Paarweiser Naive Bayes Klassifizierer

Diplomarbeit

im Studiengang `Informatik`
angefertigt am Fachbereich Informatik
der Technischen Universität Darmstadt

von

`Jan-Nikolas Sulzmann`

Darmstadt, 12. Juli 2006

Betreuer: Prof. Dr. Johannes Fürnkranz

Ehrenwörtliche 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.

Rödermark, 12. Juli 2006

Inhaltsverzeichnis

1. Einleitung

1.1. Einführung und Motivation

Die Annäherung der elektronischen Datenverarbeitung und der Kommunikation haben eine Gesellschaft geschaffen, die auf Informationen angewiesen ist. Trotzdem liegen die meisten Informationen in ihrer rohen Form – in Daten – vor. Wenn man Daten als aufgezeichnete Fakten charakterisiert, dann können Informationen als eine Menge von Mustern, Regelmäßigkeiten und Erwartungen, denen die Daten unterliegen, angesehen werden. Große Mengen von Informationen, die in Datenbanken verborgen sind, sind potentiell wichtig, aber bis jetzt nicht erschlossen oder klar formuliert worden. Ziel des Data Mining ist es, diese hervorzubringen und nutzbar zu machen.

Data Mining ist die Extraktion von impliziten, vorher unbekannten und potentiell nützlichen Informationen aus Daten. Die grundlegende Idee ist hierbei, Computerprogramme zu entwickeln, die automatisiert Datenbanken auf Regelmäßigkeiten und Muster untersuchen. Anhand von vorhandenen, starken Regelmäßigkeiten können Daten generalisiert werden, um genaue Vorhersagen über kommende Daten treffen zu können. Natürlich ist das Ganze nicht problemlos zu bewerkstelligen. Viele der Muster sind banal und uninteressant. Andere hingegen sind irreführend, da sie nur durch das zufällige Auftreten in den verwendeten Daten entstanden sind. Zusätzlich sind reale Daten nicht perfekt. Einige Teile können verrauscht sein oder fehlen, demnach werden alle entdeckten Muster nicht exakt sein. Jede Regel wird Ausnahmen haben, und nicht jede Möglichkeit wird durch eine Regel abgedeckt werden. Deshalb müssen Algorithmen robust genug sein, um mit verrauschten Daten umgehen und Regeln, die nicht exakt sind, extrahieren zu können.

Maschinelles Lernen stellt die technische Basis des Data Minings dar. Es wird dazu verwendet Informationen aus den rohen Daten in Datenbanken zu extrahieren und dadurch strukturelle Beschreibungen zu erlangen. Die gefundenen Beschreibungen können zur Vorhersage, Erklärung und zum Verständnis von Daten verwendet werden. Abhängig von der gewünschten Verwendung der Beschreibungen existieren vier grundlegende Kategorien von Lernverfahren. Das klassifizierende Lernen versucht anhand einer Menge von klassifizierten Beispielen eine Beschreibung zu finden, mit der unbekannte Beispiele in Klassen eingeordnet werden können. Beim assoziativen Lernen werden alle Assozationen gesucht, die sich zwischen den Merkmalen des Datensatzes ergeben. Im Gegensatz zur Klassifikation beschränkt man sich hierbei nicht nur auf Beziehungen, die die Klasse von Beispielen vorhersagen, sondern auch auf solche, die andere Merkmale vorhersagen. Beim Clustering sucht man Teilmengen des Datensatzes, deren Elemente zueinander gehören. Bei der numerischen Vorhersage oder Regression sind die Vorhersagen nicht

aus einer diskreten Menge von Werten, sondern numerische Größen. Unabhängig von der Kategorie des Lernverfahrens bezeichnen wir das zu Erlernende als Zielkonzept oder kurz Konzept und die generierte Ausgabe eines Lernverfahrens als Konzeptbeschreibung, Hypothese oder Modell.

Im Laufe dieser Arbeit werden wir uns nur mit dem klassifizierenden Lernen befassen. Unser Hauptaugenmerk legen wir dabei auf den Naive Bayes Klassifizierer, einem verbreiteten Vertreter des klassifizierenden Lernens, der auf einen fundierten wahrscheinlichkeitstheoretischen Ansatz aufbaut. Der Naive Bayes Klassifizierer wird wegen seiner Einfachheit und Effizienz in vielen Bereichen (unter anderem medizinische Diagnosen, Textkategorisierung, kollaboratives und E-Mail-Filtern) des klassifizierenden Lernens verwendet und erreicht dort eine gute Performanz. Verglichen mit moderneren, raffinierteren Lernverfahren erzielt der Naive Bayes Klassifizierer häufig bessere Ergebnisse. Außerdem kann der Naive Bayes Klassifizierer mit einer großen Anzahl von Variablen, die sowohl diskret als auch kontinuierlich sein können, und mit großen Datensätzen umgehen.

1.2. Ziel dieser Arbeit

Das Ziel dieser Arbeit ist es zu untersuchen, ob wir die Performanz des Naive Bayes Klassifizierers verbessern können, wenn wir ihn als Basisklassifizierer für Klassenbinarisierungen verwenden. Klassenbinarisierungen, eine Gruppe von Metaklassifizierern, wurden zur Lösung von Multiklassenproblemen durch Transformation in mehrere binäre Probleme konzipiert. Sie werden daher primär bei Klassifizierern, die nur mit binären Problemen umgehen können, eingesetzt, können jedoch auch auf Klassifizierer angewandt werden, die Multiklassenprobleme lösen können, und deren Performanz steigern. Wir werden untersuchen, ob dies auch bei dem Naive Bayes Klassifizierer der Fall ist. Dabei werden wir einerseits die bekannten Methoden der Klassenbinarisierung und andererseits eigene Methoden betrachten, die alle auf einen gemeinsamen, wahrscheinlichkeitstheoretischen Ansatz basieren.

1.3. Gliederung

Die Kapitel sind wie folgt strukturiert. In Kapitel 2 geben wir eine kurze Einführung in das Maschinelle Lernen. Wir erläutern die Grundlagen von Lernverfahren und den verwendeten Daten. Dabei gehen wir näher auf das klassifizierende Lernen ein. Anschließend zeigen wir, wie klassifizierende Lernverfahren evaluiert und verglichen werden können.

Kapitel 3 befaßt sich mit dem Naive Bayes Klassifizierer. Wir erläutern seine Anwendung und verdeutlichen sie anhand eines Beispieles. Anschließend erklären wir die Behandlung von speziellen Ausnahmen, die bei der Anwendung vorkommen können. Dazu gehören das Auftreten von fehlenden oder kontinuierlichen Attributen und die Anwendung des Naive Bayes Klassifizierers auf unstrukturierte Daten wie zum Beispiel Texte oder Web-Dokumente.

In Kapitel 4 behandeln wir Ensemble-Methoden oder kurz Ensembles, eine spezielle Kategorie von Metaklassifizierern. Ensembles bestehen aus einer Menge von Basisklassifizierern, deren Vorhersagen zu einer einzigen Vorhersage dekodiert werden. Wir stellen die Ensemble-Methoden Bagging, Boosting und die Gruppe der Klassenbinarisierungen vor. Da unser Hauptaugenmerk auf den Klassenbinarisierungen liegt, erläutern wir die geordnete, ungeordnete und die paarweise beziehungsweise Round-Robin-Klassenbinarisierung ausführlicher. Abschließend stellen wir noch zwei Gruppen von Dekodierungsmethoden vor. Bei der ersten Gruppe der Abstimmungsmethoden verwendet man die Vorhersage der Klassifizierer des Ensembles zur Abstimmung der endgültigen Vorhersage. Die zweite Gruppe der Bradley-Terry-Methoden eignet sich nur für paarweise Klassenbinarisierungen, da sie eine Annahme über den Zusammenhang zwischen den paarweisen Vorhersagen und der endgültigen Vorhersage trifft.

In Kapitel 5 untersuchen wir, wie sich Klassenbinarisierungen mit einem Naive Bayes Klassifizierer als Basisklassifizier verhalten, und stellen eigene paarweise Methoden vor. Zuerst betrachten wir die ungeordnete Klassenbinarisierung mit einem Naive Bayes Klassifizierer, die, wie wir wider Erwarten feststellen werden, nicht die gleichen Vorhersagen wie ein regulärer Naive Bayes Klassifizierer trifft. Da die beiden Methoden nicht äquivalent sind, gilt dies natürlich auch für geordnete Klassenbinarisierung. Danach betrachten wir die paarweise Klassenbinarisierung mit einem Naive Bayes Klassifizierer. Wir werden zeigen, daß dieses Verfahren äquivalent zu einem regulären Naive Bayes Klassifizierer ist und daß dies für alle Dekodierungsmethoden, die wir im vorangegangen Kapitel vorgestellt haben, gilt, falls die Implementierung des Verfahrens mit der theoretischen Berechnung übereinstimmt. Bei leicht veränderten Implementierungen kann es jedoch zu unterschiedlichen Ergebnissen kommen, wie wir im nachfolgenden Kapitel sehen werden.

Anschließend stellen wir unsere alternativen paarweisen Methoden vor, die auf dem gleichen wahrscheinlichkeitstheoretischen Ansatz basieren. Bei dessen Anwendung ergeben sich zwei Berechnungsansätze. Der erste Ansatz berechnet alle auftretenden Wahrscheinlichkeiten wie bei einem regulären Naive Bayes Klassifizierer. Der zweite Ansatz weicht von der regulären Berechnung eines Naive Bayes Klassifizierers ab. Er berechnet nicht nur die Wahrscheinlichkeiten von Klassen, sondern auch von Paaren von Klassen. Wenden wir den regulären Ansatz an, trifft ein Teil unserer Methoden die gleiche Vorhersage wie ein regulärer Naive Bayes Klassifizierer. Aus diesem Grund werden wir bei unseren Experimenten auf diese Methoden nur den paarweisen Ansatz anwenden.

In Kapitel 6 führen wir unsere Experimente mit der ungeordneten und paarweisen Klassenbinarisierung und unseren alternativen Methoden durch. Zuerst erläutern wir die Implementierung der verwendeten Verfahren und gehen auf die Testdaten und deren Format ein. Wir erläutern danach den Aufbau der Experimente. Abschließend werten wir die Ergebnisse unserer Experimente aus.

Im Kapitel 7 fassen wir nochmals die Erkenntnisse, die wir durch unsere Überlegungen in den vorangegangenen Kapiteln und durch unsere Experimente gewonnen haben, zusammen.

2. Maschinelles Lernen

Maschinelles Lernen ist ein Teilgebiet der Künstlichen Intelligenz, das sich mit der Entwicklung von Algorithmen und Techniken befaßt, die es einem Computer ermöglichen zu lernen. Diese Algorithmen und Techniken versuchen aus vorhandenen Daten zu lernen und nach Beendigung des Lernens zu verallgemeinern. Das heißt sie lernen diese Daten nicht nur auswendig, sondern sie versuchen Regelmäßigkeiten in den Lerndaten zu erkennen und für Vorhersagen auf ungesehenen Daten zu nützen.

In diesem Kapitel wollen wir einen kurzen Einblick in die Grundlagen des Maschinellen Lernens geben. Zuerst erläutern wir, welche Daten verwendet und wie diese im Bereich des Maschinellen Lernens bezeichnet werden. Danach beschreiben wir die Grundlagen von klassifizierenden Lernverfahren. Anschließend zeigen wir noch, wie man die Ergebnisse dieser Verfahren bewerten kann. Wir konzentrieren uns hierbei auf eine Bewertungsmethode, die stratifizierte 10x10-Kreuzvalidierung, die wir dann auch im Laufe unserer Experimente verwenden werden. Abschließend erklären wir, wie anhand dieser Ergebnisse zwei Lernverfahren verglichen werden können.

2.1. Daten

Für alle Verfahren des Maschinellen Lernens benötigen wir Daten als Eingabe. Daten lassen sich in zwei Kategorien aufteilen. Dabei handelt es sich zum einen um *strukturierte Daten* wie Tabellen und zum anderen um *unstrukturierte Daten* wie Texte oder Web-Dokumente. Da wir uns im Laufe dieser Arbeit hauptsächlich mit strukturierten Daten befassen werden, gehen wir zuerst auf strukturierte Daten und später kurz auf unstrukturierte Daten ein.

Ein Datensatz von strukturierten Daten besteht aus einem Header, der die Daten anhand von Attributtypen beschreibt, und aus *Beispielen* oder *Instanzen*, deren Attributwerte mit diesen Attributen konsistent sind. Beispiele werden durch Tupel von *Attributwerten* beschrieben. Bei strukturierten Daten ist die Anzahl der Attribute fest, folglich haben auch die Tupel eine feste Länge.

Attribute lassen sich in mehrere Kategorien oder Typen einteilen, die die Art der möglichen Attributwerte des jeweiligen Attributes beschreiben. Für unsere Zwecke sind die folgenden Attributtypen relevant. *Nominale* oder *diskrete* Attribute lassen nur eine eingeschränkte Liste von symbolischen Attributwerten zu. Zum Beispiel erlaubt das Attribut „Grundfarben“ die Speicherung der Attributwerte „Rot“, „Blau“ oder „Gelb“. *Kontinuierliche* Attribute hingegen haben eine unbeschränkte Menge von Attributwerten. Beispiele hierfür sind Zeichenketten, ganze oder reelle Zahlen. Handelt es sich bei den Werten des kontinuierlichen Attributes um eine Zahlenmenge spricht man auch von

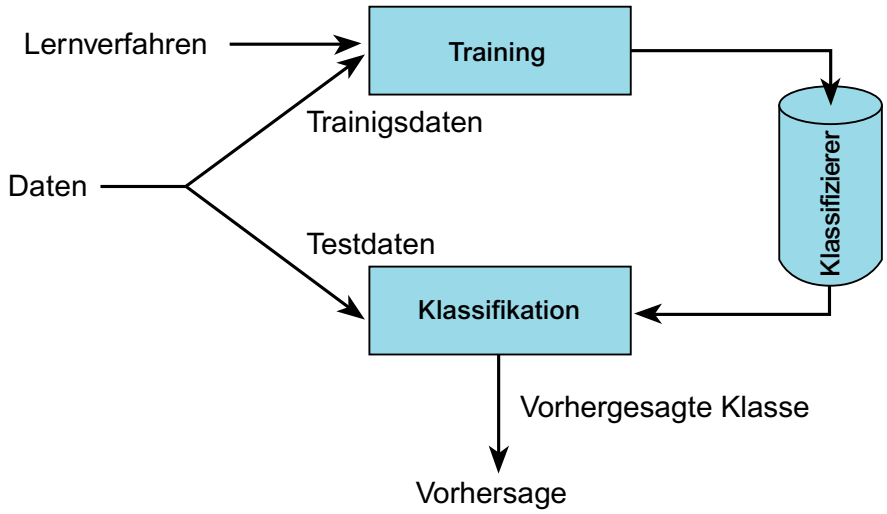


Abbildung 2.1.: Schema des klassifizierenden Lernens

numerischen Attributen. Sollte ein Attributwert einer Instanz unbekannt sein, spricht man von einem *fehlenden Attributwert*.

Betrachten wir nun die Unterschiede von unstrukturierten zu strukturierten Daten, die wir an Texten oder Dokumenten erklären werden. Diese stellen die Instanzen oder Beispiele eines unstrukturierten Datensatzes dar. Die Worte beziehungsweise die Terme eines Textes stellen im übertragenden Sinne seine Attribute dar. Die Attributwerte sind nun entweder binär (1, falls der Term im Dokument vorkommt, ansonsten 0) oder entsprechen der absoluten oder relativen Häufigkeit eines Terms innerhalb des Dokumentes. Da in einem Text selten alle Worte einer Sprache vorkommen, fehlen die meisten Attributwerte. Man speichert deshalb häufig nur die Attributwerte, die nicht den Wert 0 haben, um nicht unnötig Speicherplatz zu verwenden.

2.2. Klassifikation

Ziel des klassifizierenden Lernens ist die *Vorhersage* von Attributwerten eines bestimmten Attributes, dessen Werte nur für einen Teil der Daten bekannt ist. Dieses Attribut bezeichnet man als das *Klassenattribut* und seine Attributwerte als *Klasse* einer Instanz. Beispiele, deren Klasse uns bekannt ist, nennt man *klassifizierte Beispiele*. Entsprechend kennen wir die Klassen von unklassifizierten Beispielen nicht. Je nach Verwendungszweck bezeichnet man eine Menge von klassifizierten Beispielen als *Trainings-* oder *Testdaten* und deren Beispiele als *Trainings-* oder *Testbeispiele*.

Das klassifizierende Lernen setzt sich wie jedes andere Lernen aus zwei Schritten zusammen. Der erste Schritt besteht aus dem Aufbau oder *Training* eines *Klassifizierers*, mit dem wir im zweiten Schritt unklassifizierten Beispielen Klassen zuordnen. Diese

Zuordnung bezeichnet man als die *Klassifikation* von Beispielen oder *Vorhersage* der Klasse. Im allgemeinen verwendet man die Begriffe der Vorhersage oder Voraussage auch für die Vorhersage von anderen Attributwerten, wir beziehen uns jedoch immer auf die Vorhersage der Klasse, wenn wir in dieser Arbeit von einer Vorhersage oder Voraussage sprechen. Wenn ein Klassifizierer ein Maß für sein Vertrauen (zum Beispiel in Form einer Wahrscheinlichkeit) in seine Vorhersage hat, nennen wir dieses Maß die *Konfidenz* in seine Vorhersage. Manche Klassifizierer sagen nicht nur eine Klasse vorher, sondern geben auch eine Reihenfolge der Klassen an, in der der Klassifizierer es für wahrscheinlich hält, daß die Instanz zu der jeweiligen Klasse gehört. Diese Reihenfolge bezeichnet man als *Ranking* der Klassen. Je nach Anzahl der Klassen eines Datensatzes nennt man das Problem, eine Konzeptbeschreibung für die Daten zu finden, bei zwei Klassen ein *binäres Problem* und ansonsten ein *Multiklassenproblem*. Wir schreiben $< c_i, c_j >$ für binäre Probleme, bei denen wir zwischen den beiden Klassen c_i und c_j diskriminieren. Klassifizierer, die nur binäre Probleme behandeln können, nennen wir *binär*. Generiert ein Lernverfahren für die binären Probleme $< c_i, c_j >$ und $< c_j, c_i >$ den gleichen Klassifizierer, nennen wir das Lernverfahren und den Klassifizierer *klassensymmetrisch*. Klassifizierer, die andere Klassifizierer trainieren und zur Klassifikation von Instanzen verwenden, heißen *Metaklassizierer*. Die Klassifizierer, die von einem Metaklassifizierer verwendet werden, nennt man *Basisklassifizierer*.

Fassen wir nun den Ablauf der beiden Schritte zusammen. Im ersten Schritt erhält das klassifizierende Lernverfahren Trainingsdaten, mit deren Hilfe eine Konzeptbeschreibung beziehungsweise ein Klassifizierer für das Konzept, dem die Daten unterliegen, aufgebaut wird. Man nennt diesen Vorgang auch *überwachtes Lernen* (englisch: supervised learning), da dem Lernverfahren mitgeteilt wird, zu welcher Klasse ein Trainingsbeispiel gehört. Nachdem der Klassifizierer konstruiert wurde, kommt der zweite Schritt, die Klassifikation. Bevor man jedoch Instanzen klassifiziert, sollte man zuerst die Performanz des Klassifizierers abschätzen. Wie man die Performanz eines Klassifizierers bewerten beziehungsweise abschätzen kann, werden wir im nächsten Abschnitt behandeln. Ist die Performanz des Klassifizierers akzeptabel, kann er zur Klassifikation von unklassifizierten Instanzen verwendet werden.

2.3. Evaluierungsmethoden

Nach dem Lernen eines Klassifizierers wäre es wünschenswert, wenn wir seine *Performanz* oder Qualität bewerten könnten. Objektive Maße sind unter anderem die *Genauigkeit* (englisch: accuracy) oder die *Fehlerrate* (englisch: error rate) eines Klassifizerers, die komplementär zueinander sind. Die Genauigkeit ist der Prozentsatz der korrekt klassifizierten Beispiele, die Fehlerrate hingegen mißt den Prozentsatz der falsch klassifizierten Beispiele. Typischerweise verwendet man zur Einschätzung eines Klassifizierers seine Fehlerrate. Bei vielen, praktischen Data-Mining-Anwendungen unterliegt jedoch die Qualität eines Lernverfahrens auch noch subjektiveren Maßen. Oft werden Klassifizierer bevorzugt, die für Menschen verständlichere Konzeptbeschreibungen darstellen (zum Beispiel Regeln oder Entscheidungsbäume). In dieser Arbeit werden wir jedoch die

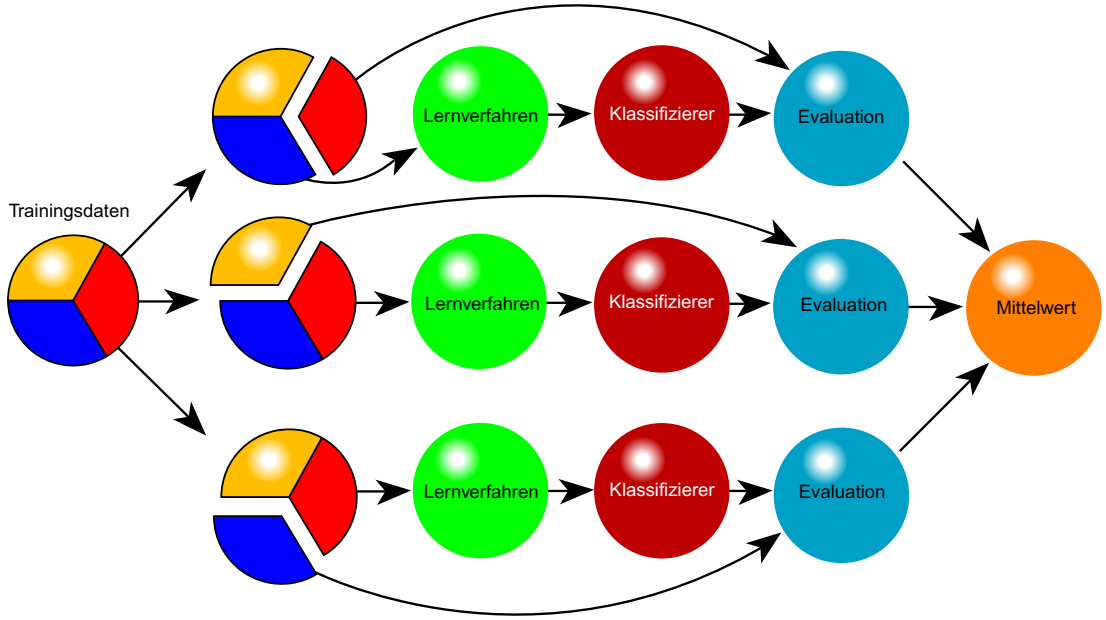


Abbildung 2.2.: Schema einer 1×3-Kreuzvalidierung

Fehlerrate verwenden, um die Performanz zu bestimmen und verschiedene Lernverfahren vergleichen zu können.

Wir sind an der *wirklichen Fehlerrate* eines Klassifizierers auf den Daten des analysierten Konzeptes interessiert. Das heißt wir suchen die relative Häufigkeit der Instanzen eines Konzeptes, denen der Klassifizierer eine falsche Klasse zuordnet. Natürlich können wir diesen Wert nicht exakt bestimmen, da wir im allgemeinen nicht die Klasse aller Instanzen, sondern nur die Klassen einer kleinen Menge von Instanzen kennen. Aus diesem Grund wollen wir anhand der uns bekannten, klassifizierten Beispiele die Fehlerrate des Klassifizierers bestimmen und damit die wirkliche Fehlerrate abschätzen.

Wenn wir jedoch die Trainingsdaten sowohl für das Training als auch zur Abschätzung eines Klassifizierers verwenden, kann das Ergebnis zu irreführenden, zu optimistischen Abschätzungen führen, falls der Klassifizierer sich zu gut an die Trainingsdaten angepasst hat. Verschiedene Techniken lösen dieses Problem, indem sie zur Evaluierung eines Klassifizierers die Trainingsdaten in zwei Teile aufteilen. Einen Teil verwenden sie wiederum zum Training eines Klassifizierers. Auf den anderen Teil wenden sie diesen Klassifizierer an und betrachten danach, wieviele Instanzen richtig klassifiziert werden. Daten mit Klassenlabel sind jedoch meistens nur wenig vorhanden und sollten besser vollständig für das Training verwendet werden. Deshalb trainiert man den Klassifizierer auf den kompletten Daten und schätzt mit der eben genannten Vorgehensweise, die man gegebenfalls mehrfach wiederholt und die Ergebnisse mittelt, die Fehlerrate ab. Betrachten wir nun verschiedene Techniken, die zur Ermittlung der Fehlerrate zufällige

Aufteilungen der Daten verwenden.

Bei der *Holdout-Methode* teilt man die klassifizierten Beispiele in zwei unabhängige Mengen, eine Trainingsmenge und eine Testmenge, auf. Normalerweise teilt man zwei Drittel der Trainingsmenge und ein Drittel der Testmenge zu. Die Trainingsmenge wird zum Aufbau des Klassifizierers verwendet, dessen Fehlerrate mit Hilfe der Testmenge abgeschätzt wird. Die Abschätzung ist pessimistisch, da nur ein Teil der ursprünglichen Daten für das Training der Klassifizierer benutzt wird. *Random subsampling* ist eine Variante der Holdout-Methode, bei der die Holdout-Methode k-mal wiederholt wird. Als endgültige Abschätzung der Fehlerrate verwendet man das Mittel der Fehlerraten, die wir bei jeder Iteration bestimmt haben.

Bei der *k-fachen Kreuzvalidierung* (englisch: cross validation) teilt man die ursprünglichen Daten in k disjunkte Teilmengen $S_1, S_2, ..., S_k$ von annähernd gleicher Größe auf. Wählt man für k die Anzahl der Trainingsbeispiele, spricht man von der *Leave-One-Out-Methode*. Das Training und das Testen wird k-mal ausgeführt. Bei jeder Iteration i wird die Teilmenge S_i als Testmenge verwendet. Die restlichen Teilmengen werden zusammen für den Aufbau des Klassifizierers verwendet. Wie beim Random subsampling bestimmen wir während jeder dieser Iterationen die Fehlerrate auf der Testmenge. Die endgültige Fehlerrate wird durch das Mitteln dieser Fehlerraten bestimmt. Bei einer *stratifizierten Kreuzvalidierung* werden die Teilmengen $S_1, S_2, ..., S_k$ so stratifiziert, daß die Verteilung der Klassen in jeder Teilmenge annähernd der Verteilung in den ursprünglichen Daten entspricht. Meistens wendet man die k-fache Kreuzvalidierung wiederum mehrfach an und mittelt die so bestimmten Fehlerraten. Bei einer l-fachen Wiederholung spricht man von einer *$l \times k$-Kreuzvalidierung*. In der Praxis liefert eine *stratifizierte* 10×10*-Kreuzvalidierung* gute Abschätzungen. Aus diesem Grund verwenden wir diese zur Evaluierung der Methoden, die wir bei unseren Experimenten untersuchen werden.

2.4. Vergleich von Klassifizierern

In diesem Abschnitt möchten wir kurz erläutern, wie die im vorangegangen Kapitel abgeschätzten Performanzmaße verwenden werden können, um Klassifizierer miteinander zu vergleichen. Dabei orientieren wir uns an [LW00].

Wir verwenden den Vorzeichentest, ein statistischer Test (siehe Anhang A), zum Vergleich von Klassifizierern. Die Idee, die dem Test zugrunde liegt, wollen wir an einem Beispiel erläutern.

Beispiel 2.1 Wir haben eine Menge von Datensätzen und zwei Klassifizierer A und B vorliegen. Die beiden Klassifizierer wurden auf allen Datensätzen trainiert, und für jeden Datensatz wurde die Performanz der Klassifizierer ermittelt. Bei diesen Meßreihen betrachten wir nur die Datensätze, bei denen die Klassifizierer eine unterschiedliche Performanz aufweisen. Angenommen wir haben 20 Datensätze mit unterschiedlichen Performanzwerten. Würde einer der Klassifizierer nun bei 20 Datensätzen eine bessere Performanz aufweisen, würde man davon ausgehen, daß dieser Klassifizierer besser ist als der andere. Auch wenn es „nur“ bei 19 oder 18 Datensätzen der Fall wäre, würde

man gefühlsmäßig diese Feststellung treffen. Wären die Klassifizierer wirklich gleichwertig bezüglich ihrer Performanz, so könnte man annehmen, daß ein Klassifizierer mit Wahrscheinlichkeit 0,5 besser ist als der andere. Da die Performanzwerte der Datensätze unabhängig zustande kommen, können wir von einer Binomialverteilung ausgehen. Damit ergibt sich für die Wahrscheinlichkeit, daß ein Klassifizierer bei 20 Datensätzen besser ist als der andere, ein Wert von $(1/2)^{20} < 10^{-6}$. Auch 18 oder 19 bessere Performanzwerte sind unter der Annahme der Gleichwertigkeit der Klassifizierer sehr unwahrscheinlich. Ihre Wahrscheinlichkeiten sind nämlich ungefähr $2 \cdot 10^{-5}$ beziehungsweise $2 \cdot 10^{-4}$. Wir werden also in diesen drei Fällen annehmen, daß die Klassifizierer A und B nicht gleichwertig sind. Ganz anders wäre unsere Einschätzung, wenn einer der Klassifizier nur bei 12 oder 13 Datensätzen eine bessere Performanz als der Klassifizierer B aufweist. Dieses Ergebnis stünde noch einigermaßen im Einklang mit dem, was man bei der Gleichwertigkeit der Klassifizierer erwarten würde.

Die in diesem Beispiel angedeutete Vorgehensweise wollen wir nun präzisieren. Wir denken uns die Vergleiche der Performanzwerte als Realisierungen von n unabhängigen Zufallsvariablen $D_1, ..., D_n$, die die Werte 1 (Klassifizierer A hat eine bessere Performanz als Klassifizierer B) und 0 (Klassifizierer B hat eine bessere Performanz als Klassifizierer A) annehmen können. Wir bezeichnen mit

$$V = D_1 + ... + D_n$$

die Zufallsvariable, mit der die Gesamtzahl der Datensätze, für die der Klassifizierer A eine bessere Performanz als Klassifizierer B hat, beschrieben wird. Unter der Nullhypothese

$$H_0 : \Pr(D_i = 1) = \Pr(D_i = 0) = \frac{1}{2},\ i = 1, ..., n$$

ist V binomial $B(n, 0, 5)$-verteilt. Bestimmt man nun zu einem vorgegebenem α $(0 < \alpha < 1)$, die größte Zahl k mit

$$\Pr(V < k \cup V > n - k) = 2 \cdot \Pr(V < k) = 2 \sum_{i=0}^{k-1} \binom{n}{i} \cdot \frac{1}{2^n} \leq \alpha$$

so haben wir mit der Entscheidungsregel:

Falls $V < k$ oder $V > n - k$, dann H_0 ablehnen.
Falls $k \leq V \leq n - k$, dann H_0 nicht ablehnen.

einen Test zum Niveau α für die Überprüfung der Nullhypothese H_0 gefunden. In unserem Beispiel mit $n = 20$ ergibt sich für α wegen

$$\left[\binom{20}{0} + ... + \binom{20}{5}\right] \cdot \frac{1}{2^{20}} \approx 0,021 \text{ und } \left[\binom{20}{0} + ... + \binom{20}{6}\right] \cdot \frac{1}{2^{20}} \approx 0,058$$

die Schranke zu k = 6. Für die Überprüfung der Gleichwertigkeit der beiden Klassifizierer anhand der Performanzmessungen ergeben sich aus den obigen Überlegungen, daß wir

wie folgt verfahren können. Falls ein Klassifizierer bei mehr als 14 oder weniger als 6 Datensätzen besser abschneidet als der andere, betrachten wir die Klassifizierer als nicht gleichwertig. Bei dieser Vorgehensweise ist die Wahrscheinlichkeit, daß wir bei Gleichwertigkeit der Medikamente auf einen Unterschied schließen, kleiner als 0,05 (circa 0,042).

