Technische Universität Darmstadt
Fachbereich Informatik
Knowledge Engineering

Diplomarbeit

Paarweise Klassifikation für Probleme mit geordneten Klassenwerten

Ge Hyun Nam

Betreuer
Prof. Dr. J. Fürnkranz

September 2007

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.

Darmstadt, September 2007 Ge Hyun Nam

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1. Einleitung

1.1. Einleitung und Motivation

Heutzutage werden enorme Mengen von Informationen in Datenbanken gespeichert. Die Datenmengen wachsen mit fortschreitender Geschwindigkeit weiter an. Dies erzeugt sowohl eine Möglichkeit als auch eine Notwendigkeit für Verfahren, die über das reine Ansammeln und Organisieren von Daten hinaus gehen. Vielmehr bedarf es an intelligenten Verfahren, mithilfe dessen Zusammenhänge und Abhängigkeiten in den Daten erkannt werden können. Das so gewonnene Wissen kann genutzt werden, um den Prozess einer Entscheidungsbildung zu verbessern.

Zum Beispiel könnten die Daten eines vorausgegangenen Einkaufs eines Kunden interessantes Wissen darüber beinhalten, welches Produkt jeder einzelne Kunde zu kaufen neigt. Dieses Wissen kann genutzt werden, um den Absatz einer Firma zu steigern. In einem anderen Beispiel könnten die Daten eines Krankenhauses bezüglich eines Patienten interessantes Wissen darüber enthalten, welche Patienten potentiell gefährdet sind an einer gegebenen Krankheit zu erkranken. Dieses Wissen kann zu einer besseren Diagnose und Behandlung von zukünftigen Patienten führen.

Maschinelles Lernen ist der Begriff, unter dem sich der Aufbau von solchen Algorithmen und Techniken zusammenfassen lässt, die es Computern ermöglichen zu lernen. Sie stellen die Verfahren dar, um Wissen aus Erfahrung zu generieren. Anhand von Trainingsdaten lernt ein Verfahren zu verallgemeinern. Die vorhandenen Trainingsdaten werden nicht einfach auswendig gelernt, sondern es werden Gesetzmäßigkeiten innerhalb der Lerndaten erkannt. Ein solches Verfahren ist auch in der Lage, unbekannte Daten zu beurteilen.

Einen speziellen Fall der Beurteilung stellt die Klassifikation dar. Hier geht es darum ein Objekt einer konkreten Klasse oder Kategorie zuzuordnen. Die vorliegende Arbeit befasst sich mit der Klassifikation. Ein Problem, welches in der Litaratur nur wenig Beachtung findet, ist das der geordneten Klassenwerte. In der Praxis kommt es häufig vor, dass die Klassenwerte in einer gewissen Relation zueinander stehen. In den oben erwähnten Anwendungsfällen gibt es die Klassen Produkt und Krankheit. Innerhalb dieser Klasse existiert eine Relation zwischen den Elementen, wie z.B. der Preis eines Produkts oder Grad der Beinflussung auf das Wohlbefinden einer Krankheit. Unter Umständen könnte dies auch relevantes Wissen darstellen.

In dieser Arbeit soll ein derartiges Verfahren, welches die Klassenordnung berücksichtigt, vorgestellt werden. Hierzu wurde ein vorhandenes Verfahren erweitert. Bei Problemen mit mehr als zwei Klassen wird gewöhnlich eine Binärisierung durchgeführt. Dabei werden die Daten in Teildaten aufgeteilt, die aus jeweils zwei Klassen bestehen. Anhand dieser wird ein paarweiser Klassifzierer trainiert. Diese Trainingsphase wurde nun so erweitert, dass ordnungsrelevante Klassen mit in die Beurteilung einfließen. Es ist zu

untersuchen, ob diese Modifikation zu einer Verbesserung der Resultate führt.

1.2. Gliederung

In Kapitel 2 wird ein Überblick über das Maschinelle Lernen gegeben. Es wird erläutert, wie Probleme dargestellt werden und wie anhand dieser ein Klassifizierer trainiert wird. Weiterhin werden verschiedene Lernverfahren vorgestellt, die für die Experimente verwendet wurden. Anschließend wird darauf eingegangen, welche Möglichkeiten es gibt Probleme mit mehreren Klassen zu behandeln.

In Kapitel 3 wird der Begriff der Klassenordnung definiert. Des weiteren werden zwei Ansätze aus der Literatur beschrieben, die in der Lage sind, Probleme mit geordneten Klassenwerte zu behandeln. Dies ist zum einen die Klassifikation mittels Regressionslernen, sowie die ordinale Klassifikation von Frank und Hall.

Kapitel 4 beschäftigt sich mit dem paarweise geordneten Klassifizierer. Neben der genauen Funktionsweise wird auch die Laufzeit des Verfahrens analysiert. Es wird untersucht, wie sich Probleme mit geordneten Klassenwerten von herkömmlichen Problemen unterscheiden und was es bei der Auswertung der Ergebnisse zu beachten gibt.

Die Ergebnisse der Experimente werden in Kapitel 5 präsentiert. Der paarweise geodnete Klassifizierer wurde dabei mit anderen Verfahren verglichen. Der genaue Aufbau der Experimente wird hier beschrieben. Abschließend werden die Ergebnisse ausgewertet.

In Kapitel 6 werden die wichtigsten Ergebnisse zusammengefasst und ein Ausblick auf weitere Möglichkeiten zur Klassifikation von Problemen mit geordneten Klassenwerten gegeben.

2. Grundlagen

2.1. Maschinelles Lernen

Maschinelles Lernen ist ein umfangreiches Teilgebiet der künstlichen Intelligenz. Es behandelt den Entwurf und die Entwicklung von Algorithmen und Techniken, die es Computern ermöglichen zu lernen. Unter Lernen versteht man die Gewinnung von Regeln und Mustern aus gegebenen Datensätzen. Dabei liegt der Schwerpunkt auf der automatischen Erkennung dieser Zusammenhänge.

Es lassen sich vier verschiedene Arten von Lernen unterscheiden. Beim Klassifikationslernen geht es darum, anhand von bereits klassifizierten Beispielen einen Ansatz zu erlernen, um neue, noch ungesehene Daten zu klassifizieren. Beim Assoziationslernen wird nicht nur nach Mustern zur Vorhersage der Klasse gesucht, sondern es werden jegliche Assoziationen zwischen den Eigenschaften eines Datensatzes untersucht. Das Clustering fasst Beispiele, die zusammengehören, zu Gruppen zusammen. Bei der numerischen Vorhersage ist das vorhergesagte Ergebnis keine diskrete Klasse, sondern ein numerischer Wert.

Im Folgenden soll näher auf die Klassifikation eingegangen werden. Einige der möglichen Anwendungsgebiete für Klassifikation umfassen z.B. Spracherkennung, Handschriftenerkennung, automatisierte Diagnoseverfahren, Aktienmarktanalysen oder Klassifikation von DNA-Sequenzen.

2.1.1. Darstellung von Problemen

Bevor der Lernvorgang beginnen kann, ist es notwendig eine einheitliche Form zur Speicherung der Daten festzulegen. Es gibt verschiedene Möglichkeiten die vorhandenen Daten der Probleme darzustellen. Eine mögliche Variante ist es, sie in strukturierten Datensätzen zusammenzufassen. Eine Instanz oder ein Beispiel eines Datensatzes beinhaltet alle relevanten Fakten, die es beschreibt. Ein Datensatz setzt sich aus einzelnen Instanzen zusammen. Eine Instanz besteht aus Attributen. Ein Attribut hat einen Namen und einen Wert, welcher nominal oder numerisch sein kann. Numerische Werte sind meist reelle Zahlen, welche miteinander verglichen und sortiert werden können. Unter nominal versteht man, dass sich die möglichen Merkmale zwar unterscheiden, nicht aber in eine Rangfolge gebracht werden können. Weiterhin beinhaltet eine Instanz den Ausgangswert, welcher bei der Klassifikation immer nominal ist. Dieser Wert stellt die Klasse oder Kategorie dar.

Für ein Problem der medizinischen Diagnose z.B. besteht ein Datensatz aus allen Daten, die während verschiedenen Untersuchungen von Patienten erfasst wurden. Eine Instanz enthält die Daten eines Patienten. Die Fakten einer Untersuchung werden als

Attribute abgebildet. Sie umfassen sowohl die Symptome als auch weitere Details, wie die Krankheitsgeschichte oder die Ergebnisse der körperlichen Untersuchung und Labordiagnostik. Außerdem umfasst die Instanz die Klasse, in dem Fall die diagnostizierte Krankheit, die dem Patienten nach der Untersuchung zugeordnet wurde.

Ein anderes Beispiel ist in Tabelle 2.1 dargestellt. In dem Datensatz sind die Wetterdaten von einzelnen Tagen gespeichert. Als Klasse dient das Attribut *Play* mit den Werten *yes* und *no*. Es gibt an, ob an diesen Tag Golf gespielt wurde oder nicht.

Outlook	Temperature	Humidity	Windy	Play
sunny	85	85	false	no
sunny	80	90	true	no
overcast	83	78	false	yes
rain	70	96	false	yes
rain	68	80	false	yes
rain	65	70	true	no
overcast	64	65	true	yes
sunny	72	95	false	no
sunny	69	70	false	yes
rain	75	80	false	yes
sunny	75	70	true	yes
overcast	72	90	true	yes
overcast	81	75	false	yes
rain	71	80	true	no

Tabelle 2.1.: Beispiel für ein Wetterproblem (Quelle [15])

2.1.2. Klassifikation

Die Klassifikation oder Klassifizierung beschreibt den Prozess der Einteilung von Objekten in Klassen. Ziel ist es eine Funktion zu ermitteln, die eine Vorhersage darüber trifft, auf welche Klasse ein Objekt abzubilden ist. Dabei sind die Eingangsgrößen die Instanzen und die Ausgangsgrößen die Klassenwerte. Solch eine Funktion wird als Klassifizierer oder Klassifikator bezeichnet.

