Diplomarbeit

Paarweise Hierarchische Klassifikation

27. Januar 2008

Fachbereich:	Informatik
Fachgebiet:	Knowledge Engineering
Verfasser:	Jan Frederik Sima
Betreuer:	Prof. Johannes Fürnkranz
	Eneldo Loza Mencia

Erklärung

Hiermit versichere ich, die vorliegende Diplomarbeit ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus den Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 27. Januar 2008

. . . . . .

Unterschrift

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

1 Einleitung

1.1 Motivation

Mit den fortschreitenden technischen Möglichkeiten ist es machbar immer größere Mengen an Daten zusammen zu tragen und zu verarbeiten. Durch die zunehmende Vernetzung verschiedenster Datenquellen wächst gleichzeitig die Zahl der zugänglichen Informationen täglich an. Eines der bemerkenswertesten Beispiele für die stark steigende Tendenz dieser Entwicklung ist das Internet und dessen Wachstum innerhalb der letzen, noch nicht einmal zwei, Jahrzehnten. Heute ist über das Internet zu jedem erdenklichen Thema eine Vielzahl von Dokumenten zu finden. Aber auch auf anderen Gebieten wie der Genforschung ist es machbar Milliarden von einzelnen Information zusammen zutragen, in denen womöglich noch unbekannte Erkenntnisse stecken. Das Problem, welches mit zunehmender Daten- und Informationsmenge einhergeht, ist die proportional fallende Möglichkeit diese Daten menschlich zu verarbeiten und damit bestmöglich, bzw. überhaupt, zu nutzen. Die Aufgabe aus riesigen Bergen von Daten, die wichtigen herauszupicken oder auch die wichtigen Konklusionen aus diesen Daten abzuleiten, ist die Aufgabe des Maschinellen Lernens und insbesondere des Data Mining.

Diese Diplomarbeit beschäftigt sich mit der Aufgabenstellung der Klassifikation. Die Klassifikation ist einer der Teilbereiche des Maschinellen Lernens und beinhaltet die maschinelle Einordnung von Daten in bestimmte Kategorien. Die Klassifikation kann so beispielsweise neue Internetseiten für eine Suchmaschine automatisch dem, auf der Seite behandelten, Thema zuordnen. Ebenso werden Methoden der Klassifikation benutzt um Krankheitsbilder an Hand von gegebenen Symptomen des Patienten zu erstellen. Genauer wird es in dieser Arbeit um eine Klassifikationsmethode gehen, welche darauf ausgelegt ist auf hierarchischen Daten zu arbeiten. Hierarchische Daten sind Daten welche in Kategorien einordnet werden, die wiederum in einer Hierarchie aufgebaut sind. Die Yahoo Directories[1], ein Verzeichnis indem Internetseiten nach Themen und sukzessive weiterspezialisierenden Unterthemen geordnet sind, sowie die U.S. Patentdatenbank[2] sind Beispiele für hierarchische Datensätze. Das Thema der hierarchischen Klassifikation wurde mit verschiedenen Ansätzen von unter anderem [Dekel *et al.*, 2004], [Blockeel *et al.*, 2006], [Koller and Sahami, 1997], [Sun and Lim, 2001], [Cesa-Bianchi *et al.*, 2004] und [McCallum *et al.*, 1998] behandelt.

Ab einer gewissen Größe von Datensammlungen ist eine Hierarchie auf den Kategorien fast schon notwendig, um einen topologischen Überblick über die Daten für den Menschen zu bewahren. Bei den meisten großen Datensätzen ergibt sich eine Hierarchie weiterhin dadurch, dass die Klassen, in welche diese Daten eingeordnet werden sollen, nicht völlig unabhängig voneinander sind, sondern bestimmte Gemeinsamkeiten und Unterschiede miteinander haben. Eine Hierarchie stellt letztendlich eine Struktur dar, welche die Klassen nach eben diesen Verhältnissen ordnet.

Eine Klassenhierarchie ist also eine Informationsquelle, welche Auskunft über die Unterschiede zwischen den einzelnen Klassen gibt. Sie macht Aussagen darüber, welche Klassen sich sehr ähnlich sind und welche Klassen wahrscheinlich nur sehr wenige oder keine Gemeinsamkeiten miteinander

[1] http://dir.yahoo.com/
[2] http://www.uspto.gov/patft/

haben. Bei bestimmten Hierarchien ist weiterhin gegeben, welche Klassen einander bedingen, dass heißt welche Klasse nicht ohne eine andere gesetzt werden kann. Es liegt auf der Hand, dass man bei der Klassifikation von Daten eine solche Informationsquelle nicht unbenutzt lassen will, sondern im Gegenteil die vorhandenen Informationen in die Klassifikation miteinbeziehen möchte, um so die Ergebnisse zu maximieren. Diese erfolgreiche Miteinbeziehung einer Klassenhierarchie in die Klassifikation der dazugehörigen Daten ist das Ziel dieser Diplomarbeit. Es soll mit der paarweisen hierarchischen Klassifikation eine Abwandlung der paarweisen Klassifikation modelliert werden, so dass die Klassenhierarchie beim Lernprozess miteingebunden wird. Dadurch soll untersucht werden, wie sich die Klassifikationsergebnisse durch diese „hierarchische“ Erweiterung verändern und ob dadurch eine Verbesserung der Klassifikationsergebnisse möglich ist.

1.2 Gliederung

Im Kapitel „2. Grundlagen“ werden die für diese Arbeit wichtigen Begriffe und Definitionen aus dem Bereich der Klassifikation vorgestellt. Es wird auf die verschiedenen Problemstellungen unterschiedlicher Klassifikationsaufgaben eingegangen. Weiter wird die konkrete Methode der paarweisen Klassifikation definiert und mit ihren Eigenschaften ausgeführt. Die, dieser Arbeit zugrunde liegende, Semantik von Klassenhierarchien wird ebenfalls dargestellt.
Im nächsten Kapitel „3. Paarweise Hierarchische Klassifikation“ wird das Modell einer hierarchischen Abwandlung der paarweisen Klassifikation vorgestellt und definiert. Es werden theoretische Überlegungen über die Unterschiede zwischen diesem Modell und der paarweisen Klassifikation angestellt und zusätzlich auch mögliche Variationen dieses Modells angedacht.
Im Kapitel „4. Äquivalenz zwischen paarweiser hierarchischer Klassifikation und der Pachinko Maschine“ wird zunächst das Modell der Pachinko Maschine vorgestellt und dann die unter bestimmten Umständen gegebene Äquivalenz zur paarweisen hierarchischen Klassifikation bewiesen.
Anschließend werden in „5. Versuche und Ergebnisse“ alle für die Versuche verwendeten Datensätze mit ihren Eigenschaften beschrieben. Es wird ein Vergleich der Ergebnisse der paarweisen Klassifikation und der paarweisen hierarchischen Klassifikation auf allen Datensätzen angestellt und mit Hinsicht auf die vorherigen theoretischen Überlegungen interpretiert. Zum Schluss wird eine mögliche Methode, das Vorhandensein einer Hierarchie in den Daten eines Datensatzes zu testen, vorgestellt und auch auf den relevanten Datensätzen angewendet. Zusammenfassend wird unter „6. Fazit und Ausblick“ ein Überblick über die Gesamtaussage der Ergebnisse der Diplomarbeit und über mögliche Anknüpfungspunkte dieser Thematik gegeben.

2 Grundlagen

2.1 Maschinelles Lernen

Maschinelles Lernen ist ein weit gefächertes Teilgebiet der Künstlichen Intelligenz. Abstrakt beschrieben ist es das Ziel des Maschinellen Lernens Algorithmen und Methoden zu entwickeln, die es Rechnern erlauben zu lernen. Wenn ein Programm lernt, dann verändert es sich abhängig von gemachten Erfahrungen, bzw. den Daten, die es gesehen hat. Meist wird dieser Lernprozess dadurch realisiert, dass versucht wird, auf automatischem Wege bestimmte Muster oder Regeln in gegebenen Datenmengen zu finden. Diese Informationen können dann einen „intelligenteren“ Umgang mit weiteren ähnlichen Daten ermöglichen.

In dieser Diplomarbeit werden zwei Methoden der Klassifikation miteinander verglichen und untersucht. Klassifikation ist der Vorgang, Daten an Hand vorher observierten Charakteristiken nach bestimmten Klassen zu kategorisieren.

2.2 Klassifikation

Die Aufgabenstellung der Klassifikation ist es Daten einer Menge von gegebenen Klassen zuzuordnen. Vorher wird dafür ein entsprechendes Modell einer solchen Zuordnung erlernt; dieses Modell wird häufig auch als Hypothese oder Konzept bezeichnet. In dieser Arbeit wird der Begriff *Klassifizierer* zusammenfassend sowohl für das Lernverfahren, welches das Zuordnungsmodell erlernt, als auch für das Verfahren, welches dieses Modell für die Klassifikation von Daten benutzt, verwendet.

In dieser Arbeit werden weiter die Begriffe *Datensatz* und *Instanz* verwendet. Ein Datensatz besteht aus mehreren Instanzen und einer Menge von Klassen, welchen die Instanzen zugeordnet sind. Beispielsweise können eine Menge an bestimmten Nachrichtentexten und eine Menge an Themen zusammen ein Datensatz sein, wobei die einzelnen Texte die Instanzen sind, welche nach ihrem Inhalt einem Thema als Klasse zugeordnet sind. Formal lässt sich die Aufgabenstellung der Klassifikation wie folgt beschreiben:

Definition (Klassifikation): *Sei K eine Menge von Klassen eines Datensatzes und sei I eine Menge von Instanzen desselben Datensatzes. Die Aufgabenstellung der* Klassifikation *ist es eine Zuordnung der Form*

$$f : I \rightarrow 2^K$$

zu erlernen, so dass diese bestimmte Kriterien erfüllt. Die Menge 2^K ist die Menge der Teilmengen von K. Bei vielen Klassifikationsaufgaben wird jeder Instanz jedoch nicht eine Menge von Klassen zugeordnet, sondern nur genau eine Klasse, so dass sich die gesuchte Zuordnung oft zu

$$f : I \rightarrow K$$

vereinfachen lässt.

Die erwähnten, aber an dieser Stelle nicht weiter aufgelisteten, Kriterien sind allgemein, dass der Klassifizierer neue Instanzen möglichst den korrekten Klassen zuordnen soll. Wie diese Kriterien formalisiert sind, hängt von der Art des verwendeten Klassifizierers und dem konkreten Klassifikationsproblem ab.

Wie später ausgeführt wird, kann man die Problemstellung der Klassifikation unter anderem nach der Gesamtzahl von Klassen und danach, wie viele dieser Klassen einer Instanz gleichzeitig zugeordnet werden können, unterscheiden.

Es existieren mehrere Ansätze wie ein Modell für die Zuordnung von Instanzen zu Klassen erlernt werden kann. Man kann diese danach unterschieden, welche Art von Trainingsdaten für den Lernprozess benutzt wird. Wir beschäftigen uns in dieser Arbeit nur mit dem so genannten *überwachten Lernen*[1]. Überwachtes Lernen bedeutet, dass die Zuordnung mit Hilfe von gegebenen Trainingsinstanzen erlernt wird, bei denen die Zugehörigkeit zu den Klassen bekannt ist. Lernprozesse bei denen die Klassenzugehörigkeit der Trainingsdaten nicht benutzt wird oder nicht vorhanden ist, werden *unüberwachtes Lernen* [2] genannt.

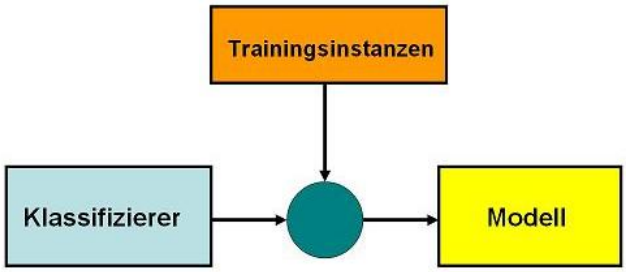

Abbildung 2.1: An Hand der Trainingsinstanzen lernt der Klassifizierer ein Zuordnungsmodell

Der Lernalgorithmus versucht aus den Trainingsdaten Regeln oder Gesetzmäßigkeiten über die Zugehörigkeit zu einzelnen Klassen abzuleiten. Ein wichtiges Ziel dabei ist es, die gegebenen Informationen nicht auswendig zu lernen, sondern stattdessen in einem Konzept so zu verallgemeinern, dass der Klassifizierer neben den Trainingsinstanzen auch bisher nicht gesehene Instanzen korrekt kategorisieren kann. Das Problem, dass ein Klassifizierer zwar die Kategorisierung der Trainingsdaten sehr gut erlernt hat, aber auf Grund mangelnder Generalisierung der Informationen neue Instanzen nur schlecht klassifizieren kann, wird als *Overfitting* (Überanpassung) bezeichnet. Beim Overfitting versagt der Klassifizierer darin, das abstrakte Konzept der Klassenzugehörigkeiten zu erkennen, und lernt stattdessen eine meist deutlich komplexere Unterscheidung, welche sich nur erfolgreich auf die gegebenen Trainingsbeispiele anwenden lässt. Betrachten wir ein sehr simples Beispiel: die Klassen seien A und B. Die Instanzen bestehen nur aus einer natürlichen Zahl; die Trainingsbeispiele für Klasse A seien $\{1, 3, 7, 11\}$, die Beispiele für Klasse B seien $\{2, 10, 20, 22\}$. Ein sehr gut generalisiertes Unterscheidungskonzept wäre es, jeder Instanz die Klasse A zuzuordnen, falls die Instanz aus einer ungerade Zahl besteht und ihr anderseits die Klasse B zuzuordnen. Mit diesem Konzept können alle zukünftigen Instanzen korrekt klassifiziert werden. Im Falle von Overfitting könnte beispielsweise das offensichtlich schlechtere und dazu noch komplexere Modell erlernt werden: eine Instanz sei der Klasse A zugeordnet genau dann wenn deren Zahl in der Menge $\{1, 3, 7, 11\}$ enthalten ist und andererseits sei die

[1] engl. supervised learning
[2] engl. unsupervised learning

Instanz der Klasse B zugeordnet. Dieses Modell arbeitet perfekt auf den Trainingsdaten, würde aber auf neuen Instanzen sehr schlecht arbeiten. Wir halten an dieser Stelle fest, dass es für die Güte eines Klassifizierers unmittelbar entscheidend ist, wie gut dieser an Hand der Trainingsdaten generalisieren kann und ein möglichst abstraktes und allgemeingültiges Konzept erlernt.

Um die Güte eines erlernten Klassifikationsmodells zu bewerten, wird dieses evaluiert. Dazu werden Testinstanzen benutzt, bei denen die korrekte Klassenzugehörigkeit bekannt ist. Wird jede Instanz mit genau einer Klasse assoziiert, dann lässt sich ein Evaluationsmaß recht einfach festlegen. Sagt der Klassifizierer die korrekte Klasse einer Testinstanz vorher, gilt dies als Treffer. Ist seine Vorhersage aber eine andere als die korrekte Klasse, dann gilt dies als Fehler. Der Anteil der gemachten Fehler über alle Testinstanzen spiegelt so die Qualität des Klassifizierers wider. Falls eine Instanz mehreren Klassen zugeordnet sein kann, dann sind meist etwas komplexere Bewertungsmethoden notwendig; auf diese Fälle wird später in dieser Arbeit noch eingegangen. Es ist weiter festzustellen, dass in jedem Fall die zum Lernen des Klassifizierers verwendeten Trainingsinstanzen nicht ebenfalls als Testinstanzen verwendet werden sollten, weil das Ergebnis der Evaluation dadurch zu optimistisch ausfallen würde. Damit wäre es kein sinnvolles Maß mehr dafür, wie das erlernte Modell auf unbekannten Daten funktioniert.

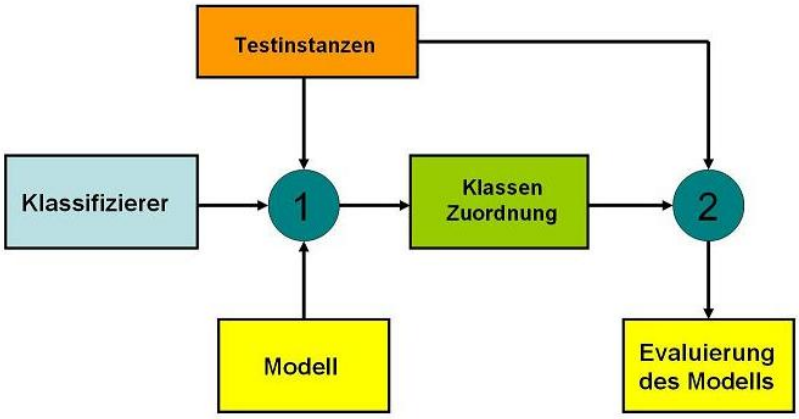

Abbildung 2.2: Evaluation des Klassifizierers
(1) Die Testinstanzen werden an Hand des Modells klassifiziert (2) Die Ergebnisse werden mit den wirklichen Klassen verglichen

2.3 Support Vector Maschinen

In diesem Abschnitt wird das Konzept der *Support Vector Maschinen (SVM)* als eines von vielen im Bereich des Maschinellen Lernens verwendeten Lernverfahren vorgestellt. Es soll nicht im Detail auf die mathematische Umsetzung eingegangen werden, sondern mehr auf die dahinterliegende Idee.

Die Instanzen bestehen aus einer Menge von Attributen, welche jeweils Eigenschaften der Instanzen beschreiben. Betrachten wir als Beispiel Nachrichtentexte als Instanzen: die Attribute eines Textes könnten jeweils ein bestimmtes Wort repräsentieren und der Wert des einzelnen Attributes ist die Anzahl, wie oft dieses Wort in dem Text vorkam. Eine Instanz ist also ein Vektor bestehend aus den Werten seiner n Attribute und kann so in einem n-dimensionalen Raum als Punkt dargestellt werden.

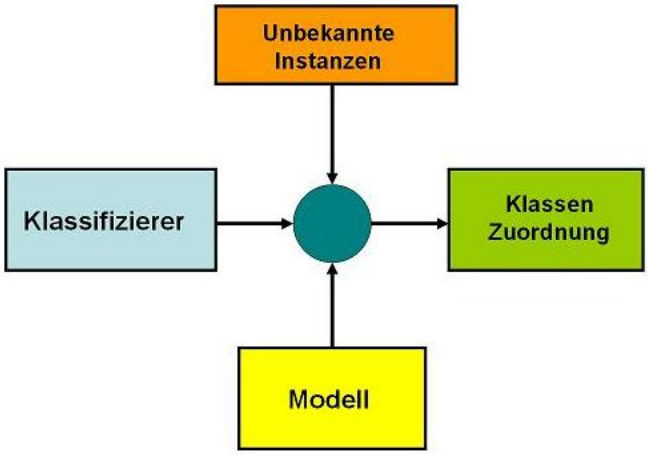

Abbildung 2.3: Das erlernte Modell wird verwendet um neue Instanzen zu klassifizieren

Support Vector Maschinen können nur die Unterscheidung zwischen zwei Klassen erlernen; eine Unterscheidung mehrerer Klassen voneinander ist nicht direkt möglich, sondern muss über ein Aufteilen des Problems gelöst werden.[3]

Die Zielsetzung der Support Vector Maschinen ist es, eine Trennlinie zwischen den Instanzen zweier Klassen zu finden. Falls die Instanzen zwei Attribute haben ist die Trennlinie eine Gerade im zweidimensionalen Raum; in höherdimensionalen Räumen sprechen wir von einer Hyperebene. Wenn die Mengen der Instanzen linear trennbar sind, ergeben sich normalerweise unendlich viele Möglichkeiten die beiden Mengen mit einer Gerade zu trennen. Gesucht wird bei Support Vector Maschinen, im Unterschied zu vielen anderen Lernverfahren, jedoch immer die Gerade, welche den maximalen *Separartion Margin* besitzt. Der Separation Margin ist der Abstand zwischen der Trenngeraden und der ihr nächsten Instanz im Raum. In Abbildung 2.3 sind mögliche Trenngeraden eingezeichnet und ihr Margin jeweils rot markiert. Die grün gezeichnete Trennlinie hat den maximalen Margin und würde daher von einer Support Vector Maschine als Unterscheidungsmodell erlernt werden. Der Sinn, von allen möglichen Trenngeraden gerade die mit dem größtmöglichen Abstand zu den Instanzen zu wählen, ist es, die Wahrscheinlichkeit eine gut generalisierte Unterscheidung zu lernen zu erhöhen, so dass dann auch Instanzen, die räumlich gesehen etwas abseits von der ursprünglichen Trainingsmenge liegen, noch korrekt zu klassifizieren.

Wenn sich in dem ursprünglichen Raum der Attribute keine Trennlinie finden lässt, dann werden die Instanzen in einen Raum höherer Dimension transformiert, in dem die Instanzen linear getrennt werden können. Das Theorem von Cover [Schölkopf and Smola, 2002] besagt, dass die Wahrscheinlichkeit eine lineare Trennung zwischen zwei Punktmengen zu finden, mit der Dimension des Darstellungsraumes steigt. Die Berechnungen in diesen höherdimensionaler Räumen werden indirekt durch so genannte *Kernel Funktionen* durchgeführt.

Eine wichtige Erweiterung der originalen Idee der SVM ist das Konzept des *Soft Margin*. Der Algorithmus kann unter Umständen nicht in der Lage sein eine Trennlinie in den Daten zu finden, weil einige

[3] Nämlich durch Binarisierung des Multi Klassen Problems. Vergleiche dazu 2.4.1

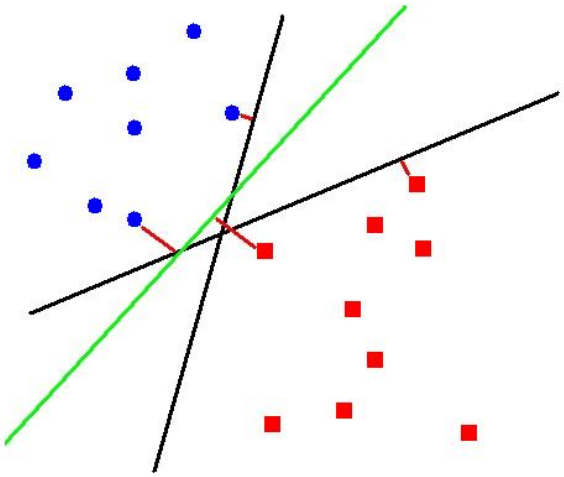

Abbildung 2.4: Verschiedene Trennlinien mit unterschiedlichem Margin

wenige der Instanzen fehlerhaft sind[4] und stets in der Menge der Instanzen der anderen Klasse liegen. Statt, dass die Suche einer Trennlinie an diesen einzelnen Instanzen scheitert, werden solche Ausreißer bei der Soft Margin Berechnung toleriert und es wird eine Unterscheidungslinie erlernt, welche die übrigen Instanzen möglichst gut trennt.

Weiterführende Informationen über Support Vector Maschinen und insbesondere Kernel Funktionen und Soft Margin kann man bei [Shawe-Taylor and Cristianini, 2004] und [Schölkopf and Smola, 2002] finden.

2.4 Fallunterscheidung bei der Klassifikation

Abhängig von den Eigenarten des Datensatzes für den ein Klassifizierer gelernt werden soll, lassen sich verschiedene Typen von Klassifikationsproblemen unterscheiden. Insbesondere wird nach der Gesamtzahl der Klassen eines Datensatzes und danach wie viele Klassen einer Instanz insgesamt zugeordnet werden können unterteilt.

Der einfachste Fall ist die binäre Klassifikation. Hierbei existieren nur genau 2 Klassen, in welche die Instanzen eingeteilt werden können. Jede Instanz kann nur einer dieser Klassen zugeordnet sein. In den meisten Fällen repräsentieren die beiden Klassen das Vorhanden- bzw. Nichtvorhandensein einer Eigenschaft. Beispielsweise ist die Unterscheidung von eMails in Spam oder Nicht-Spam ein binäres Klassifikationsproblem. Seien i und j die beiden Klassen, welchen ein binärer Klassifizierer die Instanzen zuordnen soll. Die Unterscheidung ob eine Instanz nun zur Klasse i oder j gehört, kann abhängig vom verwendeten Lernalgorithmus des Klassifizierers unterschiedlich realisiert werden. Der Klassifizierer kann die Menge der Instanzen I auf 0 oder 1 abbilden, wobei der Wert 0 eine Zuordnung zur Klasse i repräsentiert und 1 entsprechend für die Klasse j steht: $K : I \rightarrow \{0, 1\}$. Diese Art von Klassifizierer bezeichnen wir als *dichotomen* Klassifizierer. Die andere Möglichkeit ist ein Klassifizierer der Form $K : I \rightarrow [0, 1]$, wobei das Ergebnis $K(x)$ mit $x \in I$ als Wahrscheinlichkeit oder auch Konfidenz für die Zuordnung von x zur Klasse i und entsprechend $(1 - K(x))$ für die Zuordnung zur Klasse j

[4] Beispielsweise durch Rauschen

interpretiert werden kann. Diese Variante wird als *probabilistischer* Klassifizierer bezeichnet.

Die Trainingsinstanzen eines binären Klassifizierers werden in dieser Arbeit auch als *positive Beispiele*, bzw. positive Trainingsinstanzen, und *negative Beispiele*, bzw. negative Trainingsinstanzen, bezeichnet. Dabei gelten für einen Klassifizierer $K_{i,j}$, welcher zwischen den Klassen i und j unterscheidet, die Trainingsinstanzen der Klasse i als positive und die der Klasse j als negative Beispiele.

Bei den meisten Klassifikationsproblemen ist die Menge der möglichen Klassen jedoch größer als nur zwei. Zusätzlich ist es bei vielen realen Anwendungen so, dass Instanzen nicht nur einer, sondern gegebenenfalls mehreren Klassen gleichzeitig, zugeordnet werden sollen. Diese beiden Fälle sollen im Folgenden genauer beleuchtet werden.

2.4.1 Multiklassenprobleme

Betrachten wir eine vereinfachte Aufgabenstellung der Texterkennung: basierend auf Bilddaten soll ein Buchstabe erkannt werden. Die Anzahl der Klassen ist hierbei die Anzahl der möglichen Buchstaben. In diesem Fall ist es nicht möglich, dass dieselben Bilddaten gleichzeitig zwei Buchstaben zugeordnet werden. Dieses Beispiel stellt ein Multiklassenproblem dar.

Definition (Multiklassenproblem): *Wir sprechen von einem* Multiklassenproblem, *wenn ein Klassifikationsproblem vorliegt, bei dem die Mächtigkeit der Menge der Klassen größer als 2 ist:* $|K| > 2$.

Ein praktisches Problem bei der Klassifikation von Multiklassen Datensätzen ist, dass die meisten Lernalgorithmen auf die Unterscheidung von nur zwei Klassen ausgelegt sind. In [Fürnkranz, 2001] werden Erklärungen für diese Beschränkung angeführt. So ist beispielsweise das Modell der Support Vector Maschine so entwickelt, dass es eine Trennlinie zwischen zwei Instanzenmengen findet; offensichtlicherweise kann mit einer Trennlinie nur genau ein Paar von Klassen unterschieden werden. Eine Möglichkeit diese Schwierigkeit zu umgehen, stellt die Binarisierung[5] dar.

Definition: (Binarisierung): Binarisierung *ist der Vorgang ein Multiklassenproblem auf mehrere binäre Klassifikationsprobleme zurückzuführen. Dies muss so geschehen, dass aus den Einzellösungen der binären Probleme eine Gesamtvorhersage für das Multiklassenproblem erstellt werden kann. Indem das Problem in mehrere kleinere binäre Probleme heruntergebrochen wird, können diese einzeln mit binären Klassifizierern gelöst werden.*

Definition (Basisklassifizierer): *Die bei der Binarisierung zur Lösung der entstandenen binären Klassifikationsprobleme verwendeten Klassifizierer werden* Basisklassifizierer, *oder auch* Basislerner, *genannt.*

Zwei gängige Ansätze der Binarisierung sind das One-against-all Verfahren und die paarweise Binarisierung, welche auch Round Robin Binarisierung genannt wird. In dieser Arbeit wird der Ansatz der paarweisen Klassifikation, welche die paarweise Binarisierung bedingt, mit einer modifizierten Form derselben, der paarweisen hierarchischen Klassifikation, verglichen. Daher wird die Paarweise Klassifikation unter 2.5 noch genauer vorgestellt und dort auch mit dem One-against-All Verfahren verglichen.

[5] engl. Binarization

2.4.2 Multi Label Probleme

In der Praxis benötigt ein Großteil der Klassifikationsanwendungen eine Einteilung der einzelnen Instanzen zu mehreren Klassen. In diesen Fällen sprechen wir oft von *Labels* statt von Klassen, da einem Objekt bildlich verschiedene Labels auf einmal angeheftet werden. Ein Beispiel für diese Art von Klassifikation ist die Kategorisierung von Texten oder Musik. So kann ein Musikstück sowohl der Klasse $Rock$ als auch gleichzeitig der Klasse $Ballade$ angehören.

Wir nennen diese Zuordnungsprobleme Multi Label Probleme. Die bisher verwendeten Beispiele erlaubten alle nur die Zuordnung einer Instanz zu einer einzigen Klasse und sind damit so genannte *Single Label Probleme*.

Definition (Single Label): *Wir sprechen von einem* Single Label Problem, *wenn jeder Instanz genau einer Klasse zugeordnet wird. Die gesuchte Zuordnungsfunktion $f : I \rightarrow K$ ordnet jeder Instanz ein Element aus der Menge der Klassen zu.*

Definition (Multi Label): *Wir sprechen von einem* Multi Label Problem, *wenn jeder Instanz mehrere Klassen gleichzeitig, mindestens jedoch eine, zugeordnet werden können. Die gesuchte Zuordnungsfunktion $f : I \rightarrow 2^K$ ordnet jeder Instanz eine Teilmenge der Menge aller Klassen zu. Die Labels der einer Instanz $x \in I$ zugeordneten Teilmenge $P_x \subset 2^K$ werden auch als die relevanten Labels, bzw. Klassen, bezeichnet. Entsprechend sind die Elemente der Menge $N_x = 2^K \setminus P_x$ die irrelevanten Labels für die Instanz x.*

Die Tatsache, dass bei Multi Label Problemen nicht nur die jeweils „beste“ oder „wahrscheinlichste“ Klasse gesucht wird, sondern eine Menge der „wahrscheinlichsten“ Klassen, macht die Klassifikation von Multi Label Datensätzen zu einem deutlich komplexeren Problem als die von Single Label Datensätzen. Eine Einführung in diese besondere Problemstellung bietet [Tsoumakas and Katakis, 2007].

In dieser Arbeit wird, wenn nicht explizit anders beschrieben, von Single Label Problemen ausgegangen. Der unter 5.2 verwendete Reuters Datensatz stellt allerdings ein Multi Label Problem dar.

2.4.2.1 Rankingprobleme

Die Lösung eines Multi Label Problems ist eng verbunden mit der Lösung eines Rankingproblems. Bei einem Rankingproblem geht es darum einer Instanz eine Rangfolge aller Klassen zuzuordnen. Das Ziel dabei ist es die Klassen aufsteigend nach ihrer „Relevanz“, bzw. nach der Wahrscheinlichkeit, dass sie der Instanz zugeordnet sind, zu sortieren.

Definition (Rankingproblem): *Sei $K = \{k_1, k_2, \ldots, k_n\}$ die Menge der Klassen und I die Menge der Instanzen, dann nennen wir die Suche nach einer Funktion $f : I \rightarrow K^n$, so dass für $f(x) = \hat{k}$ gilt, dass die Elemente von $\hat{k} \in K^n$ in der Reihenfolge der Relevanz für die jeweilige Instanz $x \in I$ sind, ein* Rankingproblem.

Definition (Top Rank): *Die Klasse, die in einem Ranking R an erster Stelle steht, heißt* Top Rank.

Man kann Klassifikationsprobleme als Spezialfälle des Rankingproblems sehen, bei dem nur die erste, im Falle einer Single Label Klassifikation, bzw. die ersten n Klassen, bei einer Multi Label Klassifikation, des Ranking zurückgegeben werden.

Wenn ein Ranking dazu verwendet wird um ein Multi Label Problem zu lösen[6], muss entschieden

[6]Eine Möglichkeit ein Multi Label Problem ohne ein Ranking zu lösen, wäre die Binary Relevance Methode. Vergleiche

werden wo im Ranking die Trennlinie zwischen den relevanten Klassen, welche vorhergesagt werden sollen, und den irrelevanten gezogen wird. Diese Grenze bezeichnen wir als *Relevanzgrenze*. Meist ist es nicht bekannt wie vielen Klassen eine Instanz angehört, da oftmals die Anzahl der relevanten Klassen von Instanz zu Instanz variiert. Es ist möglich an Hand der Trainingsdaten zu ermitteln, wie viele Klassen einer Instanz im Schnitt zugeordnet sind. Diese Durchschnittsanzahl an Klassen wird dann bei jeder Klassifikation einer Instanz verwendet, indem ihr diese Anzahl der ersten Klassen des Rankings zugeordnet wird. Da, wie schon erwähnt, in der Regel nicht für jede Instanz dieselbe Anzahl an Klassen relevant ist, nimmt man mit dieser Methode bewusst Fehler in Kauf. Eine bessere und komplexere Methode stellt das *Calibrated Ranking* dar. Hierbei wird eine zusätzliche Klasse hinzugefügt, welche dann die Trennlinie im Ranking bestimmt: alle Klassen, die im Ranking über dieser Klasse liegen, gelten als relevant. Siehe [Brinker *et al.*, 2006] für eine ausführliche Erklärung dieser Methode und [Hüllermeier and Fürnkranz, 2004] für einen Vergleich des Calibrated Ranking mit dem oben erwähnten Vorgehen der Durchschnittsanzahl.

2.4.2.2 Multi Label Evaluation

Die Evaluation von Multi Label Klassifizierern stellt sich als deutlich komplexer dar, als die von Single Label Klassifizierern. Dies liegt darin begründet, dass es neben dem Fall einer perfekten Klassifikation, das heißt alle relevanten Klassen werden vorhergesagt und alle irrelevanten werden nicht vorgesagt, viele Möglichkeiten gibt, bei denen der Klassifizierer teilweise korrekte Ergebnisse produziert. Um diese Fälle zu unterscheiden, werden entsprechend differenzierende Bewertungsmaße benötigt.

Es gibt kein anerkanntes Standardmaß für die Evaluation von Multi Label Klassifizierern, daher werden hier nur einige von vielen möglichen Bewertungsfunktionen vorgestellt. Die folgenden Maße werden bei der Bewertung der Klassifizierer auf dem Reuters Datensatz unter 5.2 benutzt.

Für die folgenden Beschreibungen werden die Begriffe *true positive (TP)* für eine relevante Klasse, welche vorhergesagt wurde, *true negative (TN)* für eine irrelevante Klasse, die nicht vorhergesagt wurde, *false positive (FP)* für eine irrelevante Klasse, welche vorhergesagt wurde, und *false negative (FN)* für eine relevante Klasse, welche nicht vorhergesagt wurde, verwendet.

Precision:
Precision ist ein Maß für die Genauigkeit der Vorhersage des Klassifizierers. Es ist der Quotient aus der Anzahl der korrekt vorhergesagten Klassen und allen vorhergesagten Klassen. Der Wert bewegt sich im Intervall $[0, 1]$, wobei das Ergebnis umso besser ist, desto näher es an 1 liegt. Man kann den Precision Wert auch als Wahrscheinlichkeit, dass eine vorhergesagte Klasse tatsächlich relevant ist, interpretieren.

$$Prec := \frac{TP}{(TP + FP)}$$

Recall:
Recall ist ein Maß für die Vollständigkeit der Vorhersage des Klassifizierers. Es ist der Quotient aus der Anzahl der korrekt vorhergesagten Klassen und allen relevanten Klassen. Der Wert bewegt sich im Intervall $[0, 1]$, wobei das Ergebnis umso besser ist, desto näher es an 1 liegt. Man kann den Recall Wert auch als Wahrscheinlichkeit, dass eine relevante Klasse vorhergesagt wurde, interpretieren.

$$Rec := \frac{TP}{(TP + FN)}$$

dazu [Brinker *et al.*, 2006].

Precision und Recall sind Maße aus dem Bereich des Information Retrieval und dienen zur Beschreibung von Suchergebnissen. Die beiden Maße werden zusammen betrachtet, da ein gutes Ergebnis bei einem der Werte für sich alleine keine verlässliche Aussage über die Güte des Klassifizierers zulässt. Beispielsweise hat der Klassifizierer, welcher immer alle Klasse vorhersagt einen perfekten Recall von 1, jedoch eine schlechte Precision. Generell sinkt der Wert eines der beiden Maße, wenn der jeweils andere steigt. Erstrebenswert ist folglich ein Gleichgewicht der beiden Maße, bei dem beide einen zufriedenstellenden Wert haben.

F1-Score:
F1-Score, auch F1-Measure genannt, ist ein Maß, welches Precision und Recall vereinigt. Es kann als gewichteter Durchschnitt von Precision und Recall interpretiert werden. Der Wert bewegt sich im Intervall $[0, 1]$, wobei das Ergebnis umso besser ist, desto näher es an 1 liegt.

$$F1 := \frac{2}{(2 + \frac{FP+FN}{(TP)})} = \frac{Prec \cdot Rec \cdot 2}{(Prec + Rec)}$$

Hamming Loss:
Hamming Loss ist eine Verlustfunktion. Sie misst den gemachten Fehler des Klassifizierers, indem sie den Anteil der falsch zugeordneten Klassen unter allen Klassen misst. Falsch zugeordnete Klassen können entweder „false negatives“ oder „false positives“ sein. Der Wert bewegt sich im Intervall $[0, 1]$ und das Klassifikationsergebnis ist umso besser, desto näher der Wert an 0 liegt.

$$HamLoss := \frac{FN + FP}{TP + TN + FN + FP}$$

Multi Label Probleme werden oft mit Hilfe eines Rankings der Klassen gelöst. Es sollen nun einige Evaluationsfunktionen, welche Rankings bewerten, vorgestellt werden. Die oben bereits vorgestellten Maße können ebenfalls auf Rankings angewendet werden, indem alle Klassen über der Relevanzgrenze als vorhergesagt gelten und alle darunter als nicht vorgesagt. Bei den folgenden Bewertungsfunktionen ist die Festlegung einer Relevanzgrenze im Ranking nicht notwendig, da diese das Ranking als ganzes bewerten.