Dieses Verfahren kann auch angewendet werden, wenn die zu analysierenden Daten nicht als Besser-Schlechter-Antworten sondern als reelle Zahlen vorliegen.

Beispiel 2.2 Wir haben wieder eine Menge von Datensätzen und zwei Klassifizierer A und B vorliegen. Es soll nun untersucht werden, ob sich die Klassifizierer bezüglich eines Performanzmaßes unterscheiden. Wir trainieren beide Klassifizierer auf 20 Datensätzen. Mit den auf den Datensätzen ermittelten Performanzwerten $x_1, ..., x_{20}$ und $y_1, ..., y_{20}$ berechnen wir die Differenzen $d_i = x_i - y_i$ der Performanzwerte.

Die Werte $x_1, ..., x_n$ und $y_1, ..., y_n$ (im Beispiel $n = 20$) denken wir uns als Realisierungen von Zufallsvariablen $X_1, ..., X_n$ beziehungsweise $Y_1, ..., Y_n$. Dabei können wir für die Paare (X_1, Y_1),...,(X_n, Y_n), das heißt für die zweidimensionalen Zufallsvariablen, annehmen, daß sie unabhängig sind.

Eine Unabhängigkeitsannahme für X_i und Y_i bei gleichem i ist jedoch nicht angebracht. Die beiden Ergebnisse x_i und y_i des i-ten Datensatzes werden sicher durch die Eigenheiten des Datensatzes beeinflußt.

Dagegen sind die Zufallsvariablen $D_i = X_i - Y_i$, $i = 1, ..., n$, unabhängig. Der Einfachheit halber setzen wir noch voraus, daß die Differenzen D_i, $i = 1, ..., n$, identisch verteilt sind mit der selben stetigen Verteilungsfunktion. Dann tritt das Ereignis $X_i = Y_i$ für jedes $i = 1, ..., n$ nur mit Wahrscheinlichkeit 0 ein, so daß unter der Annahme gleicher Performanz der Klassifizierer die Verteilungsannahme

$$H_0 : \Pr(D_i > 0) = \Pr(D_i < 0) = \frac{1}{2} \text{ für alle } i = 1, ..., n \tag{2.1}$$

gemacht werden kann, die wir als Nullhypothese testen wollen. H_0 ist gleichbedeutend mit der Aussage, daß die Zufallsvariablen $D_1, ..., D_n$ den Median 0 besitzen.

Wir verwenden hier als Testgröße V die Anzahl der positiven Differenzen. Sie ist unter H_0 binomial $B(n, 0, 5)$-verteilt. Darum führt die Entscheidungsregel von Beispiel 2.1 ebenfalls zu einem Niveau-α-Test zum Prüfen der Nullhypothese H_0.

Da bei der dieser Entscheidungsregel ausschließlich die Vorzeichen der beobachteten Differenzen berücksichtigt werden, heißt sie *Vorzeichentest*. Wegen der paarweisen Zusammenfassung der Beobachtungsdaten (x_i, y_i), $i = 1, ..., n$, spricht man vom Vorzeichentest der paarweisen Beobachtung.

Bei unseren Experimenten verwenden wir den Vorzeichentest zum Vergleich der untersuchten Klassifizierer. Wir führen den Test mit zwei verschiedenen α durch. Lehnen wir H_0 bei einem Niveau von $\alpha = 5\%$ oder $\alpha = 1\%$ ab, gehen wir davon aus, daß die Verfahren sich *signifikant* beziehungsweise *höchst signifikant* unterscheiden. In diesem Fall nennen wir das bessere Verfahren *(höchst) signifikant besser*, das andere entsprechend *(höchst) signifikant schlechter*. Ansonsten nehmen wir an, daß die Verfahren gleichwertig sind.

3. Naives Bayes Klassifizierer

Der *Naive Bayes Lerner* oder *Klassifizierer* ist ein Lernverfahren, das in einigen Anwendungsbereichen eine bessere Performanz als ausgefeiltere Lernverfahren (wie zum Beispiel *Neuronale Netze, Nearest Neighbour oder Entscheidungsbaumlernen*) gezeigt hat [MST94]. Er zählt zu den *Bayes'schen Lernverfahren*, deren Grundlage der *Satz von Bayes* ist. Dieser Satz ermöglicht die Abschätzung der Wahrscheinlichkeit jeder Klasse c_i für ein gegebenes Test- oder Klassifizierungsbeispiel D. Anhand dieser Wahrscheinlichkeiten können wir das Beispiel klassifizieren, indem wir die Klasse voraussagen, für die die höchste Wahrscheinlichkeit geschätzt wurde.

Der *Satz von Bayes* lautet für zwei Zufallsereignisse A und B wie folgt:

$$\Pr(A|B) = \frac{\Pr(B|A) \cdot \Pr(A)}{\Pr(B)} \tag{3.1}$$

Die bedingten Wahrscheinlichkeiten $\Pr(A|B)$ und $\Pr(B|A)$ nennen wir die a posteriori Wahrscheinlichkeiten. Hingegen bezeichnen wir die Wahrscheinlichkeiten $\Pr(A)$ und $\Pr(B)$ als a priori Wahrscheinlichkeiten.

Man kann den Naive Bayes Klassifizierer auf Lernaufgaben anwenden, bei denen wir *strukturierte Daten* (z.B. Tabellen) oder *unstrukturierte Daten* (z.B. Texte, Web-Dokumente) vorliegen haben. Wir werden jedoch zuerst nur *strukturierte Daten* behandeln und in einem späteren Abschnitt noch einmal auf die Unterschiede zu *unstrukturierten Daten* eingehen. Wir betrachten nun also zuerst Daten, bei denen jede Instanz D durch eine Menge von Attributwerten und ein Klassenlabel aus einer endlichen Menge von Klassen beschrieben wird. Man verfügt über eine Menge von Trainingsbeispielen, deren Klassenlabel bekannt ist. Der Klassfizierer versucht für neue Instanzen, die jeweils durch das Tupel ihrer Attributwerte $(a_1, a_2, ..., a_n)$ beschrieben werden, ihr Klassenlabel vorherzusagen.

3.1. Notationen

Bevor wir im einzelnen auf das Training und die Klassifizierung eingehen, müssen wir zuerst noch einige Notationen einführen.

- n: Anzahl der Attribute
- A_i: Attribut $A_i \in \{A_1, A_2, ..., A_n\}$, eine Menge von Attributwerten
- a_i: ein Attributwert des Attributes A_i

- $a(k)$: der Attributwert von Attribut A des k-ten Trainingsbeispiels
- $a_i(k)$: der Attributwert von Attribut A_i des k-ten Trainingsbeispiels
- v_i: Anzahl unterschiedlicher Attributwerte des Attributes A_i
- v: durchschnittliche Anzahl unterschiedlicher Attributwerte der Attribute
- m: Anzahl der Klassen
- c_i: Klasse $c_i \in \{c_1, c_2, ..., c_m\}$
- c_{ij}: das Klassenpaar, das aus den Klassen c_i und c_j besteht.
  Das Klassenpaar tritt als Zufallsereignis ein, falls eine der beiden Klassen als Zufallsereignis eintritt. Das heißt, es gilt $c_{ij} = c_i \cup c_j$
- D: ein Klassifizierungs- oder Testbeispiel, das durch die Attributwerte $(a_1, a_2, ..., a_n)$ beschrieben wird
- $\overline{D}$: ein Trainingsbeispiel, das durch die Attributwerte und sein Klassenlabel $(a_1, a_2, ..., a_n, c_i)$ beschrieben wird
- $c(k)$: das Klassenlabel des k-ten Trainingsbeispiels
- t: Anzahl der Trainingsbeispiele
- t_{c_i}: Anzahl der Trainingsbeispiele mit Klassenlabel c_i
- $t_{c_j}^{a_i}$: Anzahl der Trainingsbeispiele mit Klassenlabel c_j und Attributwert a_i für das Attribut A_i

Wir möchten darauf hinweisen, daß wir eine vereinfachte Notation verwenden. Dabei verzichten wir bewußt auf einen weiteren Index bei a_i, um die Lesbarkeit der Gleichungen zu erhöhen. Streng genommen würde ein Index l bei a_i^l dafür stehen, daß es sich um den l-ten Attributwert des Attributes $A_i = \{a_i^1, ..., a_i^{v_i}\}$ handelt. Die Berechnungen des Naive Bayes Klassifizierers beziehen sich jedoch immer auf ein Testbeispiel $D = (a_1^{l_1}, ..., a_n^{l_n})$, so daß wir $a_i = a_i^{l_i}$ immer als einen konkreten Attributwert ansehen.

3.2. Grundversion des Naive Bayes Klassifizierers

Wir sind wie bereits erwähnt an den a posteriori Wahrscheinlichkeiten für eine Klasse c_i unter dem Auftreten des Beispiels $D = (a_1, a_2, ..., a_n)$ interessiert. Setzen wir c_i und D als Ereignisse in (3.1) ein, bekommen wir folgenden Berechnungsansatz:

$$\Pr(c_i|D) = \frac{\Pr(D|c_i) \cdot \Pr(c_i)}{\Pr(D)} \tag{3.2}$$

Mit diesem Ansatz erhalten wir bereits die folgende, vorläufige Version des Naive Bayes Klassifizierers:

$$\begin{aligned} & \arg\max_{c_i} \Pr(c_i|D) \\ = & \arg\max_{c_i} \frac{\Pr(D|c_i) \cdot \Pr(c_i)}{\Pr(D)} \\ = & \arg\max_{c_i} \Pr(D|c_i) \cdot \Pr(c_i) \\ = & \arg\max_{c_i} \Pr(a_1, a_2, ..., a_n|c_i) \cdot \Pr(c_i) \end{aligned} \tag{3.3}$$

Wir benötigen demnach für den Naive Bayes Klassifizierer nur die Wahrscheinlichkeiten $\Pr(a_1, a_2, ..., a_n|c_i)$ und $\Pr(c_i)$. Im folgenden werden wir sehen, wie wir diese abschätzen können. Wir werden uns zuerst auf *nominale* Attribute beschränken, auf *kontinuierliche* bzw. *numerische* Attribute gehen wir erst später in Abschnitt 3.5 ein.

Zur Berechnung der Wahrscheinlichkeiten benötigen wir zuerst die Werte von t, t_{c_i} und $t_{c_i}^{a_i}$, die wir während der Trainingszeit bestimmen. Hierfür durchlaufen wir für jedes Trainingsbeispiel $(a_1, a_2, ..., a_n, c_j)$ alle Attribute und das Klassenlabel und erhöhen entsprechend t_{c_j} und $t_{c_j}^{a_i}$. Diese Vorgehensweise entspricht im wesentlichen einem simplen Abzählen der absoluten Häufigkeiten der Klassen und der Attributwerte unter den Klassen. Das Training benötigt demzufolge $O(tn)$ Zeit und $O(mnv)$ Speicher, wobei v die durchschnittliche Anzahl von Attributwerten pro Attribut ist.

Jetzt haben wir alles zusammen, was wir zur Berechnung der Wahrscheinlichkeiten benötigen. Die Wahrscheinlichkeiten der Klassen berechnen sich nun wie folgt. Wir teilen einfach die absolute Häufigkeit der Klasse c_i durch die Anzahl der Trainingsbeispiele:

$$\Pr(c_i) = \frac{t_{c_i}}{t} \tag{3.4}$$

Die Berechnung der bedingten Wahrscheinlichkeiten $\Pr(a_1, a_2, ..., a_n|c_j)$ stellt uns noch vor ein Problem. Das Tupel $(a_1, a_2, ..., a_n|c_j)$ kommt höchstwahrscheinlich nur selten oder gar nicht für die einzelnen Klassen vor. Dies würde bei der Abschätzung der Wahrscheinlichkeiten $\Pr(a_1, a_2, ..., a_n|c_j)$ zu sehr kleinen, nicht aussagekräftigen Werten führen, die sogar 0 betragen können. Aus diesem Grund treffen wir eine *naive* Annahme über die Unabhängigkeit der Attribute und erhalten damit eine Abschätzungsmöglichkeit für die gesuchten bedingten Wahrscheinlichkeiten. Der *Naive Bayes Klassifizierer* wird wegen dieser *Unabhängigkeitsannahme* als *naiv* bezeichnet.

$$\Pr(a_1, a_2, ..., a_n|c_j) = \prod_{i=1}^{n} \Pr(a_i|c_j) \tag{3.5}$$

Die Wahrscheinlichkeiten $\Pr(a_i|c_j)$ können wir unabhängig von den anderen Attributen abschätzen. Wir teilen zu diesem Zweck die relativen Häufigkeiten der Beispiele, für die das Attribut A_i den Wert a_i hat und das Klassenlabel c_j ist, durch die relative

Häufigkeit der Klasse c_i, deren Abschätzung wir bereits weiter oben beschrieben haben.

$$\begin{aligned} \Pr(a_i|c_j) &= \frac{\Pr(a_i \cap c_j)}{\Pr(c_j)} = \frac{\frac{t_{c_j}^{a_i}}{t}}{\frac{t_{c_j}}{t}} \\ &= \frac{t_{c_j}^{a_i}}{t_{c_j}} \end{aligned} \tag{3.6}$$

Mit (3.6) können wir (3.5) berechnen, allerdings gibt es hierbei ein Problem. Falls die absolute Häufigkeit $t_{c_j}^{a_i}$ eines Attributwertes a_i unter einer Klasse c_j 0 beträgt, dann gilt dies auch für die Wahrscheinlichkeit $\Pr(a_i|c_j)$ und für das Produkt (3.5). Dies ist aber nicht wünschenswert, da aus diesem Grund die übrigen Wahrscheinlichkeiten $\Pr(a_k|c_j), k \neq i$ nebensächlich werden. Eine oft angewandte Lösung dieses Problems ist die μ-Abschätzung[1]. Sie geht von einer a priori Schätzung p der Wahrscheinlichkeit $\Pr(a_i|c_j)$ aus, z.B. durch Annahme einer uniformen Verteilung der Attributwerte des Attributes A_i, das heißt $p = \frac{1}{v_i}$.

$$\mu\text{-Abschätzung: } \Pr(a_i|c_j) = \frac{t_{c_j}^{a_i} + \mu p}{t_{c_j} + \mu} \tag{3.7}$$

Wir werden jedoch im Laufe der weiteren Kapitel und unserer Experimente eine spezielle Variante dieser Abschätzung, die sogenannte *Laplace-Abschätzung*, verwenden. Bei ihr wird eine a priori Verteilung angenommen, bei der jeder Attributwert einmal häufiger als in den Trainingsdaten vorkommt.

$$\text{Laplace-Abschätzung: } \Pr(a_i|c_i) = \frac{t_{c_i}^{a_i} + 1}{t_{c_i} + v_i} \tag{3.8}$$

Zusammenfassend sieht die Grundversion des Naive Bayes Klassifizierer wie folgt aus:

$$c_{NB} = \arg\max_{c_i} \Pr(c_i) \prod_{j=1}^{n} \Pr(a_j|c_i) \tag{3.9}$$

Die Wahrscheinlichkeit $\Pr(c_i)$ beziehungsweise $\Pr(a_i|c_j)$ wird hierbei mit (3.4) beziehungsweise (3.8) abgeschätzt. Die Wahrscheinlichkeiten der Klassen $\Pr(c_i)$ müssen nicht zwingend mit (3.8) geschätzt werden, da eine Klasse ohne Trainingsbeispiele immer die niedrigste Wahrscheinlichkeitsabschätzung erhält.

3.3. Illustratives Beispiel

Wir möchten in diesem Abschnitt anhand eines Beispieldatensatzes die Anwendung des Naive Bayes Klassifizierers verdeutlichen. Dabei wollen wir kurz zeigen, wieso man

[1] Diese Abschätzung wird üblicherweise als m-Abschätzung bezeichnet, wir werden sie jedoch μ-Abschätzung nennen, um eine Verwechslung mit der Anzahl der Klassen m zu vermeiden

Aussicht	Temperatur	Luftfeuchtigkeit	Windstärke	Klasse
Bewölkt	Kalt	Hoch	Stark	Golf
Bewölkt	Kalt	Hoch	Stark	Squash
Sonnig	Warm	Niedrig	Schwach	Golf
Sonnig	Warm	Niedrig	Stark	Squash
Regen	Kalt	Niedrig	Schwach	Squash
Sonnig	Warm	Hoch	Stark	Golf
Bewölkt	Kalt	Niedrig	Stark	Tennis
Bewölkt	Kalt	Niedrig	Schwach	Tennis
Bewölkt	Warm	Hoch	Schwach	Golf
Bewölkt	Kalt	Hoch	Schwach	Tennis
Sonnig	Warm	Niedrig	Stark	Golf
Bewölkt	Kalt	Hoch	Stark	Tennis
Regen	Kalt	Hoch	Schwach	Squash
Sonnig	Warm	Niedrig	Schwach	Golf
Sonnig	Warm	Niedrig	Schwach	Tennis

Tabelle 3.1.: Unser Beispieldatensatz

die Unabhängigkeitsannahme und Laplace-Abschätzung verwendet und nicht die Wahrscheinlichkeit direkt abschätzt.

Der Datensatz (Tabelle 3.1), den wir hierfür verwenden werden, ist eine Aufzeichnung der sportlichen Aktivitäten der letzten fünfzehn Tage von Herrn Mustermann abhängig von den meteorologischen Daten des jeweiligen Tages. Es handelt sich hierbei um diskrete Daten über die *Aussicht*, *Temperatur*, *Luftfeuchtigkeit*, *Windstärke* und *Sportart* des jeweiligen Tages. Die Sportarten *Golf*, *Squash* und *Tennis* sind die drei Klassen des Datensatzes. Seine Attribute sind *Aussicht*, *Temperatur*, *Luftfeuchtigkeit* und *Windstärke*. Jedes dieser Attribute hat eine Menge von möglichen Attributwerten. Zum Beispiel hat das Attribut *Aussicht* die auftretenden Attributwerte *Bewölkt*, *Sonnig* und *Regen*. Es ist leicht einzusehen, daß dies nicht unbedingt alle möglichen Attributwerte des Attributes *Aussicht* sind. Zum Beispiel wäre auch *Schnee* ein zulässiger Wert.

Das Ziel der Klassifikation von Beispielen ist bei diesem Datensatz also die Vorhersage der *Sportart* (Klasse), die Herr Mustermann ausüben wird, anhand der meteorologischen Daten (Attributwerte) eines Tages.

Bevor wir mit der Klassifikation eines Beispieles beginnen, müssen wir zuerst die Häufigkeiten aller Attributwerte für alle Klassen bestimmen. Diese können je nach Implementierung relativ oder absolut gespeichert werden. In der Regel und für dieses Beispiel bietet sich die Speicherung der absoluten Häufigkeiten an, da man diese zum einen beim Eintreffen weiterer Trainingsdaten am leichtesten anpassen kann und zum anderen sind diese zur Erläuterung der Klassifikation leichter nachvollziehbar. Wir werden also zuerst die absoluten Häufigkeiten des Ausgangsdatensatzes und später die absoluten Häufigkeiten der Laplace-Abschätzungen berechnen. Die abgeschätzten Wahrscheinlichkeiten kennzeichnen wir durch ein Dach, zum Beispiel $\widehat{\Pr}(c_i|D)$.

Attribut	Attributwert	Golf	Squash	Tennis
Aussicht	Bewölkt	2	1	4
	Regen	0	2	0
	Sonnig	4	1	1
Temperatur	Kalt	1	3	4
	Warm	5	1	1
Luftfeuchtigkeit	Hoch	3	2	2
	Niedrig	3	2	3
Windstärke	Schwach	3	2	3
	Stark	3	2	2
Klasse		6	4	5

Tabelle 3.2.: Absolute Häufigkeiten des Beispieldatensatzes

Wir verwenden folgendes Beispiel, dessen Klasse uns unbekannt ist:

$$D = (Regen, Warm, Niedrig, Schwach)$$

An diesem Beispiel werden wir sehen, daß die Unabhängigkeitsannahme sinnvoll ist. Würden wir sie nicht treffen, könnten wir die Wahrscheinlichkeit, daß das Beispiel unter Beobachtung der einzelnen Klassen eintritt, nur über die absoluten Häufigkeiten des Tupels seiner Attributwerte abhängig von den Klassen innerhalb der Daten abschätzen. Betrachten wir den Datensatz, stellen wir fest, daß dieses Tupel von Attributwerten für keine der drei Klassen vorkommt. Aus diesem Grund können wir für keine der Klassen die Wahrscheinlichkeit, daß sie unter Beobachtung von D auftritt, abschätzen. Wir müssen nun entweder die Wahrscheinlichkeiten als gleich annehmen oder die Wahrscheinlichkeiten $\Pr(c_i)$ als Abschätzungen für $\Pr(c_i|D)$ verwenden. Beide Möglichkeiten stellen nur unzureichende Lösungen des Problems dar. Verwenden wir die erste Möglichkeit, erhalten wir die folgenden abgeschätzten Wahrscheinlichkeiten.

$$\widehat{\Pr}(Regen, Warm, Niedrig, Schwach|Golf) = \frac{1}{3}$$
$$\widehat{\Pr}(Regen, Warm, Niedrig, Schwach|Squash) = \frac{1}{3}$$
$$\widehat{\Pr}(Regen, Warm, Niedrig, Schwach|Tennis) = \frac{1}{3}$$

Wir können deshalb ohne Unabhängigkeitsannahme keine Vorhersage für dieses Beispiel treffen. Bei der zweiten Möglichkeit wird die Klasse *Golf* vorhergesagt. Diese Vorhersage stellt auch nur eine grobe Schätzung dar, da sie keinen Attributwert berücksichtigt.

Betrachten wir das Beispiel nochmal, diesmal wenden wir jedoch die Unabhängigkeitsannahme an. Dafür müssen wir zuerst die Wahrscheinlichkeiten jedes Attributwertes unter jeder Klasse (zum Beispiel $\Pr(Regen|Golf)$) abschätzen, um anschließend die Schätzungen der Wahrscheinlichkeiten der Klassen unter dem Trainingsbeispiel be-

Attribut	Attributwert	Golf	Squash	Tennis
Aussicht	Bewölkt	3	2	5
	Regen	1	3	1
	Sonnig	5	2	2
Temperatur	Kalt	2	4	5
	Warm	6	2	2
Luftfeuchtigkeit	Hoch	4	3	3
	Niedrig	4	3	4
Windstärke	Schwach	4	3	4
	Stark	4	3	3

Tabelle 3.3.: Absolute Häufigkeiten des Beispieldatensatzes mit Laplace-Abschätzung

stimmen zu können. Die absoluten Häufigkeiten haben wir in Tabelle 3.2 zusammengefasst. Mit ihrer Hilfe wollen wir exemplarisch anhand des Attributwertes *Regen* die Abschätzung der Wahrscheinlichkeiten ohne die Laplace-Abschätzung verdeutlichen.

Lesen wir die absoluten Häufigkeiten aus dieser Tabelle ab, erhalten wir folgende Werte:

$$t_{Golf}^{Regen} = 0 \qquad t_{Squash}^{Regen} = 2 \qquad t_{Tennis}^{Regen} = 0$$
$$t_{Golf} = 6 \qquad t_{Squash} = 4 \qquad t_{Tennis} = 5$$

Mit diesen Werten und (3.6) erhalten wir Abschätzungen der Wahrscheinlichkeiten für den Attributwert Regen unter jeder Klasse:

$$\widehat{\Pr}(Regen|Golf) = \frac{t_{Golf}^{Regen}}{t_{Golf}} = \frac{0}{6} = 0$$
$$\widehat{\Pr}(Regen|Squash) = \frac{t_{Squash}^{Regen}}{t_{Squash}} = \frac{2}{4} = \frac{1}{2}$$
$$\widehat{\Pr}(Regen|Tennis) = \frac{t_{Tennis}^{Regen}}{t_{Tennis}} = \frac{0}{5} = 0$$

Da $\widehat{\Pr}(Regen|Golf) = 0$ und $\widehat{\Pr}(Regen|Tennis) = 0$ gilt, erhalten wir die Wahrscheinlichkeitsabschätzungen $\widehat{\Pr}(Golf|D) = 0$ und $\widehat{\Pr}(Tennis|D) = 0$. Aus diesem Grund verlieren alle anderen Attributwerte ihre Bedeutung, obwohl es aber wünschenswert ist, daß wir möglichst alle Attributwerte bei unseren Berechnungen berücksichtigen. Dieses Problem wird durch die Verwendung der Laplace-Abschätzung behoben.

Wir werden deshalb die eben berechneten Wahrscheinlichkeiten noch einmal mit der Laplace-Abschätzung berechnen. Hierfür haben wir zwei Möglichkeiten, die das gleiche Ergebnis liefern. Entweder erstellen wir eine Tabelle der absoluten Häufigkeiten der Laplace-Abschätzung (Tabelle 3.3) und berechnen mit ihr die gesuchten Wahrscheinlichkeitsabschätzungen, oder wir verwenden die absoluten Häufigkeiten der Tabelle 3.2 und (3.8) zur Abschätzung dieser Wahrscheinlichkeiten.

	Regen	Warm	Niedrig	Schwach
Golf	1/9	3/4	1/2	1/2
Squash	3/7	1/3	1/2	1/2
Tennis	1/8	2/7	4/7	4/7

Tabelle 3.4.: Wahrscheinlichkeitsabschätzungen der Attributwerte des ersten Trainingsbeispiels

Zum besseren Verständnis haben wir auch die Formel zur Berechnung der Wahrscheinlichkeitabschätzungen angegeben. Die abgeschätzten Wahrscheinlichkeiten sehen dann wie folgt aus:

$$\widehat{\Pr}(Regen|Golf) = \frac{t_{Golf}^{Regen} + 1}{t_{Golf} + 3} = \frac{1}{9}$$
$$\widehat{\Pr}(Regen|Squash) = \frac{t_{Squash}^{Regen} + 1}{t_{Squash} + 3} = \frac{3}{7}$$
$$\widehat{\Pr}(Regen|Tennis) = \frac{t_{Tennis}^{Regen} + 1}{t_{Tennis} + 3} = \frac{1}{8}$$

Analog zur Berechnung dieses Attributwertes können wir die Wahrscheinlichkeitsabschätzungen der restlichen Attributwerte bestimmen (siehe Tabelle 3.4). Mit ihrer Hilfe können wir nun die Wahrscheinlichkeit für das Trainingsbeispiel D unter Beobachtung der Klassen abschätzen:

$$\widehat{\Pr}(D|Golf) = \frac{1}{9} \cdot \frac{3}{4} \cdot \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{48}$$
$$\widehat{\Pr}(D|Squash) = \frac{3}{7} \cdot \frac{1}{3} \cdot \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{28}$$
$$\widehat{\Pr}(D|Tennis) = \frac{1}{8} \cdot \frac{2}{7} \cdot \frac{4}{7} \cdot \frac{4}{7} = \frac{4}{343}$$

Nun benötigen wir nur noch die Schätzungen für die Wahrscheinlichkeiten der einzelnen Klassen. Mit Hilfe von (3.4) und den absoluten Häufigkeiten aus Tabelle 3.2 können wir diese berechnen:

$$\widehat{\Pr}(Golf) = \frac{t_{Golf}}{t} = \frac{6}{15} = \frac{2}{5}$$
$$\widehat{\Pr}(Squash) = \frac{t_{Squash}}{t} = \frac{4}{15}$$
$$\widehat{\Pr}(Tennis) = \frac{t_{Tennis}}{t} = \frac{5}{15} = \frac{1}{3}$$

Mit diesen Wahrscheinlichkeitsabschätzungen und denen der Attributwerten, können wir bereits das Trainingsbeispiel klassifizieren. Wir müssen hierfür nur noch jeweils die

beiden Abschätzungen einer Klasse multiplizieren (siehe (3.9).

$$\widehat{\Pr}(D|Golf) \cdot \widehat{\Pr}(Golf) = \frac{1}{48} \cdot \frac{2}{5} = \frac{1}{120} \approx 0,008333333$$
$$\widehat{\Pr}(D|Squash) \cdot \widehat{\Pr}(Squash) = \frac{1}{28} \cdot \frac{4}{15} = \frac{1}{135} \approx 0,00952381$$
$$\widehat{\Pr}(D|Tennis) \cdot \widehat{\Pr}(Tennis) = \frac{4}{343} \cdot \frac{1}{3} = \frac{4}{1029} \approx 0,003887269$$

Der Naive Bayes Klassifizierer ordnet folglich das Trainingsbeispiel der Klasse *Squash* zu, da das Produkt ihrer Wahrscheinlichkeitsabschätzungen den größten Wert hat. Wenn wir jedoch an einer Abschätzung der Wahrscheinlichkeit für jede Klasse interessiert sind, müssen wir die eben bestimmten Schätzungen normieren, da ihnen noch die normierende Wahrscheinlichkeit $\Pr(D)$ fehlt. Diese Wahrscheinlichkeit schätzen wir durch

$$\begin{aligned}\widehat{\Pr}(D) &= \widehat{\Pr}(D|Golf) \cdot \widehat{\Pr}(Golf) + \widehat{\Pr}(D|Squash) \cdot \widehat{\Pr}(Squash) \\ &\quad + \widehat{\Pr}(D|Tennis) \cdot \widehat{\Pr}(Tennis) \\ &\approx 0,008333333 + 0,00952381 + 0,003887269 \\ &= 0,021744412\end{aligned}$$

ab und erhalten damit die folgenden Wahrscheinlichkeitsabschätzungen für die einzelnen Klassen unter Beobachtung des Trainingsbeispiels D:

$$\widehat{\Pr}(Golf|D) = \frac{\widehat{\Pr}(D|Golf) \cdot \widehat{\Pr}(Golf)}{\widehat{\Pr}(D)} \approx \frac{0,008333333}{0,021744412} \approx 0,383240223$$
$$\widehat{\Pr}(Squash|D) = \frac{\widehat{\Pr}(D|Squash) \cdot \widehat{\Pr}(Squash)}{\widehat{\Pr}(D)} \approx \frac{0,00952381}{0,021744412} \approx 0,437988827$$
$$\widehat{\Pr}(Tennis|D) = \frac{\widehat{\Pr}(D|Tennis) \cdot \widehat{\Pr}(Tennis)}{\widehat{\Pr}(D)} \approx \frac{0,003887269}{0,021744412} \approx 0,17877095$$

3.4. Behandlung von fehlenden Attributwerten

Bei realen Problemen können Daten aus verschiedene Gründen unvollständig sein, unter anderem wegen unvollständig ausgefüllten Formularen (z.B. Aufnahmebögen in Krankenhäusern), nicht vorhandenen Daten (z.B. sind Blutwerte nicht für alle Patienten verfügbar) oder nicht zutreffenden Attributen (zum Beispiel geschlechtsspezifische Meßwerte). Diese Fälle können sowohl in den Trainingsdaten als auch in den Testdaten vorkommen. Wir werden beide Probleme auf eine einfache Weise beheben.

