteilungen für die einzelnen numerischen Attribute modelliert.

Am einfachsten lassen sich die Wertebereiche der Attribute angeben, bezogen auf die Datenbank und die jeweiligen Klassen. Auch die Mittelwerte und die Standardabweichung konnten in den meisten Fällen gut getroffen werden. Da für die möglichen Verteilungen die entsprechenden Momente bekannt sind, muss man lediglich bei kombinierten Verteilungen die Wechselwirkungen aufeinander abstimmen. Als erschwerend erwies sich die mit 699 relativ geringe Anzahl Beispiele. Es war nicht

möglich zu entscheiden, ob gewisse Bereiche ein spezielles Muster der Daten beschreiben oder ob das gezogene Sample zu klein ist, um die zugrundeliegende Verteilung offenzulegen. Da es bei der Reproduktion um das möglichst genaue Nachbilden einer gegebenen Verteilungsstruktur geht, mussten sich die Verteilungen sehr nah an den Daten orientieren. Abbildung 10.6 und 10.7 zeigen jeweils einen Auszug aus den Ergebnissen des DCT für das Original und für die Reproduktion.

```
*** Attribute Statistics ***

Attribute: Clump_Thickness
          nr_exs     Min      Max     Mean   StdDev     Skew     Kurt    Range
DS:          699   1.000   10.000    4.418    2.814    0.592    2.372    9.000
malignan     241   1.000   10.000    7.195    2.424   -0.417    2.142    9.000
benign       458   1.000    8.000    2.956    1.672    0.341    2.208    7.000

Attribute: Uniformity_of_Cell_Size
          nr_exs     Min      Max     Mean   StdDev     Skew     Kurt    Range
DS:          699   1.000   10.000    3.134    3.049    1.230    3.090    9.000
malignan     241   1.000   10.000    6.573    2.714   -0.082    1.722    9.000
benign       458   1.000    9.000    1.325    0.907    4.200   26.268    8.000

Attribute: Uniformity_of_Cell_Shape
          nr_exs     Min      Max     Mean   StdDev     Skew     Kurt    Range
DS:          699   1.000   10.000    3.207    2.970    1.159    2.998    9.000
malignan     241   1.000   10.000    6.560    2.557   -0.035    1.810    9.000
benign       458   1.000    8.000    1.443    0.997    3.008   13.928    7.000
```

Abbildung 10.6: Auszug der Attributmaße der Original *breastLoss*-Datenbank.

Die mittlere Schiefe ist bezogen auf alle Attribute mit 2.077 anstatt 2.324 relativ gut getroffen. Die mittlere Wölbung ist nicht ganz so gut getroffen, anstatt 17.423 wurde 12.072 berechnet. Dieser Wert ließe sich jedoch durch eine noch exaktere Nachbildung der einzelnen Attribute verbessern. An den Konfidenzintervallen für die Mittelwerte kann man deutlich erkennen, dass das Attribut *Clump_Thickness* lediglich bezogen auf die gesamte Datenbank modelliert wurde und nicht speziell auf jede der beiden Klassen abgestimmt wurde. Diese Ungenauigkeit schlägt sich auch auf die Quantile durch. Die Abbildungen 10.8 und 10.9 zeigen einen Auszug der berechneten Quantile.

Neben den univariaten statistischen Maßen wurden auch die folgenden Eigenschaften der *breastLoss*-Datenbank reproduziert. Das Attribut *Mitoses* wurde sowohl im Original als auch in der Reproduktion als numerisches Attribut mit Ausreißern erkannt. Außerdem wurden die *missing values* des Attributs *Bare_Nuclei* nachgebildet.

Im folgenden sollen noch einige der multivariaten Maße betrachtet werden. Dabei muss berücksichtigt werden, dass durch Ungenauigkeiten auf der Attributebene eine stärkere Streuung der multivariaten Maße vorprogrammiert ist. Die Abbildungen

```
*** Attribute Statistics ***

Attribute: Clump_Thickness
          nr_exs     Min      Max     Mean   StdDev     Skew     Kurt    Range
DS:          699   1.000   10.000    5.552    2.652   -0.055    1.853    9.000
malignan     241   1.000   10.000    5.676    2.660   -0.140    1.823    9.000
benign       458   1.000   10.000    5.487    2.646   -0.010    1.878    9.000

Attribute: Uniformity_of_Cell_Size
          nr_exs     Min      Max     Mean   StdDev     Skew     Kurt    Range
DS:          699   1.000   10.000    3.300    3.227    1.102    2.649    9.000
malignan     241   1.000   10.000    6.938    2.745   -0.414    2.023    9.000
benign       458   1.000   10.000    1.386    1.139    4.642   27.870    9.000

Attribute: Uniformity_of_Cell_Shape
          nr_exs     Min      Max     Mean   StdDev     Skew     Kurt    Range
DS:          699   1.000   10.000    3.114    3.003    1.239    3.128    9.000
malignan     241   1.000   10.000    6.481    2.639    0.007    1.719    9.000
benign       458   1.000   10.000    1.343    0.999    4.949   33.208    9.000
```

Abbildung 10.7: Auszug der Attributmaße der reproduzierten *breastLoss*-Datenbank.

B.5 und B.6 zeigen eine Auswahl multivariater Maße für das Original und für die Reproduktion.

Nachdem gezeigt wurde, dass sich die Datenbank relativ gut reproduzieren lässt, soll im folgenden die Reproduktion durch eine Multinormalverteilung auf Basis der Kovarianzstruktur untersucht werden.

Um die Multinormalverteilung an einem realen Beispiel zu testen, wurde die *breastLoss*-Datenbank mit der vom DCT erzeugten *Kovarianzdatei* reproduziert. Dabei muss man beachten, dass lediglich die globale Kovarianzstruktur berücksichtigt werden kann. Abhängigkeiten, bezogen auf die einzelnen Klassen, werden nur bedingt berücksichtigt. Außerdem können die Bereiche nicht immer eingehalten werden, da sonst die einzelnen Normalverteilungen nicht korrekt generiert werden können. Auch in diesem Beispiel ergibt sich eine Verschiebung der Wertebereiche der Attribute. Wenn man von diesen Einschränkungen absieht, zeigt sich, dass die Abhängigkeiten der Attribute, bezogen auf die gesamte Datenbank, nahezu exakt nachgebildet werden können. Die Abbildungen 10.10 und 10.11 zeigen die globale Korrelationsmatrix der Originaldatenbank und der multinormalverteilt reproduzierten *breastLoss*-Datenbank.

10.3.3.3 *crx*-Datenbank

Im folgenden soll die *crx*-Datenbank reproduziert werden. Wie erwähnt, stammt diese Datenbank ebenfalls aus dem UCI-Repository. Hervorzuheben ist, dass es sich

```
*** location and dispersion parameter ***
                          lower                  upper  alpha-trimmed  inter quar-
fieldname               quartile     median   quartile       mean      tile range
Clump_Thickness            2.000      4.000      6.000         4.301        4.000
Uniformity_of_Cell_Size    1.000      1.000      5.000         2.880        4.000
Uniformity_of_Cell_Shape   1.000      1.000      5.000         2.960        4.000
                       deviation alpha stddev
fieldname               of median  of median
Clump_Thickness            2.000      0.902
Uniformity_of_Cell_Size    0.000      0.895
Uniformity_of_Cell_Shape   0.000      0.888
```

Abbildung 10.8: Auszug der Quantile der Original *breastLoss*-Datenbank.

```
*** location and dispersion parameter ***
                          lower                  upper  alpha-trimmed  inter quar-
fieldname               quartile     median   quartile       mean      tile range
Clump_Thickness            3.000      6.000      8.000         5.558        5.000
Uniformity_of_Cell_Size    1.000      1.000      5.000         3.063        4.000
Uniformity_of_Cell_Shape   1.000      1.000      5.000         2.857        4.000
                       deviation alpha stddev
fieldname               of median  of median
Clump_Thickness            2.000      0.893
Uniformity_of_Cell_Size    0.000      0.918
Uniformity_of_Cell_Shape   0.000      0.889
```

Abbildung 10.9: Auszug der Quantile der reproduzierten *breastLoss*-Datenbank.

um eine gemischte Datenbank handelt, d. h. es gibt sowohl symbolische als auch numerische Attribute. Die *crx*-Datenbank stellt hohe Anforderungen an die Reproduktion, da es bei den symbolischen Attributen zwei Attribute mit 9 bzw. 14 verschiedenen Ausprägungen gibt, die z.T. nur mit sehr geringen Häufigkeiten auftreten. Gepaart mit einer relativ kleinen Anzahl Beispiele, ist es sehr schwierig, hier eine exakte Nachbildung zu erreichen. Daneben gibt es zwei reellwertige Attribute mit einer Spannweite von 2000 und 100000. Diese große Spannweite führt dazu, dass die multivariaten Maße verzerrt werden.

Die Häufigkeitsverteilungen der symbolischen Attribute konnten verhältnismäßig gut getroffen werden. Da die prozentuale Abweichung stark von der Anzahl der Beispiele abhängt, sind die Ergebnisse allerdings nicht so deutlich wie bei der *nursery*-Datenbank. Gerade bei Häufigkeiten, die nahe Null liegen sollen, hängt es vom Zufall ab, ob die Werte exakt getroffen werden. Auch wenn die prozentualen Abweichungen größer sind, kann man sagen, dass es gelungen ist, die symbolischen Attribute verhältnismäßig gut nachzubilden. Ein weiteres Indiz dafür sind die informationstheoretischen Maße. Die Abbildungen B.7 und B.8 zeigen die informationstheoretischen

```
*** Global Correlation-Matrix ***
Correlation Matrix
      [1]      [2]      [3]      [4]      [5]      [6]      [7]      [8]      [9]
[1]  1.00000
[2]  0.64399  1.00000
[3]  0.65365  0.90558  1.00000
[4]  0.48566  0.70457  0.68210  1.00000
[5]  0.52107  0.75072  0.71863  0.59874  1.00000
[6]  0.58845  0.68359  0.70352  0.66477  0.58207  1.00000
[7]  0.55763  0.75464  0.73489  0.66576  0.61522  0.67058  1.00000
[8]  0.53506  0.72183  0.71841  0.60249  0.62798  0.57123  0.66492  1.00000
[9]  0.34953  0.45803  0.43828  0.41703  0.47841  0.34230  0.34368  0.42772  1.00000
```

Abbildung 10.10: Globale Korrelationsmatrix der Original *breastLoss*-Datenbank.

```
*** Global Correlation-Matrix ***
Correlation Matrix
     [1]      [2]      [3]      [4]      [5]      [6]      [7]      [8]      [9]
[1] 1.00000
[2] 0.68453  1.00000
[3] 0.65954  0.88513  1.00000
[4] 0.62950  0.68878  0.63529  1.00000
[5] 0.55728  0.70047  0.65637  0.55846  1.00000
[6] 0.61640  0.68739  0.66801  0.64243  0.54757  1.00000
[7] 0.61095  0.73470  0.70214  0.66085  0.55183  0.64820  1.00000
[8] 0.54578  0.70560  0.67844  0.58534  0.58417  0.53478  0.64083  1.00000
[9] 0.34635  0.39345  0.34040  0.40781  0.49170  0.34622  0.30047  0.37084  1.00000
```

Abbildung 10.11: Globale Korrelationsmatrix der multinormalverteilt reproduzierten *breastLoss*-Datenbank.

Maße für die *crx*-Datenbank.

Die einzelnen numerischen Attribute konnten relativ gut nachgebildet werden, wobei auch hier die Wölbung die größten Abweichungen ausmacht. Die Ergebnisse der Diskriminanzanalyse sind ebenfalls relativ gut. Trotz der großen Varianz der Attribute A14 und A15 liegen die multivariaten Maße alle in der gleichen Größenordnung[11]. An dieser Stelle werden stellvertretend die jeweiligen Ergebnisse der Diskriminanzanalyse in Tabelle 10.4 angegeben.

Insgesamt kann man festhalten, dass es den Umständen entsprechend gut gelungen ist, auch diese Datenbank von Fallbeispielen zu reproduzieren. Die vorgestellten Ergebnisse ließen sich durch eine Verfeinerung der *Regeldatei* bzw. der Verteilungsstruktur noch weiter verbessern.

[11] Eine Ausnahme bildet die Bartlett-Statistik, die ein wenig aus dem Rahmen fällt. Grund hierfür ist vermutlich eine stärkere Anfälligkeit der Bartlett-Statistik gegenüber Ausreißern.

	Fract	Cancor	Wilks Lambda	Bartlett
crx_orig	1.0000	0.482638	0.767060	171.843054
crx_repro	1.0000	0.584607	0.658234	270.989907

Tabelle 10.4: Ergebnisse der Diskriminanzanalyse der *crx*-Datenbank

10.4 Zusammenfassung

In diesem Kapitel wurden verschiedene Möglichkeiten zur Generierung von Datenbanken von Fallbeispielen angesprochen. Zunächst wurde die Generierung symbolischer Attribute betrachtet. In Abschnitt 10.1.2 wurde auf die Generierung numerischer Attribute mit uni- und multivariaten Verteilungen sowie auf Verfahren zur Dichteschätzung eingegangen.

Im Anschluss wurde die Generierung mit Hilfe von Regeln beleuchtet. Dabei wurde speziell der Generator von Gabor Melli vorgestellt. Es wurden außerdem Überlegungen angestellt, auf Basis von Assoziationsregeln und der Verwendung von Diskretisierungsverfahren eine entsprechende Regelbasis aufzustellen. Bei diesem Ansatz erweist sich allerdings die hohe Anzahl berechneter Assoziationsregeln als kritisch. Darüber hinaus sind zwar effiziente Verfahren zur Diskretisierung der numerischen Attribute bekannt, schwierig wird es im Anschluss, aus den diskret generierten Werten wieder sinnvoll kontinuierliche Attribute zu erzeugen. Um den Informationsverlust im Rahmen zu halten, müsste man eine ganze Reihe von Informationen zu den einzelnen Attributen bereitstellen. Insgesamt folgt aus diesen Überlegungen, dass ein kombinierter Ansatz vielversprechender erscheint.

Ein kombinierter Ansatz, d.h. ein Ansatz der sowohl die Spezifikation von symbolischen als auch numerischen Attributen erlaubt, wurde in Abschnitt 10.1.3.3 als Ansatz des Generators vorgestellt und in Abschnitt 10.2 wurde die Umsetzung dieses Ansatzes beschrieben und ein Beispiel angegeben.

In 10.3 wurde untersucht, inwieweit der Generator seiner Aufgabe gerecht wird. Dazu wurde zunächst die Generierung synthetischer Datenbanken von Fallbeispielen untersucht. In diesem Zusammenhang wurde auf die durchgeführten Tests zur Überprüfung der gewünschten Verteilung eingegangen.