Sei P_x die Menge aller für x relevanten Klassen, $K = \{k_1, k_2, \ldots, k_n\}$ die Menge aller Klassen und R_x das Ranking für die Instanz x. Die Menge der für x irrelevanten Klassen ergibt sich folglich als $N_x = K \setminus P_x$. Die Funktion $\pi : K \to \{1, 2, \ldots, n\}$ gibt für jede Klasse deren Position im Ranking R_x wieder. Es gilt, dass k_i im Ranking höher eingestuft ist als k_j genau dann wenn $\pi(k_i) < \pi(k_j)$. Für den Top Rank gilt $\pi(TopRank) = 1$.

Zunächst sollte geklärt werden wie das bestmögliche Ranking für eine Instanz x aussehen sollte. Wir sprechen von einem *perfekten Ranking* genau dann wenn alle relevanten Klassen im Ranking höher liegen als alle irrelevanten Klassen: R_x ist perfekt $\iff \forall r \in P_x . \forall i \in N_x . \pi(r) < \pi(i)$. Die Reihenfolge der relevanten, bzw. irrelevanten, Klassen untereinander spielt in diesem Fall keine Rolle.

OneError:
Das Fehlermaß OneError ist sehr simple und beschreibt lediglich ob der Top Rank des Rankings eine relevante Klasse ist. Ist dies der Fall gibt OneError 0 zurück, ansonsten 1.

$$OneError(R_x) = 0 \iff \exists r \in P_x . \pi(r) = 1$$

$$OneError(R_x) = 1 \iff \forall r \in P_x . \pi(r) \neq 1$$

Margin Loss:
Das Fehlermaß Margin Loss ist die Differenz zwischen der Position der am schlechtesten platzierten

relevanten Klasse und der Position der am besten platzierten irrelevanten Klasse im Ranking. Bei einem perfekten Ranking ist der Wert 0.

$$MarginLoss(R_x) := \max(0, \max_i\{i|i = \pi(r), r \in P_x\} - \min_j\{j|j = \pi(i), i \in N_x\})$$

Rank Loss:
Das Fehlermaß Rank Loss betrachtet alle möglichen Klassenpaare, bei denen eine Klasse relevant und eine Klasse irrelevant ist. Das Maß entspricht dem Anteil der fehlerhaften Paare von allen Paaren. Ein Klassenpaar gilt dann als fehlerhaft, wenn die irrelevante Klasse im Ranking höher liegt als die relevante Klasse.

$$RankLoss(R_x) := \frac{|\{(r,i)|\pi(i) < \pi(r), i \in N_x, r \in P_x\}|}{|P_x| \cdot |N_x|}$$

Average Precision:
Bei dem Maß Average Precision wird für jede relevante Klasse im Ranking ein Precision Wert berechnet, als ob die Relevanzgrenze so läge, dass die jeweilige relevante Klasse und alle im Ranking vor ihr liegenden Klassen über dieser Grenze liegen. Es ergeben sich so viele Precision Werte wie es relevante Klassen gibt; Average Precision stellt den Durchschnitt dieser Werte dar. Der Wert bewegt sich im Intervall $[0, 1]$, wobei der Wert 1 einem perfektem Ranking entspricht.

$$AvgPrec(R_x) := \frac{1}{|P_x|} \sum_{r \in P_x} \frac{|\{\hat{r}|\pi(\hat{r}) \leq \pi(r), \hat{r} \in P_x\}|}{\pi(r)}$$

2.5 Paarweise Klassifikation

In diesem Abschnitt soll das Verfahren der paarweisen Klassifikation vorgestellt und mit einer weiteren Binarisierungsmethode, dem One-against-All, verglichen werden. Die folgenden Erläuterungen basieren im Wesentlichen auf der folgenden Arbeit von Fürnkranz: [Fürnkranz, 2001].

2.5.1 Definition

Die paarweise Klassifikation ist ein Verfahren um Multiklassenprobleme zu lösen, in dem diese in mehrere binäre Probleme aufgeteilt werden. Diese 2-Klassen Probleme werden dann unabhängig voneinander von jeweils einem Basisklassifizierer gelöst und die Einzelergebnisse zu einer Gesamtlösung des Multiklassenproblems zusammengefasst. Jeder Basisklassifizierer soll dafür die Unterscheidung zwischen genau zwei Klassen lernen und treffen. Für jedes mögliche Paar an Klassen wird demnach ein Klassifizierer benötigt.

Die entsprechende Aufteilung des Problems geschieht durch die paarweise Binarisierung, auch Round Robin Binarisierung genannt:

Definition (Paarweise Binarisierung): *Die* paarweise Binarisierung *wandelt ein k-Klassen Problem in $\frac{k(k-1)}{2}$ binäre Probleme um, eines für jedes Paar an Klassen (i, j) mit $i = 1 \ldots (k-1)$ und $j = (1+i) \ldots k$. Der Klassifizierer für das Klassenpaar (i, j) verwendet die Trainingsinstanzen der Klasse i als positive Beispiele und die Trainingsinstanzen der Klasse j als negative Beispiele; alle anderen Instanzen des Problems werden bei dem Lernprozess dieses Basisklassifizierers ignoriert.*
Quelle: [Fürnkranz, 2001]

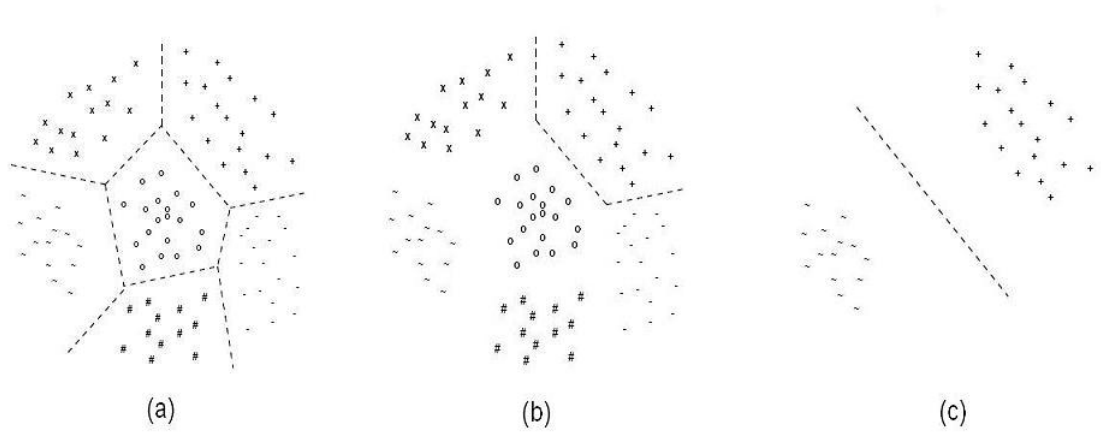

Abbildung 2.5: Binarisierungsmethoden
(a) Multiklassenproblem: Es soll zwischen 6 Klassen unterschieden werden (b) One-against-All: Ein Klassifizierer unterscheidet zwischen einer Klasse und allen Anderen (c) Paarweise Klassifikation: Ein Klassifizierer unterscheidet zwischen je zwei Klassen Quelle: [Fürnkranz, 2001]

Der verwendete Basisklassifizierer kann bei diesem Verfahren beliebig gewählt werden. Lernt ein Klassifizierer, trainiert für das Klassenpaar (i, j), dasselbe Unterscheidungsmodell, wie wenn er für das Klassenpaar (j, i) trainiert wäre, nennt man diesen Klassifizierer *klassensymmetrisch*. Nicht alle Klassifizierer haben diese Eigenschaft; beispielsweise versuchen viele Regellerner beim Lernprozess Regeln zu finden, welche möglichst alle positiven Trainingsbeispiele abdecken sollen. Als Resultat ist das Lernergebnis unterschiedlich, abhängig davon welche Instanzenmenge als die positiven Beispiele gesehen wurde. Diese Klassifizierer nennen wir *klassenasymmetrisch*.

Für die Verwendung von klassenasymmetrischen Basisklassifizierern bietet es sich an die paarweise Binarisierung entsprechend anzupassen:

Definition (Double Round Robin): *Die* Double Round Robin Binarisierung *wandelt ein k-Klassen Problem in $k(k-1)$ binäre Probleme um, eines für jedes Paar an Klassen (i, j) mit $i, j = 1 \ldots k$ und $j \neq i$. Der Klassifizierer für das Klassenpaar (i, j) verwendet die Trainingsinstanzen der Klasse i als positive Beispiele und die Trainingsinstanzen der Klasse j als negative Beispiele; alle anderen Instanzen des Problems werden bei dem Lernprozess dieses Basisklassifizierers ignoriert.* Quelle: [Fürnkranz, 2001]

Da nun für jedes Klassenpaar zwei Klassifizierer gelernt werden, nämlich jeweils einer sowohl für (i, j) als auch einer für (j, i), wird die Klassenasymmetrie des Klassifizierers ausgeglichen. Für die Experimente dieser Arbeit wurden jedoch nur klassensymmetrische Klassifizierer verwendet und entsprechend auch nur die „einfache“ paarweise Binarisierung.

2.5.2 Dekodierung

Das Zusammensetzen der einzelnen Ergebnisse, der von den Basisklassifizierern gelösten binären Probleme, zu einer Gesamtvorhersage als Lösung des Multiklassenproblems wird als Dekodieren bezeichnet. Es existieren einige relativ komplexe Methoden, welche die Effizienz der Klassifizierung steigern

können, indem nicht alle binären Einzelprobleme gelöst werden müssen. Die simpelste, aber sehr oft verwendete, Lösung ist das Voting.

Definition (Voting): *Beim* Voting, *auch als Max Wins bekannt, stimmt jeder Basisklassifizierer für eine Klasse. Die Klasse mit den meisten* Stimmen*, das heißt mit der größten Summe, wird dann vorhergesagt. Sei $K_{i,j} : I \to [0, 1]$ der binäre Basisklassifizierer, welcher zwischen den Klassen i und j unterscheidet, $x \in I$ die zu klassifizierende Instanz und K die Menge der Klassen, dann lässt sich der* Voting-Prozess *wie folgt formulieren: Die vorhergesagte Klasse ist*

$$k := \arg\max_{j \in K} \sum_{i \in K} K_{i,j}(x)$$

Es sei festgestellt, dass bei klassensymmetrischen Basisklassifizierern in der obigen Formel $K_{x,y} = 1 - K_{y,x}$ gilt.

Bei einem Unentschieden, wird die Gewinnerklasse in der Regel entweder nach Zufall entschieden oder es wird die Klasse gewählt, welche in den Trainingsinstanzen am häufigsten vorkommt, weil man dies als Indikator für eine höhere Wahrscheinlichkeit dieser Klasse sehen kann.

Bei der paarweisen Klassifikation kommt es zwingend zu dem Fall, dass ein Basisklassifizierer $K_{i,j}$ seine Stimme im Voting für eine irrelevante Klasse abgeben muss, nämlich dann wenn weder i noch j für die betrachtete Instanz relevant sind. Dieser „Fehler“ wird jedoch beim Gesamtergebnis des Votings wieder ausgeglichen, wenn die Basisklassifizierer, welche für die relevanten Klasse stimmen können, korrekt entscheiden. Betrachten wir dazu den Fall von dichotomen Basisklassifizierern. Sei die Instanz $x \in I$ der Klasse $i \in K$ zugeordnet. Die Mächtigkeit von K sei n. Es existieren also genau $\frac{(n-1)n}{2}$ Basisklassifizierer, welche beim Voting eine Stimme abgeben können. Weiter gibt es für jede Klasse z genau $(n-1)$ Klassifizierer der Form $K_{z,y}$ mit $y = (1, \ldots, n)$ und $y \neq z$. Angenommen alle Klassifizierer der Form $K_{i,y}$ stimmen bei der Instanz x korrekterweise für die Klasse i, dann hat die Klasse i insgesamt $n-1$ Punkte im Voting. Die Klasse i hat damit das Voting bereits gewonnen, da jede andere Klasse nun nur maximal $n-2$ Stimmen erhalten kann. Für jede andere Klasse gibt es ebenfalls $n-1$ Klassifizierer, welche für diese Klasse stimmen könnten, da aber der Klasssifizier, welcher diese Klasse von der Klasse i unterscheidet, bereits für i und damit gegen die andere Klasse gestimmt hat, sind nur noch $n-2$ mögliche Stimmen übrig.
Ähnlich verhält es sich bei der Verwendung von probabilistischen Basisklassifizierern, bei denen wir, beim obigen Beispiel bleibend, für eine Entscheidung zwischen i und einer weiteren Klasse einen deutlich höheren Wert für i erwarten[7], während man bei Entscheidung zwischen zwei irrelevanten Klassen zumeist, sofern keine der Klassen eine hohe Ähnlichkeit zu i aufweist, ein Ergebnis mit geringer Konfidenz[8] annimmt.

Ein weitere Methode der Dekodierung ist die Variante Vote-against, welche von Cutzu in [Cutzu, 2003] vorgestellt wird. Die zugrunde liegende Idee ist, dass der Basisklassifizierer mit seinem Ergebnis nicht für die vorhergesagte Klasse stimmt, sondern gegen die nicht vorhergesagt Klasse. Dieses Verfahren produziert bei einer kompletten Auswertung aller Klassifizierer zwar dasselbe Ergebnis, falls jedoch nicht alle Klassifizierer zur Verfügung stehen[9] liefert es verlässlichere Ergebnisse.

[7] Beispielsweise $K_{i,y}(x) = 0.9$

[8] Beispielsweise $K_{y,z}(x) = 0.6$

[9] Zum Beispiel wenn für einige Klassifizierer keine passenden Trainingsinstanzen vorhanden sind und diese deshalb nicht verwendet werden können

2.5.3 Multi Label Probleme

Mit der paarweisen Klassifikation ist es auch möglich Multi Label Probleme zu lösen. Nehmen wir dazu das einfache Voting als Dekodierungsmethode an. Statt beim Voting nun nur die Klasse mit den meisten Stimmen zu identifizieren, erstellt man geordnet nach Anzahl der Stimmen ein Ranking der Klassen. Umso höher eine Klasse in diesem Ranking platziert ist, desto höher ist die Wahrscheinlichkeit, dass sie für die jeweilige Instanz relevant ist. Wie bereits unter 2.4.2.1 genauer beschrieben, werden nun einfach alle Klassen, welche über einer festgelegten Grenze im Ranking liegen, vorhergesagt.

Bei Multi Label Problemen ist bei der paarweisen Binarisierung zu beachten, dass Trainingsinstanzen, denen eine Menge X an Klassen zugeordnet ist, nicht zum Training dieser Basisklassifizierer verwendet werden kann, welche zwischen zwei Klassen $i \in X$ und $j \in X$ unterscheiden sollen. Eine solche Trainingsinstanz kann nicht dazu dienen ein Modell für diese Unterscheidung zu lernen, da sie offensichtlich beide Klassen repräsentiert. In solchen Fällen werden diese Instanzen für den jeweiligen Lernprozess ignoriert. Wenn in den Versuchen die paarweise Klassifizierung auf Multi Label Datensätzen angewendet wird, dann werden also für einen Klassifizierer $K_{i,j}$ als positive Trainingsinstanzen alle Instanzen gewählt, welche der Klasse i, aber nicht der Klasse j zugehören; entsprechend gelten alle Instanzen mit Klasse j aber nicht Klasse i als negative Beispiele.

Wenn auf hierarchischen Daten mit Superklassen[10] gearbeitet wird, dann ergibt es sich, dass alle binären Klassifizierer der Form $K_{i,i'}$ mit i als Superklasse von i' nicht erlernt werden können, weil es für diesen Klassifizierer keine passenden Trainingsinstanzen für die Klasse i' gibt. Dies liegt daran, dass jede Instanz, welcher die Klasse i' zugeordnet ist auf Grund der Superklasseneigenschaft auch die Klasse i zugeordnet ist, so dass diese Instanz sich nicht für die Unterscheidung der beiden Klassen eignet. Unter 3.3.2 wird festgehalten, dass dies ebenso bei der paarweisen hierarchischen Klassifikation gilt.

2.5.4 Eigenschaften und Vergleich mit One-against-All

Ein weiteres Binarisierungsverfahren ist die One-against-All Binarisierung, bei der jeweils ein Klassifizierer für die Unterscheidung zwischen einer Klasse und allen Anderen erlernt wird. Praktisch soll jeder dieser Klassifizierer vorhersagen, ob eine Instanz einer bestimmten Klasse zugeordnet ist oder nicht. Vergleiche dieser Methode mit der paarweisen Klassifikation finden sich neben [Fürnkranz, 2001] auch bei [Platt *et al.*, 1999], [Yao *et al.*, 2001] und [Hsu and Lin, 2002].

Definition (One-against-All): *Die* One-against-All Binarisierung *wandelt ein k-Klassenproblem in k binäre Probleme um. Der Klassifizierer verwendet zum Lernen jeweils die Instanzen der Klasse i als positive Beispiele und die Instanzen der Klassen j, mit $j = 1 \ldots k, j \neq i$, als negative Beispiele.* Quelle: [Fürnkranz, 2001]

Obgleich bei der paarweisen Binarisierung die zu erlernenden Klassifikationsmodelle quadratisch zur Anzahl der Klassen sind, wird in [Fürnkranz, 2001] bewiesen, dass die eigentliche Komplexität des Lernprozesses linear ist. Dies liegt daran, dass für den einzelnen Basisklassifizierer nur ein Teil der vorhandenen Trainingsinstanzen verwendet wird, nämlich genau diese Instanzen, deren Klassen der Klassifizierer unterscheiden soll. Da die Zeit des Lernvorganges von der Menge der Trainingsinstanzen abhängt, geht das Lernen der paarweisen Klassifizierer schneller als beispielsweise das der Klassifizierer der One-against-All Binarisierung, welche für jeden Basisklassifizierer stets alle Trainingsinstanzen

[10] Vergleiche dazu die Definition von Superklassen unter 2.6.1

benutzt. Dieser Geschwindigkeitsvorteil fällt umso mehr ins Gewicht, desto komplexer, das heißt langsamer mit steigender Anzahl der Instanzen, das verwendete Lernverfahren ist. Ein bleibender Nachteil der paarweisen Klassifikation im Vergleich zum One-against-All Verfahren ist die Tatsache, dass bei der Klassifizierung einer Instanz deutlich mehr Basisklassifizierer ausgeführt werden müssen, nämlich eben $n(n-1)/2$ anstatt nur n.[11]

Weiter wird in [Fürnkranz, 2001] aufgezeigt, dass die paarweise Klassifikation bei Versuchen mit mehreren Datensätzen in der Genauigkeit der Ergebnisse nie schlechter als One-against-All, und oft sogar signifikant besser, abgeschnitten hat. Bei der paarweisen Binarisierung ergeben sich durch die Beschränkung auf die Trainingsinstanzen nur zweier Klassen für jedes binäre Problem weitere allgemeine Vorteile: Die einzelnen binären Probleme benötigen offensichtlich weniger Speicher als das Gesamtproblem, so dass die binären Probleme jeweils komplett in den Speicher passen können, wenn das ganze Multiklassenproblem selbst dafür zu umfangreich wäre. Da die binären Unterscheidungsprobleme voneinander unabhängig sind, können die Unterscheidungen parallel gelernt und die Klassifizierer bei der Klassifikation auch parallel angewendet werden. Bei der Verwendung von Basisklassifizieren, welche bei stark unbalancierten Trainingsmengen[12] dazu tendieren die stärker vertretene Klasse zu bevorzugen, ist die paarweise Binarisierung dem One-vs-All vorzuziehen, da sich beim paarweisen Ansatz mit den Instanzen zweier Klassen eher zahlenmäßig ähnliche Mengen gegenüberstehen, als bei One-against-All, wo die Instanzen nur einer Klasse allen restlichen Instanzen gegenüberstehen.

Einer der Grundideen der paarweisen Binarisierung ist die Einfachheit der zu lösenden Probleme. Durch die Beschränkung der verwendeten Trainingsinstanzen auf nämlich nur die Instanzen dieser Klassen, zwischen denen der jeweiligen Basisklassifizierer unterscheiden soll, kann allgemein eine einfachere Lösung erlernt werden, als zum Beispiel bei One-against-All. Die bei der paarweisen Klassifikation betrachteten Instanzen können in der Regel klarer voneinander unterschieden werden, weil die Unterschiede zwischen nur zwei Klassen stärker hervortreten, als wenn die negativen Beispiele aus den Instanzen mehrerer Klassen bestehen. Es wird daher generell angenommen, dass einfachere Probleme zu einer besseren Lösung führen, was sich mit den Versuchsergebnissen in [Fürnkranz, 2001] und [Hsu and Lin, 2002] deckt.

2.6 Hierarchien

Wenn von hierarchischen Daten die Rede ist, dann bedeutet dies, dass auf den Klassen eines Datensatzes eine Hierarchie definiert ist. Eine Hierarchie ist nur mit mehreren Klassen sinnvoll und daher ist die Klassifikation von solchen Daten ein Multiklassenproblem. Hierarchische Daten finden sich in vielen Anwendungsgebieten des Maschinellen Lernens wieder. So werden Dokumente bei der Aufgabe der Textklassifikation oft entsprechend einer Hierarchie klassifiziert, so beispielsweise bei Suchmaschinen im Internet, wo es um die inhaltliche Kategorisierung von Internetseiten geht. Bei der maschinellen Vorhersage von Genfunktionen wird ebenfalls mit Hierarchien gearbeitet, welche die verschiedenen Funktionen eines Gens aufschlüsseln und schrittweise spezifizieren.[13] Eine Hierarchie von Klassen ist genau dann möglich, wenn es Unterschiede in den Verhältnissen der Klassen zueinander gibt. Also wenn Unterteilungen zwischen den Klassen nach bestimmten Kriterien sinnvoll möglich sind.

[11] Es gibt Verfahren bei denen nur die Ergebnisse einiger Basisklassifizierer zur Vorhersage verwendet werden. Vergleiche dazu beispielsweise die DAG (Directed Acyclic Graphs) Algorithmen [Platt *et al.*, 1999]. Diese Fälle stellen eine Ausnahme zu den obigen Aussagen dar.

[12] Gemeint ist, dass entweder die positiven, oder die negativen, Beispiele von deutlich höherer Anzahl sind, als die jeweils anderen.

[13] Vergleich hierzu [Blockeel *et al.*, 2006], wo auf hierarchischen Daten aus der Genfunktionsvorhersage gearbeitet wird

Die Hierarchie wird üblicherweise mit Hilfe eines Baumes dargestellt. In diesem Baum werden die Klassen durch Knoten repräsentiert, wobei jedoch nicht alle Knoten einer Klasse entsprechen müssen, sondern auch nur der Unterteilung im Hierarchiebaum dienen können. Die Blätter des Baumes stellen jedoch alle eine Klasse dar.

Im Folgenden soll geklärt werden, wie Klassenhierarchien in dieser Arbeit semantisch gedeutet werden.

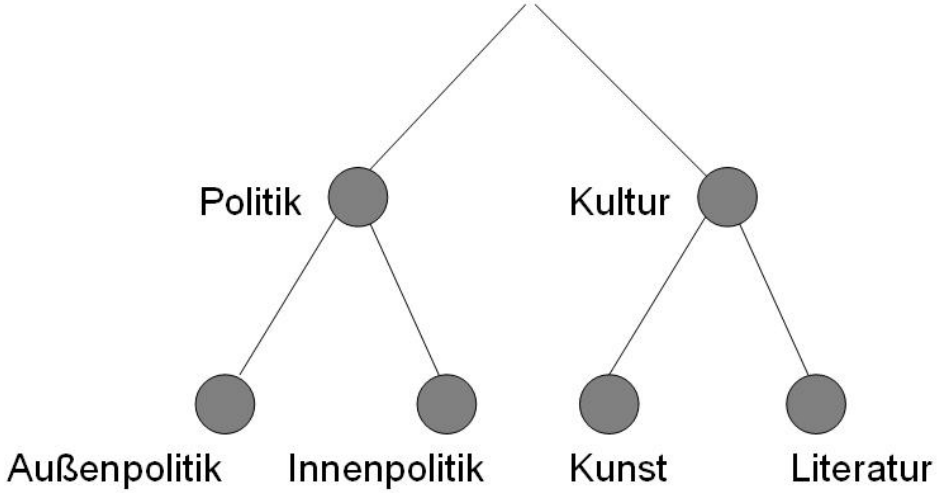

Abbildung 2.6: Beispiel einer Klassenhierarchie für die Textklassifikation

2.6.1 Semantik von Hierarchien

In diesem Abschnitt soll festgelegt und erklärt werden, wie Klassenhierarchien in dieser Arbeit interpretiert werden. Es geht dabei um die Bedeutung von Superklassen im Hierarchiebaum und um die relativen Positionen der Klassen zueinander im Baum.

Bei bestimmten Datensätzen kann eine Klassenhierarchie eine ganz andere Aussage, als die hier beschriebene, haben. Es gibt keine Norm für Klassenhierarchien und daher sollte eine Hierarchie stets im Kontext ihres konkreten Datensatzes interpretiert werden. Allerdings ist es für die weiteren Überlegungen, Definitionen und Ergebnisse dieser Arbeit notwendig, dass eine klare Semantik festgelegt wird. Die hier vorgestellte Interpretation von Klassenhierarchien steht entsprechend im Einklang mit den später in den Versuchen verwendeten Datensätzen. Die Parameter[14] der paarweisen hierarchischen Klassifikation sind ebenfalls auf diese Hierarchiesemantik abgestimmt.

2.6.1.1 Vaterknoten und Superklassen

In einem Baum gilt ein Knoten als Vaterknoten eines anderen, wenn er auf dem Weg zwischen diesem und der Wurzel des Baumes liegt.

[14] Gemeint sind in erster Linie die Regelungen zur Bestimmung der zusätzlichen Trainingsinstanzen.

Definition (Superklasse): *Die Klassen A und B seien durch jeweils einen Knoten im Hierarchiebaum repräsentiert. Ist der Knoten A ein Vaterknoten von B, dann gilt A als eine* Superklasse *von B.*

Dies bedeutet, dass jede Instanz der Klasse B zwingend auch der Klasse A zugeordnet ist. Dies gilt nicht umgekehrt. Konkret für den Datensatz heißt dies in der Regel, dass die Klasse B eine Spezialisierung der Klasse A ist. Verwenden wir das oben abgebildete Beispiel 2.6 aus der Textklassifikation: Die Klasse $Politik$ ist die Superklasse der Klassen $Innenpolitik$ und $Außenpolitik$. Ist ein Text mit der Klasse $Innenpolitik$ assoziiert, dann ist ihm ebenfalls die Superklasse $Politik$ zugeordnet. Ist ein Text andersherum nur mit der Klasse $Politik$ gekennzeichnet, dann können wir davon nicht auf die Zuordnung einer der spezielleren Klassen $Innenpolitik$ oder $Außenpolitik$ schließen. Bei der Betrachtung von Trainings- oder Testinstanzen eines Datensatzes ist es auf Grund dieser Eigenschaft wichtig, dass geprüft wird, ob neben den gesetzten Klassen weitere Superklassen implizit zugeordnet sind. Wenn nicht explizit anders erwähnt, wird in dieser Arbeit immer davon ausgegangen, dass in der Menge der, einer Instanz zugeordneten, Klassen immer alle entsprechenden Superklassen enthalten sind.

Diese Interpretation von Vaterknoten in der Klassenhierarchie lässt sich mit dem Hierarchietyp der so genannten *Subset Hierarchie*[15] assoziieren. Die Klassen der Kinderknoten eines Vaterknotens, stellen in gewisser Weise Teilmengen oder Spezialisierungen der übergeordneten Klasse dar. Sinnvollerweise ist eben dies auch bei der Hierarchie der Reuters Datensätze, welche bei den Versuchen dieser Arbeit als Vertreter von Multi Label Datensätzen verwendet werden, der Fall. Grundsätzlich ist es denkbar, dass bei einigen Datensätzen die Bedeutung von Superklassen eine ganz andere ist und beispielsweise die Notwendigkeit, dass die Zuordnung einer Klasse automatisch auch alle Klassen der Vaterknoten miteinbezieht, nicht gilt.

2.6.1.2 Nähe in Hierarchien

Eine für den Zweck der hierarchischen Klassifikation notwendige Annahme bei Hierarchien ist es, dass die gegebene hierarchische Unterteilung nicht nur einer inhaltlichen Aussage der Klassen entlehnt ist, sondern dass sich diese Unterteilung, sprich Gemeinsamkeiten und Unterschiede der Klassen, auch in den eigentlichen Daten der entsprechenden Instanzen findet. Im Idealfall bedeutet dies, dass eine Instanz in ihren Daten mehr Gemeinsamkeiten mit den Daten einer zweiten Instanz aufweist als mit denen einer dritten Klasse, wenn die entsprechenden ersten beiden Klassen sich im Hierarchiebaum gegenseitig näher sind als jeweils der dritten Klasse. So könnten wir an Hand unseres Beispieles folgern, dass ein Text der Gattung $Innenpolitik$ mehr Gemeinsamkeiten mit einem Text über $Außenpolitik$ hat, als mit einem Text der Klasse $Literatur$.
Die Nähe zwischen Klassen im Baum kann verschiedenartig definiert werden; in dieser Arbeit wird der gemeinsame Weg innerhalb des Baumes als Kriterium gewählt. Eine genaue Definition des verwendeten Maßes findet sich unter 3.3.

Die Eigenschaft von einer Nähe im Hierarchiebaum auf eine Nähe in den konkreten Daten schließen zu können, ist Voraussetzung dafür, die Informationen einer Hierarchie sinnvoll für die Klassifikation solcher Daten verwenden zu können. Entsprechend wird in dieser Arbeit der Begriff von *Nähe*, sofern nicht explizit anders angegeben, gleichbedeutend sowohl für die Nähe von Klassen im Hierarchiebaum als auch für die Nähe zwischen den Daten der Instanzen dieser Klassen verwendet. Die interessante Frage, wie man Nähe schlussendlich in den Daten konkret messen kann, steht hierbei noch offen. Beim theoretischen Vergleich der paarweisen hierarchischen Klassifikation mit der paarweisen Klassifikation im Abschnitt 3.4 und bei den Erklärungen zur Generierung der künstlichen Datensätze unter

[15] Subset Hierarchie lässt mit Teilmengen Hierarchie übersetzen

5.1.2 wird jeweils ein einfaches Maß, basierend auf der euklidschen Distanz, für die Daten-Nähe von Klassen erklärt und verwendet. Da es jedoch generell sehr schwer ist ein solches Maß, welches für jeden Datensatz sinnvoll verwendbar wäre, zu finden, ist auch dieses Maß für den jeweiligen Zweck zwar praktisch und effektiv, aber wohl nicht zu verallgemeinern. Unter 5.3 werden die Ergebnisse einer Untersuchung über die tatsächliche Widerspiegelung der Hierarchie in den bei den Versuchen verwendeten Datensätzen angeführt.

2.6.2 Fallunterscheidung

An dieser Stelle wird eine Fallunterscheidung zwischen den zwei, in dieser Arbeit relevanten, Arten von Hierarchien eingeführt.

Definition (Blatthierarchie): *Eine Hierarchie, welche in einem Baum realisiert, alle Klassen nur als Blätter dargestellt, wird* Blatthierarchie *genannt.*

Dies bedeutet, dass die inneren Knoten des Baumes nur zur Unterteilung dienen und insbesondere keine Superklassen existieren.

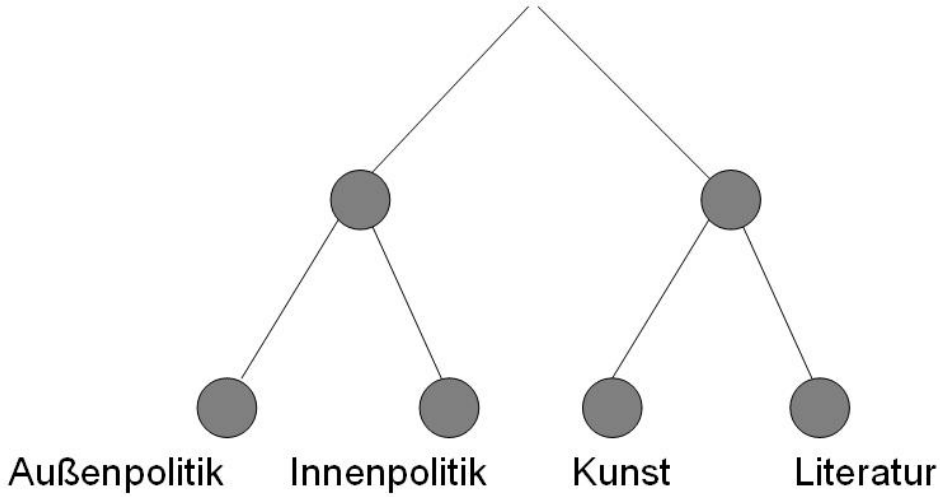

Abbildung 2.7: Beispiel einer Blatthierarchie für die Textklassifikation
Die vorherigen Superklassen existieren hier nicht

Definition (Knotenhierarchie): *Eine Hierarchie, welche in einem Baum realisiert, mindestens eine Klasse als einen inneren Knoten repräsentiert, wird* Knotenhierarchie *genannt.*

Es ist weiter zu erwähnen, dass der Fall eines Single Label Problems nicht sinnvoll mit einer Knotenhierarchie zusammenfallen kann, da es in einer Knotenhierarchie nach Definition Klassen gibt, welche andere Klassen als Vorgänger im Baum haben. Ist eine solche Klasse für eine Instanz gesetzt, dann sind deren Superklassen, nach bereits erläuterter Semantik einer Hierarchie, ebenfalls gesetzt und damit würde es sich um ein Multi Label Problem handeln.

2.6.3 Hierarchische Evaluation

Mit einer gegebenen Hierarchie auf den Klassen ist es möglich bei der Evaluation eines Klassifizierers die, in der Hierarchie enthaltenen, Verhältnisse der Klassen zu berücksichtigen. So kann eine Fehlklas-

sifikation danach gewichtet werden, wie groß der gemachte Fehler aus Sicht der Hierarchie ist. Unter Annahme eines geeigneten Maßes für die Nähe zweier Klassen in der Hierarchie, könnte man die Vorhersage des Klassifizierers danach bewerten, wie nah oder weit die echte Klasse hierarchisch von der vorhergesagten Klasse entfernt ist. In [Dekel *et al.*, 2004] wird eine solches hierarchischen Evaluationsmaß zur Bewertung von verschiedenen Lernverfahren benutzt. Der Fehler wird daran gemessen, wie weit die vorhergesagte Klasse von der korrekten Klasse im Hierarchiebaum entfernt ist; gezählt werden die Kanten, welche zwischen den beiden Knoten liegen. Eines der dort angeführten Ergebnisse ist, dass das verwendete hierarchische Lernverfahren im Vergleich mit einem „flachen“[16] Lernverfahren erwartungsgemäß einen geringeren hierarchischen Fehler produziert. Allerdings lieferte es auch bei den „nicht hierarchischen“ Fehlermaßen, trotz der Tatsache, dass das Verfahren auf die Minimierung des hierarchischen Fehlers trainiert wurde, teilweise bessere Ergebnisse als das „flache“ Verfahren.
Weitere Untersuchungen in denen eine hierarchische Evaluation zur Verwendung kommt sind [Hofmann *et al.*, 2003], [Sun and Lim, 2001] und [Cesa-Bianchi *et al.*, 2004]. In letzter Arbeit von Cesa-Bianchi u.a. wird eine Evaluation für eine Multi Label Knotenhierarchie definiert, so wie wir sie bei den Reuters Datensätzen vorfinden. Im Gegensatz zu den anderen aufgeführten Arbeiten ist hierbei die Tatsache berücksichtigt, dass es Superklassen gibt, welche automatisch durch Vorhersage ihrer „Kinderklassen“ als gesetzt gelten. Weiter ist auch bedacht, dass einige Instanzen nur mit einer Superklasse assoziiert sind, aber gleichzeitig nicht mit einer Nachfolgerklasse dieser. Das Evaluationsmaß basiert grob darauf, zu messen an welchem Knoten in der Hierarchie der Algorithmus den ersten Fehler macht. Falls erstmal ein falscher Teilbaum zur Suche nach der korrekten Klasse gewählt wurde, werden weitere Fehler, die zwingend sind, weil alle Klassen in diesem Teilbaum inkorrekt sind, nicht mehr gezählt.

Eine hierarchische Evaluation wurde in dieser Arbeit nicht vorgenommen. Es wäre jedoch ein durchaus interessanter Anknüpfungspunkt für weitere Versuche.

[16] „Flach“ in dem Sinne, dass die Hierarchie beim Lernen ignoriert wird

3 Paarweise Hierarchische Klassifikation

3.1 Einleitung

In diesem Kapitel soll das Modell der paarweisen hierarchischen Klassifikation vorgestellt werden. Dieser Ansatz basiert auf der „einfachen“ paarweisen Klassifikation mit der Ergänzung, dass Informationen einer gegebenen Klassenhierarchie genutzt werden, um für den Lernprozess des einzelnen Basisklassifizierers zusätzliche Trainingsinstanzen zu bestimmen. Eine Hierarchie stellt eine Unterteilung der Klassen nach Gemeinsamkeiten, bzw. Unterschieden, dar. Diese Informationen sollen nicht wie bei einem „flachen“ Klassifizierer ignoriert werden, sondern die darin gegebenen Verhältnisse der Klassen zueinander sollen mit in die Klassifikation einfließen.

Die wesentliche Umsetzungsidee dieses Konzeptes ist es, jedem paarweisen Basisklassifizierer für den Lernprozess zusätzliche Informationen zu geben. Jeder Basisklassifizierer lernt genau zwei Klassen voneinander zu unterscheiden. Bei unserem hierarchischen Ansatz soll er als zusätzliche positive, bzw. negative, Beispiele die Instanzen dieser Klassen erhalten, welche den beiden zu unterscheidenden Klassen in der Hierarchie nahe stehen. Dafür ist ein entsprechend geeignetes Maß zu finden, welches aus der Hierarchie ableitet kann, welche Klassen sich näher sind als andere.

Im Folgenden wird die paarweise hierarchische Klassifikation definiert. Zunächst wird das verwendete Maß für hierarchische Nähe vorgestellt und anschließend wird formal aufgezeigt, wie an Hand dieses Maßes die zusätzlichen Trainingsinstanzen identifiziert werden. Unter 3.4 werden theoretische Überlegungen der Unterschiede und deren Konsequenzen auf die Klassifikationsergebnisse von paarweiser hierarchischer Klassifikation und der paarweisen Klassifikation dargestellt. Am Ende des Kapitels werden mögliche Variationen dieses hierarchischen Ansatzes kurz angesprochen. Zuerst soll jedoch genauer auf die zugrunde liegende Fragestellung der paarweisen hierarchischen Klassifikation eingegangen werden.