Während dem Training ignorieren wir die fehlenden Attributwerte einer Instanz und verwenden zur Berechnung der absoluten Häufigkeiten t_{c_i} und $t_{c_i}^{a_i}$ nur deren vorhandenen Attributwerte.

Bei der Klassifizierung einer Instanz mit fehlenden Attributwerten überspringen wir während der Berechnung des Produktes (3.5) diese Attribute. Dies entspricht der Annahme, daß diese Attributwerte für alle Klassen gleich wahrscheinlich sind. Sei $\widetilde{A}$ die Menge

der bekannten Attributwerte, dann können wir die Grundversion des Naive Bayes Klassifizier (3.9) folgendermaßen modifizieren:

$$c_{NB} = \arg\max_{c_i} \Pr(c_i) \prod_{a \in \widetilde{A}} \Pr(a|c_i) \tag{3.10}$$

Mit den oben genannten Maßnahmen haben wir die Möglichkeit, Daten mit fehlenden Attributwerten genauso zu behandeln wie vollständige Daten. In den folgenden Kapiteln werden wir jedoch nur vollständige Daten betrachten, die Aussagen und Feststellungen dieser Kapitel gelten aber entsprechend auch für unvollständige Daten.

3.5. Behandlung von kontinuierlichen Attributen

Es existieren drei Hauptmethoden zur Behandlung von kontinuierlichen bzw. numerischen Attributen bei der Anwendung eines Naive Bayes Klassifizierers. Die *Normalmethode* ist die klassische Methode, die eine Verteilung der kontinuierlichen Variablen durch eine parametrisierte Verteilung wie zum Beispiel die Normalverteilung approximiert. Die *Kernelmethode* [JL95] verwendet einen nicht parametrisierten Ansatz. *Diskretisierungsmethoden* [DKS95] diskretisieren zuerst die kontinuierlichen Variablen in diskrete Werte und transformieren das Lernproblem in eines ohne kontinuierliche Variablen. In den letzten Jahren hat man sich damit beschäftigt, wieso Diskretisierungsmethoden funktionieren [HHW00, YW03a, YW03b]. Im allgemeinen wird anerkannt, daß die *Normalmethode* eine schlechtere Performanz aufweist als die beiden anderen Methoden. In [Bou04] hat man jedoch festgestellt, daß keine der drei Methoden signifikant besser beziehungsweise vorzuziehen ist.

3.5.1. Normalmethode

Die klassische Methode die Wahrscheinlichkeit $\Pr(a|c_i)$ für einen Attributwert a eines kontinuierlichen Attributes A zu approximieren nimmt an, daß diese einer bekannten Verteilung zugrunde liegt, für die man die Parameter anhand der Trainingsdaten abschätzen kann. Die populärste Annahme ist, daß die Verteilung eine Normalverteilung ist. Aus diesem Grund nennen wir sie *Normalmethode*. Für jede Klasse c_i nehmen wir folgendes an:

$$\Pr(a|c_i) = N(a|\mu_{c_i}, \sigma_{c_i}), \tag{3.11}$$

wobei es sich bei $N(a|\mu, \sigma)$ um die Normalverteilung mit Mittelwert μ und Varianz σ handelt und wie folgt berechnet wird:

$$N(x|\mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \cdot e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{3.12}$$

Sei $a(k)$ $(0 < k \leq t)$ der Wert des Attributes A des k-ten Beispiels der Trainingsdaten und entsprechend $c(k)$ die Klasse des k-ten Beispiels. Die Mittelwerte μ_{c_i} und Varianzen σ_{c_i} werden durch die Attributwerte von A der Trainingsbeispiele, die zur Klasse c_i

gehören, unvoreingenommen abgeschätzt.

$$\mu_{c_i} = \frac{1}{t_{c_i}} \sum_{\substack{k=1 \\ c(k)=c_i}}^{t} a(k) \tag{3.13}$$

$$\sigma_{c_i} = \frac{1}{t_{c_i} - 1} \sum_{\substack{k=1 \\ c(k)=c_i}}^{t} (a(k)^2 - \mu_{c_i}^2) \tag{3.14}$$

Die Vorteile der Normalmethode sind, daß sie gut funktioniert, wenn die zugrundeliegende Verteilung einer Normalverteilung entspricht, daß sie eine schnelle Trainings- und Klassifikationszeit aufweist und daß sie nur wenig Speicher benötigt.

3.5.2. Kernelmethode

Die *Kernelmethode* approximiert $\Pr(a|c_i)$ für einen Attributwert a eines kontinuierlichen Attributes A durch eine Anzahl von sogenannten *Kernels*, bei denen es sich um Funktionen handelt, die um Datenpunkte zentriert sind. John und Langley [JL95] schlagen vor für diese Kernels Normalverteilungen zu verwenden, deren Mittelwerte diese Datenpunkte sind und deren Varianz durch $1/\sqrt{t_{c_i}}$ abgeschätzt wird. Die Wahrscheinlichkeit $\Pr(a|c_i)$ berechnet sich dann wie folgt:

$$\Pr(a|c_i) = \sum_{\substack{k=1 \\ c(k)=c_i}}^{t} N(a|a(i), \sigma_{c_i}) \tag{3.15}$$

Hierbei gelten die Notationen des vorangegangen Unterabschnitts 3.5.1.

Die Vorteile der Kernelmethode sind, daß keine Annahmen über die Verteilung der Variable a gemacht werden müssen, daß jede Verteilung approximiert werden kann und daß diese Methode bekannt für seine erwünschten asymptotischen Eigenschaften [JL95] ist. Für diese Methode muß jedoch jeder Attributwert für jedes der kontinuierlichen Attribute gespeichert werden. Dies kann bei großen Datensätzen zu einem erheblichen Speicherbedarf führen. Außerdem müssen für die Berechnung der Wahrscheinlichkeiten $\Pr(a|c_i)$ bis zu t Terme aufsummiert werden, daher verlängert sich die Klassifikationszeit bei größer werdenden Datensätzen.

3.5.3. Diskretisierungsmethoden

Diskretisierungsmethoden konvertieren eine kontinuierliche Variable a in eine diskrete Variable $\overline{a}$, mit der die Wahrscheinlichkeit $\Pr(\overline{a}|c_i)$ wie gewohnt abgeschätzt wird. Für die Diskretisierung müssen wir die Anzahl von Werten $v_{\bar{a}}$ der diskreten Variable $\overline{a}$ festlegen und die $v_{\bar{a}} - 1$ Grenzwerte oder Teilungspunkte bestimmen. Die Wahl von $v_{\bar{a}}$ und der Grenzwerte hat eine dramatische Auswirkung auf die Performanz des Klassifizierers [YW02].

Wir betrachten eine Diskretisierungsmethode, die auf der *Minimum-Description-Length (MDL)*-Methode von Fayyad und Irani [FI93] basiert, da sie weit verbreitet ist, häufig verwendet wird und verglichen mit anderen Methoden eine bessere Performanz aufweist [DKS95]. Jedes der Attribute mit kontinuierlichen Werten wird getrennt betrachtet. Falls Werte von a fehlen, werden diese Trainingsbeispiele ignoriert, aber zur Vereinfachung der Erklärung gehen wir davon aus, daß keine fehlenden Werte existieren.

Die Methode beginnt mit einem einzigen Intervall, das alle Trainingsbeispiele beinhaltet, und teilt die Intervalle rekursiv in zwei neue Intervalle auf. Bevor wir ein Intervall splitten, müssen wir entscheiden, welches der bereits vorhandenen Intervalle sich dafür am besten eignet beziehungsweise den meisten Nutzen bringt. Wir berechnen für jedes der vorhandenen Intervalle und alle Teilungspunkte, die im jeweiligen Intervall liegen, den *Information Gain*. Diese Maß ist die Differenz der *Entropie* (englisch: entropy) des ursprünglichen Intervalles und der Summe der *Entropien* von den zwei Teilintervallen. Das heißt, wenn I das ursprüngliche Intervall mit $|I|$ Beispielen ist und $I_{a<a_T}$ und $I_{a\geq a_T}$ die beiden Teilintervalle mit $|I_{a<a_T}|$ beziehungsweise $|I_{a\geq a_T}|$ Beispielen für den Teilungspunkt a_T sind, berechnet sich der Information Gain für diese Teilung wie folgt:

$$\begin{aligned} & \mathit{InformationGain}(I, a_T) \\ = \; & |I| \cdot \mathit{Entropie}(I) - |I_{a<a_T}| \cdot \mathit{Entropie}(I_{a<a_T}) - |I_{a\geq a_T}| \cdot \mathit{Entropie}(I_{a\geq a_T}), \end{aligned}$$

wobei die *Entropie* ein Maß für die Unreinheit von einer Menge von Werten ist und für ein Intervall I folgendermaßen berechnet wird:

$$\mathit{Entropie}(I) = -\sum_{c_i} t^I_{c_i} \log_2 \left| t^I_{c_i} \right|, \tag{3.16}$$

wobei $t^I_{c_i}$ die Anzahl der Trainingsbeispiele der Klasse c_i im Intervall I ist.

Haben wir für jeden Teilungspunkt den Information Gain berechnet, entscheiden wir uns für den Teilungspunkt, der den maximalen Wert erzielt hat. Die Intervallteilung an diesem Teilungspunkt wird nur akzeptiert, wenn der dadurch erzielte Information Gain größer ist als ein spezieller Schwellenwert. Dieser berechnet sich wie folgt:

$$\begin{aligned} \log_2(n-1) + \log_2(3^{m_I} - 2) - m_I \cdot \mathit{Entropie}(I) + m_{I_{a<a_T}} \cdot \mathit{Entropie}(I_{a<a_T}) \\ + m_{I_{a\geq a_T}} \cdot \mathit{Entropie}(I_{a\geq a_T}), \end{aligned}$$

wobei m_I, $m_{I_{a<a_T}}$ und $m_{I_{a\geq a_T}}$ die Anzahl der Klassen in den Intervallen I, $I_{a<a_T}$ und $I_{a\geq a_T}$ sind.

Sollte kein Teilungspunkt einen Information Gain aufweisen, der diesen Schwellenwert übertrifft, ist die Diskretisierung beendet. Eine Rechtfertigung für die Entscheidung für diesen Schwellenwert kann in [FI93] nachgelesen werden.

3.6. Behandlung von unstrukturierten Daten

Unstrukturierte Daten wie Texte, Web-Dokumente oder ähnliches haben weder eine feste Länge noch eine abgeschlossene Menge von Attributen. Aus diesem Grund können

wir nicht die in den vorherigen Kapiteln besprochenen Methoden auf diese Daten anwenden. Es existieren zwei Modelle, die das Konzept des Naive Bayes Klassifizierers auf unstrukturierte Daten übertragen. Bevor wir diese vorstellen, müssen wir noch eine alternative Notation für diesen Unterabschnitt einführen, die die am Anfang dieses Kapitels besprochene Notation ersetzt.

- T: die Menge aller Terme einer Sprache
- t: ein Term aus T
- E: die Trainingsmenge von Dokumenten
- E_c: die Beispielsmenge der Dokumente, die zur Klasse c gehören
- E_c^t: die Beispielsmenge der Dokumente, die den Term t beinhalten und zur Klasse c gehören
- $|E|$: die Kardinalität einer Menge von Dokumenten
- D: ein Dokument
- d: die Anzahl der Terme oder die Dokumentlänge von D
- n_c: die absolute Anzahl von Termen in E_c
- n_c^t: die absolute Häufigkeit des Terms t in E_c
- n_D^t: die absolute Häufigkeit des Terms t in D

Beide Modelle schätzen die Wahrscheinlichkeit der Klassen folgendermaßen ab:

$$\Pr(c_i) = \frac{|E_c|}{|E|} \tag{3.17}$$

Sie unterscheiden sich nur durch den Ansatz, den sie zur Berechnung der Wahrscheinlichkeit $\Pr(D|c_i)$ verwenden.

3.6.1. Binäres Modell

Das *binäre Modell* versucht die Wahrscheinlichkeit $\Pr(D|c_i)$ über die Wahrscheinlichkeiten der Terme des Dokumentes D abzuschätzen. Dabei bezeichnet der Parameter $\Phi_{c,t}$ die Wahrscheinlichkeit, daß in einem Dokument der Klasse c der Term t mindestens einmal auftritt:

$$\Phi_{c,t} = \frac{|E_c^t|}{|E_c|} \tag{3.18}$$

Mit dieser Definition erhalten wir folgenden Ansatz:

$$\Pr(D|c) = \prod_{t \in D} \Phi_{c,t} \prod_{\substack{t \in T \\ t \notin D}} (1 - \Phi_{c,t}) \tag{3.19}$$

Da wir nicht $\prod_{t \in T, t \notin D}(1 - \Phi_{c,t})$ für jedes Testdokument berechnen wollen, schreiben wir (3.19) wie folgt um:

$$\Pr(D|c) = \prod_{t \in D} \frac{\Phi_{c,t}}{1 - \Phi_{c,t}} \prod_{t \in T}(1 - \Phi_{c,t}) \tag{3.20}$$

Wir berechnen und speichern $\prod_{t \in T}(1-\Phi_{c,t})$ im voraus für alle c. Zur Klassifikationszeit bestimmen wir nur noch das erste Produkt.

Normalerweise ist die Anzahl der möglichen Terme $|T|$ erheblich größer als die Anzahl der Terme des Dokumentes D, deshalb sollte dieses Modell bei kleinen Dokumenten nicht verwendet werden.

3.6.2. Multinomiales Modell

Bei dem *multinomialen Modell* berechnen wir die Wahrscheinlichkeit $\Pr(D|c_i)$ durch die Wahrscheinlichkeiten, daß ein Term t in D_c vorkommt. Diese Wahrscheinlichkeit bezeichnen wir mit $\Theta_{c,t}$:

$$\Theta_{c,t} = \frac{n_c^t}{n_c} \tag{3.21}$$

Für ein Dokument der Länge d berechnet man das Modell wie folgt:

$$\begin{aligned} \Pr(D|c) &= \Pr(d|c) \Pr(D|d,c) \\ &= \Pr(d|c) \binom{d}{\{n_D^t\}} \prod_{t \in D} (\Theta_{c,t})^{n_D^t} \end{aligned} \tag{3.22}$$

$\binom{d}{\{n_D^t\}}$ bezeichnet hier den Multinomialkoeffizenten. Falls wir nur an dem Ranking der Klassen bzw. der Klassifikation interessiert sind, können wir ihn wegfallen lassen, da er für alle Klassen gleich ist. Man trifft häufig die fragwürdige Annahme, daß die Verteilung der Dokumentlänge für alle Klassen gleich ist, und läßt deshalb den Term $\Pr(d|c)$ wegfallen.

Die Wahrscheinlichkeit $\Pr(d|c)$, daß ein Dokument der Klasse c die Länge d hat, kann mit den Methoden zur Behandlung von kontinuierlichen Variablen aus Abschnitt 3.5 abgeschätzt werden.

4. Ensemble-Methoden

Ensemble-Methoden haben in den letzten Jahren beträchtliche Aufmerksamkeit innerhalb der Literatur des Maschinellen Lernens genossen [Die97, Die00a, OM99, BK99]. Die Idee, eine Menge unterschiedlicher Klassifizierer für ein einzelnes Lernproblem zu erlernen und deren Vorhersagen zur Abstimmung zu verwenden oder zu mitteln, ist sowohl einfach als auch wirksam. Die resultierenden Genauigkeitsverbesserungen haben oft eine solide theoretische Grundlage [FS97, Bre96a]. Das Mitteln der Vorhersagen von multiplen Klassifizierern reduziert die Varianz und erhöht oft die Verläßlichkeit des Klassifizierers.

Die bekannteste Technik ist die Anwendung von *Subsampling*, um die Trainingsmenge zu diversifizieren, wie zum Beispiel bei *Bagging* [Bre96a] und *Boosting* [FS97]. Andere Techniken nutzen die Zufälligkeit der Basisalgorithmen [KP90], zum Beispiel indem man ihr Verhalten künstlich zufällig macht [Die00b], verwenden unterschiedliche *Feature-Subsets* [Bay99] oder multiple Repräsentationen der *Objekte des Wertebereiches* (zum Beispiel die Verwendung von Informationen, die von verschiedenen Hyperlinks stammen, die auf eine Web-Seite linken [Für02a]). Zu guter Letzt können wir verschiedene Klassifizierer erhalten, indem wir die Vorhersagelabel modifizieren. Das heißt wir transformieren die Lernaufgabe in eine Sammlung von verwandten Lernaufgaben, die die gleichen Trainingsbeispiele, aber unterschiedliche Zuweisungen der Klassenlabels verwenden. *Error-Correcting-Output-Codes* (ECOC) sind die bekanntesten Vertreter dieser Art von Ensemble-Methoden [DB95].

Trotz all dieser Möglichkeiten möchten wir uns auf Bagging, Boosting und Klassenbinarisierungen beschränken. Wir werden im Verlauf dieses Kapitels die Grundidee und mindestens einen Vertreter jeder dieser Methoden vorstellen.

4.1. Bagging

Der *Bagging*-Algorithmus (**B**ootstrap **agg**regat**ing** von Breiman [Bre96a]) verwendet Klassifizierer zur Abstimmung, die auf unterschiedlichen *Bootstrap-Samples* trainiert werden. Ein Bootstrap-Sample [ET93] wird durch *gleichmäßiges Ziehen mit Zurücklegen* von t Instanzen aus der Trainingsmenge generiert. Bei R Wiederholungen erhalten wir die Bootstrap-Samples $B_1, B_2, ..., B_R$. Dieses Vorgehensweise wird mit *Resampling* bezeichnet. Für jedes Bootstrap-Sample B_i wird ein Klassifizierer C_i gelernt. Ein endgültiger Klassifizierer C^* wird aus den Klassifizierern $C_1, C_2, ..., C_R$ gebildet. Dessen Vorhersage ist die Klasse, die von seinen Basisklassifizierern am häufigsten vorhergesagt wird, wobei bei einem Unentschieden willkürlich klassifiziert wird.

Für ein gegebenes Bootstrap-Sample wird eine Instanz der Trainingsmenge mit der

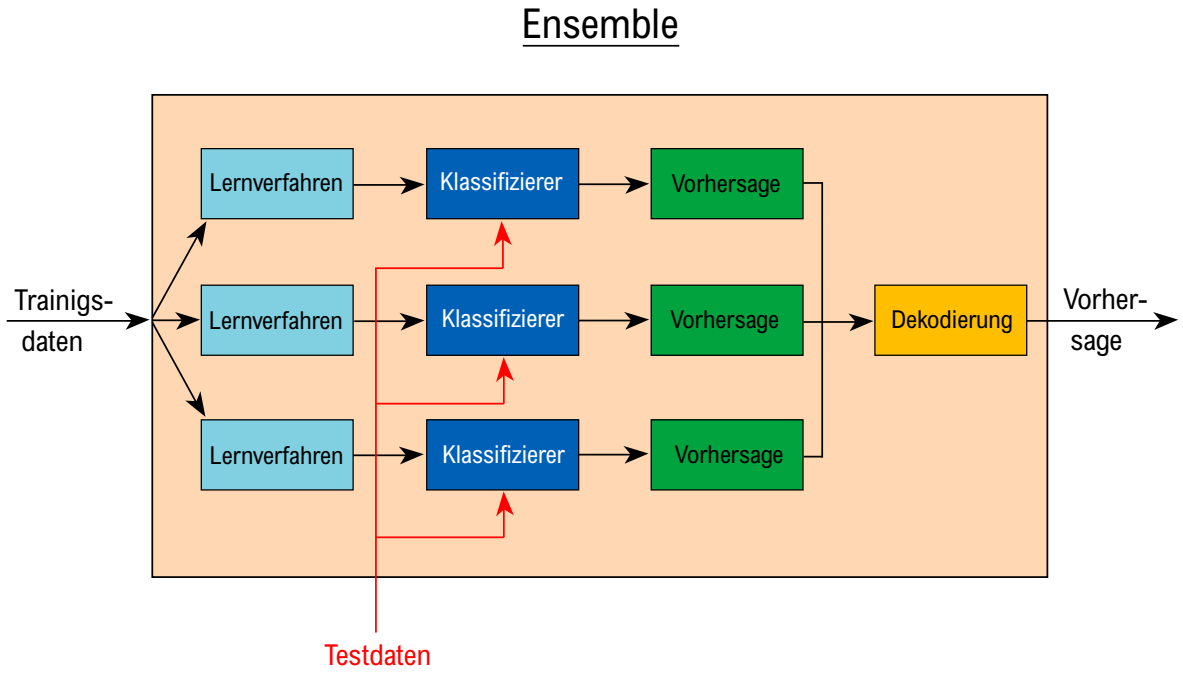


Abbildung 4.1.: Schema eines Ensemble bestehend aus drei Basisklassifizierern

Wahrscheinlichkeit $1 - (1 - 1/t)^t$ mindestens einmal gezogen, wenn t Trainingsbeispiele zufällig ausgewählt werden. Für große t beträgt diese Wahrscheinlichkeit in etwa $1 - 1/e = 63,2\%$. Dies bedeutet, daß jedes Bootstrap-Sample nur zu ungefähr 63,2% aus einfach vorkommenden Instanzen der Trainingsmenge besteht. Diese Perturbation erzeugt unterschiedliche Klassifizierer, falls der Basisklassifizierer instabil ist (zum Beispiel Neuronale Netze, Entscheidungsbäume) [Bre96b]. Die Performanz wird verbessert, falls die Basisklassifizierer gut und nicht korreliert sind. Dennoch kann Bagging die Performanz eines stabilen Algorithmus (z.B. k-Nearest Neighbor) leicht verschlechtern, weil effektiv eine kleinere Trainingsmenge zum Trainieren verwendet wird [Bre96a].

4.2. Boosting

Boosting wurde von Schapire [Sch90] als eine Methode zur Erhöhung (englisch: boosting) der Performanz eines schwachen Lernalgorithmus eingeführt. Nach Verbesserungen durch Freund [Fre95] wurde *AdaBoost* (**Ada**ptive **Boost**ing) von Freund und Schapire [FS97] vorgeschlagen. Wir konzentrieren uns auf AdaBoost, oft auch AdaBoost.M1 oder AdaBoostM1 genannt (z.B. [FS96]), als Boosting-Methode.

Wie Bagging generiert der AdaBoost-Algorithmus ein Ensemble von Klassifizierern und klassifiziert mit ihrer Hilfe. Bis auf diesen Punkt unterscheiden sich diese beiden Algorithmen grundlegend. Der AdaBoost-Algorithmus generiert sequentiell die Klassifi-

zierer, während sie bei Bagging parallel berechnet werden können. AdaBoost verändert im Gegensatz zu Bagging die Gewichte der Trainingsbeispiele, die als Eingabe für jeden Basisklassifizierer dienen, abhängig von den bereits trainierten Klassifizierern. Diese Methode nennt man *Reweighting*. Falls der Basisklassifizierer nicht mit gewichteten Trainingsbeispielen umgehen kann, verwenden wir anstatt dessen *Resampling*. Ziel ist, den zu erwartenden Fehler durch verschiedene Verteilungen der Eingabe zu minimieren. Für eine gegebene Anzahl R von Versuchen generieren wir aufeinanderfolgend R gewichtete Trainingsmengen $T_1, T_2, ..., T_R$ und trainieren R Klassifizierer $C_1, C_2, ..., C_R$. Ein endgültiger Klassifizierer C^* wird durch Weighted Voting der Vorhersagen der $C_1, C_2, ..., C_R$ gebildet. Das Gewicht jedes Klassifizierers ist abhängig von seiner Performanz auf der Trainingsmenge, die zu seinem Training verwendet wurde.

4.3. Klassenbinarisierung

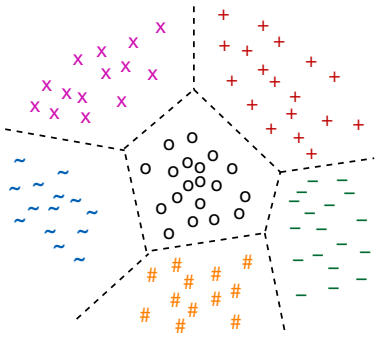
In diesem Abschnitt werden wir uns mit *Klassenbinarisierungen* (englisch: class binarization) beschäftigen. Diese Techniken lösen Multiklassenprobleme durch Transformation in mehrere binäre Probleme. Für jedes binäre Problem wird ein eigener Klassifizierer aufgebaut. Anhand der Vorhersagen dieser Klassizierer wird die Vorhersage für das Multiklassenproblem abgeschätzt. Für diese Vorgehensweise gibt es verschiedene Gründe.

Zum einen sind reale Probleme häufig Multiklassenprobleme. Viele Klassifizierer sind jedoch inhärent binär, das heißt sie können nur zwischen zwei Klassen diskriminieren. Hierfür können unter anderem Beschränkungen der Hypothesensprachen (z.B. lineare Diskriminanten oder Support-Vektor-Maschinen), die Lernarchitektur (z.B. Neuronale Netze mit einem einzelnen Ausgabeknoten) oder das Lern-Framework (z.B. sind viele Regellerner auf das Konzeptlernen, das heißt auf das Problem eine Konzeptbeschreibung von positiven und negativen Beispielen zu finden, zugeschnitten) verantwortlich sein.

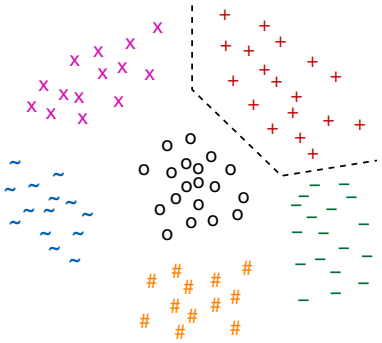
Bekannte Beispiele für binäre Klassifizierer, auf die wir nicht weiter eingehen werden, sind Perceptrons, Support-Vektor-Maschinen, der ursprüngliche AdaBoostAlgorithmus und Separate-and-Conquer-Regellerner. Es existieren zwei Ansätze derartige Algorithmen auf Multiklassenprobleme anzuwenden. Bei dem ersten generalisiert man den Algorithmus, dies wurde zum Beispiel für Support Vektor Maschinen [WW99] und Boosting [FS97] realisiert. Wir werden uns jedoch auf die zweite Möglichkeit beschränken, bei der Techniken der Klassenbinarisierung angewandt werden.

Zum anderen können diese Klassenbinarisierungen auch bei Klassifizierern, die mit Multiklassenproblemen umgehen können, angewandt werden. Wir erhalten auf diese Weise eine Ensemble von Klassifizierern [Für03]. Dies kann, wie bereits am Anfang dieses Abschnittes erwähnt, zu einer Verbesserung der Performanz dieser Basisklassifizierer führen.

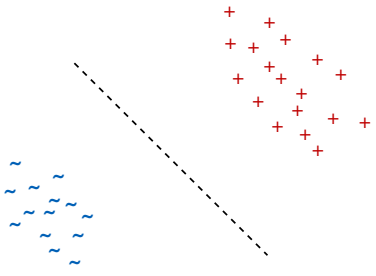
Definition 4.1 (Klassenbinarisierung) Eine *Klassenbinarisierung* ist eine Aufteilung eines Multiklassenlernproblems in mehrere binäre Lernprobleme. Diese Aufteilung geschieht auf eine Weise, die eine *Dekodierung* der Vorhersage ermöglicht, das heißt man kann die Vorhersage für das Multiklassenproblem von den Vorhersagen der binären Klas-



(a) *Multiklassenproblem*
Ein Klassifizierer separiert alle Klassen



(b) *Ungeordnete Klassenbinarisierung*
m Klassifizierer, jeder separiert eine Klasse von allen anderen.
Hier: + gegen die anderen Klassen



(c) *Round Robin Klassenbinarisierung*
$m(m\text{-}1)/2$ Klassifizierer, jeder für ein Klassenpaar. Hier: + gegen ~.

Abbildung 4.2.: Ungeordnete und Round Robin Klassenbinarisierung für ein Multiklassenproblem mit 6 Klassen [Für02b]

sifizierer ableiten. Den Lernalgorithmus, der zur Lösung der binären Probleme verwendet wird, nennt man *Basisklassifizierer*.

4.3.1. Ungeordnete/one-against-all Klassenbinarisierung

Die bekannteste Klassenbinarisierungstechnik ist die *ungeordnete* oder *One-against-all-Klassenbinarisierung*, bei der für jede Klasse c_i ein binäres Konzept gelernt wird, das zwischen dieser Klasse und allen anderen Klassen diskriminiert. Diese Technik wurde unabhängig voneinander für das Regellernen [CB91], Neuronale Netze [AMMR95] und Support-Vektor-Maschinen [CV95] vorgeschlagen.

Definition 4.2 (Ungeordnete/one-against-all Klassenbinarisierung) Die *ungeordnete Klassenbinarisierung* transformiert ein m-Klassenproblem in m binäre Probleme. Diese konstruiert man, indem man die Beispiele der Klasse c_i als positive Beispiele und die Beispiele der Klasse c_j $(j = 1...m, j \neq i)$ als negative Beispiele verwendet.

Die Bezeichnung „*ungeordnet*“ stammt von Clark und Boswell [CB91], die diesen Ansatz als Alternative zum „Decision-list learning“-Ansatz vorgeschlagen haben, der ursprünglich für CN2 [CN89] verwendet wurde. „Ungeordnet“ ist im Bereich des Regellernens gebräuchlicher, während in anderen Bereichen häufiger der Begriff „one-against-all“ verwendet wird. Obwohl beide für den gleichen Ansatz zur Klassenbinarisierung stehen und austauschbar verwendet werden können, werden wir im folgenden diese Klassenbinarisierung mit „ungeordnet“ bezeichnen und leiten nur ihre Abkürzung „1vsAll“ von dem Begriff „one-against-all“ ab.

Die *ungeordnete* Klassenbinarisierung klassifiziert ein Beispiel, indem die einzelnen Vorhersagen der binären Klassifizierer zu einer einzigen Vorhersage kombiniert werden. Typische Dekodierungsmethoden verwenden die Vorhersagen der einzelnen Klassifizierer als Stimmen, die gegebenfalls die Konfidenz der Vorhersagen berücksichtigen. Wir gehen im Abschnitt 4.4 näher auf diese Methoden ein.

4.3.2. Geordnete Klassenbinarisierung

Die *geordnete Klassenbinarisierung* ist eine Variante der *ungeordneten* Klassenbinarisierung, die eine feste Reihenfolge der induzierten Klassifizierer festlegt. Während der Klassifizierung testet man in dieser Reihenfolge, ob ein Testbeispiel zu einer Klasse gehört. Das heißt, wir verwenden zuerst den Klassifizierer, der zwischen der Klasse c_1 und den übrigen Klassen $c_2, ..., c_n$ diskriminiert. Falls der Klassifizierer das Beispiel der Klasse c_1 zuordnet, rufen wir keine weiteren Klassifizierer auf. Ansonsten reichen wir das Beispiel an den nächsten Klassifizierer weiter, der zwischen der Klasse c_2 und den restlichen Klassen unterscheidet.