Definition 2.1 (Klassifizierer) *Sei V eine Klasse mit den Werten v_1, v_2, v_3, ..., v_c und x eine Instanz mit unbestimmtem Klassenwert. Der Klassifizierer $K : x \rightarrow V$, weist der Instanz x einen Klassenwert v_i aus V zu.*

Die Klassifikation lässt sich in zwei Schritte aufteilen, der erste ist die Trainingsphase, der zweite die Testphase oder Klassifikationsphase. Im ersten Schritt wird der Klassifizierer mittels Trainingsdaten gelernt. Man bezeichnet diesen Vorgang auch als überwachtes Lernen, da eine Funktion aus gegebenen Paaren von Ein- und Ausgaben gelernt wird. Es gibt unterschiedliche Verfahren wie das Wissen aus den Trainingsdaten dazu genutzt werden kann, um daraus Klassifizierer zu lernen und zu verbessern. Einige der wichtigsten

Lernverfahren werden im nächsten Abschnitt vorgestellt. Im zweiten Schritt wird einer Instanz eine Klasse zugewiesen.

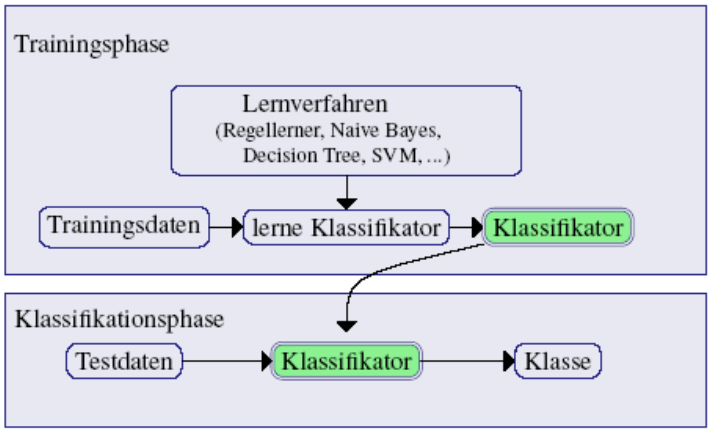

Abbildung 2.1.: Aufbau eines Klassifizierers (Quelle [12])

Der Trainingsdatensatz besteht aus einer Menge von klassifizierten Instanzen, anhand dieser die Klassifizierer gelernt werden. Nicht immer ist die von einem Klassifizierer getroffene Vorhersage korrekt. Hinzu kommt, dass der Trainingsdatensatz teilweise auch unvollständig sein kann. Es können Attribute mit fehlenden oder ungenauen Werten existieren. Die erlernten Klassifizierer stellen somit immer nur eine Vorhersage der wahrscheinlichsten Klasse dar. Um die Performanz eines Klassifizierer abschätzen zu können, wird dieser anhand eines Testdatensatzes überprüft. Der Testdatensatz besteht aus Instanzen von denen die Klassen bekannt sind. Üblicherweise tauchen die Testinstanzen nicht in der Trainingsmenge auf. Der Klassifizierer wird auf die Testmenge angewandt. Im Anschluss kann überprüft werden, ob die Instanzen der richtigen Klasse zugeordnet wurden. Als Maß für die Performanz dient die Genauigkeit (*accuracy*) oder Fehlerrate (*error rate*), welche zueinander komplementär sind.

Probleme die Instanzen mit mehr als zwei Klassen enthalten, bezeichnet man als Multiklassenprobleme. Sie stellen einen speziellen Fall dar, da manche Lernverfahren darauf ausgelegt sind nur Probleme mit zwei Klassen zu behandeln. Solche Probleme nennt man binäre Probleme. Diese werden durch $<v_i, v_j>$ veranschaulicht, indem die Klasse v_i von v_j unterschieden wird. Einen auf ein binäres Problem angewandten Klassifizierer bezeichnet man als binären Klassifizierer, dieser wird durch $K(v_i, v_j)$ beschrieben. Mithilfe von Umwandlungen sind auch binäre Klassifizierer in der Lage Multiklassenprobleme zu behandeln. Diese Vorgehensweisen werden in 2.3 beschrieben.

2.2. Lernverfahren

Es gibt eine Reihe von verschiedenen Lernverfahren, die allesamt eine andere Lerntechnik verwenden. Datensätze können die unterschiedlichsten Strukturen aufweisen. Ein Lern-

verfahren, welches auf eine bestimmte Struktur achtet, könnte für eine andere Art von Struktur komplett ungeeignet sein. In diesem Abschnitt werden die in dieser Arbeit verwendeten Techniken vorgestellt. Die Beispiele wurden aus dem Buch von Witten und Frank [15] übernommen.

2.2.1. Entscheidungsbäume

Die Entscheidungsbäume stellen einen *divide-and-conquer* Ansatz des Lernvorgangs dar. Ein Baum besteht aus einer Wurzel, inneren Knoten und Blättern. Beginnend von der Wurzel wird in jedem inneren Knoten ein Test ausgeführt. Je nach Testausgang wird verzweigt und der nächste Knoten bzw. das Blatt zurückgegeben. Jedem Blatt wird ein Klassenwert zugewiesen. Bei der Klassifikation einer Instanz wird der Pfad von der Wurzel zum Blatt verfolgt. An jeder Verzweigung wird für ein bestimmtes Attribut ein Test durchgeführt. Am Ende wird der entsprechende Klassenwert zurück gegeben. Entscheidungsbäume eignen sich gut dazu, die Repräsentation bildlich darzustellen. Abbildung 2.2 zeigt den Entscheidungsbaum für das Wetterproblem aus Tabelle 2.1.

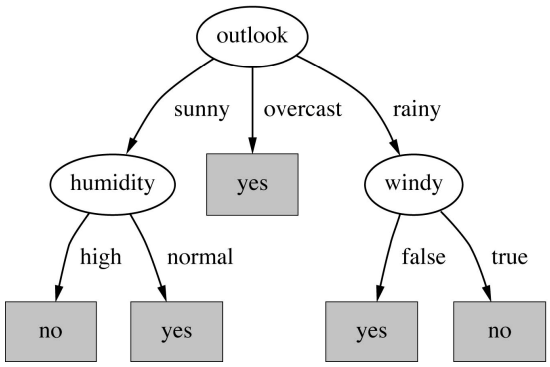

Abbildung 2.2.: Entscheidungsbaum für das Wetterproblem (Quelle [15])

Bei den Tests, die in den Knoten durchgeführt werden, kann man zwischen nominalen und numerischen Attributen unterscheiden. Bei nominalen Attributen stellt der Testausgang die möglichen Attributwerte dar. Für das Attribut *Outlook* aus dem Wetterproblem wären die möglichen Ausgänge *sunny*, *overcast* und *rainy*. Für jeden Ausgang wird ein separater Pfad erzeugt. Numerische Attribute bilden einen binären Test ab. Der zu untersuchende Wert wird mit einer Konstanten verglichen. Entweder trifft die Aussage zu oder nicht. Ein Beispiel dafür ist der Test $Temperature \geq 23$.

Die Konstruktion von Entscheidungsbäumen ist ein rekursiver Vorgang. Zunächst wird dem Wurzelknoten der Test eines bestimmten Attributes zugewiesen. Entsprechend der möglichen Ausgänge wird verzweigt. Dadurch wird der Trainingsdatensatz in Teilmengen unterteilt, für jeden Wert des Attributes entsprechend eine. Der Vorgang wird nun für jede der Teilmenge rekursiv wiederholt. Wenn alle Instanzen einer Teilmenge der gleichen Klasse zugehören, wird der entsprechende Pfad abgebrochen und der Wert als Blatt zurückgegeben.

Die Aufgabe besteht darin einen effizienten Entscheidungsbaum zu generieren. Dies soll zielorientiert und ohne unnötige Umwege erreicht werden. Dazu ist es notwendig die Baumgröße minimal zu halten, um so die Entscheidungsgeschwindigkeit zu erhöhen. Nachfolgend wird veranschaulicht, wie man bestimmt, welche Attribute für einen effizienten Test besonders gut geeignet sind. Betrachten wir das Wetterproblem, so gibt es vier Attribute zur Auswahl. Abbildung 2.3 zeigt die daraus resultierende Verteilung der Klassen *yes* und *no*. Es wird das Attribut gewählt, welches die "reinsten" Tochterknoten produziert.

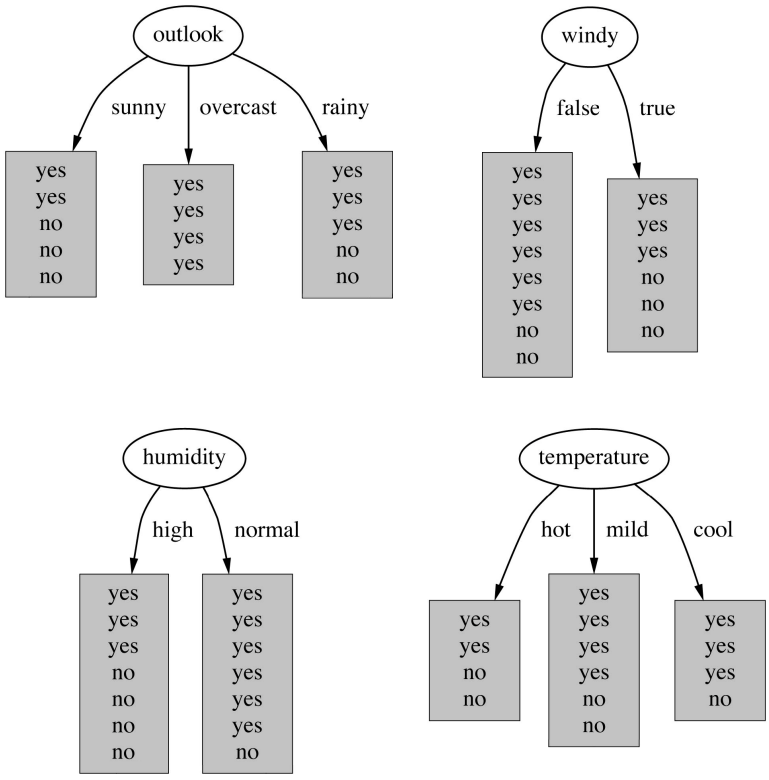

Abbildung 2.3.: Mögliche Verzweigungen und die daraus resultierenden Pfade (Quelle [15])

Als Maßeinheit für diese Reinheit dient die Entropie. Die Entropie gibt die erwartete Anzahl von Bits an, die benötigt werden, um eine Klassifikation zu kodieren. Dieser Wert steht für die Menge an Informationen, die benötigt werden, um zu bestimmen welcher Klasse eine Instanz zugehört. Die Klassifikation bezieht sich dabei auf eine zufällig gezogene Instanz aus der Menge der Trainingsdaten.