Im Anschluss wurde die Reproduktion von bestehenden Datenbanken betrachtet. Zunächst wurde die Verwendung der Ergebnisse des DCT zur Generierung von Datenbanken erläutert, bevor auf die Generierung von Struktureigenschaften eingegangen wurde. Im Rahmen der Betrachtung der numerischen Attribute wurde kurz auf gemischte Verteilungen eingegangen. In den Abschnitten 10.3.3.1 bis 10.3.3.3 wurde die Reproduktion von drei Datenbanken mit den gesammelten Ergebnissen dargestellt.

Aufgrund der erzielten Ergebnisse lässt sich festhalten, dass der Generator unter gewissen Randbedingungen, wie z.B. einer ausreichend großen Anzahl Beispiele, in der Lage ist, Datenbanken von Fallbeispielen zu reproduzieren. Diese Angaben sind

natürlich auf die berechneten Maße bezogen. Unter Umständen reicht es nicht aus, die Reproduktion lediglich bezogen auf die Klassifikation zu untersuchen. Wenn man eine Datenbank wirklich exakt reproduzieren möchte, muss man auch andere wichtige Abhängigkeiten zwischen den Attributen modellieren.

Eine detaillierte Beschreibung des Generators findet man in [Kleiner 1998].

Kapitel 11

Zusammenfassung, Grenzen und offene Punkte

Diese Dissertation hatte drei Ziele

1. Verbesserung des Produktbewährungsprozesses mittels KDD
2. Entwicklung eines Ansatzes zur Unterstützung der Methodenauswahl im KDD–Prozess, motiviert aus den Erfahrungen der praktischen Anwendungen
3. Umsetzung des Ansatzes in einem Prototyp

11.0.1 Produktbewährungsprozess

In Teil I dieser Arbeit wurde gezeigt, wie Aufgaben im Produktbewährungsprozess mit KDD besser gelöst werden können. Hierzu wurde die Analyse von Schadensschwerpunkten mittels KDD–Methoden verbessert und durch eine neue Trendprognose die Möglichkeit geschaffen, frühzeitiger neue Schadensschwerpunkte zu erkennen oder die Wirkung von Maßnahmen zu beurteilen. In beiden Themen wurden die Ansätze soweit entwickelt, dass sie mit Prototypen in die Linie übergeben werden konnten. Diese Ergebnisse wurden auch auf verschiedenen Konferenzen [Lindner und Klose 1997, Lindner und Studer 1999b] veröffentlicht. Aus der Praxis heraus wurde klar, dass der Prozess der Methodenauswahl im KDD–Prozess eine wichtige Rolle spielt.

11.0.2 Benutzerunterstützung

Basis für den hier entwickelten Ansatz zur Unterstützung der Methodenauswahl ist der in Kapitel 5.2 vorgestellte UGM–Ansatz. Während bei Robert Engels in seiner Dissertation [Engels 1999] der Schwerpunkt auf der *top down*–Unterstützung lag, d.h. Aufgabenanalyse und Aufgabenzerlegung, war der Fokus dieser Arbeit in Teil II und III die *bottum up*–Unterstützung gemäß dem UGM–Ansatz. Die *bottum*

up–Komponente unterstützt ausgehend von den gegebenen Daten die Auswahl geeigneter Methoden zu einer gegebenen Anwendung. Hierzu wurde eine Datencharakteristik DCT entwickelt, die eine Erweiterung der aus STATLOG [Michie u. a. 1994] definierten Maße ist. Diese Maße bildeten die Basis für die Unterstützung in der Methodenauswahl. Anders als in den bisherigen Ansätzen zur Methodenauswahl via Meta-Learning wurde ein Ansatz vorgestellt, die Methodenauswahl mittels CBR zu unterstützen. Dieser Ansatz hat gegenüber den Meta–Learning Ansätzen folgende Vorteile:

1. Erweiterbarkeit: Neue Algorithmen können leicht als neue Fälle der Fallbasis hinzugefügt werden.

2. Flexibilität: Auswertungen können nach unterschiedlichen Kriterien, wie Fehlerrate, Performance und nicht funktionalen Anforderungen durchgeführt werden.

3. Offenheit: Neue Fälle können jederzeit zur Fallbasis hinzugefügt werden.

4. Wiederverwendbarkeit: Die bisherigen Datencharakteristiken können auch bei einer Erweiterung der Fallbasis bis zu einem gewissen Grad weiterbenutzt werden. CBR Anätze können mit fehlenden Werten auch in der Fallbasis umgehen. Aber auch neue Maße können hinzugefügt werden.

5. Abbild der *Wirklichkeit*: Im Meta–Learning stellt sich immer die Frage, ob die Menge der genutzten Datensätze repräsentativ ist. *Reale Datensätze* werden nur selten der Allgemeinheit zur Verfügung gestellt. Ohne Wiederholung der Auswertung oder Neugenerierung des Meta–Modells können Datencharakteristiken von neuen Datensätzen hinzugefügt werden. Die Daten müssen nicht zur Verfügung gestellt werden. Weiterhin stellt sich die Frage, ob mit den bekannten Datensätzen überhaupt ein Meta–Learning zum Beispiel mit Regellernern sinnvoll ist. Kalousis und Metal standen 65 Datensätze zur Verfügung.

Die Güte der Empfehlungen war bei Datensätzen mit numerischen Attributen besser als bei reinen symbolischen Datensätzen.

11.0.3 Aussagekraft der Datencharakteristik

Mit DCT wurde eine gute Basis für die Auswahl von Methoden geschaffen, die auch im Vergleich mit anderen Charakteristiken, wie von Kalousis in seiner Dissertation durchgeführt, nicht statistisch signifikant schlechter ist als die beste Charakteristik in der Untersuchung [Kalousis 2002].

Die hier erzielten Ergebnisse zeigen, dass die verwendete Datencharakteristik eine Hilfe zur Auswahl der Methoden ist. Weiterhin zeigen die Ergebnisse auch ein Potential der Verbesserung in Bezug auf die symbolischen Attribute. Alexandros Kalousis untersuchte in seiner Dissertation *Algorithm Selection via Meta–Learning* einen eigenen Ansatz der Datencharakterisierung auch im Vergleich mit DCT [Kalousis 2002].

Ergebnis der Untersuchung war, dass keine statistisch signifikant besseren Ergebnisse erzielt werden konnten. Aber auch Kalousis erkennt, dass die Datencharakteristik der entscheidende Faktor bei der Unterstützung der Methodenauswahl ist.

Auch die Ergebnisse mit der Generierung von Datensätzen haben gezeigt, dass man noch nicht am Ziel angekommen ist. Mittels des Datengenerators können Datensätze nach einer vorgegebenen Datencharakteristik generiert werden (siehe auch Kapitel 10). Die Anwendung der Methoden auf diese Datensätze zeigte allerdings Varianzen in den Ergebnissen. Dieses lässt nur den Schluss zu, dass die Maße noch nicht ausreichen.

Die Anzahl der Beispiele ist im Meta–Learning allgemein ein Problem. Wie schon besprochen, stehen zu wenige Datensätze für solche Arbeiten zur Verfügung um ausreichend allgemeingültige Ergebnisse zu erzielen. Da auch die Varianz der Datencharakteristik nicht bekannt ist, kann man keine Referenzmenge von Datensätzen bestimmen. Basis für die meisten Untersuchungen waren bisher die UCI–Datensätze, wie auch in dieser Arbeit, für METAL und in der Dissertation von Alexandros Kalousis. Aus der praktischen Arbeit in Teil I dieser Arbeit konnten ein paar weitere Datensätze der Fallbasis hinzugefügt werden. Da dies nicht ausreicht und in Zukunft sich auch nicht ändern wird, stellt sich die Frage nach der Generierung von künstlichen Datensätzen.

Als Lösung auf diese Frage, wird in Kapitel 10 ein komplexer Datengenerator vorgestellt, mit dem Datensätze zu einer gegebenen Datencharakteristik berechnet oder Varianten von bekannten Datensätzen erzeugt werden können.

11.0.4 Zukünftige Arbeit

Die Ergebnisse dieser Arbeit und der vorgestellte Ansatz, sowie die vergleichbaren Ergebnisse und Ansätze aus anderen internationalen Forschungsprojekten zeigen, dass in der vorliegenden Arbeit der richtige Weg beschritten wurde. Aber natürlich konnten nicht alle Aspekte bis ins letzte Detail berücksichtigt werden und so wurden plausible und praxisbezogene Einschränkungen gemacht. Hier könnte man die vorgestellten Ideen auf weitere Methodengruppen evaluieren. Allerdings gibt es keinen Anhaltspunkt warum die Ergebnisse signifikant anders sein sollten.

Ähnlich wie der DM Advisor des Projektes METAL oder die Unterstützungskomponente des Projektes MiningMART ihre Komponenten im Internet anbieten, war es Ziel, AST im Internet zur Verfügung zu stellen. Im Jahre 2000 konnte, dank der Unterstützung einiger Sponsoren AST online zur Verfügung gestellt werden. Aufgrund der administrativen Aufwände wurde die Aktivität wieder eingestellt. Dieses könnte man vielleicht durch eine Kooperation mit den Betreibern des DM Advisor wieder aufleben lassen, zumal beide die gleiche Datenbasis DCT nutzen.

Ein weiterer Punktist, dass die Fallbasis von AST ausgebaut werden könnte. Die Methoden sind aktuell z.B. nicht mit ihren Parametereinstellungen hinterlegt. Der Ansatz erlaubt es, leicht die Fallbasis in diesem Punkt zu erweitern. Das Objekt muss nur um diese erweitert werden.

Der entscheidende Erfolgsfaktor für eine Benutzerunterstützung in der Methodenauswahl ist eine ausreichende Datencharakterisierung. Die Ergebnisse dieser Arbeit haben gezeigt, dass man einen Schritt weitergekommen ist. Aber für rein symbolische Datensätze sind die Ergebnisse etwas schlechter als für Datensätze, die auch numerische Attribute enthalten. Auch in Datensätzen mit symbolischen und numerischen Attributen sind die Ergebnisse besser. Das lässt den Schluss zu, dass hier Verbesserungen notwendig sind. In Zukunft sollte der Fokus auf eine Verfeinerung oder Verbesserung der Datencharakteristik liegen. Die Verfeinerung der Datencharakteristik ist sicherlich keine triviale Aufgabe, die einfach zu lösen ist, sie ist aber der Schlüssel zur Lösung des Problems. Hierzu muss man sich vor Augen halten, dass alle Informationen in Form der Datensätze vorliegen. Der Datensatz ist die ultimative, detaillierteste Beschreibung des Datensatzes. Was gesucht wird ist eine ausreichend kompakte Beschreibung dieser Daten.

Anhang A

Ergebnisse aus AST

A.1 Clusterstruktur der Trainings- und Klassifikationszeiten

Inducer	durchschnittliche Trainingszeit	maximale Trainingszeit	KM-Kmeans	KMD-Kmeans
C5.0	1.907196	4.983607	cluster-1	0.000074
IB	1.725848	5.342334	cluster-1	0.000052
ID3	2.347147	5.204007	cluster-1	0.000143
LAZYDT	0.089612	0.693147	cluster-1	0.000097
MC4	1.258461	4.094345	cluster-1	0.000035
NAIVE-BAYES	0.32405	2.890372	cluster-1	0.000079
PEBLS	0.0	0.0	cluster-1	0.000097
PERCEPTRON	1.462834	4.85203	cluster-1	0.000023
WINNOW	1.582409	4.912655	cluster-1	0.00003
MLPN	9.302413	11.462474	cluster-2	0.000013
RBFN	9.295859	12.686933	cluster-2	0.000013
T2	8.186264	11.31445	cluster-2	0.000179
IB1	4.579249	7.713785	cluster-3	0.000028
IB2	4.630293	7.757479	cluster-3	0.00002
ONER	5.370231	8.600983	cluster-3	0.0001
RIPPER	4.420178	7.957177	cluster-3	0.000053
CN2	3.876199	7.086738	cluster-4	0.000029
IB3	3.516799	7.095893	cluster-4	0.000028
IB4	3.679646	7.26473	cluster-4	0.000003
NBTREE	6.301759	9.506586	cluster-5	0.000011
OC1	6.431739	9.646335	cluster-5	0.00001

Tabelle A.1: KMeans-Cluster der Trainingszeiten

Anmerkung: Die Zeiten sind der natürliche Logarithmus der benötigten Zeiten (in Sekunden). Die Spalte "KM-Kmeans" enthält den dem jeweiligen Verfahren (Inducer) zugeordneten Cluster und "KMD-Kmeans" ein Abstandsmaß vom Zentrum des Clusters.

Inducer	durchschnittliche Testzeit	maximale Testzeit	KM-Kmeans	KMD-Kmeans
C5.0	0.087011	0.693147	cluster-1	0.0
CN2	0.0	0.0	cluster-1	0.0
ID3	0.0	0.0	cluster-1	0.0
MC4	0.0	0.0	cluster-1	0.0
NAIVE-BAYES	0.0	0.0	cluster-1	0.0
NBTREE	0.0	0.0	cluster-1	0.0
OC1	0.0	0.0	cluster-1	0.0
ONER	0.044452	1.098612	cluster-1	0.000002
PERCEPTRON	0.0	0.0	cluster-1	0.0
RIPPER	0.0	0.0	cluster-1	0.0
T2	0.0	0.0	cluster-1	0.0
WINNOW	0.0	0.0	cluster-1	0.0
IB	1.059997	3.433987	cluster-2	0.001423
IB1	0.769687	2.397895	cluster-2	0.000001
IB2	0.769687	2.484907	cluster-2	0.000001
LAZYDT	2.364279	4.997212	cluster-2	0.032362
RBFN	1.374866	2.197225	cluster-2	0.008892
IB3	0.358945	1.94591	cluster-3	0.000003
IB4	0.390198	1.94591	cluster-3	0.000003
MLPN	0.986495	1.609438	cluster-3	0.000005
PEBLS	0.422857	1.791759	cluster-3	0.000001

Tabelle A.2: KMeans-Cluster der Klassifikationszeiten

Anmerkung: Die Zeiten sind der natürliche Logarithmus der benötigten Zeiten (in Sekunden). Die Spalte "KM-Kmeans" enthält den dem jeweiligen Verfahren (Inducer) zugeordneten Cluster und "KMD-Kmeans" ein Abstandsmaß vom Zentrum des Clusters.