Im Gegensatz zur *ungeordneten* Klassenbinarisierung müssen wir hier nicht alle Klassifizierer aufrufen. Wir benötigen auch keine Dekodierung der Vorhersagen der einzelnen Klassifizierer.

Definition 4.3 (Geordnete Klassenbinarisierung) Die *geordnete Klassenbinarisierung* transformiert ein m-Klassenproblem in $m - 1$ binäre Probleme. Diese konstruiert man, indem man die Beispiele der Klasse c_i $(i = 1...m - 1)$ als positive Beispiele und die Beispiele der Klassen c_j (j > i) als negative Beispiele verwendet.

4.3.3. Round Robin Klassenbinarisierung

In diesem Abschnitt befassen wir uns mit einer komplexeren Technik zur Klassenbinarisierung, der *paarweisen Klassenbinarisierung* [Für02b, Für03]. Deren Grundidee ist es, für jedes Klassenpaar einen eigenen Klassifizierer zu erlernen. Man nennt diese Vorgehensweise auch *Round Robin Klassenbinarisierung*. Diese Bezeichnung stammt von

Round Robin Wettbewerben in den Bereichen des Sportes und der Spiele, bei denen jeder Teilnehmer gegen jeden anderen Teilnehmer antritt.

Definition 4.4 (Round Robin/paarweise Klassenbinarisierung) Die *Round Robin* oder *paarweise Klassenbinarisierung* transformiert ein m-Klassenproblem in $m(m-1)/2$ binäre Probleme $< c_i, c_j >$, eines für jedes Klassenpaar $c_{ij}, i = 1...c-1, j = i+1...c$. Den binären Klassifizierer des Problems $< c_i, c_j >$ trainiert man mit den Beispielen der Klassen c_i und c_j. Die Beispiele aller anderen Klassen werden bei diesem Problem ignoriert.

Bei der obigen Definition nehmen wir an, daß das Problem die Klasse c_i von der Klasse c_j zu diskriminieren das gleiche ist wie das Problem zwischen der Klasse c_j und der Klasse c_i zu unterscheiden. Dies ist nur der Fall, falls der Basisklassifizierer *klassensymmetrisch* ist. Algorithmen des Regellernens sind nicht zwingend *klassensymmetrisch*. Viele von ihnen wählen eine der beiden Klassen als Standardklasse aus und lernen nur Regeln, die die andere Klasse abdecken. In solchen Fällen können $< c_i, c_j >$ und $< c_j, c_i >$ zwei unterschiedliche Klassifikationsprobleme sein, da bei ersterem die Klasse c_i die Standardklasse ist und bei letzterem c_j. Ein einfacher Ansatz dieses Problem zu lösen ist die *doppelte Round Robin* Klassenbinarisierung, bei der separate Klassifizierer für beide Probleme $< c_i, c_j >$ und $< c_j, c_i >$ trainiert werden.

Definition 4.5 (Doppelte Round Robin/paarweise Klassenbinarisierung) Die *doppelte Round Robin Klassenbinarisierung* transformiert ein m-Klassenproblem in $m(m-1)$ binäre Probleme $< c_i, c_j >$, eines für jedes Klassenpaar $(c_i, c_j), j = 1...c, j \neq i$. Die Beispiele der Klasse c_i verwendet man als positive Beispiele und die Beispiele der Klasse c_j als negative Beispiele.

Sowohl bei der *einfachen* als auch bei der *doppelten Round Robin Klassenbinarisierung* benötigen wir Dekodierungsmethoden, um die Vorhersagen der einzelnen, binären Klassifizierer zu einer einzigen Vorhersage zu kombinieren. Auf diese gehen wir im folgenden Abschnitt ein.

4.4. Dekodierungsmethoden

Wir werden in diesem Abschnitt verschiedene Möglichkeiten zur *Dekodierung* der Vorhersagen eines Ensembles von Klassifizierern zu einer einzigen Vorhersage vorstellen. Dabei beschränken wir uns auf Basisklassifizierer, die zusätzlich zu ihrer Klassifikation ein Konfidenzmaß für diese generieren, da dieses Maß für alle Methoden außer Voting benötigt wird. Diese Einschränkung ist nicht gravierend, weil wir für jeden Basisklassifizierer ein Konfidenzmaß künstlich erzeugen können (z.B. durch eine Abschätzung der Genauigkeit eines Klassifizierers anhand einer Kreuzvalidierung). Wir werden zwei Typen von Methoden vorstellen: *Abstimmungsmethoden* und *Bradley-Terry-Methoden*.

Die Methoden *Voting* und *Weighted Voting* verwenden die Vorhersagen der einzelnen Klassifizierer für eine gegebenenfalls gewichtete Abstimmung und sind deshalb für alle vorgestellten Ensemble-Methoden geeignet.

Die Bradley-Terry-Methoden eignen sich hingegen nur für die Round Robin Klassenbinarisierung, da sie anhand von paarweisen Vorhersagen ein Ranking der Klassen bestimmen. Bei der *Methode von Refregier und Vallet* verwenden wir eine Anzahl dieser Vorhersagen zum Aufbau eines linearen Gleichungssystems, dessen Lösung ein Ranking der Klassen ergibt [RV91]. Die *Methode von Price, Knerr, Personnaz und Dreyfus* berechnet anhand einer Formel das Ranking der Klassen [PKPD94]. Bei der *Methode von Hastie und Tibshirani* wird ein konvergenter Algorithmus verwendet, dessen Resultat ein Ranking der Klassen ist [HT97].

Bei der Beschreibung dieser Methoden werden wir das Konfidenzmaß durch Wahrscheinlichkeiten ausdrücken. Dabei erweitern wir die Notationen aus Kapitel 3 folgendermaßen.

- $\Pr(c_i|D)$: die Konfidenz eines Klassifizierers in Klasse c_i. Bei allgemeinen Ensemble-Methoden kennzeichnen wir diese Wahrscheinlichkeit noch durch den Index dieses Klassifizierers. Bei der geordneten Klassenbinarisierung bezieht er sich auf das binäre Problem c_i gegen die restlichen Klassen.
- $\Pr(c_i|D, c_{ij})$: die Konfidenz des Klassifizierers in Klasse c_i des Klassenpaares c_{ij} bei einer Round Robin Klassenbinarisierung
- E: ein Ensembles von Klassifizierern, zum Beispiel bei Boosting und Bagging

4.4.1. Abstimmungsmethoden

Bei *Abstimmungsmethoden* verwenden wir wie bereits erwähnt die Vorhersagen und gegebenenfalls die Konfidenz der einzelnen Klassifizierer eines Ensembles in diese Vorhersagen zur Bestimmung einer endgültigen Vorhersage. Alle haben gemeinsam, daß wir zuerst die Vorhersagen und die Konfidenz aller Klassifizierer eines Ensembles bestimmen und danach durch Aufsummierung dieser Stimmen eine Vorhersage treffen. Falls man statt der Vorhersagen die Konfidenz zur Abstimmung verwendet, redet man von einer *gewichteten Abstimmung*.

Allgemeine Ensemble-Verfahren (wie z.B. Bagging, Boosting, usw.) und die Round Robin Klassenbinarisierung unterscheiden sich etwas in der Dekodierung dieser Stimmen, weshalb wir bei der Beschreibung der Abstimmungsmethoden auf beide getrennt eingehen werden. Beide lassen sich jedoch durch ein Grundgerüst beschreiben, das abhängig von der gewählten Abstimmungsfunktion die gewünschte Abstimmung realisiert.

Das Grundgerüst für allgemeine Ensemble-Verfahren sieht wie folgt aus:

$$\arg\max_{c_i} \sum_{e \in E} vote(c_i, e)$$

Die Voting-Funktion $vote(c_i, e)$ berechnet hier die Stimme für die Klasse c_i, die sie von dem Klassifizierers e aus dem Ensemble E erhält.

Betrachten wir nun das Grundgerüst der Round Robin Klassenbinarisierung, sehen wir schon die Unterschiede zu allgemeinen Ensemble-Verfahren:

$$\arg\max_{c_i} \sum_{j \neq i} vote(c_i, c_j)$$

Die Voting-Funktion $vote(c_i, c_j)$ berechnet hier die Stimme für die Klasse c_i, die sie von dem Klassifizierer des Klassenpaares c_{ij} erhält.

Nachdem wir die beiden Grundgerüste kennen, können wir uns mit den unterschiedlichen Abstimmungsfunktionen befassen.

Voting

Bei der *ungewichteten Abstimmung*, die wir im folgenden mit *Voting* bezeichnen, verwenden wir nur die Klassifikationen der einzelnen Klassifizierer als diskrete Stimmen. Dabei erhält jede Klasse für jeden Klassifizierer, der diese Klasse vorhersagt, eine ganzzahlige Stimme. Wir kennzeichnen die zugehörige Abstimmungsfunktion durch den Index V.

Die Abstimmungsfunktion $vote_V$ lautet für ein allgemeines Ensemble-Verfahren wie folgt:

$$vote_V(c_i, e) = \begin{cases} 1, & \text{falls der Klassifizier e die Klasse } c_i \text{ vorhersagt} \\ 0, & \text{sonst} \end{cases}$$

Für die Round Robin Klassenbinarisierung sieht die Abstimmungsfunktion $vote_V$ ähnlich aus:

$$vote_V(c_i, c_j) = \begin{cases} 1, & \text{falls der Klassifizierer des Klassenpaars } c_{ij} \text{ die Klasse } c_i \text{ vorhersagt} \\ 0, & \text{sonst} \end{cases}$$

Weighted Voting

Wir verwenden bei der *gewichteten Abstimmung*, die wir im folgenden auch mit *Weighted Voting* bezeichnen, nicht nur die Klassifikation der einzelnen Klassifizierer sondern auch deren Konfidenz zur Abstimmung. Jede Klasse erhält von jedem Klassifizierer eine Stimme in Höhe seiner Konfidenz in diese Klasse als gewichtete Stimme, die im allgemeinen nicht ganzzahlig ist.

Wir wollen dies kurz an einem Beispiel verdeutlichen. Angenommen ein Klassifizierer ist sich zu 35% (Konfidenz = 0,35) sicher, daß das Testbeispiel zu einer Klasse gehört, dann erhält diese Klasse demnach von dem Klassifizierer 0,35 als gewichtete Stimme.

Die Abstimmungsfunktion $vote_{WV}$ sieht für allgemeine Ensemble-Verfahren folgendermaßen aus:

$$vote_{WV}(c_i, e) = \Pr(c_i|D)_e$$

Für die Round Robin Klassenbinarisierung lautet die Abstimmungsfunktion $vote_{WV}$ wie folgt:

$$vote_{WV}(c_i, c_j) = \Pr(c_i|D, c_{ij})$$

4.4.2. Bradley-Terry-Methoden

Die Methoden, die wir als *Bradley-Terry-Methoden* bezeichnen, basieren auf dem *Bradley-Terry-Modell*. Dieses besteht aus der Annahme, daß

$$\Pr(c_i|D, c_{ij}) = \frac{\Pr(c_i|D)}{\Pr(c_i|D) + \Pr(c_j|D)} \tag{4.1}$$

gilt. Diesem Ansatz zufolge kann von den Wahrscheinlichkeiten der Klassenpaare auf die Wahrscheinlichkeiten der Klassen geschlossen werden und umgekehrt. Verständlicherweise ist dies nicht zwingend der Fall. Man kann sich zum Beispiel leicht anhand von Sportturnieren überlegen, daß die Spielergebnisse von verschiedenen Spielerpaarungen nicht unbedingt auf die Ergebnisse anderer Spielerpaarungen bezwiehungsweise auf die Fähigkeiten der beteiligten Spieler schließen lassen. Nichtsdestotrotz versuchen die Methoden, die wir in diesem Abschnitt vorstellen werden, mittels dieser Annahme die Wahrscheinlichkeit $\Pr(c_i|D)$ für alle Klassen c_i zu bestimmen. Aus diesem Grund eignen sich die Bradley-Terry-Methoden nur für Basisklassifizierer, die zusätzlich zur Klassifikation auch ein Maß von Konfidenz in diese beziehungsweise eine Wahrscheinlichkeit von dieser bereitstellen.

Diese Methoden erhalten alle als Ausgangsgröße eine Schätzung $\widehat{\Pr}(c_i|D, c_{ij})$ der paarweisen Wahrscheinlichkeiten $\Pr(c_i|D, c_{ij})$ für alle Klassenpaare c_{ij}. Anhand dieser Schätzungen versuchen die Methoden eine Abschätzung $\widehat{\Pr}(c_i|D)$ der Wahrscheinlichkeiten der einzelnen Klassen zu bestimmen, mit denen wir die abgeschätzten Wahrscheinlichkeiten $\overline{\Pr}(c_i|D, c_{ij})$ mit Hilfe der Annahme 4.1 berechnen.

$$\widehat{\Pr}(c_i|D, c_{ij}) \approx \frac{\widehat{\Pr}(c_i|D)}{\widehat{\Pr}(c_i|D) + \widehat{\Pr}(c_j|D)} = \overline{\Pr}(c_i|D, c_{ij}) \tag{4.2}$$

Dabei versuchen die Methoden die Differenz zwischen den geschätzten Wahrscheinlichkeiten $\widehat{\Pr}(c_i|D, c_{ij})$ und $\overline{\Pr}(c_i|D, c_{ij})$ zu minimieren.

Die Methoden unterscheiden sich durch den Ansatz, den sie zur Abschätzung dieser Wahrscheinlichkeiten verwenden. Wir beschränken uns auf drei Methoden, die oft in der Literatur erwähnt werden. Sie decken jedoch in keinsterweise das ganze Spektrum von auf dem Bradley-Terry-Modell basierenden Methoden ab [WLW04].

Methode von Refregier und Vallet

Refregier und Vallet [RV91] berücksichtigen, daß

$$\frac{\widehat{\Pr}(c_i|D, c_{ij})}{\widehat{\Pr}(c_j|D, c_{ij})} \approx \frac{\overline{\Pr}(c_i|D, c_{ij})}{\overline{\Pr}(c_j|D, c_{ij})} = \frac{\widehat{\Pr}(c_i|D)}{\widehat{\Pr}(c_j|D)} \tag{4.3}$$

gilt. Wenn wir aus (4.3) ein Gleichung machen, erhalten wir eine Möglichkeit die $\widehat{\Pr}(c_i|D)$ zu berechnen:

$$\widehat{\Pr}(c_i|D, c_{ij}) \cdot \widehat{\Pr}(c_j|D) = \widehat{\Pr}(c_j|D, c_{ij}) \cdot \widehat{\Pr}(c_i|D) \tag{4.4}$$

Die Anzahl von Gleichungen $(m \cdot (m-1)/2)$ ist jedoch größer als die Anzahl der unbekannten Wahrscheinlichkeiten (m). Refregier und Vallet schlagen deswegen vor, nur $m-1$ Gleichungen auszuwählen. Nehmen wir dann noch die Bedingung $\sum_i^m \widehat{\Pr}(c_i|D) = 1$ hinzu, können wir die Wahrscheinlichkeiten $\widehat{\Pr}(c_i|D)$ durch das Lösen eines linearen Gleichungssystems bestimmen.

Ein Problem dieser Methode ist jedoch, daß das Ergebnis stark von der Auswahl der $m-1$ Gleichungen abhängig ist [PKPD94]. In [WLW04] wurde ein Vorschlag gemacht, wie dieses Problem gemindert werden kann.

Methode von Price, Knerr, Personnaz und Dreyfus

Price, Knerr, Personnaz und Dreyfus [PKPD94] ziehen in Betracht, daß

$$\left(\sum_{j\neq i} \Pr\left(c_{ij}|D\right)\right) - (m-2) \cdot \Pr\left(c_i|D\right) = \sum_{j=1}^{m} \Pr\left(c_j|D\right) = 1 \tag{4.5}$$

gilt. Lösen wir (4.5) nach $\Pr\left(c_i|D\right)$ auf, erhalten wir folgende Gleichung:

$$\begin{aligned} & \left(\sum_{j\neq i} \Pr\left(c_{ij}|D\right)\right) - (m-2) \cdot \Pr\left(c_i|D\right) = 1 \\ \Leftrightarrow \quad & \left(\left(\sum_{j\neq i} \frac{\Pr\left(c_{ij}|D\right)}{\Pr\left(c_i|D\right)}\right) - (m-2)\right) \cdot \Pr\left(c_i|D\right) = 1 \\ \Leftrightarrow \quad & \Pr\left(c_i|D\right) = \frac{1}{\left(\sum_{j\neq i} \frac{\Pr(c_{ij}|D)}{\Pr(c_i|D)}\right) - (m-2)} \end{aligned} \tag{4.6}$$

Übertragen wir (4.2) mit Hilfe von

$$\widehat{\Pr}(c_{ij}|D) = \widehat{\Pr}(c_i|D) + \widehat{\Pr}(c_j|D) \tag{4.7}$$

auf (4.6) an, erhalten wir folgenden Berechnungsansatz:

$$\widehat{\Pr}\left(c_i|D\right)_{PKPD} = \frac{1}{\left(\sum_{j\neq i} \frac{1}{\widehat{\Pr}(c_i|D,c_{ij})}\right) - (m-2)} \tag{4.8}$$

Da für diese Schätzung $\sum_{i=1}^{m} \widehat{\Pr}\left(c_i|D\right)_{PKPD} = 1$ nicht gilt, müssen wir die Wahrscheinlichkeiten noch normalisieren. In den folgenden Kapiteln werden wir diese Methode durch PKPD abkürzen.

Algorithmus 4.1 Methode von Hastie und Tibshirani

Input: $\widehat{\Pr}(c_i|D, c_{ij})$, $t_{c_{ij}}$, $i, j \in \{1, ..., m\}$, $j \neq i$

Output: $\widehat{\Pr}(c_i|D)$, $i = 1, ..., m$

1: Starte mit beliebigen $\widehat{\Pr}(c_i|D) > 0$ für alle i und entsprechenden $\overline{\Pr}(c_i|D, c_{ij}) = \frac{\widehat{\Pr}(c_i|D)}{\widehat{\Pr}(c_i|D)+\widehat{\Pr}(c_i|D)}$
2: **repeat**
3: **for all** $i \in 1, 2, ..., m$ **do**
4: $\alpha = \frac{\sum_{j \neq i} t_{c_{ij}} \cdot \widehat{\Pr}(c_i|D,c_{ij})}{\sum_{j \neq i} t_{c_{ij}} \cdot \overline{\Pr}(c_i|D,c_{ij})}$
5: $\overline{\Pr}(c_i|D, c_{ij}) \longleftarrow \frac{\alpha \cdot \overline{\Pr}(c_i|D,c_{ij})}{\alpha \cdot \overline{\Pr}(c_i|D,c_{ij})+\overline{\Pr}(c_j|D,c_{ij})}$
6: $\overline{\Pr}(c_j|D, c_{ij}) \longleftarrow 1 - \overline{\Pr}(c_i|D, c_{ij})$
7: $\widehat{\Pr}(c_i|D) = \alpha \cdot \widehat{\Pr}(c_i|D)$
8: **end for**
9: **until** m aufeinanderfolgende α gegen 1 konvergieren

Methode von Hastie und Tibshirani

Der Ansatz von Hastie und Tibshirani [HT97] versucht die Kullback-Leibler-Distanz $l(p)$ zwischen $\widehat{\Pr}(c_i|D, c_{ij})$ und $\overline{\Pr}(c_i|D, c_{ij})$ zu minimieren:

$$\begin{aligned} l(p) &= \sum_{i \neq j} t_{c_{ij}} \widehat{\Pr}(c_i|D, c_{ij}) \cdot \lg \frac{\widehat{\Pr}(c_i|D, c_{ij})}{\overline{\Pr}(c_i|D, c_{ij})} \\ &= \sum_{i<j} t_{c_{ij}} \cdot \left(\widehat{\Pr}(c_i|D, c_{ij}) \cdot \lg \frac{\widehat{\Pr}(c_i|D, c_{ij})}{\overline{\Pr}(c_i|D, c_{ij})} + \widehat{\Pr}(c_j|D, c_{ij}) \cdot \lg \frac{\widehat{\Pr}(c_i|D, c_{ij})}{\overline{\Pr}(c_i|D, c_{ij})} \right), \end{aligned}$$

wobei $\overline{\Pr}(c_i|D, c_{ij}) = \frac{\widehat{\Pr}(c_i|D)}{\widehat{\Pr}(c_i|D)+\widehat{\Pr}(c_j|D)}$, $\widehat{\Pr}(c_j|D, c_{ij}) = 1 - \widehat{\Pr}(c_i|D, c_{ij})$ und $t_{c_{ij}} = t_{c_i} + t_{c_j}$ gilt.

Um $l(p)$ zu minimieren, berechnet [HT97] zuerst:

$$\frac{\partial l(p)}{\partial \widehat{\Pr}(c_i|D)} = \sum_{j \neq i} t_{c_{ij}} \cdot \left(-\frac{\widehat{\Pr}(c_i|D, c_{ij})}{\widehat{\Pr}(c_i|D)} + \frac{1}{\widehat{\Pr}(c_i|D) + \widehat{\Pr}(c_j|D)} \right) \tag{4.9}$$

Es genügt also, daß $\frac{\partial l(p)}{\partial \widehat{\Pr}(c_i|D)} = 0$ gilt. Setzen wir dies in (4.9), erhalten wir den folgen-

den Ansatz:

$$
\begin{aligned}
0 = \frac{\partial l(p)}{\partial \widehat{\Pr}\,(c_i|D)} &= \sum_{j \neq i} t_{c_{ij}} \cdot \left(-\frac{\widehat{\Pr}(c_i|D, c_{ij})}{\widehat{\Pr}\,(c_i|D)} + \frac{1}{\widehat{\Pr}\,(c_i|D) + \widehat{\Pr}\,(c_j|D)} \right) \\
\Leftrightarrow \quad \sum_{j \neq i} t_{c_{ij}} \cdot \frac{\widehat{\Pr}(c_i|D, c_{ij})}{\widehat{\Pr}\,(c_i|D)} &= \sum_{j \neq i} t_{c_{ij}} \cdot \frac{1}{\widehat{\Pr}\,(c_i|D) + \widehat{\Pr}\,(c_j|D)} \\
\Leftrightarrow \quad \sum_{j \neq i} t_{c_{ij}} \widehat{\Pr}(c_i|D, c_{ij}) &= \sum_{j \neq i} t_{c_{ij}} \cdot \frac{\widehat{\Pr}\,(c_i|D)}{\widehat{\Pr}\,(c_i|D) + \widehat{\Pr}\,(c_j|D)} \\
\Leftrightarrow \quad \sum_{j \neq i} t_{c_{ij}} \widehat{\Pr}(c_i|D, c_{ij}) &= \sum_{j \neq i} t_{c_{ij}} \overline{\Pr}(c_i|D, c_{ij})
\end{aligned}
$$

Zusammenfassend schlägt [HT97] vor, die Wahrscheinlichkeiten $\Pr(c_i|D)$ zu finden, die folgende Bedingungen erfüllen.

$$
\begin{aligned}
&\sum_{j \neq i} t_{c_{ij}} \widehat{\Pr}(c_i|D, c_{ij}) = \sum_{j \neq i} t_{c_{ij}} \overline{\Pr}(c_i|D, c_{ij}) \\
&\sum_{i=1}^{m} \widehat{\Pr}(c_i|D) = 1 \\
&\widehat{\Pr}(c_i|D) > 0, i = 1, ..., m
\end{aligned}
$$

Mit dem Algorithmus 4.1 können wir Wahrscheinlichkeiten $\Pr(c_i|D)$ bestimmen, die diese Bedingungen erfüllen. Ein Beweis für Korrektheit dieses Algorithmus kann in [HT97] nachgelesen werden.

5. Paarweiser Naive Bayes Klassifizierer

In diesem Kapitel wollen wir einen Naive Bayes Klassifizierer mit Methoden der Klassenbinarisierung kombinieren und gegebenfalls modifizieren. Der erste Abschnitt befaßt sich mit den im vorherigen Kapitel besprochenen Methoden der ungeordneten und der paarweisen Klassenbinarisierung. Dabei untersuchen wir, inwiefern diese auf einen Naive Bayes Klassifizierer anwendbar sind und ob diese zu einer Verbesserung der Genauigkeit führen können. Im zweiten Abschnitt stellen wir einen alternative Ansatz zur paarweisen Berechnung eines Naive Bayes Klassifizierer vor, der bei der Anwendung dieses Ansatz gegebenenfalls modifiziert wird. Die hieraus resultierenden Methoden werden wir auch auf ihre Anwendbarkeit und Genauigkeitsverbesserung untersuchen.

5.1. Klassenbinarisierungen mit einem Naive Bayes Klassifizierer

In diesem Abschnitt werden wir uns mit den Methoden der ungeordneten und paarweisen Klassenbinarisierung mit einem Naive Bayes Klassifizierer befassen. Wir werden dabei zwei wichtige Feststellungen machen. Zum einen ist die ungeordnete Klassenbinarisierung nicht äquivalent zu einem regulären Naive Bayes Klassifizierer. Zum anderen sind die Round Robin Klassenbinarisierung und der reguläre Naive Bayes Klassifizierer äquivalent zueinander. Bei unseren Erläuterungen werden wir nicht immer explizit erwähnen, daß wir bei den Klassenbinarisierungen einen Naive Bayes Klassifizierer als Basisklassifizierer verwenden.

5.1.1. Ungeordnete Klassenbinarisierung

Wir möchten in diesem Unterabschnitt auf die Anwendung und die Besonderheiten einer ungeordneten Klassenbinarisierung mit einem Naive Bayes Klassifizierer als Basisklassifizierer eingehen. Dabei werden wir wider Erwarten sehen, daß dieses Verfahren nicht äquivalent zu einem regulären Naive Bayes Klassifizierer ist, obwohl die beiden Verfahren eine ähnliche Struktur aufweisen.

Betrachten wir nun aber zuerst einmal seine Anwendung. Wir teilen für die ungeordnete Klassenbinarisierung das m-Klassenproblem in m binäre Probleme auf, bei denen jeweils die Beispiele einer Klasse c_i als positive Beispiele und die Beispiele der restlichen Klassen als negative Beispiele behandelt werden. Diese restlichen Klassen werden

folglich zu einer Klasse zusammengefaßt, die wir als Gegenklasse $\overline{c_i}$ bezeichnen. Da die Berechnung des Naive Bayes Klassifizierers auch bei den binären Problemen nur von den absoluten Häufigkeiten der Klassen c_i und $\overline{c_i}$ und der Attributwerte unter Auftreten dieser Klassen abhängt, möchten wir untersuchen, ob und wie fern sich diese Häufigkeiten für die einzelnen binären Probleme verändern.

Innerhalb jedes dieser Probleme $< c_i, \overline{c_i} >$ bleiben die absoluten Häufigkeiten der Attributwerte unter der Klasse c_i und die absolute Häufigkeit dieser Klasse unverändert. Die Wahrscheinlichkeitsabschätzungen von $\Pr(D|c_i)$ und $\Pr(c_i)$ sind demnach für den regulären Naive Bayes Klassifizierer und die ungeordnete Klassenbinarisierung mit einem Naive Bayes Klassifizierer gleich.

Bei der Zusammenfassung der restlichen Klassen müssen wir jedoch die absoluten Häufigkeiten der Attributwerte der Gegenklasse und die absolute Häufigkeit der Gegenklasse neu bestimmen. Diese Häufigkeiten lassen sich aus den absoluten Häufigkeiten der Klassen, die wir als Gegenklasse zusammengefaßt haben, berechnen.

Für die absolute Häufigkeit $t^a_{\overline{c_i}}$ eines Attributwertes a des Attributes A unter der Klasse c_i gilt:

$$t^a_{\overline{c_i}} = \sum_{j \neq i} t^a_{c_j} \tag{5.1}$$

Entsprechend berechnet sich die absolute Häufigkeit der Gegenklasse $t_{\overline{c_i}}$:

$$t_{\overline{c_i}} = \sum_{j \neq i} t_{c_j} \tag{5.2}$$

Befassen wir uns nun mit den Wahrscheinlichkeitsabschätzungen der ungeordneten Klassenbinarisierung, die durch den Index *UK* gekennzeichnet sind. Die Wahrscheinlichkeitsabschätzungen des regulären Naive Bayes Klassifizierer erhalten den Index *NB*. Sollte eine Wahrscheinlichkeitsabschätzung für beide Verfahren gleich sein, verzichten wir gänzlich auf einen Index.

Mit den oben bestimmten Häufigkeiten erhalten wir

$$\widehat{\Pr}(a_k|\overline{c_i})_{UK} = \frac{t^{a_k}_{\overline{c_i}} + 1}{t_{\overline{c_i}} + v_k} = \frac{\left(\sum_{j \neq i} t^{a_k}_{c_j}\right) + 1}{\left(\sum_{j \neq i} t_{c_j}\right) + v_k} \tag{5.3}$$

und

$$\widehat{\Pr}(\overline{c_i}) = \frac{t_{\overline{c_i}}}{t} = \frac{\sum_{j \neq i} t_{c_j}}{t} = \sum_{j \neq i} \widehat{\Pr}(c_j) = 1 - \widehat{\Pr}(c_i). \tag{5.4}$$

Nun können wir $\widehat{\Pr}(D|\overline{c_i})$ wie folgt berechnen:

$$\widehat{\Pr}(D|\overline{c_i})_{UK} = \prod_{k=1}^{n} \widehat{\Pr}(a_k|\overline{c_i})_{UK} = \prod_{k=1}^{n} \frac{\left(\sum_{j \neq i} t^{a_k}_{c_j}\right) + 1}{\left(\sum_{j \neq i} t_{c_j}\right) + v_k} \tag{5.5}$$

Es ist leicht einzusehen, daß für die Abschätzung $\widehat{\Pr}(D|\overline{c_i})_{UK}$ der ungeordneten Klassenbinarisierung und die Abschätzungen $\widehat{\Pr}(c_j)_{NB}$ des Naive Bayes Klassifizierer folgendes gilt:

$$\widehat{\Pr}(D|\overline{c_i})_{UK} \cdot \widehat{\Pr}(\overline{c_i}) \neq \sum_{j \neq i} \widehat{\Pr}(D|c_j)_{NB} \cdot \widehat{\Pr}(c_j) \tag{5.6}$$

Demnach gilt für die Abschätzungen $\widehat{\Pr}(c_i|D)$ des Naive Bayes Klassifizierers und der ungeordneten Klassenbinarisierung folgendes:

$$\begin{aligned} \widehat{\Pr}(c_i|D)_{UK} &= \frac{\widehat{\Pr}(D|c_i) \cdot \widehat{\Pr}(c_i)}{\widehat{\Pr}(D|c_i) \cdot \widehat{\Pr}(c_i) + \widehat{\Pr}(D|\overline{c_i})_{UK} \cdot \widehat{\Pr}(\overline{c_i})} \\ &\neq \frac{\widehat{\Pr}(D|c_i) \cdot \widehat{\Pr}(c_i)}{\widehat{\Pr}(D|c_i) \cdot \widehat{\Pr}(c_i) + \sum_{j \neq i} \widehat{\Pr}(D|c_j)_{NB} \cdot \widehat{\Pr}(c_j)} \\ &= \widehat{\Pr}(c_i|D)_{NB} \end{aligned} \tag{5.7}$$

Folglich berechnen die ungeordnete Klassenbinarisierung und der Naive Bayes Klassifizierer unterschiedliche Wahrscheinlichkeiten. Wir werden an einem Beispiel zeigen, daß dies auch zu unterschiedlichen Vorhersagen führen kann.