3.2 Fragestellung

Wenn es um die Klassifikation von hierarchischen Daten geht, dann ist es ganz allgemein zunächst einmal intuitiv, die gegebene Hierarchie auf irgendeine Weise für diese Aufgabe nutzen zu wollen. Ein Großteil der üblichen Klassifikationsmethoden und Lernalgorithmen entstand jedoch nicht mit der Zielsetzung eine Klassenhierarchie in den Klassifikationsprozess miteinzubinden. Diese „flachen“, also eine Hierarchie auf den Klassen ignorierenden, Methoden funktionieren sogar meist auch relativ zufriedenstellend auf hierarchischen Datensätzen. Es bleiben die allgemeinen Fragen,

- auf welche Art und Weise man die Hierarchie in bestehende flache Methoden einbinden kann und nicht zuletzt auch
- in welchen Fällen man dadurch tatsächlich einen Gewinn, im Sinne von verbesserten Klassifikationsergebnissen, erreichen kann.

Die letztere Frage beruht auf der Überlegung, dass eine Klassenhierarchie an sich ja auch in den eigentlichen Daten der Instanzen zeigt. So haben Instanzen in den Werten ihrer Attribute Gemeinsamkeiten und auch Unterschiede gemäß dem Verhältnis ihrer Klassen zueinander. Ein flacher Klassifizierer kann die Unterschiede und Gemeinsamkeiten von Klassen also durchaus beim Lernen der Unterscheidungsmodelle quasi „indirekt“ über die Attribute mitverwenden. Es kann natürlich sein, dass die in der Hierarchie enthaltenen Verhältnisse nicht ganz mit den konkreten Daten übereinstimmen; wenn dies jedoch der Fall ist, dann wäre eine Einbeziehung dieser „inkorrekten“ Hierarchie in den Klassifikationsprozess grundsätzlich schon nicht sinnvoll. Daher werden wir, wenn nicht anders erwähnt, immer davon auszugehen, dass eine Hierarchie auch mit den eigentlichen Daten der Instanzen im Einklang ist.

Neben den bereits gestellten, eher allgemein formulierten, Fragen, geht es beim Ansatz der paarweisen hierarchischen Klassifikation konkreter darum,

- ob die Unterscheidung zweier Klassen verbessert werden kann, wenn man sie beim Lernen zusammen mit den Instanzen der ihnen *nahen* Klassen betrachtet.

Unter der Annahme obiges sei grundsätzlich möglich, muss man sich im nächsten Schritt überlegen

- in welchen Fällen, also unter welchen Umständen, das erlernte Modell dadurch besser werden kann.

Im Abschnitt 3.4 werden einige verschiedene Konstellationen von Trainingsdaten unter diesem Aspekt, nämlich wann und warum die paarweise hierarchische Klassifikation einen Vorteil gegenüber der paarweisen Klassifikation haben könnte, betrachtet.

Eine weitere Überlegung bezieht sich auf die Konsequenzen für das Ranking. Aus den Ergebnissen der einzelnen Basisklassifizierer wird mit einer Dekodierungsmethode eine Rangfolge der Klassen erstellt, aus welcher dann die erste Klasse vorhergesagt wird, bzw. bei Multi Label Problemen die ersten n Klassen. Neben den Veränderungen für das einzelne zu lernende Entscheidungsmodell zwischen zwei Klassen ändert sich bei der paarweise hierarchischen Klassifikation auch das Ranking als ganzes. Die Fragen,

- wie das Ranking sich bei paarweiser und paarweiser hierarchischer Klassifikation verändert und
- von welchem Nutzen diese Änderungen des Ranking sein könnten,

werden ebenfalls im Abschnitt 3.4 behandelt.

Alle diese Fragen stellen die Basis für die Beschäftigung mit dem Thema einer hierarchischen Klassifikation und der ganz konkreten Ausprägung der paarweisen hierarchischen Klassifikation in dieser Arbeit dar. Einige der hier aufgeworfenen Frage sind sehr weit gefasst und es gibt wahrscheinlich keine vollständigen Antworten darauf, da einzelne Klassifikationsaufgaben sich zu sehr voneinander entscheiden können, als dass man eine konkrete Methode als generelle Lösung für alle Datensätze betrachten könnte.

Für die konkreteren Fragestellungen jedoch werden wir mit den Ergebnissen der Versuche eine ungefähre Einschätzung der Leistungsfähigkeit der paarweisen hierarchischen Klassifikation erhalten. Innerhalb des Rahmens dieser Arbeit wurde versucht die Datensätze der Versuche so zu wählen und zu variieren, dass die meisten interessanten Fälle abgedeckt sind, so dass wir schlussendlich in der Lage sein sollten, ein grobes Fazit für den vorgestellten hierarchischen Ansatz zu ziehen.

3.3 Definition

Im Folgenden wird das Modell der paarweisen hierarchischen Klassifikation definiert. Insbesondere ist zu klären, nach welchen Kriterien die zusätzlichen Trainingsinstanzen der einzelnen Basisklassifizierer ausgewählt werden. Dafür ist es zunächst notwendig ein Maß für die hierarchische Nähe der Klassen zu definieren. Das hier beschriebene Modell wird in dieser Form bei den später folgenden Versuchen verwendet; es sind jedoch grundsätzlich verschiedene Modifizierungen möglich, von denen einige grob unter dem Punkt 3.5 angedeutet werden.

3.3.1 Verwandtschaft: Ein Maß für hierarchische Nähe

Bei der paarweisen hierarchischen Klassifikation wird zunächst die bereits unter 2.5 definierte paarweise Binarisierung vorausgesetzt. Nun soll weiter festgelegt werden, nach welchem Regeln dem einzelnen Basisklassifizierer für sein binäres Problem zusätzliche Trainingsinstanzen zugeordnet werden.

Zuerst soll geklärt werden, wie die Hierarchie interpretiert wird. Dafür definieren wir ein Maß für hierarchische Nähe:

Definition (Verwandtschaft): *Als* Verwandtschaft *zwischen zwei Klassen gilt der gemeinsame Weg im Hierarchiebaum. Beide Klassen sind im Baum durch jeweils einen Knoten repräsentiert. Die* Verwandtschaft *ist dann die Anzahl der, von der Wurzel des Baumes an gezählten, gemeinsamen Kanten dieser Knoten. Dieser Wert kann bei zwei nichtgleichen Klassen entsprechend zwischen* 0 *und* $(Baumtiefe - 1)$ *liegen. Die* Verwandtschaft *zwischen zwei Klassen* A *und* B *wird dargestellt mit* $V(A, B)$.

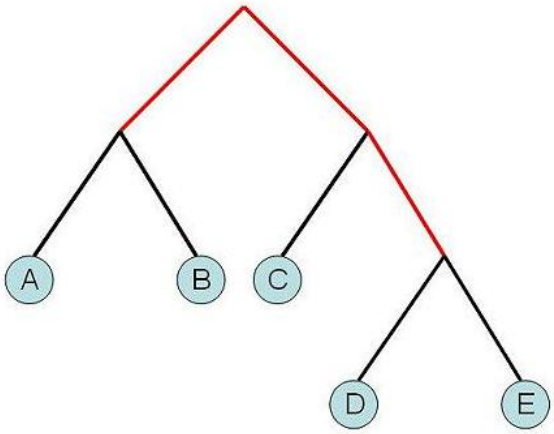

Abbildung 3.1: Verwandtschaftmaß im Hierarchiebaum

Die Verwandtschaft dient als ein Maß für die Nähe von Klassen in der Hierarchie. Da der Wert der Verwandtschaft allerdings davon abhängt auf welcher Ebene des Baumes die Klassen liegen, dient sie nicht zwingend zur Unterscheidung ob zwei Klassen sich hierarchisch näher sind als zwei andere Klassen. Betrachten wir hierzu die Abbildung 3.1, in der der gemeinsame Weg jeweils rot gekennzeichnet ist; die Verwandtschaft zwischen A und B beträgt 1 und die Verwandtschaft zwischen D und E beträgt 2. Daraus lässt sich nun aber nicht schließen, dass die Klassen D und E mehr Gemeinsamkeiten haben

als A und B.[1] Unser Maß gibt hierarchische Nähe also nicht in einem *absoluten* Sinne an, sondern nur *relativ* als Vergleich welche von zwei Klassen einer Dritten hierarchisch näher steht.
Folglich ergibt sich eine Relation auf Tripeln von Klassen. Das heißt es lassen sich Aussagen der Form: „Klasse A ist hierarchisch näher an Klasse B als an Klasse C“ aus $V(A,B) > V(A,C)$ ableiten. In der Abbildung 3.1 kann man erkennen, dass zum Beispiel die Klasse C näher an D und auch an E ist, als an entweder A oder B.

Der gemeinsame Weg zweier Klassen im Hierarchiebaum ist selbstverständlich nur eine von vielen Möglichkeiten um das hierarchische Verhältnis zwischen Klassen zu messen. Entscheidend für die Auswahl eines Maßes ist die Konformität dieses mit dem zu bearbeitenden Datensatz; genauer gesagt mit der Semantik der Klassenhierarchie. Das Maß der Verwandtschaft deckt sich mit der, in dieser Arbeit durchgehend angenommenen, Hierarchiesemantik wie sie unter 2.6.1 beschrieben wurde. Gleichzeitig heißt dies aber nicht, dass die Verwandtschaft das einzige unter diesen Umständen verwendbare Maß ist. Vergleiche hierzu auch den Abschnitt über Variationen der paarweisen hierarchischen Klassifikation unter 3.5.

3.3.2 Trainingsinstanzen

Bei der paarweisen hierarchischen Klassifikation werden bei jedem Klassifizierer, neben den beim „klassischen“ paarweisen Ansatz zugeordneten Instanzen, noch zusätzliche Instanzen zum Training verwendet. Nach welchen Kriterien diese Menge an Trainingsinstanzen erstellt wird, soll in diesem Abschnitt definiert werden.

Betrachtet sei jeweils der paarweise Klassifizierer $K_{A,B}$, der zwischen den Klassen A und B unterscheidet. Sei T_A die Menge der positiven Trainingsinstanzen, T_B die Menge der negativen Trainingsinstanzen des Klassifizierers $K_{A,B}$ und x eine Instanz der Klasse X.

An dieser Stelle wird eine Fallunterscheidung zwischen Single Label und Multi Label Datensätzen getroffen, weil die beiden Fälle sehr unterschiedlich behandelt werden müssen.

Single Label Datensätze
Die *zusätzlichen* Trainingsinstanzen für $K_{A,B}$ sind:

- Alle Trainingsinstanzen, deren Klasse eine höhere Verwandtschaft mit A als mit B hat, werden als Trainingsbeispiele für A verwendet

$$x \in T_A \iff V(A,X) > V(B,X)$$

- Alle Trainingsinstanzen, deren Klasse eine höhere Verwandtschaft mit B als mit A hat, werden als Trainingsbeispiele für B verwendet

$$x \in T_B \iff V(A,X) < V(B,X)$$

- Bei gleicher Verwandtschaft wird die Instanz nicht zum Training verwendet.

Betrachten wir nun mit welchen Trainingsinstanzen die paarweisen Klassifizierer bei unserer Beispielhierarchie in Abbildung 3.1 lernen würden. Wie in der Tabelle 3.1 zu erkennen ist, werden zum Lernen der Klassifizierer oft die Instanzen aller Klassen eines ganzes Teilbaumes der Hierarchie verwendet. Bei einer größeren Anzahl an Klassen und insbesondere größeren Teilbäumen resultiert dies in relativ

[1] Es gibt sicherlich Datensätze, bei denen man dies korrekt folgern könnte, aber im Allgemeinen ist das nicht der Fall.

umfangreichen Mengen an Trainingsinstanzen. Dies hat zwei wesentliche Konsequenzen: einmal wird er Lernprozess umso langsamer desto mehr Trainingsinstanzen berücksichtigt werden, weiterhin steigt auch die Komplexität des zu lösenden Unterscheidungsproblems mit der Anzahl der Instanzen. Diese Eigenschaften werden im Abschnitt 3.4 genauer beleuchtet.

Tabelle 3.1: Single Label Trainingsinstanzen der Klassifizierer.
In den Spalten 2 und 3 sind stets die Instanzen der jeweiligen Klassen gemeint.

KLASSIFIZIERER	POSITIVE BEISPIELE	NEGATIVE BEISPIELE
$K_{A,B}$	A	B
$K_{A,C}$	A,B	C,D,E
$K_{A,D}$	A,B	C,D,E
$K_{A,E}$	A,B	C,D,E
$K_{B,C}$	A,B	C,D,E
$K_{B,D}$	A,B	C,D,E
$K_{B,E}$	A,B	C,D,E
$K_{C,D}$	C	D,E
$K_{C,E}$	C	D,E
$K_{D,E}$	D	E

Ein anderer interessanter Punkt, der ebenfalls gut der Tabelle 3.1 zu entnehmen ist, ist die Tatsache, dass bei der paarweisen hierarchischen Klassifikation einige der Basisklassifizierer dieselbe Menge an Trainingsinstanzen zugeordnet bekommen. Das bedeutet, dass diese Klassifizierer auch das genau gleiche Unterscheidungsmodell erlernen; die Klassifizierer sind also exakt gleich. Dieser Fall tritt für alle Klassifizierer auf, welche zwei Klassen aus denselben zwei unabhängigen[2] Teilbäumen der Hierarchie unterscheiden sollen. Unter 4.2 wird aufgezeigt, dass die paarweise hierarchische Klassifikation auf Grund dieser Eigenschaft bei Single Label Problemen teilweise mit dem Modell der Pachinko Maschine übereinstimmt.

Multi Label Datensätze

Bei Multi Label Problemen ist einer Trainingsinstanz eine Menge von Klassen zugeordnet[3]. Diese Tatsache macht die Identifikation der zusätzlichen Trainingsinstanzen für den einzelnen Klassifizierer etwas komplizierter, weil beispielsweise eine Klasse der Trainingsinstanz näher an der Klasse A als an der Klasse B ist, während eine andere Klasse derselben Instanz dagegen näher an B als an A liegt. In solchen Fällen kann diese Trainingsinstanz nicht für den Lernprozess des Klassifizierers $K_{A,B}$ verwendet werden. Weiterhin ist es im Vergleich zur Behandlung bei Single Label Problemen nicht immer möglich jede der Instanzen zum Trainings zu verwenden, welche der Klasse A oder B direkt zugeordnet ist, weil eine Instanz möglicherweise sowohl A als auch B gleichzeitig zugeordnet sein könnte.

Es seien weiterhin T_A und T_B die Mengen der Trainingsinstanzen des Klassifizierers $K_{A,B}$. Sei x eine Trainingsinstanz und X die Menge der ihr zugeordneten Klassen.

Dem Klassifizierer $K_{A,B}$ lernt sein Unterscheidungsmodell mit *genau* den folgend angeführten Trainingsinstanzen:

- Die Trainingsinstanzen, welchen die Klasse A zugeordnet ist, aber gleichzeitig nicht die Klasse

[2] Gemeint ist hier, dass die beiden Teilbäume keine gemeinsamen Knoten oder Kanten haben.
[3] Die Mächtigkeit der Menge kann auch 1 sein, wenn eine Instanz mit nur einer Klasse assoziiert wird

B, werden als Trainingsinstanzen für die Klasse A verwendet.

$$x \in T_A \Longleftrightarrow A \in X \wedge B \notin X$$

- Die Trainingsinstanzen, welchen die Klasse B zugeordnet ist, aber gleichzeitig nicht die Klasse A, werden als Trainingsinstanzen für die Klasse B verwendet.

$$x \in T_B \Longleftrightarrow B \in X \wedge A \notin X$$

- Alle Trainingsinstanzen, bei denen *mindestens eine* der zugeordneten Klassen eine höhere Verwandtschaft mit A als mit B hat und gleichzeitig *keine* der zugeordneten Klassen eine höhere Verwandtschaft mit B als mit A hat, werden als Trainingsbeispiele für A verwendet.

$$x \in T_A \Longleftrightarrow \forall x_i \in X.V(x_i, A) \geq V(x_i, B) \wedge \exists x_j \in X.V(x_j, A) > V(x_j, B)$$

- Alle Trainingsinstanzen, bei denen *mindestens eine* der zugeordneten Klassen eine höhere Verwandtschaft mit B als mit A hat und gleichzeitig *keine* der zugeordneten Klassen eine höhere Verwandtschaft mit A als mit B hat, werden als Trainingsbeispiele für B verwendet.

$$x \in T_B \Longleftrightarrow \forall x_i \in X.V(x_i, B) \geq V(x_i, A) \wedge \exists x_j \in X.V(x_j, B) > V(x_j, A)$$

Wie auch bei der paarweisen Klassifikation werden hier Klassifizierer der Form $K_{A,A'}$ mit A als Superklasse von A' nicht erlernt, weil es keine Trainingsinstanzen für A' gibt, welche nicht auch der Klasse A angehören.

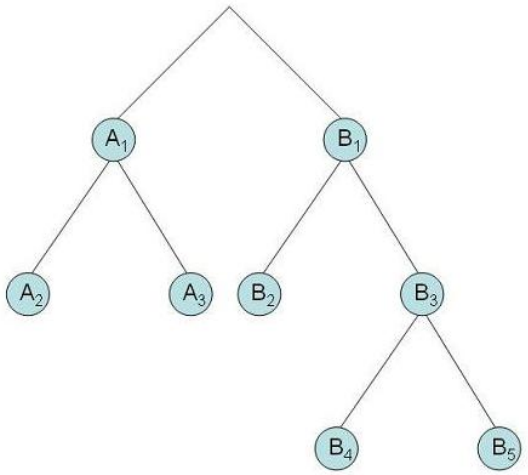

Abbildung 3.2: Beispielhierarchie für Multi Label

Die Tabelle 3.2 zeigt beispielhaft an Hand der Hierarchie aus der Abbildung 3.2 für welche der paarweisen Klassifizierer einige Beispielinstanzen verwendet werden würden.

3.4 Vergleich mit der paarweisen Klassifikation

In diesem Abschnitt sollen die beiden Klassifikationsmethoden, paarweise Klassifikation und paarweise hierarchische Klassifikation, miteinander verglichen werden. Insbesondere sollen auf theoretischer

Tabelle 3.2: Zuordnung von Multi Label Instanzen zu Klassifizierern.
In Spalte 1 sind Trainingsinstanzen durch die Menge ihrer Klassen repräsentiert. Spalte 2 listet die Klassifizierer auf für welche die jeweilige Instanz als positives Beispiel verwendet wird. Der besseren Übersicht und Lesbarkeit halber wird der Klassifizierer $K_{A,B}$ als $A > B$ dargestellt.

INSTANZ	KLASSIFIZIERER
A_2, A_1	$A_2 > A_3, \quad A_x > B_x$
B_4, B_3, B_1	$B_3 > B_2, \quad B_4 > B_2, \quad B_4 > B_5, \quad B_x > A_x$
A_2, A_1, B_2, B_1	$A_2 > A_3, \quad B_2 > B_3, \quad B_2 > B_4, \quad B_2 > B_5$
B_1	$B_x > A_x$

Ebene Überlegungen über die möglichen Klassifikationsergebnisse des hierarchischen Ansatzes angestellt werden.

Um uns die Konsequenzen der paarweisen hierarchischen Klassifikation für einzelne Fälle anschaulich zu machen, betrachten wir im Folgenden eine „vereinfachte“ Klassifikationsaufgabe: Gesucht wird von den Basisklassifizierern jeweils eine lineare Trenngerade zwischen den Punktmengen der positiven und negativen Trainingsinstanzen. Die Mengen der Instanzen sind, zusammengefasst nach Klassenzugehörigkeit, als Kreise dargestellt. Man kann eine Gleichverteilung der Instanzen in diesen Kreisen annehmen. Durchgehend verwenden wir den Klassifizierer $K_{A,B}$ und die Klassen A und B, wobei Klassen der Form A' Klassen darstellen, die A hierarchisch näher sind als B und entsprechend zusätzliche Trainingsinstanzen für den Lernprozess der hierarchischen Basisklassifizierer stellen. Die hierarchische Nähe wird, in den hier angeführten Beispielen, durch die Distanz der Mittelpunkte der Kreise umgesetzt. Dies bedeutet der Kreis der Klasse A' ist dem der Klasse A näher als dem der Klasse B.

Definitionsgemäß wird der einzelne Basisklassifizierer der paarweisen hierarchischen Klassifikation in den meisten Fällen [4] zum Lernen seines Unterscheidungsmodells mehr Trainingsinstanzen verwenden als das nicht-hierarchische Modell. Die Anzahl der Klassen, deren Instanzen hinzugefügt werden, hängt davon ab, wie weit die beiden eigentlich zu unterscheidenden Klassen, im Falle von $K_{A,B}$ also eben A und B, in der Hierarchie auseinander liegen. Wenn diese hierarchisch weit auseinander liegen, dann erfüllen mehr Klassen das Kriterium, einer der beiden Klassen hierarchisch näher zu sein als der anderen. Überlegen wir also was die generellen Auswirkungen sind, wenn die Anzahl der Trainingsinstanzen erhöht wird.

Zunächst benötigt der Lernprozess und auch die spätere Ausführung eines Basisklassifizierers mehr Zeit, wenn die Menge der Trainingsinstanzen größer ist. Man müsste also bei der paarweisen hierarchischen Klassifikation von einer höheren Laufzeit ausgehen als bei der paarweisen Klassifikation. Unter Verwendung des Verwandtschaftsmaßes, so wie es unter 3.3.1 definiert ist, muss dies jedoch relativiert werden, da viele der hierarchischen Basisklassifizierer die gleiche Trainingsmenge besitzen, so dass effektiv nur einer dieser Klassifizierer erlernt werden muss. Bei einem Single Label Datensatz entspricht die Anzahl der effektiv zu lernenden Klassifizierer auf Grund dieser Tatsache der Anzahl der internen Knoten des Hierarchiebaumes. Eine genaue Abschätzung der Gegenüberstellung der Laufzeiten ist nur mit Wissen über die Hierarchiestruktur sowie über die Klassenverteilung unter den Instanzen möglich. Der Fokus in diesem Abschnitt soll jedoch mehr auf den Unterschieden in der Qualität der Klassifikation als auf der Komplexität der Methoden liegen.

[4] Bei der Unterscheidung von 2 Klassen, welche die einzigen Nachfolger eines internen Knoten im Hierarchiebaumes sind, lernen beide Klassifikationsmethoden dasselbe Modell.

Eine höhere Anzahl an Trainingsinstanzen hat natürlich auch Einfluss auf das zu lernende Unterscheidungsmodell. Es wird oft argumentiert, dass ein einfacheres Modelle einem komplexeren Modell vorzuziehen ist. Dies liegt unter anderem daran, dass die Gefahr des Overfitting bei einfacheren Modellen weniger geben ist und diese Modelle daher auf neuen, bisher ungesehenen, Daten besser funktionieren. Die erlernten Modelle werden trivialer Weise einfacher wenn man zum Lernen weniger Trainingsinstanzen betrachtet. Man kann sich dies unter anderem am Beispiel eines linearen Problems so veranschaulichen: Mit einer steigenden Anzahl von Instanzen wird es schwerer eine Trennlinie zwischen den positiven und negativen Beispielen zu finden, weil der Abstand zwischen den zwei Punktmengen durch Hinzufügen von Instanzen entweder gleich bleiben oder kleiner werden wird. Umso kleiner dieser Abstand wird, desto schwerer ist es auch eine Trennlinie zwischen die Mengen zu setzen und desto komplexer wird in der Regel das erlernte Modell. Bei der paarweisen hierarchischen Klassifikation können wir generell davon ausgehen, dass die meisten Basisklassifizierer auf Grund der größeren Mengen an Trainingsinstanzen komplexere Unterscheidungsmodelle als die entsprechenden Basisklassifizierer der paarweisen Klassifikation lernen werden.

Für das zu lernende Modell ist es ein entscheidender Faktor, wie gut die Trainingsinstanzen der beiden Klassen, die unterschieden werden sollen, die jeweilige Charakteristik ihrer Klasse wiedergeben. Betrachten wir ein einfaches Beispiel: Zur Klasse A gehören alle Zahlen des Intervalls $[0; 50]$ und zur Klasse B alle Zahlen im Intervall $[51; 100]$. Wenn die Trainingsinstanzen von A die drei Zahlen $\{1, 3, 12\}$ sind, dann repräsentieren diese Instanzen die Klasse A nur mangelhaft; eine zum Lernen des eigentlichen Unterscheidungskonzeptes zwischen A und B deutlich bessere Trainingsmenge wäre $\{0, 25, 50\}$.

Stellen wir also fest, dass unter der Annahme die Traingsinstanzen zweier Klassen repräsentieren diese bereits sehr gut, die Hinzunahme zusätzlicher Instanzen auf beiden Seiten die Klassifikation wahrscheinlich nicht verbessern, sondern wahrscheinlich nur verschlechtern könnte, weil dadurch die Komplexität des Problem erhöht und damit die Suche nach einer einfachen Lösung erschwert wird. Es ist durchaus anschaulich, dass das eigentliche Problem A und B voneinander trennen zu können, für einen Lerner oft schwerer zu lösen sein könnte, wenn neben Instanzen von A und B noch einige weitere Instanzen von ähnlichen Klassen miteinbezogen werden, da auf diese Weise die wesentlichen Unterscheidungsmerkmale von A und B in den Daten der anderen Instanzen „untergehen“ können.

Falls die ursprünglichen Trainingsinstanzen jedoch nicht optimal oder sogar mangelhaft sind, dann lassen sich einige Szenarien gestalten, in denen die paarweise hierarchische Klassifikation eine Verbesserung des Unterscheidungsmodells bewirkt. Einige dieser Fälle sollen nun betrachtet werden.

In der Abbildung 3.3 ist ein Beispiel dargestellt, in dem eine kleine Menge der Trainingsinstanzen der Klasse A verrauscht, d.h. fehlerhaft, ist und daher außerhalb des eigentlichen Bereiches der Instanzen der Klasse A liegt. Einige Lernverfahren würden, wie im ersten Bild aufgezeigt, eine Trennlinie zur Unterscheidung von A und B lernen, welche versucht die verrauschten Instanzen korrekt zu klassifizieren. Tatsächlich ist dies eine Form des Overfittings und folglich ist das gelernte Konzept nicht optimal generalisiert. Zwar würden Instanzen, welche sich innerhalb des Kreises ihrer Klasse befinden, in diesem Fall trotzdem korrekt eingeordnet werden, es können jedoch schon bei kleinen Abweichungen Fehlklassifikationen auftreten, da die Trennlinie sehr nahe an den Grenzen der Kreise liegt. Weiterhin kann es bei realen Anwendung vorkommen, dass die Trainingsinstanzen den eigentlichen Bereich der Klassen nicht komplett darstellen, was bildlich bedeuten würde, dass Instanzen der Klasse sich nicht nur im Bereich des Kreises der Trainingsinstanzen sondern auch etwas darüber hinaus bewegen. Daher ist es im Allgemeinen wünschenswert, wenn das Unterscheidungsmodell Trennlinien findet, die einen möglichst großen Abstand zu den beiden Instanzenmengen haben.[5]

[5] Vergleiche das Konzept des Separation Margin bei Support Vector Maschinen. Siehe dazu auch 2.3

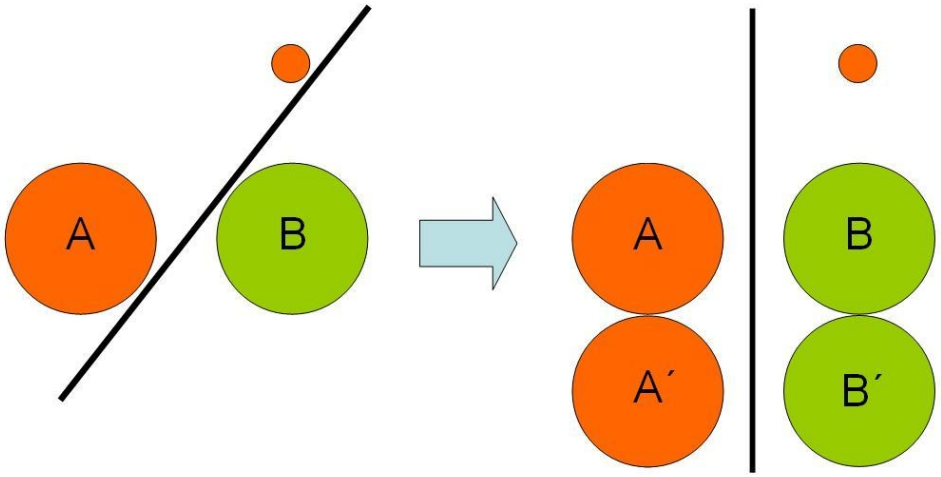

Abbildung 3.3: Relativierung weniger verrauschter Instanzen bei der paarweisen hierarchischen Klassifikation

Bei zweiten Bild der Abbildung 3.3 wurden die positiven Trainingsbeispiele für die Klasse A mit den Trainingsinstanzen der ihr hierarchisch nahen Klasse A' erweitert. Die negativen Trainingsbeispiele sind nun entsprechend die Trainingsinstanzen der Klassen B und B'. In dieser Konstellation fallen die fehlerhaften Instanzen weniger ins Gewicht, weil sie in ihrer Anzahl nun geringer sind, relativ gesehen zu der Menge der anderen Instanzen, welche für sich gut in ein Unterscheidungskonzept zu fassen sind. Viele Lernalgorithmen[6] würden in diesem Fall die Ausreißer eher als solche erkennen und sie beim Lernvorgang ignorieren. Als Resultat würde der Basisklassifizierer $K_{A,B}$ der hierarchischen Klassifikation ein besseres, im Sinne von generalisierteres, Modell erlernen, als der Klassifizierer der normalen paarweisen Klassifikation.

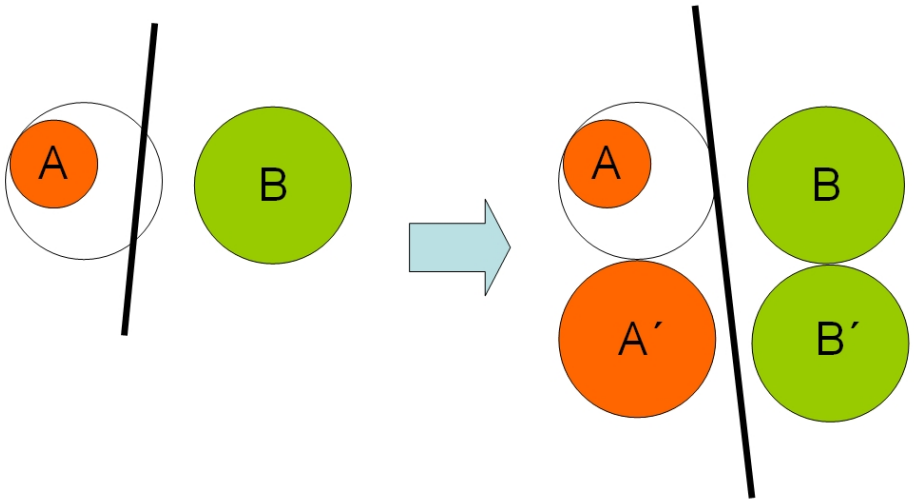

Abbildung 3.4: Klasse A hat nur unzureichende Trainingsinstanzen

Als nächstes soll ein Beispiel für den Fall, dass eine der Klassen nur sehr wenige Trainingsinstanzen

[6] Vergleiche hierzu das Konzept des Soft Margin bei Support Vector Maschinen unter 2.3

besitzt, behandelt werden. Stehen nur sehr wenige Beispielinstanzen einer Klasse zur Verfügung, dann können diese die klasseneigene Charakteristik nur unzureichend beschreiben. Daraus folgt natürlich auch, dass es schwer ist ein gutes Konzept zu finden, um diese Klasse von anderen zu unterscheiden. Die paarweise hierarchische Klassifikation kann in diesen Fällen das Wissen um Gemeinsamkeiten und Unterschiede zwischen den Klassen, welches sie aus der Hierarchie bezieht, nutzen und eine einzelne Klasse in dem Kontext der ihr nahe Klassen betrachten, so dass die Unterrepräsentation einer einzelnen Klassen eventuell durch die hierarchischen Nachbarn ausgeglichen wird. In Abbildung 3.4 wird ein solcher Fall gezeigt. Die wenigen Trainingsinstanzen der Klasse A sind konzentriert auf einen kleinen Bereich innerhalb des äußeren Kreises, welcher die eigentlichen Ausbreitung der Klasse A beschreibt. Bei der paarweisen Klassifikation liegt in diesem konkreten Fall die Trennlinie so, dass es bei der Klassifikation von Instanzen der Klasse A in mehreren Fällen zu inkorrekten Klassifikationen kommt.

Bei der paarweisen hierarchischen Klassifikation, die im zweiten Bild der Abbildung 3.4 zu sehen ist, wurden die Instanzen zweier hierarchienaher Klassen zur Trainingsmenge hinzugefügt. Man kann erkennen, wie die Trennlinie nun weniger von der geringen Menge der Instanzen von A beeinflusst ist und die Unterscheidung zwischen A und B dadurch verbessert wurde. Es ist vorstellbar, dass die Unterscheidungslinie sich weiter einer „optimalen“ Trennlinie, genau zwischen den beiden Klassen A und B, annähren würde, wenn auf beiden Seiten noch weitere Instanzen ergänzt werden.

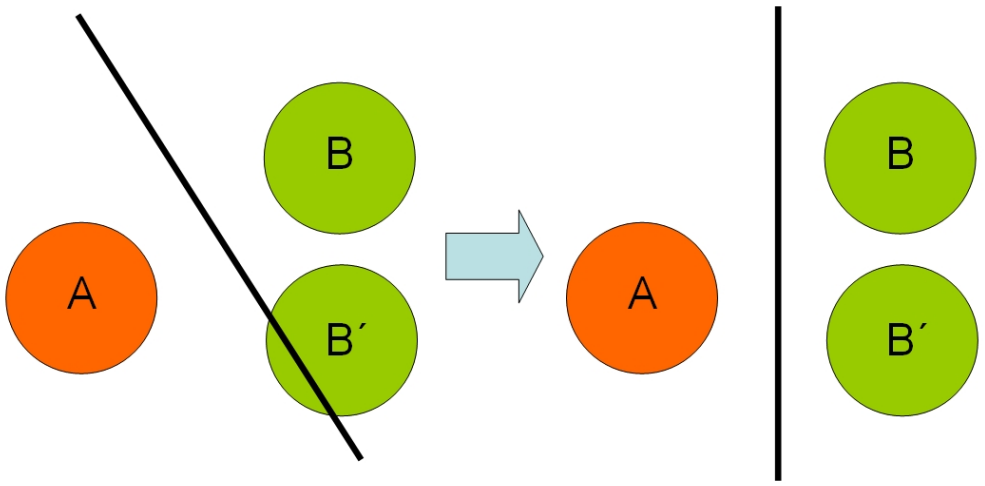

Abbildung 3.5: Vergleich der gelernten Trennlinie mit und ohne zusätzliche Instanzen

Wie im Abschnitt 3.2 „Fragestellung“ schon beschrieben wurde, sollte man, neben einer Betrachtung des einzelnen paarweisen Klassifizierers und dessen Unterschieden bei der paarweisen und der paarweisen hierarchischen Klassifikation, auch die Gesamtauswirkungen des hierarchischen Ansatzes auf das Ranking als Ganzes untersuchen. Für das Ranking relevant ist es nicht nur wie der Klassifizierer $K_{A,B}$ entscheidet wenn er mit einer Instanz des Klasse A oder B konfrontiert wird, sondern auch wie er für die Instanzen aller anderen Klassen entscheidet. Betrachten wir dazu die Abbildung 3.5, in der man die unterschiedliche Trennlinie einmal ohne und einmal mit Einbeziehung der Klasse B' beim Lernprozess sieht. Zunächst lässt sich feststellen, dass sich die Qualität[7] der Unterscheidung zwischen A und B in den beiden Fällen wenig unterscheidet. Unterschiedlich ist aber das Verhalten der beiden Klassifizierer für den Fall, dass diese eine Instanz der Klasse B' einordnen sollen. Der paarweise Klassifizierer im ersten Bild wird die Instanz zwar größtenteils der Klasse B aber auch teilweise der Klasse A zuordnen. Der paarweise hierarchische Klassifizierer dagegen wird Instanzen der Klasse B'

[7] Die Qualität ist hier daran gemessen wie weit die Trennlinie von den beiden Ausbreitungsbereichen der Klassen entfernt ist

höchstwahrscheinlich stets der Klasse B zuordnen.
Man kann zusammenfassen, dass wir allgemein von einem Basisklassifizierer der paarweisen hierarchischen Klassifikation erwarten, dass er einer Instanz öfter diejenige Klasse zuordnet, welche ihrer eigentlichen Klasse hierarchisch näher ist, als es derselbe Basisklassifizierer der paarweisen Klassifikation tun würde. Daraus folgt automatisch, dass wir bei der Klassifikation einer Instanz der Klasse X die hierarchienahen Klassen von X höher im Ranking der paarweise hierarchische Klassifikation erwarten können als in dem Ranking des nicht-hierarchischen Ansatzes.
Unter 2.5.2 wurde bereits aufgezeigt, dass bei der paarweisen Klassifikation die korrekte Klassenvorhersage für eine Instanz der Klasse X alleine durch die korrekte Vorhersage aller Klassifizierer der Form $K_{X,Y}$ gesichert ist und damit für diese Fälle die Ergebnisse aller anderen Basisklassifizierer irrelevant sind. Dies gilt offensichtlich ebenso auch für die paarweise hierarchische Klassifikation. In diesen Fällen ist es also unwichtig, wie ein Klassifizierer $K_{A,B}$ für eine Instanz der Klasse C entscheidet. Wenn nun der Top Rank aber nicht korrekt bestimmt wurde, weil einige Klassifizierer die falsche Klasse vorhergesagt haben, kann sich bei den beiden Klassifikationsmethoden durchaus ein Unterschied in dem gemachten Fehler einstellen. Da, wie schon festgestellt, die hierarchienahen Klassen der gesuchten Klasse X höher im vom hierarchischen Ansatz erstellten Ranking stehen, ist die Chance höher, dass der (inkorrekte) Top Rank sich ebenfalls hierarchisch näher an der gesuchten Klasse X befindet, als es bei dem nicht-hierarchischen Ranking der Fall wäre. Das würde bedeuten, dass der gemachte Fehler des Gesamtklassifizierers nach einer hierarchischen Evaluation[8] bei der paarweisen hierarchischen Klassifikation als geringer zu erwarten ist.
Diese Annahmen bestärkten sich unter 5.2.4 bei einer stichprobenhaften Betrachtung ausgewählter Rankings der beiden Methoden, welche für Instanzen des Reuters Datensatzes erstellt wurden. Eine systematischere Überprüfung wäre zur Weiterführung interessant.