A.2 Datencharakteristiken und Fehlerraten

DataSet	Nr Attr	Sym	Nr Tupel	Nr Class	def Error	StdDev	Miss Val	Miss Line	Outl	Fract	Cancor	WLambda	Disc Fct	Span Sym	Avg Sym	Class Entr	Attr Entr	Mut Inf	Joint Entr	EqNr Attr	Noise Ratio
abalone	8	0.125	4177	28	83.505	0.050	0.000	0.000	0.857	0.711	0.794	0.207	0.857	0	3.000	3.602	1.582	0.234	4.950	15.394	5.759
adult	14	0.571	48842	2	23.928	0.261	0.009	0.074	0	1.000	0.474	0.776	0	39	12.375	0.794	1.771	0.081	2.484	9.801	20.869
anneal	32	0.813	898	5	23.831	0.283	0.566	1.000	0.500	0.026	0.152	0.481	0.500	5	2.423	1.190	0.427	0.946	0.671	1.257	-0.548
anneal-u	38	0.842	898	5	23.831	0.283	0.000	0.000	0.500	0.026	0.152	0.481	0.500	6	2.656	1.190	0.464	0.082	1.572	14.538	4.665
australian	14	0.571	690	2	44.493	0.055	0.000	0.000	0	1.000	0.480	0.769	0	12	4.500	0.991	1.310	0.098	2.203	10.143	12.403
auto	25	0.400	205	6	67.317	0.102	0.011	0.224	0.267	0.611	0.736	0.240	0.267	20	6.000	2.272	1.393	0.289	3.377	7.861	3.820
auto_clean	20	0.500	205	6	67.317	0.102	0.014	0.224	0.400	0.667	0.719	0.304	0.400	20	6.000	2.272	1.393	0.289	3.377	7.861	3.820
balance-scale	4	0	625	3	53.920	0.180	0.000	0.000	0.250	1.000	0.823	0.323	0.250	*	*	*	*	*	*	*	*
breast	9	0	699	2	34.478	0.155	0.002	0.023	0	1.000	0.916	0.161	0	*	*	*	*	*	*	*	*
breast-cancer	9	1	286	2	29.720	0.203	0.003	0.031	*	*	*	*	*	9	4.556	0.878	1.509	0.038	2.349	23.287	39.038
breast_clean	8	0	699	2	34.478	0.155	0.003	0.023	0	1.000	0.914	0.164	0	*	*	*	*	*	*	*	*
car	6	1	1728	4	29.977	0.271	0.000	0.000	*	*	*	*	*	1	3.500	1.206	1.792	0.114	2.884	10.538	14.666
cars	8	0.125	392	3	37.500	0.207	0.000	0.000	0.143	0.941	0.728	0.440	0.143	0	30.000	1.328	4.273	1.328	4.273	1.000	2.217
cars_clean	5	0.200	392	3	37.500	0.207	0.000	0.000	0.250	0.058	0.211	0.545	0.250	0	30.000	1.328	4.273	1.328	4.273	1.000	2.217
chess	36	1	3196	2	47.779	0.022	0.000	0.000	*	*	*	*	*	1	2.028	0.999	0.590	0.019	1.570	52.140	29.814
cleve	13	0.538	303	2	45.545	0.045	0.002	0.023	0	1.000	0.610	0.628	0	2	2.714	0.994	1.113	0.110	1.997	9.071	9.150
crx	15	0.600	690	2	44.493	0.055	0.006	0.054	0	1.000	0.481	0.768	0	12	4.444	0.991	1.257	0.097	2.152	10.239	11.987
dna	60	1	3186	3	48.085	0.131	0.000	0.000	*	*	*	*	*	0	4.000	1.480	1.975	0.056	3.399	26.515	34.383
ecoli	7	0	336	8	57.440	0.136	0.000	0.000	0.857	0.407	0.904	0.005	0.857	*	*	*	*	*	*	*	*
flag-language	28	0.643	194	10	76.289	0.078	0.000	0.000	0.800	0.414	0.602	0.304	0.800	6	3.611	2.875	1.264	0.262	3.877	10.959	3.816
flag-religion	28	0.643	194	8	69.072	0.097	0.000	0.000	0.600	0.531	0.570	0.452	0.600	8	3.722	2.546	1.282	0.246	3.582	10.365	4.218
german	20	0.650	1000	2	30.000	0.200	0.000	0.000	0	1.000	0.251	0.937	0	8	4.154	0.881	1.434	0.020	2.295	43.593	69.927
german-org	24	0.500	1000	2	30.000	0.200	0.000	0.000	0	1.000	0.468	0.781	0	0	2.000	0.881	0.596	0.004	1.473	245.012	164.610
glass	8	0	214	6	64.486	0.127	0.000	0.000	0.500	0.812	0.897	0.090	0.500	*	*	*	*	*	*	*	*
glass2	9	0	163	2	46.626	0.034	0.000	0.000	0.111	1.000	0.529	0.720	0.111	*	*	*	*	*	*	*	*

Tabelle A.2: Datencharakteristiken der verwendeten Datensätze

DataSet	Nr Attr	Sym	Nr Tupel	Nr Class	def Error	StdDev	Miss Val	Miss Line	Outl	Fract	Cancor	WLambda	Disc Fct	Span Sym	Avg Sym	Class Entr	Attr Entr	Mut Inf	Joint Entr	EqNr Attr	Noise Ratio
glass2_clean	7	0	163	2	46.626	0.034	0.000	0.000	0	1.000	0.498	0.752	0	*	*	*	*	*	*	*	*
heart	13	0	270	2	44.444	0.056	0.000	0.000	0	1.000	0.738	0.455	0	*	*	*	*	*	*	*	*
hepatitis	19	0.684	155	2	20.645	0.294	0.054	0.484	0	1.000	0.528	0.721	0	1	2.769	0.735	0.793	0.074	1.454	9.995	9.785
horse-colic	22	0.773	368	2	36.957	0.130	0.228	0.981	0	1.000	0.275	0.924	0	44	7.353	0.950	1.474	0.340	2.084	2.794	3.333
hypothyroid	25	0.720	3163	2	4.774	0.452	0.065	0.999	0	1.000	0.638	0.592	0	0	2.000	0.277	0.351	0.003	0.625	110.444	139.005
hypothyroid_clean	24	0.750	3163	2	4.774	0.452	0.064	0.999	0	1.000	0.612	0.625	0	0	2.000	0.277	0.351	0.003	0.625	110.444	139.005
ionosphere	34	0	351	2	35.897	0.141	0.000	0.000	0	*	*	*	*	*	*	*	*	*	*	*	*
iris	4	0	150	3	66.667	0.000	0.000	0.000	0.250	0.991	0.985	0.024	0.250	*	*	*	*	*	*	*	*
iris_clean	3	0	150	3	66.667	0.000	0.000	0.000	0.333	0.989	0.978	0.035	0.333	*	*	*	*	*	*	*	*
labor-neg	16	0.500	57	2	35.088	0.149	0.336	0.982	0	1.000	0.662	0.562	0	1	2.625	0.935	1.175	0.505	1.605	1.853	1.328
led24	24	1	3200	10	89.469	0.004	0.000	0.000	*	*	*	*	*	0	2.000	3.321	0.965	0.119	4.166	27.802	7.080
led7	7	1	3200	10	89.344	0.006	0.000	0.000	*	*	*	*	*	0	2.000	3.319	0.877	0.409	3.787	8.109	1.144
lenses	4	1	24	3	37.500	0.207	0.000	0.000	*	*	*	*	*	1	2.250	1.326	1.146	0.251	2.221	5.279	3.563
letter	16	0	20000	26	95.935	0.001	0.000	0.000	0.938	0.313	0.888	0.001	0.938	*	*	*	*	*	*	*	*
letter_clean	15	0	20000	26	95.935	0.001	0.000	0.000	0.933	0.317	0.888	0.001	0.933	*	*	*	*	*	*	*	*
liver	6	0	345	2	42.029	0.080	0.000	0.000	0	1.000	0.366	0.866	0	*	*	*	*	*	*	*	*
monk1	6	1	432	2	50.000	0.000	0.000	0.000	*	*	*	*	*	2	2.833	1.000	1.459	0.052	2.407	19.275	27.126
monk1-bin	10	1	556	2	50.000	0.000	0.000	0.000	*	*	*	*	*	0	2.000	1.000	0.951	0.017	1.935	60.047	56.117
monk1-corrupt	6	1	432	2	50.000	0.000	0.259	0.903	*	*	*	*	*	2	2.833	1.000	1.378	0.341	2.037	2.935	3.044
monk1-cross	5	1	432	2	50.000	0.000	0.000	0.000	*	*	*	*	*	7	4.000	1.000	1.751	0.154	2.597	6.490	10.364
monk1-full	6	1	432	2	50.000	0.000	0.000	0.000	*	*	*	*	*	2	2.833	1.000	1.459	0.052	2.407	19.275	27.126
monk1-local	17	1	556	2	50.000	0.000	0.000	0.000	*	*	*	*	*	0	2.000	1.000	0.912	0.023	1.889	43.289	38.488
monk1-org	6	1	432	2	50.000	0.000	0.000	0.000	*	*	*	*	*	2	2.833	1.000	1.459	0.052	2.407	19.275	27.126
monk2	6	1	432	2	32.870	0.171	0.000	0.000	*	*	*	*	*	2	2.833	0.914	1.459	0.003	2.369	279.905	446.050
monk2-bin	10	1	601	2	34.276	0.157	0.000	0.000	*	*	*	*	*	0	2.000	0.927	0.951	0.001	1.878	865.993	887.308

Tabelle A.2: Datencharakteristiken der verwendeten Datensätze

DataSet	Nr Attr	Sym	Nr Tupel	Nr Class	def Error	StdDev	Miss Val	Miss Line	Outl	Fract	Cancor	WLambda	Disc Fct	Span Sym	Avg Sym	Class Entr	Attr Entr	Mut Inf	Joint Entr	EqNr Attr	Noise Ratio
monk2-local	17	1	432	2	32.870	0.171	0.000	0.000	*	*	*	*	*	0	2.000	0.914	0.912	0.002	1.824	537.337	535.597
monk3	6	1	432	2	47.222	0.028	0.000	0.000	*	*	*	*	*	2	2.833	0.998	1.459	0.112	2.345	8.921	12.047
monk3-full	6	1	432	2	47.222	0.028	0.000	0.000	*	*	*	*	*	2	2.833	0.998	1.459	0.112	2.345	8.921	12.047
monk3-local	17	1	554	2	48.014	0.020	0.000	0.000	*	*	*	*	*	0	2.000	0.999	0.912	0.051	1.860	19.528	16.834
monk3-org	6	1	432	2	47.222	0.028	0.000	0.000	*	*	*	*	*	2	2.833	0.998	1.459	0.112	2.345	8.921	12.047
mushroom	22	1	8124	2	48.240	0.018	0.013	0.305	*	*	*	*	*	10	5.364	0.999	1.420	0.212	2.207	4.723	5.711
mux6	6	1	128	2	50.000	0.000	0.000	0.000	*	*	*	*	*	0	2.000	1.000	1.000	0.030	1.970	32.919	31.919
nursery	8	1	12960	5	66.667	0.153	0.000	0.000	*	*	*	*	*	3	3.375	1.716	1.708	0.161	3.263	10.630	9.576
page-blocks	10	0	5473	5	10.232	0.349	0.000	0.000	0.300	0.490	0.721	0.190	0.300	*	*	*	*	*	*	*	*
page-blocks_clean	8	0	5473	5	10.232	0.349	0.000	0.000	0.375	0.507	0.712	0.215	0.375	*	*	*	*	*	*	*	*
parity5+5	10	1	1024	2	50.000	0.000	0.000	0.000	*	*	*	*	*	0	2.000	1.000	1.000	0.000	2.000	Inf	Inf
pima	8	0	768	2	34.896	0.151	0.000	0.000	0	1.000	0.551	0.697	0	*	*	*	*	*	*	*	*
quisclas	18	0	5891	3	57.376	0.066	0.000	0.000	0.167	0.775	0.380	0.816	0.167	*	*	*	*	*	*	*	*
satimage	36	0	8870	6	75.829	0.065	0.000	0.000	0.111	0.442	0.934	0.006	0.111	*	*	*	*	*	*	*	*
satimage_clean	19	0	8870	6	75.829	0.065	0.000	0.000	0.211	0.463	0.933	0.007	0.211	*	*	*	*	*	*	*	*
segment	19	0	2310	7	85.714	0.000	0.000	0.000	0.211	*	*	*	*	*	*	*	*	*	*	*	*
solar	12	0.750	323	6	72.755	0.090	0.000	0.000	0.667	0.040	0.068	0.893	0.667	4	2.778	2.322	0.958	0.273	3.007	8.511	2.513
sonar	60	0	208	2	46.635	0.034	0.000	0.000	0	*	*	*	*	*	*	*	*	*	*	*	*
soybean-large	35	1	683	19	86.530	0.043	0.095	0.177	*	*	*	*	*	5	2.829	3.836	1.032	0.976	3.892	3.932	0.058
soybean-small	35	0	47	4	63.830	0.064	0.000	0.000	0	*	*	*	*	*	*	*	*	*	*	*	*
threeOf9	9	1	512	2	46.484	0.035	0.000	0.000	*	*	*	*	*	0	2.000	0.996	1.000	0.037	1.959	26.613	25.708
tic-tac-toe	9	1	958	2	34.655	0.153	0.000	0.000	*	*	*	*	*	0	3.000	0.931	1.538	0.019	2.450	49.450	80.698
titanic	3	1	2201	2	32.303	0.177	0.000	0.000	*	*	*	*	*	2	2.667	0.908	0.959	0.069	1.797	13.085	12.824
vehicle	18	0	846	4	74.232	0.009	0.000	0.000	0.111	0.527	0.842	0.083	0.111	*	*	*	*	*	*	*	*
vehicle_clean	10	0	846	4	74.232	0.009	0.000	0.000	0.200	0.667	0.722	0.304	0.200	*	*	*	*	*	*	*	*

Tabelle A.2: Datencharakteristiken der verwendeten Datensätze

DataSet	Nr Attr	Sym	Nr Tupel	Nr Class	def Error	StdDev	Miss Val	Miss Line	Outl	Fract	Cancor	WLambda	Disc Fct	Span Sym	Avg Sym	Class Entr	Attr Entr	Mut Inf	Joint Entr	EqNr Attr	Noise Ratio
vote	16	1	435	2	38.621	0.114	0.000	0.000	*	*	*	*	*	0	3.000	0.962	1.200	0.256	1.907	3.765	3.696
waveform-21	21	0	4998	3	66.126	0.004	0.000	0.000	0.048	0.548	0.695	0.292	0.048	*	*	*	*	*	*	*	*
waveform-40	40	0	4999	3	66.153	0.004	0.000	0.000	0.025	0.529	0.694	0.284	0.025	*	*	*	*	*	*	*	*
wdbc	30	0	569	2	37.258	0.127	0.000	0.000	0.133	1.000	0.880	0.226	0.133	*	*	*	*	*	*	*	*
wine	13	0	178	3	60.112	0.053	0.000	0.000	0.077	0.687	0.949	0.019	0.077	*	*	*	*	*	*	*	*
zoo	16	1	101	7	59.406	0.118	0.000	0.000	*	*	*	*	*	4	2.250	2.391	0.888	0.578	2.701	4.138	0.537