Die Entropie soll noch mal anhand eines Beispiels erläutert werden. Zur Anschaulichkeit dient der Münzwurf mit Kopf oder Zahl als mögliche Ausgänge. Die Entropie für den unbekannten Ausgang des nächsten Wurfes hat ihr Maximum wenn man davon ausgeht, dass die Münze fair ist, d. h. Kopf und Zahl haben beide gleiche Wahrscheinlichkeiten

von 1/2. Dies ist die Situation mit der größten Ungewissheit, da es am schwierigsten ist das Ergebnis des nächsten Wurfs zu bestimmen. Der Ausgang für jeden Münzwurf liefert daher Information von 1 bit.

Geht man jedoch davon aus, dass die Münze nicht fair ist, so ergibt sich weniger Ungewissheit. Das Aufkommen einer Seite ist nun wahrscheinlicher als das der anderen Seite. Dies macht sich auch an der Entropie bemerkbar. Im Durchschnitt liefert jeder Wurf in solch einem Fall weniger als 1 bit an Informationen.

Den Extremfall bildet eine Münze mit dem Kopfsymbol auf beiden Seiten. Hier existiert keine Ungewissheit. Die Entropie ist 0, da jeder Münzwurf keinerlei Information liefert.

Die Entropie wird folgendermaßen berechnet:

$$Entropie(p_1, p_2, \ldots, p_n) = -p_1 \cdot \log p_1 - p_2 \cdot \log p_2 \ldots - p_n \cdot \log p_n$$

p_i stellt die Wahrscheinlichkeitswerte dar, die addiert 1 ergeben

Mit Hilfe der Entropie kann nun ermittelt werden, welches Attribut des Wetterproblems sich am besten als Wurzelknoten eignet. Für das Attribut *Outlook* ergibt sich der erste Baum aus Abbildung 2.3. Die Verteilung der Klassen *yes* und *no* in den Knoten sind [2, 3],[4, 0] und [3, 2]. Daraus ergeben sich folgende Entropiewerte:

$$Entropie(2/5, 3/5) = 0.971\ bits$$
$$Entropie(4/4, 0/4) = 0.0\ bits$$
$$Entropie(3/5, 2/5) = 0.971\ bits$$

Unter Berücksichtigung der Instanzenanzahl in jedem Zweig kann man den Durchschnittswert für diese Baumstruktur berechnen.

$$AVG(outlook) = 5/14 \cdot 0.971 + 4/14 \cdot 0 + 5/14 \cdot 0.971 = 0.693\ bits$$

Bevor irgendeine Verzweigung erzeugt wurde, umfasste das ursprüngliche Problem neun *yes* und fünf *no* Knoten.

$$Entropie(9/14, 5/14) = 0.940\ bits$$

Die Verzweigung des Problems mithilfe des Attributs *outlook* hat zu einem Informationszuwachs (*gain*) geführt.

$$\begin{aligned} gain(outlook) &= Entropie(9/14, 5/14) - AVG(outlook) \\ &= 0.940 - 0.693 = 0.247\ bits \end{aligned}$$

Diese Berechnung wird für die restlichen Attribute durchgeführt.

$$gain(temperature) = 0.029\ bits$$
$$gain(humidity) = 0.152\ bits$$
$$gain(windy) = 0.048\ bits$$

Das Attribut mit den größten Informationszuwachs wird als Verzweigungsknoten genutzt. Dieser Vorgang wird rekursiv für die resultierenden Pfade durchgeführt, bis die Daten nicht mehr weiter aufgeteilt werden können. In dieser Arbeit wurde eine Implementation des Entscheidungsbaumlerners C4.5 [13] benutzt.

2.2.2. Regelbasierte Klassifikatoren

Regelbasierte Klassifikatoren bestehen aus einer oder mehreren Bedingungen und einer Folgerung:

$$IF\ a\ THEN\ x$$
$$IF\ b\ AND\ c\ THEN\ y$$

Die Bedingungen einer Regel sind eine Folge von Tests, die durch logische *AND*s verbunden sind. Diese Tests entsprechen im Grunde den Knoten eines Entscheidungsbaums. Man sagt eine Regel deckt eine Instanz ab, wenn die Instanz die Bedingungen der Regel erfüllt. Die Folgerung gibt die Klasse an, der die Instanz zugehört, die von der Regel abgedeckt werden.

Eine Möglichkeit eine Menge von Regeln zu erzeugen ist es, einen Entscheidungsbaum auf Regeln abzubilden. In diesem Fall wird für jeden Pfad zum Blatt eine Regel erzeugt. Für jeden Knoten, der auf dem Pfad von der Wurzel zum Blatt liegt, wird eine Bedingung zur Regel hinzugefügt. Die Folgerung entspricht der Klasse, dem das Blatt zugeordnet ist. Dieses Verfahren erzeugt eine Menge von eindeutigen Regeln, jedoch sind solche Regelsätze meistens komplexer als notwendig. Gewöhnlich wird darauf eine Vereinfachung in Form von *Pruning* angewandt, um überflüssige Tests zu entfernen.

Regelsätze sind mindestens genauso aussagestark wie Entscheidungsbäume. Viele Konzepte haben jedoch eine kürzere Beschreibung, wenn diese als Regeln dargestellt werden. Ein Beispiel dafür ist der Fall, wenn Regeln die gleiche Struktur, aber verschiedene Bedingungen besitzen.

$$IF\ a\ AND\ b\ THEN\ x$$
$$IF\ c\ AND\ d\ THEN\ x$$

Entscheidungsbäume haben Probleme, die durch Regeln veranschaulichten Disjunktionen in einer einfachen Form darzustellen. Zunächst muss ein Test als Wurzelknoten bestimmt werden. Wenn man, wie in Abbildung 2.4 gezeigt, a als ersten Test bestimmt, so muss die zweite Regel innerhalb des Baumes ein weiteres Mal wiederholt werden. Dies führt zu einem komplexen Aufbau des Baumes, indem die eigentliche Symmetrie des Problems nicht ersichtlich wird. Dieser Sachverhalt wird als das *replicated subtree problem* [15] bezeichnet.

Das Verfahren, die Regelmengen von einem Entscheidungsbaum abzubilden, basiert auf einen *top-down* Ansatz. In jedem Schritt wird nach einem Attribut gesucht, welches die Klassen am besten separiert. Eine andere Vorgehensweise ist es, jede Klasse nacheinander durchzugehen und alle in der Klasse enthaltenen Instanzen abzudecken und die restlichen Instanzen auszugrenzen. Dieser Vorgang wird als *covering*-Methode bezeichnet, da in jedem Schritt eine Regel bestimmt wird, die bestimmte Instanzen abdeckt.

Abbildung 2.5 stellt eine Menge von Instanzen bildlich in einem zweidimensionalen Raum dar. Es soll eine Bedingung gefunden werden, die möglichst viele Instanzen der gesuchten Klasse abdeckt und so viele Instanzen wie möglich der anderen Klassen ausschließt. Sei t die Anzahl der Instanzen, von denen p den Anteil der positiven Beispiele

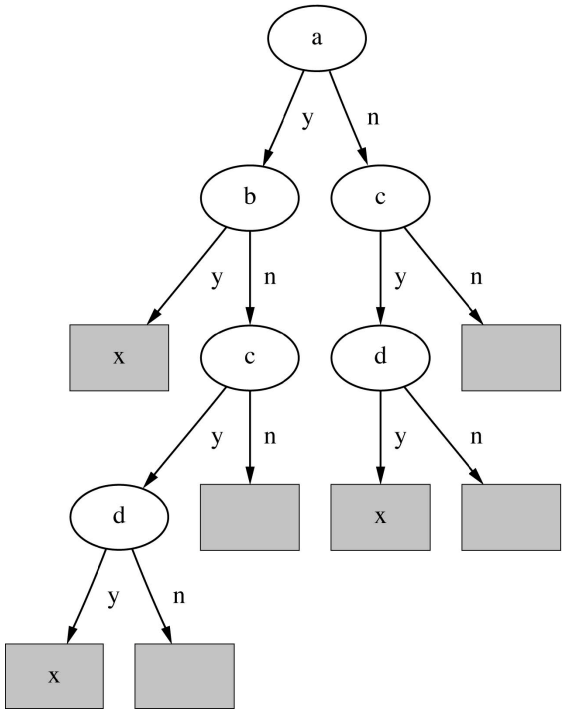

Abbildung 2.4.: Darstellung des *replicated subtree problems* - eine Regel muss mehrmals ausgeführt werden (Quelle [15])

bilden und $t - p$ beschreibe die restlichen Beispiele. Die neue Bedingung wird so gewählt, dass das Verhältnis p/t maximiert wird. Der vertikaler Schnitt erweist sich als die beste Lösung, um die Instanzen der Klasse a zu bestimmen. Die Regel dazu sieht folgendermaßen aus:

$$IF\ x > 1.8\ THEN\ a$$

Durch diese Regel werden 11 Instanzen abgedeckt, von denen 6 zu den positiven Beispielen zählen. p/t beträgt hier 6/11 und ist somit besser als ein horizontaler Schnitt, welcher einen p/t-Wert von 5/11 erzeugt hätte. Jedoch deckt diese Regel immer noch viele bs ab, so dass eine weitere Bedingung hinzugefügt wird. Ein weiterer horizontaler Schnitt führt zu der erweiterten Regel:

$$IF\ x > 1,8\ AND\ y > 1.3\ THEN\ a$$

Diese Regel deckt nun alle as außer einem ab. Um dieses letzte a auch noch abzudecken ist eine weitere Regel notwendig.