Beispiel 5.1 Wir verwenden für dieses Beispiel den Datensatz aus Abschnitt 3.3 und ein neues Testbeispiel D.

$$D = (Sonnig, Kalt, Hoch, Schwach)$$

Die für das Beispiel benötigten Wahrscheinlichkeitsabschätzungen für die einzelnen Attributwerte und Klassen haben wir in Tabelle 5.1 zusammengefaßt. Die Abschätzungen der Gegenklassen wurden mit (5.3) und (5.4) ermittelt. Alle anderen Abschätzungen wurden mit den Formeln aus Abschnitt 3.2 berechnet.

Wir bestimmen zuerst das Produkt von $\widehat{\Pr}(D|c)$ und $\widehat{\Pr}(c)$ für alle Klassen c.

$$\begin{aligned} \widehat{\Pr}(D|\,Golf) \cdot \widehat{\Pr}(\,Golf) &= \frac{5}{9} \cdot \frac{1}{4} \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{2}{5} = \frac{1}{72} \approx 0,013889 \\ \widehat{\Pr}(D|Squash) \cdot \widehat{\Pr}(Squash) &= \frac{2}{7} \cdot \frac{2}{3} \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{4}{15} = \frac{4}{315} \approx 0,012698 \\ \widehat{\Pr}(D|\,Tennis) \cdot \widehat{\Pr}(\,Tennis) &= \frac{1}{4} \cdot \frac{5}{7} \cdot \frac{3}{7} \cdot \frac{4}{7}\frac{1}{3} = \frac{5}{343} \approx 0,014577 \end{aligned}$$

Mit diesen geschätzten Wahrscheinlichkeiten und den Abschätzungen für die Wahrscheinlichkeiten der Klassen können wir $\widehat{\Pr}(D)_{NB}$ berechnen.

$$\begin{aligned} \widehat{\Pr}(D)_{NB} &= \widehat{\Pr}(D|\,Golf) \cdot \widehat{\Pr}(\,Golf) + \widehat{\Pr}(D|Squash) \cdot \widehat{\Pr}(Squash) \\ &\quad + \widehat{\Pr}(D|\,Tennis) \cdot \widehat{\Pr}(\,Tennis) \\ &= \frac{1}{72} + \frac{4}{315} + \frac{5}{343} \\ &\approx 0,041165 \end{aligned}$$

	Sonnig	Kalt	Hoch	Schwach	Klasse
Golf	5/9	1/4	1/2	1/2	2/5
Squash	2/7	2/3	1/2	1/2	4/15
Tennis	1/4	5/7	3/7	4/7	1/3
$\overline{\text{Golf}}$	1/4	8/11	5/11	6/11	3/5
$\overline{\text{Squash}}$	3/7	6/13	6/13	7/13	11/15
$\overline{\text{Tennis}}$	6/13	5/12	1/2	1/2	2/3

Tabelle 5.1.: Wahrscheinlichkeitsabschätzungen für das zweite Trainingsbeispiel

Damit erhalten wir die folgenden Abschätzungen für die Wahrscheinlichkeiten der Klassen:

$$\widehat{\Pr}(\mathit{Golf}|D)_{NB} = \frac{\widehat{\Pr}(D|\mathit{Golf}) \cdot \widehat{\Pr}(\mathit{Golf})}{\widehat{\Pr}(D)_{NB}} \approx 0,3374$$

$$\widehat{\Pr}(\mathit{Squash}|D)_{NB} = \frac{\widehat{\Pr}(D|\mathit{Squash}) \cdot \widehat{\Pr}(\mathit{Squash})}{\widehat{\Pr}(D)_{NB}} \approx 0,3085$$

$$\widehat{\Pr}(\mathit{Tennis}|D)_{NB} = \frac{\widehat{\Pr}(D|\mathit{Tennis}) \cdot \widehat{\Pr}(\mathit{Tennis})}{\widehat{\Pr}(D)_{NB}} \approx 0,3541$$

Der reguläre Naive Bayes Klassifizierer ordnet das Beispiel D der Klasse *Tennis* zu. Betrachten wir nun die ungeordnete Klassenbinarisierung. Hierfür berechnen wir zuerst das Produkt von $\widehat{\Pr}(D|\overline{c})$ und $\widehat{\Pr}(\overline{c})$ für alle Klassen c.

$$\widehat{\Pr}(D|\overline{\mathit{Golf}})_{UK} \cdot \widehat{\Pr}(\overline{\mathit{Golf}})_{UK} = \frac{1}{4} \cdot \frac{8}{11} \cdot \frac{5}{11} \cdot \frac{6}{11} \cdot \frac{3}{5} \approx 0,027047$$

$$\widehat{\Pr}(D|\overline{\mathit{Squash}})_{UK} \cdot \widehat{\Pr}(\overline{\mathit{Squash}})_{UK} = \frac{3}{7} \cdot \frac{6}{13} \cdot \frac{6}{13} \cdot \frac{7}{13} \cdot \frac{11}{15} \approx 0,036049$$

$$\widehat{\Pr}(D|\overline{\mathit{Tennis}})_{UK} \cdot \widehat{\Pr}(\overline{\mathit{Tennis}})_{UK} = \frac{6}{13} \cdot \frac{5}{12} \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{2}{3} \approx 0,032051$$

Da für jedes binäres Problem $< c, \overline{c} >$ die Wahrscheinlichkeit $\Pr(D)$ einen anderen geschätzten Wert erhält, werden wir nur die Abschätzung für das Problem $< \mathit{Golf}, \overline{\mathit{Golf}} >$ exemplarisch bestimmen. Die Berechnung für die anderen Probleme erfolgt analog.

$$\begin{aligned} \widehat{\Pr}(D)_{UK} &= \widehat{\Pr}(D|\mathit{Golf}) \cdot \widehat{\Pr}(\mathit{Golf}) + \widehat{\Pr}(D|\overline{\mathit{Golf}})_{UK} \cdot \widehat{\Pr}(\overline{\mathit{Golf}}) \\ &\approx 0,013889 + 0,027047 = 0,040936 \end{aligned}$$

Wenn wir $\widehat{\Pr}(D)_{UK}$ für alle binären Probleme berechnet haben, können wir die Wahr-

scheinlichkeiten $\Pr(c|D)$ abschätzen.

$$\begin{aligned}
\widehat{\Pr}(Golf|D)_{UK} &= \frac{\widehat{\Pr}(D|Golf)\cdot\widehat{\Pr}(Golf)}{\widehat{\Pr}(D|Golf)\cdot\widehat{\Pr}(Golf)+\widehat{\Pr}(D|\overline{Golf})_{UK}\cdot\widehat{\Pr}(\overline{Golf})} \\
&\approx \frac{0,013889}{0,013889+0,027047} \approx 0,3393 \\
\widehat{\Pr}(Squash|D)_{UK} &= \frac{\widehat{\Pr}(D|Squash)\cdot\widehat{\Pr}(Squash)}{\widehat{\Pr}(D|Squash)\cdot\widehat{\Pr}(Squash)+\widehat{\Pr}(D|\overline{Squash})_{UK}\cdot\widehat{\Pr}(\overline{Squash})} \\
&\approx \frac{0,012698}{0,012698+0,036049} \approx 0,2605 \\
\widehat{\Pr}(Tennis|D)_{UK} &= \frac{\widehat{\Pr}(D|Tennis)\cdot\widehat{\Pr}(Tennis)}{\widehat{\Pr}(D|Tennis)\cdot\widehat{\Pr}(Tennis)+\widehat{\Pr}(D|\overline{Tennis})_{UK}\cdot\widehat{\Pr}(\overline{Tennis})} \\
&\approx \frac{0,032051}{0,032051+0,032051} \approx 0,3126
\end{aligned}$$

Die ungeordnete Klassenbinarisierung ordnet das Beispiel D der Klasse *Golf* zu.

Wie man sieht, unterscheiden sich die Vorhersagen der ungeordneten Klassenbinarisierung und des Naive Bayes Klassifizierers. Wir möchten noch darauf hinweisen, daß das Ergebnis der ungeordneten Klassenbinarisierung keine Wahrscheinlichkeitsverteilung ist, da die Methode die drei Wahrscheinlichkeiten unabhängig voneinander schätzt und die Klasse mit der größten Wahrscheinlichkeit vorhersagt.

Bevor wir uns nun der paarweisen Klassenbinarisierung zuwenden, möchten wir noch anmerken, daß die Feststellung über die Unterschiede zwischen der ungeordneten Klassenbinarisierung und des Naive Bayes Klassifizierers entsprechend auch für die geordnete Klassenbinarisierung gelten. Das heißt, der Naive Bayes Klassifizierer und die geordnete Klassenbinarisierung berechnen unterschiedliche Wahrscheinlichkeiten und gegebenenfalls auch unterschiedliche Vorhersagen, da die geordnete Klassenbinarisierung im Grunde genommen nur eine Variante der ungeordneten Klassenbinarisierung ist.

5.1.2. Round Robin Klassenbinarisierung

Dieser Unterabschnitt befaßt sich mit der Round Robin Klassenbinarisierung mit einem Naive Bayes Klassifizierer. Unser Ziel ist es zu zeigen, daß einerseits die Abstimmungsmethoden Voting und Weighed Voting für dieses Verfahren äquivalent sind und daß dieses Verfahren einem regulären Naive Bayes Klassifizierer entspricht. Hierfür erläutern wir zuerst die Anwendung dieser Klassenbinarisierung und die Parallelen zwischen den beiden Verfahren. Diese werden wir in einigen Lemmata zusammenfassen, um danach die Äquivalenzen der genannten Methoden und Verfahren zu beweisen. Anschließend betrachten wir die Round Robin Klassenbinarisierung mit den in Abschnitt 4.4.2 vorgestellten Bradley-Terry-Methoden. Wir werden sehen, daß die hieraus resultierenden Verfahren auch äquivalent zum regulären Naive Bayes Klassifizierer sind.

Betrachten wir nun die Round Robin Klassenbinarisierung mit einem Naive Bayes Klassifizierer. Wir teilen hierfür das Multiklassenproblem mit m Klassen in $\frac{m(m-1)}{2}$ binäre Lernprobleme auf. Diese Lernprobleme bezeichnen wir als Klassenpaare und notieren sie durch c_{ij} für zwei Klassen c_i und c_j mit $c_i \neq c_j$. Für jedes dieser Klassenpaare trainieren wir auf den Trainingsbeispielen der beiden Klassen des jeweiligen Klassenpaares einen eigenen regulären Naive Bayes Klassifizierer. Jeder Klassifizierer berechnet für ein Klassenpaar die Wahrscheinlichkeit der beiden Klassen des Paares. Diese Wahrscheinlichkeiten, daß die Klasse c_i bzw. c_j unter der Beobachtung der Daten und des Klassenpaares c_{ij} eintritt, bezeichnen wir als $\Pr(c_i|D, c_{ij})$ beziehungsweise $\Pr(c_j|D, c_{ij})$. Wir können diese Wahrscheinlichkeit wie folgt berechnen:

$$\begin{aligned}\Pr(c_i|D, c_{ij}) &= \frac{\Pr(c_i \cap c_{ij} \cap D)}{\Pr(D \cap c_{ij})} = \frac{\Pr(c_i \cap D)}{\Pr(c_{ij}|D)\Pr(D)} = \frac{\Pr(c_i|D) \cdot \Pr(D)}{\Pr(c_{ij}|D) \cdot \Pr(D)} \\ &= \frac{\Pr(c_i|D)}{\Pr(c_{ij}|D)}, \end{aligned} \tag{5.8}$$

wobei c_{ij} das Zufallsereignis $c_i \cup c_j$ bezeichnet.

Mit Hilfe von

$$\Pr(c_i|D, c_{ij}) + \Pr(c_j|D, c_{ij}) = 1. \tag{5.9}$$

können wir (5.8) noch weiter vereinfachen und erhalten folgende Berechnungsformel für die Wahrscheinlichkeit $\Pr(c_i|D, c_{ij})$:

$$\begin{aligned}\Pr(c_i|D, c_{ij}) &\overset{(5.9)}{=} \frac{\Pr(c_i|D, c_{ij})}{\Pr(c_i|D, c_{ij}) + \Pr(c_j|D, c_{ij})} \\ &\overset{(5.8)}{=} \frac{\frac{\Pr(c_i|D)}{\Pr(c_{ij}|D)}}{\frac{\Pr(c_i|D)}{\Pr(c_{ij}|D)} + \frac{\Pr(c_j|D)}{\Pr(c_{ij}|D)}} \\ &= \frac{\Pr(c_i|D)}{\Pr(c_i|D) + \Pr(c_j|D)} \\ &\overset{\text{(Satz von Bayes)}}{=} \frac{\Pr(D|c_i)\Pr(c_i)}{\Pr(D|c_i)\Pr(c_i) + \Pr(D|c_j)\Pr(c_j)} \end{aligned} \tag{5.10}$$

Erinnern wir uns noch einmal, wie man die Wahrscheinlichkeit $\Pr(c_i|D)$ berechnet.

$$\Pr(c_i|D) = \frac{\Pr(D|c_i)\Pr(c_i)}{\Pr(D)} = \frac{\Pr(D|c_i)\Pr(c_i)}{\sum_{c_j}\Pr(D|c_j)\Pr(c_j)} \tag{5.11}$$

Man sieht an den Berechnungsformeln (5.10) und (5.11), daß sich die beiden Wahrscheinlichkeiten $\Pr(c_i|D)$ und $\Pr(c_i|D, c_{ij})$ nur durch ihre Normalisierungsfaktoren unterscheiden. Aus diesem Grund stehen diese Wahrscheinlichkeiten in einer Beziehung, die wir im folgenden untersuchen und festhalten wollen.

Lemma 5.2 *Für die beliebigen unterschiedichen Klassen c_i und c_j gilt:*

$$\frac{\Pr(c_i|D, c_{ij})}{\Pr(c_j|D, c_{ij})} = \frac{\Pr(c_i|D)}{\Pr(c_j|D)}.$$

Beweis.

$$\frac{\Pr(c_i|D,c_{ij})}{\Pr(c_j|D,c_{ij})} \overset{(5.8)}{=} \frac{\frac{\Pr(c_i|D)}{\Pr(c_{ij}|D)}}{\frac{\Pr(c_j|D)}{\Pr(c_{ij}|D)}} = \frac{\Pr(c_i|D)}{\Pr(c_j|D)}$$

□

Aus diesem Verhältnis können wir einige transitive Beziehungen zwischen den Klassenpaaren folgern, die wir später für die Äquivalenzbeweise der verschiedenen Round Robin Klassenbinarisierungen benötigen.

Lemma 5.3 *Für die beliebigen unterschiedichen Klassen c_i, c_j und c_k gilt:*

(a) Die folgenden Ungleichungen sind äquivalent:

$$\begin{aligned} & \Pr(c_i|D) < \Pr(c_j|D) && (1)\\ \Leftrightarrow\quad & \Pr(c_i|D,c_{ij}) < \Pr(c_j|D,c_{ij}) && (2)\\ \Leftrightarrow\quad & \Pr(c_i|D,c_{ik}) < \Pr(c_j|D,c_{jk}) && (3)\\ \Leftrightarrow\quad & \Pr(c_k|D,c_{ik}) > \Pr(c_k|D,c_{jk}) && (4) \end{aligned}$$

(b) $\dfrac{\Pr(c_i|D,c_{ij})}{\Pr(c_j|D,c_{ij})} = \dfrac{\Pr(c_i|D,c_{ik})}{\Pr(c_k|D,c_{ik})} \cdot \dfrac{\Pr(c_k|D,c_{jk})}{\Pr(c_j|D,c_{jk})}$

(c) $\Pr(c_i|D,c_{ik}) < \Pr(c_k|D,c_{ik}) \wedge \Pr(c_k|D,c_{jk}) < \Pr(c_j|D,c_{jk})$
$\Rightarrow \Pr(c_i|D,c_{ij}) < \Pr(c_j|D,c_{ij})$

Beweis.

(a) (1)⇔(2)

$$\begin{aligned} \Pr(c_i|D) < \Pr(c_j|D) \quad &\Leftrightarrow \quad \frac{\Pr(c_i|D)}{\Pr(c_i|D)+\Pr(c_j|D)} < \frac{\Pr(c_j|D)}{\Pr(c_i|D)+\Pr(c_j|D)}\\ &\Leftrightarrow \quad \Pr(c_i|D,c_{ij}) < \Pr(c_j|D,c_{ij}) \end{aligned}$$

(1)⇔(4)

$$\begin{aligned} \Pr(c_i|D) < \Pr(c_j|D) \quad &\Leftrightarrow \quad \Pr(c_i|D)+\Pr(c_k|D) < \Pr(c_j|D)+\Pr(c_k|D)\\ &\Leftrightarrow \quad \frac{\Pr(c_k|D)}{\Pr(c_i|D)+\Pr(c_k|D)} > \frac{\Pr(c_k|D)}{\Pr(c_j|D)+\Pr(c_k|D)}\\ &\Leftrightarrow \quad \Pr(c_k|D,c_{ik}) > \Pr(c_k|D,c_{jk}) \end{aligned}$$

(3)⇔(4)

$$\begin{aligned} \Pr(c_i|D,c_{ik}) < \Pr(c_j|D,c_{jk}) \quad &\overset{(5.9)}{\Leftrightarrow} \quad 1-\Pr(c_k|D,c_{ik}) < 1-\Pr(c_k|D,c_{jk})\\ &\Leftrightarrow \quad \Pr(c_k|D,c_{ik}) > \Pr(c_k|D,c_{jk}) \end{aligned}$$

(b)

$$\begin{aligned}
\frac{\Pr(c_i|D,c_{ij})}{\Pr(c_j|D,c_{ij})} &\overset{\text{Lemma 5.2}}{=} \frac{\Pr(c_i|D)}{\Pr(c_j|D)} \\
&= \frac{\Pr(c_i|D)}{\Pr(c_k|D)} \cdot \frac{\Pr(c_k|D)}{\Pr(c_j|D)} \\
&\overset{\text{Lemma 5.2}}{=} \frac{\Pr(c_i|D,c_{ik})}{\Pr(c_k|D,c_{ik})} \cdot \frac{\Pr(c_k|D,c_{jk})}{\Pr(c_j|D,c_{jk})}
\end{aligned}$$

(c)

$$\begin{aligned}
& \Pr(c_i|D,c_{ik}) < \Pr(c_k|D,c_{ik}) \wedge \Pr(c_k|D,c_{jk}) < \Pr(c_j|D,c_{jk}) \\
\overset{(a)}{\Leftrightarrow}\ & \Pr(c_i|D) < \Pr(c_k|D) \wedge \Pr(c_k|D) < \Pr(c_j|D) \\
\Rightarrow\ & \Pr(c_i|D) < \Pr(c_j|D) \\
\overset{(a)}{\Leftrightarrow}\ & \Pr(c_i|D,c_{ij}) < \Pr(c_j|D,c_{ij})
\end{aligned}$$

□

Diese transitiven Beziehungen gelten entsprechend auch bei Gleichheit der Wahrscheinlichkeiten eines Klassenpaares.

Korollar 5.4 *Für die beliebigen unterschiedichen Klassen c_i, c_j und c_k gilt:*

(a) Die folgenden Gleichungen sind äquivalent:

$$\begin{aligned}
& \Pr(c_i|D) = \Pr(c_j|D) && (1)\\
\Leftrightarrow\ & \Pr(c_i|D,c_{ij}) = \Pr(c_j|D,c_{ij}) && (2)\\
\Leftrightarrow\ & \Pr(c_i|D,c_{ik}) = \Pr(c_j|D,c_{jk}) && (3)\\
\Leftrightarrow\ & \Pr(c_k|D,c_{ik}) = \Pr(c_k|D,c_{jk}) && (4)
\end{aligned}$$

(b) $\Pr(c_i|D,c_{ik}) = \Pr(c_k|D,c_{ik}) \wedge \Pr(c_k|D,c_{jk}) = \Pr(c_j|D,c_{jk})$
$\Rightarrow \Pr(c_i|D,c_{ij}) = \Pr(c_j|D,c_{ij})$

Beweis. Folgt direkt aus Lemma 5.3. □

Wir haben nun alle Hilfsmittel zusammen, um später die Äquivalenz der Round Robin Klassenbinarisierung mit den Abstimmungsmethoden Voting und Weighted Voting zu einem regulären Naive Bayes Klassifizierer zeigen zu können.

Betrachten wir zunächst noch einmal das Grundgerüst einer Round Robin Klassenbinarisierung, bevor wir auf die verschiedenen Möglichkeiten der Abstimmung eingehen:

$$\arg\max_{c_i} \sum_{j \neq i} vote(c_i, c_j), \tag{5.12}$$

wobei *vote* diejenige Funktion ist, die abhängig von der Abstimmungsmethode entscheidet, wie die Klasse c_i unter dem Klassenpaar c_{ij} bewertet wird. Wenn wir nun die gängigsten Methoden, auf die wir im vorangegangen Kapitel näher eingegangen sind, auf einen Naive Bayes Klassifizierer anwenden, erhalten wir folgende Varianten für die Abstimmungsfunktion *vote*:

- Voting:

$$vote_V(c_i, c_j) = \begin{cases} 1, & \textbf{falls } \Pr(c_i|D, c_{ij}) \geq \Pr(c_j|D, c_{ij}) \\ 0, & \textbf{sonst} \end{cases}$$

- Weighted Voting:

$$vote_{WV}(c_i, c_j) = \Pr(c_i|D, c_{ij})$$

Da für die Wahrscheinlichkeiten $\Pr(c_i|D, c_{ij})$ und $\Pr(c_j|D, c_{ij})$ (5.9) gilt, können wir $vote_V$ folgendermaßen umformen, um eine der Funktion $vote_{WV}$ ähnlichere Form zu erhalten.

$$\begin{aligned} vote_V(c_i, c_j) &= \begin{cases} 1, & \text{falls } \Pr(c_i|D, c_{ij}) \geq \Pr(c_j|D, c_{ij}) \\ 0, & \text{sonst} \end{cases} \\ &= [\Pr(c_i|D, c_{ij})] \end{aligned} \tag{5.13}$$

Die Gauß-Klammern [.] haben hier und im Verlauf dieser Arbeit folgende Bedeutung;

$$[.] : [0, 1] \to \{0, 1\} \text{ mit } [x] = \begin{cases} 1, & \text{falls } x \geq 0,5 \\ 0, & \text{sonst} \end{cases}$$

Das heißt, die Wahrscheinlichkeiten werden auf 1 gerundet, falls die Wahrscheinlichkeiten größer oder gleich 0,5 sind. Ansonsten werden sie auf 0 gerundet.

Vergleichen wir jetzt die beiden Abstimmungsfunktionen miteinander und mit dem regulären Naive Bayes Klassifizierers, liegen zwei Vermutungen nahe. Erstens sind die beiden Abstimmungsmethoden bezüglich der Abstimmung äquivalent. Zweitens ist die Round Robin Klassenbinarisierung mit diesen Abstimmungsmethoden äquivalent zu einem regulären Naive Bayes Klassifizierer. Bevor wir aber diese Äquivalenzen zeigen, benötigen wir noch ein paar kleinere Erweiterungen von Lemma 5.3 und Korollar 5.4, die dann auch die Berechnung von $vote_V$ abdecken.

Lemma 5.5 *Für die beliebigen unterschiedichen Klassen c_i und c_j gilt:*

(a) $\Pr(c_i|D, c_{ij}) < \Pr(c_j|D, c_{ij}) \Leftrightarrow [\Pr(c_i|D, c_{ij})] < [\Pr(c_j|D, c_{ij})]$

(b) $\Pr(c_i|D, c_{ij}) < \Pr(c_j|D, c_{ij}) \Rightarrow [\Pr(c_i|D, c_{ik})] < [\Pr(c_j|D, c_{jk})]$

Beweis.

(a)

$$\begin{aligned} & \Pr(c_i|D,c_{ij}) < \Pr(c_j|D,c_{ij}) \\ \overset{(5.9)}{\Leftrightarrow} \quad & \Pr(c_i|D,c_{ij}) < \frac{1}{2} < \Pr(c_j|D,c_{ij}) \\ \Leftrightarrow \quad & [\Pr(c_i|D,c_{ij})] < [\Pr(c_j|D,c_{ij})] \end{aligned}$$

(b)

$$\begin{aligned} & \Pr(c_i|D,c_{ij}) < \Pr(c_j|D,c_{ij}) \\ \overset{\text{Lemma 5.3}}{\Leftrightarrow} \quad & \Pr(c_i|D,c_{ik}) < \Pr(c_j|D,c_{jk}) \\ \Rightarrow \quad & [\Pr(c_i|D,c_{ik})] < [\Pr(c_j|D,c_{jk})] \end{aligned}$$

□

Die direkte Konsequenz aus diesem Lemma wollen wir durch die nachfolgende Folgerung festhalten.

Korollar 5.6 *Für die beliebigen unterschiedichen Klassen c_i und c_j gilt:*

(a) $\Pr(c_i|D,c_{ij}) = \Pr(c_j|D,c_{ij}) \Leftrightarrow [\Pr(c_i|D,c_{ij})] = [\Pr(c_j|D,c_{ij})]$

(b) $\Pr(c_i|D,c_{ij}) = \Pr(c_j|D,c_{ij}) \Leftrightarrow [\Pr(c_i|D,c_{ik})] = [\Pr(c_j|D,c_{jk})]$

Wir haben jetzt alles zusammen, um eine Aussage über die Beziehung zwischen den Rankings der Round Robin Klassenbinarisierung und des regulären Naive Bayes Klassifizierers treffen zu können.

Lemma 5.7 *Für die beliebigen unterschiedichen Klassen c_i und c_j gilt*

(a) $\Pr(c_i|D) \leq \Pr(c_j|D) \Leftrightarrow \sum_{k\neq i} \Pr(c_i|D,c_{ik}) \leq \sum_{k\neq j} \Pr(c_j|D,c_{jk})$

(b) $\Pr(c_i|D) \leq \Pr(c_j|D) \Leftrightarrow \sum_{k\neq i} [\Pr(c_i|D,c_{ik})] \leq \sum_{k\neq j} [\Pr(c_j|D,c_{jk})]$

Beweis.

(a) „⇒“ Für alle Klassen c_k mit $c_i \neq c_k \neq c_j$ gilt einerseits wegen Lemma 5.3

$$\Pr(c_i|D) < \Pr(c_j|D) \Leftrightarrow \Pr(c_i|D,c_{ik}) < \Pr(c_j|D,c_{jk})$$

und andererseits wegen Korollar 5.4

$$\Pr(c_i|D) = \Pr(c_j|D) \Rightarrow \Pr(c_i|D,c_{ik}) = \Pr(c_j|D,c_{jk}).$$

Zusammenfassend erhalten wir dann folgendes:

$$\Pr(c_i|D) \leq \Pr(c_j|D) \Rightarrow \sum_{k\neq i} \Pr(c_i|D,c_{ik}) \leq \sum_{k\neq j} \Pr(c_j|D,c_{jk})$$

„⇐“ Analog zum Beweis der Gegenrichtung gilt wegen Lemma 5.3:

$$\Pr(c_i|D, c_{ij}) > \Pr(c_j|D, c_{ij}) \Rightarrow \sum_{k\neq i} \Pr(c_i|D, c_{ik}) > \sum_{k\neq j} \Pr(c_j|D, c_{jk})$$

Mit Hilfe des Kontrapositionsgesetzes erhalten wir folgendes:

$$\begin{aligned} & \sum_{k\neq i} \Pr(c_i|D, c_{ik}) \leq \sum_{k\neq j} \Pr(c_j|D, c_{jk}) \\ \Leftrightarrow \quad & \neg \left(\sum_{k\neq i} \Pr(c_i|D, c_{ik}) > \sum_{k\neq j} \Pr(c_j|D, c_{jk}) \right) \\ \Rightarrow \quad & \neg \left(\Pr(c_i|D, c_{ij}) > \Pr(c_j|D, c_{ij}) \right) \\ \Leftrightarrow \quad & \Pr(c_i|D, c_{ij}) \leq \Pr(c_j|D, c_{ij}) \end{aligned}$$

(b) Mit Hilfe von Lemma 5.5 und Korollar 5.6 erfolgt der Beweis analog zu (a).

□

Das Ranking der Round Robin Klassenbinarisierung mit den Methoden Voting oder Weighted Voting und das Ranking des regulären Naive Bayes Klassifizierer sind nach Lemma 5.7 äquivalent. Daraus folgt auch die Gleichheit der Klassifikation dieser drei Verfahren.

Satz 5.8 *Round Robin Klassenbinarisierungen mit einem Naive Bayes Klassifizierer als Basisklassifizierer und den Abstimmungsmethoden Voting und Weighted Voting erzeugen das gleiche Ranking und die gleiche Klassifikation wie ein regulärer Naive Bayes Klassifizierer, das heißt:*

$$\begin{aligned} & \arg\max_{c_i} \Pr(c_i|D) \\ = \quad & \arg\max_{c_i} \sum_{j\neq} \Pr(c_i|D, c_{ij}) \\ = \quad & \arg\max_{c_i} \sum_{j\neq} [\Pr(c_i|D, c_{ij})] \end{aligned}$$

Beweis. Wegen Lemma 5.7 sind die Rankings der drei Verfahren äquivalent zueinander. Aus der Äquivalenz der Rankings folgt auch, daß diese Verfahren die gleiche Klassifikation generieren. □

Nach Satz 5.8 besteht also eine theoretische Äquivalenz zwischen den drei oben genannten Verfahren. Wir werden sehen, daß diese Äquivalenz auch in der Praxis vorliegt und Unterschiede nur auf die Implementierung der Verfahren zurückzuführen sind. Die Äquivalenzbeweise basieren alle auf der Annahme, daß die Wahrscheinlichkeiten $\Pr(D|c_i)$, $\Pr(c_i)$ und damit auch $\Pr(c_i|D)$ für alle Klassen und Klassenpaare gleich sind. Es genügt zu zeigen, daß die Wahrscheinlichkeitsabschätzungen $\widehat{\Pr}(D|c_i)$ und $\widehat{\Pr}(c_i)$ für

alle Klassen und Klassenpaare des regulären Naive Bayes Klassifizierers und der Round Robin Klassenbinarisierung mit einem Naive Bayes Klassifizierer als Basisklassifizierer den gleichen Wert haben.