Tabelle A.2: Datencharakteristiken der verwendeten Datensätze

DataSet	C5.0	CN2	IB	IB1	IB2	IB3	IB4	ID3	MC4	NAIVE-B	NBTREE	OC1	ONER	PEBLS	PERCEPT	RIPPER	T2	WINNOW	MLPN	RBFN	LAZYDT
abalone	77.87	76.30	79.74	79.70	79.70	78.10	77.60	*	77.66	77.16	75.43	74.21	76.44	79.70	*	78.95	*	*	71.05	74.93	*
adult	13.82	*	20.82	*	*	*	*	*	*	16.87	13.97	15.63	*	*	18.62	15.63	16.26	27.02	15.93	18.61	*
anneal	8.67	7.30	1.00	5.70	5.70	9.00	16.00	0.00	1.33	9.00	2.33	7.00	18.00	*	*	5.00	6.33	*	0.67	7.33	4.00
anneal-u	1.33	0.70	1.00	1.00	1.00	3.30	3.70	0.00	1.33	7.67	1.00	1.33	18.00	1.30	*	1.33	6.33	*	0.67	7.33	1.33
australian	13.04	17.40	18.26	18.30	18.30	13.90	12.20	18.70	13.04	22.17	14.35	*	13.48	19.60	17.39	13.91	19.13	18.26	20.00	15.65	14.78
auto	39.13	33.30	26.09	29.00	29.00	40.60	53.60	26.09	34.78	44.93	33.33	*	60.87	*	*	26.09	20.29	*	30.77	46.00	33.33
auto_clean	44.93	26.10	26.09	27.50	27.50	40.60	52.20	28.99	40.58	43.48	33.33	*	46.38	*	*	42.03	20.29	*	38.46	46.00	31.88
balance-scale	22.49	19.10	15.79	15.80	16.70	13.90	16.30	21.53	22.97	10.53	26.79	11.96	43.06	15.30	*	21.53	34.45	*	5.74	10.53	27.27
breast	4.29	4.70	3.86	3.90	3.90	3.40	2.60	7.30	4.29	3.43	3.43	6.01	7.30	*	5.15	5.58	5.58	4.29	4.42	2.65	6.44
breast-cancer	25.26	31.60	28.42	34.70	28.40	29.50	38.90	28.42	25.26	25.26	27.37	29.47	29.47	*	71.58	22.11	29.47	35.79	36.84	28.42	27.37
breast_clean	5.58	5.20	3.43	3.40	3.40	3.00	2.60	5.15	4.29	3.43	3.86	3.43	8.15	*	4.72	5.15	6.01	4.72	3.54	3.10	5.58
car	11.11	6.60	9.03	6.80	7.60	13.40	44.80	8.85	11.11	14.93	8.16	3.12	29.69	4.70	*	16.49	23.26	*	0.00	18.23	5.21
cars	*	5.30	1.53	1.50	1.50	1.50	1.50	2.29	2.29	1.53	1.53	14.50	1.53	*	*	9.92	*	*	1.53	2.29	10.69
cars_clean	*	4.60	1.53	1.50	1.50	3.10	3.10	3.82	1.53	1.53	1.53	14.50	1.53	*	*	6.87	*	*	1.53	4.58	10.69
chess	0.47	*	4.97	4.00	4.60	11.90	6.80	1.31	0.47	12.85	0.66	3.94	32.18	4.60	12.76	1.88	13.41	4.97	1.22	18.29	2.91
cleve	22.77	29.70	22.77	22.80	22.80	20.80	17.80	35.64	23.76	17.82	18.81	*	24.75	*	21.78	30.69	29.70	22.77	19.19	18.18	26.73
crx	17.00	21.00	22.50	22.50	22.50	20.00	18.00	27.50	17.00	23.50	17.00	*	17.00	*	56.50	19.00	24.50	55.00	21.65	19.07	20.50
dna	6.59	8.40	27.50	24.30	24.90	25.20	13.20	9.79	6.59	4.90	4.90	9.23	35.78	6.60	*	9.51	22.50	*	6.78	5.74	*
ecoli	21.43	21.40	19.64	19.60	19.60	13.40	14.30	20.54	21.43	17.86	16.96	18.75	34.82	24.10	*	17.86	29.46	*	22.32	13.39	23.21
flag-language	47.69	58.50	55.38	58.50	58.50	53.80	44.60	56.92	47.69	61.54	49.23	46.15	50.77	55.40	*	55.38	55.38	*	63.08	43.08	58.46
flag-religion	40.00	35.40	41.54	41.50	41.50	50.80	46.20	40.00	40.00	49.23	44.62	44.62	46.15	43.10	*	50.77	49.23	*	41.54	40.00	63.08
german	28.74	27.80	30.24	30.20	30.20	32.60	23.70	33.23	26.95	22.45	24.85	28.14	35.93	33.50	24.25	25.45	27.84	23.95	27.54	28.44	32.04
german-org	25.15	24.90	29.64	29.60	29.60	31.10	29.00	*	25.45	25.15	25.15	25.15	32.34	28.70	23.95	26.05	26.95	29.04	32.63	28.44	26.65
glass	37.50	23.60	29.17	26.40	26.40	33.30	33.30	37.50	37.50	50.00	36.11	37.50	51.39	*	*	33.33	34.72	*	26.39	34.72	30.56
glass2	32.73	18.20	25.45	25.50	25.50	23.60	32.70	30.91	30.91	34.55	7.27	16.36	25.45	29.10	49.09	16.36	23.64	34.55	23.64	27.27	25.45

Tabelle A.3: Fehlerraten der Verfahren

DataSet	C5.0	CN2	IB	IB1	IB2	IB3	IB4	ID3	MC4	NAIVE-B	NBTREE	OC1	ONER	PEBLS	PERCEPT	RIPPER	T2	WINNOW	MLPN	RBFN	LAZYDT
glass2_clean	23.64	20.00	20.00	20.00	20.00	25.50	21.80	30.91	21.82	36.36	23.64	20.00	52.73	23.60	54.55	25.45	20.00	41.82	20.00	27.27	21.82
heart	16.67	22.20	20.00	20.00	20.00	17.80	13.30	23.33	16.67	14.44	18.89	18.89	26.67	21.10	12.22	20.00	26.67	18.89	20.00	14.44	16.67
hepatitis	19.23	19.20	21.15	23.10	23.10	25.00	23.10	21.15	28.85	23.08	17.31	*	17.31	*	44.23	23.08	32.69	86.54	6.02	8.54	25.00
horse-colic	14.71	19.10	33.82	22.10	22.10	20.60	20.60	26.47	17.65	26.47	23.53	*	17.65	*	44.12	13.24	16.18	39.71	29.55	13.64	25.00
hypothyroid	0.76	1.40	3.70	3.90	3.90	7.90	6.80	0.95	0.76	2.27	1.71	*	2.75	*	5.31	0.76	0.85	94.79			1.23
hypothyroid_clean	0.76	1.30	4.17	4.40	4.40	8.00	5.30	0.95	0.76	2.46	1.14	*	2.75	*	5.31	0.76	0.85	94.79			1.42
ionosphere	10.26	9.40	11.97	12.00	12.00	13.70	12.00	8.55	11.97	15.38	5.98	8.55	22.22	6.80	17.95	7.69	11.11	39.32	5.13	8.55	11.97
iris	8.00	6.00	4.00	4.00	4.00	4.00	2.00	6.00	8.00	6.00	4.00	6.00	6.00	4.00	*	8.00	6.00	*	6.00	2.00	4.00
iris_clean	8.00	8.00	6.00	6.00	6.00	6.00	4.00	6.00	8.00	6.00	6.00	6.00	6.00	6.00	*	6.00	6.00	*	6.00	4.00	6.00
labor-neg	17.65	29.40	0.00	6.20	6.20	18.80	37.50	5.88	23.53	11.77	5.88	*	23.53	*	47.06	29.41	5.88	35.29			29.41
led24	34.33	42.30	60.57	61.90	61.10	62.20	37.80	44.67	34.43	35.90	35.87	34.40	82.00	42.90	*	41.87	66.47	*	52.20	47.17	*
led7	32.47	31.30	31.40	30.50	30.80	27.80	29.10	33.47	32.60	31.07	30.73	32.87	81.67	42.50	*	33.50	67.20	*	33.53	29.27	36.77
lenses	37.50	37.50	37.50	37.50	37.50	62.50	62.50	37.50	37.50	62.50	62.50	62.50	62.50	37.50	*	37.50	*	*	37.50	37.50	37.50
letter	13.02	29.10	4.48	*	*	*	*	13.02	*	36.04	11.08	17.30	83.52	7.10	*	15.36	*	*			*
letter_clean	12.88	29.70	4.26	*	*	*	*	12.96	*	35.10	12.64	17.22	83.52	7.20	*	14.52	*	*			*
liver	36.52	33.90	40.87	40.00	40.00	40.90	35.70	38.26	33.04	43.48	31.30	31.30	39.13	44.30	38.26	33.04	37.39	49.56	33.04	38.26	42.61
monk1	25.69	1.40	16.20	14.60	17.10	35.00	22.20	18.98	24.31	28.70	3.01	5.79	25.00	13.00	37.96	27.78	16.67	36.34	0.00	21.06	8.10
monk1-bin	0.00	0.00	15.97	15.70	15.50	30.60	18.10	12.27	8.33	34.03	15.51	10.65	33.33	14.80	40.97	25.00	30.56	37.73	0.00	24.31	12.50
monk1-corrupt	34.72	25.00	35.42	33.30	30.60	36.10	27.80	33.33	29.17	36.11	27.78	37.50	28.47	*	37.50	24.31	26.39	36.81	30.56	32.64	34.03
monk1-cross	5.56	0.00	14.12	14.10	14.60	28.00	29.20	4.17	5.56	0.69	0.69	0.00	16.67	0.00	0.46	25.00	0.00	0.00	0.00	15.05	10.42
monk1-full	25.69	1.40	16.20	14.60	17.10	35.00	22.20	18.98	24.31	28.70	3.01	5.79	25.00	13.00	37.96	27.78	16.67	36.34	0.00	21.06	8.10
monk1-local	0.00	0.00	16.20	14.60	17.10	35.00	18.50	7.41	5.56	29.63	5.56	0.00	25.00	18.50	31.02	0.00	27.78	32.64	0.00	30.79	2.78
monk1-org	25.69	1.40	16.20	14.60	17.10	35.00	22.20	18.98	24.31	28.70	3.01	5.79	25.00	13.00	37.96	27.78	16.67	36.34	0.00	21.06	8.10
monk2	34.95	24.30	27.78	29.20	29.20	41.90	43.50	30.09	35.42	38.43	31.71	0.69	32.87	25.50	49.77	37.50	39.35	53.01	0.00	37.50	17.82
monk2-bin	34.49	27.10	37.50	38.90	39.40	43.10	43.10	28.70	34.72	34.95	34.26	6.02	32.87	35.40	51.39	37.50	37.50	32.87	2.08	40.51	31.94

Tabelle A.3: Fehlerraten der Verfahren

DataSet	C5.0	CN2	IB	IB1	IB2	IB3	IB4	ID3	MC4	NAIVE-B	NBTREE	OC1	ONER	PEBLS	PERCEPT	RIPPER	T2	WINNOW	MLPN	RBFN	LAZYDT
monk2-local	27.78	25.50	27.78	29.20	29.20	41.90	39.60	13.43	30.56	39.35	29.17	4.40	32.87	25.00	36.11	34.95	39.35	38.66	0.00	34.95	26.39
monk3	2.78	9.30	12.50	13.40	13.90	22.20	39.80	8.33	2.78	2.78	2.78	5.09	19.44	6.20	5.56	8.33	2.78	15.28	6.94	3.94	3.47
monk3-full	2.78	9.30	12.50	13.40	13.90	22.20	39.80	8.33	2.78	2.78	2.78	5.09	19.44	6.20	5.56	8.33	2.78	15.28	6.71	3.94	3.47
monk3-local	0.00	7.40	12.50	13.40	13.90	22.20	2.80	10.19	0.00	2.78	3.01	2.08	19.44	5.60	4.63	0.00	2.78	11.11	7.87	5.56	3.94
monk3-org	2.78	9.30	12.50	13.40	13.90	22.20	39.80	8.33	2.78	2.78	2.78	5.09	19.44	6.20	5.56	8.33	2.78	15.28	6.02	3.94	3.47
mushroom	0.04	0.00	0.00	*	*	*	*	0.04	0.04	0.33	0.04	0.41	1.85	*	0.07	0.15	0.70	0.07	0.04	2.99	0.04
mux6	0.00	0.00	0.00	0.00	0.00	25.00	37.50	0.00	0.00	31.25	25.00	6.25	37.50	37.50	31.25	12.50	25.00	37.50	0.00	10.94	0.00
nursery	4.03	3.00	2.66	*	*	*	*	2.78	3.84	10.39	3.59	1.02	28.73	1.00	*	3.94	17.78	*	4.44	22.87	*
page-blocks	3.34	4.00	4.66	4.50	4.50	7.90	7.80	3.29	3.34	7.73	3.23	3.84	5.97	8.30	*	3.45	4.16	*	3.40	7.29	*
page-blocks_clean	3.07	4.10	4.66	4.50	4.50	7.90	7.90	3.40	2.96	5.81	2.85	3.29	5.97	8.50	*	2.79	4.49	*	3.73	7.34	*
parity5+5	48.44	47.00	47.36	47.30	47.10	51.00	50.00	49.22	50.00	50.00	51.27	41.41	50.00	50.20	50.00	50.00	50.00	50.10	47.46	51.17	48.93
pima	23.83	25.40	31.25	30.90	30.90	27.70	27.00	29.30	24.61	21.88	21.09	22.66	26.17	28.50	23.44	21.88	23.44	28.12	27.34	23.05	20.70
quisclas	38.34	42.80	42.82	42.80	42.80	40.30	40.20	*	*	63.80	43.63	34.32	53.77	*	*	37.02	41.50	*	32.28	43.08	*
satimage	2.30	25.30	0.00	*	*	*	*	0.00	*	20.23	16.80	8.86	38.76	0.00	*	8.55		*	15.40	13.60	*
satimage_clean	3.61	25.70	0.00	*	*	*	*	0.00	*	20.81	8.07	9.31	41.28	0.00	*	8.03		*	26.56	14.32	*
segment	5.58	17.70	3.90	3.90	3.90	6.50	6.50	3.77	5.97	20.65	7.79	5.71	42.21	6.10	*	7.79	*	*	4.16	17.66	*
solar	29.63	28.70	26.85	28.70	29.60	31.50	37.00	27.78	30.56	36.11	32.41	35.19	50.00	31.50	*	29.63	34.26	*	23.15	29.63	30.56
sonar	25.71	25.70	14.29	14.30	14.30	25.70	18.60	21.43	25.71	30.00	22.86	42.86	34.29	24.30	20.00	38.57	27.14	54.29	15.71	22.86	31.43
soybean-large	10.53	9.20	7.89	10.50	9.60	12.30	7.50	6.14	6.58	9.65	5.70	8.33	59.65	*	*	7.02	33.77	*	8.33	18.42	*
soybean-small	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	50.00	25.00	0.00	*	0.00	0.00	*	0.00	0.00	0.00
threeof9	2.92	0.00	5.26	3.50	3.50	19.30	23.40	2.34	2.92	25.15	11.70	25.73	38.01	17.50	25.73	0.00	32.16	25.15	0.58	19.88	4.68
tic-tac-toe	18.44	1.60	1.88	1.60	0.60	8.10	25.00	19.06	18.12	28.75	19.06	14.69	30.00	11.60	2.19	1.88	32.50	17.50	1.56	25.31	5.63
titanic	21.80	21.50	21.25	21.30	21.30	24.80	24.80	21.25	21.80	21.53	21.25	21.25	22.62	29.60	22.07	21.80	21.80	25.61	21.25	20.71	21.25
vehicle	32.27	42.60	36.17	36.20	36.20	37.20	35.50	30.14	31.91	60.64	29.43	32.98	51.77	33.30	*	34.04	42.91	*	23.40	41.49	*
vehicle_clean	34.75	42.90	41.13	41.10	41.10	39.70	39.70	37.59	36.52	53.90	37.23	33.33	54.97	36.50	*	39.01	46.81	*	30.85	48.58	41.49