$$IF\ x > 2.8\ and\ x < 3.0\ THEN\ a$$

Das gleiche Verfahren führt zu den Regeln, die b abdecken:

$$IF\ x \leq 1.8\ THEN\ b$$
$$IF\ x > 1.8\ AND\ y \leq 1.3\ THEN\ b$$

Auch in diesem Fall ist ein a falsch abgedeckt. Um dieses a auszuschließen, muss die zweite Regel erweitert werden.

$$IF\ x > 1.8\ AND\ y \leq 1.3\ AND\ x \leq 1.8\ THEN\ b$$

Eine weitere Regel muss hinzugefügt werden, um das ausgeschlossene b abzudecken.

$$If\ x > 3.0\ THEN\ b$$

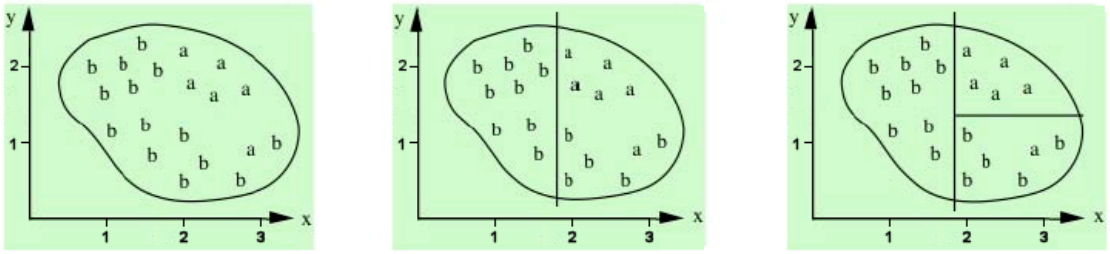

Abbildung 2.5.: *Covering*-Methode - in jedem Schritt wird eine Regel bestimmt die bestimmte Instanzen abdeckt (Quelle [15])

In Abbildung 2.6 ist die Vorgehensweise der Regellerner noch mal bildlich dargestellt. Es gibt einen Raum, in dem alle Instanzen enthalten sind, eine teilweise fertiggestellte Regel und dieselbe Regel nachdem eine neue Bedingung hinzugefügt wurde. Die neue Bedingung beschränkt die Menge der abgedeckten Instanzen. Gegebenfalls muss der Regelsatz erweitert werden, um alle Instanzen einer Klasse korrekt zu erfassen.

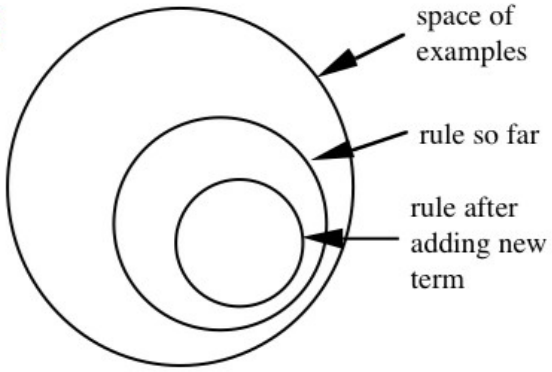

Abbildung 2.6.: Durch Hinzufügen von neuen Bedingungen wird die Menge der abgedeckten Instanzen beschränkt (Quelle [15]).

2.2.3. Support Vector Machine

Eine Support Vector Machine führt eine Klassifikation durch, indem sie eine Hyperebene erzeugt, die die Objekte optimal in zwei Klassen unterteilt. Sie nutzt lineare Trennfunk-

tionen um nicht lineare Klassengrenzen zu bestimmen. Dabei wird so vorgegangen, dass der Raum der Instanzen in einen höher dimensionalen Raum projeziert wird.

Bevor auf solch einen Fall eingegangen wird, soll zunächst das einfachere Beispiel aus Abbildung 2.7 erläutert werden. Die Instanzen werden durch Vektoren in einem Vektorraum repräsentiert, in dem folgendem Beispiel ist dieser zweidimensional. Es wird nach einer Linie, also einer 1-dimensionalen Hyperebene gesucht um die Instanzen der beiden Klassen zu separieren. Im dargestellten idealen Fall ist eine lineare Separation möglich. Es gibt sogar eine unendliche Anzahl von möglichen Linien. Zwei mögliche Separationen sind in der Grafik gezeigt. Die Aufgabe ist es nun die optimale Linie zu bestimmen. Die gestrichelten Linien links und rechts neben der Hyperebene veranschaulichen den Abstand zwischen der Trennlinie und den ihr am nächsten stehenden Vektoren. Dieser Abstand wird *margin* genannt. Die Vektoren, die die Breite des *margins* bestimmen, nennt man Support Vektoren, da sie als Stützvektoren benötigt werden, um die Hyperebene exakt zu ermitteln. Die optimale Linie ist nun die, die den *margin* zwischen den Support Vektoren maximiert. Die rechte Grafik zeigt diesen Fall.

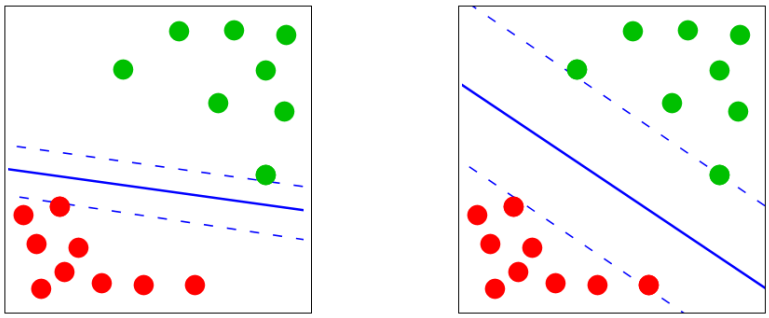

Abbildung 2.7.: Mögliche lineare Trennungen in einem zweidimensionalen Raum

Der oben beschriebene Fall geht von einer linearen Separation aus. Im Folgenden soll beschrieben werden wie vorgegangen wird, wenn die Menge der Instanzen nicht linear trennbar sind. Abbildung 2.8 stellt solch ein Problem dar. Anstatt eine nicht lineare Kurve zu bestimmen, machen Support Vector Machines Gebrauch von dem Kernel-Trick. Die zugrundeliegende Idee ist, dass der Vektorraum auf einen höher dimensionalen Raum projeziert wird, in dem es möglich ist die Menge der Vektoren linear zu trennen. In diesem Raum wird dann die entsprechende Hyperebene bestimmt. Mit Hilfe von geeigneten Kernelfunktionen ist es möglich die Kosten für die Hin- und Rücktransformation in den höher- bzw. niedrigerdimensionalen Raum so gering wie möglich zu halten. Auch hier reicht es aus nur die Support Vektoren zur Bestimmung der Klassengrenze zu verwenden. Durch diese Transformationen ist dieser Lerner enorm rechenlastig [10].

2.2.4. Bayes Lernen

Der Bayes Klassifikator ist ein auf dem *Bayes-Theorem* basierendes Verfahren. Mit Hilfe von Mitteln der Wahrscheinlichkeitstheorie wird die wahrscheinlichste Klasse unter Anwendung der Testdaten ermittelt. Dafür werden alle Attribute angesichts einer Klasse

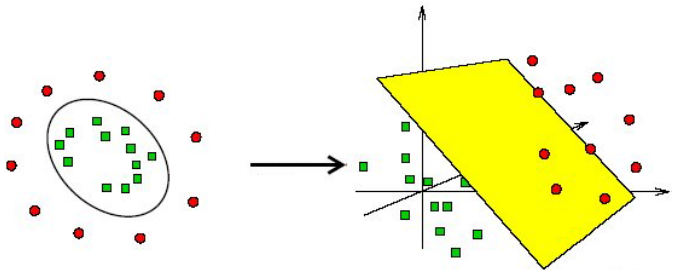

Abbildung 2.8.: Vektorraum wird auf einen höher dimensionalen Raum projeziert, in dem eine lineare Trennung möglich ist.

betrachtet und als gleich wichtig und unabhängig voneinander betrachtet. Diese Annahme wird auch als *independent assumption* bezeichnet. Sie ist zwar so nicht richtig, da einzelne Attribute meißtens verschiedene Stellenwerte aufweisen und auch untereinander abhängig sind, jedoch zeigt dieses Lernverfahren in der Praxis trotzdem sehr gute Ergebnisse.

Tabelle 2.2 zeigt eine Zusammenfassung der Wetterdaten, darin aufgelistet ist wie oft jedes Attribut mit welchem Wert auftaucht. In der oberen Hälfte ist aufgezählt wie oft die verschschiedenen Werte der vier Attribute *Outlook*, *Temperature*, *Humidity* und *Windy* vorkommen. Es wird unterteilt, wie oft davon der Wert zu einer Instanz der Klasse *yes* gehört und wie oft zur Klasse *no*. Außerdem wird erfasst, wie oft die Klasse *no* und *yes* aufgetaucht ist. Die untere Hälfte der Tabelle zeigt an, welchen Anteil die Instanzen mit einem bestimmten Attributwert von allen Instanzen der jeweiligen Klasse ausmachen. Diesen Anteil kann man auch als Wahrscheinlichkeit betrachten. Wenn 2/9 der Instanzen der Klasse *yes* für das Attribut *Outlook* den Wert *sunny* haben, so ist die Wahrscheinlichkeit, dass die Instanzen mit *Outlook* = *sunny* der Klasse *yes* angehören 2/9 (22,2%). Für die Klasse *Play* wird beschrieben wie hoch der Anteil von *yes* und *no*, bzw. wie hoch die Wahrscheinlichkeit dafür ist.