Die absoluten Häufigkeiten innerhalb der Klassen (unter anderem t_{c_i} und $t_{c_j}^{a_i}$ für alle Attributwerte a_i des Testbeispiels), die wir zur Berechnung dieser Abschätzungen benötigen, werden durch die Klassenbinarisierung beziehungsweise Aufteilung in Klassenpaare nicht verändert. Die Berechnung der oben erwähnten Wahrscheinlichkeitsabschätzungen für eine Klasse des Klassenpaares ist nur von den absoluten Häufigkeiten dieser Klasse abhängig, deshalb haben die besagten geschätzten Wahrscheinlichkeiten für alle Klassenpaare der Round Robin Klassenbinarisierung den gleichen Wert wie die des Naive Bayes Klassifizierers. Es gilt also für die geschätzten Wahrscheinlichkeiten $\widehat{\Pr}(c_i|D)$ des regulären Naive Bayes Klassifizierers und $\widehat{\Pr}(c_i|D, c_{ij})$:

$$\widehat{\Pr}(c_i|D, c_{ij}) = \frac{\widehat{\Pr}(c_i|D)}{\widehat{\Pr}(c_i|D) + \widehat{\Pr}(c_j|D)} \tag{5.14}$$

Wie bereits erwähnt können abhängig von der Implementierung der Round Robin Klassenbinarisierung Unterschiede zum regulären Naive Bayes Klassifizierer auftreten. Wir wollen exemplarisch eine Ursache kurz erklären. Angenommen der Datensatz beinhaltet kontinuierliche Attribute, dann haben wir zusätzlich zur Wahl der Behandlung dieser Attribute (Normalmethode, Kernelmethode und Diskretisierung), die alleine noch keinen Unterschied verursacht, noch eine weitere Entscheidung zu treffen, die wir am Beispiel der Diskretisierung erläutern wollen. Beim Training der Klassifizierer der Klassenpaare gibt es zwei mögliche Zeitpunkte für die Diskretisierung der kontinuierlichen Attribute, vor und nach der Aufteilung der Daten in Klassenpaare. Offensichtlich führen die beiden Zeitpunkte zu unterschiedlichen, diskreten Datensätzen und damit auch möglicherweise zu unterschiedlichen Klassifikationen, da die globale Transitivität des Naive Bayes Klassifizierers dadurch aufgehoben wird. Demnach ist die Round Robin Klassenbinarisierung mit einem Naive Bayes Klassifizierer in der Praxis nicht äquivalent zu einem regulären Naive Bayes Klassifizierer, falls wir die Diskretisierung nicht auf den Trainingsdaten sondern auf den Daten der Klassenpaare anwenden. Bei der Normal- und Kernelmethode existiert ein ähnliches Problem, da diese bei der realen Anwendung auch zuerst einige Werte (zum Beispiel ihre Genauigkeit) auf den Trainingsdaten beziehungsweise den Daten der Klassenpaare berechnen. Wir werden bei unseren Experimenten testen, welcher der beiden Zeitpunkte für die Diskretisierung besser ist. Hierfür genügt der Vergleich des Naive Bayes Klassifizierers mit der Round Robin Klassenbinarisierung, bei der die Diskretisierung innerhalb der Klassenpaare vorgenommen wird, da die Diskretisierung vor der Aufteilung der Daten die gleichen Ergebnisse liefert wie der reguläre Naive Bayes Klassifizierer.

Befassen wir uns nun mit den Bradley-Terry-Methoden. Die Berechnung der Round Robin Klassenbinarisierung mit diesen Methoden erfolgt, wie wir es bereits für die Abstimmungsmethoden gezeigt haben. Der Unterschied besteht nur in der Dekodierung der abgeschätzten Wahrscheinlichkeiten.

Erinnern wir uns nochmal an die Grundidee dieser Methoden. Wir versuchen mit Hilfe der abgeschätzten Wahrscheinlichkeiten $\widehat{\Pr}(c_i|D, c_{ij})$ der Round Robin Klassenbinarisierung die globalen Wahrscheinlichkeitsabschätzungen $\widehat{\Pr}(c_i|D)$ so zu bestimmen, daß die Differenz zwischen der geschätzten Wahrscheinlichkeit $\widehat{\Pr}(c_i|D, c_{ij})$ und der mit Hilfe des Bradley-Terry-Modells berechneten $\overline{\Pr}(c_i|D, c_{ij})$ für alle Klassenpaare c_{ij} minimiert wird. Betrachten wir noch einmal (5.14), sehen wir, daß diese beiden Wahrscheinlichkeitsabschätzungen gleich sind, wenn wir die letztere mit einem Naive Bayes Klassifizierer bestimmen. Das heißt, es gilt für alle Klassenpaare c_{ij}:

$$\widehat{\Pr}(c_i|D, c_{ij}) = \overline{\Pr}(c_i|D, c_{ij}) \tag{5.15}$$

Die von dem Naive Bayes Klassifizierer abgeschätzten Wahrscheinlichkeiten sind für alle Bradley-Terry-Methoden eine Lösung mit minimaler Differenz zwischen $\widehat{\Pr}(c_i|D, c_{ij})$ und $\overline{\Pr}(c_i|D, c_{ij})$. Da es aber nur genau eine minimale Lösung für die Dekodierung geben kann, berechnen die Bradley-Terry-Methoden exakt die gleichen Wahrscheinlichkeitsabschätzungen wie ein regulärer Naive Bayes Klassifizierer. Dies führt zu folgendem Satz.

Satz 5.9 *Round Robin Klassenbinarisierungen mit einem Naive Bayes Klassifizierer als Basisklassifizierer und den im vorangegangen Kapitel vorgestellten Bradley-Terry-Methoden erzeugen die gleiche Klassifikation wie ein regulärer Naive Bayes Klassifizierer.*

Beweis. Die Bradley-Terry-Methoden versuchen die Differenz zwischen $\widehat{\Pr}(c_i|D, c_{ij})$ und $\overline{\Pr}(c_i|D, c_{ij})$ zu minimieren. Da für die Wahrscheinlichkeitsabschätzungen $\widehat{\Pr}(c_i|D)$ eines Naive Bayes Klassifizierers (5.15) gilt, sind diese Abschätzungen eine minimale Lösung für alle Bradley-Terry-Methoden.

Bei den Methoden von Hastie und Tibshirani und Price, Knerr, Personnaz und Dreyfus existiert nach Konstruktion nur eine Lösung, deshalb sind die Lösungen der beiden Methoden und die des Naive Bayes Klassifizierers gleich.

Bei der Methode von Refregier und Vallet muß eine Lösung alle ausgewählten Gleichungen erfüllen. Jede mögliche Gleichung der Form (4.4) wird von den Wahrscheinlichkeitsabschätzungen des Naive Bayes Klassifizierers gelöst, deshalb sind diese Abschätzungen für jede beliebige Auswahl dieser Gleichungen die Lösung, die die minimalste Differenz zwischen $\widehat{\Pr}(c_i|D, c_{ij})$ und $\overline{\Pr}(c_i|D, c_{ij})$ hat. □

Im Laufe unserer Experimente lassen wir dennoch die Round Robin Klassenbinarisierungen mit einem Naive Bayes Klassifizierer und den Abstimmungs- und Bradley-Terry-Methoden nicht außer acht. Wir lösen die Transitivität des Naive Bayes Klassifizierers auf, indem wir auf die Naive Bayes Klassifizierer der Klassenpaare die Ensemble-Methoden Bagging und Boosting anwenden. Wir erhoffen uns dadurch andere Ergebnisse als mit einem Naive Bayes Klassifizierer, den wir regulär berechnet oder nur mit diesen Ensemble-Methoden kombiniert haben, zu erzielen.

Bevor wir nun diesen Abschnitt beenden, möchten wir die eben festgestellten Äquivalenzen anhand eines Beispieles verdeutlichen. Dabei werden werden wir uns auf

die Abstimmungsmethoden und die Methode von Price, Knerr, Personnaz und Dreyfus beschränken.

Beispiel 5.10 Wir verwenden wieder den Datensatz und das Beispiel aus Abschnitt 3.3. Uns ist bekannt, daß der Naive Bayes Klassifizierer das Beispiel der Klasse *Squash* zuordnet. Die Wahrscheinlichkeitsabschätzungen $\widehat{\Pr}(D|c)$, $\widehat{\Pr}(c)$ und deren Produkt sind uns bereits bekannt.

$$\widehat{\Pr}(D|Golf) = \frac{1}{48} \qquad \widehat{\Pr}(Golf) = \frac{2}{5}$$
$$\widehat{\Pr}(D|Squash) = \frac{1}{28} \qquad \widehat{\Pr}(Squash) = \frac{4}{15}$$
$$\widehat{\Pr}(D|Tennis) = \frac{4}{343} \qquad \widehat{\Pr}(Tennis) = \frac{1}{3}$$

$$\widehat{\Pr}(Golf|D) \approx 0,383240209$$
$$\widehat{\Pr}(Squash|D) \approx 0,43798885$$
$$\widehat{\Pr}(Tennis|D) \approx 0,178770941$$

Mit diesen Werten werden wir nun die paarweisen Wahrscheinlichkeiten $\Pr(c_i|D, c_{ij})$ abschätzen. Die folgenden Berechnungen werden wir nicht weiter erläutern.

$$\widehat{\Pr}(Golf|D, Golf \cap Squash) \approx 0,466666644$$
$$\widehat{\Pr}(Squash|D, Golf \cap Squash) \approx 0,533333356$$

$$\widehat{\Pr}(Golf|D, Golf \cap Tennis) \approx 0,681908551$$
$$\widehat{\Pr}(Tennis|D, Golf \cap Tennis) \approx 0,318091449$$

$$\widehat{\Pr}(Squash|D, Squash \cap Tennis) \approx 0,710144948$$
$$\widehat{\Pr}(Tennis|D, Squash \cap Tennis) \approx 0,289855052$$

Betrachten wir jetzt die Abstimmungsmethode Voting. *Golf* erhält 1 Stimme, *Squash* zwei Stimmen und *Tennis* keine. Demnach sagt auch die Round Robin Klassenbinarisierung mit Voting die Klasse *Squash* voraus.

Für die Abstimmungsmethode Weighted Voting müssen wir noch die paarweisen Wahrscheinlichkeitsabschätzungen addieren.

$$\widehat{\Pr}(Golf|D, Golf \cap Squash) + \widehat{\Pr}(Golf|D, Golf \cap Tennis) = 1,148575195$$
$$\widehat{\Pr}(Squash|D, Golf \cap Squash) + \widehat{\Pr}(Squash|D, Squash \cap Tennis) = 1,243478304$$
$$\widehat{\Pr}(Tennis|D, Golf \cap Tennis) + \widehat{\Pr}(Tennis|D, Squash \cap Tennis) = 0,607946501$$

Die Abstimmungsmethode Weighted Voting sagt auch die Klasse *Squash* voraus.

Befassen wir uns nun mit der Bradley-Terry-Methode von Price, Knerr, Personnaz und Dreyfus. Wir werden bei ihrer Berechnung nur die Formel für die Klasse *Golf* angeben.

$$\begin{aligned}
\widehat{\Pr}(Golf|D)_{PKPD} &= \frac{1}{\frac{1}{\widehat{\Pr}(Golf|D,Golf\cap Squash)} + \frac{1}{\widehat{\Pr}(Golf|D,Golf\cap Tennis)} - 1} \\
&\approx \frac{1}{\frac{1}{0{,}466666644} + \frac{1}{0{,}681908551} - 1} \\
&= \frac{1}{2,609329544} \approx 0,383240209 \\
&\approx \widehat{\Pr}(Golf|D)_{NB}
\end{aligned}$$

$$\begin{aligned}
\widehat{\Pr}(Squash|D)_{PKPD} &\approx \frac{1}{\frac{1}{0{,}533333356} + \frac{1}{0{,}710144948} - 1} \\
&= \frac{1}{2,283163146} \approx 0,43798885 \\
&\approx \widehat{\Pr}(Squash|D)_{NB}
\end{aligned}$$

$$\begin{aligned}
\widehat{\Pr}(Tennis|D)_{PKPD} &\approx \frac{1}{\frac{1}{0{,}318091449} + \frac{1}{0{,}289855052} - 1} \\
&= \frac{1}{5,593750265} \approx 0,178770941 \\
&\approx \widehat{\Pr}(Tennis|D)_{NB}
\end{aligned}$$

Wie man sieht, berechnet die Methode von Price, Knerr, Personnaz und Dreyfus nicht nur die gleiche Vorhersage, sondern auch die gleichen Wahrscheinlichkeiten wie ein regulärer Naive Bayes Klassizierer.

5.2. Alternative paarweise Methoden

Im vorherigen Abschnitt haben wir gesehen, daß die Round Robin Klassenbinarisierung die Performanz eines Naive Bayes Klassifizierers nicht verbessert. Aus diesem Grund haben wir uns alternative Methoden zur paarweisen Klassifizierung überlegt, die auf dem gleichen wahrscheinlichkeitstheoretischen Ansatz basieren. Bei diesen Methoden versuchen wir die gesuchten Wahrscheinlichkeiten $\Pr(c_i|D)$ durch die paarweisen Wahrscheinlichkeiten $\Pr(D|c_{ij})$ abzuschätzen. Für die Berechnung der paarweisen Wahrscheinlichkeiten haben wir zwei Möglichkeiten. Wir werden zuerst auf diese eingehen, bevor wir uns in den folgenden Unterabschnitten mit den einzelnen Methoden befassen werden.

Bei der ersten Möglichkeit zur Berechnung der Wahrscheinlichkeit $\Pr(D|c_{ij})$ verwenden wir die Wahrscheinlichkeiten $\Pr(D|c_i)$, die wir durch einen regulären Naive Bayes

Klassifizierer berechnen. Wenn wir diese Möglichkeit verwenden, kennzeichnen wir es mit dem Index R für regulär.

$$\begin{aligned}
\Pr(D|c_{ij})_R &= \frac{\Pr(c_{ij}|D)\Pr(D)}{\Pr(c_{ij})} \\
&= \frac{(\Pr(c_i|D) + \Pr(c_j|D))\Pr(D)}{\Pr(c_{ij})} \\
&= \frac{\Pr(D|c_i)\Pr(c_i) + \Pr(D|c_j)\Pr(c_j)}{\Pr(c_{ij})} \\
&= \Pr(D|c_i)\Pr(c_i|c_{ij}) + \Pr(D|c_j)\Pr(c_j|c_{ij}) \qquad (5.16)
\end{aligned}$$

Die zweite Möglichkeit berechnen wir, indem wir nicht nur die Häufigkeiten für Klassen sondern auch für Klassenpaare bestimmen. Wir kennzeichnen diese Berechnungsmöglichkeit mit dem Index P für paarweise.

$$\Pr(D|c_{ij})_P = \prod_{k=1}^{n} \Pr(a_k|c_{ij}) \qquad (5.17)$$

Ein Unterschied zwischen den beiden Methoden ist, daß wir bei der zweiten eine andere Unabhängigkeitsannahme verwenden, bei der wir die Unabhängigkeit von Klassenpaaren annehmen. Ein anderer Unterschied ist die erhöhte Trainingszeit der zweiten Methode. Wir betrachten bei ihr jedes Trainingsbeispiel der Klasse c_i nicht einmal sondern einmal pro Klassenpaar c_{ij}. Der Lernaufwand erhöht sich also im Gegensatz zum regulären Naive Bayes Klassifizierer beziehungsweise zur ersten Methode um den Faktor $m-1$. Ihre Trainingszeit hat somit die Laufzeitkomplexität $O(tnm)$, während die Trainingszeit der ersten Methode nur eine Laufzeitkomplexität von $O(tn)$ hat.

5.2.1. Wahrscheinlichkeitstheoretischer Ansatz

Die alternativen Klassifizierungsmethoden, die wir in diesem Abschnitt vorstellen, beruhen auf folgendem wahrscheinlichkeitstheoretischen Ansatz.

Es gilt

$$\begin{aligned}
(m-1)\Pr(c_i|D) &= \sum_{j\neq i} \Pr(c_i|D) = \sum_{j\neq i} \frac{\Pr(c_i \cap D)}{\Pr(D)} = \sum_{j\neq i} \frac{\Pr(c_i \cap c_{ij} \cap D)}{\Pr(D)} \\
&= \sum_{j\neq i} \frac{\Pr(c_i|c_{ij} \cap D) \cdot \Pr(c_{ij}|D) \cdot \Pr(D)}{\Pr(D)} \\
&= \sum_{j\neq i} \Pr(c_i|c_{ij} \cap D) \cdot \Pr(c_{ij}|D) \\
&= \sum_{j\neq i} \Pr(c_i|D, c_{ij}) \cdot \Pr(c_{ij}|D)
\end{aligned}$$

und damit

$$\Pr(c_i|D) = \frac{1}{m-1} \sum_{j\neq i} \Pr(c_i|D, c_{ij}) \cdot \Pr(c_{ij}|D). \qquad (5.18)$$

Mit diesem Ansatz erhalten wir dann den folgenden Klassifizierer:

$$\begin{aligned}\arg\max_{c_i} \Pr(c_i|D) &= \arg\max_{c_i} \frac{1}{m-1} \sum_{j \neq i} \Pr(c_i|D, c_{ij}) \cdot \Pr(c_{ij}|D) \\ &= \arg\max_{c_i} \sum_{j \neq i} \Pr(c_i|D, c_{ij}) \cdot \Pr(c_{ij}|D) \end{aligned} \tag{5.19}$$

Die beiden Terme des Klassifizierers benennen wir wie folgt:

$$v_{ij} = \Pr(c_i|D, c_{ij}) \tag{5.20}$$

und

$$w_{ij} = w_{ji} = \Pr(c_{ij}|D) \tag{5.21}$$

Diese Terme können wir folgendermaßen berechnen:

$$v_{ij} = \frac{\Pr(D|c_i) \cdot \Pr(c_i)}{\Pr(D|c_i) \cdot \Pr(c_i) + \Pr(D|c_j) \cdot \Pr(c_j)} \tag{5.22}$$

$$w_{ij} = \frac{\Pr(D|c_{ij}) \cdot \Pr(c_{ij})}{\Pr(D)} \tag{5.23}$$

Berechnen wir die w_{ij} mit einem regulären Naive Bayes Klassifizierer, erhalten wir wieder die gleiche Voraussage wie mit einem regulären Naive Bayes Klassifizierer, da folgendes gilt:

$$\begin{aligned} & \frac{1}{m-1} \sum_{j \neq i} \widehat{\Pr}(c_i|D, c_{ij}) \cdot \widehat{\Pr}(c_{ij}|D) \\ = & \frac{1}{m-1} \sum_{j \neq i} \frac{\widehat{\Pr}(D|c_i) \cdot \widehat{\Pr}(c_i)}{\widehat{\Pr}(D|c_i) \cdot \widehat{\Pr}(c_i) + \widehat{\Pr}(D|c_j) \cdot \widehat{\Pr}(c_j)} \cdot \frac{\widehat{\Pr}(D|c_{ij}) \cdot \widehat{\Pr}(c_{ij})}{\widehat{\Pr}(D)} \\ = & \frac{1}{m-1} \sum_{j \neq i} \frac{\widehat{\Pr}(D|c_i) \cdot \widehat{\Pr}(c_i)}{\widehat{\Pr}(D)} \cdot \frac{\widehat{\Pr}(D|c_i) \cdot \widehat{\Pr}(c_i) + \widehat{\Pr}(D|c_j) \cdot \widehat{\Pr}(c_j)}{\widehat{\Pr}(D|c_i) \cdot \widehat{\Pr}(c_i) + \widehat{\Pr}(D|c_j) \cdot \widehat{\Pr}(c_j)} \\ = & \frac{1}{m-1} \sum_{j \neq i} \widehat{\Pr}(c_i|D) \\ = & \widehat{\Pr}(c_i|D) \end{aligned}$$

Wir müssen also andere Möglichkeiten in Betracht ziehen, um diesen Klassifizierer anwenden zu können. Wir haben uns vier Möglichkeiten überlegt, von denen sich jeweils zwei nur durch die Art der Abstimmung unterscheiden. Bei dem ersten Paar verwenden wir die v_{ij} als Abstimmungsfunktion und die w_{ij} als Gewichte. Ein Klassifizierer dieses Paares entspricht somit dem Ansatz (5.18), falls wir w_{ij} regulär berechnen. Das zweite Paar verwendet anstatt $\Pr(c_i|D, c_{ij})$ die Wahrscheinlichkeiten $\Pr(c_i|c_{ij})$ zur Abstimmung und die w_{ij} als Gewichte.

5.2.2. PNB1 und PNB2

Bei den ersten beiden Methoden, die wir zur besseren Unterscheidung mit PNB1 und PNB2 bezeichnen, verwenden wir die v_{ij} zur Abstimmung. Bei PNB1 ordnen wir der Klasse, die bei der Abstimmung gewinnt, w_{ij} als gewichtete Stimme zu. PNB2 entspricht dem Ansatz (5.18), das heißt wir verwenden das Produkt von v_{ij} und w_{ij} als gewichtete Stimme. Wir haben bereits gesehen, daß PNB2 äquivalent zu einem regulären Naive Bayes Klassifizierer ist, falls wir w_{ij} regulär berechnen. Aus diesem Grund berechnen wir w_{ij} bei PNB2 nur paarweise.

Befassen wir uns nun mit dem Aufbau von PNB1. Wenden wir die nötigen Modifikationen auf (5.19) an, erhalten wir den folgenden Klassifizierer:

$$c_{\text{PNB1}} = \arg\max_{c_i} \sum_{\substack{j \neq i \\ v_{ij} \geq v_{ji}}} w_{ij} \tag{5.24}$$

$$= \arg\max_{c_i} \sum_{\substack{j \neq i \\ v_{ij} \geq v_{ji}}} \frac{\Pr(D|c_{ij}) \cdot \Pr(c_{ij})}{\Pr(D)}$$

$$= \arg\max_{c_i} \sum_{\substack{j \neq i \\ v_{ij} \geq v_{ji}}} \Pr(D|c_{ij}) \cdot \Pr(c_{ij}) \tag{5.25}$$

Wie wir bereits am Anfang dieses Abschnitts angesprochen haben, können wir die Wahrscheinlichkeit $\Pr(D|c_{ij})$ entweder regulär (5.16) oder paarweise (5.17) berechnen. Setzen wir die reguläre Berechnung ein, erhalten wir den folgenden Klassifizierer.

$$\begin{aligned}
c_{\text{PNB1}_{Reg}} &= \arg\max_{c_i} \sum_{\substack{j \neq i \\ v_{ij} \geq v_{ji}}} \Pr(D|c_{ij}) \Pr(c_{ij}) \\
&= \arg\max_{c_i} \sum_{\substack{j \neq i \\ v_{ij} \geq v_{ji}}} (\Pr(D|c_i) \Pr(c_i|c_{ij}) + \Pr(D|c_j) \Pr(c_j|c_{ij})) \Pr(c_{ij})) \\
&= \arg\max_{c_i} \sum_{\substack{j \neq i \\ v_{ij} \geq v_{ji}}} \Pr(D|c_i) \Pr(c_i \cap c_{ij}) + \Pr(D|c_j) \Pr(c_j \cap c_{ij}) \\
&= \arg\max_{c_i} \sum_{\substack{j \neq i \\ v_{ij} \geq v_{ji}}} (\Pr(D|c_i) \Pr(c_i) + \Pr(D|c_j) \Pr(c_j)) \\
&= \arg\max_{c_i} \sum_{\substack{j \neq i \\ v_{ij} \geq v_{ji}}} (\Pr(c_i|D) + \Pr(c_j|D))
\end{aligned}$$

Dieser Klassifizierer ähnelt stark einer Round Robin Klassenbinarisierung mit einer Kombination aus Voting und Weighted Voting. Wir vermuten deshalb, daß dieser Klassifizierer zu einem regulären Naive Bayes Klassifizierer äquivalent ist.

Satz 5.11 *Die Methode* PNB1_R *ist äquivalent zu einem regulären Naive Bayes Klassifizierer, das heißt:*

$$\arg\max_{c_i} \sum_{\substack{k \neq i \\ v_{ik} \geq v_{ki}}} (\Pr(c_i|D) + \Pr(c_k|D)) = \arg\max_{c_i} \Pr(c_i|D)$$

Beweis. Damit dies tatsächlich der Fall ist, müßte für zwei beliebige unterschiedliche Klassen c_i und c_j folgendes gelten:

$$\begin{aligned} \sum_{\substack{k \neq i \\ v_{ik} \geq v_{ki}}} (\Pr(c_i|D) + \Pr(c_k|D)) &\geq \sum_{\substack{k \neq j \\ v_{jk} \geq v_{kj}}} (\Pr(c_j|D) + \Pr(c_k|D)) \\ \Leftrightarrow \Pr(c_i|D) &\geq \Pr(c_j|D) \end{aligned} \tag{5.26}$$

Wir zeigen kurz die Gültigkeit dieser Äquivalenz.

Nach Lemma 5.3 ist $\Pr(c_i|D) \geq \Pr(c_j|D)$ äquivalent zu $v_{ij} \geq v_{ji}$, deswegen können wir auch $\Pr(c_i|D) \geq \Pr(c_j|D)$ zur Abstimmung verwenden.

„⇐“: Angenommen für zwei beliebige Klassen c_i und c_j gilt $\Pr(c_i|D) \geq \Pr(c_j|D)$. Sei c_k eine weitere beliebige Klasse mit $\Pr(c_j|D) \geq \Pr(c_k|D)$, dann gilt:

$$w_{ik} = \Pr(c_i|D) + \Pr(c_k|D) \geq \Pr(c_j|D) + \Pr(c_k|D) = w_{jk}.$$

Für alle c_k mit $\Pr(c_k|D) > \Pr(c_j|D)$ bekommt c_j keine Gewichte w_{jk} zugeteilt. Aus diesen Gründen ist die Summe für die Klasse c_i größer als die Summe der Klasse c_j, damit ist die folgende Ungleichung gültig:

$$\sum_{\substack{k \neq i \\ \Pr(c_i|D) \geq \Pr(c_k|D)}} (\Pr(c_i|D) + \Pr(c_k|D)) \geq \sum_{\substack{k \neq j \\ \Pr(c_j|D) \geq \Pr(c_k|D)}} (\Pr(c_j|D) + \Pr(c_k|D))$$

„⇒“: Angenommen für die Klassen c_i und c_j gilt:

$$\sum_{\substack{k \neq i \\ \Pr(c_i|D) \geq \Pr(c_k|D)}} (\Pr(c_i|D) + \Pr(c_k|D)) \geq \sum_{\substack{k \neq j \\ \Pr(c_j|D) \geq \Pr(c_k|D)}} (\Pr(c_j|D) + \Pr(c_k|D))$$

Würde jetzt auch $\Pr(c_i|D) < \Pr(c_j|D)$ gelten, müßte analog zum Beweis der Gegenrichtung folgendes gelten:

$$\sum_{\substack{k \neq i \\ \Pr(c_i|D) \geq \Pr(c_k|D)}} (\Pr(c_i|D) + \Pr(c_k|D)) < \sum_{\substack{k \neq j \\ \Pr(c_j|D) \geq \Pr(c_k|D)}} (\Pr(c_j|D) + \Pr(c_k|D))$$

Dies ist ein Widerspruch zur Voraussetzung, deshalb muß $\Pr(c_i|D) \geq \Pr(c_j|D)$ gelten.

Demnach ist die Äquivalenz (5.26) gültig. □

Der reguläre PNB1_R Klassifizierer ist folglich äquivalent zu einem regulären Naive Bayes Klassifizier, deshalb werden wir bei unseren Experimenten nur den paarweisen PNB1_P Klassifizierer betrachten.

5.2.3. PNB3 und PNB4

Bei den Methoden PNB3 und PNB4 verwenden wir anstatt den Wahrscheinlichkeiten $\Pr(c_i|D, c_{ij})$ die Wahrscheinlichkeiten $\Pr(c_i|c_{ij})$ zur Abstimmung. Diese erfolgt bei PNB3 ungewichtet und bei PNB4 gewichtet. Damit erhalten wir die folgenden Klassifizerer.

Nach Lemma 5.3 und Korollar 5.4 ist $v_{ij} \geq v_{ji}$ äquivalent zu $\Pr(c_i) \geq \Pr(c_j)$, deshalb können wir PNB3 folgendermaßen berechnen.

$$c_{\text{PNB3}} = \arg\max_{c_i} \sum_{\substack{j \neq i \\ \Pr(c_i) \geq \Pr(c_j)}} w_{ij} \tag{5.27}$$

PNB4 kann ohne weitere Transformationen wie folgt berechnet werden.

$$c_{\text{PNB4}} = \arg\max_{c_i} \sum_{j \neq i} \Pr(c_i|c_{ij}) \cdot w_{ij} \tag{5.28}$$

5.2.4. Überblick über die Verfahren

Abschließend möchten wir unsere alternativen Methoden und die Berechnung der benötigten Terme nochmals auf einen Blick zusammenfassen.

- PNB1:
$$c_{\text{PNB1}} = \arg\max_{c_i} \sum_{\substack{j \neq i \\ v_{ij} \geq v_{ji}}} w_{ij}$$

- PNB2:
$$c_{\text{PNB2}} = \arg\max_{c_i} \sum_{j \neq i} v_{ij} \cdot w_{ij}$$

- PNB3:
$$c_{\text{PNB3}} = \arg\max_{c_i} \sum_{\substack{j \neq i \\ \Pr(c_i) \geq \Pr(c_j)}} w_{ij}$$

- PNB4:
$$c_{\text{PNB4}} = \arg\max_{c_i} \sum_{j \neq i} \Pr(c_i|c_{ij}) \cdot w_{ij}$$

- Berechnung von v_{ij} und w_{ij}:
$$v_{ij} = \Pr(c_i|D, c_{ij})$$
$$v_{ji} = \Pr(c_j|D, c_{ij})$$
$$w_{ij} = w_{ji} = \frac{\Pr(D|c_{ij}) \cdot \Pr(c_{ij})}{\Pr(D)}$$

- Berechnung von $\Pr(D|c_{ij})$:
  - regulär:

$$\Pr(D|c_{ij})_R = \Pr(D|c_i)\Pr(c_i|c_{ij}) + \Pr(D|c_j)\Pr(c_j|c_{ij})$$

  - paarweise:

$$\Pr(D|c_{ij})_P = \prod_{i=1}^{n} \Pr(a|c_{ij})$$

6. Experimente

In diesem Kapitel werden wir die im vorherigen Kapitel vorgestellten Methoden – die ungeordnete und paarweise Klassenbinarisierung und unsere eigenen Methoden PNB1 bis PNB4 – mit einem regulären Naive Bayes Klassifizierer vergleichen. Wir beschreiben zunächst den Aufbau der Experimente und die verwendeten Testdaten, danach werden wir die Ergebnisse dieser Vergleiche präsentieren. Anschließend geben wir eine kurze Zusammenfassung der Erkenntnisse, die wir aus diesen Experimenten gewonnen haben.

6.1. Implementierung

Für unsere Experimente verwenden wir die frei erhältliche Lernumgebung WEKA (Waikato Environment for Knowledge Analysis) der Universität von Waikato, Neuseeland. Das Java-Package WEKA ermöglicht zum einen die Anwendung vieler bekannter Methoden des Maschinellen Lernens und zum anderen die Implementierung eigener Methoden.

Wir haben für unsere Experimente zu WEKA zwei neue Klassifizierer hinzugefügt. Bei dem ersten Klassifizierer *ExtendedMultiClassClassifier* handelt es sich um eine Modifikation des Metaklassifizierers *MultiClassClassifier*, der durch die Anwendung von binären Klassifizierern die Behandlung von Multiklassenproblemen ermöglicht. Er verfügt bereits über die ungeordnete und paarweise Klassenbinarisierung mit Voting und verschiedene ECOCs (Error Correcting Output Codes). Bis auf die Wahl der Methode und des Basisklassifizierers sind die Einstellmöglichkeiten des MultiClassClassifier für unsere Experimente irrelevant, da sich die übrigen Optionen nur auf die ECOCs beziehen.