Es ist zu bedenken, dass alle in diesem Abschnitt angedachten Situationen sehr speziell sind und bei weitem keine umfangreiche Information über das konkrete Verhalten der paarweisen hierarchischen Klassifikation geben können. Letztendlich lassen sich die Ergebnisse auf realen Datensätzen praktisch nicht vorhersagen, weil alleine die jeweilige Struktur der Hierarchie, der Instanzen und deren Attributen eine entscheidende Rolle darin spielen können, welche Klassifikationsmethoden gut und welche weniger gut funktionieren. Die, für die aufgeführten Beispiele, verwendeten Klassifikationsaufgaben sind in ihrem Aufbau sehr simple und es ist nicht einfach die hier gezeigten Eigenschaften problemlos auf komplexere Probleme[9] zu verallgemeinern. Die ursprüngliche Intention jedoch, nämlich einige theoretische Überlegungen über mögliche Folgen des hierarchischen Ansatzes anzustellen, wurde erreicht.

Bei den Versuchsergebnissen, insbesondere denen der künstlichen Datensätze unter 5.1.4, werden einige der hier angesprochenen Punkte wieder aufgegriffen und deren Gültigkeit auf den konkreten Datensätzen überprüft. Bei den künstlichen Datensätzen wurden, wie unter 5.1.3 beschrieben ist, die Trainingsinstanzen so generiert, dass verschiedene Stufen von Rauschen, sowie Fälle mit nur sehr wenigen Trainingsinstanzen für einige Klassen, getestet werden können.

3.5 Variationen

Im Kapitel 5 „Versuche und Ergebnisse“ wird durchweg die paarweise hierarchische Klassifikation in genau der oben definierten Form verwendet. Die grundlegende Idee dieses Ansatzes lässt sich jedoch in

[8] Das heißt der Fehler wird als umso größer gesehen, desto weiter die vorhergesagte Klasse in der Hierarchie von der echten Klasse entfernt ist.
[9] Zum Beispiel schon bei Instanzen mit 30 Attributen statt nur 2

vieler Hinsicht variieren und es kann für weiterführende Arbeiten durchaus interessant sein, die Möglichkeiten dieser Variationen zu untersuchen. Insbesondere kann das Verfahren durch entsprechende Veränderungen auf bestimmte Gegebenheiten eines Datensatzes angepasst werden, um die Klassifikationsergebnisse zu optimieren. Im Folgenden sollen einige dieser Möglichkeiten kurz angedacht werden.

3.5.1 Anderes Maß für Hierarchienähe

Die hierarchische Nähe zwischen Klassen kann neben dem Verwandtschaftsmaß auch mit anders definierten Methoden gemessen werden. Entscheidend sollte die Kompatibilität des Maßes mit der Semantik der Hierarchie sein. Eine intuitive Methode ist es den Abstand zwischen zwei Klassen im Hierarchiebaum zu messen, indem man die Kanten, welche auf dem kürzesten Weg zwischen den beiden Klassen liegen, zählt. Dieser Ansatz wurde in [Dekel *et al.*, 2004] für eine hierarchische Evaluation von Klassifizierern verwendet. Die Methode kann nach Bedarf noch an Besonderheiten der Hierarchie angepasst werden, indem zum Beispiel bestimmte Wege, wie solche, die über die Wurzel des Baumes führen, speziell gezählt werden.
Durch die Variation des Maßes für Hierarchienähe kann man die Anzahl und die Zusammensetzung der, für den einzelnen Basisklassifizierer ausgewählten, zusätzlichen Trainingsinstanzen beeinflussen. Neben diesen offensichtlichen Konsequenzen, kann ein bestimmtes Hierarchiemaß auch andere nicht augenscheinliche Auswirkungen haben. Wie wir unter 4.2 genauer sehen werden, führt die Verwendung des Verwandtschafts-Maßes dazu, dass die paarweise hierarchische Klassifikation sich bei Single Label Problemen weitgehend wie ein, auf einer ganz anderen Idee basierendes, Modell, nämlich die Pachinko Maschine, verhält.

3.5.2 Einschränkung der zusätzlichen Trainingsinstanzen

Eine mögliche Variation des vorgestellten Ansatzes ergibt sich durch Definition einer Hürde, welche die Menge der zusätzlichen Trainingsinstanzen einschränkt, wodurch die Komplexität der Unterscheidungsprobleme verringert werden kann. Betrachten wir erneut den Klassifizierer $K_{A,B}$. Denkbar ist ein Mindestwert für die hierarchischen Nähe zwischen den Klassen der zusätzlichen Instanzen und A, bzw. B. So werden die Instanzen der Klasse X nur dann als zusätzliche positive Beispiele verwendet, wenn neben den ursprünglichen Voraussetzungen beispielsweise zusätzlich auch $V(X, A) \geq 2$ gilt.[10] Eine andere Möglichkeit mit ähnlicher Intention wäre es nur die jeweils ersten n Klassen zu berücksichtigen, die hierarchisch am nächsten zu A sind.

Statt einer Einschränkung der Trainingsinstanzen kann auch eine Gewichtung vorgenommen werden, sofern dies vom Basisklassifizierer unterstützt wird. Das Gewicht der Instanzen würde sich daran orientieren, wie nahe sich die Klassen in der Hierarchie sind. So könnte man den Instanzen von A das höchste Gewicht zuordnen und dann für die weiteren Instanzen ein absteigendes Gewicht, umso weiter deren Klassen in der Hierarchie von A entfernt sind.

[10] Es ist anzumerken, dass das Maß der Verwandtschaft sich hierbei als Kriterium nicht eignen würde, da es wie unter 3.3.1 erklärt, Nähe zwischen zwei Klassen nicht in einem absoluten Sinne angibt. Es wäre also für diese Variation ein anderes Maß für hierarchische Nähe zu benutzen.

3.5.3 Selektive Anreicherung der Trainingsinstanzen

Wie wir unter 3.4 feststellen konnten, kann man sich sowohl Fälle vorstellen in denen ein paarweiser Klassifizierer von zusätzlichen Trainingsinstanzen hierarchienaher Klassen profitiert, als auch Fälle in denen diese Anreicherung das Unterscheidungsproblem unnötig komplexer macht und so das Lernen eines gut generalisierten Modells erschwert. Basierend auf diesen Gedanken wäre es wünschenswert, wenn man in letzteren Fälle die paarweise Unterscheidung mit „normaler“ Trainingsmenge lernt und die paarweise hierarchische Klassifikation nur bei Bedarf verwendet. Die eigentliche Aufgabe bei einem solchen zweigeteilten Vorgehen ist die Wahl von Kriterien, an welchen man für den einzelnen Fall zwischen den beiden Methoden unterscheiden kann.

Betrachten wir den Klassifizierer $K_{A,B}$ und die Mengen T_A und T_B, welche jeweils die Trainingsinstanzen der beiden Klassen enthalten. Ein denkbares Kriterium, um zu entscheiden, ob zusätzliche Trainingsinstanzen im Sinne der paarweisen hierarchischen Klassifikation verwendet werden sollten, ist das Mengenverhältnis von T_A und T_B. Falls die Aussage

$$C \geq \frac{|T_A|}{|T_B|} \geq \frac{1}{C}$$

mit einer zu wählenden Konstante C gilt, dann würde der Klassifizierer nur mit T_A und T_B lernen, andererseits würden nach dem hierarchischen Ansatz Trainingsinstanzen hinzugefügt. Denn gilt die obige Aussage nicht, dann haben wir ein unbalanciertes Verhältnis der Instanzen der beiden Klassen, was bedeuten kann, dass eine der Klassen eventuell zu wenige Instanzen hat, um ein gutes Unterscheidungskonzept zu lernen. Zusätzlich könnten die Trainingsmengen bei Basislernern, welche anfällig dafür sind bei unbalancierten Trainingsmengen die besser vertretene Klasse zu bevorzugen, zu Problemen führen. Dieses „relative“ Kriterium sollte sinnvollerweise noch mit einem „absoluten“ Kriterium der Form

$$|T_A| \geq C' \wedge |T_B| \geq C'$$

ergänzt werden, damit auch Fälle, wie zum Beispiel $|T_A| = |T_B| = 1$ [11], erfasst werden, bei denen einer der beiden Klassen höchstwahrscheinlich durch ihre wenigen Trainingsinstanzen unterrepräsentiert[12] ist.

[11] Dieser Fall würde das „relative“ Kriterium bestehen

[12] Unterrepräsentiert in dem Sinne, dass die Charakteristik der Klasse nur unzureichend von den vorhandenen Instanzen beschrieben werden kann

4 Äquivalenz zwischen paarweiser hierarchischer Klassifikation und der Pachinko Maschine

In diesem Kapitel soll zunächst das Konzept der von Koller und Sahami vorgestellten Pachinko Maschine beschrieben werden. Die Pachinko Maschine ist, ebenso wie die paarweise hierarchische Klassifikation, welche im Kapitel 3 vorgestellt wird, eine Methode mit der Single Label Probleme auf hierarchischen Daten gelöst werden können. Im Anschluss soll die unter bestimmten Bedingungen vorhandene Äquivalenz der Pachinko Maschine mit dem Modell der paarweisen hierarchischen Klassifikation formal aufgezeigt werden.

4.1 Pachinko Maschine

Die Idee der Pachinko Maschine wurde in „Hierarchically classifying documents using very few words“ [Koller and Sahami, 1997] von Daphne Koller und Mehran Sahami beschrieben. Das Ziel der Pachinko Maschine ist es Instanzen eines Datensatzes unter Ausnutzung der Klassenhierarchie zu kategorisieren.
Betrachten wir zur Erklärung des Prinzips die Blatthierarchie in Abbildung 4.1. Eine Instanz soll einer der fünf Klassen A,B,C,D,E zugeordnet werden: zunächst entscheidet der im Bild oberste Klassifizierer ob diese Instanz eher einer der beiden Klassen A,B oder einer der drei Klassen C,D,E entspricht. Ist es die Einschätzung des Klassifizierers, dass die Instanz eher der Gruppe C,D,E zugeordnet ist, dann entscheidet anschließend ein weiterer Klassifizierer darüber ob die Instanz zur Klasse C oder zur den Klassen D,E gehört. Wird sich für den Zweig mit D,E entschieden, dann wird schlussendlich der Klassifizierer befragt, welcher Instanzen entweder die Klasse D oder E zuordnet.

Bildlich gesehen wird für jede Instanz ein Weg im Baum gegangen, angefangen von der Wurzel bis hin zu einem Blatt, wobei dieses Blatt dann der vorhergesagten Klasse entspricht. An jedem internen Knoten des Baumes kann man sich einen Klassifizierer vorstellen, welcher über den weiteren Weg entscheidet. Das bedeutet weiter, dass eine Pachinko Maschine genau so viele Klassifizierer erlernen muss wie es Abzweigungen im Hierarchiebaum gibt. Jeder Klassifizierer entscheidet für jede Instanz also stets zwischen den Mengen der Klassen in den Teilbäumen seiner Abzweigung. Um diese Entscheidung lernen zu können werden die Trainingsinstanzen aller Klassen in den entsprechenden Teilbäumen betrachtet. Später werden wir sehen, dass einige Klassifizierer der paarweisen hierarchischen Klassifikation diese selbe Trainingsmenge verwenden.

In der oben angeführten Arbeit [Koller and Sahami, 1997] wurde das Modell der Pachinko Maschine auf mehreren Datensätzen getestet, welche allesamt auf Daten des Nachrichtentextarchives Reuters 22173[1] basieren. Es geht also konkret um das Problem der Textklassifikation. Die Instanzen sind durch eine Menge von dichotomen Attributen dargestellt; jedes Attribut beschreibt das Vorhandensein oder

[1] http://www.daviddlewis.com/resources/testcollections/reuters21578/

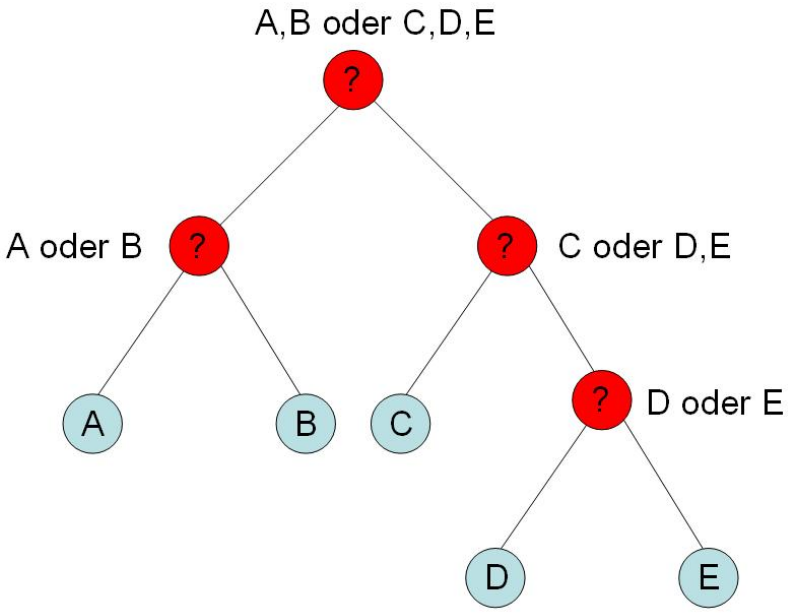

Abbildung 4.1: Klassifizierer der Pachinko Maschine im Hierarchiebaum
(?) zeigt die bildliche Position der Klassifizierer an

Nichtvorhandensein eines Wortes in dem Nachrichtentext. Entsprechend ist die Anzahl der Attribute sehr groß. Die Klassenhierarchien sind allesamt 2-stufig und mit maximal 8 Klassen aufgebaut.

Die Idee des Pachinko Modells basiert wesentlich auf den folgenden Grundgedanken. Da man von den Textdokumenten erwarten kann, dass diese umso mehr Gemeinsamkeiten haben, desto näher sie sich in der Hierarchie sind, sollte die Unterscheidung von weit auseinander liegenden Texten relativ einfach sein. Das heißt, selbst wenn es nicht klar ist, zu welchem Thema ein Text genau gehört, dann kann man trotzdem gut entscheiden, ob dieser eher aus dem Bereich $Kunst$ oder aus dem der $Politik$ kommt. Das Vorhandensein eines Wortes wie $Atelier$ würde das Dokument beispielsweise relativ eindeutig der $Kunst$ zuordnen, die genau Klasse, wie $Moderne\ Kunst$ oder $Klassische\ Kunst$ ist jedoch damit nicht entschieden. Daher kann eine solche Entscheidung, die bildlich in einem der höheren internen Knoten der Hierarchie gelöst werden muss, über das Vorhanden- oder Nichtvorhandensein von nur sehr wenigen signifikanten Wörtern gefällt werden.
Gleichzeitig sind es ebenfalls nur wenige Worte, die für die Entscheidung zwischen zwei relativ nahen Klassen ausreichen, weil diese viele Worte entweder gemeinsam besitzen oder gemeinsam nicht besitzen. So sind die Worte $Atelier$ und $Bundeskanzler$ für die Unterscheidung zwischen $Moderne\ Kunst$ und $Klassische\ Kunst$ unbedeutend. An Hand dieser Annahmen, kann man folgern, dass an jeder Abzweigung im Hierarchiebaum theoretisch nur ein relativ einfaches Entscheidungsproblem unter der Berücksichtigung von nur sehr wenigen Attributen gelöst werden muss. Entsprechend wird von Koller und Sahami für jeden Klassifizierer eine sehr umfassende „Feature Selection“[2] angewendet.

Im Vergleich zur paarweisen Klassifikation hat das Pachinko Modell den offensichtlichen Vorteil, dass sehr viel weniger Klassifizierer erlernt werden müssen und von diesen bei der Klassifikation einer Instanz auch nur ein Teil benutzt wird. Die Anzahl der Klassifizierer entspricht der Anzahl der internen Knoten des Hierarchiebaumes und für die Klassifikation einer einzelnen Instanz sind nur die Vorhersagen der Klassifizierer, welche auf Weg von der Wurzel zum entsprechenden Blatt liegen, notwendig.

[2] Feature Selection ist der Vorgang nur eine Menge aller Attribute der Instanzen auszuwählen, die dann für das Lernen und Klassifizierern verwendet werden.

Allerdings muss jeder, der bei einer Klassifikation verwendeten, Klassifizierer korrekt entscheiden, da sonst bildlich der falsche Weg im Baum gewählt wird und so die korrekte Klasse nicht mehr erreicht werden kann. Je höher im Hierarchiebaum ein solcher möglicher Fehler passiert, desto weiter wird die vorhergesagte Klasse hierarchisch von der korrekten Klasse entfernt sein.

Das Modell ist bedingt durch seinen Aufbau nicht in der Lage Multi Label Probleme zu lösen, da der bildlich gegangene Weg im Hierarchiebaum nur bei einen Blatt enden kann.

4.2 Äquivalenz zum Pachinko Modell

In diesem Abschnitt soll die Äquivalenz der paarweisen hierarchischen Klassifikation, so wie sie in Kapitel 3 definiert ist, mit dem Pachinko Modell von Koller und Sahami untersucht werden. Es wird aufgezeigt werden, dass unter bestimmten Bedingungen der Klassenhierarchie beide Methoden äquivalent arbeiten.

Betrachten wir nun die paarweise hierarchische Klassifikation für den Fall eines Single Label Problems. Es soll im Folgenden aufgezeigt werden, dass bei der paarweisen hierarchischen Klassifikation dieselben Klassifizierer erlernt werden wie bei der Pachinko Maschine. Weiter soll gezeigt werden, dass diese Klassifizierer auch dieselbe Funktion, wie der entsprechende Gegenpart der Pachinko Maschine, ausüben, nämlich die bildliche Entscheidung, welcher Weg an einem internen Knoten der Klassenhierarchie eingeschlagen werden soll.

Dafür müssen wir uns zunächst einer Eigenschaft des paarweisen hierarchischen Ansatzes klar werden. In der Tabelle 3.1 des Kapitels 3 „Paarweise hierarchische Klassifikation“ wurde bereits deutlich, dass viele der paarweisen Basisklassifizierer die genau gleiche Menge der positiven und negativen Trainingsinstanzen verwenden, so dass alle diese Basisklassifizierer schließlich genau gleich sind. Betrachten wir diese Fälle genauer.

Lemma A: *Zwei Basisklassifizierer K_{A_1,B_1} und K_{A_2,B_2} der paarweisen hierarchischen Klassifikation haben die gleichen Trainingsdaten, wenn die beiden Klassen A_1 und A_2 in einem gemeinsamen Teilbaum der Hierarchie liegen und B_1 und B_2 gemeinsam in einem anderen Teilbaum liegen. Ihre Trainingsdaten sind jeweils die Instanzen aller Klassen aus den beiden Teilbäumen. Genauer sind die positiven Beispiele für diese Klassifizierer alle Instanzen der Klassen des Teilbaumes von A_1,A_2 und die negativen Beispiele sind die aus dem Teilbaum von B_1,B_2.*

Beweis: Da A_1 und A_2 in einem anderen Teilbaum liegen als B_1 und B_2 muss es im Hierarchiebaum einen internen Knoten geben, welcher die beiden Teilbäume trennt. Bis zu diesem Knoten haben alle Klassen der beiden Teilbäume einen gemeinsamen Weg im Baum. Alle Klassen, welche im Teilbaum von A_1,A_2 liegen, haben aber einen längeren gemeinsamen Weg mit jeweils A_1,A_2 als mit B_1,B_2, weil ihr Weg an dem trennenden Knoten in Richtung A_1,A_2 geht. Entsprechend haben allen Klassen, die im Teilbaum von B_1,B_2 liegen, einen längeren gemeinsamen Weg mit B_1,B_2 als mit A_1,A_2. Ein Basisklassifizierer der Form $K_{A,B}$ erhält als positive Beispiele alle jene Instanzen, deren Klasse entweder A ist oder eine höhere Verwandtschaft[3] mit A als mit B hat. Als negative Beispiele werden entsprechend, neben allen Instanzen der Klasse B, die Instanzen verwendet, deren Klassen eine höhere Verwandtschaft mit B als mit A haben. So lässt sich trivial erkennen, dass alle Klassifizierer, welche zwischen zwei Klassen aus den zwei selben Teilbäumen unterscheiden, dieselben Trainingsdaten erhalten. Und zwar bestehen die Trainingsdaten aus allen Instanzen der Klassen, die in den beiden Teilbäumen liegen. □

[3] Das Maß Verwandtschaft wurde unter 3.3.1 definiert und entspricht dem gemeinsamen Weg von zwei Knoten.

Lemma B: *Der Pachinko Klassifizierer eines internen Knotens mit zwei Teilbäumen A, B entspricht den Basisklassifizierern K_{A_x,B_x}, welche bei der paarweisen hierarchischen Klassifikation für die Unterscheidung zwischen zwei Klassen aus den beiden Teilbäumen A und B erlernt werden.*

Beweis: Mit dem Wissen über die in **Lemma A** beschriebene Eigenschaft können wir nun einen internen Knoten der Klassenhierarchie betrachten und feststellen, dass es mehrere[4] Basisklassifizierer der paarweisen hierarchischen Klassifikation gibt, welche die gesamten Instanzen der beiden Teilbäume als Trainingsmenge verwenden. Effektiv hat also jeder dieser Klassifizierer die Entscheidung erlernt, ob eine Instanz eher zu den Klassen des einen oder des anderen Teilbaumes gehört. Damit treffen diese Basisklassifizierer dieselbe Unterscheidung wie der, dem internen Knoten entsprechende, Pachinko Klassifizierer, da beide Klassifizierer ihr Unterscheidungsmodell mit Hilfe derselben Trainingsinstanzen erlernt haben. □

Nun wird bei der paarweisen hierarchischen Klassifikation die vorhergesagte Klasse über ein Voting entschieden, statt wie bei der Pachinko Maschine über ein bildliches Ablaufes des Weges in der Hierarchie. Es bleibt nun also noch zu zeigen, dass diese beiden unterschiedlichen Bestimmungsmethoden, unter den bereits gezeigten Umständen, auch äquivalent sind:

Theorem A: *Die paarweise hierarchische Klassifikation in ihrer in Kapitel 3 definierten Form mit dichotomen Basisklassifizierern und normalem Voting als Dekodierungsmethode arbeitet auf Datensätzen mit binären Hierarchiebäumen äquivalent zu dem Modell der Pachinko Maschine.*

Beweis: Beginnen wir bei der Wurzel des Hierarchiebaumes und betrachten dort die zwei Teilbäume: A und B. Der eine Teilbaum habe insgesamt a Klassen: $(A_1, A_2, \ldots, A_a)$; der zweite b Klassen: $(B_1, B_2, \ldots, B_b)$. Der Baum hat insgesamt n Klassen und es gilt offensichtlich $a + b = n$. Die Klassifizierer, welche zwischen zwei Klassen aus jeweils einem der beiden Teilbäume unterscheiden, sind nach **Lemma A** allesamt genau gleich und produzieren immer dasselbe Ergebnis, also entweder A_x oder B_x. Es gibt insgesamt $a \cdot b$ Klassifizierer der Form K_{A_x,B_x}.
Bei der Pachinko Maschine entscheidet an dieser Stelle ein Klassifizierer ob eine Instanz eher zu einer Klasse des ersten oder des zweiten Teilbaumes gehört. Die Menge der Basisklassifizierer K_{A_x,B_x} tut laut **Lemma B** faktisch dasselbe. Angenommen das Ergebnis dieser Klassifizierer ist jeweils A_x, dann gilt für jede Klasse des Teilbaumes A, dass sie bereits b Punkte im Voting hat, denn jede Klasse aus A gewinnt bei jedem der b Vergleiche mit den Klassen aus dem Teilbaum B. Für die Klassen des Teilbaumes B gilt, dass sie selber nun nur noch maximal $b - 1$ Punkte im Voting erreichen können, nämlich nur von den Klassifizierern der Form K_{B_x,B_y}, da alle anderen Entscheidungen verloren werden. Das bedeutet, dass durch die einstimmige Entscheidung der Klassifizierer K_{A_x,B_x} für A_x alle Klassen des Teilbaumes A im Ranking vor denen des Teilbaumes B liegen. Die Entscheidung welche Klasse der Top Rank ist, wird nur noch unter den Klassen des Teilbaumes A ausgetragen. Trivial gilt gleiches nach derselben Logik auch für den anderen Teilbaum, falls K_{A_x,B_x} stets B_x vorhersagt. Damit wurde die Suche nach der relevanten Klasse effektiv auf einen der beiden Teilbäume beschränkt, genauso wie es der Pachinko Klassifizierer an der Wurzel des Baumes tut.
Betrachtet man nun den Teilbaum A dann lässt sich dieser wiederum am nächsten Knoten in zwei Teilbäume aufteilen und alle Basisklassifizierer, welche zwei Klassen aus diesen beiden Teilbäumen unterscheiden, erfüllen hier dieselbe Funktion, wie es der Pachinko Klassifizierer dieses Knotens tut. Wir wissen bereits, dass die Klassen der beiden nun betrachteten Teilbäume im Ranking vor allen anderen Klassen liegen. Die Menge der Klassifizierer, welche eine Klasse des einen Teilbaumes mit einer Klasse des anderen Teilbaumes vergleichen, entscheidet an dieser Stelle wieder, nach der gleichen Logik wie vorher, die Klassen welches Teilbaumes kollektiv vor den Klassen des anderen Teilbaumes im Ranking liegen werden. Dies lässt sich fortführen bis zu einem internen Knoten, welcher nur zwei

[4]Für den Fall eines internen Knotens dem nur genau zwei Blätter folgen, gibt es nur einen Basisklassifizierer.

Blätter als Nachfolger hat. An dieser Stelle existiert nur ein Klassifizierer, welcher dann zwischen den beiden Klassen der Blätter die vorhergesagte Klasse wählt. □

Es bleibt nun noch der Fall zu beleuchten, bei dem der Hierarchiebaum interne Knoten besitzt, aus welchen mehr als nur zwei Teilbäume hervorgehen. Für die Pachinko Maschine ist diese Unterscheidung unerheblich, da die einzelnen Klassifizierer dem Modell nach problemlos auch nicht binäre Entscheidungen treffen können.[5] Bei der paarweisen hierarchischen Klassifikation sind alle Basisklassifizierer jedoch binäre Klassifizierer.

Theorem B: *Die paarweise hierarchische Klassifikation in ihrer in Kapitel 3 definierten Form mit dichotomen Basisklassifizierern und normalem Voting als Dekodierungsmethode arbeitet auf Datensätzen mit beliebigen Hierarchiebäumen nur genau dann äquivalent zu dem Modell der Pachinko Maschine wenn die Menge der Basisklassifizierer, welche für einen internen Knoten im Hierarchiebaum zwischen je zwei Klassen aus verschiedenen Teilbäumen unterscheidet, für eine gegebene stets Instanz* eindeutig[6] *entscheidet.*

Beweis: Betrachten wir die Wurzel des Hierarchiebaumes von welcher n Teilbäume $(n_1, n_2 \dots n_n)$ ausgehen. Der Teilbaum n_i habe jeweils m_i Klassen. Für jedes Paar an Teilbäumen (n_i, n_j) existieren $m_i \cdot m_j$ Klassifizierer der Form K_{i_x,j_x} wobei i_x eine Klasse des Teilbaumes n_i und j_x eine Klasse des Teilbaumes n_j ist.
Analog zu dem Vorgehen des Beweises von **Theorem A** und ebenfalls aufbauend auf den **Lemmata A** und **B** werden wir nun zunächst nach einer Menge an Klassifizierern suchen, welche dieselbe Trainingsmenge wie der entsprechende Pachinkoklassifizierer besitzen und dann zeigen, dass diese Klassifizierer beim Voting dieselbe Funktion wie der Pachinkoklassifizierer erfüllen, nämlich die bildliche Entscheidung, in welchem Teilbaum die relevante Klasse liegt. Da es offensichtlich nicht genau einen binären Basisklassifizierer gibt, welcher die Unterscheidung zwischen mehreren Teilbäumen leisten kann, betrachten wir eine Art Metaklassifizierer, welcher aus den Klassifizierern der Form K_{i_x,j_x} mit $i, j \in (1, 2 \dots n)$ und $i \neq j$ besteht. Es gibt $\frac{n(n-1)}{2}$ verschiedene Klassifizierer dieser Art. Da diese Klassifizierer jeweils alle Trainingsinstanzen der jeweiligen beiden Teilbäume zum Erlernen ihres Unterscheidungsmodelles benutzen, besitzt dieser Metaklassifizierer dieselbe Trainingsmenge wie der entsprechende Pachinkoklassifizierer.
Es ist nun zu zeigen, dass dieser Metaklassifzierer beim Vorgang des Votings die Funktion erfüllt, den Teilbaum der relevanten Klasse zu bestimmen, was beim Voting bedeutet, dass alle Klassen des ausgewählten Teilbaumes im Ranking vor den Klassen aller anderen Teilbäume liegen sollen. Nehmen wir an für einen beliebigen Teilbaum n_i gilt, dass die Klassifizierer der Form K_{i_x,j_x} für alle $j \in (1, 2 \dots n), j \neq i$ jeweils die Klasse i_x voraussagen. Diesen Fall bezeichnen wir als eine *eindeutige* Entscheidung. Dies hat zur Folge, dass jede Klasse des Teilbaumes n_i bereits $\sum_{k=1,k\neq i}^{n} m_k$ Stimmen im Voting erreicht. Gleichzeitig ergibt sich, dass eine beliebige Klasse j_x eines beliebigen Teilbaumes n_j selber nur noch maximal $(m_j - 1) + \sum_{k=1,k\neq i,k\neq j}^{n} m_k$ Stimmen erreichen kann[7]. $(m_j - 1)$ ist die maximale Anzahl der Stimmen die eine Klasse des Teilbaumes n_j aus den „internen“ Vergleichen mit Klassen des eigenen Teilbaumes gewinnen kann. Diese Gesamtstimmanzahl ist offensichtlich stets geringer als die Anzahl der Stimmen, welche jede Klasse des Teilbaumes n_i mindestens hat. Damit

[5] In der Arbeit von Koller und Sahami [Koller and Sahami, 1997] wurde unter anderem Naive Bayes als Basisklassifizierer verwendet.

[6] Unter einer *eindeutigen* Entscheidung verstehen wir an dieser Stelle, dass eine Menge an Klassifizierern, welche Klassen eines Teilbaumes A mit den Klassen aller anderer Teilbäume vergleicht, stets die jeweilige Klasse des Teilbaumes A vorhersagt.

[7] Die angeführte Summe entspricht dem Fall, dass die Klassifizierer K_{j_x,h_x} für alle $h \in (1, 2 \dots n), h \neq i, h \neq j$ jeweils die Klasse j_x voraussagen. Dies ist der bestmöglichste Fall für den Teilbaum n_j.

liegen alle Klassen des Teilbaumes n_i im Ranking über allen anderen Klassen der anderen Teilbäume. Dies bedeutet weiter, dass die Suche nach der relevanten Klasse nun auf Klassen des Teilbaumes n_i beschränkt ist. Damit erfüllt der Metaklassifizierer dieselbe Funktion wie der entsprechende Pachinkoklassifizierer. Nach derselben Logik entscheiden auf den tieferen Stufen des Hierarchiebaumes wieder die, dem nächsten internen Knoten im ausgewählten Teilbaum zugeordneten, Klassifizierer die Klassen welches Teilbaumes kollektiv vor den Klassen aller anderen Teilbäume im Ranking liegen. So lässt sich dies fortführen bis zu einem Blattknoten, welcher dann die vorhergesagte Klasse darstellt.
Damit wurde die Äquivalenz zum Modell der Pachinko Maschine für die Bedingung, dass der Metaklassifizierer *eindeutig* entscheidet, aufgezeigt. Nun ist es noch notwendig zu zeigen, dass diese Äquivalenz bei nicht *eindeutigen* Entscheidungen nicht zwingend ist.
Dazu nehmen wir an, dass die Klassifizierer der Form K_{i_x,j_x} für alle $j \in (1, 2 \dots n), j \neq i$ *und* $j \neq h$ jeweils die Klasse i_x voraussagen. Es liegt keine *eindeutige* Entscheidung vor, da die Klassifizierer der Form K_{i_x,h_x} stets die Klasse h_x vorhersagen. Es lässt sich nun einfach an einem pathologischen Fall zeigen, dass unter diesen Bedingungen nicht mehr garantiert ist, dass alle Klassen des Teilbaumes n_i im Ranking vor allen Klassen der anderen Teilbäume liegen. Nehmen wir dafür an, die Anzahl m_h der Klassen des Teilbaumes n_h sei deutlich größer als die Anzahl der Stimmen, welche jede Klasse i_x aus dem Teilbaum n_i bereits durch die oben angeführten Entscheidungen der Klassifizierer K_{i_x,j_x} besitzt. Nun ist es möglich, dass eine Klasse h_x durch Vergleiche der Klassifizierer der Form K_{h_x,h_y} bis zu $(m_h - 1)$ zusätzliche Stimmen erhält, was diese Klasse im Ranking über Klassen des Teilbaumes n_i stellen könnte. Es wird also klar, dass alleine durch die Auswertung des Metaklassifizierers des betrachteten internen Knotens nicht eindeutig der Teilbaum der relevanten Klasse entschieden werden kann, sondern dass für diese Entscheidung das Wissen über die Ergebnisse der „internen“ Klassifizierer, welche zwischen Klassen eines gemeinsamen Teilbaumes entscheiden, nötig ist. Damit kann der Metaklassifizierer in diesen Fällen nicht die Funktion des entsprechenden Pachinkoklassifzierers erfüllen und die beiden Modelle arbeiten nicht äquivalent. □

Betrachten wir zu diesen Überlegungen zwei Beispiele. Nehmen wir den Fall von drei Teilbäumen A,B und C an der Wurzel des Baumes an, welche jeweils Klassen der Form A_x, B_x und C_x enthalten.

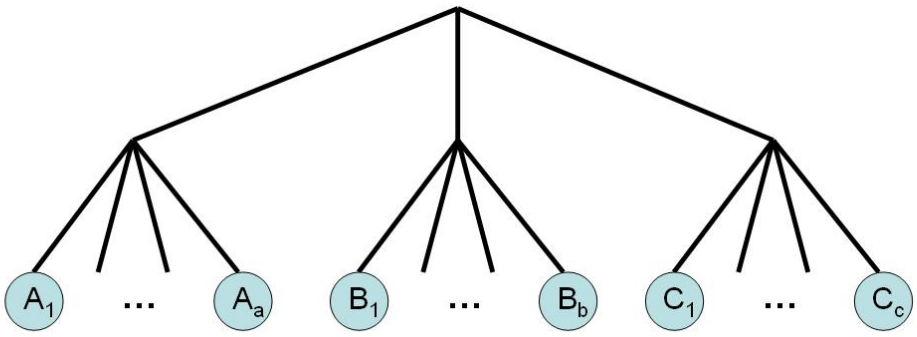

Abbildung 4.2: Beispiel eines Hierarchiebaumes mit drei Teilbäumen an der Wurzel

Unser Metaklassifizierer besteht aus den binären Klassifizierern der Form K_{A_x,B_x}, K_{A_x,C_x} und K_{B_x,C_x}. Dieser Metaklassifizierer hat nun dieselbe Trainingsmenge wie der Pachinko Klassifizierer des entsprechenden internen Knotens. Gelte nun für eine Instanz i die *eindeutige* Entscheidung $K_{A_x,B_x}(i) = A_x$, $K_{A_x,C_x}(i) = A_x$ und $K_{B_x,C_x}(i) = B_x$, dann folgt, dass alle Klassen von A bereits $b + c$ Punkte im Voting gesichert haben; b und c seien die jeweilige Anzahl der Klassen der Teilbäume B und C. Eine Klasse des Teilbaumes B dagegen kann nur maximal $c + (b - 1)$ Punkte erreichen (c aus den gewonnen Vergleichen mit C_x und $b-1$ aus Vergleichen mit Klassen des eigenen Teilbaumes).

Klassen des Teilbaumes C kommen nur auf maximal $c-1$ Punkte. Es folgt, dass alle Klassen von A im Ranking über denen von B und C liegen.
Betrachten wir nun den Fall, bei dem keiner der Teilbäume alle seine Vergleiche gewinnt und damit also keine *eindeutige* Entscheidung vorliegt. Gelte nun $K_{A_x,B_x}(i) = A_x$, $K_{A_x,C_x}(i) = C_x$ und $K_{B_x,C_x}(i) = B_x$ mit einer Klassenanzahl von $a = 5$, $b = 3$ und $c = 2$. Es lässt sich zwar erkennen, dass die Klassen der Form A_x mindestens $b = 3$ Stimmen gesichert haben und die Klassen C_x mindestens $a = 5$ Stimmen haben, aber die konkrete Rangfolge dieser Klassen im Ranking hängt in diesem Fall davon ab, wie die Klassifizierer innerhalb der Teilbäume entscheiden (Klassifizierer der Form K_{A_x,A_y} und K_{C_x,C_y}). Gewinnt beispielsweise eine Klasse A_x alle Vergleiche mit den anderen Klassen des Teilbaumes A, dann hat diese Klasse $b + (a-1) = 3 + 4 = 7$ Stimmen während eine der Klassen aus C maximal $a + (c-1) = 5 + 1 = 6$ Stimmen erhalten kann. Der beschriebene Metaklassifizierer weiß aber nicht wie die genaue Stimmenverteilung aussehen wird und daher kann dieser Metaklassifizierer in diesem Fall nicht die Aufgabe, den Teilbaum des Top Rank zu bestimmen, erfüllen.

5 Versuche und Ergebnisse

Die paarweise hierarchische Klassifikation wurde in verschiedenen Kontexten und auf verschiedenen Datensätzen in ihrer Performanz mit der paarweisen Klassifikation verglichen. Die durchgeführten Versuche, die Ergebnisse und deren Interpretation werden in diesem Kapitel vorgestellt. Es wird die Anwendung der Methode auf Daten des Reuters Corpus Volume 1 Datensatzes, sowie auf speziell generierten künstlichen Datensätzen untersucht. Abschließend wird eine Methode für die Messung der „Hierarchietreue“ von Daten vorgestellt und auch auf den Datensätzen angewendet.

Die Daten des Reuters Datensatzes bieten eine gute Testplattform für das Verhalten der Klassifizierer auf realen Daten, wie sie in dieser und ähnlicher Form in konkreten Anwendungen vorkommen können. Da es sich um Multi Label Daten, eine große Knotenhierarchie auf den Klassen und Instanzen mit sehr vielen Attributen handelt, sind die Ansprüche an einen Klassifizierer hierbei relativ hoch.
Bei den künstlichen Datensätzen wird sich auf eine im Gegensatz deutliche einfachere Problemstellung konzentriert. Es handelt sich um Single Label Datensätze mit einer Blatthierarchie und nur zwei Attributen pro Instanz. Hierbei steht im Vordergrund, die konkreten Unterschiede der beiden Klassifikationsverfahren zu beobachten und durch den einfachen Aufbau eher die Möglichkeit zu haben, die Ergebnisse zielgerichtet auf die Unterschiede der Klassifikationsverfahren hin zu bewerten.
Es sei an dieser Stelle auch gesagt, dass bei den Versuchen dieses Kapitels nicht die absoluten Ergebnisse der Klassifikationen interessant sind, sondern der Vergleich dieser beiden Klassifikations-, bzw. Binarisierungsmethoden. In anderen Arbeiten, wie zum Beispiel [Fürnkranz, 2001] und
[Hsu and Lin, 2002], wurde bereits die gute Performanz der paarweisen Klassifikation im Vergleich zu anderen Methoden, wie dem One-against-All Verfahren aufgezeigt; unser primäres Ziel hier ist es dagegen nur zu untersuchen, ob die paarweise hierarchische Klassifikation in bestimmten Szenarien einen Vorteil haben kann.