Outlook			Temperature			Humidity			Windy			Play	
	yes	no		yes	no		yes	no		yes	no	yes	no
sunny	2	3	hot	2	2	high	3	4	false	6	2	9	5
overcast	4	0	mild	4	2	normal	6	1	true	3	3		
rainy	3	2	cool	3	1								
sunny	2/9	3/5	hot	2/9	2/5	high	3/9	4/5	false	6/9	2/5	9/14	5/14
overcast	4/9	0/5	mild	4/9	2/5	normal	6/9	1/5	true	3/9	3/5		
rainy	3/9	2/5	cool	3/9	1/5								

Tabelle 2.2.: Zusammengefasstes Wetterproblem mit Mengenanzahl und Wahrscheinlichkeiten

Taucht nun ein neues Beispiel mit unbekannter Klassenzugehörigkeit für *Play* auf, so kann man mittels der bedingten Wahrscheinlichkeiten bestimmen welcher Klasse dieses Beispiel höchstwahrscheinlich angehört. Angenommen es gilt folgendes neues Beispiel zu

klassifizieren:

$$Outlook = sunny,\ Temperature = cool,\ Humidity = high,\ Windy = true,\ Play = ?$$

Die zugrundeliegende Idee ist, dass man für jede Klasse die Wahrscheinlichkeit bezogen auf die Wahrscheinlichkeitsverteilung aus den Trainingsdaten berechnet. Zunächst wird die Wahrscheinlichkeit jedes Attributs in Betracht gezogen. Jedes Attribut wird als gleich wichtig angesehen. Somit werden die Werte einfach multipliziert.

$$P(yes) = 2/9 \cdot 3/9 \cdot 3/9 \cdot 3/9 = 0.0082$$
$$P(no) = 3/5 \cdot 1/5 \cdot 4/5 \cdot 3/5 = 0.0577$$

Als nächstes wird die allgemeine Wahrscheinlichkeit für eine Klasse mitberücksichtigt.

$$P(yes) = 0.0082 \cdot 9/14 = 0.0053$$
$$P(no) = 0.0577 \cdot 5/14 = 0.0206$$

Normalisiert man die Werte, kommt man auf die prozentuale Verteilung von 20,5% für *yes* und 79,5% für *no*. Das Ergebnis deutet darauf hin, dass das neue Beispiel vorraussichtlich der Klasse *no* angehört.

Diese Berechnung basiert auf dem *Bayes-Theorem*, welcher angibt wie man mit bedingten Wahrscheinlichkeiten rechnet.

$$Bayes\text{-}Theorem : P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

Die Funktion $P(h)$ stellt die Wahrscheinlichkeit für das Zutreffen einer Hypothese, in dem Fall Klasse, vor betrachten der Trainingsdaten dar. Dies wird auch als die *a priori* Wahrscheinlichkeit bezeichnet. $P(D)$ ist die Wahrscheinlichkeit für das Beobachten der Trainingsdaten D. $P(D|h)$ ist die bedingte Wahrscheinlichkeit, dass die Trainingsdaten D beobachtet werden, wenn h zutrifft. Entsprechend ist $P(h|D)$ die bedingte Wahrscheinlichkeit, dass h für die Trainingsdaten D zutrifft. Dies wird auch als die *a posteriori* Wahrscheinlichkeit bezeichnet.

Gesucht ist die wahrscheinlichste Hypothese h aus einer Hypothesenmenge H gegeben eine Trainingsmenge. Diese Hypothese bezeichnet man als die *Maximum a posteriori Hypothese* - $hMAP$.

$$\begin{aligned} hMAP &= argmax P(h|D) \\ &= argmax \frac{P(D|h)P(h)}{PD} \end{aligned}$$

Da $P(D)$ unabhängig von h ist, kann es entfernt werden:

$$hMAP = argmax P(D|h)P(h)$$

Der oben vorgestellte Bayeslerner geht von einer starken Unabhängigkeit der Attribute aus. Daher wird dieser Lerner als der Naive Bayes Klassifizierer bezeichnet.

2.3. Multiklassenprobleme

Viele Klassifizierer, wie die Support Vector Machine, sind von ihrer Konstruktion her nur in der Lage binäre Probleme mit zwei Klassen zu behandeln. In der Praxis tauchen jedoch Probleme auf, die meistens mehr als zwei Klassen aufweisen. Diese Art von Problemen nennt man Multiklassenprobleme. Entweder ist ein Klassifizierer darauf ausgelegt diese direkt zu bearbeiten oder man macht Gebrauch von einer Binärisierung. Bei der Binärisierung wird ein Multiklassenproblem in eine Serie von binären Problemen umgewandelt. Das Lernverfahren, das dann schließlich darauf angewendet wird, nennt man Basislerner. Es gibt unterschiedliche Ansätze wie solch eine Umwandlung durchgeführt werden kann. Die Konzepte sind dabei unabhängig vom zugrunde liegenden Basislerner. Im folgenden werden einige Möglichkeiten vorgestellt. Das in Abbildung 2.9 dargestellte Multiklassenproblem soll dabei als Beispiel genutzt werden, um die unterschiedlichen Herangehensweisen zu veranschaulichen. Die verschiedenen Symbole stellen die unterschiedlichen Klassen eines Trainingsdatensatzes dar.

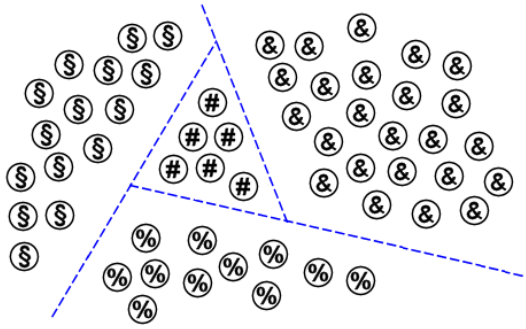

Abbildung 2.9.: Beispiel eines 4 Klassen Problems

2.3.1. Ungeordnete Binärisierung

Ein häufig eingesetztes Verfahren ist die ungeordnete oder one-against-all Binärisierung. Diese wandelt ein Multiklassenproblem mit c Klassen in c binäre Probleme $<i, j>$ um. Jedes dieser Probleme besteht aus allen Beispielen des ursprünglichen Datensatzes, dabei beinhaltet eine Seite nur die Instanzen einer Klasse und die andere Seite die restlichen. Für jedes Problem wird ein separater Klassifikator gelernt. Die Trainingsinstanzen der Klasse i werden dabei als positive Beispiele gelernt und die restlichen Traininginstanzen der Klassen j mit $j = 1, \ldots, c,\ j \neq i$, werden als negative Beispiele eingesetzt.

Abbildung 2.10 zeigt die vier binären Klassifizierungsprobleme, die aus dem Multiklassenbeispiel erzeugt werden. Die grün eingezeichneten Symbole repräsentieren die positiven Beispielen, die roten die negativen. Soll eine neue Instanz klassifiziert werden, so werden die gelernten Klassifizierer darauf angewendet. Es wird von den Klassifizierern überprüft, ob das Beispiel positiv oder negativ einzuordnen ist. Der Klassifizierer mit

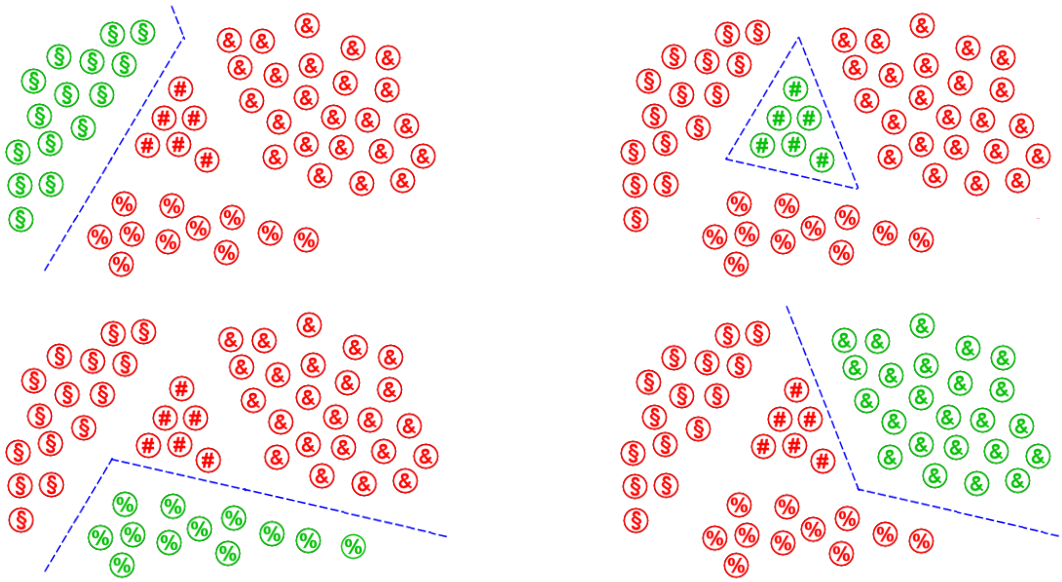

Abbildung 2.10.: Die ungeordnete Binärisierung teilt das 4 Klassen Problem in vier binäre Probleme auf.

einer positiven Zuordnung als Ergebnis entscheidet welche Klasse der Instanz zugeordnet werden soll.

2.3.2. Geordnete Binärisierung

Bei der geordneten Klassenbinärisierung wird das c Klassen Problem in $c-1$ binäre Probleme umgewandelt. Als positive Beispiele dienen hier die Instanzen der Klasse i mit $i = 1, \ldots, c-1$ und als negative die Instanzen der Klassen $j > i$.

Ist es bei der one-against-all Binärisierung so gewesen, dass jedes binäre Problem alle Beispiele des ursprünglichen Datensatzes beinhaltet, so gilt dies nicht für die geordnete Binärisierung. Die binärisierten Klassifzierungsprobleme werden hier in einer bestimmten Reihenfolge erstellt. Zunächst werden die Instanzen der ersten Klasse als positiv trainiert und die Instanzen der restlichen Klassen als negativ. Im zweiten Schritt werden die Instanzen der ersten Klasse entfernt und nur die Instanzen der zweiten Klasse als positiv trainiert. Die restlichen Instanzen der anderen Klassen werden als negativ gelernt. Im nächsten Schritt werden auch die Instanzen der zweiten Klasse entfernt und die nächste Klasse wird als positive Klasse trainiert. Diese Schritte werden solange fortgesetzt, bis für den (c-1)-ten Klassifizierer die Instanzen der vorletzten Klassen als positiv gelernt werden und die restlichen Instanzen als negativ trainiert werden. In diesem letzten Schritt beinhaltet die Menge der restlichen Klassen nur noch die Instanzen der einen letzten Klasse.