Wir haben zu dem MultiClassClassifier die Abstimmungsmethode Weighted Voting und die Bradley-Terry-Methoden von Hastie und Tibshirani und von Price, Knerr, Personnaz und Dreyfus hinzugefügt. Zusätzlich haben wir den MultiClassClassifier um eine Einstellmöglichkeit der Diskretisierung erweitert. Bei dem unmodifizierten MultiClassClassifier erfolgt die Diskretisierung der Trainingsdaten immer innerhalb der binären Probleme, hingegen können wir bei unserem ExtendedMultiClassClassifier wählen, ob die Diskretisierung innerhalb der binären Probleme oder global auf den gesamten Trainingsdaten erfolgen soll.

Der zweite Klassifizierer *PairwiseNaiveBayes* ist eine Modifikation des NaiveBayes-Klassifizierers, die unsere paarweisen Methoden PNB1, PNB2, PNB3 und PNB4 implementiert. Bei diesen Methode kann zwischen der regulären und der paarweisen Berechnung von paarweisen Wahrscheinlichkeiten gewählt werden. Der Unterschied zwischen der regulären und der paarweisen Berechnung besteht darin, daß wir anstatt nur die absoluten Häufigkeiten der Klassen und der Attributwerte innerhalb dieser Klassen auch noch die absoluten Häufigkeiten entsprechend für Klassenpaare bestimmen und diese

```
@RELATION Wetter
@ATTRIBUTE Aussicht {Bewölkt, Regen, Sonnig}
@ATTRIBUTE Temperatur {Kalt, Warm}
@ATTRIBUTE Luftfeuchtigkeit {Hoch, Niedrig}
@ATTRIBUTE Windstärke {Schwach, Stark}
@ATTRIBUTE Klasse {Golf, Squash, Tennis}
@DATA
Bewölkt,Kalt,Hoch,Stark,Golf
Bewölkt,Kalt,Hoch,Stark,Squash
Sonnig,Warm,Niedrig,Schwach,Golf
Sonnig,Warm,Niedrig,Stark,Squash
Regen,Kalt,Niedrig,Schwach,Squash
Sonnig,Warm,Hoch,Stark,Golf
Bewölkt,Kalt,Niedrig,Stark,Tennis
Bewölkt,Kalt,Niedrig,Schwach,Tennis
Bewölkt,Warm,Hoch,Schwach,Golf
Bewölkt,Kalt,Hoch,Schwach,Tennis
Sonnig,Warm,Niedrig,Stark,Golf
Bewölkt,Kalt,Hoch,Stark,Tennis
Regen,Kalt,Hoch,Schwach,Squash
Sonnig,Warm,Niedrig,Schwach,Golf
Sonnig,Warm,Niedrig,Schwach,Tennis
```

Abbildung 6.1.: Unser Beispieldatensatz im ARFF-Dateiformat

zur Berechnung verwenden.

Zusätzlich zu unseren eigenen Lernverfahren verwenden wir noch die bereits in WEKA implementierten Methoden *AdaBoostM1*, *Bagging*, *NaiveBayes* und *Discretize*. Bei den beiden Ensemble-Methoden verwenden wir die Grundeinstellungen und als Basisklassifizierer den *NaiveBayes*-Klassifizierer. Bei diesem wählen wir die Diskretisierung *Discretize*, eine Implementierung von [FI93], zur Behandlung von kontinuierlichen Variablen aus. Die Diskretisierung erfolgt entweder auf den gesamten Trainingsdaten oder für jedes binäres Problem auf den Trainingsdaten des jeweiligen Problems. Die erste Möglichkeit bezeichnen wir als globale Diskretisierung (gekennzeichnet durch den Index G) und die letztere als binäre Diskretisierung (entsprechend gekennzeichnet durch den Index B).

6.2. Testdaten

Wir verwenden als Testdaten Datensätze des *UCI Repository*. Diese liegen bereits im benötigten *ARFF-Dateiformat* vor. Eine ARFF-Datei (Attribute-Relation File Format) ist eine ASCII-Datei, die einen Satz von Instanzen beschreibt, die eine Menge von Attri-

Datensatz	Attribute			Klassen	Instanzen
	Total	Nominal	Numerisch		
anneal	38	32	6	5	898
audiology	69	69	0	24	226
autos	25	12	13	6	205
balancescale	4	0	4	3	625
glass	9	0	9	6	214
hypothyroid	29	23	6	4	3772
iris	4	0	4	3	150
letter	16	0	16	26	20000
lymph	18	15	3	4	148
primary-tumor	17	17	0	21	339
segment	19	0	19	7	2310
soybean	35	35	0	19	683
splice	61	61	0	3	3190
vehicle	18	0	18	4	846
vowel	13	3	10	11	990
waveform-5000	40	0	40	3	5000
yeast	8	0	8	10	1484
zoo	17	16	1	7	101

Tabelle 6.1.: Statistik der Datensätze

buten gemeinsam haben. ARFF-Dateien wurden von dem Machine Learning Projekt am Fachbereich der Informatik der Universität von Waikato zur Verwendung mit WEKA entwickelt. Die Dateien bestehen aus zwei Teilen. Die Header-Informationen bilden den ersten Teil, dem der zweite Teil mit den Daten folgt. Der Header einer ARFF-Datei beinhaltet den Namen des Datensatzes, eine zeilenweise Liste der Attribute und deren Typen. Der Datenteil beinhaltet die einzelnen Instanzen. Jede Instanz wird durch eine Zeile repräsentiert. Das Ende einer Instanz wird durch ein Zeilenumbruch markiert. Die Attributwerte einer Instanz müssen mit Kommata separiert in der Reihenfolge aufgeführt werden, in der sie zeilenweise im Header deklariert worden sind. Fehlende Attributwerte werden durch ein Fragezeichen dargestellt. Das ARFF-Dateiformat erlaubt numerische und normale Attribute, Zeichenketten und Daten. Wir werden nicht näher auf den genauen Syntax des Formates eingehen. Zum besseren Verständnis haben wir jedoch unser Beispieldatensatz aus Abschnitt 3.3 im ARFF-Dateiformat in Abbildung 6.1 angegeben.

Ingesamt betrachten wir 18 Datensätze, die alle Multiklassenprobleme darstellen. Die Datensätze unterscheiden sich in der Anzahl ihrer Attributen, Klassen und Instanzen. Eine Statistik der Datensätze über diese Werte haben wir in Tabelle 6.2 zusammengefaßt. Bei den Attributen geben wir zusätzlich noch die Anzahl von numerischen und nominalen Attributen an.

	Fehlerraten der Verfahren in Prozent						
Datensatz	NB	$PNB1_P$	$PNB2_P$	$PNB3_P$	$PNB3_R$	$PNB4_P$	$PNB4_R$
anneal	4,00	4,17	3,91	3,51	3,49	3,44	3,42
audiology	26,95	26,72	26,89	31,13	27,79	29,13	27,80
autos	34,97	35,00	34,34	33,78	34,05	33,61	34,79
balancescale	27,93	27,93	27,97	35,59	38,25	27,82	27,88
glass	28,70	29,04	29,06	31,18	31,05	31,41	29,48
hypothyroid	1,74	1,70	1,84	3,19	4,77	3,11	2,86
iris	6,69	6,69	6,69	6,65	6,69	6,65	6,69
letter	25,95	26,80	27,20	47,38	31,27	28,99	25,95
lymph	14,83	14,83	14,70	19,06	17,76	16,12	17,73
primary-tumor	49,66	49,54	49,95	50,97	50,49	50,03	49,95
segment	8,93	9,27	9,36	14,63	8,93	14,63	8,93
soybean	7,14	7,86	8,05	15,05	11,54	10,39	7,28
splice	4,63	4,78	4,93	33,88	48,12	20,00	6,45
vehicle	39,32	39,33	39,04	57,68	57,83	41,65	39,11
vowel	41,27	41,51	41,44	42,77	41,27	42,77	41,27
waveform-5000	20,03	20,03	20,03	54,68	66,16	26,34	19,91
yeast	42,68	42,72	42,74	46,19	43,81	43,79	42,94
zoo	7,22	7,22	6,18	11,18	11,01	10,25	5,95
Mittel	21,81	21,95	21,91	29,92	29,68	24,45	22,13
Vergleich zu NB							
Win	–	4	6	3	2	4	6
Loss	–	10	11	15	13	14	9
Tie	–	4	1	0	3	0	3
Verfahren gleichwertig zu NB?							
G/HS/S		G	G	HS	HS	S	G

Tabelle 6.2.: Ergebnisse der ersten Testreihe: PNB1-PNB4

6.3. Aufbau

Unsere Experimente bestehen aus vier Testreihen, bei denen jeweils eine Gruppe von Klassifizierern mit dem regulären Naive Bayes Klassifizierer verglichen werden. Bei allen Testreihen werden die Daten entweder binär oder global diskretisiert. Wenn wir es nicht ausdrücklich erwähnen, erfolgt die Diskretisierung global. Wir verwenden als Performanzmaß die Fehlerraten, die wir durch eine stratifizierte 10x10-Kreuzvalidierung ermittelt haben. Die Ergebnisse der einzelnen Testreihen haben wir jeweils in einer Tabelle zusammengefaßt. Diese beinhalten die ermittelten Fehlerraten, die Anzahl der Datensätze, bei denen das jeweilige Verfahren verglichen mit dem Naive Bayes Klassifizierer besser (Win), schlechter (Loss) oder gleich (Tie) gut gewesen ist, und die Ergebnisse der Vorzeichentests, die wir anhand der Werte von Win und Loss ermittelt haben. Wir werten unsere Experimente anhand der Ergebnisse der Vorzeichentests aus. Die Ergebnisse des Vorzeichentest besagen, daß das Verfahren entweder gleichwertig (G) zu einem Naive

	Fehlerraten der Verfahren in Prozent						
Datensatz	NB	1vsAll$_B$	1vsAll$_G$	V	WV	HT	PKPD
anneal	4,00	2,40	2,28	3,10	3,07	3,31	3,09
audiology	26,95	27,75	27,75	26,95	26,95	26,82	26,95
autos	34,97	32,05	32,80	32,70	31,89	31,84	32,78
balancescale	27,93	26,87	27,97	26,20	26,20	26,20	26,20
glass	28,70	31,90	28,35	32,08	30,21	30,76	31,38
hypothyroid	1,74	2,25	2,13	1,73	1,68	1,77	1,69
iris	6,69	6,97	6,69	6,56	6,56	6,56	6,56
letter	25,95	27,38	26,10	26,48	26,31	26,28	26,37
lymph	14,83	14,72	14,44	14,80	14,97	14,72	14,73
primary-tumor	49,66	48,62	48,62	49,66	49,66	49,42	49,66
segment	8,93	9,07	10,88	8,54	8,51	22,25	8,53
soybean	7,14	7,63	7,63	7,14	7,14	7,13	7,14
splice	4,63	5,11	5,11	4,63	4,63	4,63	4,63
vehicle	39,32	37,79	38,96	37,66	37,49	38,26	37,55
vowel	41,27	35,17	41,03	36,15	33,41	33,34	34,16
waveform-5000	20,03	21,32	21,22	20,08	20,08	23,47	20,08
yeast	42,68	41,51	42,63	42,80	41,57	41,83	42,26
zoo	7,22	3,96	5,35	7,49	6,12	6,42	6,64
Mittel	21,81	21,25	21,66	21,37	20,91	21,94	21,13
Vergleich zu NB							
Win		9	9	9	10	12	11
Loss		9	8	5	4	6	3
Tie		0	1	4	4	0	4
Verfahren gleichwertig zu NB?							
G/HS/S		G	G	G	G	G	G

Tabelle 6.3.: Ergebnisse der zweiten Testreihe: Klassenbinarisierungen

Bayes Klassifizierer ist oder sich (höchst) signifikant (HS beziehungsweise S) von diesem unterscheidet.

Die erste Testreihe untersucht unsere eigenen Methoden PNB1 bis PNB4, die wir jeweils regulär und paarweise berechnen. Nur bei PNB1 und PNB2 verzichten wir auf eine reguläre Berechnung, da die daraus resultierenden Klassifizierer, wie wir bereits gesehen haben, äquivalent zu einem Naive Bayes Klassifizierer sind. Eine Zusammenfassung der Ergebnisse der ersten Testreihe befindet sich in Tabelle 6.2.

Bei der zweiten Testreihe vergleichen wir die ungeordnete und paarweise Klassenbinarisierung mit einem Naive Bayes Klassifizierer. Bei der Round Robin Klassenbinarisierung verwenden wir die Abstimmungsmethoden Voting und Weighted Voting und die Bradley-Terry-Methoden von Hastie und Tibshirani und von Price, Knerr, Personnaz und Dreyfus. Die Diskretisierung erfolgt bei allen Klassenbinarisierungen auf den binären Problemen. Nur bei der ungeordneten Klassenbinarisierung betrachten wir zusätzlich

	Fehlerraten der Verfahren in Prozent						
Datensatz	NB	Ada	1vsAll$_B$	V	WV	HT	PKPD
anneal	4,00	0,63	0,38	0,40	0,40	0,41	0,40
audiology	26,95	21,31	24,10	25,41	24,52	23,86	24,78
autos	34,97	31,03	23,54	20,93	21,72	21,63	20,76
balancescale	27,93	24,83	10,80	12,56	12,56	12,56	12,56
glass	28,70	28,70	32,36	27,11	26,99	26,84	26,54
hypothyroid	1,74	1,19	1,27	1,08	1,08	1,11	1,11
iris	6,69	6,26	5,56	6,14	6,14	6,14	6,14
letter	25,95	25,95	23,32	13,04	12,13	12,13	11,91
lymph	14,83	16,38	16,06	16,15	15,61	15,69	15,90
primary-tumor	49,66	49,66	58,23	54,80	53,99	55,36	55,54
segment	8,93	6,18	4,99	4,16	3,98	18,37	4,01
soybean	7,14	7,28	7,43	6,83	6,47	6,52	6,48
splice	4,63	6,29	5,25	5,84	5,73	5,78	5,72
vehicle	39,32	39,32	32,28	28,30	27,74	27,78	27,83
vowel	41,27	41,27	28,21	21,25	17,39	17,37	17,95
waveform-5000	20,03	19,85	15,85	18,15	18,03	18,04	18,05
yeast	7,22	7,17	8,17	7,02	5,65	4,85	6,17
zoo	42,68	42,68	45,28	43,17	42,87	43,50	43,28
Mittel	21,81	20,89	19,06	17,35	16,83	17,66	16,95
Vergleich zu NB							
Win		9	11	14	14	13	14
Loss		3	7	4	4	5	4
Tie		6	0	0	0	0	0
Verfahren gleichwertig zu NB?							
G/HS/S		G	G	S	S	G	S

Tabelle 6.4.: Ergebnisse der dritten Testreihe: AdaBoostM1 und Klassenbinarisierungen

noch die globale Diskretisierung. Wir verwenden die folgenden Abkürzungen bei dieser und den folgenden Testreihen. 1vsAll steht für die ungeordnete Klassenbinarisierung. Dabei stehen die Indizes B und G wie bereits erwähnt für die Diskretisierung auf dem binären Problem und die globale Diskretisierung. Bei den paarweisen Klassenbinarisierungen lassen wir den Index B wegfallen. Die Ergebnisse der zweiten Testreihe haben wir in Tabelle 6.3 zusammengefaßt.

Bei der dritten und der vierten Testreihe betrachten wir jeweils eine Ensemble-Methode, die ungeordnete und paarweise Klassenbinarisierung, auf deren binären Problemen diese Ensemble-Methoden angewendet werden. Wir untersuchen dieselben Dekodierungsmethoden wie bei der zweiten Testreihe. Die Diskretisierung erfolgt nur auf den binären Problemen. Die dritte Testreihe verwendet als Ensemble-Methode AdaBoostM1 und die vierte Bagging. Die Ergebnisse der dritten und der vierten Testreihe befinden sich in Tabelle 6.4 beziehungsweise Tabelle 6.5.

	Fehlerraten der Verfahren in Prozent						
Datensatz	NB	Bagging	1vsAll$_B$	V	WV	HT	PKPD
anneal	4,00	4,03	2,39	2,88	2,84	3,05	2,86
audiology	26,95	28,40	29,04	28,99	29,83	29,26	29,03
autos	34,97	31,37	31,68	30,72	31,12	31,12	31,02
balancescale	27,93	16,75	15,25	15,74	15,67	15,51	15,38
glass	28,70	28,43	30,96	30,37	29,09	29,23	29,30
hypothyroid	1,74	1,79	2,23	1,81	1,73	1,88	1,75
iris	6,69	5,95	5,53	5,76	5,76	5,76	5,76
letter	25,95	26,01	26,80	26,23	26,02	26,01	26,13
lymph	14,83	14,32	14,60	15,19	15,11	15,13	15,21
primary-tumor	49,66	50,10	49,31	51,66	50,81	50,50	50,91
segment	8,93	8,78	9,19	8,46	8,49	19,48	8,48
soybean	7,14	7,42	8,08	7,28	7,25	7,19	7,26
splice	4,63	4,59	5,07	4,66	4,67	4,66	4,66
vehicle	39,32	37,94	38,03	37,24	37,03	37,22	37,10
vowel	41,27	34,89	31,34	33,05	31,43	31,34	31,69
waveform-5000	20,03	19,72	21,05	19,90	19,90	22,54	19,90
yeast	42,68	40,85	40,75	41,10	40,59	40,59	40,78
zoo	7,22	7,06	4,89	7,65	7,08	7,83	7,13
Mittel	21,81	20,47	20,34	20,48	20,25	21,02	20,24
Vergleich zu NB							
Win		12	10	9	11	7	10
Loss		6	8	9	7	11	8
Tie		0	0	0	0	0	0
Verfahren gleichwertig zu NB?							
G/HS/S		G	G	G	G	G	G

Tabelle 6.5.: Ergebnisse der vierten Testreihe: Bagging und Klassenbinarisierungen

6.4. Auswertung

Bei der ersten Testreihe (siehe Tabelle 6.2) haben wir die Verfahren PNB1 bis PNB4 untersucht. Dabei haben wir festgestellt, daß diese Verfahren eine höhere Fehlerrate als der Naive Bayes Klassifizierer aufweisen. Während die Differenzen zur Fehlerrate des Naive Bayes Klassifizierers bei den Verfahren PNB1, PNB2 und dem regulär berechneten PNB4 noch relativ gering ausfallen, sind die Differenzen zur Fehlerrate des Naive Bayes Klassifizierers bei den Verfahren PNB3 und dem paarweise berechneten PNB4 schon gravierender. Vergleichen wir die Fehlerraten der regulär und der paarweise berechneten Verfahren, sehen wir, daß die paarweise Berechnung geringfügig schlechtere Ergebnisse als die reguläre Berechnung liefert. Hierbei sticht PNB4 heraus, da bei diesem Verfahren die Differenz zwischen den Fehlerraten der beiden Berechnungsmöglichkeiten deutlich höher ist.

Betrachten wir die jeweiligen Werte von Win und Loss und werten diese mit einem Vor-

zeichentest aus, kommen wir zu den folgenden Ergebnissen. Statistisch sind die Verfahren PNB1, PNB2 und der regulär berechnete PNB4 trotz ihrer leicht höheren Fehlerraten gleichwertig zu einem Naive Bayes Klassifizierer. Ihre Anwendung ist dennoch nicht ratsam, da sie einen erhöhten Aufwand darstellen. Die Verfahren PNB3 und der paarweise berechnete PNB4 sind höchst signifikant beziehungsweise signifikant schlechter als der Naive Bayes Klassifizierer. Vergleichen wir sowohl die Fehlerraten als auch die Ergebnisse der Vorzeichentests der regulär berechneten Verfahren mit denen der paarweise berechneten, kommen wir zu dem Schluß, daß die paarweise Berechnung geringfügig schlechter als die reguläre ist. Zusammenfassend ist die Anwendung der Verfahren PNB1 bis PNB4 demnach unabhängig von der gewählten Berechnung nicht gerechtfertigt.

Bei der zweiten Testreihe (siehe Tabelle 6.3) haben wir die ungeordnete und die paarweise Klassenbinarisierung untersucht. Die ungeordnete Klassenbinarisierung, bei der wir sowohl die binäre als auch die globale Diskretisierung getestet haben, hat eine geringfügig niedrigere Fehlerrate als ein Naive Bayes Klassifizierer aufgewiesen, wobei die Fehlerraten bei der binären Diskretisierung im Vergleich zur globalen noch niedriger gewesen sind. Die Fehlerraten der paarweise Klassenbinarisierung, bei der die binäre Diskretisierung angewandt wurde, sind ebenfalls niedriger gewesen als die des Naive Bayes Klassifizierers. Nur die paarweise Klassenbinarisierung, bei der die Dekodierungsmethode HT verwendet wurde, weist eine höhere Fehlerrate als ein Naive Bayes Klassifizierer auf.

Vergleichen wir die Fehlerraten der verwendeten Dekodierungsmethoden miteinander, sehen wir, daß sie sich nur geringfügig unterscheiden, wobei die Dekodierungsmethoden, die den geringsten Berechnungsaufwand haben (Voting, Weighted Voting und PKPD), besser abgeschnitten haben als die aufwendigste Methode HT. Die besten Ergebnisse wurden mit Weighted Voting erzielt.

Betrachten wir die Vorzeichentests, die mit den Werten von Win und Loss ausgewertet wurden, kommen wir zu zwei Schlußfolgerungen. Erstens ist die ungeordnete Klassenbinarisierung statistisch gleichwertig zu einem Naive Bayes Klassifizierer. Dies ist sowohl bei der binären als auch bei der globalen Diskretisierung der Fall. Die beiden Diskretisierungen erzeugen bei der ungeordneten Klassenbinarisierung nur minimale Unterschiede. Zweitens sind die paarweisen Klassenbinarisierungen, bei denen die Diskretisierung auf den Daten der Klassenpaare ausgeführt wurde, gleichwertig zu einem Naive Bayes Klassifizierer. Zusammenfassend weisen die getesteten Klassenbinarisierungen mit den verwendeten Dekodierungsmethoden niedrigere Fehlerraten als ein Naive Bayes Klassifizierer auf, obwohl sie statistisch gleichwertig zu diesem sind. Die Wahl der Diskretisierung und der Dekodierungsmethode hat nur geringe Auswirkungen auf die Ergebnisse.

Bei der dritten und vierten Testreihe haben wir Klassenbinarisierungen untersucht, auf deren binären Problemen die Ensemble-Methoden AdaBoostM1 und Bagging angewendet wurden. Wir betrachten hierfür auch diese Ensemble-Methoden ohne die Anwendung von Klassenbinarisierungen. Als Basisklassifizierer der Ensemble-Methoden wurde ein Naive Bayes Klassifizierer verwendet. Wir werden zuerst die Ergebnisse der dritten Testreihe (Tabelle 6.4) betrachten, bei der AdaboostM1 verwendet wurde, und danach auch die Resultate der vierten Testreihe (Tabelle 6.5) analysieren, bei der Bagging ein-

gesetzt wurde. Abschließen wollen wir dann die Auswertungen der beiden Testreihen zusammenfassen.

Schauen wir uns nun zuerst die Fehlerraten der dritten Testreihe (Tabelle 6.4) an. Dabei fällt uns auf, daß AdaBoostM1 und die Klassenbinarisierungen mit AdaBoostM1 niedrigere Fehlerraten als ein Naive Bayes Klassifizierer aufweisen. Interessant ist auch, daß die Fehlerraten der Klassenbinarisierungen auch niedriger als die von AdaBoostM1 ohne Klassenbinarisierung gewesen sind. Die paarweisen Klassenbinarisierungen haben bessere Ergebnisse als die ungeordnete Klassenbinarisierung erzielt. Die Fehlerraten der Dekodierungsmethoden unterscheiden sich nur geringfügig, wobei Weighted Voting wieder die besten Werte aufweist. Die schlechteste Dekodierungsmethode ist wiederum die Methode HT.

Betrachten wir nun die Resultate der Vorzeichentests der dritten Testreihe. AdaboostM1 und die ungeordnete Klassenbinarisierung, bei der die binäre Diskretisierung angewandt wurde, sind statistisch gleichwertig zu einem Naive Bayes Klassifizierer. Alle paarweisen Klassenbinarisierungen, bei denen die binäre Diskretisierung eingesetzt wurde, sind statistisch signifikant besser als ein regulärer Naive Bayes Klassifizierer. Nur die paarweise Klassenbinarisierung, bei der die Dekodierungsmethode HT verwendet wird, stellt eine Ausnahme dar, weil sie als einzige gleichwertig zu einem Naive Bayes Klassifizierer ist. Dieses Ergebnis kann aber auf das konservative Verhalten des Vorzeichentests zurückgeführt werden.

Wenden wir uns nun den Fehlerraten der vierten Testreihe (Tabelle 6.5) zu. Die Fehlerraten von Bagging, der ungeordneten und der paarweisen Klassenbinarisierung sind geringfügig niedriger als die eines Naive Bayes Klassifizierers. Die ungeordnete Klassenbinarisierung hat auch etwas besser als Bagging abgeschnitten. Die Ergebnisse der paarweisen Klassenbinarisierungen sind jeweils zur Hälfte geringfügig besser oder schlechter als Bagging. Die Dekodierungsmethoden Weighted Voting und PKPD haben bessere Werte aufgewiesen, Voting und HT entsprechend schlechtere. Abermals schneidet HT am schlechtesten ab. Die Verbesserungen der ungeordneten Klassenbinarisierungen und gegebenfalls die der paarweisen Klassenbinarisierungen sind nicht hoch genug, um den Einsatz von Klassenbinarisierungen mit Bagging zu rechtfertigen.

Zur gleichen Schlußfolgerung kommen wir, wenn wir die Ergebnisse der Vorzeichentests der vierten Testreihe betrachten. Die Verfahren Bagging, die ungeordnete und die paarweise Klassenbinarisierung sind statistisch gleichwertig zu einem Naive Bayes Klassifizierer. Interessanterweise hat die paarweise Klassenbinarisierung, bei der die Dekodierungsmethode HT verwendet wurde, als einziges Verfahren häufiger gegen einen Naive Bayes Klassifizierer verloren als gewonnen.

Fassen wir nun die Ergebnissen der dritten und vierten Testreihe zusammen. Die Kombination von Klassenbinarisierungen mit Ensemble-Methoden hat in allen Fällen zu einer niedrigeren Fehlerrate als die eines Naive Bayes Klassifizierers geführt. Jedoch hat diese Vorgehensweise nur bei AdaboostM1 zu statistisch besseren Verfahren geführt. Das reguläre Bagging hat in einigen Fällen eine bessere Fehlerrate aufgewiesen als die Klassenbinarisierungen mit Bagging. Nur die Klassenbinarisierungen mit AdaboostM1 waren immer besser als das reguläre AdaboostM1. Zusammenfassend führt die Kombination von Klassenbinarisierungen mit Ensemble-Methoden zwar zu niedrigeren Fehlerraten,

aber der erhöhte Aufwand dieser Vorgehensweise zahlt sich nur in wenigen Fällen aus.

Ingesamt führen die Ergebnisse der Testreihen, bei denen wir Klassenbinarisierungen untersucht haben, zu mehreren Schlußfolgerungen. Die binäre Diskretisierung stellt eine geringfügige Verbesserung zur globalen dar. Klassenbinarisierungen sind trotz meist niedrigerer Fehlerraten gleichwertig zu einem Naive Bayes Klassifizierer, wobei die ungeordnete und die paarweise Klassenbinarisierung im wesentlichen gleich gute Ergebnisse liefern. Wenden wir auf Klassenbinarisierungen Ensemble-Methoden an, erhalten wir niedrigere Fehlerraten und teilweise statistisch bessere Verfahren. Diese Vorgehensweise stellt jedoch einen stark erhöhten Aufwand dar, der sich in vielen Fällen nicht auszahlt. Hierbei haben die paarweisen Klassenbinarisierungen leicht bessere Ergebnisse aufgewiesen als die ungeordneten Klassenbinarisierungen. Bei den Dekodierungsmethoden hat in den meisten Fällen Weighted Voting am besten abgeschnitten. In allen Fällen hat die Dekodierungsmethode HT die schlechtesten Ergebnisse geliefert, obwohl ihre Berechnung am aufwendigsten ist.

7. Zusammenfassung

Abschließend möchten wir die Erkenntnisse, die wir im Laufe dieser Arbeit gewonnen haben, nochmals zusammenfassen. Bei den Klassenbinarisierungen mit einem Naive Bayes Klassifizierer haben wir zwei interessante Feststellungen gemacht. Erstens entspricht die ungeordnete Klassenbinarisierung nicht einem regulärem Naive Bayes Klassifizierer, obwohl beide eine ähnliche Struktur aufweisen. Allerdings sind die beiden Verfahren statistisch gleichwertig zueinander. Zweitens sind die paarweisen Klassenbinarisierungen unabhängig von den geläufigen Dekodierungsmethoden Voting, Weighted Voting und denen, die von [HT97, PKPD94] vorgeschlagen wurden, äquivalent zu einem Naive Bayes Klassifizierer, falls die Implementierung dieser Verfahren mit der theoretischen Berechnung übereinstimmt. Dies ist der Fall, falls die Vorverarbeitung der Daten (zum Beispiel die Diskretisierung von kontinuierlichen Attributen) auf dem gesamten Datensatz und nicht jeweils auf den Daten der Klassenpaare erfolgt.

Bei unseren Experimenten haben wir untersucht, wie sich die Vorverarbeitung auf den Daten der binären Probleme beziehungsweise der Klassenpaare auf die Ergebnisse der Klassenbinarisierungen auswirkt. Hierfür haben wir die Diskretisierung von kontinuierlichen Attributen betrachtet. Die Wahl zwischen einer Diskretisierung auf den gesamten Daten beziehungsweise auf den Daten der binären Probleme hat bei den verschiedenen Klassenbinarisierungen zu unterschiedlichen Ergebnissen geführt. Während bei der ungeordneten Klassenbinarisierung diese Wahl keine Auswirkung gehabt hat, wurden die Ergebnisse der paarweisen Klassenbinarisierungen leicht, aber nicht signifikant verbessert, wenn die Diskretisierung auf die binären Probleme angewandt wurde. Es wäre nun interessant zu untersuchen, wie sich die beiden unterschiedlichen Ansätze der Diskretisierung beziehungsweise der Vorverarbeitung bei anderen Basisklassifizierern auswirken.

Bei unseren Experimenten haben wir auf die binären Probleme der Klassenbinarisierungen die Ensemble-Methoden AdaBoostM1 und Bagging angewandt. Dies hat bei allen verwendeten Klassenbinarisierungen und Dekodierungsmethoden zu einer Senkung der Fehlerrate geführt. In wenigen Fällen waren die Fehlerraten jedoch höher als die der regulären Ensemble-Methode. Dennoch sind fast alle der resultierenden Klassifizierer gleichwertig zu einem regulären Naive Bayes Klassifizierer. Nur die paarweisen Klassenbinarisierungen, bei denen wir AdaBoostM1 und die Dekodierungsmethoden Voting, Weighted Voting und [PKPD94] verwendet haben, sind signifikant besser als der reguläre Naive Bayes Klassifizierer. Zusammenfassend zahlt sich der hohe Aufwand dieser Verfahren nur in wenigen Fällen aus.

Im Verlauf dieser Arbeit haben wir einen wahrscheinlichkeitstheoretischen Ansatz vorgestellt, der eine paarweise Berechnung eines Naive Bayes Klassifizierer vorschlägt. Hierfür benötigen wir zusätzlich zu den ohnehin für einen Naive Bayes Klassifizierer erforderlichen Wahrscheinlichkeiten noch die Wahrscheinlichkeiten von Klassenpaaren.