5.1 Künstliche Datensätze

5.1.1 Einleitung

Für den Vergleich der paarweisen hierarchischen Klassifikation mit seinem nicht-hierarchischen Gegenpart, der paarweisen Klassifikation, wurden künstliche Datensätze generiert. Die Gründe dafür liegen zum einem darin, dass kaum hierarchische Single Label Datensätze, die sich für diesen Vergleich eignen würden, gefunden wurden. Zum Zweiten bietet die Möglichkeit, einen eigenen Datensatz zu erstellen, viele Vorteile. Durch diese Vorgehen ist es möglich die Eigenschaften der Daten direkt und zielgerichtet zu manipulieren, so dass die Ergebnisse der Klassifizierer unten den verschiedenen gewünschten Umständen beobachtet werden können. Es ist dabei insbesondere vorteilhaft, dass festgelegt werden kann, in welchem Umfang die Daten der gegebenen Hierarchie entsprechen. Auf diese Weise kann der hierarchische Ansatz unter den theoretischen Idealbedingungen getestet werden, welche man bei realen Datensätzen so gut wie nie auffindet.

5.1.2 Aufbau und Generierung der Datensätze

Die dem Datensatz zugrunde liegende Klassenhierarchie in Abbildung 5.1 ist eine 4-stufige Blatthierarchie und umfasst 12 Klassen. Da jeder Instanz genau eine der Klassen zugeordnet wird, handelt es sich um ein Single Label Problem. Es wurden bewusst die für eine Klassifikation „einfachsten“ Konstellationen gewählt, um so die Ergebnisse des Vergleiches von den Einflüssen zusätzlicher „Schwierigkeiten“, wie es Multi Label Probleme und Knotenhierarchien sind, freizuhalten.

Jede generierte Instanz umfasst genau 2 Attribute, welche reale Werte enthalten. Mit dieser Struktur ist es möglich die Instanzen als Punkte in einem Koordinatensystem darzustellen, indem die Werte der Attribute als Koordinaten interpretiert werden.

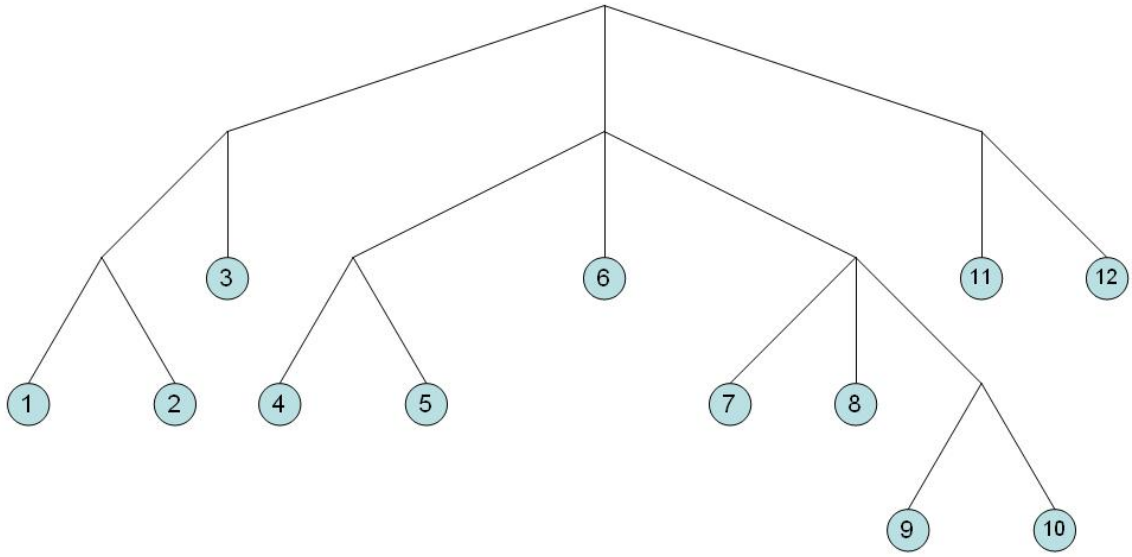

Abbildung 5.1: Klassenhierarchie der künstlichen Datensätze

Bei der Generierung der Instanzen stellt sich zunächst die Frage, wie die Vorgaben der Hierarchie umgesetzt werden sollen. Das Konzept der paarweisen hierarchischen Klassifikation basiert darauf, dass eine aus der Hierarchie abgeleitete Aussage der Form „Klasse A steht der Klasse B in der Hierarchie näher als der Klasse C“ auch bedingt, dass auch in den konkreten Daten eine höhere Gemeinsamkeit zwischen den Instanzen der Klasse A und denen der Klasse B besteht als zwischen Instanzen von A und C. Entsprechend können wir nur dann mit sinnvollen[1] Ergebnissen rechnen, wenn diese Eigenschaft zu gewissem Grad bei unserem Datensatz erfüllt ist. Da wir die Instanzen durch ihre zwei Attribute Punkten im 2-dimensionalen Raum entsprechen, wurde sich bei der Generierung der Instanzen an der euklidschen Distanz orientiert um die hierarchische Struktur wiederzugeben. Das heißt wir wollen, dass die hierarchische Nähe zweier Klassen sich auch in eine geringe euklidsche Distanz zwischen den Instanzen dieser Klassen übersetzt. Instanzen der Klasse A sollen zu denen der Klasse B also eine geringere Distanz haben als zu denen der Klasse C genau dann wenn A hierarchisch näher an B als an C liegt.

Es soll an dieser Stelle festgehalten werden, dass die Wahl der euklidschen Distanz für die Umsetzung von Hierarchietreue in Daten nur eine von vielen denkbaren Möglichkeiten für diesen Zweck

[1] „Sinnvoll“ bezogen auf unsere vorherigen theoretische Überlegungen

ist. Inwiefern eine Methode für diese Aufgabe geeignet ist, kann von vielen Gegebenheiten abhängen, insbesondere sind der konkrete Datensatz und die Semantik der einzelnen Attribute entscheidend. In diesem Fall jedoch scheint der Abstand der Instanzen im Raum als Maß durchaus sinnvoll und zusätzlich für die Bearbeitung und Erklärung anschaulich.

Einige Abbildungen in diesem Abschnitt wurden mit Hilfe des WEKA Explorers[2] erstellt, welcher die Funktion bietet Instanzen im 2-dimensionalen Raum darzustellen.

Insgesamt wurden zwei Typen von Datensätzen erstellt, bei denen dann jeweils Variationen vorgenommen wurden, um das Verhalten der beiden Klassifikationsmethoden in speziellen Fällen zu untersuchen. Bei den ersten Typ wurden in einem 2-dimensionalen Raum der Größe $x \in [0; 2300]$ und $y \in [0; 1000]$ mit einem rekursiven Algorithmus jeder Klasse an Hand ihrer Position in der Hierarchie ein Rechteck zugeteilt. Das Ergebnis ist in Abbildung 5.2 zu sehen. Die Instanzen jeder Klasse wurden dann nach gaußscher Normalverteilung um den Mittelpunkt des zugewiesenen Rechteckes verteilt. Das heißt, dass für jede Instanz auf jeweils beide Koordinaten des Mittelpunktes ein Zufallswert mit Mittelwert 0 und einer bestimmten Standardabweichung aufaddiert wurde. Diese Zufallswerte können auch negativ sein. Die Standardabweichung in beiden Dimensionen wurde bei verschiedenen Datensätzen variiert; es sind jedoch immer Bruchteile, bzw. Vielfache, der Höhe[3] und Breite [4] des entsprechenden Rechteckes. Für die Berechnung der gaußschen Normalverteilung wurde die Java Funktion $Random.nextgaussian()$[5] verwendet.
Ein Datensatz, bei dem die Standardabweichung recht gering ist, nämlich jeweils nur ein Drittel der Höhe und der Breite des entsprechenden Rechteckes, ist in Abbildung 5.3 zu sehen.

Betrachten wir nun die euklidschen Abstände der Mittelpunkte der Rechtecke. Wir wollen, dass diese Abstände das hierarchische Verhältnis der Klassen wiedergeben. So soll gelten, dass die Distanz zwischen A und B, respektive der Mittelpunkte ihrer Rechtecke, kleiner ist als die Distanz zwischen A und C, genau dann wenn A und B sich hierarchisch näher sind als A und C. Es lässt sich in den Abbildungen leicht erkennen, dass diese Eigenschaft zwar meist erfüllt ist, aber nicht in allen Fällen. Beispielsweise ist der Mittelpunkt der Klasse 3 dem der Klasse 7 näher als dem der Klasse 1, was der Hierarchie widerspricht. Im Weiteren werden wir die Datensätze dieses Aufbaus entsprechend als *semi-hierarchisch* bezeichnen.

Beim zweiten Typ an Datensätzen wurde der räumliche Aufbau so angepasst, dass die Distanzen der Mittelpunkte der Klassen allesamt getreu der Hierarchie sind. Dies wurde erreicht indem die Rechtecke gegebenenfalls weiter voneinander weg gesetzt wurden. Ansonsten wurden diese Datensätze aber nach den gleichen, oben angeführten, Vorgehen erstellt. Diese Datensätze werden nachfolgend als die *hierarchischen* Datensätze bezeichnet. In Abbildung 5.4 lassen sich bei einem dieser Datensätze die größeren Abstände bei der Verteilung der Instanzen gut erkennen.

5.1.3 Beschreibung der Modifikationen

Es sollen nun die konkreten Datensätze, die bei den Versuchen verwendet wurden, vorgestellt werden. Sowohl für den semi-hierarchischen als auch für den hierarchischen Aufbau der Datensätze wurden verschiedene Variationen umgesetzt. Grundsätzlich beinhaltet jeder Datensatz, wenn nicht anders beschrieben, jeweils 1200 Trainings- und Testinstanzen, nämlich jeweils 100 Trainings- und Testinstanzen pro Klasse.

[2] http://www.cs.waikato.ac.nz/ml/weka/
[3] für die Standardabweichung auf der Y-Achse
[4] für die Standardabweichung auf der X-Achse
[5] http://java.sun.com/j2se/1.4.2/docs/api/java/util/Random.html

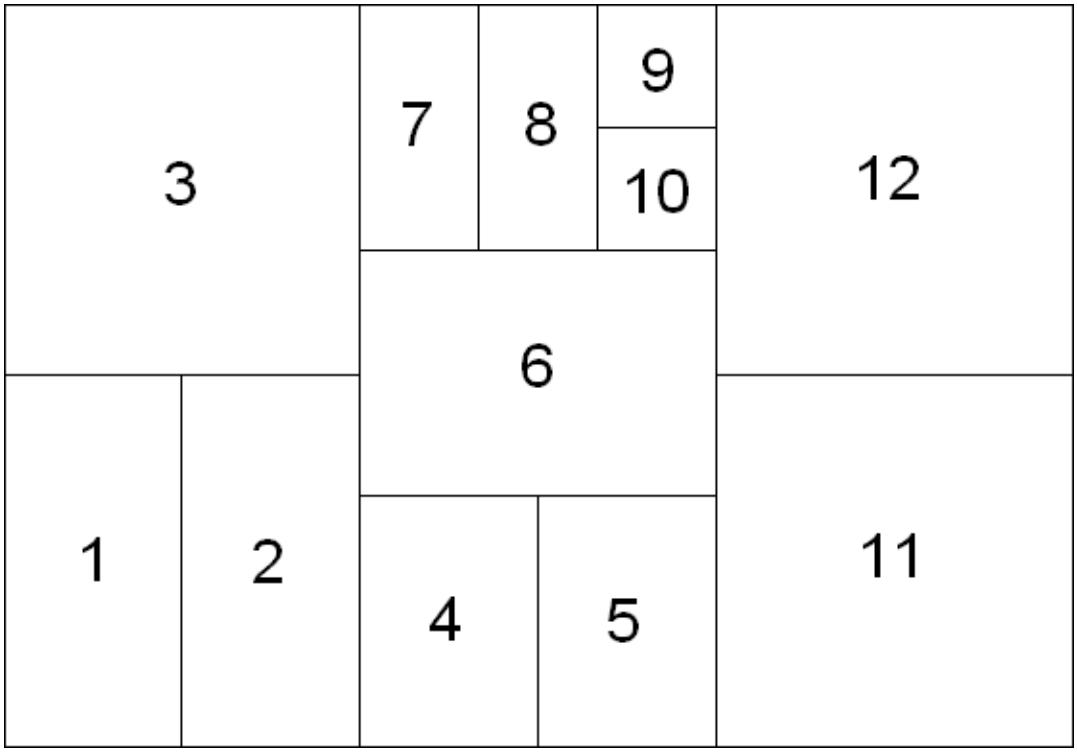

Abbildung 5.2: Räumliche Aufteilung des semi-hierarchischen künstlichen Datensatzes

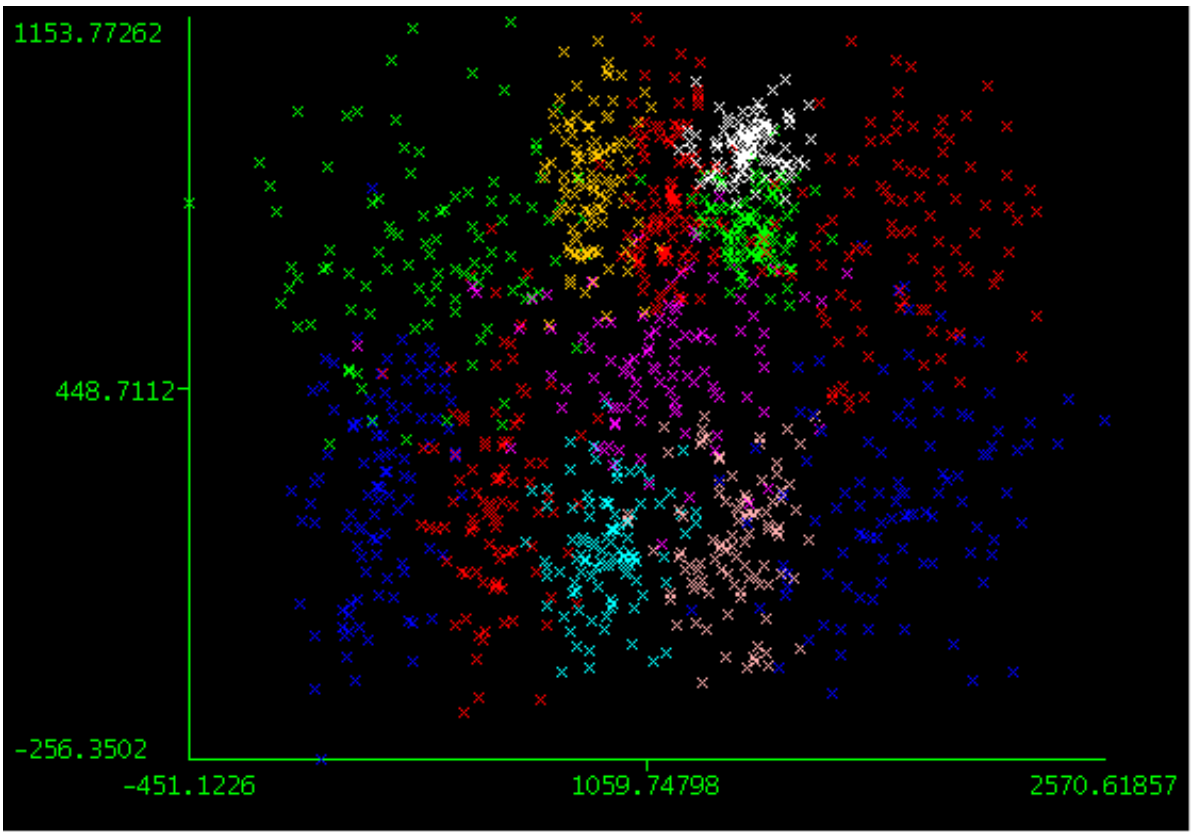

Abbildung 5.3: Beispiel der Instanzenverteilung in einem semi-hierarchischen Datensatz.

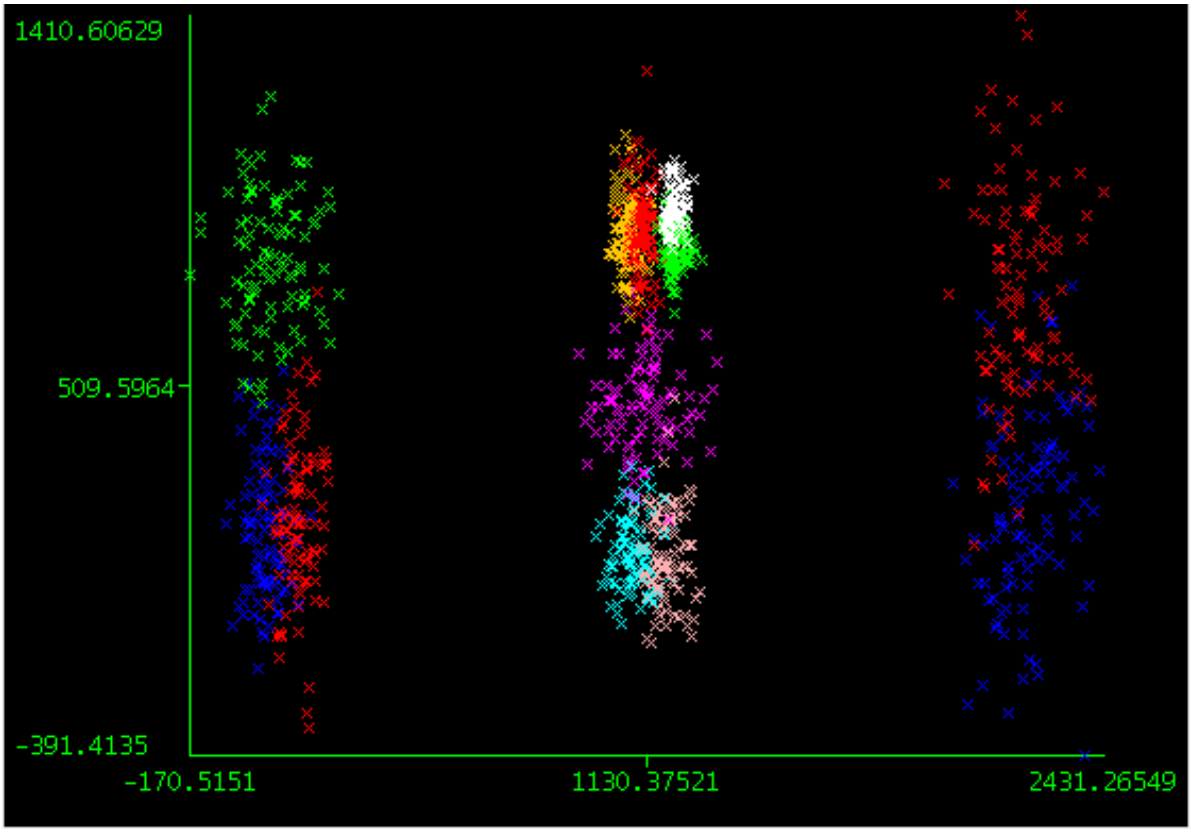

Abbildung 5.4: Beispiel der Instanzenverteilung in einem hierarchischen Datensatz.

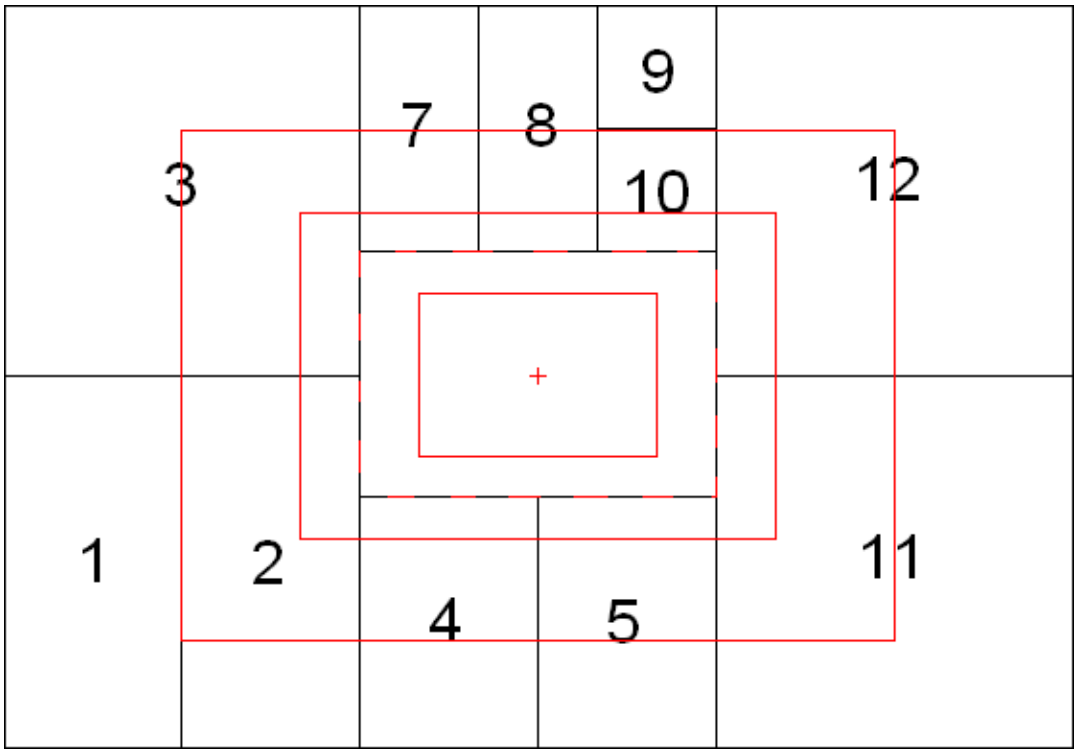

Abbildung 5.5: Die ersten 4 Stufen der Standardabweichungen der Klasse 6 graphisch in rot dargestellt.

Rauschen:
Bei der Generierung der Instanzen wurde jeweils die Standardabweichung bei der Normalverteilung der Instanzen um den vorher bestimmten Punkt im 2-dimensionalen Raum variiert. Durch eine hohe Standardabweichung steigt das Rauschen der Daten und die hierarchische Struktur in den Daten nimmt ab, weil die Instanzen sich weiter von dem Mittelpunkt, welcher an Hand der Hierarchie errechnet wurde, entfernen. Dieses Rauschen wurde in insgesamt 5 Stufen geregelt. Abbildung 5.5 zeigt die entsprechende Standardabweichung dieser Stufen für eine der Klassen bei einem semi-hierarchischen Datensatz. In der Abbildung sind nur die ersten vier Stufen eingezeichnet, da die fünfte über den Bildrand hinausgeht. Bei einer Normalverteilung gibt die Standardabweichung Informationen über die Streuung der Punkte um einen gegeben Mittelpunkt. Bei einer idealen Verteilung gelten die folgenden Aussagen. Die Standardabweichung sei mit σ bezeichnet.

- 68% aller Werte haben eine Abweichung von höchstens σ vom Mittelwert
- 95% aller Werte haben eine Abweichung von höchstens 2σ vom Mittelwert
- 99.8% aller Werte haben eine Abweichung von höchstens 3σ vom Mittelwert

Da bei der Generierung der Instanzen eine doppelte Normalverteilung um den Mittelpunkt vorliegt, nämlich einmal entlang der X-Achse und einmal entlang der Y-Achse, können wir davon ausgehen, dass sich ca. 50%[6] der Instanzen innerhalb des, der Standardabweichung entsprechenden und in der Abbildung rot gekennzeichneten, Rechteckes befinden.

Es sei an dieser Stelle vermerkt, dass die Streuung der Instanzen in diesem Modell von der Größe des zugeordneten Rechteckes der Klasse abhängig ist, weil die verwendeten Standardabweichungen jeweils Vielfache, bzw. Bruchteile, der Maße dieses Rechteckes sind. Der verwendete Algorithmus für die Generierung der Datensätze ordnet einer Klasse, welche im Hierarchiebaum auf einer tieferen Ebene liegt, ein kleineres Rechteck zu, als einer Instanz auf höherer Ebene. Für die Instanzen der tiefer liegenden Klasse folgt daraus, dass diese auf einem kleineren Raum verteilt werden. Diese Eigenschaft folgt aus dem rekursiven Aufbau des Algorithmus und dient dabei der einfacheren Einhaltung der Hierarchie in den Daten. Daher ist diese Eigenschaft im Wesentlichen praktischer Natur; die Tatsache, dass die räumliche Verteilung von Instanzen verschiedener Klassen unterschiedlich ist, kann sich jedoch auch durchaus in realen Daten wiederfinden.

Die 5 verwendeten Stufen der Streuung werden wie folgt berechnet:

- **Stufe 1 (Sehr geringe Streuung):** Die Standardabweichung ist jeweils $1/3$ der Höhe, bzw. Breite, des Rechteckes. Diese Datensätze wurden vornehmlich als Testdatensätze verwendet, da zur Evaluation der Klassifizierer eine hohe Genauigkeit in den Daten sinnvoll ist.
- **Stufe 2 (Geringe Streuung):** Die Standardabweichung ist jeweils $1/2$ der Höhe, bzw. Breite, des Rechteckes.
- **Stufe 3 (normale Streuung):** Die Standardabweichung ist jeweils entsprechend der Höhe, bzw. Breite, des Rechteckes.
- **Stufe 4 (Starke Streuung):** Die Standardabweichung ist jeweils $3/2$ der Höhe, bzw. Breite, des Rechteckes.
- **Stufe 5 (Sehr starke Streuung):** Die Standardabweichung ist jeweils das doppelte der Höhe, bzw. Breite, des Rechteckes.

[6] Die Zahl errechnet sich aus $68\% \cdot 68\% = 46\%$

Alle Datensätze mit weiteren Modifikationen wurden mit jeweils allen diesen Streuungsstufen getestet, so dass sich bei allen Fällen eine Übersicht ergibt, wie sich die beiden Klassifikationsmethoden bei steigendem Rauschen der Daten verhalten.

Geringe Anzahl an Instanzen:
Bei einigen Datensätzen wurden die Trainingsinstanzen einiger Klassen bewusst stark verringert. Üblicherweise enthält jeder Datensatz für jede der Klassen 100 Trainingsinstanzen. Bei den gekürzten Versionen erhielten die ungeraden Klassen $(1, 3, 5, 7, 9, 11)$ jeweils nur $1/5$, bzw. $1/10$, der Trainingsinstanzen, also jeweils nur 20, bzw. 10.

Der Sinn dieser Modifikation ist es, zu untersuchen ob die paarweise hierarchische Klassifikation auf diesen Datensätzen einen Vorteil gegenüber der paarweisen Klassifikation hat. Bei binären Klassifizierern kann es passieren, dass das erlernte Modell auf Grund unzureichender Anzahl an Trainingsinstanzen einer oder beider der Klassen, ein schlechteres Unterscheidungskonzept lernt, als es dies bei mehr Trainingsinstanzen würde. Das liegt daran, dass eine gute Unterscheidung natürlich schwerer wird, umso weniger Informationen man über die beiden Klassen hat. Bei den hier verwendeten 2-dimensionalen Instanzen, ist ungleich schwerer den eigentlichen Ausbreitungsraum einer Klasse zu repräsentieren, wenn man nur eine Stichprobe von wenigen Instanzen hat. Ein weiterer Faktor, der zum Tragen kommen kann, ist die Tatsache, dass einzelne verrauschte Instanzen, welche auf Grund von Streuung weit ab von den anderen Instanzen der Klasse liegen, beim Lernprozess des Klassifizierers stärker gewichtet werden, wenn sie bereits ein $1/10$ der gesamten Instanzenmenge ausmachen, als wenn es nur ein $1/100$ ist.

Die paarweise hierarchische Klassifikation könnte durch diese „Mängel“ theoretisch weniger beeinflusst werden, weil die geringe Anzahl an Trainingsinstanzen einer Klasse durch die Hinzunahme der Trainingsinstanzen hierarchienaher Klassen ausgeglichen werden kann. Da die wenigen Trainingsinstanzen einer Klasse beim Lernen nun im Kontext der anderen ähnlichen Instanzen betrachtet werden, fallen einzelne verrauschte Instanzen außerdem deutlich weniger ins Gewicht. Weiterhin ist es denkbar, dass das Entscheidungsmodell für eine Klasse, welche an sich durch ihre Trainingsinstanzen nur unvollständig beschrieben wird, trotzdem durch Betrachtung in ihrem hierarchischen Kontext, nämlich zusammen mit den Instanzen ähnlicher Klassen, relativ gut erlernt werden kann.

Vertauschte Klassen:
Um die Reaktion der paarweisen hierarchischen Klassifikation auf einzelne grobe Widersprüche der Daten zur Klassenhierarchie zu testen, wurden bei einigen der Datensätze zwei der Klassen in ihrer Position in der Hierarchie vertauscht. Konkret wurden die Klassen 3 und 10 gegeneinander ausgetauscht. Das heißt bei den entsprechenden Datensätzen wurden die, ansonsten normal generierten, Instanzen der Klasse 3 mit nicht als Klasse 3, sondern eben als Klasse 10 gekennzeichnet. Und zur Klasse 3 wurden die ursprünglichen Klasse 10 Instanzen zugeordnet. Das Klassifikationsverfahren verwendet weiterhin die normale unveränderte Hierarchie. Beim Lernprozess werden also nun Annahmen aus der Hierarchie verwendet, welche sich nicht mit den konkreten Daten decken. Die Klasse 3 ist beispielsweise laut der Hierarchie den Klassen 1 und 2 am nächsten und daher werden deren Instanzen auch mitverwendet für das Training von, zum Beispiel dem Klassifizierer $K_{1,12}$. Dieses hierarchische Verhältnis ist nach den Veränderungen in Daten aber nicht mehr gültig, denn die Instanzen der Klasse 3 sind nun nicht mehr in der Nähe der Klassen 1 und 2 anzusiedeln, sondern sie haben nun mehr Gemeinsamkeiten mit unter anderem den Instanzen der Klassen 7 und 8.

Es ist zu erwarten, dass die Ergebnisse der paarweisen hierarchischen Klassifikation bei diesen Datensätzen deutlich schlechter ausfallen als bei den unveränderten Datensätzen, da bei vielen der Basisklassifizierer nun Trainingsinstanzen verwendet werden, welche zwar unter der Annahme der Ähnlichkeit ausgewählt werden, aber tatsächlich sehr verschieden von den anderen Beispielen sein werden. Die

Folge sind Trainingsmengen, bei denen es wahrscheinlich nicht möglich sein wird, ein sinnvolles Unterscheidungsmodell zu finden. An den Ergebnissen der paarweisen Klassifikation sollten diese Änderungen offensichtlich nichts ändern, da die Hierarchie nicht berücksichtig wird, sondern jede Klasse als unabhängig von den anderen betrachtet wird.

Testinstanzen:
Die Testinstanzen wurden für alle aufgeführten Fälle mit einer sehr geringen Streuung (Stufe 1) generiert, da es für den Test der Klassifizierer wünschenswert ist, dass die Testinstanzen möglichst genau und ohne großes Rauschen sind.[7] Es sind in allen Fällen 100 Testinstanzen pro Klasse in einem Datensatz enthalten. Bei den Datensätzen mit vertauschten Klassen, gilt dieser Tausch sinnvollerweise auch bei den Testinstanzen.

5.1.4 Versuchsergebnisse und Interpretation

Für die Versuche wurde die Support Vector Maschine SMO[8], welche Teil des Projektes WEKA[9] ist, als Basisklassifizierer verwendet. Als einziger, von den Standardparametern der Support Vector Maschine SMO abweichend, wurde der Parameter „-M“ gesetzt, welcher bewirkt, dass die Vorhersage des einzelnen Basisklassifizierers nicht dichotom, sondern probabilistisch ausgegeben wird.[10]

Die vorhergesagte Klasse wurde jeweils durch einfaches Voting ermittelt. Für jeden in der Tabelle aufgeführten Fall wurden 20 unterschiedliche, aber nach gleichem Muster erstellte, Datensätze zum Lernen und Evaluieren verwendet. Die Ergebnisse in der Tabelle sind, die über diese 20 Datensätze gemittelten, prozentualen Anteile der korrekt klassifizierten Testinstanzen.

In der jeweils linken Spalte befindet sich das Ergebnis der paarweisen Klassifikation und in der jeweils rechten Spalte das der paarweisen hierarchischen Klassifikation. Es wurde das jeweils bessere der beiden Ergebnisse in fetter Schrift hervorgehoben.

Betrachten wir die Ergebnisse der paarweisen Klassifikation: Auf den semi-hierarchischen
Datensätzen mit normalem Aufbau sind die Resultate der Klassifikation bei den ersten zwei Streuungsstufen sehr nah beieinander und auch bei der dritten Streuungsstufe liegt der Anteil der korrekt klassifizierten Trainingsinstanzen nur ca. 3 Prozentpunkte niedriger. Bei weiter zunehmendem Rauschen fallen die Ergebnisse stark ab bis zu unter 50% korrekt eingeordneten Instanzen bei der größten Streuung. Bei den Datensätzen mit vertauschten Klassen sind die Resultate erwartungsgemäß nahezu gleich; die geringen Unterschiede lassen sich auf die „zufällige“ Instanzenverteilung der Datensätze zurückführen.
Für die beiden Fälle, in denen die Hälfte der Klassen eine geringere Anzahl an Trainingsinstanzen besitzen, sind die Ergebnisse im Vergleich zur „normalen“ Instanzenverteilung durchgehend schlechter, was ebenfalls der Erwartung entspricht. Es lässt sich beobachten, dass die Resultate in diesen Fällen deutlich stärker vom zunehmenden Rauschen beeinflusst werden, so dass der Anteil der richtig klassifizierten Testinstanzen bei der dritten Streuungsstufe bereits bei ca. 50% liegt und tendenziell weiter stark abfällt. Der Unterschied zwischen der Beschränkung auf entweder 20 oder 10 Trainingsinstan-

[7] Andererseits ist es bei der Anwendung in Praxis im Allgemeinen so, dass man gleiche Eigenschaften bei sowohl Trainings- als auch Testdaten erwarten kann. In diesen Untersuchungen jedoch soll der Fokus unter anderem darauf liegen, zu untersuchen wie gut die Klassifikationsmethode die, in den teilweise verrauschen oder mit nur unzureichenden Trainingsinstanzen ausgestatteten Daten mehr oder weniger stark vorhandene, Hierarchiestruktur in dem gelernten Unterscheidungskonzept umsetzen kann. Für diesen Zweck ist es sinnvoll die Hierarchie in den Testdaten möglichst unverändert zu belassen.

[8] http://www.dbs.ifi.lmu.de/Lehre/KDD_Praktikum/weka/doc/weka/classifiers/functions/SMO.html

[9] http://www.cs.waikato.ac.nz/ml/weka/

[10] Vergleiche hierzu 2.4

Tabelle 5.1: Klassifikationsergebnisse auf künstlichen Datensätzen
In der jeweils linken Spalte befindet sich das Ergebnis der paarweisen Klassifikation; in der jeweils rechten Spalte das Ergebnis der paarweisen hierarchischen Klassifikation.

Datensatz	**Stufe der Streuung**									
Semi-hierarchisch	**1**		**2**		**3**		**4**		**5**	
Normal	**82.94**	81.88	**82.04**	74.61	**79.35**	62.4	**67.13**	45.98	**47.97**	14.97
Vertauschte Klassen	**83.11**	70.52	**82.64**	66.64	**79.86**	60.5	**65.55**	46.64	**49.18**	15.72
100-20	78.39	**78.89**	**65.41**	64.53	**50.41**	41.03	**39.02**	20.61	**25.5**	10.41
100-10	74.52	**76.37**	**57.31**	55.62	**45.43**	33.0	**37.85**	19.47	**26.3**	10.59
Hierarchisch	**1**		**2**		**3**		**4**		**5**	
Normal	93.98	**93.99**	**94.08**	93.96	**93.85**	93.36	**92.68**	89.19	**84.4**	66.33
Vertauschte Klassen	**94.32**	79.78	**94.35**	78.56	**94.02**	77.32	**93.26**	75.65	**84.52**	52.22
100-20	**91.7**	91.69	**83.8**	82.8	**68.72**	68.1	**55.99**	51.22	**47.85**	34.92
100-10	88.28	**88.5**	**73.42**	72.73	**59.81**	59.28	**53.11**	50.17	**48.58**	34.82

zen zeigt sich bei geringem Rauschen noch, scheint aber dann zu Gunsten des negativen Effektes des zunehmenden Rauschens in den Hintergrund zu treten.

Die paarweise hierarchische Klassifikation zeigt auf den semi-hierarchischen Daten im Vergleich generell schlechtere Ergebnisse als die nicht-hierarchische Klassifikation. Auf den Datensätzen mit normalem Aufbau ist nur das Ergebnis bei der sehr geringen Streuung mit dem der paarweisen Klassifikation vergleichbar, von dem es sich um ca. 1 Prozentpunkt unterscheidet. Die zunehmende Streuung scheint sich auf die Resultate des hierarchischen Ansatzes deutlich stärker und auch früher, also bereits bei relativ geringem Rauschen, auszuwirken, so dass die Differenz in den Ergebnissen schon bei der dritten Stufe der Streuung bei ca. 17 Prozentpunkten liegt.
Durch das Vertauschen der Klassen, welches eine Inkonsistenz in der Hierarchie verursacht, verschlechtern sich die Ergebnisse erwartungsgemäß um ca. 10 Prozentpunkte bei den ersten beiden Streuungstufen, ab der der dritten Stufe scheint der negative Effekt des Rauschens die Fehler in der Hierarchie zu überdecken und die Ergebnisse gleichen sich denen auf normalen Daten an.
Auch auf den Datensätzen mit beschränkten Trainingsinstanzen sind die Ergebnisse nachvollziehbarerweise im Vergleich zu den normalen Daten schlechter. Es lässt sich jedoch beobachten, dass insbesondere bei der ersten und zweiten Streuungstufe diese Verschlechterung beim hierarchischen Ansatz geringer ausfällt als bei der paarweisen Klassifikation. Beim sehr geringen Rauschen liegen die Ergebnisse der paarweisen hierarchischen Klassifikation sogar leicht vor denen des nicht-hierarchischen Ansatzes.[11] Bei der zweiten Stufe liegen beide Ergebnispaare bis auf maximal ca. 2 Prozentpunkte beieinander, obwohl das Ergebnis auf den normalen Datensätzen für die paarweise Klassifikation um ca. 8 Prozentpunkte besser ausfiel. Man kann also für geringes Rauschen (Stufe 1 und 2) eine gewisse erhöhte Robustheit der paarweisen hierarchischen Klassifikation gegenüber einer partiellen Unterrepräsentation der Klassen in der Trainingsmenge erkennen.