Die Reihenfolge, in der die Klassifizierer erstellt werden, entspricht gewöhnlich der Anzahl der Instanzen einer Klasse. Dies bedeutet die Klasse, die am häufigsten auftaucht,

wird zuerst separiert und die Klasse, die am seltensten auftaucht, zuletzt. Abbildung 2.11 zeigt wie die Trainingsphase des Multiklassenbeispiels abläuft.

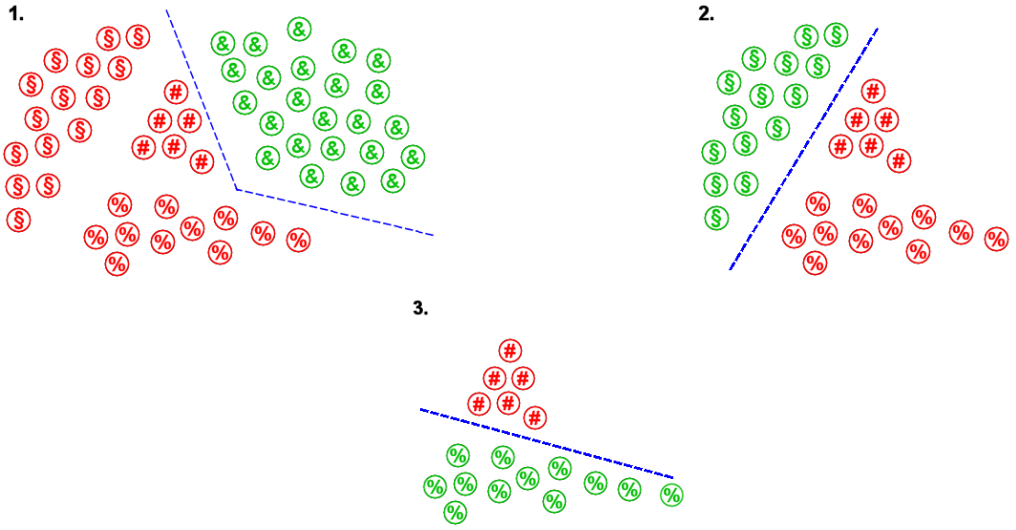

Abbildung 2.11.: Die geordnete Binärisierung erstellt binäre Probleme in einer bestimmten Reihenfolge.

Mit dieser Herangehensweise bietet die geordnete Binärisierung einen Vorteil in der Laufzeit gegenüber der one-against-all Binärisierung, da nicht alle Trainingsinstanzen für das Lernen hinzugezogen werden.

Auch während der Klassifikationsphase werden die Klassifizierer in der geordneten Reihenfolge aufgerufen. Soll eine neue Instanz klassifiziert werden, so wird zunächst überprüft, ob diese der am häufigsten auftauchenden Klasse zugehört. Ist dies der Fall wurde die zutreffende Klasse gefunden und der Klassifizierungsvorgang wird abgebrochen, ansonsten wird nach einer Zugehörigkeit für die nächsthäuftigste Klasse geprüft. Im schlechtesten Fall müssen alle Klassifzierer durchgegangen werden.

Wurden bei der one-against-all Klassifizierung, im schlimmsten Fall, alle c Klassifikatoren ausgewertet, um die passende Klasse zu finden, so wird bei diesem Verfahren die durchschnittliche Anzahl der Klassifikatorauswertungen reduziert.

Ein Verfahren, welches von dieser Art der Binärisierung Gebrauch macht ist der Regellerner RIPPER [3].

2.3.3. Paarweise Binärisierung

Die paarweise oder Round-Robin Binärisierung lernt für jedes einzelne Klassenpaar einen separaten Klassifizierer. Somit wird ein c Klassen Problem in $c(c-1)/2$ binäre Probleme zerlegt. Für jedes Klassenpaar $<i, j>$ mit $i = 1, \ldots, c-1$ und $j = i+1, \ldots, c$ wird ein Klassifizierer trainiert. Es ist zu beachten, dass die restlichen Klassen ungleich i, j von

den jeweiligen Klassifizierern ignoriert werden.

Abbildung 2.12 zeigt die Trainingsphase, die sich für das Multiklassenbeispiel ergibt. Jeder Klassifizierer $K(i, j)$ trainiert die Instanzen der Klasse i als positiv und die Instanzen der Klasse j als negativ. Die Anzahl der binären Probleme ist zwar höher als bei anderen Binärisierungsarten, jedoch besteht jedes Problem nur aus den Instanzen von zwei Klassen. Dadurch sollte es für den Klassifizierer einfacher sein eine Funktion zu finden, die die beiden Klassen voneinander separiert.

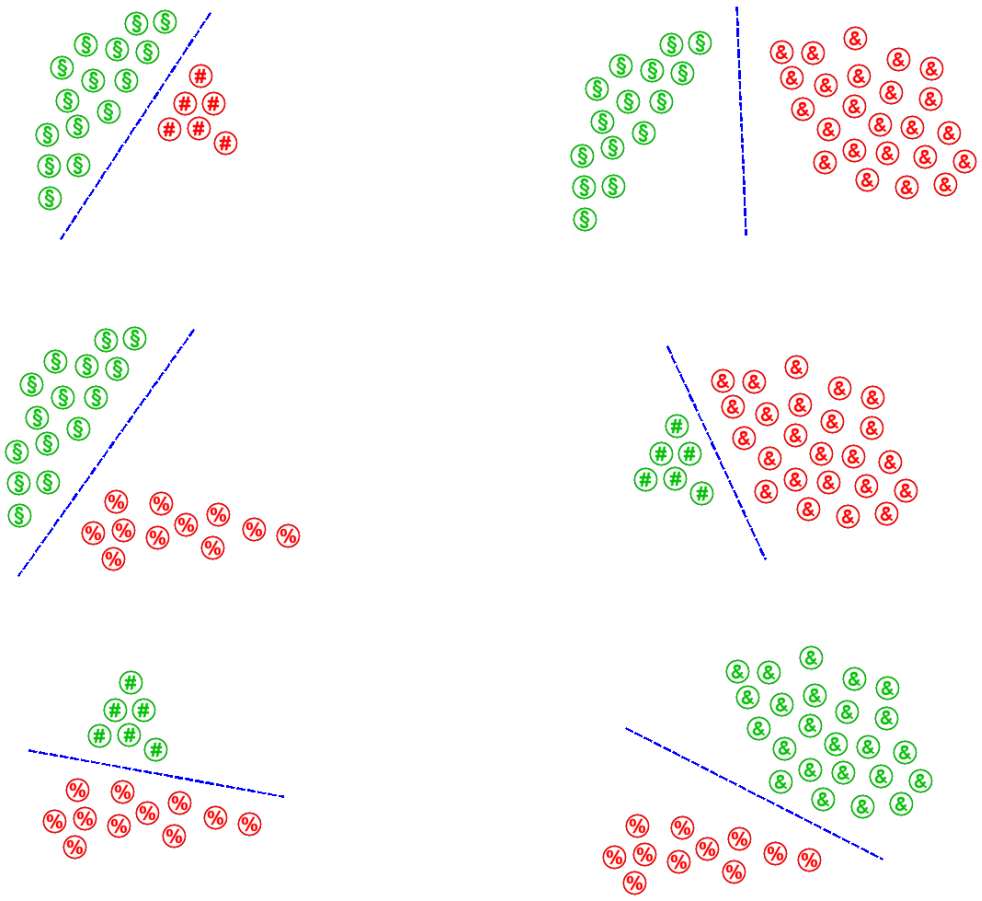

Abbildung 2.12.: Die paarweise Binärisierung erstellt ein binäres Problem für jedes Klassenpaar $<i, j>$.

Während der Klassifizierungphase entscheidet jeder Klassifizierer, ob eine neue Instanz der Klasse i oder der Klasse j zugehört. Dies kann man vergleichen mit Round-Robin Turnieren im Sport. Der Gewinnerklasse wird ein Punkt zugesprochen und am Ende wird die Klasse gewählt, welche die meisten Punkte bekommen hat. Diesen Vorgang nennt man *voting*. Sollte es dabei zu einem Gleichstand kommen, so wird die häufiger auftauchende Klasse bevorzugt oder es wird per Zufall entschieden.

Jeder der binären Klassifizier $K(i, j)$ muss eine Entscheidung treffen, ob eine neue Instanz der Klasse i oder j zugehört. Dadurch kommt es gezwungenermaßen zu Falschklassifizierungen bei Instanzen die weder der Klasse i noch der Klasse j zugehören. Im Prinizp könnte es vorkommen, dass auf diese Weise beim *voting* eine falsche Klasse die meisten Stimmen erhält. Jedoch geht man davon aus, dass die anderen Klassifizier, die die korrekte Klasse der neuen Instanz enthalten, diese Falschklassifizierung wieder wettmachen.

Zur Verdeutlichung ein weiteres Beispiel. Soll eine neue Instanz der Klasse x klassifiziert werden, so geht man idealerweise davon aus, dass die Klassifizierer in denen die Klasse x auftaucht ($<x, 1>$, $<x, 2>$, ..., $<x, c>$) diese auch richtig klassifizieren. Für eine andere Klasse ist es dann nicht möglich auf die gleiche oder höhere Anzahl von Stimmen zu kommen. Es wird als unwahrscheinlich angesehen, dass alle Klassifizier einstimmig für die selbe falsche Klasse abstimmen. Die Wahrscheinlichkeit für solch eine Situation steigt jedoch, wenn die Klassen untereinander abhängig sind. Auf diesen Fall wird im späteren Abschnitt 4.1 näher eingegangen.

Untersuchungen von Fürnkranz in [6] haben ergeben, dass die Round Robin Binärisierung im Vergleich zu der one-against-all oder geordneten Binärisierung zu signifikanten Verbesserungen führt.