Tabelle A.3: Fehlerraten der Verfahren

DataSet	C5.0	CN2	IB	IB1	IB2	IB3	IB4	ID3	MC4	NAIVE-B	NBTREE	OC1	ONER	PEBLS	PERCEPT	RIPPER	T2	WINNOW	MLPN	RBFN	LAZYDT
vote	2.96	4.40	5.18	4.40	4.40	5.90	4.40	5.93	2.96	8.15	4.44	1.48	2.96	3.00	1.48	2.96	3.70	2.22	2.96	3.70	5.93
waveform-21	29.75	31.20	24.30	24.60	24.60	24.70	23.30	29.77	27.92	19.34	22.00	22.30	48.09	25.50	*	26.22	38.37	*	45.32	15.30	*
waveform-40	29.81	33.40	30.49	31.00	31.00	26.50	27.30	31.21	29.51	19.47	28.21	25.55	47.06	25.10	*	26.79	33.77	*	20.89	17.81	*
wdbc	6.32	4.70	4.21	4.20	4.20	2.60	3.70	7.89	6.32	7.89	6.84	5.26	8.95	3.70	7.37	4.74	10.00	57.37	5.26	4.74	8.95
wine	15.00	10.00	6.67	6.70	6.70	5.00	5.00	15.00	15.00	5.00	13.33	11.67	26.67	1.70	*	11.67	16.67	*	0.00	3.33	28.33
zoo	11.77	17.60	11.77	11.80	11.80	20.60	20.60	2.94	20.59	35.29	11.77	26.47	32.35	11.80	*	23.53	29.41	*	14.71	14.71	14.71

Tabelle A.3: Fehlerraten der Verfahren

A.3 Ähnlichste Datensätze

DataSet	SimDS1	sim1	SimDS2	sim2	SimDS3	sim3
abalone	ecoli	0.872	letter_clean	0.865	letter	0.861
adult	australian	0.932	crx	0.926	cleve	0.857
anneal	anneal-u	0.835	solar	0.727	monk1-corrupt	0.703
anneal-u	anneal	0.835	titanic	0.801	solar	0.791
australian	crx	0.991	adult	0.932	cleve	0.923
auto	auto_clean	0.953	flag-religion	0.846	vehicle_clean	0.83
auto_clean	auto	0.953	flag-religion	0.867	flag-language	0.815
balance-scale	iris	0.928	cars	0.911	iris_clean	0.91
breast	breast_clean	0.996	heart	0.925	pima	0.905
breast-cancer	monk1	0.89	monk1-org	0.89	monk1-full	0.89
breast_clean	breast	0.996	heart	0.923	pima	0.908
car	nursery	0.92	monk3-org	0.915	monk3	0.915
cars	balance-scale	0.911	heart	0.906	glass2	0.888
cars_clean	cars	0.872	balance-scale	0.778	quisclas	0.772
chess	monk1-local	0.887	mux6	0.88	threeof9	0.879
cleve	australian	0.923	crx	0.92	hepatitis	0.907
crx	australian	0.991	adult	0.926	cleve	0.92
dna	tic-tac-toe	0.823	threeof9	0.823	monk1	0.819
ecoli	letter_clean	0.896	letter	0.892	abalone	0.872
flag-language	flag-religion	0.922	auto_clean	0.815	led24	0.798
flag-religion	flag-language	0.922	auto_clean	0.867	auto	0.846
german	crx	0.875	australian	0.873	german-org	0.872
german-org	hypothyroid_clean	0.885	hypothyroid	0.883	monk2-local	0.882
glass	iris_clean	0.884	iris	0.868	ecoli	0.853
glass2	glass2_clean	0.966	pima	0.95	liver	0.93
glass2_clean	pima	0.966	glass2	0.966	liver	0.964
heart	pima	0.928	breast	0.925	glass2_clean	0.925
hepatitis	cleve	0.907	crx	0.878	labor-neg	0.877
horse-colic	mushroom	0.874	crx	0.867	australian	0.861
hypothyroid	hypothyroid_clean	0.991	german-org	0.883	hepatitis	0.839
hypothyroid_clean	hypothyroid	0.991	german-org	0.885	hepatitis	0.841
ionosphere	vote	0.53	monk2-bin	0.53	monk2	0.529
iris	iris_clean	0.978	balance-scale	0.928	wine	0.899
iris_clean	iris	0.978	balance-scale	0.91	glass	0.884
labor-neg	cleve	0.897	hepatitis	0.877	monk1-corrupt	0.83
led24	led7	0.884	flag-language	0.798	monk3-local	0.794
led7	led24	0.884	zoo	0.841	solar	0.823
lenses	vote	0.914	monk3	0.908	monk3-org	0.908
letter	letter_clean	0.996	ecoli	0.892	abalone	0.861
letter_clean	letter	0.996	ecoli	0.896	abalone	0.865

Tabelle A.4: Ähnlichste Datensätze

DataSet	SimDS1	sim1	SimDS2	sim2	SimDS3	sim3
liver	glass2_clean	0.964	pima	0.947	glass2	0.93
monk1	monk1-org	1	monk1-full	1	monk3	0.961
monk1-bin	monk1-local	0.968	mux6	0.965	threeof9	0.963
monk1-corrupt	monk3	0.913	monk3-org	0.913	monk3-full	0.913
monk1-cross	monk3	0.934	monk3-org	0.934	monk3-full	0.934
monk1-full	monk1-org	1	monk1	1	monk3	0.961
monk1-local	monk1-bin	0.968	monk3-local	0.96	mux6	0.956
monk1-org	monk1	1	monk1-full	1	monk3	0.961
monk2	tic-tac-toe	0.956	monk2-bin	0.946	monk2-local	0.929
monk2-bin	monk2-local	0.976	monk1-bin	0.951	tic-tac-toe	0.949
monk2-local	monk2-bin	0.976	monk1-local	0.937	monk2	0.929
monk3	monk3-org	1	monk3-full	1	monk1	0.961
monk3-full	monk3	1	monk3-org	1	monk1	0.961
monk3-local	monk1-local	0.96	threeof9	0.951	monk1-bin	0.928
monk3-org	monk3	1	monk3-full	1	monk1	0.961
mushroom	monk1-cross	0.875	horse-colic	0.874	crx	0.861
mux6	threeof9	0.97	monk1-bin	0.965	monk1-local	0.956
nursery	cars	0.92	monk3	0.89	monk3-org	0.89
page-blocks	page-blocks_clean	0.974	vehicle_clean	0.883	satimage_clean	0.858
page-blocks_clean	page-blocks	0.974	vehicle_clean	0.866	glass	0.853
parity5+5	monk1-bin	0.869	threeof9	0.862	mux6	0.853
pima	glass2_clean	0.966	glass2	0.95	liver	0.947
quisclas	glass2	0.887	liver	0.878	glass2_clean	0.866
satimage	satimage_clean	0.9	vehicle	0.865	wine	0.839
satimage_clean	vehicle	0.922	satimage	0.9	wine	0.883
segment	zoo	0.611	led24	0.608	satimage_clean	0.577
solar	zoo	0.89	nursery	0.859	led7	0.823
sonar	dna	0.636	waveform-40	0.565	monk3	0.531
soybean-large	led7	0.806	zoo	0.785	flag-language	0.75
soybean-small	soybean-large	0.525	wine	0.498	glass2_clean	0.491
threeof9	mux6	0.97	monk1-bin	0.963	monk1-local	0.954
tic-tac-toe	monk2	0.956	monk2-bin	0.949	monk1-bin	0.929
titanic	monk3	0.931	monk3-org	0.931	monk3-full	0.931
vehicle	wine	0.922	satimage_clean	0.922	waveform-21	0.922
vehicle_clean	vehicle	0.912	waveform-21	0.905	balance-scale	0.897
vote	monk3-local	0.915	lenses	0.914	monk3	0.91
waveform-21	vehicle	0.922	waveform-40	0.915	vehicle_clean	0.905
waveform-40	waveform-21	0.915	vehicle	0.851	vehicle_clean	0.85
wdbc	breast	0.902	breast_clean	0.9	heart	0.879
wine	vehicle	0.922	iris	0.899	breast	0.892
zoo	solar	0.89	vote	0.848	led7	0.841

Tabelle A.4: Ähnlichste Datensätze

DataSet	Nr Attr	Sym	Nr Tupel	Nr Class	def Error	StdDev	Miss Val	Miss Line	Outl	Fract	Cancor	WLambda	Disc Fct	Span Sym	Avg Sym	Class Entr	Attr Entr	Mut Inf	Joint Entr	EqNr Attr	Noise Ratio
breast-cancer	9	1	286	2	29.720	0.203	0.003	0.031	*	*	*	*	*	9	4.556	0.878	1.509	0.038	2.349	23.287	39.038
breast-cancer_gen	9	1	300	2	29.667	0.203	0.003	0.033	*	*	*	*	*	9	4.556	0.877	1.521	0.042	2.357	21.082	35.555
breast	9	0	699	2	34.478	0.155	0.002	0.023	0	1.000	0.916	0.161	0	*	*	*	*	*	*	*	*
breast_gen	9	0	700	2	34.429	0.156	0.003	0.026	0	1.000	0.954	0.090	0	*	*	*	*	*	*	*	*
crx	15	0.600	690	2	44.493	0.055	0.006	0.054	0	1.000	0.481	0.768	0	12	4.444	0.991	1.257	0.097	2.152	10.239	11.987
crx_f	15	0.600	653	2	45.329	0.047	0.000	0.000	0	1.000	0.483	0.767	0	12	4.444	0.994	1.250	0.091	2.152	10.884	12.691
crx_gen	15	0.600	700	2	45.286	0.047	0.000	0.000	0	1.000	0.551	0.697	0	12	4.333	0.994	1.267	0.088	2.173	11.341	13.458

DataSet	C5.0	CN2	IB	IB1	IB2	IB3	IB4	ID3	MC4	NAIVE-B	NBTREE	OC1	ONER	PEBLS	PERCEPT	RIPPER	T2	WINNOW	MLPN	RBFN	LAZYDT
breast-cancer	25.26	31.60	28.42	34.70	28.40	29.50	38.90	28.42	25.26	25.26	27.37	29.47	29.47	*	71.58	22.11	29.47	35.79	36.84	28.42	27.37
breast-cancer_gen	40.00	36.00	40.00	39.00	39.00	33.00	38.00	46.00	39.00	30.00	30.00	43.00	40.00	*	33.00	41.00	39.00	37.00	34.00	38.00	40.00
breast	4.29	4.70	3.86	3.90	3.90	3.40	2.60	7.29	4.29	3.43	3.43	6.01	7.29	*	5.15	5.58	5.58	4.29	4.42	2.65	6.44
breast_gen	0.86	3.00	0.86	0.40	0.40	0.90	0.90	1.72	1.29	4.29	2.15	0.43	7.29	*	0.43	1.72	3.43	48.93	0.00	0.43	2.15
crx	17.00	21.00	22.50	22.50	22.50	20.00	18.00	27.50	17.00	23.50	17.00	*	17.00	*	56.50	19.00	24.50	55.00	21.65	19.07	20.50
crx_f	16.49	17.00	21.81	21.80	21.80	21.30	17.00	25.53	15.96	22.34	19.15	19.15	15.96	21.80	44.15	15.96	15.96	43.08	17.11	20.32	18.62
crx_gen	13.30	12.00	14.59	14.60	14.60	15.50	15.90	18.03	13.30	19.74	6.87	11.16	14.16	11.20	11.59	12.88	12.45	21.46	17.03	13.10	9.87

Tabelle A.5: Datencharakteristiken und Fehlerraten der Verfahren auf ausgewählten Datensätzen