Diese können regulär mit den Wahrscheinlichkeiten eines Naive Bayes Klassifizierers oder paarweise durch die Bestimmung der relativen Häufigkeiten von Klassenpaaren abgeschätzt werden. Bei unseren Experimenten haben die aus diesem Ansatz resultierenden Methoden sehr unterschiedliche Ergebnisse geliefert. Keine der Methoden hat bessere Ergebnisse erzielt als ein Naive Bayes Klassifizierer. Dennoch war nur die Hälfte von ihnen signifikant schlechter als der Naive Bayes Klassifizierer. Interessanter ist jedoch die Tatsache, daß der paarweise Berechnungsansatz, bei dem wir die Wahrscheinlichkeiten von Klassenpaaren durch deren relative Häufigkeiten abgeschätzt haben, schlechtere Ergebnisse als der reguläre Ansatz, dessen Berechnung durch einen regulären Naive Bayes Klassifizierer erfolgt, liefert. Insgesamt ist die Anwendung dieser Methoden nicht ratsam.

A. Grundlagen der Statistik

Wir möchten in diesem Anhang einen Einblick in die Grundlagen der Statistik geben. Dabei orientieren wir uns an [LW00].

A.1. Wahrscheinlichkeitsräume

In der Wahrscheinlichkeitstheorie werden mathematische Modelle zur Beschreibung von zufälligen Vorgängen, wie sie meist der Ermittlung von Meßreihen zugrunde liegen, bereitgestellt und analysiert. Ein Vorgang, der so präzise beschrieben wurde, daß er beliebig oft wiederholt werden kann, dessen Endzustand oder *Ergebnis* außerdem vom Zufall abhängt und daher nicht vorhersagbar ist, heißt *Zufallsexperiment*. Wir gehen davon aus, daß die Menge der möglichen Ergebnissen soweit bekannt ist, daß wir jedem Ergebnis in eindeutiger Weise ein Element ω einer Menge Ω zuordnen können. Wenn wir im folgendem von einem Zufallsexperiment reden, soll jeweils geklärt sein, welche *Ergebnismenge* zugrunde liegt. Dann ist auch klar, was es bedeuten soll, wenn wir kurz vom Ergebnis ω sprechen.

Teilmengen A, B, C,... von Ω heißen *Ereignisse*. Wenn ein Ergebnis ω mit $\omega \in A$ auftritt, sagt man „das Ereignis A tritt ein“. Häufig ist man an Ereignissen interessiert, die durch Zusammensetzen anderer Ereignis zustandekommen. Wir sagen:

- Das Ereignis „A oder B“ tritt ein, wenn ein $\omega \in A \cup B$ auftritt.
- Das Ereignis „A und B“ tritt ein, wenn ein $\omega \in A \cap B$ auftritt.

$\overline{A}$ nennen wir das zu A komplementäre Ereignis ($\overline{A} = \Omega \backslash A$). Zwei Ereignisse A und B heißen *unvereinbar* oder *disjunkt*, wenn $A \cap B = \emptyset$ gilt. Die leere Menge $\emptyset$ nennt man das *unmögliche Ereignis*. Einelementige Teilmengen $\{\omega\}$ von Ω heißen *Elementarereignisse*. Für eine Folge von Ereignissen $(A_i)_{i\in\mathbb{N}}$ betrachtet man auch $\bigcup_{i=1}^{\infty} A_i$ und $\bigcap_{i=1}^{\infty} A_i$ als Ereignisse. Das „System der Ereignisse“ bildet gemäß der folgenden Definition eine σ-Algebra.

Definition A.1 (σ-Algebra) Sei Ω eine nichtleere Menge. Ein System $\mathcal{A}$ von Teilmengen von Ω heißt σ-Algebra über Ω, falls

1. $\Omega \in \mathcal{A}$.
2. Für $A \in \mathcal{A}$ gilt auch $\overline{A} \in \mathcal{A}$.
3. Aus $A_i \in \mathcal{A}$ für alle $i \in \mathbb{N}$ folgt $\bigcup_{i=1}^{\infty} A_i \in \mathcal{A}$.

Ist neben einer Ergebnismenge Ω eine σ-Algebra über Ω gegeben, so heißen alle $A \in \mathcal{A}$ *Ereignisse*.

Führt man ein Zufallsexperiment wiederholt unter den gleichen Bedingungen durch und bezeichnet n_A die Anzahl der Versuchsdurchführungen unter den ersten n, bei denen ein bestimmtes Ereignis A auftritt, so strebt die Folge der relativen Häufigkeiten $h_n(A) = \frac{n_A}{n}$ mit wachsendem n in der Regel einem für A charakteristischem Zahlenwert zu. Dieser *Stabilisierungseffekt der Folge der relativen Häufigkeiten* legt es nahe, bei der mathematischnen Beschreibung von Zufallsexperimenten jedem Ereignis A einen charakteristischen Zahlenwert $\Pr(A)$ zuzuordnen, der ein Maß dafür darstellt, wie stark mit seinem Eintreten zu rechnen ist. Wenn wir nun bei einer Folge von n Versuchsdurchführungen für zwei unvereinbare Ereignisse A und B die relativen Häufigkeiten $h_n(A)$, $h_n(B)$ und $h_n(A \cup B)$ betrachten, so gilt offensichtlich $h_n(A)+h_n(B) = h_n(A \cup B)$. Daher liegt es nahe, die Zahlen $\Pr(A)$, $\Pr(B)$ und $\Pr(A \cup B)$ so zu wählen, daß auch $\Pr(A) + \Pr(B) = \Pr(A \cup B)$ gilt, da diese Zahlenwerte sich den relativen Häufigkeiten annähern sollen. Diese Additivitätseigenschaft erweist sich auch bei Folgen von paarweise unvereinbaren Ereignissen als nützlich. Man spricht dabei von *σ-Additivität*. Dies führt zur folgenden Definition.

Definition A.2 (Wahrscheinlichkeitsmaß, Wahrscheinlichkeitsraum) Ist Ω eine Ergebnismenge und ist $\mathcal{A}$ eine σ-Algebra von Ereignissen (über Ω), so heißt eine Abbildung $\Pr : \mathcal{A} \to \mathbb{R}$ ein *Wahrscheinlichkeitsmaß*, wenn gilt:

1. $\Pr(A) \geq 0$ für alle $A \in \mathcal{A}$,
2. $\Pr(\Omega) = 1$,
3. $\Pr(\bigcup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} \Pr(A_i)$ für paarweise unvereinbare Ereignisse $A_1, A_2, ... \in \mathcal{A}$.

Das Tripel $(\Omega, \mathcal{A}, \Pr)$ heißt *Wahrscheinlichkeitsraum*. $\Pr(A)$ heißt *Wahrscheinlichkeit* des Ereignisses A.

Bemerkung A.3 Die Eigenschaften 1. bis 3. eines Wahrscheinlichkeitsmaßes werden als *Axiome von Kolmogoroff* bezeichnet. Der russische Mathematiker A. N. Kolmogoroff (1903-1987) schlug 1933 einen axiomatischen Aufbau der Wahrscheinlichkeitstheorie vor, der sich als nützlich erwiesen hat.

Im Folgenden gehen wir ohne besondere Erwähnung immer von einem Wahrscheinlichkeitsraum $(\Omega, \mathcal{A}, \Pr)$. Aus den Eigenschaften von Pr in Definition A.2 ergeben sich noch folgende Rechenregeln:

Satz A.4 *1.* $0 \leq \Pr \leq 1$ *für alle* $A \in \mathcal{A}$

2. $\Pr(\emptyset) = 0$,

3. $\Pr(\overline{A}) = 1 - \Pr(A)$ *für alle* $A \in \mathcal{A}$,

4. $A, B \in \mathcal{A}$, $A \subset B \Rightarrow \Pr(A) \leq \Pr(B)$,

5. $\Pr(A \cup B) = \Pr(A) + \Pr(B) - \Pr(A \cap B)$ *für alle* $A, B \in \mathcal{A}$,

Beweis. Folgt unmittelbar aus Definition A.2. □

A.2. Bedingte Wahrscheinlichkeit und Unabhängigkeit

Angenommen wir wiederholen ein Zufallsexperiment n-mal unter den gleichen Bedingungen. Tritt bei den einzelnen Versuchsdurchführungen das Ereignis A genau n_A-mal ein, das Ereignis B genau n_B-mal und das Ereignis $A \cap B$ („A und B gleichzeitig“) genau $n_{A\cap B}$-mal, dann ist $n_{A\cap B}/n_B$ die relative Häufigkeit des Eintretens von A in der Serie von Versuchsdurchführungen, bei denen das Ereignis B eintritt. Wir betrachten bei dieser Überlegung nur jene Versuchsdurchführungen, die die Bedingung „B tritt ein“ erfüllen. Man spricht daher auch von der bedingten Wahrscheinlichkeit von A unter der Bedingung B. Die offensichtlich gültige Gleichung

$$\frac{n_{A\cap B}}{n_B} = \frac{\frac{n_{A\cap B}}{n}}{\frac{n_B}{n}} \tag{A.1}$$

legt die folgende Definition der *bedingten Wahrscheinlichkeit* nahe:

Definition A.5 (Bedingte Wahrscheinlichkeit) Sei $(\Omega, \mathcal{A}, \Pr)$ ein Wahrscheinlichkeitsmaß. Sind $A, B \in \mathcal{A}$ Ereignisse und gilt $\Pr(B) > 0$, so heißt

$$\Pr(A|B) = \frac{\Pr(A \cap B)}{\Pr(B)} \tag{A.2}$$

die bedingte Wahrscheinlichkeit von A unter der Bedingung B.

Wir gehen bei der obigen Definition davon aus, daß die Wahrscheinlichkeiten der Ereignisse B und $A \cap B$ bekannt sind, damit wir die bedingte Wahrscheinlichkeit $\Pr(A|B)$ berechnen können. In der Praxis werden jedoch häufig Experimente durch Angabe bedingter Wahrscheinlichkeiten beschrieben und die Wahrscheinlichkeit von Ereignissen der Form $A \cap B$ durch

$$\Pr(A \cap B) = \Pr(A|B) \cdot \Pr(B) \tag{A.3}$$

beziehungsweise von A durch

$$\Pr(A) = \Pr(A \cap B) + \Pr(A \cap \overline{B}) = \Pr(A|B) \cdot \Pr(B) + \Pr(A|\overline{B}) \cdot \Pr(\overline{B}) \tag{A.4}$$

berechnet. Allgemein gilt für das Rechnen mit bedingten Wahrscheinlichkeiten folgende Sätze, die wir nicht beweisen werden, aber leicht nachvollziehbar sind.

Satz A.6 (Satz von der vollständigen Wahrscheinlichkeit) *Seien* $B_1, ..., B_n$ *paarweise disjunkte Ereignisse mit* $\bigcup_{i=1}^{n} B_i = \Omega$ *und* $\Pr(B_i) > 0, i = 1, ..., n$. *Dann gilt für ein Ereignis* A

$$\Pr(A) = \sum_{i=1}^{n} \Pr(A|B_i) \cdot \Pr(B_i). \tag{A.5}$$

Satz A.7 (Satz von Bayes) *Unter den Voraussetzungen von Satz A.6 gilt im Falle von* $\Pr(A) > 0$

$$\Pr(B_i|A) = \frac{Pr(A|B_i) \cdot \Pr(B_i)}{\Pr(A)} \text{ für } i = 1, ..., n. \tag{A.6}$$

Satz A.8 (Multiplikationssatz) *Seien* $A_1, ..., A_n$ *Ereignisse mit* $\Pr(\bigcap_{i=1}^{n} A_i) > 0$. *Dann gilt:*

$$\Pr(\bigcap_{i=1}^{n} A_i) = \Pr(A_1) \cdot \Pr(A_2|A_1) \cdot \Pr(A_3|A_1 \cap A_2) \cdot ... \cdot \Pr(A_n|A_1 \cap ... \cap A_{n-1}) \quad \text{(A.7)}$$

Für unvereinbare Ereignisse A und B kann es Situationen geben, in denen das Wissen davon, daß B eingetreten ist, überhaupt nichts darüber aussagt, ob A eingetreten ist oder nicht. Das heißt, daß A und B *unabhängig* voneinander eintreten.

Definition A.9 (Unabhängigkeit) Zwei Ereignisse A und B heißen (stochastisch) *unabhängig*, falls

$$\Pr(A \cap B) = \Pr(A) \cdot \Pr(B) \quad \text{(A.8)}$$

gilt.

Bemerkung A.10 (Unabhängigkeit) Sind die Ereignisse A und B unabhängig, so sind es auch die Ereignisse A und $\overline{B}$, die Ereignisse $\overline{A}$ und B sowie die Ereignisse $\overline{A}$ und $\overline{B}$.

Definition A.11 (Unabhängigkeit) n Ereignisse $A_1, ..., A_n$ heißen *unabhängig* (oder vollständig unabhängig), falls für jede Zahl $k = 2, ..., n$ und jede nichtleere k-elementige Teilmenge $\{i_1, i_2, ..., i_k\}$ von $\{1, ..., n\}$

$$\Pr(A_{i_1} \cap A_{i_2} \cap ... \cap A_{i_k}) = \Pr(A_{i_1}) \cdot \Pr(A_{i_2}) \cdot ... \cdot \Pr(A_{i_k}) \quad \text{(A.9)}$$

gilt.

Bemerkung A.12 Man beachte, daß im Allgemeinen die Unabhängigkeit von n Ereignissen $A_1, ..., A_n$ nicht aus der Unabhängigkeit von je zwei der Ereignisse folgt.

A.3. Zufallsvariablen und Verteilungen

Bei einer Vielzahl von Zufallsexperimenten tritt als Versuchergebnis unmittelbar ein Zahlenwert auf. Dies kann zum Beispiel bei der Messung der Körpergrößen von zufällig ausgewählten Person vorkommen. In vielen Fällen interessiert man sich auch für einen durch das Ergebnis bestimmten Zahlenwert, obwohl die bei einem Zufallsexperiment auftretenden Ergebnisse nicht unbedingt selbst Zahlenwerte sind.

Bei der mathematischen Beschreibung dieser Situation wird jedem Ereignis ω eine reelle Zahl $X(\omega)$ zugeordnet. Die Zuordnungsvorschrift ist damit eine Abbildung $X : \Omega \to \mathbb{R}$.

Das eintretende Ergebnis ω des Zufallsexperimentes hängt vom Zufall ab, deshalb wird auch der daraus ermittelte Zahlenwert $X(\omega)$ zufallsabhängig sein. Aus diesem Grund interessiert man sich für die Wahrscheinlichkeit, daß $X(\omega)$ in einem bestimmten Intervall

$I \subset \mathbb{R}$ liegt. Wir betrachten hierfür die Gesamtheit aller Ergebnisse ω, für die $X(\omega) \in I$ gilt, also die folgende Teilmenge von Ω:

$$A_I = \{\omega \in \Omega : X(\omega) \in I\}.$$

Ist diese Teilmenge ein Ereignis des Wahrscheinlichkeitsraumes $(\Omega, \mathcal{A}, \Pr)$, das heißt es gilt $A_I \in \mathcal{A}$, so ist $\Pr(A_I)$ die gesuchte Wahrscheinlichkeit. Damit die Wahrscheinlichkeit, daß $X(\omega)$ in einem Intervall I liegt, angegeben werden kann, muß sichergestellt sein, daß für die Menge A_I eine Wahrscheinlichkeit definiert ist. Diese Überlegung führt zur folgenden Definition.

Definition A.13 (Zufallsvariable) Sei $(\Omega, \mathcal{A}, \Pr)$ ein Wahrscheinlichkeitsraum. Eine Abbildung $X : \Omega \to \mathbb{R}$ heißt *Zufallsvariable* (Zufallsgröße) über $(\Omega, \mathcal{A}, \Pr)$, falls

$$\{\omega \in \Omega : X(\omega) \in I\} \in \mathcal{A}$$

für alle Intervalle $I \subset \mathbb{R}$ gilt.

Für die Wahrscheinlichkeit eines Ereignisses $\{\omega \in \Omega : X(\omega) \in I\}$ schreiben wir abkürzend $Pr(X \in I)$ und entsprechend $\Pr(a < X \leq b)$, $\Pr(a \leq X \leq b)$, $\Pr(X = a)$ und so weiter.

Mit Hilfe der in folgender Definition erklärten Verteilungsfunktion lassen sich die Wahrscheinlichkeiten solcher Ereignisse berechnen.

Definition A.14 (Verteilungsfunktion) Sei X eine Zufallsvariable über $(\Omega, \mathcal{A}, \Pr)$. Dann heißt die Abbildung $F : \mathbb{R} \to [0, 1]$ mit

$$F(x) = \Pr(X \leq x), x \in \mathbb{R},$$

Verteilungsfunktion der Zufallsvariablen X.

Bei den kommenden Definitionen verwenden wir die folgenden Abkürzungen:

$$F(x-0) = \lim_{h>0, h\to 0} F(x-h) \qquad F(x+0) = \lim_{h>0, h\to 0} F(x+h)$$
$$F(-\infty) = \lim_{x\to-\infty} F(x) \qquad F(\infty) = \lim_{x\to\infty} F(x)$$

Verteilungsfunktionen haben stets ganz bestimmte Eigenschaften, die wir in dem folgenden Satz festhalten wollen.

Satz A.15 *Ist F die Verteilungsfunktion einer Zufallsvariablen, so gilt:*

1. *F ist monoton wachsend (nicht fallend).*
2. *F ist rechsseitig stetig, das heißt $F(x) = F(x+0)$ für alle $x \in \mathbb{R}$-*
3. *$F(-\infty) = 0$ und $F(\infty) = 1$.*

Der folgende Satz zeigt, wie mit Hilfe von F die Wahrscheinlichkeit $\Pr(X \in I)$ für alle Intervalle $I \subset \mathbb{R}$ berechnet werden kann.

Satz A.16 *Ist F die Verteilungsfunktion einer Zufallsvariablen, so gilt für $a, b \in \mathbb{R}$, $a < b$:*

1. $\Pr(a < X \leq b) = F(b) - F(a)$.
2. $\Pr(X = a) = F(a) - F(a - 0)$.
3. $\Pr(a \leq X \leq b) = F(b) - F(a) + \Pr(X = a) = F(b) - F(a - 0)$.
4. $\Pr(a \leq X < b) = F(b - 0) - F(a - 0)$.
5. $\Pr(X > a) = 1 - F(a)$

Man sieht, daß durch die Verteilungsfunktion F einer Zufallsvariablen X das „Wahrscheinlichkeitsgesetz“, nach dem die Zufallsvariable X ihre Werte annimmt, vollständig beschrieben ist. Die Wahrscheinlichkeiten $\Pr(X \in B)$ für viele Teilmengen B von $\mathbb{R}$ (wie zum Beispiel Durchschnitte und Vereinigungen von endlich abzählbar unendlich vielen Intervallen sowie für beliebige offene und abgeschlossene Teilmengen von $\mathbb{R}$) sind durch F eindeutig bestimmt. Dieses durch F festgelegte „Wahrscheinlichkeitsgesetz“, nach dem die Zufallsvariable X ihre Werte annimmt, wollen wir im folgenden die *Verteilung* von X nennen.

Definition A.17 (Diskrete Zufallsvariable) Eine Zufallsvariable X heißt *diskret* (*diskret verteilt*), wenn ihr Wertebereich endlich oder abzählbar unendlich ist.

Die Verteilungsfunktion (und damit auch die Verteilung) einer diskreten Zufallsvariablen X ist durch die Angabe der Werte $x_1, x_2, ...$ und der Wahrscheinlichkeiten $\Pr(X = x_1), \Pr(X = x_2), ...$ festgelegt, die man oft in Form einer Tabelle darstellt:

x_i	x_1	x_2	x_3	...
$\Pr(X = x_i)$	p_1	p_2	p_3	...

Dabei sind $p_1, p_2, ...$ nichtnegative Zahlen mit $\sum_i p_i = 1$. In diesem Fall ist die Verteilungsfunktion F eine Treppenfunktion mit Sprungstellen $x_1, x_2, ...$ und zugehörige Sprunghöhe $p_1, p_2, ...$

Betrachten wir nun die Verteilung einer diskreten Zufallsvariable, die wir im Laufe dieser Arbeit benötigen.

Definition A.18 (Binomialverteilung) Sei $n \in \mathbb{N}$ und $0 < p < 1$. Eine Zufallsvariable X mit dem Wertebereich $\{0, 1, ..., n\}$ heißt *binomialverteilt* mit den Parametern n und p (kurz: *B(n,p)-verteilt*), falls

$$\Pr(X = i) = \binom{n}{i} p^i (1 - p)^{n-i}, \ i = 0, 1, ..., n \tag{A.10}$$

gilt.

Binomialverteilte Zufallsvariablen treten in folgendem Zusammenhang auf. Ein Zufallsexperiment mit dem möglichen Ereignissen „Erfolg“ (= 1) und „Mißerfolg“ (= 0) wird unter gleichen Bedingungen genau n-mal wiederholt. Dabei soll jeweils „Erfolg“ mit der Wahrscheinlichkeit p auftreten. Die Anzahl der „Erfolge“ in diesen n Wiederholungen läßt sich durch eine $B(n,p)$-verteilte Zufallsvariable beschreiben.

Bevor wir nun eine weitere, im Laufe dieser Arbeit benötigte Verteilung vorstellen, müssen wir noch den Begriff der *stetig verteilten* Zufallsvariable erklären.

Definition A.19 (Stetig verteilte Zufallsvariable, Dichte) Eine Zufallsvariable X heißt *stetig verteilt* mit der Dichte f, falls sich ihre Verteilungsfunktion $F : \mathbb{R} \to \mathbb{R}$ in der folgenden Weise schreiben läßt:

$$F(x) = \int_{-\infty}^{x} f(t)dt,\ x \in \mathbb{R}.$$

Betrachten wir nun die Verteilung einer stetig verteilten Zufallsvariable, die wir für die Behandlung von kontinuierlichen Variablen benötigen.

Definition A.20 (Normalverteilung) Sei $p \in \mathbb{R}$ und $\sigma > 0$. Eine Zufallsvariable X heißt *normalverteilt* mit den Parametern μ und σ^2 (kurz: $N(\mu, \sigma^2)$), falls X stetig verteilt ist mit der folgenden Dichte f:

$$f(t) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{t-\mu}{\sigma}\right)^2}\ t \in \mathbb{R}. \tag{A.11}$$

Für $\mu = 0$ und $\sigma^2 = 1$ heißt X *standard-normalverteilt*. In diesem Fall ist die Verteilungsfunktion von X die Funktion

$$\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{1}{2}t^2} \tag{A.12}$$

A.4. Statistische Tests

Ein *Test* ist ein Verfahren zur Überprüfung von Annahmen über Verteilungen, die das Zustandekommen von Beobachtungsdaten beschreiben. Solche Annahmen heißen *statistische Hypothesen* oder kurz *Hypothesen*. In der Testtheorie, die einen wichtigen Bereich der Mathematischen Statistik bildet, werden solche Verfahren hergeleitet und analysiert. Liegen die Beobachtungsdaten in Form einer Meßreihe $x_1, ..., x_n$ vor, so soll aufgrund eines Tests entschieden werden, ob eine bestimmte Annahme (oder Hypothese) als widerlegt zu betrachten (zu verwerfen oder abzulehnen) ist. Ein solcher Test ist daher bereits durch die Angabe eines *Ablehnungsbereichs* (oder kritischen Bereichs) $K \subset \mathbb{R}^n$ beschrieben, falls vereinbart wird, daß die Hypothese immer dann abzulehnen ist, wenn für das n-Tupel der Beobachtungswerte $(x_1, ..., x_n) \in K$ gilt.

Wir betrachten nun die Gruppe der *Signifikanz-Tests*, die prüfen, ob die Beobachtungen mit einer Hypothese H_0 verträglich sind oder ob sie signifikante Abweichungen zeigen. Die zu prüfende Hypothese H_0 heißt *Nullhypothese*.

Die allgemeine Vorgehensweise bei einem Signifikanztest zum *(Signifikanz-)Niveau* α:

1. Verteilungsannahme
2. Formulierung der Nullhypothese H_0
3. Wahl der Testgröße T und Bestimmung ihrer Verteilung unter H_0
4. Bestimmung des kritischen Bereiches K zum Niveau α
5. Entscheidungsregel: Gilt für das Beobachtungsereignis $(x_1, ..., x_n) \in K$, so wird H_0 abgelehnt, im anderen Fall wird gegen H_0 nichts eingewendet.

Durch das gewählte Niveau α wird das Risiko einer Fehlentscheidung quantifiziert. Falls die Hypothese H_0 zutrifft, ist α die Wahrscheinlichkeit dafür, daß sie (zu Unrecht) abgelehnt wird.

Literaturverzeichnis

[AMMR95] Rangachari Anand, Kishan G. Mehrotra, Chilukuri K. Mohan, and Sanjay Ranka. Efficient classification for multiclass problems using modular networks. *IEEE Transactions on Neural Networks*, 6:117–124, 1995.

[Bay99] Stephen D. Bay. Nearest neighbor classification from multiple feature subsets. *Intelligent Data Analysis*, 3(3):191–209, 1999.

[BK99] Eric Bauer and Ron Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36(1-2):105–139, 1999.

[Bou04] Remco R. Bouckaert. Naive bayes classifiers that perform well with continuous variables. In Geoffrey I. Webb and Xinghuo Yu, editors, *Australian Conference on Artificial Intelligence (ACAI)*, volume 3339 of *Lecture Notes in Computer Science*, pages 1089–1094. Springer, 2004.

[Bre96a] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[Bre96b] Leo Breiman. Heuristics of instability and stabilization in model selection. *Annals of Statistics*, 24(6):2350–2383, 1996.

[CB91] Peter Clark and Robin Boswell. Rule induction with CN2: Some recent improvements. In Yves Kodratoff, editor, *Proceedings of the 5th European Working Session on Learning (EWSL)*, volume 482 of *Lecture Notes in Computer Science*, pages 151–163. Springer, 1991.

[CN89] Peter Clark and Tim Niblett. The CN2 induction algorithm. *Machine Learning*, 3:261–283, 1989.

[CV95] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[DB95] Thomas G. Dietterich and Ghulum Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research (JAIR)*, 2:263–286, 1995.

[Die97] Thomas G. Dietterich. Machine learning research: Four current directions. *AI Magazine*, 18(4):97–136, 1997.

[Die00a] Thomas G. Dietterich. Ensemble methods in machine learning. In Josef Kittler and Fabio Roli, editors, *Multiple Classifier Systems*, volume 1857 of *Lecture Notes in Computer Science*, pages 1–15. Springer, 2000.

[Die00b] Thomas G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, 40(2):139–157, 2000.

[DKS95] James Dougherty, Ron Kohavi, and Mehran Sahami. Supervised and unsupervised discretization of continuous features. In *Proceedings of the 12th International Conference on Machine Learning (ICML)*, pages 194–202. Morgan Kaufmann, 1995.

[ET93] Bradley Efron and Robert J. Tibshirani. *An Introduction to the Bootstrap*. Chapman & Hall, 1993.

[FI93] Usama M. Fayyad and Keki B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1022–1029, 1993.

[Fre95] Yoav Freund. Boosting a weak learning algorithm by majority. *Information and Computation*, 121(2):256–285, 1995.

[FS96] Yoav Freund and Robert E. Schapire. Experiments with a new boosting algorithm. In *Proceedings of the 13th International Conference (ICML)*, pages 148–156. Morgan Kaufmann, 1996.

[FS97] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences (JCSS)*, 55(1):119–139, 1997.

[Für02a] Johannes Fürnkranz. Hyperlink ensembles: a case study in hypertext classification. *Information Fusion*, 3(4):299–312, 2002.

[Für02b] Johannes Fürnkranz. Round robin classification. *Journal of Machine Learning Research (JMLR)*, 2:721–747, 2002.

[Für03] Johannes Fürnkranz. Round robin ensembles. *Intelligent Data Analysis*, 7(5):385–403, 2003.

[HHW00] Chun-Nan Hsu, Hung-Ju Huang, and Tzu-Tsung Wong. Why discretization works for naive bayesian classifiers. In Pat Langley, editor, *Proceedings of the 17th International Conference on Machine Learning (ICML)*, pages 399–406. Morgan Kaufmann, 2000.

[HT97] Trevor Hastie and Robert Tibshirani. Classification by pairwise coupling. In Michael I. Jordan, Michael J. Kearns, and Sara A. Solla, editors, *Advances in Neural Information Processing Systems (NIPS)*. The MIT Press, 1997.

[JL95] George H. John and Pat Langley. Estimating continuous distributions in bayesian classifiers. In Philippe Besnard and Steve Hanks, editors, *Proceedings of the 11th Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 338–345. Morgan Kaufmann, 1995.

[KP90] John F. Kolen and Jordan B. Pollack. Back propagation is sensitive to initial conditions. In Richard Lippmann, John E. Moody, and David S. Touretzky, editors, *Advances in Neural Information Processing System (NIPS)*, pages 860–867. Morgan Kaufmann, 1990.

[LW00] Jürgen Lehn and Helmut Wegmann. *Einführung in die Statistik*. Teubner Studienbücher Mathematik. Teubner Verlag, third edition, 2000.

[MST94] Donald Michie, David J. Spiegelhalter, and C.C. Taylor. *Machine learning, neural and statistical classification*. Ellis Horwood, 1994. edited collection.

[OM99] David W. Opitz and Richard Maclin. Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research (JAIR)*, 11:169–198, 1999.

[PKPD94] David Price, Stefan Knerr, Léon Personnaz, and Gérard Dreyfus. Pairwise neural network classifiers with probabilistic outputs. In Gerald Tesauro, David S. Touretzky, and Todd K. Leen, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 1109–1116. MIT Press, 1994.

[RV91] Philippe Réfrégier and Francois Vallet. Probabilistic approach for multiclass classification with neural networks. In *Proceedings of the 1st International Conference on Artificial Neural Networks (ICANN)*, pages 1003–1007, 1991.

[Sch90] Robert E. Schapire. The strength of weak learnability. *Machine Learning*, 5:197–227, 1990.

[WLW04] Ting-Fan Wu, Chih-Jen Lin, and Ruby C. Weng. Probability estimates for multi-class classification by pairwise coupling. *Journal of Machine Learning Research (JMLR)*, 5:975–1005, 2004.

[WW99] Jason Weston and Chris Watkins. Support vector machines for multi-class pattern recognition. In *Proceedings of the 7th Europan Symposium on Artificial Neural Networks (ESANN-99)*, pages 219–224, Bruges, Belgium, April 1999.

[YW02] Ying Yang and Geoffrey I. Webb. A comparative study of discretization methods for naive-bayes classifiers. In *Proceedings of the Pacific Rim*

Knowledge Acquisition Workshop (PKAW), pages 159–173, Tokyo, Japan, 2002.

[YW03a] Ying Yang and Geoffrey I. Webb. Discretization for naive-bayes learning: Managing discretization bias and variance. Technical Report 2003/131, School of Computer Science and Software Engineering, Monash University, 2003.

[YW03b] Ying Yang and Geoffrey I. Webb. On why discretization works for naive-bayes classifiers. In Tamás D. Gedeon and Lance Chun Che Fung, editors, *Australian Conference on Artificial Intelligence*, volume 2903 of *Lecture Notes in Computer Science*, pages 440–452. Springer, 2003.

Abbildungsverzeichnis

Tabellenverzeichnis