3. Ordnung in der Klassifikation

In diesem Kapitel wird beschrieben, was man unter dem Begriff Ordnung in Bezug auf die Klassenwerte, versteht. Es wird erläutert, was es bei der Klassifikation mit geordneten Klassenwerten zu beachten gibt. Weiterhin werden zwei aus der Literatur bekannte Verfahren vorgestellt, welche die Klassenordnung in ihrer Vorhersage berücksichtigen. Das erste Verfahren macht Gebrauch von einer Umwandlung der Klassenwerte in numerische Werte; anschließend wird auf dieses Problem ein numerisches Prognoseverfahren, der sogenannte Regressionslerner, angewandt. Bei dem zweiten Verfahren handelt es sich um eine Binärisierungsmethode, die bei der Aufteilung eines Multiklassenproblems weitere relevante Informationen mitberücksichtigt.

3.1. Klassenordnung

Im Grundlagenteil wurde ein Überblick über vorhandene Lernverfahren gegeben. Es wurden verschiedene Binärisierungsarten vorgestellt, welche es den Lernverfahren ermöglichen Multiklassenprobleme zu bearbeiten. Dabei wurde ein Muliklassenproblem mit c Klassen in eine Reihe von binären Klassifikationsproblemen umgewandelt. In diesem Kontext wurde in Abschnitt 2.3.2 die geordnete Binärisierung erwähnt. Dort bezog sich die Ordnung auf die Reihenfolge, in der die verschiedenen Klassifizierer aufgerufen wurden. Dies ist nicht zu verwechseln mit der Ordnung der Klassenwerte, auf die sich der Begriff hier bezieht.

Unter der Ordnung der Klassenwerte oder Klassenordnung versteht man die Größenrelationen, die zwischen den einzelnen nominalen Klassenwerten vorherrschen. Klassenwerte, die solch eine Eigenschaft aufweisen, werden als ordinal bezeichnet. Dadurch lassen sich die Werte in einer Skala vom kleinsten bis zum größten Wert einreihen. Ein Beispiel für solch eine Klasse wäre die Temperaturangabe. Hohe Temperaturen bezeichnet man als *heiss*, mittlere als *warm* und niedrige als *kalt*. Daraus lässt sich die Klassenordnung $heiss > warm > kalt$ ableiten. Bei dieser Art von Klassifizierungsproblemen ist nicht nur darauf zu achten, die Klassifikationsgenauigkeit zu maximieren, sondern im Falle einer Fehlklassifizierung auch den Fehler zu minimieren, d.h. den Abstand der vorhergesagten Klasse zur tatsächlichen Klasse so gering wie möglich zu halten. Für eine Instanz der Klasse *heiss* ist die falsche Vorhersage *warm* genauer als die Vorhersage *kalt*. Auf diese Problemstellung wird im nächsten Kapitel ausführlicher eingegangen.

Definition 3.1 (Klassenordnung) *Sei V eine Klasse mit ordinalen Attributwerten v_1, v_2, v_3, ..., v_c. Man spricht von einer Klassenordnung, wenn zwischen den einzelnen Klassenwerten eine Grössenrelation existiert. So dass sich eine eindeutige Skala $v_1 < v_2 < v_3 < \ldots < v_c$ der Klassenwerte, vom kleinsten zum größten Element, ableiten lässt.*

In der Praxis gibt es viele Bereiche, die Probleme mit einer Klassenordnung aufweisen. Ein Beispiel für ein Problem aus dem Gebiet der Betriebswirtschaftlehre ist die Bewertung der Anleihen von Unternehmen [9]. Spezielle Codes stellen das Ausfallrisiko eines Schuldners dar. Diese Ratingcodes bestehen aus Buchstabenschlüssel und reichen von A bis D. Die Werte können der zunehmenden Ausfallwahrscheinlichkeit nach geordnet werden, angefangen von A - sichere Geldanlage, B - spekulativ, C - hochspekulativ bis D - Totalausfall sehr wahrscheinlich. Unterschiedliche Buchstabenkombinationen ermöglichen eine noch detailliertere Unterteilung dieser Ordnung. Die Ratingcodes geordnet nach aufsteigendem Ausfallrisiko sehen etwa so aus:

$$AAA < AA+ < AA < AA- < A+ < A < A- < BBB+ < BBB < BBB- < BB+ < BB < BB- < B+ < B < B- < CCC < CC < C < D$$

Ein anderer Bereich, in der die Ordnung bedeutend ist, ist das *collaborative filtering* [14]. Hier geht es darum, die von einer Person abzugebende Bewertung bezüglich eines Objektes vorherzusagen. Die Vorhersage basiert dabei auf Bewertungen, die die Person zu anderen Objekten hinterlassen hat oder auf Bewertungen, die von anderen Personen bezüglich des Objektes abgegeben wurden. Solche Systeme werden in vielen Online Shops wie z.B. amazon.de genutzt, um den Kunden speziell angepasste Empfehlungen darzustellen. Abbildung 3.1 zeigt die Werte einer solchen Bewertungsskala angefangen von einem Stern - Mag ich überhaupt nicht - bis zu fünf Sternen - gefällt mir sehr.

Abbildung 3.1.: Ordnung innerhalb einer Bewertungsskala

Bisher gibt es nur wenige Klassifizierer, die in der Lage sind die zusätzliche Information der Klassenordnung in ihrer Vorhersage miteinzubeziehen. Dadurch gehen wichtige Informationen verloren.

3.2. Geordnete Klassifikation mittels Regressionslerner

3.2.1. Regression

Bei den Datensätzen, die man für die Klassifikation nutzt, sind die Ausgangswerte der Instanzen feste nominale Klassenwerte. Zwar sind Klassifikatoren in der Lage Datensätze mit numerischen Attributen zu behandeln, jedoch muss der Ausgangswert (Klassenwert) ein nominaler Wert sein. Eine andere Art von Problemen stellen Datensätze mit einem numerischen Ausgangswert dar.

Bei der numerischen Vorhersage geht es nicht darum, einer Instanz die richtige Klasse zuzuordnen, sondern einen reelen Wert zu ermitteln, welcher dem tatsächlichen Wert möglichst nahe ist. Die Differenz zwischen zwei Werten kann durch Subtraktion ermittelt werden. Ein Beispiel für solch einen Datensatz ist in Tabelle 3.1 zu sehen. Es stellt die

relative Leistung eines CPUs in Abhängigkeit von verschiedenen relevanten Attributen dar.

Eine Möglichkeit den Zielwert zu bestimmen ist es, diesen als lineare Summe der Attribute darzustellen. Dazu ist eine entsprechende Gewichtung der Attribute notwendig. Diesen Vorgang nennt man Regression. Daraus ergibt sich die Regressionsgleichung. Für das Problem der CPU Leistung lässt sich etwa folgende Gleichung aufstellen:

$$PRP = -55.9 + 0.0489 \cdot MYCT + 0.0153 \cdot MMIN + 0.056 \cdot MMAX + 0.6410 \cdot CACH - 0.2700 \cdot CHMIN + 1.480 \cdot CHMAX$$

Cycle time *MYCT*	Min. mem. *MMIN*	Max. mem. *MMAX*	Cache *CACH*	Min. Chan. *CHMIN*	Max. Chan. *CHMAX*	Performance *PRP*
125	256	6000	256	16	128	198
29	8000	32000	32	8	32	269
125	2000	8000	0	2	14	52
480	512	8000	32	0	0	67
480	1000	4000	0	0	0	45

Tabelle 3.1.: Regressionsproblem - *CPU Performance*

3.2.2. Umwandlung in Regressionsprobleme

Die numerischen Intervallmengen der Regression stellen wie auch die geordneten Klassenwerte eine gewisse Ordnung dar. Dabei kann man jeden numerischen Wert als eine eigene Klasse betrachten. In diesem Fall gibt es auch hier nicht nur verschiedene Klassen zu berücksichtigen, sondern deren Abstände zu einander sind ebenfalls relevant.

Ein möglicher Ansatz, um geordnete Klassenprobleme zu bearbeiten, ist es, diese als Regressionsprobleme zu behandeln. Dieses Verfahren wird unter anderem in [8] und [7] vorgestellt. Die Klassen einer Ordnungsskala mit der Relation $A<B<C<D<E$ werden in numerische Werte umgewandelt. Eine mögliche Umwandlung wäre hier z.B. die einfache Abbildung auf ganze Zahlen wie $A = 1$, $B = 2$, $C = 3$, $D = 4$ und $E = 5$. Da die Ausgangswerte der Instanzen nun numerische Werte darstellen, kann man einen gewöhnlichen Regressionslerner auf dieses Problem anwenden. Nachdem eine Vorhersage gemacht wurde, wird der zurückgegebene numerische Wert wieder in den entsprechenden nominalen Wert umgewandelt. Diese Umwandlungungen nennt man *pre-* und *post-processing*.

In [8] werden für das *pre-processing* verschiedene Transformationsarten vorgestellt. Die Idee dahinter ist, dass Zahlen grundsätzlich verschiedene Mengengaben darstellen können. Es wird unter anderem zwischen den folgenden Typen unterschieden: Menge und Aufzählung (kann keinen negativ Wert annehmen), Rang (z.B. 1 = kleinste, 2 = zweitkleinste, ...), Note (geordnete Kennzeichen, wie A, B, C, D, E). Für die verschiedenen Typen von Mengenangaben werden in Tabelle 3.2 Pre-Processing Umwandlungen vorgestellt.

Eine mögliche *post-processing* Methode ist das Runden. Die vorhergesagte reelle Zahl wird, auf die der ursprünglichen Klasse am nächstgelegenen Zahl, gerundet. Dies muss nicht die nächste ganzzahlige Zahl sein, da die Klassen auch nicht ganzzahlig sein können.

Name	Formel
Rohdaten	kein pre-processing ($tk = k$)
Aufzählung	$tk = \log(k + 1 - \min(Klassen))$
Rang	$tk = \log((k - 1/3)/(N - c + 2/3))$, mit $N = max(Klassen)$
Note	$tk = (\phi(P) - \phi(p))/(P - p)$, mit $\phi(x) = x\log(x) + (1 - x)\log(1 - x)$, $P =$ Anteil der beobachteten Klassenwerte $\geq k$, $p =$ Anteil der beobachteten Klassenwerte $> k$

Tabelle 3.2.: *Pre-processing* Umwandlungen für Regressionslerner (k = ursprünglicher Klassenwert, tk = transformierter Klassenwert) (Quelle [8])

Ein Nachteil dieses Ordnungsverfahrens besteht darin, dass es nur in Verbindung mit dem Regressionsmodell angewandt werden kann.