Anhang B

Datengenerator

B.1 Beschreibung von *SCDS* Version 2.10

```
Version 2.10    Gabor_Melli@cs.sfu.ca   1996/04/08

SYNTAX: scds [-hvpc] [-AefgIMmPRrO value] [-DCTd value[,value]] [-X string]

        more @ www.cs.sfu.ca/cs/people/GradStudents/melli/SCDS/parameters.html
        h:      help (this report) [default value]
        v:      verbose report [false]
        p:      pseudo randomness [false]
        c:      plain column banner [false]

        A:      Number of relevant attributes
        e:      Ratio of erroneously entered attribute-values
        f:      File path to hold rules [stdout]
        g:      Ratio of erroneously entered class-values
        I:      Number of irrelevant attributes
        M:      Number of masked relevant attributes
        m:      Ratio of missing attribute-values
        P:      Name of predicted attribute [Class]
        R:      Number of DNF rules
        r:      Rule distribution 0=uniform,1=random,2=standard normal [1]
        O:      Number of objects

        Ranges
        D:      Disjunctions per rule
        C:      Conjunctions per CNF rule component
        T:      Disjunctions per attribute term. e.g. B=(2,5)
        d:      Domain of all predicting attributes

        X:      Explicit data set definition. DOM#,DISJ#,[R,I],[N,O],[V,M]
                R-Relevant, I-Irrelevant, N-Nominal, O-Ordinal, M-Masked
```

```
Examples

1       A verbose report for 5 objects based on 3 unary rules
        from 2 predicting attributes with domain of 5
        % scds -v -O5 -R3 -A2 -d5

2       100 objects which abide by a balanced binary tree.
        Each leaf has a separate class
        % scds -O100 -R4 -A2 -d2 -C1 -T0 -r0

3       An explicit data dictionary definition
        % scds -O13 -R3 -C2 -X12,2,R:7,R,N:5,M:12,I:R -T0.5 -d10
```

B.2 Beispiel *nursery*-Datenbank

```
    ============================================================

    *** Statistic ***

    Nr. attributes = 9 (including target attribute)

    Nr. sym. attributes = 8
    Nr. num. attributes = 0

    Nr. examples = 12960
    Nr. classes = 5

    Class attribute = class
    Frequency: absolute     relative
    not_recom:    4320        0.333
    recommend:       2        0.000
    very_recom:    328        0.025
    priority:     4266        0.329
    spec_prior:   4044        0.312

    Default class is not_recom (33.3%)

    ============================================================
```

Abbildung B.1: Grundlegenden statistische Maße der original *nursery*-Datenbank.

```
============================================================

*** Statistic ***

Nr. attributes = 9 (including target attribute)

Nr. sym. attributes = 8
Nr. num. attributes = 0

Nr. examples = 12960
Nr. classes = 5

Class attribute = class

Frequency: absolute relative
not_recom:    4318    0.333
recommend:       3    0.000
very_recom:    326    0.025
priority:     4267    0.329
spec_prior:   4046    0.312

Default class is not_recom (33.3%)

============================================================
```

Abbildung B.2: Grundlegenden statistische Maße der reproduzierten *nursery*-Datenbank.

```
*** Information-theoretic Measures ***

Gini-index
Gini_sym (class, parents)  = 0.041
Gini_sym (class, has_nurs) = 0.081
Gini_sym (class, form)     = 0.002
Gini_sym (class, children) = 0.005
Gini_sym (class, housing)  = 0.010
Gini_sym (class, finance)  = 0.003
Gini_sym (class, social)   = 0.008
Gini_sym (class, health)   = 0.513

Measure of Relevance
R(class, parents)  = 0.226
R(class, has_nurs) = 0.326
R(class, form)     = 0.211
R(class, children) = 0.217
R(class, housing)  = 0.185
R(class, finance)  = 0.135
R(class, social)   = 0.102
R(class, health)   = 0.402

g-Function (normalised)
g(class, parents)  = -1.650
g(class, has_nurs) = -1.530
g(class, form)     = -1.719
g(class, children) = -1.713
g(class, housing)  = -1.703
g(class, finance)  = -1.716
g(class, social)   = -1.701
g(class, health)   = -0.766

Class Entropy (Hc):  1.7165
Entropy Attribute (parents)  =  1.5850  I_gain_ratio =  0.0407
Entropy Attribute (has_nurs) =  2.3219  I_gain_ratio = -0.1576
Entropy Attribute (form)     =  2.0000  I_gain_ratio = -0.0764
Entropy Attribute (children) =  2.0000  I_gain_ratio = -0.0765
Entropy Attribute (housing)  =  1.5850  I_gain_ratio =  0.0401
Entropy Attribute (finance)  =  1.0000  I_gain_ratio =  0.2642
Entropy Attribute (social)   =  1.5850  I_gain_ratio =  0.0401
Entropy Attribute (health)   =  1.5850  I_gain_ratio =  0.0561

Entropy of Attributes:     1.7077
Mutual Information:        0.1615
Joint Entropy:             3.2627
Equivalent nr. of attrs:  10.6302
Noise Signal Ratio:        9.5759
```

Abbildung B.3: Informationstheoretische Maße der Original *nursery*-Datenbank.

```
*** Information-theoretic Measures ***

Gini-index
Gini_sym (class, parents)  = 0.042
Gini_sym (class, has_nurs) = 0.081
Gini_sym (class, form)     = 0.002
Gini_sym (class, children) = 0.005
Gini_sym (class, housing)  = 0.010
Gini_sym (class, finance)  = 0.003
Gini_sym (class, social)   = 0.009
Gini_sym (class, health)   = 0.512

Measure of Relevance
R(class, parents)  = 0.232
R(class, has_nurs) = 0.309
R(class, form)     = 0.206
R(class, children) = 0.216
R(class, housing)  = 0.187
R(class, finance)  = 0.135
R(class, social)   = 0.145
R(class, health)   = 0.403

g-Function (normalised)
g(class, parents)  = -1.649
g(class, has_nurs) = -1.531
g(class, form)     = -1.720
g(class, children) = -1.711
g(class, housing)  = -1.702
g(class, finance)  = -1.718
g(class, social)   = -1.699
g(class, health)   = -0.768

Class Entropy (Hc):  1.7168
Entropy Attribute (parents)  =  1.5849  I_gain_ratio =  0.0408
Entropy Attribute (has_nurs) =  2.3215  I_gain_ratio = -0.1574
Entropy Attribute (form)     =  1.9998  I_gain_ratio = -0.0763
Entropy Attribute (children) =  1.9999  I_gain_ratio = -0.0765
Entropy Attribute (housing)  =  1.5846  I_gain_ratio =  0.0403
Entropy Attribute (finance)  =  1.0000  I_gain_ratio =  0.2641
Entropy Attribute (social)   =  1.5849  I_gain_ratio =  0.0402
Entropy Attribute (health)   =  1.5848  I_gain_ratio =  0.0563

Entropy of Attributes:     1.7076
Mutual Information:        0.1618
Joint Entropy:             3.2625
Equivalent nr. of attrs:  10.6089
Noise Signal Ratio:        9.5520
```

Abbildung B.4: Informationstheoretische Maße der reproduzierten *nursery*-Datenbank.

B.3 Beispiel *breastLoss*-Datenbank

```
*** Preconditions for V-Statistic ***

1. Coefficient of Multiple Correlation
        Clump_Thickness:              0.691
        Uniformity_of_Cell_Size:      0.928
        Uniformity_of_Cell_Shape:     0.920
        Marginal_Adhesion:            0.770
        Single_Epithelial_Cell_Size:  0.779
        Bare_Nuclei:                  0.776
        Bland_Chromatin:              0.807
        Normal_Nucleoli:              0.768
        Mitoses   :                   0.529

2. BHEP-Test
        Result = 58.577

        BHEP_ALPHA = 0.10  BHEP_BETA = 1.00:
        Critical value = 0.964

        BHEP_ALPHA = 0.05  BHEP_BETA = 1.00:
        Critical value = 0.973

        BHEP_ALPHA = 0.01  BHEP_BETA = 1.00:
        Critical value = 1.007

3. M-Statistic
        Result = 4708.238         degrees of freedom: 45
        chi^2(45,0.95) = 61.66

        (Result and chi^2-value should be approximately equal)

4. Sd Ratio
        Result = 2.118

*** Eigenvalues ***

Eigenvalue[1] = 5.205525478     1.000000

Fract  = 1.000000
Cancor = 0.915889

*** V-Statistic ***

Wilks Lambda
        Result = 0.161147

Bartletts Statistic
        Result = 1264.117270
        chi^2(9,0.95) = 16.92

        (Result should be greater than the chi^2-value)
```

Abbildung B.5: Auszug multivariater Maße der Original *breastLoss*-Datenbank.

```
*** Preconditions for V-Statistic ***

1. Coefficient of Multiple Correlation
        Clump_Thickness:              0.126
        Uniformity_of_Cell_Size:      0.789
        Uniformity_of_Cell_Shape:     0.749
        Marginal_Adhesion:            0.682
        Single_Epithelial_Cell_Size:  0.627
        Bare_Nuclei:                  0.753
        Bland_Chromatin:              0.660
        Normal_Nucleoli:              0.622
        Mitoses   :                   0.381

2. BHEP-Test
        Result = 37.365

        BHEP_ALPHA = 0.10  BHEP_BETA = 1.00:
        Critical value = 0.964

        BHEP_ALPHA = 0.05  BHEP_BETA = 1.00:
        Critical value = 0.973

        BHEP_ALPHA = 0.01  BHEP_BETA = 1.00:
        Critical value = 1.007

3. M-Statistic
        Result = 3231.355         degrees of freedom: 45
        chi^2(45,0.95) = 61.66

        (Result and chi^2-value should be approximately equal)

4. Sd Ratio
        Result = 1.674

*** Eigenvalues ***

Eigenvalue[1] = 8.735604451     1.000000

Fract  = 1.000000
Cancor = 0.947251

*** V-Statistic ***

Wilks Lambda
        Result = 0.102716

Bartletts Statistic
        Result = 1575.984386
        chi^2(9,0.95) = 16.92

        (Result should be greater than the chi^2-value)
```

Abbildung B.6: Auszug multivariater Maße der reproduzierten *breastLoss*-Datenbank.

B.4 Beispiel *crx*-Datenbank

```
*** Information-theoretic Measures ***

Gini-index
Gini_sym (class, A1)  = 0.000
Gini_sym (class, A4)  = NaN
Gini_sym (class, A5)  = 0.031
Gini_sym (class, A6)  = 0.056
Gini_sym (class, A7)  = 0.038
Gini_sym (class, A9)  = 0.546
Gini_sym (class, A10) = 0.204
Gini_sym (class, A12) = 0.003
Gini_sym (class, A13) = 0.010

Measure of Relevance
R(class, A1)  = 0.018
R(class, A4)  = NaN
R(class, A5)  = 0.148
R(class, A6)  = 0.295
R(class, A7)  = 0.170
R(class, A9)  = 0.740
R(class, A10) = 0.451
R(class, A12) = 0.050
R(class, A13) = 0.056

g-Function (normalised)
g(class, A1)  = -1.007
g(class, A4)  = -0.984
g(class, A5)  = -0.984
g(class, A6)  = -0.942
g(class, A7)  = -0.973
g(class, A9)  = -0.556
g(class, A10) = -0.855
g(class, A12) = -1.005
g(class, A13) = -0.999

Class Entropy (Hc):  0.9937
Entropy Attribute (A1)  =  0.8942  I_gain_ratio =  0.0527
Entropy Attribute (A4)  =  0.8116  I_gain_ratio =  0.1023
Entropy Attribute (A5)  =  0.8116  I_gain_ratio =  0.1023
Entropy Attribute (A6)  =  3.5081  I_gain_ratio = -0.5719
Entropy Attribute (A7)  =  1.8060  I_gain_ratio = -0.2955
Entropy Attribute (A9)  =  0.9966  I_gain_ratio = -0.0019
Entropy Attribute (A10) =  0.9894  I_gain_ratio =  0.0023
Entropy Attribute (A12) =  0.9959  I_gain_ratio = -0.0011
Entropy Attribute (A13) =  0.4359  I_gain_ratio =  0.3923

Entropy of Attributes:     1.2499
Mutual Information:        0.0913
Joint Entropy:             2.1523
Equivalent nr. of attrs:  10.8843
Noise Signal Ratio:       12.6909
```

Abbildung B.7: Auszug der informationstheoretischen Maße der Original *crx*-Datenbank.

```
*** Information-theoretic Measures ***

Gini-index
Gini_sym (class, A1)  = 0.001
Gini_sym (class, A4)  = NaN
Gini_sym (class, A5)  = 0.049
Gini_sym (class, A6)  = 0.057
Gini_sym (class, A7)  = 0.030
Gini_sym (class, A9)  = 0.579
Gini_sym (class, A10) = 0.212
Gini_sym (class, A12) = 0.011
Gini_sym (class, A13) = NaN

Measure of Relevance
R(class, A1)  = 0.025
R(class, A4)  = NaN
R(class, A5)  = 0.178
R(class, A6)  = 0.300
R(class, A7)  = 0.158
R(class, A9)  = 0.764
R(class, A10) = 0.459
R(class, A12) = 0.106
R(class, A13) = NaN

g-Function (normalised)
g(class, A1)  = -1.006
g(class, A4)  = -0.990
g(class, A5)  = -0.969
g(class, A6)  = -0.942
g(class, A7)  = -0.982
g(class, A9)  = -0.534
g(class, A10) = -0.849
g(class, A12) = -0.999
g(class, A13) = -0.994

Class Entropy (Hc):  0.9937
Entropy Attribute (A1)  =  0.8558  I_gain_ratio =  0.0746
Entropy Attribute (A4)  =  0.8131  I_gain_ratio =  0.1009
Entropy Attribute (A5)  =  0.7730  I_gain_ratio =  0.1279
Entropy Attribute (A6)  =  3.5404  I_gain_ratio = -0.5751
Entropy Attribute (A7)  =  1.9438  I_gain_ratio = -0.3287
Entropy Attribute (A9)  =  1.0000  I_gain_ratio = -0.0041
Entropy Attribute (A10) =  0.9877  I_gain_ratio =  0.0033
Entropy Attribute (A12) =  0.9915  I_gain_ratio =  0.0011
Entropy Attribute (A13) =  0.4116  I_gain_ratio =  0.4179

Entropy of Attributes:     1.2574
Mutual Information:        0.0962
Joint Entropy:             2.1549
Equivalent nr. of attrs:  10.3319
Noise Signal Ratio:       12.0741
```

Abbildung B.8: Auszug der informationstheoretischen Maße der reproduzierten *crx*-Datenbank.

Literaturverzeichnis

[A. Linden 2002] A. Linden, Gartner G.: Emerging Trends in Data Mining Through 2010. In: *Research note T-16-1727, www.gartner.com* (2002)

[Aamodt 1995] Aamodt, Agnar: CBR in Context: The Technology and its Impacts. In: *Case-based reasoning; A new force in advanced systems development* Unicom Seminars, 27 April 1995 (1995), S. 9–23.