Die absoluten Ergebnisse der beiden Verfahren auf den hierarchischen Datensätzen fallen bei allen Abstufungen des Rauschens deutlich besser aus als bei den semi-hierarchischen Datensätzen was in ersten Linie wohl damit begründet werden muss, dass durch den veränderten Aufbau, welcher die Hierarchietreue der Daten garantieren soll, viele der Klassen nun räumlich weiter voneinander entfernt sind, so dass eine Trennung der Klassen klarer getroffen werden kann und auch das zunehmende Rauschen die

[11] Diese geringe Differenz muss jedoch nicht zwingend auf die Methoden selber zurückgeführt werden, da sich dies auch durch die „zufällige Natur“ der Datensätze erklären lässt.

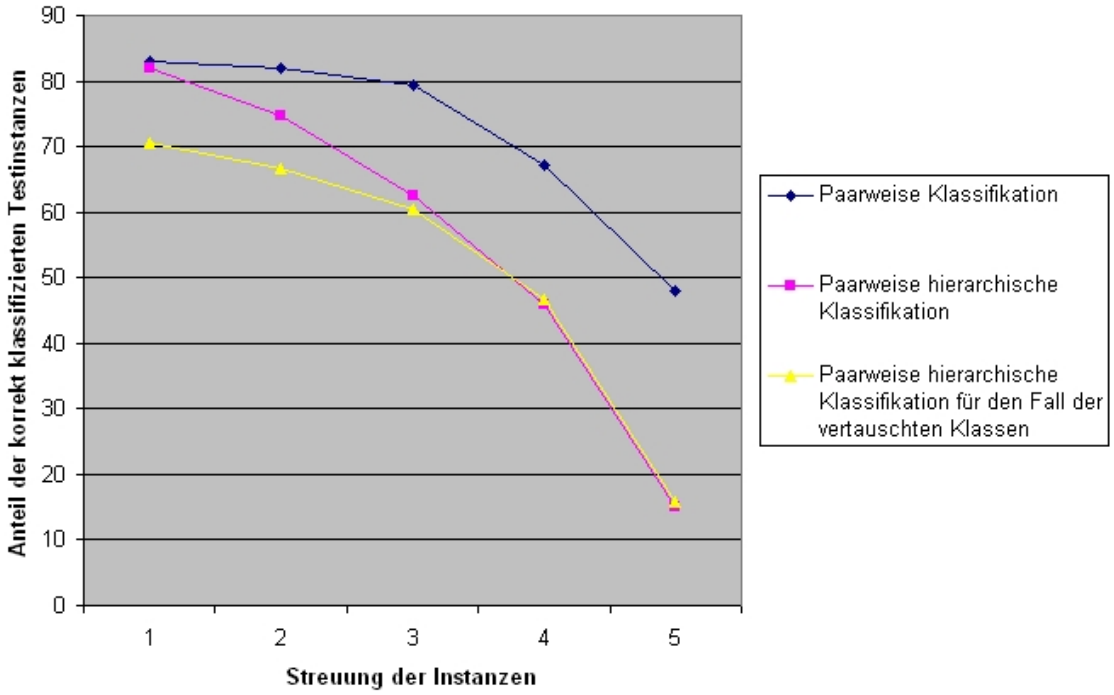

Abbildung 5.6: Ergebnisse auf den semi-hierarchischen Daten

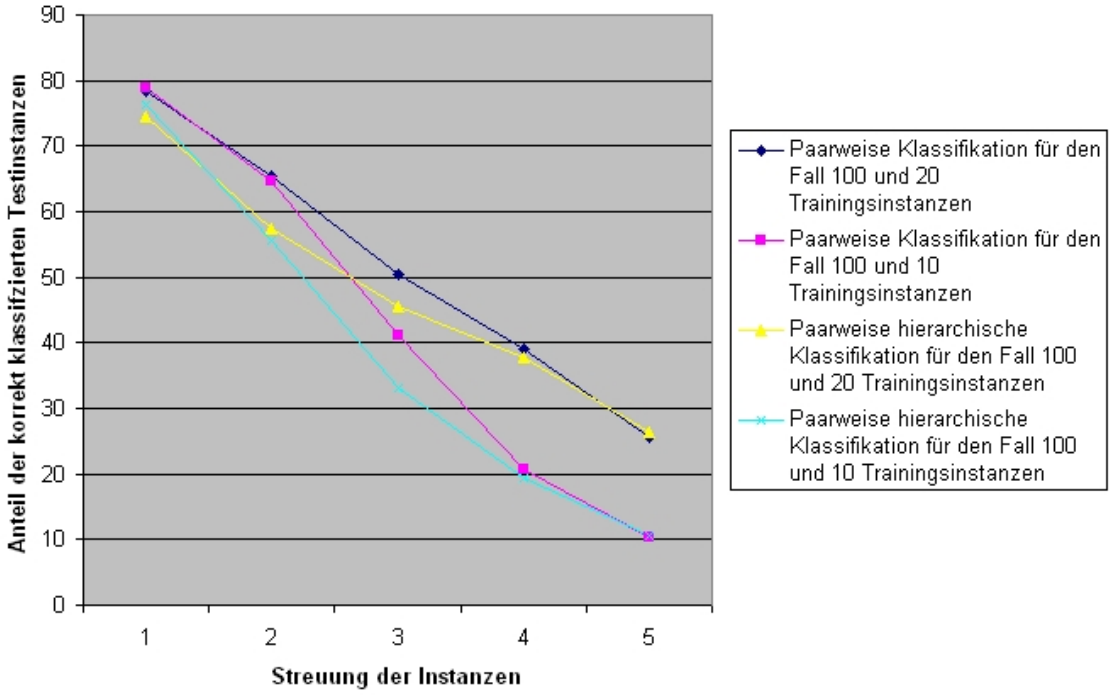

Abbildung 5.7: Ergebnisse auf den semi-hierarchischen Daten

Instanzen nicht so stark miteinander vermischen kann, wie es bei den semi-hierarchischen Datensätzen der Fall ist.
Die Resultate der paarweisen Klassifikation liegen bei den normalen hierarchischen Datensätzen bei den ersten vier Streuungstufen sehr nahe beieinander um jeweils 93% herum. Erst ab der fünften Streuungsstufe lässt sich der negative Effekt des Rauschens tendenziell deutlicher erkennen. Auch hier veränderte das Vertauschen zweier Klassen die Ergebnisse dieses Ansatzes natürlich nicht.
Auf den Datensätzen, bei denen jede zweite Klasse nur 20%, bzw. 10%, der ursprünglichen Trainingsinstanzen gestellt kriegt, fallen die Ergebnisse für sehr geringes Rauschen nur wenig ab (2 bis 3 Prozentpunkte), jedoch verschlechtern sich die Ergebnisse mit zunehmendem Rauschen bereits sehr deutlich ab der zweiten Stufe. So liegen die beiden Ergebnisse bei der dritte Stufe bereits ca. 25 Prozentpunkte (bei 20% der Instanzen) und ca. 35 Prozentpunkte (bei 10% der Instanzen) niedriger als das Ergebnis für den normalen Datensatz mit kompletter Trainingsinstanzenmenge.

Die Werte der paarweisen hierarchischen Klassifikation bewegen sich beim normalen Datensatz bei den ersten vier Streuungstufen sehr nahe an denen der paarweisen Klassifikation. Die größte Differenz liegt bei ca. 3 Prozentpunkten bei der vierten Stufe. Für die fünfte Stufe lässt bereits erkennen, dass der durch weiteres Rauschen entstehende Werteabfall tendenziell stärker ist als bei der paarweisen Klassifikation. Für die Datensätze mit vertauschten Klassen verhält sich die hierarchische Methode sehr ähnlich wie auf den normalen Datensätzen nur mit durchweg um ca. 14 Prozentpunkte niedrigeren Werten.
Für die beiden Fälle der Datensätze mit verringerten Trainingsinstanzen verhält sich die paarweise hierarchische Klassifikation bei den ersten drei Streuungsstufen fast wie die paarweise Klassifikation und nur ab der vierten Stufe zeigt sich wiederum die stärkere Anfälligkeit für zunehmendes Rauschen.

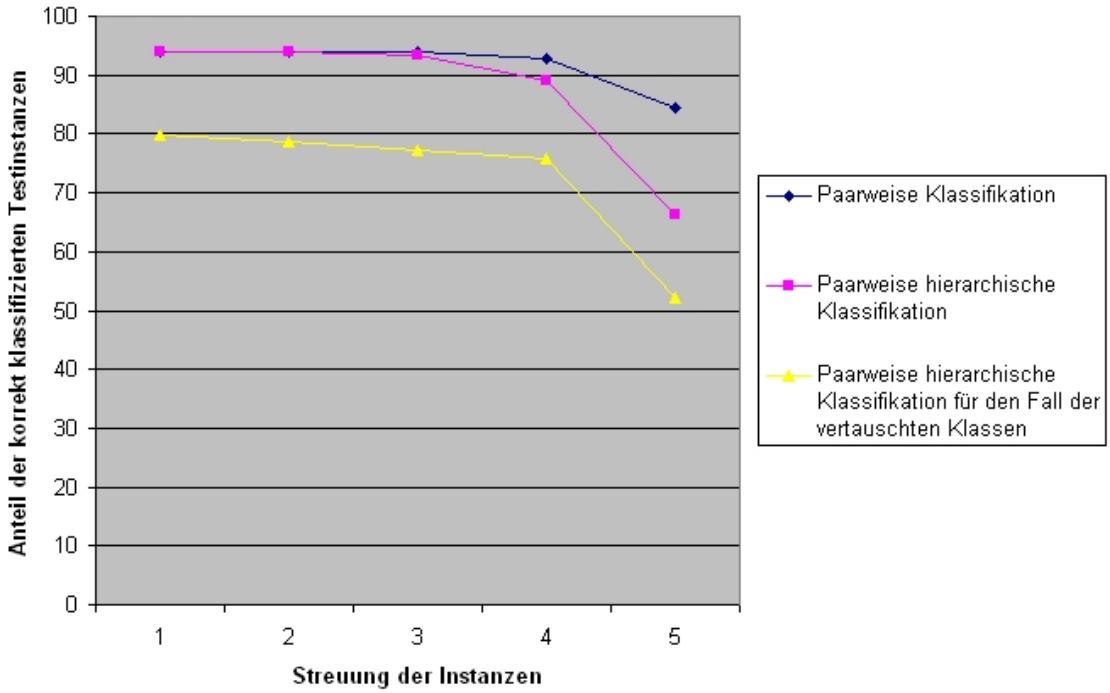

Abbildung 5.8: Ergebnisse auf den hierarchischen Daten

Die paarweise hierarchische Klassifikation scheint der paarweisen Klassifikation in diesen Versuchen also generell unterlegen. Eine mögliche Erklärung für diese Ergebnisse könnte sein, dass unsere Über-

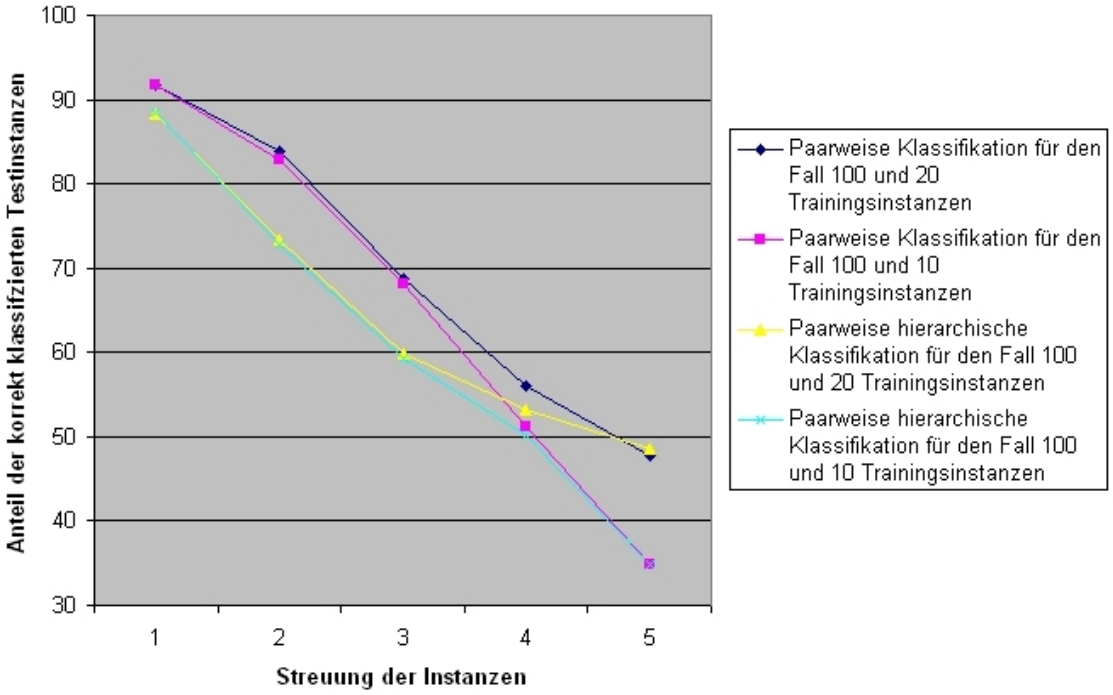

Abbildung 5.9: Ergebnisse auf den hierarchischen Daten

legungen unter 3.4 über mögliche Vorteile der paarweisen hierarchischen Klassifikation in bestimmten Klassifikationsszenarien sich in der Praxis nicht bewahrheiten. Eventuell sind die Szenarien in den Datensätzen auch nicht häufig genug, um andere potentielle Schwachstellen der Methode, zum Beispiel die Tatsache, dass komplexere Modelle erlernt werden als bei der paarweisen Klassifikation, auszugleichen oder gar zu überdecken.

Insbesondere lässt sich festhalten, dass die Anfälligkeit gegenüber Rauschen bei der paarweisen hierarchischen Klassifikation deutlich prägnanter zum Vorschein kam. Bei den theoretischen Überlegungen unter 3.4 haben wir bei dem Vorhandensein von vereinzelten verrauschen Instanzen mit eventuellen Vorteilen für den hierarchischen Ansatz gerechnet. Es bleibt die Frage, warum die Versuche nun das Gegenteil auszusagen scheinen. Erklären könnte dies der Umstand, dass bei dem Rauschen der Datensätze nicht nur einzelne Instanzen sondern alle Instanzen gleichmäßig vom Rauschen beeinflusst wurden[12], so dass die theoretische Überlegung, dass vereinzeltes Rauschen durch die größere Menge an Trainingsinstanzen relativiert werden kann, hierbei nicht zutrifft. Weiter könnte man an Hand der Ergebnisse die Überlegung anstellen, dass die hierarchische Methode, da sie die Instanzen mehrerer Klassen zusammen für einen Basisklassifizierer benutzt, stärker von dieser Art des Rauschens beeinflusst wird, da jeder Basisklassifizierer dadurch die verrauschten Instanzen gleich mehrere Klassen zu „verkraften“ hat.

Ein weiterer interessanter Punkt zeigte sich beim Vergleich der Resultate auf den Datensätzen mit normalem Aufbau und denen mit für einige Klassen verringerten Trainingsinstanzen. Für die semihierarchischen Datensätze fallen die Ergebnisse der paarweisen hierarchischen Klassifikation durch die Verringerung der Trainingsinstanzen einzelner Klassen weniger stark ab, als es bei der paarwei-

[12] Gemeint ist, dass jede einzelne Instanz mittels Normalverteilung und der Standardabweichung entsprechend der Stufe des Rauschens generiert wird.

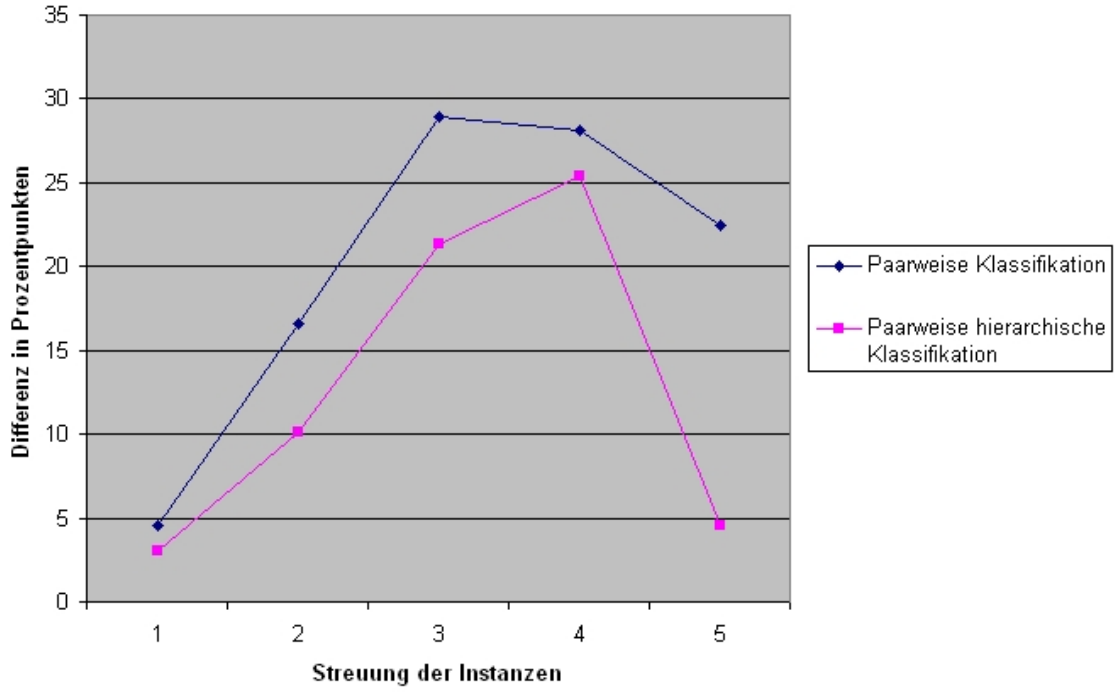

Abbildung 5.10: Differenz der Ergebnisse auf den semi-hierarchischen Datensätzen: normal und 100-20

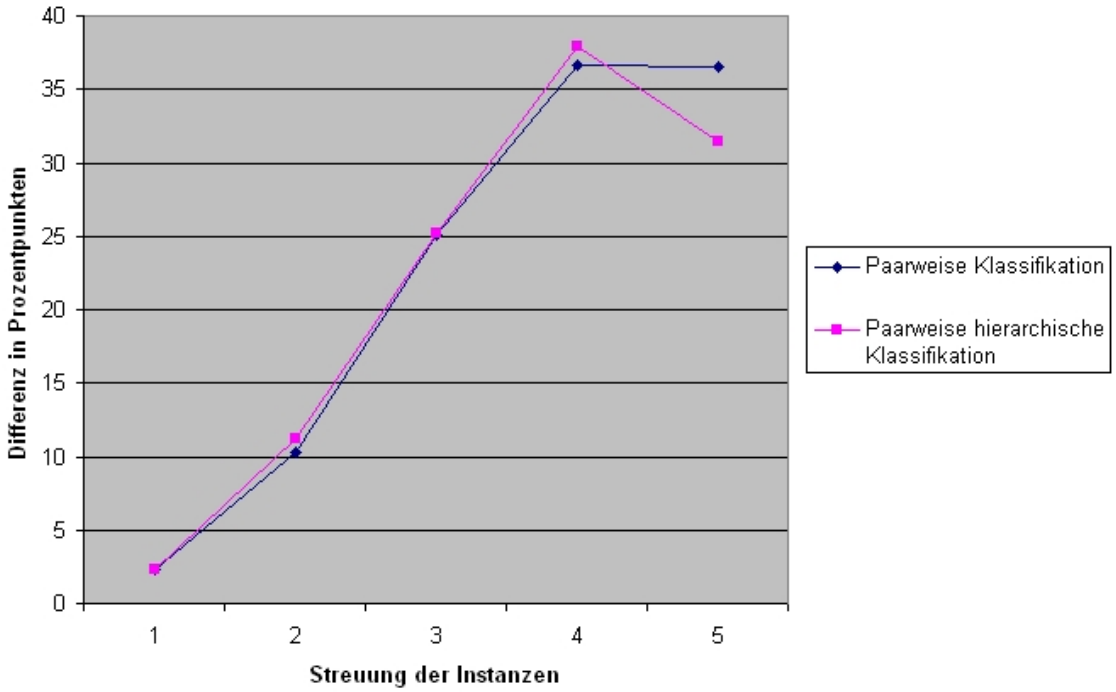

Abbildung 5.11: Differenz der Ergebnisse auf den hierarchischen Datensätzen: normal und 100-20

sen hierarchischen Klassifikation der Fall ist. Dies ließe sich eventuell mit unseren Überlegungen aus 3.4 erklären, dass das Hinzufügen von hierarchienahen Trainingsinstanzen die Unterrepräsentation von Klassen ausgleichen kann. Allerdings muss beachtet werden, dass die absoluten Ergebnisse trotzdem stets unter oder nur sehr gering über denen der paarweisen Klassifikation liegen und bei erhöhtem Rauschen schnell in Bereiche unter 50% korrekt klassifizierter Testinstanzen abfallen. Die Abbildung 5.10 zeigt die jeweiligen Differenzen in den Ergebnissen auf normalen Daten und denen mit verringerten Trainingsinstanzen. Bei den Datensätzen mit hierarchischem Aufbau tritt dieser Effekt jedoch nicht auf, was in der Abbildung 5.11 deutlich wird.

5.2 Reuters Datensatz

Für den Test der paarweisen hierarchischen Klassifikation auf Multi Label Daten wurden Trainings- und Testinstanzen aus dem Datensatz Reuters Corpus Volume 1 v2 (RCV1v2)[13] verwendet. Zunächst soll dieser Datensatz mit seinen Eigenschaften kurz vorgestellt werden, danach werden die genauen Parameter des Versuches beschrieben und schließlich die Ergebnisse aufgeführt und interpretiert.

5.2.1 Eigenschaften des Datensatzes

Der Reuters Corpus Volume 1 (RCV1) ist ein von der Nachrichtenagentur Reuters im Jahre 2000 herausgegebenes Textarchiv aus echten Nachrichtentexten des Zeitraumes zwischen dem 20. August 1996 und dem 19. August 1997. Das Archiv wurde für die Forschung veröffentlicht und findet, wie auch sein Vorgänger Reuters 21578, weite Verwendung. Der Datensatz besteht aus insgesamt über 800.000 Nachrichtentexten in englischer Sprache. Die Wortanzahl pro Text beläuft sich im Schnitt auf ca. 1000 Worte. Lewis u.a. haben in [Lewis *et al.*, 2004] eine überarbeitete Version des Reuters Datensatzes vorgestellt: den Reuters Corpus Volume 1 v2 (RCV1v2). Es wurden dabei im Wesentlichen einige Fehler und Inkonsistenzen des Originals behoben.

Auf den Reuters Daten sind insgesamt drei Mengen von Kategorien definiert:

- **Topics:** Zuordnung nach dem Thema der Nachricht
- **Industry:** Zuordnung nach dem wirtschaftlichen, bzw. industriellen, Bezug
- **Region:** Zuordnung nach geographischem Gesichtspunkt

Als Klassen für die Klassifizierung wird in dieser Arbeit die Menge der Topics (im Weiteren als *Themen* bezeichnet) verwendet, da sie der üblichen Textklassifizierung entspricht und eine hierarchische Struktur besitzt. Betrachten wir also die Struktur der Menge der Themen genauer:

Die insgesamt 103 verschiedenen Themen sind in einer Knotenhierarchie organisiert. Die Zuordnung der Themen zu den Nachrichtentexten erfolgt nach zwei wesentlichen Prinzipien: der *Minimum Code Policy* und der *Hierarchy Policy*. Die Minimum Code Policy besagt, dass eine Instanz den treffendsten Klassen zugeordnet werden soll. Die Hierarchy Policy erwirkt, dass falls einer Instanz eine Klasse zugeordnet wird, dieser damit auch alle entsprechenden Superklassen zugeordnet sind. Der komplette hierarchische Aufbau der Themen-Klassen findet sich im Anhang A. Die Hierarchie ist bis zu 4 Stufen tief und 23 der Themen werden im Hierarchiebaum als interne Knoten repräsentiert und sind somit die Superklassen mindestens einer weiteren Klasse. Entsprechend sind die restlichen 80 Themen als

[13] $http://www.jmlr.org/papers/volume5/lewis04a/lyrl2004\_rcv1v2\_README.htm$

Blätter dargestellt. Von der Wurzel des Baumes gehen 4 Kanten aus zu den obersten Klassen GCAT (Government/Social), CCAT (Corporate/Industrial), MCAT (Markets) und ECAT (Economics) aus. Die klassenmäßig größten Gruppen sind CCAT und GCAT mit 33, bzw. 32, Unterklassen.

5.2.2 Aufbereitung des Datensatzes

Bei unserem Versuch wurden insgesamt 5 verschiedene Datensätze zu je 3000 Trainings- und Testinstanzen verwendet. Jeder dieser Datensätze stellt nur eine Teilmenge aller Instanzen des RCV1v2 Datensatzes dar. Die fünf Datensätze stehen unter
http://www.csie.ntu.edu.tw/%7Ecjlin/libsvmtools/datasets/multilabel.html zur Verfügung.
Bei diesen Datensätzen gibt es effektiv nur 101 Klassen, also 2 weniger als beim gesamten Datensatz, da für die Themen GMIL (Millennium Issues) und E312 (Capacity Utilization) weder Trainings- noch Testinstanzen vorhanden sind. Die durchschnittliche Anzahl zugeordneter Klassen pro Instanz über allen 5 Datensätzen ist 4[14]. Jede Instanz besitzt über 40.000 Attribute[15], wobei jedes dieser Attribute ein Wort repräsentiert. Der Wert eines Attributes entspricht dem statistischen Maß *tf-idf* des jeweiligen Wortes. Tf-idf steht für *term frequency - inverse document frequency*; es ist ein Maß für die Wichtigkeit eines Wortes in einem Dokument in dem Kontext der gesamten Dokumentsammlung und kommt aus den Bereichen des Text-Mining und Information Retrieval.

Text frequency ist schlicht die Häufigkeit mit der ein Wort in einem Dokument vorkommt; dieser Wert wird zusätzlich in Hinblick auf die Länge (Anzahl aller Worte) des Dokumentes normalisiert:

$$tf_i := \frac{w_i}{\sum_j w_j}$$

w_i ist hierbei die Anzahl des relevanten Wortes, welches durch die Summe der Anzahl aller, im Dokument vorkommenden, Worte geteilt wird.

Inverse document frequency ist ein Maß für die Wichtigkeit, bzw. Spezialität, eines Wortes. Kommt ein Wort nur in sehr wenigen Dokumenten vor, dann wird angenommen, dass dieses Wort die Texte, in denen es vorkommt, gut beschreibt und sich entsprechend als Kriterium für z.B. eine Klassifikation eignet. Berechnet wird dieser Wert durch Logarithmieren des Quotienten von allen Dokumenten und den Dokumenten in welchen das Wort vorkommt:

$$idf_i := \log \frac{|D|}{|\{d|w_i \in d, d \in D\}|}$$

D ist die Menge aller Dokumente und es gilt $w_i \in d$ genau dann wenn das Wort w_i im Dokument d vorkommt. Bei der inverse document frequency schneiden insbesondere sehr oft vorkommende, und damit meist für die Klassifikation nutzlose, Worte, wie zum Beispiel „und“, schlecht ab.

Tf-idf ist das Produkt dieser beiden Maße und berücksichtigt somit sowohl die Wichtigkeit eines Wortes als auch die Häufigkeit mit der es in einem betreffenden Dokument vorkommt.

$$tf - idf_i := tf_i \cdot idf_i$$

Je höher der tf-idf Wert für ein Wort in einem Dokument ist desto besser charakterisiert dieses Wort das Dokument. Der niedrigste mögliche Wert ist 0, nämlich wenn das Wort in dem Dokument nicht

[14] Gerundet auf eine Nachkommastelle
[15] Die genaue Anzahl ist bei den 5 Datensätzen leicht unterschiedlich

vorkommt. Bei [Salton and Buckley, 1988] findet sich eine genauere Beschreibung des tf-idf Maßes, sowie ein Vergleich mit anderen ähnlichen gewichteten Maßen.

Um eine praktikable Laufzeit auf den Datensätzen zu gewährleisten wurde eine *Feature Selection* (Auswahl der Attribute) vorgenommen. Dies bedeutet, dass nicht alle Attribute für den Lernvorgang und die Klassifikation verwendet wurden, sondern nur eine vorher bestimmte Teilmenge. Es wurden insgesamt jeweils nur 5.000 der ca. 40.000 Attribute ausgewählt. Es wurden für alle 5 Datensätze jeweils an Hand der Trainingsdaten alle Attribute danach geordnet, bei wie vielen der 3000 Instanzen diese einen Wert ungleich 0 haben. In dieser Rangfolge wurden die ersten 5.000 Attribute ausgewählt und beim Lernvorgang verwendet. Bei der Klassifikation der Testdaten wurden dann natürlich ebenfalls genau diese 5.000 Attribute genutzt. Dieses Verfahren der Attributauswahl entspricht der als „Dokumenthäufigkeit“ bekannten Methode, die bei der Textklassifikation oft Verwendung findet. Üblicherweise sind die Attribute dabei jedoch keine tf-idf Werte, sondern nur die Anzahl eines Wortes. Vergleiche dazu Yang und Pedersen, bei deren Untersuchungen in [Yang and Pedersen, 1997] die Dokumenthäufigkeit gegenüber anderen Feature Selection Methoden gut abschneidet.
Es ist zu beachten, dass bei einer solchen relativ umfangreichen Selektion der Attribute eine generelle Verschlechterung der Klassifikationsergebnisse zu erwarten ist. Allerdings sind die absoluten Ergebnisse der beiden zu vergleichenden Klassifikationsmethoden für unseren Versuch nicht ausschlaggebend. Interessant sind vielmehr die Unterschiede in den Ergebnissen zwischen der paarweisen Klassifikation und ihrer hierarchischen Abwandlung.

5.2.3 Versuchsparameter

Für die Versuche wurde die Support Vector Maschine SMO[16], welche Teil des Projektes WEKA[17] ist, als Basisklassifizierer verwendet. Als einziger, von den Standardparametern der Support Vector Maschine SMO abweichend, wurde der Parameter „-M“ gesetzt, welcher bewirkt, dass die Vorhersage des einzelnen Basisklassifizierers nicht dichotom, sondern probabilistisch ausgegeben wird.[18]

Für die Erstellung des Rankings wird das unter 2.5.2 vorgestellte Voting verwendet. Die Relevanzgrenze liegt bei der durchschnittlichen Anzahl relevanter Klassen, also 4, das heißt die ersten vier Klassen gelten bei jedem Ranking als vorhergesagt während alle anderen Klassen als nicht vorhergesagt gelten. Die Entscheidung eine fixe Grenze im Ranking zu setzen, maximiert sicherlich nicht die Klassifikationsergebnisse, jedoch ist dies für die relativen Ergebnisse und den Vergleich der beiden Klassifikationsmethoden nicht entscheidend.

5.2.4 Versuchsergebnisse und Interpretation

Die Ergebnisse werden mit zwei Arten von Evaluationsfunktionen bewertet: die Ergebnisse von Precision, Recall, F1 und Hamming Loss sind abhängig von der Relevanzgrenze im Ranking, alle anderen Maße dagegen bewerten das Ranking als ganzes und sind entsprechend unabhängig von der gewählten Grenze.
Bei ersteren Funktionen gibt es zwei Arten wie man die durchschnittlichen Ergebnisse über alle Testinstanzen, bzw. über alle erstellten Rankings auf den Testinstanzen, errechnen kann: *Micro Average* und *Macro Average*. Bei der Macro Average Methode wird beispielsweise der Precision Wert für jedes Ranking einzeln berechnet und dann wird die Summe der Ergebnisse durch die Anzahl der Rankings

[16] http://www.dbs.ifi.lmu.de/Lehre/KDD_Praktikum/weka/doc/weka/classifiers/functions/SMO.html
[17] http://www.cs.waikato.ac.nz/ml/weka/
[18] Vergleiche hierzu 2.4

geteilt. Bei Micro Average dagegen wird der Precision Wert nur einmalig berechnet über die von allen Rankings aufsummierten Werte „true positive“ (TP) und „false positive“(FP). Sei R die Menge der Rankings für die Testinstanzen.

$$Prec_{macro} := \frac{1}{|R|} \sum_{i \in R} \frac{TP_i}{TP_i + FP_i}$$

$$Prec_{micro} := \frac{\sum_{i \in R} TP_i}{\sum_{i \in R} TP_i + \sum_{i \in R} FP_i}$$

Wir verwenden für Precision, Recall, F1 und Hamming Loss die Micro Average Methode.

Da nach unserem Hierarchieverständnis Superklassen stets automatisch für eine Instanz gesetzt sind, wenn eine der Kinderklassen gesetzt ist, stehen wir bei der Evaluation vor der Frage, ob wir diesen Umstand berücksichtigen wollen. Denkbar wäre so, dass die Vorhersage einer Klasse A folglich implizit die Vorhersage aller Superklassen von A bedeutet, also dass eventuell mit nur einer Klassen im Ranking zum Beispiel mehrere „true positives“ gegeben sind. Dieses Vorgehen würde einige methodische Probleme mit sich bringen, wie die Frage ob bereits implizit gezählte Klassen später im Ranking nun ignoriert werden oder als „true positive“ oder „false negative“ erscheinen. Weiter sind die beiden Funktionen Precision und Recall generell nicht dafür ausgelegt, dass einzelne Klassen die Vorhersage anderer Klassen bedingen. Daher wurden durchgehend alle Klassen nur für sich selber gewertet. Das heißt auch, dass für ein perfektes Ranking erwartet wird, dass alle Klassen inklusive Superklassen der Testinstanz vorhergesagt werden müssen.

Bei den Funktionen, die das Ranking ohne Rücksicht auf die Relevanzgrenze beschreiben, stellen die Werte in der folgenden Tabelle den Durchschnitt über alle Testinstanzen dar.

Der Wert *„korrekte n Klassen“* beschreibt die durchschnittliche Anzahl der aufeinander folgenden relevanten Klassen in einem Ranking angefangen vom Top Rank. Sind beispielsweise die ersten drei Klassen im Ranking relevant und die vierte irrelevant, dann ist dieser Wert genau 3, gleichgültig wie viele andere relevante Klassen weiter unten im Ranking angeführt sind.

In der Tabelle 5.2 finden sich die Ergebnisse der einzelnen Evaluationsmaße für die fünf verwendeten Datensätze. Es sind die, im Vergleich der beiden Klassifikationsverfahren, besseren Ergebnisse in fetter Schrift hervorgehoben. Die Tabelle 5.3 enthält den Durchschnittswert der Ergebnisse über alle fünf Datensätze.

Wir müssen feststellen, dass die Ergebnisse der paarweisen hierarchischen Klassifikation ziemlich gleichmäßig unter denen der paarweisen Klassifikation liegen, jedoch bei beispielsweise Precision und Recall mit 5, bzw. 4, Prozentpunkten nicht sehr viel niedriger sind.
Es fällt auf, dass sich der erste Datensatz mit seinen Werten bei der paarweisen hierarchischen Klassifikation deutlich negativ von allen anderen Datensätzen abhebt, während er bei den Ergebnissen der paarweisen Klassifikation nur leicht unter dem Durchschnitt aller Datensätze liegt. Daher liegt der Schluss nahe, dass bei diesem Datensatz ein die Hierarchie betreffender Unterschied zu den anderen Datensätzen besteht. Wie dieser Unterschied jedoch aussieht ist nur schwer zu vermuten, insbesondere da dieser Verdacht im Abschnitt 5.3 bei der Überprüfung der Umsetzung der Hierarchie in den Datensätzen nicht bestärkt werden konnte.

Beim Vergleich der Werte der Tabelle fallen insbesondere die beiden Maße „Margin Loss“ und „korrekte n Klassen“ auf. „Margin Loss“ ist die Differenz zwischen dem Rang der am besten im Ranking platzierten irrelevanten Klasse und der am schlechtesten platzierten relevanten Klasse. Bei allen Datensätzen liegt dieser Wert bei der hierarchischen Methode deutlich über dem der nicht-hierarchischen.

Tabelle 5.2: Klassifikationsergebnisse auf den Reuters Daten

Evaluationsmaß	**1**	**2**	**3**	**4**	**5**
Paarweise Klassifikation:					
Precision	**0.58**	**0.61**	**0.59**	**0.58**	**0.62**
Recall	**0.56**	**0.60**	**0.60**	**0.58**	**0.60**
F1	**0.58**	**0.60**	**0.60**	**0.58**	**0.61**
Hamming Loss	**0.03**	0.03	0.03	**0.03**	0.03
Margin Loss	**10.30**	**10.28**	**10.57**	**10.97**	**10.48**
One Error	**0.17**	**0.13**	**0.11**	**0.18**	**0.12**
Rank Loss	**0.04**	**0.04**	**0.04**	**0.04**	**0.04**
Avg Precision	**0.73**	**0.75**	**0.75**	**0.73**	**0.76**
korrekte n Klassen	**1.90**	2.10	2.03	1.89	2.07
Paarw. Hierar. Klassifikation:					
Precision	0.49	0.57	0.56	0.56	0.59
Recall	0.49	0.57	0.57	0.56	0.57
F1	0.49	0.57	0.56	0.56	0.58
Hamming Loss	0.04	0.03	0.03	0.04	0.03
Margin Loss	24.62	16.50	15.15	18.62	14.63
One Error	0.27	0.17	0.14	0.22	0.17
Rank Loss	0.14	0.08	0.08	0.10	0.07
Avg Precision	0.66	0.73	0.73	0.72	0.75
korrekte n Klassen	1.83	**2.16**	**2.16**	**2.10**	**2.22**

Tabelle 5.3: Durchschnitt der Klassifikationsergebnisse über alle fünf Datensätze der Reuters Daten Die Ergebnisse der paarweisen Klassifikation finden sich in der linken Spalte und die Ergebnisse der paarweisen hierarchischen Klassifikation in der rechten Spalte.