[Aamodt und Plaza 1994] Aamodt, Agnar ; Plaza, Enric: Case-Based Reasoning; Foundational Issues, Methodological Variations, and System Approaches. In: *AI Communications, Vol.7, No.1* (1994), S. 39–59

[Agrawal u. a. 1993] Agrawal, R. ; Imielinski, T. ; Swami, A.: Mining Associations between Sets of Items in Massive Databases. In: *Proc. Of the ACM SIGMOD Int'l Conference on Management of Data, Washington D.C*, May 1993, S. 207–216

[Agrawal u. a. 1996] Agrawal, R. ; Mannila, H. ; Srikant, R. ; Toivonen, H. ; Verkamo, A. I.: Fast Discovery of Association Rules. In: Fayyad, U. M. (Hrsg.) ; Piatetsky-Shapiro, G. (Hrsg.) ; Smyth, P. (Hrsg.) ; Uthurusamy, R. (Hrsg.): *Advances in Knowledge Discovery and Data Mining.* Menlo Park, California : AAAI Press, 1996, S. 307–328

[Agrawal und Srikant 1994] Agrawal, R. ; Srikant, R.: Fast Algorithms for Mining Assciation Rules. In: *Proceedings of the 20th Conference on Very Large Databases (VLDB'94), Santiago, Chile*, 1994

[Angele u. a. 1996] Angele, J. ; Fensel, D. ; Studer, R.: Domain and Task Modelling in MIKE. In: Sutcliffe, A.G. (Hrsg.) ; Assche, F. van (Hrsg.) ; Benyon, D. (Hrsg.): *Domain Knowledge for Interactive System Design, Proceedings of the IFIP WG8.1/13.2 Joint Working Conference on Domain Knowledge for Interactive System Design.*, Chapman & Hall, Geneva, May 1996

[Bacher 1996] Bacher, J.: *Clusteranalyse.* München : Oldenbourg Verlag, 1996

[Baim 1988] Baim, P.W.: A Methode for Attribute Selection in Inductive Learning Systems. In: *IEEE Trans. on Pattern Analysis and Machine Intelligence, PAMI–10* 10 (1988), S. 888–896

[Barnett und Lewis 1994] Barnett, V. ; Lewis, T.: *Outliers in Statistical Data*. 3. John Wiley & Sons, 1994 (Wiley Series in Probability and Mathmatical Statistics)

[Becker 1995] Becker, Lothar: *Anforderungsanalyse für den Einsatz automatischer Lernverfahren in Datenbanken am Beispiel eines Qualitäts-Informations-Systems in der Automobilindustrie*, Universität Kaiserslautern AG Künstliche Intelligenz - Expertensysteme, Diplomarbeit, 1995

[Bergmann und Stahl 1998] Bergmann, R. ; Stahl, S.: Similarity Measures for Object-Oriented Case Representations. In: *Proceedings of the European Workshop on Case-Based Reasoning, EWCBR'98*, 1998

[Bergmann 1998a] Bergmann, Ralph: On the Use of Taxonomies for Representing Case Features and Local Similarity Measures. In: *Proceedings of the 6th German Workshop on Case-Based Reasoning (GWCBR'98)*, 1998

[Bergmann 1998b] Bergmann, Ralph. *Skript zur Vorlesung Fallbasiertes Schließen*. 1998b

[Borgelt u. a. 1996] Borgelt, C. ; Gebhard, J. ; Kruse, R.: Concepts for Probabilistic and Possibilistic of Decission Trees on Real World Data. In: *Proc. of the EUFIT 96*, 1996, S. 1556–1560,Volume 3

[Borgelt und Kruse 1997a] Borgelt, C. ; Kruse, R.: Evaluation Measures for Learning Probabilistic and Possibilistic Networks. In: *Proc. of the FUZZ–IEEE'97*, 1997

[Borgelt und Kruse 1997b] Borgelt, C. ; Kruse, R.: Some Experimental Results on Learning Probabilistic and Possibilistic Networks with Different Evaluation Measures. In: *Proc. of the ECSQARU/FAPR'97*, 1997

[Borgelt u. a. 1998] Borgelt, Ch. ; Kruse, R. ; Lindner, G.: Lernen probabilistischer und possibilistischer Netze aus Daten: Theorie und Anwendung. In: *KI Fachzeitschrift Themenheft Data Mining* (1998)

[Borgelt und Kruse 1998] Borgelt, Chr. ; Kruse, R.: Attributauswahlmaße für die Induktion von Entscheidungsbäumen: Ein Überblick. In: Nakhaeizadeh, G. (Hrsg.): *Data Mining: Theoretische Aspekte und Anwendungen*. Heidelberg, Germany : Physica-Verlag, 1998, S. 77–98

[Brachman und Anand 1996] Brachman, R. ; Anand, T.: Advances in Knowledge Discovery and Data Mining. In: Fayyad, U. (Hrsg.) ; Piatetsky-Shapiro, G. (Hrsg.) ; Smyth, P. (Hrsg.) ; Uthurusamy, R. (Hrsg.): *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press, 1996, Kapitel The Process of Knowledge Discovery in Databases: A Human–Centered Approach, S. 33–52

[Brandt 1981] Brandt, S.: *Datenanalyse: mit statistischen Methoden und Computerprogrammen.* 2. erw. Mannheim : Bibliographisches Institut, 1981

[Brazdil u. a. 2003] Brazdil, Pavel ; Soares, C. ; Costa, J. Pinto D.: Ranking Learning Algorithms: Using IBL and Meta-Learning on Accuracy and Time Results. In: *Machine Learning* 50(3) (2003), S. 251–277

[Breiman u. a. 1984] Breiman, L. ; Friedman, J.H. ; Olshen, R.A. ; Stone, C.J.: *Classification and Regression Trees.* Pacific Grove : Wadsworth and Brooks, 1984

[Breuker und van de Velde 1994] Breuker, J. ; Velde, W. van d.: *CommonKADS Library for Expertise Modelling.* IOS Press, 1994

[Brodley und Smyth 1997] Brodley, C.E. ; Smyth, P.: Applying Classification Algorithms in Practice. In: *Journal of Statistics and Computing* (1997)

[Büchter und Wirth 1998] Büchter, Oliver ; Wirth, Rüdiger: Discovery of Association Rules over Ordinal Data: A New and Faster Algorithm and its Application to Basket Analysis. In: *Research and Development in Knowledge Discovery and Data Mining (Conference Proceedings of PAKDD-98).* Melbourne, Australia, April 1998

[CBR-Works 1998] CBR-Works: *CBR-Works 3 Reference Manual.* : TECINNO GmbH, 1998

[Chapman u. a. 1999] Chapman, Pete ; Clinton, Julian ; Hejlesen, Jens H. ; Kerber, Randy ; Khabaza, Tom ; Reinartz, Thomas ; Wirth, Rüdiger. *The Current CRISP-DM Process Model for Data Mining.* http://www.ncr.dk/CRISP/index.html. 1999

[Chow und Liu 1968] Chow, C.K. ; Liu, C.N.: Approximate Discrete Probability Distributions with Dependence Trees. In: *Trees. IEEE Trans on Information Theory, Volume 14*, IEEE, 1968, S. 462–467

[Clementine™1998] Clementine™ : *Clementine™ Data Mining System, Installation and Administration Guide.* Integral Solutions Limited, 1998

[Consortium 1993] Consortium, M.: Final public report / Esprit II Project 2154. 1993. – Forschungsbericht

[Cooper und Herskovits 1992] Cooper, G.F. ; Herskovits, E.: A Bayesian Method for the Induction of Probabilistic Networks from Data. In: *Machine Learning* 9 (1992), S. 309–347

[Cornell 1990] Cornell, J. A.: *Experiments with Mixtures: Designs, Models and the Analysis of Mixture Data.* 2. New York : Wiley, 1990

[Cybenko 1988] Cybenko, G.: Continuous valued neural networks with two hidden layers are sufficient / Department of Computer Science, Tufts University. Medford, Massachusetts, 1988. – Technical Report

[Cybenko 1989] Cybenko, G.: Approximation by superpositions of a sigmoidal function. In: *Mathematics of Controls, Signals, and Systems* 2 (1989), S. 303–314

[Deutsche Gesellschaft für Qualität e.V. 1995] Deutsche Gesellschaft für Qualität e.V.: *Begriffe zum Qualitätsmanagement*. Bd. DGQ–Schrift 11–04. Beuth Verlag GmbH, Berlin, 1995

[Engels u. a. 1997] Engels, E. ; Lindner, G. ; Studer, R.: A Guided Tour through the Data Mining Jungle. In: David Heckerman, Daryl P. (Hrsg.): *Proc. of the 3rd International Conference on Knowledge Discovery and Data Mining*, AAAI Press, 1997, ISBN 1-57735-027-8, 1997. – KDD 1997, Newport Beach, California, USA, August 14-17, S. 163–166

[Engels 1996] Engels, R.: Planning Tasks for Knowledge Discovery in Databases; Performing Task-oriented User-Guidance. In: Simounis, E. (Hrsg.) ; Han, J. (Hrsg.) ; Fayyad, U. (Hrsg.): *Proc. of the 2nd International Conference on Knowledge Discovery and Data Mining*, AAAI Press, 1996, ISBN 1-57735-004-9, 1996, S. 170–175

[Engels 1999] Engels, R.: *Component–based User Guidance for Knowlegde Discovery and Data Mining Processes.* Institute AIFB University Karlsruhe, D-76128 Karlsruhe, Germany, University Karlsruhe, Germany, Dissertation, Februar 1999. – 170–175 S. – Sankt Augustin: Infix, 1999 (Dissertationen zur künstlichen Intelligenz; Bd.211)

[Fayyad 1999] Fayyad, U.: Editorial. In: *SIGKDD Explorations* 1 (1999), Juni, S. 1– 3. – http://research.microsoft.com/datamine/sigkdd

[Fayyad u. a. 1996] Fayyad, U. ; Piatetsky-Shapiro, G. ; Smyth, P.: Knowledge Discovery and Data Mining: Towards a Unifying Framework. In: Simoudis, E. (Hrsg.) ; Han, J. (Hrsg.) ; U.Fayyad (Hrsg.): *Proc. of the 2nd International Conference on Knowledge Discovery and Data Mining.* Portland, Oregon, August 1996, S. 82 – 88

[Fisher 1987] Fisher, Douglas H.: Knowledge Acquisition Via Incremental Conceptual Clustering. In: *Machine Learning* 2 (1987), Nr. Douglas H. Fisher, S. 139 – 172

[Frawley u. a. 1991] Frawley, W. ; Piatetsky-Shapiro, G. ; Matheus, C.: Knowledge Discovery in Databases: An Overview. In: Piatetsky-Shapiro, G. (Hrsg.) ; Frawley, W. (Hrsg.): *Knowledge Discovery in Databases.* Cambridge, Mass. : AAAI/MIT Press., 1991, S. 1–27

[Gama und Brazdil 1995] GAMA, Joao ; BRAZDIL, Pavel: Characterization of Classification Algorithms. In: MAMEDE, N. (Hrsg.) ; FERREIRA, C. (Hrsg.): *Advances on Artificial Intelligence - EPIA95*, Springer Verlag, 1995

[Giraud-Carrier und Hilario 1998] GIRAUD-CARRIER, Ch. ; HILARIO, M. ; GIRAUD-CARRIER, Ch. (Hrsg.) ; HILARIO, M. (Hrsg.): ECML'98 Workshop Notes – Upgrading Learning to the Meta-Level: Model Selection and Data Transformation / Computer Science, TU Chemnitz. 1998 (CSR-98-02). – Forschungsbericht

[Giraud-Carrier und Pfahringer 1999] GIRAUD-CARRIER, Ch. (Hrsg.) ; PFAHRINGER, B. (Hrsg.): *Proceedings of Workshop Recent Advances in Meta-Learning and Future Work at the ICML 1999, Bled, Slowenia, June, 26-31, 1999.* 1999

[Gordon 1999] GORDON, A.D.: *Classification.* 2 nd. London : Chapmann and Hall, 1999 (Monographs on Statistics and Applied Probability)

[Grimmer und Mucha 1998] GRIMMER, U. ; MUCHA, H.: Datensegmentierung Mittels Clusteranalyse. In: NAKHAEIZADEH, G. (Hrsg.): *Data Mining: Theoretische Aspekte und Anwendungen.* Heidelberg, Germany : Physica-Verlag, 1998, S. 109–141

[Hand 1994a] HAND, D.: Deconstructing Statistical Questions. In: *Journal of the Royal Statistical Society* (1994), S. 317 – 356

[Hand 1994b] HAND, D.: Statistical Strategy: step1. In: CHEESEMAN, P. (Hrsg.) ; OLDFORD, R.W. (Hrsg.): *Selecting Models from Data: AI and Statistics IV* Bd. 89. Lecture Notes in Statistics, 1994b

[Hand 1999] HAND, David J.: Statistics and Data Mining: Intersecting Disciplines. In: *SIGKDD Explorations* 1 (1999), S. 16 – 19. – http://research.microsoft.com/datamine/sigkdd/

[Hartung und Elpelt 1995] HARTUNG, J. ; ELPELT, B.: *Multivariate Statistik: Lehr- und Handbuch der angewandten Statistik.* Oldenbourg Verlag GmbH, München, 1995

[Hartung u. a. 1995] HARTUNG, J. ; ELPELT, B. ; KLÖSENER, K.-H.: *Statistik: Lehr- und Handbuch der angewandten Statistik.* 10. München : Oldenbourg, 1995

[Henze und Wagner 1991] HENZE, N. ; WAGNER, T.: A New Approach to the BHEP Tests for Multivariate Normality. In: *unknown* (1991)

[Hipp u. a. 1998] HIPP, Jochen ; MYKA, Andreas ; WIRTH, Rüdiger ; GÜNTZER, Ulrich: A New Algorithm for Faster Mining of Generalized Association Rules. In: *Proceedings of the 2nd European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD '98).* Nantes, France, September 1998

[Huber 1977] HUBER, P. J.: *Robust Statistical Procedures.* Philadelphia, Pennsylvania : Society for Industrial and Applied Mathmatics, 1977 (CBMS-NSF: Regional Conference Series in Applied Mathmatics 27)

[Kalousis 2002] KALOUSIS, Alexandros: *Algorithm Selection Via Meta–Learning*, University Geneve, Dissertation, 2002

[Kauderer und Nakhaeizadeh 1998] KAUDERER, H. ; NAKHAEIZADEH, G.: Skalierung als alternative Datentransformation und deren Auswirkungen auf die Leistungsfähigkeit von Supervised Learning Algorithmen. In: NAKHAEIZADEH, G. (Hrsg.): *Data Mining – Theoretische Aspekte und Anwendungen.* Physica-Verlag, 1998, Kapitel 5, S. 99–108

[Kietz und Wrobel 1992] KIETZ, Jörg-Uwe ; WROBEL, Stefan: Controlling the Complexity of Learning in Logic through Syntactic and Task-Oriented Models. In: MUGGLETON, Stephen (Hrsg.): *Inductive Logic Programming.* London : Academic Press, 1992. – Also available as Arbeitspapiere der GMD No. 503, 1991, Kapitel 16, S. 335 – 360

[Kleiner 1998] KLEINER, Thorsten: *Entwicklung eines Datengenerators zur Evaluierung von Datencharakteristiken*, Technische Universität Braunschweig, Diplomarbeit, 1998. – Betreuer: Guido Lindner

[Klenk 1997] KLENK, T.: *Ein Verfahren zur Suche von sequentiellen Mustern*, Universität Stuttgart, Institut für Informatik, Diplomarbeit, Dezember 1997

[Klose 1997] KLOSE, A.: *Ein KDD-Ansatz zur Prognose und Früherkennung von Ausfallquoten im Automobilbereich*, Universität Braunschweig, Diplomarbeit, 1997. – Erstellt in Zusammenarbeit mit der DaimlerChrysler AG. Betreut von Guido Lindner.

[Kohavi und Sommerfield 2002] KOHAVI, R. ; SOMMERFIELD, D.: MLC++. In: KLÖSGEN, W. (Hrsg.) ; ZYTKOW, J. (Hrsg.): *Handbook of Data Mining and Knowledge Discovery.* 198 Madison Avenue, New York : Oxford University Press, 2002, Kapitel 24.1.2, S. 548–553

[Kohavi u. a. 1997] KOHAVI, R. ; SOMMERFIELD, D. ; DOUGHERTY, J.: Data Mining using MLC++, A Machine Learning Library in C++. In: *Int. Journal on Artificial Intelligence Tools* 6 (1997), Nr. 4, S. 537–566. – [http://www.sgi.com/Technology/mlc]

[Kohonen 1997] KOHONEN, T.: *Self-Organizing Maps.* Second Extended Edition. Springer, Berlin, Heidelberg, New York, 1997 (Springer Series in Information Sciences)

[Kolodner 1993] KOLODNER, Janet: *Case-Based Reasoning.* San Mateo : Morgan Kaufmann Publishers, 1993

[Kolodner 1996] In: KOLODNER, Janet L.: *Making the Implicit Explicit: Clarifying the Principles of Case-Based Reasoning.* Menlo Park : The MIT Press, 1996, S. 349–370

[Kolodner und Leake 1996] In: KOLODNER, Janet L. ; LEAKE, David B.: *A Tutorial Introduction to Case-Based Reasoning.* Menlo Park : The MIT Press, 1996, S. 31–65

[Kononenko 1995] KONONENKO, I.: On Biases in Estimating Multi-Valued Attributes. In: FAYYAD, U. (Hrsg.) ; UTHURUSAMY, R (Hrsg.): *Proc. 1st Int.Conference on Knowledge Discovery and DataMining*, AAAI, 1995, S. 1034–1040

[Krichevsky und Trofimov 1983] KRICHEVSKY, R.E. ; TROFIMOV, V.K.: The Performance of Universal Coding. In: *IEEE Trans. on Information Theory* IT-27 (1983), S. 199–207

[Lavrač und Džeroski 1994] LAVRAČ, N. ; DŽEROSKI, S.: *Inductive Logic Programming: Techniques and Applications.* Ellis Horwood, 1994

[Leake 1996] LEAKE, David B. ; LEAKE, David B. (Hrsg.): *Case-Based Reasoning Experiences, Lessons & Future Directions.* Menlo Park : The MIT Press, 1996

[Lebowitz 1986] LEBOWITZ, M.: Integrated Learning: Controlling Explanation. In: *Cognitive Science* 10 (1986), Nr. 2

[Legner u. a. 1996] LEGNER, C. ; OHL, S. ; NAKHAEIZADEH, R.: Mining in Warranty Cost Data. In: *Proc. of the Thirteenth European Meeting on Cybernetics and Systems Research* (1996)

[Lindner u. a. 2002] LINDNER, G. ; ENGELS, R. ; STUDER, R.: Selection of Data Mining Methodes for Tasks. In: KLÖSGEN, W. (Hrsg.) ; ZYTKOW, J. (Hrsg.): *Handbook of Data Mining and Knowledge Discovery.* 198 Madison Avenue, New York : Oxford University Press, 2002, Kapitel 17.2, S. 444–450

[Lindner und Hipp 1999] LINDNER, G. ; HIPP, J.: Analysing Warranty Claims of Automobiles; An Application Description Following the CRISP-DM Data Mining. In: LUCAS CHI KWONG HUI, DIK LUN LEE (EDS.) (Hrsg.): *Internet Applications, 5th International Computer Science Conference, ICSC'99, Proceedings. Lecture Notes in Computer Science, Vol. 1749, Springer, 1999, ISBN 3-540-66903-5*, Dez. 1999

[Lindner und Klose 1997] LINDNER, G. ; KLOSE, A.: ML and Statistics for Trend Prognosis of complaints in the Automobile Industry. In: *Proccedings of the Workshop on Machine Learning Applications in the real world: Methodological Aspects and Implications, at the ICML 1997, Nashville TN.*, 1997, S. 20 – 26

[Lindner und Morik 1995] Lindner, G. ; Morik, K.: Coupling a Relational Learning Algorithm with a Database System. In: Kodratoff, Y. (Hrsg.) ; Nakhaeizadeh, G. (Hrsg.) ; Taylor, Charles (Hrsg.): *Statistics, Machine Learning and Knowledge Discovery in Databases* MLnet Familiarization Workshops, 28-29 April 1995 on the ECML'95, 1995, S. 163–168

[Lindner und Studer 1999a] Lindner, G. ; Studer, R.: AST: Support for Algorithm Selection with a CBR Approach. In: Zytkow, J. (Hrsg.) ; Rauch, J. (Hrsg.): *Principles of Data Mining and Knowledge Discovery, Third European Conference, PKDD'99 ,Prague.* Berlin : Springer Verlag, September 1999a, S. 418–424

[Lindner und Studer 1999b] Lindner, G. ; Studer, R.: Forecasting the Fault Behaivior Rate for Cars. In: *Proccedings of the Workshop From Machine Learning to Knowledge Discovery in Databases , at the ICML 1999,Bled, Solwenien.*, 1999, S. 20 – 26

[Lindner und Studer 2000] Lindner, G. ; Studer, R.: Algorithm Selection Support for Classification. In: Decker, R. (Hrsg.) ; (Eds.), W. G. (Hrsg.): *Classification and Information Processing at the Turn of the Millennium, Proc. Of the 23rd Annual Conf. Of the Gesellschaft Fuer Klassifikation E.V., University of Bielefeld, March 1999, Studies in Classification, Data Analysis, and Knowledge Organization.* Berlin : Springer-Verlag, 2000, 2000

[Lindsay 1995] Lindsay, B. G.: *Mixture Models: Theory, Geometry and Applications.* Hayward, California : Institute of Mathmatical Statistics, 1995 (Regional Conference Series in Probability and Statistics 5)

[Matheus u. a. 1993] Matheus, C.J. ; Chan, P. ; Piatetsky-Shapiro, G.: Systems for Knowledge Discovery in Databases. In: *IEEE Transactions on Knowledge and Data Engineering* 5 (1993), Dezember, Nr. 6, S. 903–913

[Melli 1996] Melli, G. *Synthetic Classification Data Sets (SCDS 2.10).* [http://www.cs.sfu.ca/cs/people/GradStudents/melli/SCDS]. 1996

[Merz und Murphy 1996] Merz, C.J. ; Murphy, P.M.: UCI Repository of Machine Learning Databases. In: *Http://www.ics.uci.edu/ mlearn/MLRepository,* (1996)

[Michie u. a. 1994] Michie, D. ; Taylor, C. ; Spiegelhalter, D.: *Machine Learning, Neural and Statistical Classification.* Ellis Hoorwood, 1994

[Mitchell 1997] Mitchell, T.: *Machine Learning.* McGraw Hill, 1997

[Morik und Scholz 2003] Morik, Katharina ; Scholz, Martin: The MiningMart Approach to Knowledge Discovery in Databases. In: Zhong, Ning (Hrsg.) ; Liu, Jiming (Hrsg.): *Handbook of Intelligent IT.* IOS Press, 2003, to appear., 2003

[Mucha 1992] Mucha, H.J.: *Clusteranayse mit Mikrocomputern.* Berlin : Akademie Verlag, 1992

[Muggleton und Raedt 1993] Muggleton, St. ; Raedt, L. D.: Inductive Logic Programming: Theory and Methods / Department of Computing Science, K.U. Leuven. 1993. – Forschungsbericht

[Muggleton 1990] In: Muggleton, Stephen: *Inductive Logic Programming.* 1990

[Nakhaeizadeh u. a. 1998] Nakhaeizadeh, G. ; Reinartz, Th. ; Wirth, R.: Wissensentdeckung in Datenbanken und Data Mining: Ein Überblick. In: Nakhaeizadeh, G. (Hrsg.): *Data Mining – Theoretische Aspekte und Anwendungen.* Physica-Verlag, 1998, Kapitel 1, S. 1–33

[Oestereich u. a. 1999] Oestereich, B. ; P.Hruschka ; Reinhold, N. Josuittisand H. Kocherand H.Krasemannand M.: *Erfolgreich mit Objektorientierung: Vorgehensmodelle und Managementpraktiken für die objektorientierte Softwareentwicklung.* Rosenheimer Str.145, D-81671 München : Oldenbourg Wissenschaftsverlag GmbH, 1999. – ISBN 3-486-25277-1

[Quinlan 1990] Quinlan, J. R.: Learning Logical Definitions from Relations. In: *Machine Learning* 5 (1990), Nr. 3, S. 239 – 266

[Quinlan 1993] Quinlan, J. R.: *C4.5: Programs for Maschine Learning.* San Mateo, CA : Morgan Kaufmann, 1993

[Quinlan 1987] Quinlan, J.R.: Rule induction with statistical data – a comparison with multiple regression. In: *Journal of the Operational Research Society* 38 (1987), S. 347 – 352

[Quinlan 1986] Quinlan, R.J.: Induction of Decision Trees. In: *Machine Learning* 1 (1986), Nr. 1, S. 81–106

[Reich 1994] Reich, Y.: Macro and Micro Perspectives of Multistrategy Learning. In: *Machine Learning: A Multistrategy Approach* Vol. IV, Morgan Kaufmann, San Francisco, S. 379-401, 1994. (1994), S. 379–401

[Rey 1993] Rey, W. J. J.: *Introduction to Robust and Quasi-Robust Statistical Methods.* Berlin : Springer-Verlag, 1993

[Rissannen 1983] Rissannen, J.: A Universal Prior for Integers and Estimations by Minimum. In: *Annals of Staistics* 11 (1983), S. 416–431

[Rissannen 1995] Rissannen, J.: Complexityand Its Applications. In: *Proc. Workshop on Model Uncertainty and Model Robustness* Bd. 11. Bath,England, 1995, S. 416–431

[Sang 2002] Sang, K.: Overview of SAS Enterprise Miner. In: Klösgen, W. (Hrsg.) ; Zytkow, J. (Hrsg.): *Handbook of Data Mining and Knowledge Discovery.* 198 Madison Avenue, New York : Oxford University Press, 2002, Kapitel 24.2.5, S. 589–600

[Schaffer 1994] Schaffer, Cullan: A conservation law for generalization performance. In: Cohen, W. W. (Hrsg.) ; Hirsh, H. (Hrsg.): *Machine Learning: Proceedings of the Eleventh International Conference, Rutgers University, New Brunswick, NJ, USA, July 10-13.* Palo Alto ,CA : Morgan Kaufmann Publisher, 1994, S. 259–265

[Schreiber u. a. 1993] Schreiber, A.T. ; Wielinga, B.J. ; Breuker, J.A.: *KADS: A Principled Approach to Knowledge-Based System Development.* London : Academic Press, 1993

[Silverman 1986] Silverman, B. W.: *Density Estimation for Statistics and Data Analysis.* Reprinted. London : Chapman & Hall, 1986 (Monographs on Statistics and Applied Probability 26)

[Srikant und Agrawal 1996a] Srikant, R. ; Agrawal, R.: Mining Quantitative Association Rules in Large Relational Tables. In: *Proceedings of the 1996 ACM SIGMOD Conference on Management of Data.* Montreal, Canada, June 1996a

[Srikant und Agrawal 1996b] Srikant, R. ; Agrawal, R.: Mining Sequential Patterns: Generalizations and Performance Improvements. In: *Proc. of the 5th International Conference on Extending Databases Technology, EDBT*, 1996

[Srikant und Agrawal 1995] Srikant, R. ; Agrawal, R.: Mining Generalized Association Rules. In: *Proceedings of the 21th Conference on Very Large Databases (VLDB'95), Zürich, Switzerland*, September 1995

[Theusinger 1998] Theusinger, C.: *Beschreibung von Datensätzen zum Ziel der Auswahl von Data Mining-Algorithmen*, Universität Karlsruhe - Fakultät Informatik, Diplomarbeit, 1998

[Wirth u. a. 1997] Wirth, R. ; Shearer, C. ; Grimmer, U. ; Reinartz, T. ; Schloesser, J. ; Breitner, C. ; Engels, R. ; Lindner, G.: Towards Process–Oriented Tool Support for KDD. In: *Proceedings of 1st Intern. Conference on Principles of KDD*, Springer Verlag, 1997. – LNCS Serie

[Wolpert 1994] Kap. The relationship between PAC, the statistical physics framework, the Bayesian framework and the VC framework In: Wolpert, D.H.: *The Mathematics of Generalizations.* Addison Wesley, 1994

[Zhou und Dillon 1991] Zhou, X. ; Dillon, T.S.: A statistical-heuristic Feature Selection Criterion for Decision Tree Induction. In: *IEEE Trans. on Pattern Analysis and Machine Intelligence* PAMI-13 (1991), S. 834–841

[Zintz 1996] Zintz, U.: Evaluierung von INES für Data Mining anhand einer Anwendung aus der Automobilindustrie / Universität Karlsruhe; Fakultät für Informatik; Institut für Programmstrukturen und Datenorganisation; In Zusammenarbeit mit der Daimler Chrysler AG. Betreuer: Guido Lindner. 1996. – Forschungsbericht

[Zintz 1998] Zintz, Ulrike: *Evaluierung von Datencharakteristiken zur Auswahl von Data Mining Verfahren*, Universität Karlsruhe, Diplomarbeit, 1998. – Betreuer: Dipl. Inf. Guido Lindner