Evaluationsmaß	**Pw. Kl.**	**Pw. Hier. Kl.**
Precision	**0.60**	0.55
Recall	**0.59**	0.55
F1	**0.59**	0.55
Hamming Loss	0.03	0.03
Margin Loss	**10.52**	17.90
One Error	**0.14**	0.19
Rank Loss	**0.04**	0.09
Avg Precision	**0.74**	0.72
korrekte n Klassen	2.00	**2.09**

Dies bedeutet, dass in den Rankings der paarweisen hierarchischen Klassifikation im Durchschnitt mindestens eine der relevanten Klassen relativ weit hinten im Ranking liegen muss. Selbst für den Fall, dass der Top Rank eine irrelevante Klasse wäre, ist es trotzdem notwendig, dass im Schnitt eine relevante deutlich abgeschlagen von der Spitze des Rankings liegt, um den durchschnittlichen Wert des „Margin Loss“ von fast 18 zu erreichen.[19] Aus der Betrachtung der Ergebnisse des Maßes „Hamming Loss“, welche bei beiden Varianten und allen Datensätzen sehr nahe beieinander liegen, kann man zusätzlich schließen, dass die Anzahl der korrekt eingeordneten Klassen bei der hierarchischen Methode in den konkreten Fällen durchschnittlich wohl nur um ein bis zwei Klassen niedriger ist.[20] Vermutlich liegen in den hierarchischen Rankings also einige wenige relevante Klassen oft relativ weit hinten.
Gleichzeitig liegt der Wert „korrekte n Klassen“ bei allen, außer dem ersten, Datensatz bei der paarweisen hierarchischen Klassifikation im Vergleich höher, was impliziert, dass die allerersten Plätze des Rankings öfter nur von, oder von mehr, relevanten Klassen belegt sind, als bei es bei der paarweisen Klassifikation der Fall ist. Der Wert „One Error“[21] liegt bei der paarweisen hierarchischen Klassifikation dagegen im Schnitt 5 Prozentpunkte höher, was in Kombination mit der vorherigen Feststellung impliziert, dass die hierarchische Methode zwar öfters einen falschen Top Rank vorhersagt, was einem relativ gewichtigen Fehler im Ranking entspricht, aber dafür anscheinend bei den anderen Rankings teilweise deutlich bessere Ergebnisse liefern muss, um unterm Strich dennoch den höheren Durchschnittswert bei „korrekte n Klassen“ zu erreichen.

Um diese aus den Werten abgeleiteten Vermutungen zu überprüfen, haben wir eine zufällige Teilmenge der Rankings beider Methoden, welche für Testinstanzen der Datensätze erstellt wurden, unter diesen Gesichtspunkten vergleichend betrachtet:
Zunächst ließ sich feststellen, dass sich bei der paarweisen hierarchischen Klassifikation tatsächlich im Vergleich öfter und auch stärker ausgeprägtere Ausreißer, also relevante Klassen, welche relativ weit hinten im Ranking liegen, finden. Als Beispiel für die Schwere einiger dieser Ausreißer, dient die Tatsache, dass sich bei den vierzig untersuchten Rankings neun Fälle fanden, bei denen mindestens eine der relevanten Klassen niedriger als Platz 40 eingeordnet wurde. Es konnte allerdings kein zuverlässiges Muster über die Natur dieser abgeschlagenen Klassen gefunden werden; tendenziell könnte man nur vermuten, dass es eher Klassen aus den Blättern des Baumes sind, bzw. Klassen, welche für diese Instanz nur irrelevante Kinderklassen haben.

Teilweise sind Rankings der paarweisen hierarchischen Klassifikation sogar so aufgebaut, dass sie auf den ersten Plätzen bessere Ergebnisse produzieren, dann aber eine einzelne relevante Klasse sehr weit hinten eingeordnet wird. Diese Rankings erhalten im Vergleich zu denen der paarweisen Klassifikation bessere Precision und Recall Werte, sowie ein höheres „n korrekte Klassen“, jedoch deutlich schlechtere Werte bei „Margin Loss“ und „Rank Loss“. Ein solches Beispiel ist in Abbildung 5.12 auf der rechten Seite zu sehen.

Es gibt weiter einige Rankings, welche bei beiden Methoden gleichermaßen perfekt oder nahezu perfekt ausfallen. In ein paar Fällen, bei denen beide Methoden gut abschneiden, konnte die paarweise hierarchische Klassifikation das bessere Ergebnis produzieren. Einer dieser Fälle ist auf der linken Seite der Abbildung 5.12 zu sehen. Die generell zu betrachtende Eigenschaft der hierarchischen Methode Klassen, welche in der Klassenhierarchie nahe beieinander sind, insbesondere Klassen zusammen mit ihren Superklassen, auch im Ranking öfter als bei der paarweisen Klassifikation als „Einheit“ zu behandeln, scheint bei diesen Fällen vorteilhaft zu sein. So sind die im Beispiel relevanten Klassen allesamt in einem Strang des Hierarchiebaumes angeordnet, dass heißt es ist eine Blattklasse mit allen ihren

[19] Die Instanzen haben im Schnitt ca. 4 relevante Klassen. Das heißt, das der Margin Loss durchschnittlich selbst bei falschem Top Rank „nur“ 4 betragen könnte, wenn nämlich alle relevanten Klassen direkt hinter dem Top Rank eingeordnet sind.
[20] „Hamming Loss“ misst den Anteil der inkorrekt eingeordneten Klassen von allen Klassen
[21] „One Error“ gibt die Prozentzahl der Testinstanzen an, bei denen der Top Rank keine relevante Klasse ist.

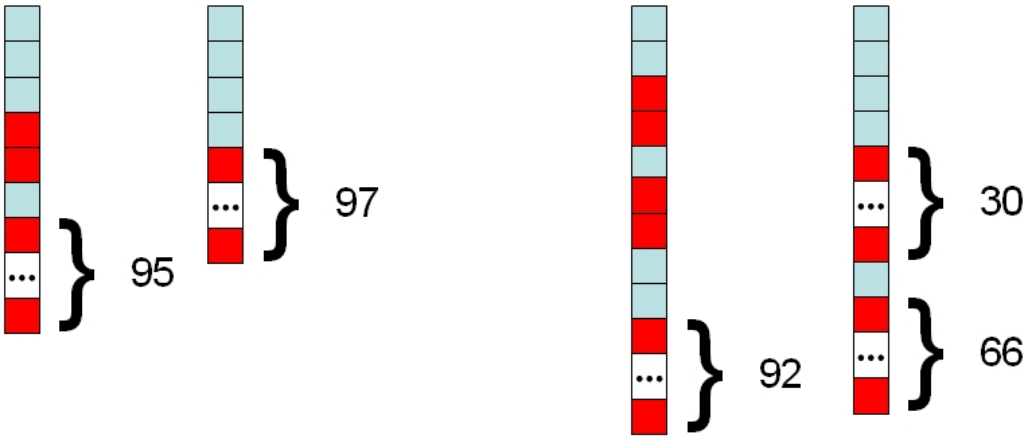

Abbildung 5.12: Zwei Rankings der Testinstanzen des Reuters Datensatzes.
Blaue Kästen stellen relevante Klassen da; rote Kästen sind irrelevante Klassen. Zahlen stellen eine Zusammenfassung von aufeinanderfolgenen Klassen eines Types dar. Das Ranking beginnt oben mit dem Top Rank. Das jeweils linke Ranking der beiden Paare ist das der paarweisen Klassifikation. Jeweils rechts ist das Ranking der paarweisen hierarchischen Klassifikation.

Superklassen. Als Folge dieser Eigenschaft ergeben sich zwei weitere Punkte. Unsere unter 3.4 formulierte Vermutung, dass die paarweise hierarchische Klassifikation die Klassen, welche den relevanten Klassen hierarchisch nahe sind, im Ranking vergleichsweise höher einstufen wird als die paarweise Klassifikation, wodurch ein eventueller Fehler bei der Vorhersage einen geringeren hierarchischen Fehler darstellen könnte, hat sich in vielen der konkreten Rankings angedeutet. Weiter zeigt sich in einigen der betrachteten Rankings auch ein Nachteil dieser „Clusterung“ von hierarchienahen Klassen: bei einigen falsch klassifizierten Instanzen hatte dies zur Folge, dass die ersten Plätze des Ranking mit einer ganzen „Klassenfamilie“[22] von irrelevanten Klassen belegt waren. Dieser Fäll war öfter zu beobachten bei Instanzen, für welche beide Methoden einen falschen Top Rank vorhersagten. Bei der paarweisen Klassifikation kamen meist direkt nach dem falschen Top Rank die relevanten Klassen im Ranking während die paarweise hierarchische Klassifikation dagegen oft noch weitere mit der falschen Top Rank Klasse verwandten Klassen an der Spitze des Rankings aufführte.

Wie aus den Werten der Tabelle bereits klar wurde, lieferte die paarweise Klassifikation durchschnittlich die besseren Ergebnisse. Bei den betrachteten Testinstanzen scheint es so, dass der Vorteil der paarweisen Klassifikation meist darin lag, bei Rankings, in denen beide Verfahren gewisse Probleme haben, bzw. beide nicht sehr gut klassifizieren, die besseren Ergebnisse zu produzieren. Ein Beispiel eines solchen Falles sind die beiden Rankings auf der rechten Seite der Abbildung 5.13. Wir fanden gleichzeitig kein Ranking, bei dem ein sehr gutes Klassifikationsergebnis der paarweisen Klassifikation einem schlechten Klassifikationsergebnis der paarweisen hierarchischen Klassifikation gegenüberstand.

Allgemein musste man bei der Betrachtung der konkreten Rankings feststellen, dass die beiden Methoden in der Praxis neben gleichen oder sehr ähnlichen Ergebnissen auch teilweise völlig unterschiedliche

[22] Gemeint ist eine Blattklasse mit ihren Superklassen und einigen weiteren hierarchisch sehr nahen Klassen

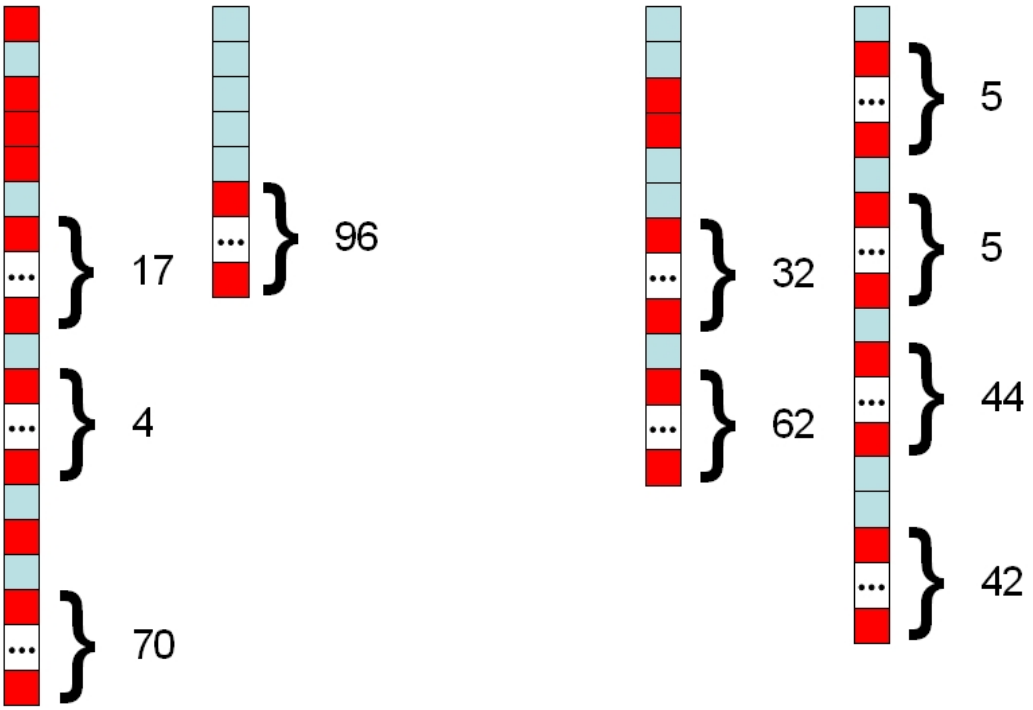

Abbildung 5.13: Zwei Rankings der Testinstanzen des Reuters Datensatzes.
Blaue Kästen stellen relevante Klassen da; rote Kästen sind irrelevante Klassen. Zahlen stellen eine Zusammenfassung von aufeinanderfolgenen Klassen eines Types dar. Das Ranking beginnt oben mit dem Top Rank. Das jeweils linke Ranking der beiden Paare ist das der paarweisen Klassifikation. Jeweils rechts ist das Ranking der paarweisen hierarchischen Klassifikation.

Klassenvorhersagen machten. So finden sich einzelne Klassen bei einem der beiden Rankings an der Spitze während dieselbe Klasse bei dem anderen Ranking völlig abgeschlagen in der Mitte der Rangfolge der 101 Klassen liegt. Ebenso sind ganze Rankings derselben Instanz teilweise sehr verschieden voneinander. Ein scheinbar unerklärliches Beispiel dafür ist in Abbildung 5.13 auf der linken Seite zu sehen. Die paarweise hierarchische Klassifikation stellt hierbei ein perfektes Ranking für eine Instanz, für welche die paarweise Klassifikation ein offensichtlich sehr fehlerhaftes Ranking produzierte.

Die Implikationen der Ergebnisse aus den Versuchen konnten also bei den konkreten Rankings durchweg bestätigt werden. Es bleibt jedoch vor allem die interessante Frage offen, ob sich konkrete Eigenschaften der Instanzen identifizieren lassen könnten, welche bedingen, dass eine der beiden Methoden in ihren Klassifikationsergebnissen substanzielle Unterschiede zur anderen aufweist.

5.3 Test auf Hierarchietreue

Wie in den Versuchsergebnissen auf den künstlichen Datensätzen deutlich wurde, ist die Frage, ob, bzw. in welchem Umfang, die konkreten Daten eine Klassenhierarchie widerspiegeln kritisch für die Performanz der paarweisen hierarchischen Klassifikation. Auch unabhängig von den Versuchergebnissen ist dieser Zusammenhang offensichtlich, da die Korrektheit einer Hierarchie ja Basis der Idee der paarweisen hierarchischen Klassifikation ist. Bei den künstlichen Datensätzen konnte die Umsetzung der Hierarchie bei der Generierung der Instanzen direkt gesteuert werden. Auch die Tatsache, dass der

Datensatz mit Single Label Instanzen, mit jeweils nur zwei Attributen, erstellt wurde, macht die Überprüfung der Hierarchietreue intuitiv und relativ anschaulich. Ganz anders verhält es sich allerdings bei den verwendeten Reuters Datensätzen. Die hohe Anzahl an Attributen und vor allen die Zugehörigkeit der Instanzen zu mehreren Klassen, welche teilweise auch aus verschiedenen Teilbäumen der Hierarchie stammen, macht eine „menschliche“ Einschätzung der Umsetzung der Hierarchie in den Daten nahezu unmöglich.

Um die Ergebnisse der Versuche besser interpretieren zu können, ist es von Interesse zu wissen, inwiefern die Instanzen der hierarchischen Struktur entsprechen. Bei Unwissen über die Beschaffenheit der Daten, ist es nicht möglich die Qualität der Ergebnisse der Klassifikation eindeutig dem Klassifikationsverfahren zuzuschreiben, da es unter Umständen falsche Annahmen über die Daten gemacht hat.

Einen Hinweis auf die Hierarchietreue eines Datensatzes kann man mit folgendem Verfahren erhalten: Auf den Trainingsdaten werden mit normaler paarweiser Binarisierung Klassifizierer für jedes Klassenpaar gelernt; soll ein solcher Klassifizierer $K_{A,B}$ nun eine Instanz einer dritten Klasse C kategorisieren, dann kann man erwarten, dass er sich generell öfter für die Klasse entscheidet, welche der Klasse C ähnlicher ist. Entsprechend würde man bei hierarchischen Daten erwarten, dass $K_{A,B}$ sich bei Instanzen der Klasse C überdurchschnittlich oft für die Klasse A entscheidet genau dann wenn A und C sich in der Hierarchie näher sind als B und C. Dieses Verhalten haben wir auf allen verwendeten Datensätzen getestet.
Es wurde konkret wie folgt vorgegangen. Auf den Trainingsdaten wurden nach paarweiser Klassifikation die binären Basisklassifizierer erlernt. Danach wurde jeder dieser Basisklassifizierer auf allen Testinstanzen angewendet und jeweils notiert, ob die Vorhersage der hierarchischen Erwartung entspricht; also ob für eine Testinstanz x, welche der Klasse C zugeordnet ist, gilt:

$$V(A,C) > V(B,C) \rightarrow K_{A,B}(x) = A$$

Festgehalten für jeden Datensatz wurde dann der Anteil der Fälle, in denen obiges zutrifft, von allen Fällen, für die diese Untersuchung möglich ist. Das heißt einige Fälle können nicht mit in Betracht gezogen werden. Betrachten wir beispielsweise eine Instanz y der Klasse Y, für die $V(A,Y) = V(B,Y) = 0$ gilt; in diesem Fall kann aus der Vorhersage des Klassifizierers $K_{A,B}(y)$ keinerlei Schluss auf das Vorhandensein der Hierarchie in den Daten gezogen werden. Formalisieren wir also die Fälle, welche in die Wertung miteinfließen:
Der paarweise Klassifizierer sei jeweils $K_{A,B}$ und die fragliche Instanz sei x mit der zugeordneten Klasse X.
Single Label: Das Ergebnis von $K_{A,B}$ für x ist relevant genau dann wenn

$$(V(A,X) > V(B,X) \vee V(A,X) < V(B,X)) \wedge A \neq X \wedge B \neq X$$

Multi Label: Bei Multi Label Datensätzen ist es wie gewohnt etwas komplizierter. Die Instanz x hat hierbei eine Menge X an zugeordneten Klassen. Zunächst müssen die folgende Bedingungen erfüllt sein, damit $K_{A,B}(x)$ zur Bewertung in Frage kommt:

$$\forall i \in X.(A \neq i \wedge B \neq i)$$

Als das nach der Hierarchie erwartete Ergebnis von $K_{A,B}(x)$ verstehen wir:

$$K_{A,B}(x) = A \Longleftrightarrow \forall i \in X.V(A,i) \geq V(B,i) \wedge \exists i \in X.V(A,i) > V(B,i)$$

$$K_{A,B}(x) = B \Longleftrightarrow \forall i \in X.V(B,i) \geq V(A,i) \wedge \exists i \in X.V(B,i) > V(A,i)$$

Das bedeutet, dass mindesten eine Klasse der Instanz x hierarchisch näher an A oder B sein muss und gleichzeitig keine Klasse von x das Gegenteil, nämlich der jeweils anderen Klasse hierarchisch näher zu sein, erfüllt. Instanzen, die dies nicht erfüllen, werden bei diesem Test ignoriert.

Die Eigenschaft, dass ein Klassifizierer bei hierarchischen Daten einer Instanz diejenige Klasse zuordnet, welche ihrer echten Klasse hierarchisch am nächsten ist, lässt sich bei einfachen Datensätzen, welche 2-dimensional dargestellt werden können, einfach und nachvollziehbar vorstellen.[23] Es ist aber zu bedenken, dass wir nicht zwingend davon ausgehen können, dass diese „Logik“ bei komplexeren Datensätzen so zutrifft. Haben Instanzen eines Datensatzes sehr viele, eventuell auch unterschiedlich ausgeprägte[24] Attribute, dann kann eine bestimmte Unterscheidung zwischen zwei Klassen auf nur wenigen dieser Attribute basieren.[25] Gleichzeitig kann eine dieser Klassen mit einer ihr hierarchisch nahen Klasse zwar viele Gemeinsamkeiten haben, welche aber in ganz anderen, vom Klassifizierer nicht berücksichtigten Attributen liegt, so dass der Klassifizierer bei dem hier vorgeschlagenen Hierarchietest trotz vorhandener Hierarchie oft nicht unserer Erwartung nach entscheidet.
Andererseits ist es als unwahrscheinlich anzusehen, dass die hier überprüfte Eigenschaft von einem nicht-hierarchischen Datensatz gut erfüllt wird. So dass uns dieser Hierarchietest doch einen gewissen Hinweis, dessen Genauigkeit allerdings unter Umständen variieren könnte, über die Hierarchietreue von Daten geben kann.

Für die in den Tabellen aufgeführten Ergebnisse wurden dieselben Klassifikationsparameter wie bei den vorherigen Versuchen verwendet und als Basisklassifizierer kommt ebenfalls die Support Vector Maschine SMO von WEKA zum Einsatz. Bei den künstlichen Datensätzen wurde jeder Fall auf 20 Datensätzen getestet und die Ergebnisse sind der jeweilige Durchschnitt. Die Testinstanzen haben jeweils dieselbe Stufe an Streuung wie die entsprechenden Trainingsinstanzen.

Tabelle 5.4: Hierarchietreue der künstlichen Datensätze

Datensatz	**Stufe der Streuung**				
Semi-hierarchisch	**1**	**2**	**3**	**4**	**5**
Normal	0.775	0.746	0.735	0.709	0.649
Vertausche Klassen	0.739	0.713	0.7	0.674	0.616
Hierarchisch	**1**	**2**	**3**	**4**	**5**
Normal	0.999	0.996	0.994	0.986	0.937
Vertausche Klassen	0.826	0.824	0.821	0.816	0.8

Für die künstlichen Datensätze mit hierarchischem Aufbau ist zu erkennen, dass die überprüfte Eigenschaft in allen Fällen nahezu perfekt zutrifft. Selbst bei der sehr starken Streuung liegt der Wert noch bei über 90%. Bei den Datensätzen, welche durch das Vertauschen zweier Klassen eine bewusste Verletzung der Hierarchie inne haben, liegen die Werte erwartungsgemäß niedriger bei einen Verlust von ca. 13 bis 17 Prozentpunkten, was in etwa den Fällen entsprechen sollte, die von dem Klassentausch direkt betroffen sind. Interessanterweise sind die Ergebnisse in beiden Fällen nur relativ gering davon beeinflusst wie stark die Streuung der Instanzen ausgeprägt ist, so dass die Differenz der ersten und letzten Streuungsstufe bei den normalen Daten bei ca. 6 Prozentpunkten und bei den Daten mit vertauschten Klassen bei ca. 2 Prozentpunkten liegt.

Bei den semi-hierarchisch aufgebauten Datensätzen liegen die Werte deutlich unter denen der hier-

[23] So wie bei den Beispielen aus 3.4 und 5.1.2
[24] Zum Beispiel können Attribute ganz unterschiedliche Wertemengen haben
[25] Vergleiche hierzu die Idee der Pachinko Maschine unter 4.1

archischen Datensätze, was selbstverständlich unserer Erwartung entspricht, weil die Hierarchie bei diesen Datensätzen auch nicht vollständig umgesetzt wurde. Der zusätzliche Verlust durch den vorgenommen Klassentausch wirkt sich nur relativ gering mit einem durchgängigen Verlust von ca. 3.5 Prozentpunkten aus. Die Werte fallen mit erhöhter Streuung der Instanzen klar ab bis zu Werten von knapp über 0.6, was nicht mehr weit über der Erwartung[26] eines nichthierarchischen Datensatzes liegt. Dieser Effekt lässt sich wohl dadurch erklären, dass die Instanzenmengen bei diesen Datensätzen im Gegensatz zu den hierarchischen Datensätzen viel näher aneinander liegen und so durch die zunehmende Streuung die Grenzen zwischen den Klassen stark verschwimmen, wodurch die hierarchische Struktur ebenfalls durcheinander kommt.

Es stellt sich nun die Frage, ob wir beim Vergleich der Hierarchietreuewerte und den Klassifikationsergebnissen der paarweisen hierarchischen Klassifikation direkte Zusammenhänge feststellen können. Intuitiv würden wir erwarten, dass die Ergebnisse der Klassifikation umso besser ausfallen, desto genauer die Hierarchie in den Daten umgesetzt ist, was hierbei mit der Hierarchietreue gemessen wurde. Zunächst ist einfach zu erkennen, dass sowohl die Werte der Hierarchietreue als auch die Ergebnisse der Klassifikation beide bei zunehmendem Rauschen abfallen. Dies ist für sich allerdings noch kein Hinweis auf einen direkten Zusammenhang, weil wir dieses Verhalten bei beiden Werten unabhängig voneinander erwarten würden: die Ergebnisse einer Klassifikation werden schlechter, wenn man die Daten zunehmend verrauscht und ebenso verliert die Hierarchie in den Daten bei hohem Rauschen ihre Struktur. Um genaueres zu erfahren, müssen wir die beiden Ergebnisse also näher betrachten.
In Abbildung 5.14 sind die Werte für die semi-hierarchischen Datensätze gegeneinander aufgetragen. Es lässt sich erkennen, dass die Klassifikationsergebnisse der normalen Daten und der Daten mit vertauschten Klassen ab der dritten Streuungstufe sehr ähnlich sind, obgleich die Werte der Hierarchietreue auf hierarchische Unterschiede der beiden Datensätze hinweisen. Man kann wohl vermuten, dass der Effekt des Rauschens ab einer bestimmten Stufe einen stärkeren Einfluss auf die paarweise hierarchische Klassifikation hat als der hierarchische Fehlers; dies deckt sich auch mit unseren Beobachtungen auf den Klassifikationsergebnissen unter 5.1.4. Für die ersten zwei Stufen der Streuung verhalten sich die Ergebnisse jedoch nahezu linear [27] zu den gemessenen Hierarchietreuewerten. Abbildung 5.15 beinhaltet die gleiche Gegenüberstellung für die hierarchischen Datensätze. Hierbei können wir einen ähnlichen Effekt wie bei den semi-hierarchischen Daten erkennen: die fünfte Streuungsstufe zeigt bereits die Tendenz, dass die Klassifikationsergebnisse bei weiterem Rauschen voraussichtlich stark abfallen werden, wobei die Hierarchietreue relativ dazu tendenziell nicht so stark nachlassen sollte. Gleichzeitig verhalten sich in diesem Fall die Klassifikationswerte der ersten vier Streuungsstufen fast linear zu den entsprechenden Hierarchietreuewerten.
Es kann zusammenfassend gesagt werden, dass sich durchaus ein Zusammenhang zwischen dem hier beschriebenen Maßes für Hierarchietreue und den Ergebnissen der paarweisen hierarchischen Klassifikation erkennen lässt, wobei jedoch nur für geringes Rauschen Rückschlüsse von einem der Werte auf den anderen möglich sind.

Die Werte auf den Reuters Datensätzen sehen zunächst einmal deutlich schlechter aus. Ein direkter Vergleich zwischen den Reuters Daten und den künstlichen Datensätzen ist jedoch auf Grund des komplett unterschiedlichen Aufbaus der Hierarchien und Instanzen wahrscheinlich nicht direkt sinnvoll; allerdings sind die Werte, welche allesamt 0.63 oder 0.64 betragen, durchaus ernüchternd, da sie nicht besonders weit über der Erwartung von 0.5 eines Datensatzes ohne hierarchische Struktur liegen. Andererseits spricht der Vergleich der Ergebnisse der paarweisen hierarchischen Klassifikation und der paarweisen Klassifikation auf diesen Daten dafür, dass es zumindest keine durchgehenden groben

[26] Wenn es keine Beziehungen zwischen den einzelnen Klassen gibt, sollte man erwarten, dass ein paarweiser Klassifizierer $K_{A,B}$ für eine beliebige Instanz, welche nicht der Klasse A oder B angehört, quasi zufällig mit gleicher Wahrscheinlichkeit A oder B vorhersagt. Dies würde zu Werten um 0.5 führen.

[27] Abweichungen zeigen sich ab der 2. Nachkommastelle

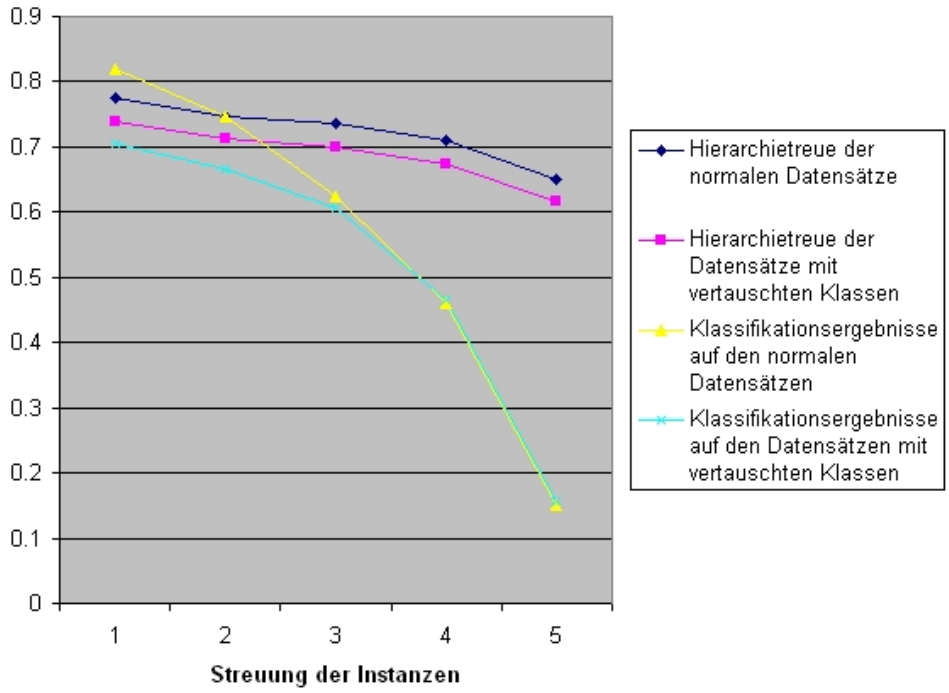

Abbildung 5.14: Gegenüberstellung der Hierarchietreue und den Ergebnissen der paarweisen hierarchischen Klassifikation auf den semi-hierarchischen Datensätzen.

Tabelle 5.5: Hierarchietreue der Reuters Datensätze

Datensätze	**Hierarchietreue**
1	0.63
2	0.64
3	0.64
4	0.63
5	0.63

hierarchischen Widersprüche in den Daten gibt. Als vorsichtiges Fazit aus diesen Werten müsste man demnach wohl ziehen, dass die Hierarchie höchstwahrscheinlich nicht ganz konsistent in den Daten umgesetzt ist.

Unsere Vermutung, dass im ersten Datensatz die Hierarchie betreffende Unterschiede zu den anderen Datensätzen existieren, was das Ergebnis der paarweisen hierarchischen Klassifikation auf diesen Daten erklären könnte, lässt sich mit diesem Test nicht unterstützen. Die Frage, warum das hierarchische Verfahren auf dem ersten Datensatz im Vergleich unterdurchschnittlich ausfällt, bleibt damit weiterhin offen.

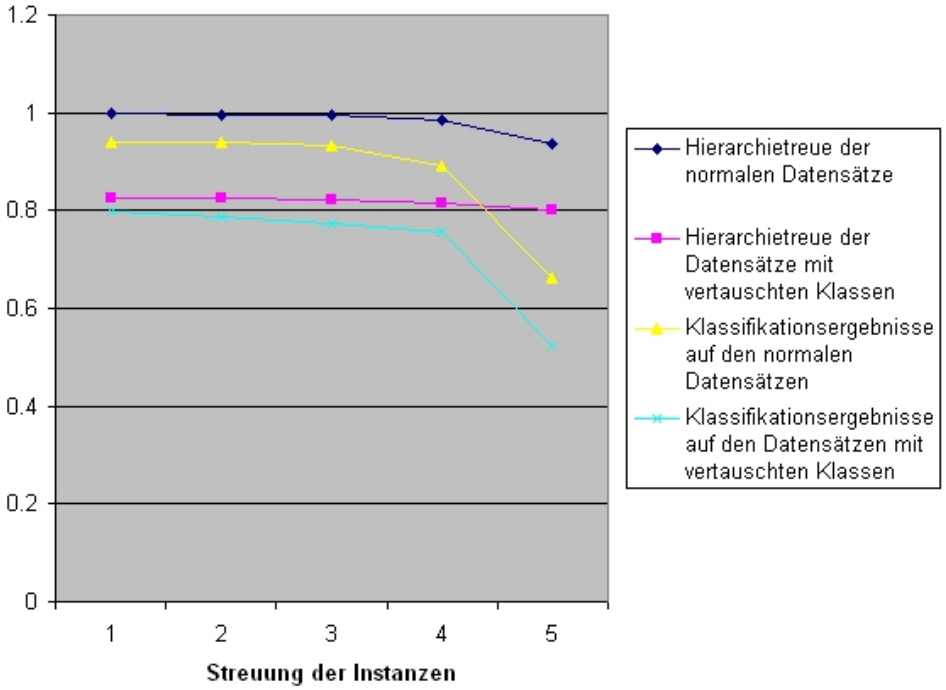

Abbildung 5.15: Gegenüberstellung der Hierarchietreue und den Ergebnissen der paarweisen hierarchischen Klassifikation auf den hierarchischen Datensätzen.

6 Fazit und Ausblick

Abschließend müssen wir feststellen, dass die paarweise hierarchische Klassifikation in der hier verwendeten Form bei allen Datensätzen und unter Betrachtung fast aller Evaluationsmaße der paarweisen Klassifikation unterlegen war. Sowohl auf den Daten des Reuters Datensatzes, welcher als Vertreter von praxisnahen Datensätzen gesehen werden kann, als auch auf den künstlichen Datensätzen, welche teilweise gemäß den theoretischen Idealbedingungen der paarweisen hierarchischen Klassifikation erstellt sind, liegen die Ergebnisse unseres hierarchischen Ansatzes unter denen der paarweisen Klassifikation. Insbesondere konnte eine stärkere Anfälligkeit für zunehmendes Rauschen bei den Werten der Instanzen bei der hierarchischen Klassifikation festgestellt werden. Zusätzlich stellen Inkonsistenzen zwischen der Klassenhierarchie und den Daten der Instanzen, welche grundsätzlich bei „echten“ Datensätzen nicht auszuschließen sind, Schwierigkeiten für das hierarchische Modell dar, welche sich auch deutlich in den Klassifikationsergebnissen zeigen. Im Gegensatz ist die paarweise Klassifikation gegenüber letzterer Gegebenheit quasi per Definition immun. Zusammenfassend zeigte die paarweise Klassifikation auf den hierarchischen Datensätzen, trotz der Tatsache, dass diese die Klassenhierarchie nicht verwendet, klar bessere und robustere Klassifikationsergebnisse.

Weiterhin wurde in dieser Arbeit eine Möglichkeit vorgestellt, das Vorhandensein einer Klassenhierarchie in den Instanzen eines Datensatzes zu überprüfen. Dieses Maß zeigte bei geringem Rauschen der Daten auf den künstlichen Datensätzen einen Zusammenhang mit den Klassifikationsergebnissen der paarweisen hierarchischen Klassifikation. Die Nützlichkeit des Maßes für die Verwendung auf Multi-Label Datensätzen müsste noch weiter überprüft werden, indem es neben den Reuters Datensätzen noch auf weiteren Multi-Label Datensätzen angewendet wird.

In dieser Arbeit konnte gezeigt werden, dass die paarweise hierarchische Klassifikation für Single Label Probleme und binäre Hierarchiebäume zu dem hierarchischen Klassifikationsmodell der Pachinko Maschine äquivalent ist. Diese Äquivalenz ist bedingt durch die Wahl des Maßes für die hierarchische Nähe von Klassen. Es wurde in diesem Fall der gemeinsame Weg von zwei Klassen im Hierarchiebaum verwendet, um Aussagen über deren hierarchische Nähe zu machen. Es wäre daher neben den anderen vorgeschlagenen Variationen der paarweisen hierarchischen Klassifikation vor allem interessant andere Maße für die Hierarchienähe zu testen und deren Konsequenzen auf das Klassifikationsverhalten des Modells zu untersuchen.

Es konnten neben der scheinbar generellen Unterlegenheit der paarweisen hierarchischen Klassifikation jedoch noch einige andere interessante Aspekte im Vergleich aufgedeckt werden. Bei den künstlichen Datensätzen des semi-hierarchischen Aufbaus konnte festgestellt werden, dass die paarweise hierarchische Klassifikation in ihren Klassifikationsergebnissen im Vergleich weniger abfiel, wenn die Trainingsinstanzen einiger der Klassen radikal verringert wurden. Für diesen Effekt wurde unter 3.4 auch eine mögliche theoretische Erklärung gegeben. Die weitere Untersuchung und Überprüfung dieses Effektes könnte interessant sein, um Fälle zu identifizieren, bei denen die paarweise hierarchische Klassifikation eventuell für eine Verbesserung der Ergebnisse verwendet werden kann. Eine relevante Frage, welche zu dieser Problemstellung gehören würde, ist es, warum dieser Effekt nicht bei den künstlichen Datensätzen mit hierarchischem Aufbau auftrat.
Bei der Betrachtung der Ergebnisse auf den Reuters Daten und einer stichprobenhaften Untersuchung

der konkreten Rankings, welche für die Testinstanzen erstellt wurden, fanden sich ein paar Fälle in denen die paarweise hierarchische Klassifikation einen Vorteil zu haben scheint. Bei einigen Testinstanzen konnte der hierarchische Ansatz scheinbar auf Grund der verwendeten Hierarchieinformation im Gegensatz zur paarweisen Klassifikation die relevanten Klassen kompakter an der Spitze des Rankings anordnen. Es fanden sich weiter noch einzelne Fälle, bei denen die hierarchische Klassifikation aus nicht offensichtlich erkennbaren Gründen auf Instanzen gute Ergebnisse lieferte, während die paarweise Klassifikation bei diesen Fälle verhältnismäßig schlechte Rankings erstellte.
Für eine weitere Beschäftigung mit diesem Thema wäre es folglich interessant, mögliche Eigenschaften von Instanzen oder eventuell sogar Datensätzen zu identifizieren, welche mit guten, bzw. im Vergleich besseren, Ergebnissen der paarweisen hierarchischen Klassifikation korrelieren. Mögliche Ansatzpunkte könnten hier vielleicht die Anzahl der relevanten Klassen einer Instanz und auch die hierarchischen Verhältnisse dieser Klassen untereinander sein. [1] Darauf aufsetzend ist es denkbar mit einem hybriden Modell aus paarweiser und paarweiser hierarchischer Klassifikation zu experimentieren. Unter 3.5 wurde bereits ein anderes hybrides Modell angedacht: beim Lernprozess könnte an Hand der Eigenschaften der Trainingsmengen des jeweiligen Klassenpaares entschieden werden, ob ein „normaler“ oder ein hierarchischen Basisklassifizierer erlernt werden soll. Statt die Entscheidung zwischen den Modellen beim Lernprozess zu treffen, könnte man, gegeben dass beide Typen von Basisklassifizierern vorher erlernt wurden, bei der Betrachtung der einzelnen Instanz entscheiden, welche Art der Klassifikation angewendet werden soll.

Ein weiterer Anknüpfungspunkt, welcher in dieser Arbeit nicht mehr miteinbezogen werden konnten, ist die Verwendung einer hierarchischen Evaluation, um unsere Vermutung, dass die paarweise hierarchische Klassifikation bei Fehlklassifikationen generell hierarchisch geringere Fehler produziert als die paarweise Klassifikation, genauer zu überprüfen.

[1] Diese Informationen sind bei unbekannten Instanzen allerdings natürlich nicht bekannt.

7 Anhang A

Die Hierarchie der Topics aus der Datei $rcv1.topics.hier.orig$ von der Internetseite [Lewis, 2004]:

parent: None child: Root child-description: No Description
parent: CCAT child: C11 child-description: STRATEGY/PLANS
parent: CCAT child: C12 child-description: LEGAL/JUDICIAL
parent: CCAT child: C13 child-description: REGULATION/POLICY
parent: CCAT child: C14 child-description: SHARE LISTINGS
parent: CCAT child: C15 child-description: PERFORMANCE
parent: C15 child: C151 child-description: ACCOUNTS/EARNINGS
parent: C151 child: C1511 child-description: ANNUAL RESULTS
parent: C15 child: C152 child-description: COMMENT/FORECASTS
parent: CCAT child: C16 child-description: INSOLVENCY/LIQUIDITY
parent: CCAT child: C17 child-description: FUNDING/CAPITAL
parent: C17 child: C171 child-description: SHARE CAPITAL
parent: C17 child: C172 child-description: BONDS/DEBT ISSUES
parent: C17 child: C173 child-description: LOANS/CREDITS
parent: C17 child: C174 child-description: CREDIT RATINGS
parent: CCAT child: C18 child-description: OWNERSHIP CHANGES
parent: C18 child: C181 child-description: MERGERS/ACQUISITIONS
parent: C18 child: C182 child-description: ASSET TRANSFERS
parent: C18 child: C183 child-description: PRIVATISATIONS
parent: CCAT child: C21 child-description: PRODUCTION/SERVICES
parent: CCAT child: C22 child-description: NEW PRODUCTS/SERVICES
parent: CCAT child: C23 child-description: RESEARCH/DEVELOPMENT
parent: CCAT child: C24 child-description: CAPACITY/FACILITIES
parent: CCAT child: C31 child-description: MARKETS/MARKETING
parent: C31 child: C311 child-description: DOMESTIC MARKETS
parent: C31 child: C312 child-description: EXTERNAL MARKETS
parent: C31 child: C313 child-description: MARKET SHARE
parent: CCAT child: C32 child-description: ADVERTISING/PROMOTION
parent: CCAT child: C33 child-description: CONTRACTS/ORDERS
parent: C33 child: C331 child-description: DEFENCE CONTRACTS
parent: CCAT child: C34 child-description: MONOPOLIES/COMPETITION
parent: CCAT child: C41 child-description: MANAGEMENT
parent: C41 child: C411 child-description: MANAGEMENT MOVES
parent: CCAT child: C42 child-description: LABOUR
parent: Root child: CCAT child-description: CORPORATE/INDUSTRIAL
parent: ECAT child: E11 child-description: ECONOMIC PERFORMANCE
parent: ECAT child: E12 child-description: MONETARY/ECONOMIC
parent: E12 child: E121 child-description: MONEY SUPPLY
parent: ECAT child: E13 child-description: INFLATION/PRICES

parent: E13 child: E131 child-description: CONSUMER PRICES
parent: E13 child: E132 child-description: WHOLESALE PRICES
parent: ECAT child: E14 child-description: CONSUMER FINANCE
parent: E14 child: E141 child-description: PERSONAL INCOME
parent: E14 child: E142 child-description: CONSUMER CREDIT
parent: E14 child: E143 child-description: RETAIL SALES
parent: ECAT child: E21 child-description: GOVERNMENT FINANCE
parent: E21 child: E211 child-description: EXPENDITURE/REVENUE
parent: E21 child: E212 child-description: GOVERNMENT BORROWING
parent: ECAT child: E31 child-description: OUTPUT/CAPACITY
parent: E31 child: E311 child-description: INDUSTRIAL PRODUCTION
parent: E31 child: E312 child-description: CAPACITY UTILIZATION
parent: E31 child: E313 child-description: INVENTORIES
parent: ECAT child: E41 child-description: EMPLOYMENT/LABOUR
parent: E41 child: E411 child-description: UNEMPLOYMENT
parent: ECAT child: E51 child-description: TRADE/RESERVES
parent: E51 child: E511 child-description: BALANCE OF PAYMENTS
parent: E51 child: E512 child-description: MERCHANDISE TRADE
parent: E51 child: E513 child-description: RESERVES
parent: ECAT child: E61 child-description: HOUSING STARTS
parent: ECAT child: E71 child-description: LEADING INDICATORS
parent: Root child: ECAT child-description: ECONOMICS
parent: GCAT child: G15 child-description: EUROPEAN COMMUNITY
parent: G15 child: G151 child-description: EC INTERNAL MARKET
parent: G15 child: G152 child-description: EC CORPORATE POLICY
parent: G15 child: G153 child-description: EC AGRICULTURE POLICY
parent: G15 child: G154 child-description: EC MONETARY/ECONOMIC
parent: G15 child: G155 child-description: EC INSTITUTIONS
parent: G15 child: G156 child-description: EC ENVIRONMENT ISSUES
parent: G15 child: G157 child-description: EC COMPETITION/SUBSIDY
parent: G15 child: G158 child-description: EC EXTERNAL RELATIONS
parent: G15 child: G159 child-description: EC GENERAL
parent: Root child: GCAT child-description: GOVERNMENT/SOCIAL
parent: GCAT child: GCRIM child-description: CRIME, LAW ENFORCEMENT
parent: GCAT child: GDEF child-description: DEFENCE
parent: GCAT child: GDIP child-description: INTERNATIONAL RELATIONS
parent: GCAT child: GDIS child-description: DISASTERS AND ACCIDENTS
parent: GCAT child: GENT child-description: ARTS, CULTURE, ENTERTAINMENT
parent: GCAT child: GENV child-description: ENVIRONMENT AND NATURAL WORLD
parent: GCAT child: GFAS child-description: FASHION
parent: GCAT child: GHEA child-description: HEALTH
parent: GCAT child: GJOB child-description: LABOUR ISSUES
parent: GCAT child: GMIL child-description: MILLENNIUM ISSUES
parent: GCAT child: GOBIT child-description: OBITUARIES
parent: GCAT child: GODD child-description: HUMAN INTEREST
parent: GCAT child: GPOL child-description: DOMESTIC POLITICS
parent: GCAT child: GPRO child-description: BIOGRAPHIES, PERSONALITIES, PEOPLE
parent: GCAT child: GREL child-description: RELIGION
parent: GCAT child: GSCI child-description: SCIENCE AND TECHNOLOGY

parent: GCAT child: GSPO child-description: SPORTS
parent: GCAT child: GTOUR child-description: TRAVEL AND TOURISM
parent: GCAT child: GVIO child-description: WAR, CIVIL WAR
parent: GCAT child: GVOTE child-description: ELECTIONS
parent: GCAT child: GWEA child-description: WEATHER
parent: GCAT child: GWELF child-description: WELFARE, SOCIAL SERVICES
parent: MCAT child: M11 child-description: EQUITY MARKETS
parent: MCAT child: M12 child-description: BOND MARKETS
parent: MCAT child: M13 child-description: MONEY MARKETS
parent: M13 child: M131 child-description: INTERBANK MARKETS
parent: M13 child: M132 child-description: FOREX MARKETS
parent: MCAT child: M14 child-description: COMMODITY MARKETS
parent: M14 child: M141 child-description: SOFT COMMODITIES
parent: M14 child: M142 child-description: METALS TRADING
parent: M14 child: M143 child-description: ENERGY MARKETS
parent: Root child: MCAT child-description: MARKETS

8 Anhang B

Folgend ist die Instanzenverteilung jeweils eines künstlichen Datensatzes für sowohl den semi-hierarchischen als auch den hierarchischen Aufbau mit variierter Streuung dargestellt. Die Bilder wurden mit dem WEKA-Explorer[1] erstellt.

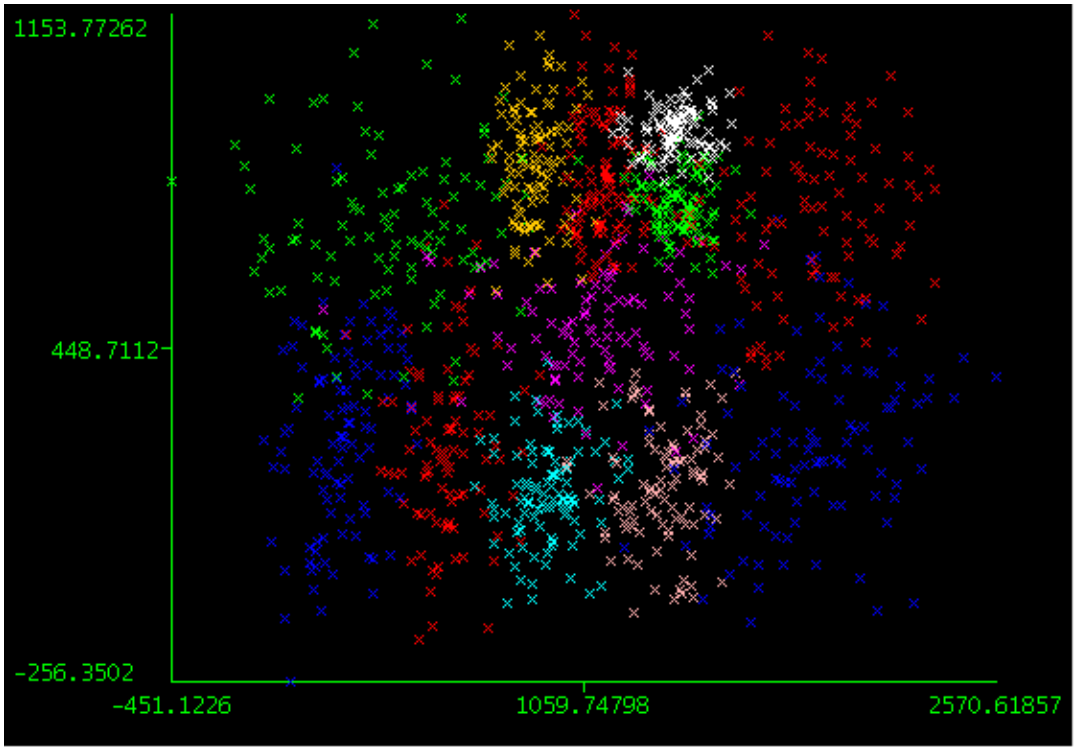

Abbildung 8.1: Instanzenverteilung in einem semi-hierarchischen Datensatz mit sehr geringer Streuung

[1] http://www.cs.waikato.ac.nz/ml/weka/

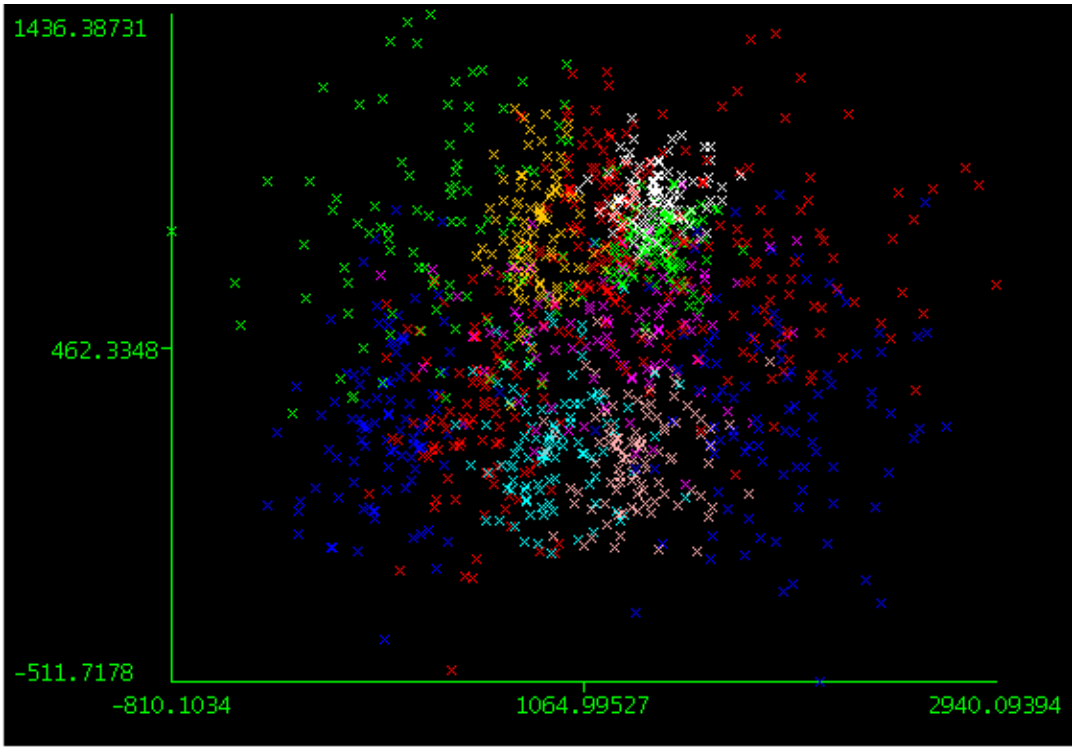

Abbildung 8.2: Instanzenverteilung in einem semi-hierarchischen Datensatz mit geringer Streuung

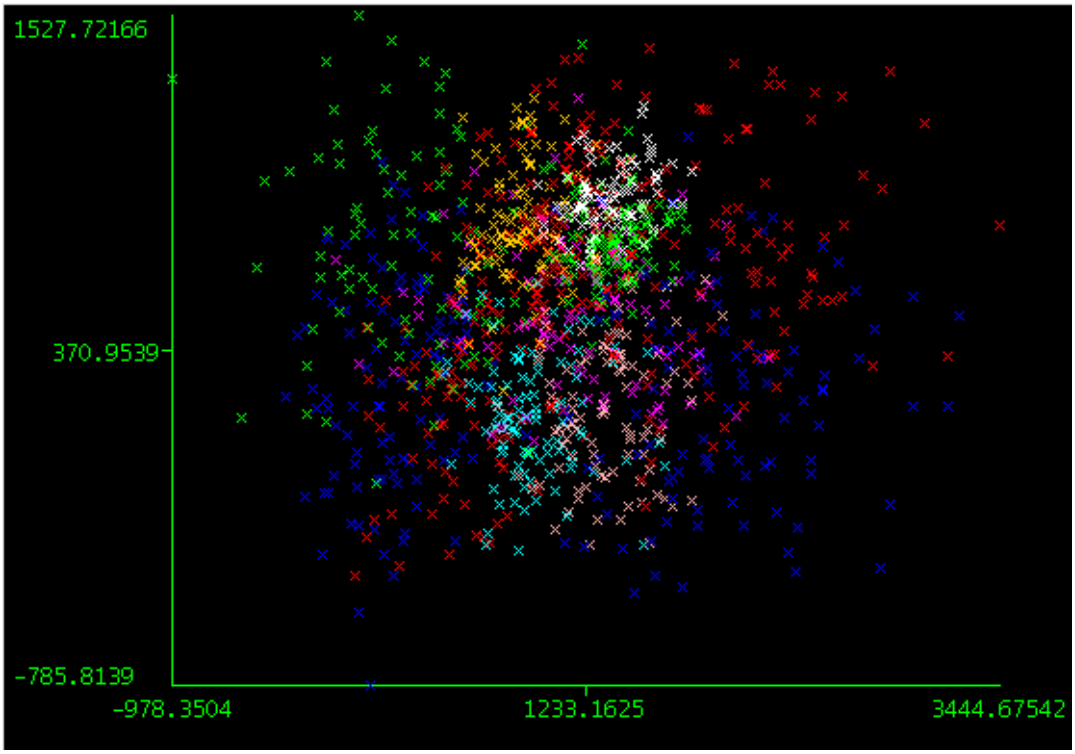

Abbildung 8.3: Instanzenverteilung in einem semi-hierarchischen Datensatz mit normaler Streuung

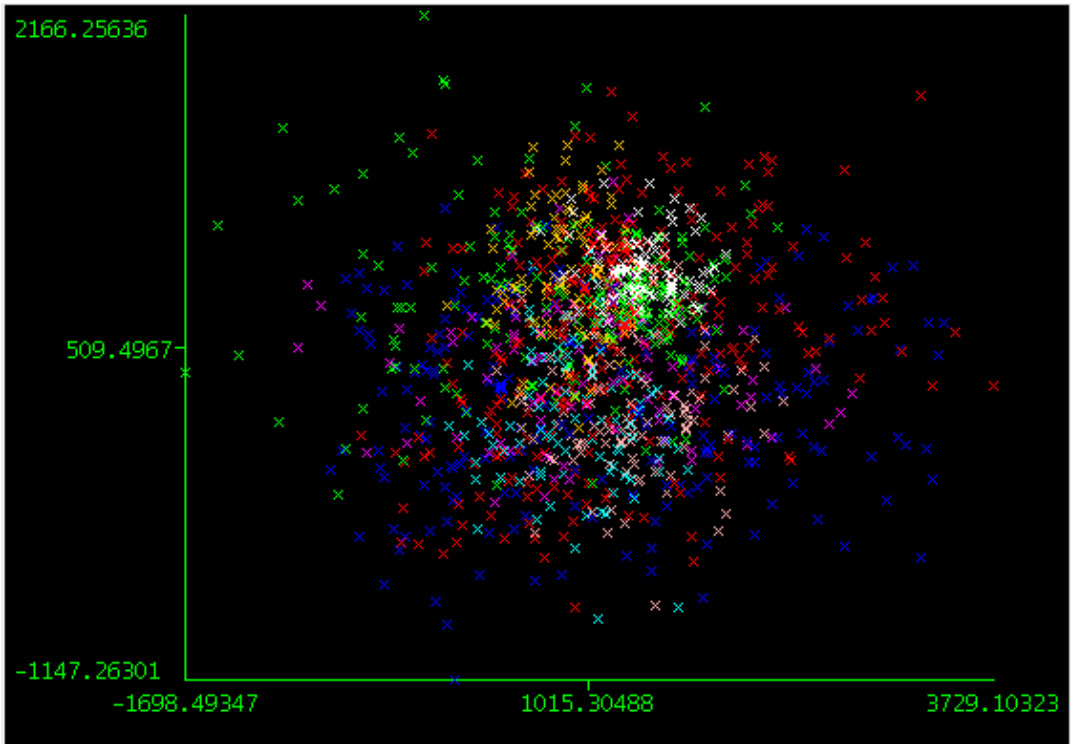

Abbildung 8.4: Instanzenverteilung in einem semi-hierarchischen Datensatz mit starker Streuung

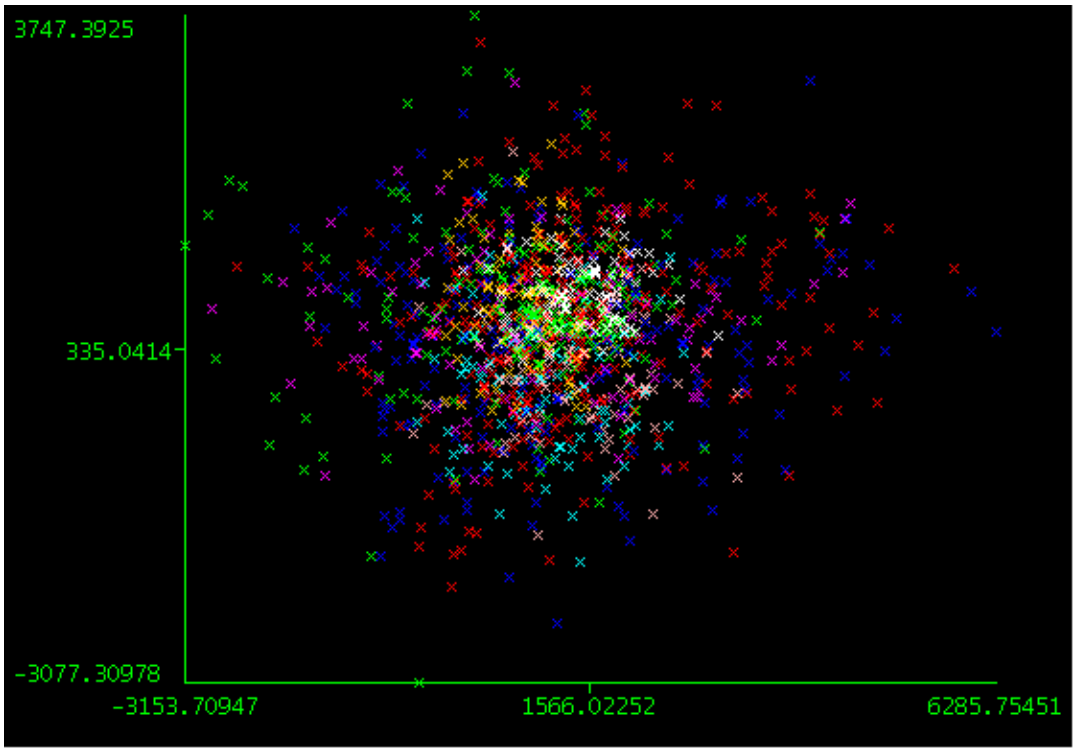

Abbildung 8.5: Instanzenverteilung in einem semi-hierarchischen Datensatz mit sehr starker Streuung

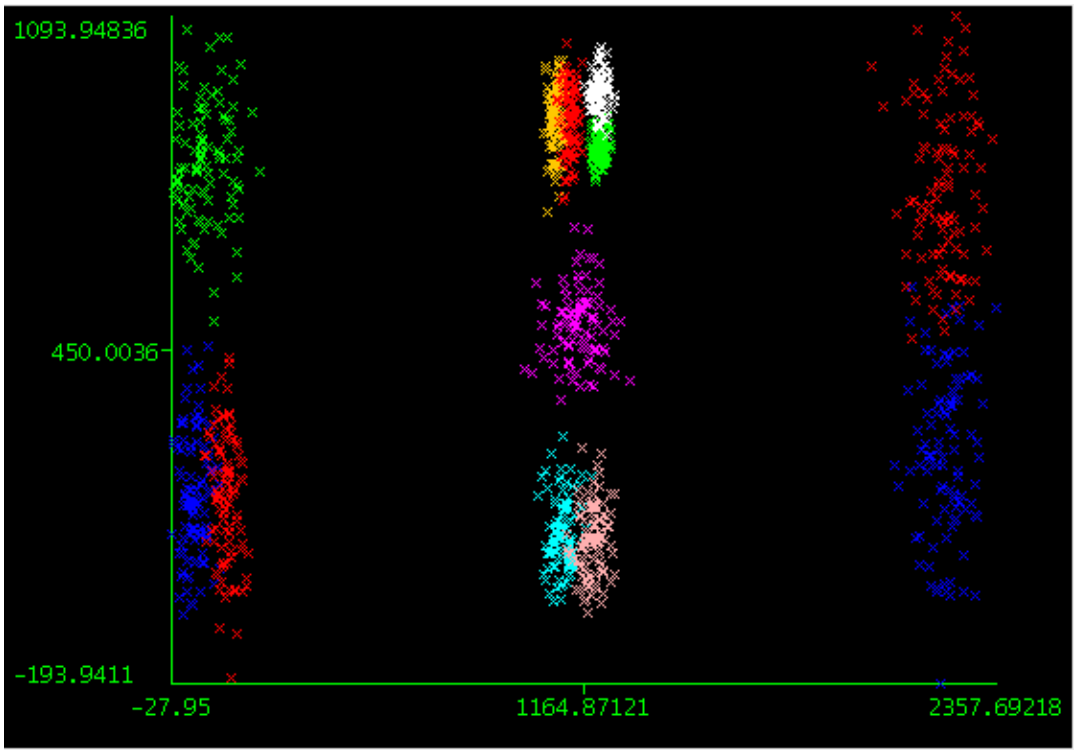

Abbildung 8.6: Instanzenverteilung in einem hierarchischen Datensatz mit sehr geringer Streuung

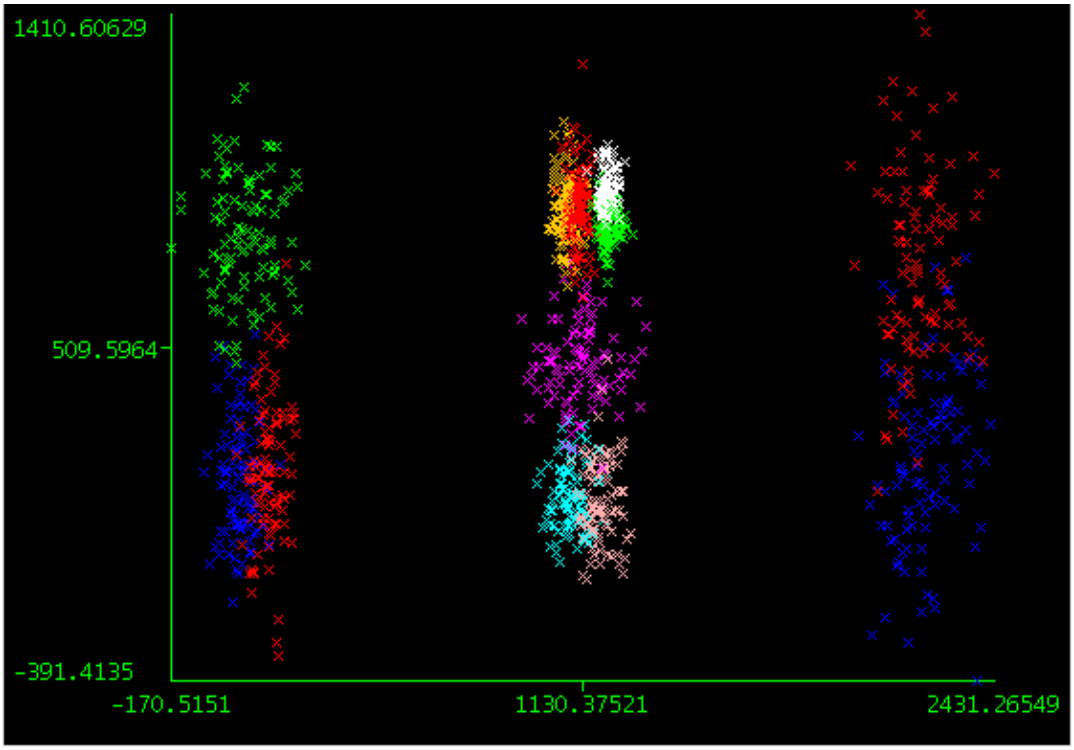

Abbildung 8.7: Instanzenverteilung in einem hierarchischen Datensatz mit geringer Streuung

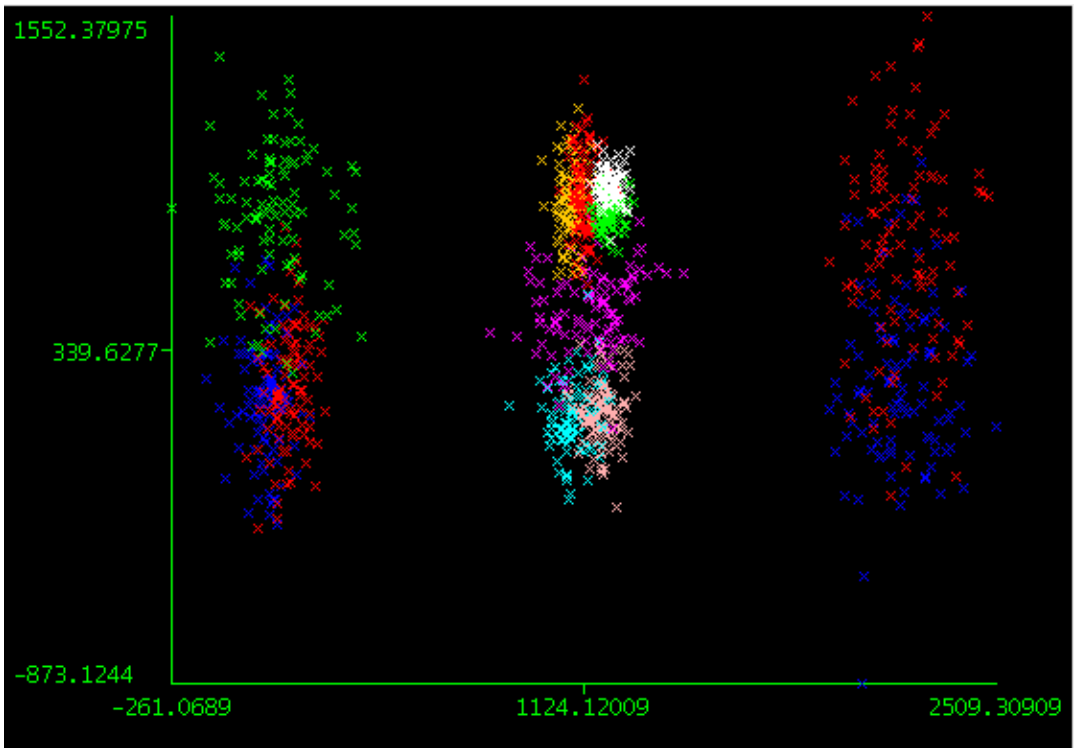

Abbildung 8.8: Instanzenverteilung in einem hierarchischen Datensatz mit normaler Streuung

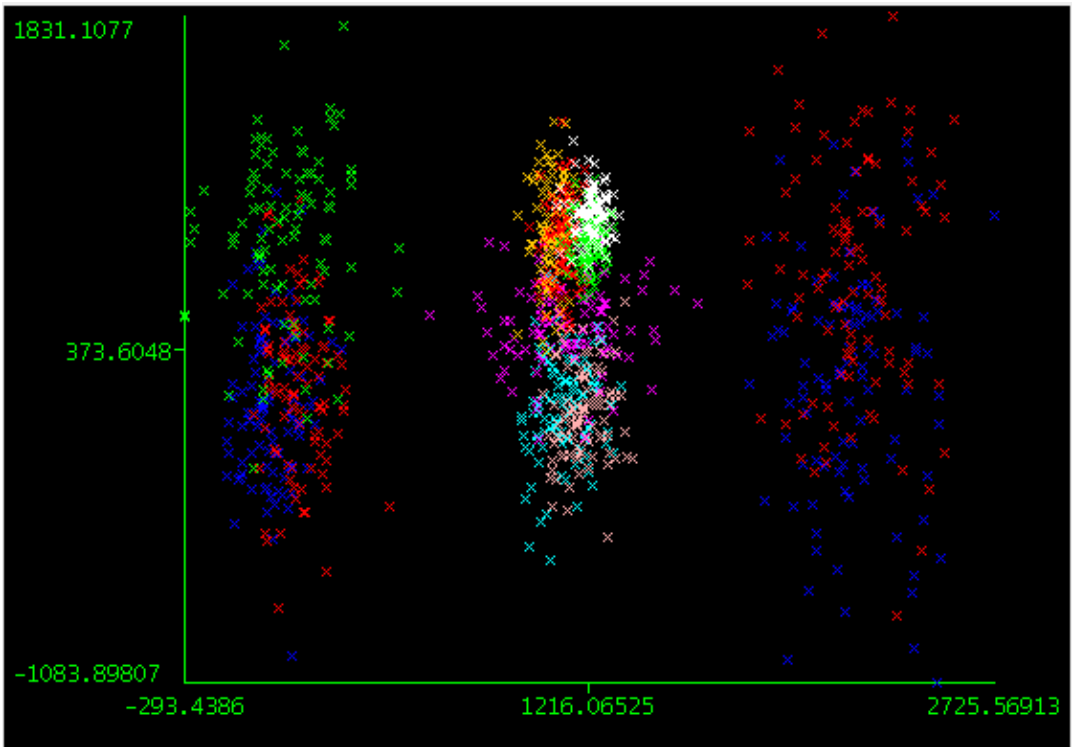

Abbildung 8.9: Instanzenverteilung in einem hierarchischen Datensatz mit starker Streuung

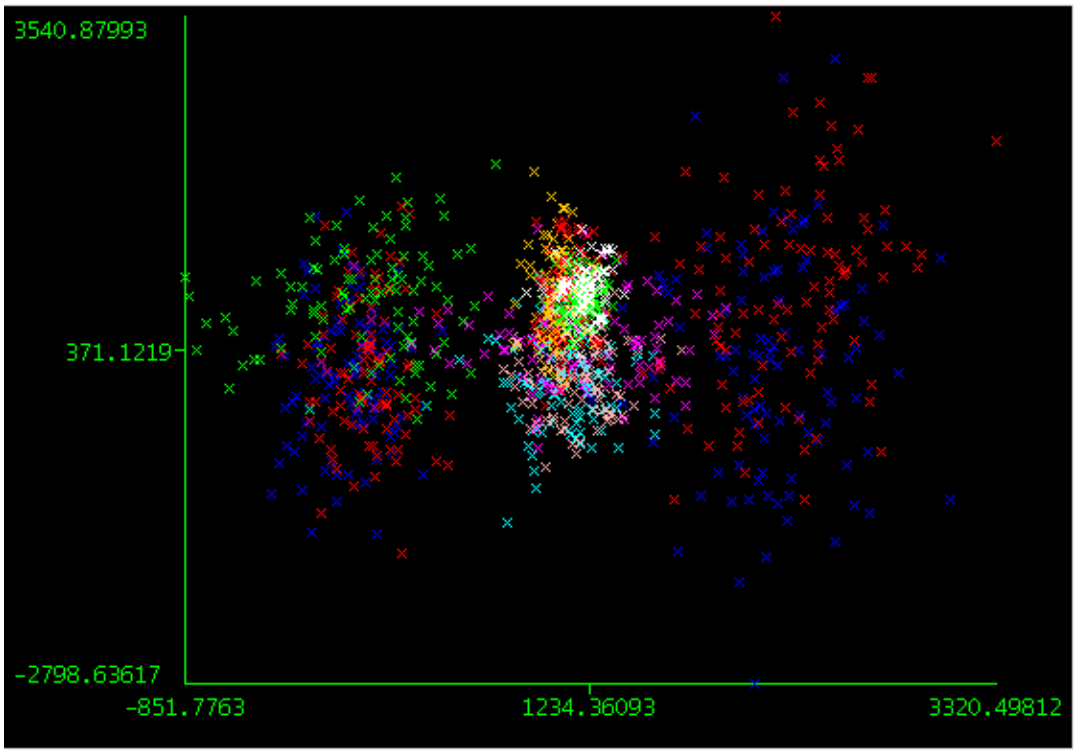

Abbildung 8.10: Instanzenverteilung in einem hierarchischen Datensatz mit sehr starker Streuung

Literaturverzeichnis

[Blockeel *et al.*, 2006] Hendrik Blockeel, Leander Schietgat, Jan Struyf, Amanda Clare, and Saso Dzeroski. Hierarchical multilabel classification trees for gene function prediction (extended abstract). *Workshop on Probabilistic Modeling and Machine Learning in Structural and Systems Biology*, 2006.

[Brinker *et al.*, 2006] Klaus Brinker, Johannes Fürnkranz, and Eyke Hüllermeier. A unified model for multilabel classification and ranking. *European Conference on AI*, pages 489–493, 2006.

[Cesa-Bianchi *et al.*, 2004] Nicolò Cesa-Bianchi, Claudio Gentile, Andrea Tironi, and Luca Zaniboni. Incremental algorithms for hierarchical classification. *Neural Information Processing Systems*, 2004.

[Cutzu, 2003] Florin Cutzu. Polychotomous classification with pairwise classifiers: A new voting principle. *Multiple Classifier Systems*, pages 115–124, 2003.

[Dekel *et al.*, 2004] Ofer Dekel, Joseph Keshet, and Yoram Singer. Large margin hierarchical classification. *International Conference on Machine Learning*, 2004.

[Fürnkranz, 2001] Johannes Fürnkranz. Round robin rule learning. *International Conference on Machine Learning*, pages 146–153, 2001.

[Hofmann *et al.*, 2003] T. Hofmann, L. Cai, and M. Ciaramita. Learning with taxonomies: Classifying documents and words, 2003.

[Hsu and Lin, 2002] Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multi-class support vector machines. Band 13, Nr. 2, pages 415–425, 2002.

[Hüllermeier and Fürnkranz, 2004] Eyke Hüllermeier and Johannes Fürnkranz. Comparison of ranking procedures in pairwise preference learning. *IPMU-04, International Conference on Information Processing and Management of Uncertainty of Knowledge-Based Systems. Perugua, Italy.*, 2004.

[Koller and Sahami, 1997] Daphne Koller and Mehran Sahami. Hierarchically classifying documents using very few words. *International Conference on Machine Learning*, pages 170–178, 1997.

[Lewis, 2004] David D. Lewis. Rcv1-v2/lyrl2004: The lyrl2004 distribution of the rcv1-v2 text categorization test collection (12-apr-2004 version)

[Lewis *et al.*, 2004] David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. Rcv1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, pages 5:361–397, 2004.

[McCallum *et al.*, 1998] Andrew McCallum, Ronald Rosenfeld, Tom M. Mitchell, and Andrew Y. Ng. Improving text classification by shrinkage in a hierarchy of classes. *International Conference on Machine Learning*, pages 359–367, 1998.

[Platt *et al.*, 1999] John C. Platt, Nello Cristianini, and John Shawe-Taylor. Large margin dags for

multiclass classification. *Neural Information Processing Systems*, pages 547–553, 1999.

[Salton and Buckley, 1988] Gerard Salton and Chris Buckley. Term-weighting approaches in automatic text retrieval. *Inf. Process. Manage.*, pages 24(5):513–523, 1988.

[Schölkopf and Smola, 2002] Bernhard Schölkopf and Alex J. Smola. A short introduction to learning with kernels. *Machine Learning Summer School*, pages 41–64, 2002.

[Shawe-Taylor and Cristianini, 2004] John Shawe-Taylor and Nello Cristianini. Kernel methods for pattern analysis. Cambridge University Press, 2004.

[Sun and Lim, 2001] Aixin Sun and Ee-Peng Lim. Hierarchical text classification and evaluation. *International Conference on Data Mining*, pages 521–528, 2001.

[Tsoumakas and Katakis, 2007] G. Tsoumakas and I. Katakis. Multi-label classification: An overview. *International Journal of Data Warehousing and Mining*, pages 3(3):1–13, 2007.

[Yang and Pedersen, 1997] Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. *International Conference on Machine Learning*, pages 412–420, 1997.

[Yao *et al.*, 2001] Yuan Yao, Gian Luca Marcialis, Massimiliano Pontil, Paolo Frasconi, and Fabio Roli. A new machine learning approach to fingerprint classification. *Italian Association for Artificial Intelligence*, pages 57–63, 2001.