3.3. Ordinale Klassifikation von Frank und Hall

Die ordinale Klassifikation von Frank und Hall, welche in [4] vorgestellt wurde, ist ein Verfahren, welches die Klassenordnung bei ihrer Vorhersage miteinbezieht. Vom Ansatz her ähnelt sie etwas einer one-against-all Binärisierung. Ein c Klassen Problem wird in $c-1$ binäre Probleme zerlegt anhand dessen die einzelnen Klassifizierer trainiert werden.

Abbildung 3.2 zeigt wie das Multiklassenproblem A^* mit den vier Werten v_1, v_2, v_3 und v_4 in drei binäre Attribute umgewandelt wird. Innerhalb des Multiklassenproblems gilt die Ordnung $v_1 < v_2 < v_3 < v_4$. Als Resultat erhält man die binären Probleme ${A_1}^*$, ${A_2}^*$ und ${A_3}^*$.

Für jedes der Probleme gilt, dass die Klasse(n) aus der linken Menge in der Ordnungsskala niedriger einzuordnen (ist) sind als die Klasse(n) aus der rechten Menge. Die Probleme stellen eine Grössenrelation dar. Entweder ist einer Instanz ein Klassenwert aus der kleineren Klasse zuzuordnen oder ein Wert aus der grösseren Klasse. Für ${A_1}^*$ gilt somit die Relation $X > v_1$, wobei X die Zielklasse der getesteten Instanz darstellt. Der erste Fall ist, dass eine Instanz den Klassenwert v_1 hat und die Aussage trifft nicht zu. Die Aussage $X > v_1$ ist also nicht erfüllt und somit 0. Der zweite Fall ist, dass die Instanz einen Klassenwert von v_2, v_3, oder v_4 hat. Für diese Fälle trifft die Aussage $X > v_1$ zu, also bekommt sie den Wert 1.

Wie diese Binärisierung benutzt wird, um den Klassifzierer zu trainieren wird anhand folgendem Beispiel verdeutlicht. Betrachtet wird ein Multiklassenattribut mit Ordnung, welche die Außentemperatur angibt. Es besteht aus den drei Werten *kalt*, *warm* und *heiss*. Diese lassen sich in die Ordnungsskala $kalt < warm < heiss$ einreihen. Die Trainingsphase beginnt damit, dass der eigentliche Datensatz in binäre Datensätze umgewandelt wird (in diesem Fall sind es zwei). In Abbildung 3.3 ist diese Transformation dargestellt. Mithilfe des linken Datensatzes kann die Aussage $X > kalt$ und mit dem rechten die Aussage $X > warm$ getroffen werden. X stellt dabei den Zielwert dar.

Die gelernten Klassifizierer sind in der Lage folgende Aussagen zu treffen. Nicht immer

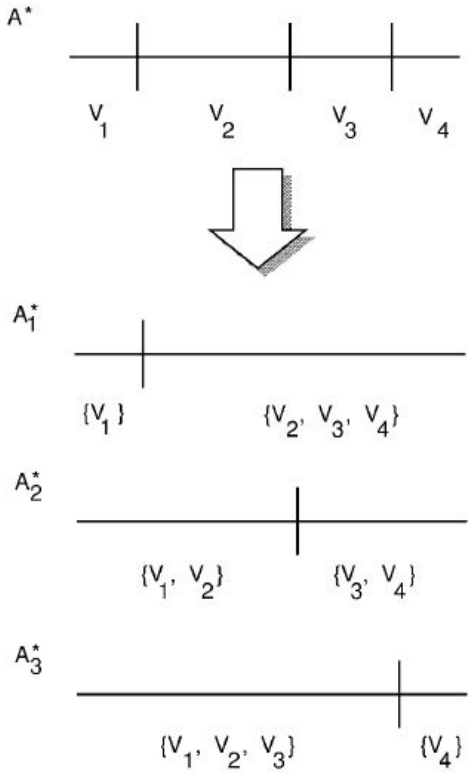

Abbildung 3.2.: Umwandlung eines Multiklassenproblems mit vier Klassen in drei binäre Probleme (Quelle [4]).

lässt sich eindeutig eine Klasse zuweisen.

$$(X > kalt) = 0 \rightarrow X = kalt$$
$$(X > kalt) = 1 \rightarrow X = warm|heiss$$
$$(X > warm) = 0 \rightarrow X = kalt|warm$$
$$(X > warm) = 1 \rightarrow X = heiss$$

Im nächsten Schritt folgt die Klassifizierung. Um einer neuen Instanz ihren Klassenwert zuzuordnen, wird mithilfe der $c - 1$ binären Klassifizierer für jede mögliche Klassenzugehörigkeit die Wahrscheinlichkeit berechnet. Das heisst, um in den obigen Beispiel die Wahrscheinlichkeit zu berechnen, dass die neue Instanz der Klasse *heiss* zuzuordnen ist, benötigt man den Klassifizier der die Aussage $X > warm$ überprüft. Mithilfe diesem kann man den gesuchten Wert $Pr(X > warm)$ berechnen. Das Berechnen der Wahrscheinlichkeit für die Klasse *kalt* erweist sich als etwas aufwendiger. Da der Klassifizierer für die Aussage $X > kalt$ nur die Wahrscheinlichkeit für die Klassen größer als *kalt* be-

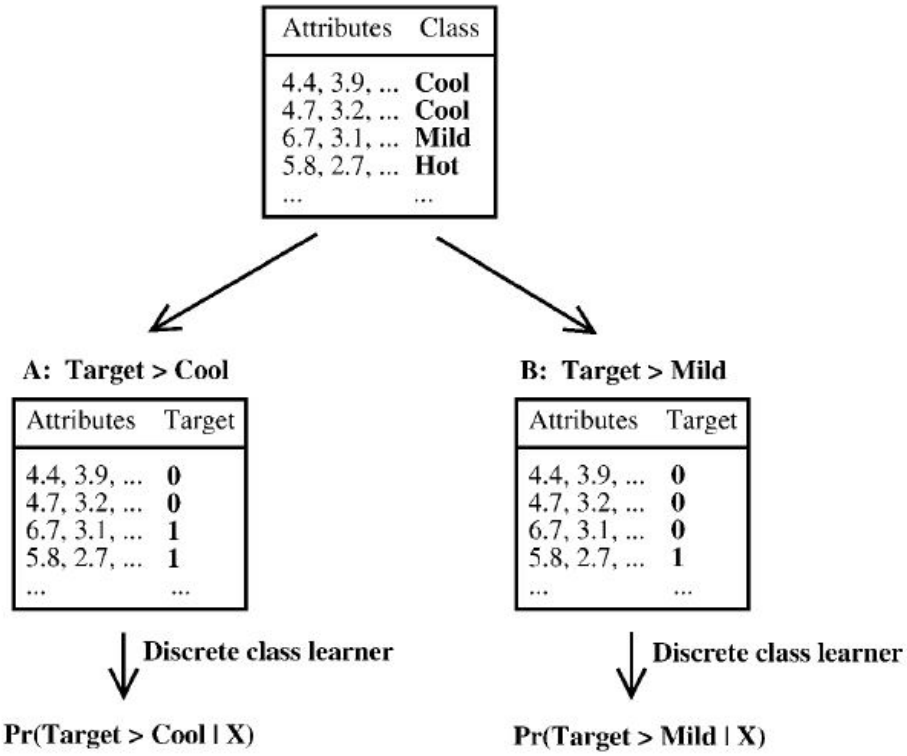

Abbildung 3.3.: Trainingsphase der ordinalen Klassifikation (Quelle [4])

rechnen kann, muss dieses Ergebnis invertiert werden. Der gesuchte Wert berechnet sich dann mit $1 - Pr(X > kalt)$.

Diese beiden Werte sind Spezialfälle, da es sich hier um den kleinsten und den grössten Wert des geordneten Attributes handelt. Will man die Wahrscheinlichkeit für eine andere Klasse innerhalb der Ordnung anfragen, so müssen mehrere Klassifizierer in Betracht gezogen werden. Der Wert von $warm$ wird durch zwei Klassifizierer bestimmt: $Pr(warm) = Pr(X > kalt) \times (1 - Pr(X > warm))$. Die Schnittmenge von $\{warm, heiss\}$ und $\{kalt, warm\}$ ergibt $warm$. Verallgemeinert lassen sich die Formeln folgendermaßen darstellen:

$$
\begin{aligned}
Pr(V_1) &= 1 - Pr(X > V_1) \\
Pr(V_i) &= Pr(X > V_{i-1}) \times (1 - Pr(X > V_i)),\ 1 < i < k \\
Pr(V_k) &= Pr(X > V_{k-1})
\end{aligned}
$$

Nachdem nun für jede der c Klassen die Wahrscheinlichkeit der korrekten Zuordnung berechnet wurde, kann eine Vorhersage darüber getroffen werden, auf welche Klasse die neue Instanz abzubilden ist. Die Klasse, bei der die höchste Wahrscheinlichkeit ermit-

telt wurde, wird der Instanz zugeteilt. Das Verfahren funktioniert in Zusammenhang mit allen Basislernern, die in der Lage sind Wahrscheinlichkeitswerte bezüglich ihrer Klassenzugehörigkeit zurückzugeben.

4. Paarweise geordneter Klassifizierer

In Grundlagenteil wurde der paarweise Klassifizierer vorgestellt. Dieses Verfahren macht in ihrer Vorhersage nicht Gebrauch von der Klassenordnung. In vorhergehenden Arbeiten von Fürnkranz in [5] hatten direkte Vergleiche mit der ordinalen Klassifikation von Frank und Hall ergeben, dass der herkömmliche Round-Robin Klassifizierer genauso gute Ergebnisse liefert, wie die speziell der Klassenordnung angepasste Methode. Auf Basis dieser Ergebnisse enstand die Idee, den paarweisen Klassifizierer derart zu erweitern, dass Informationen bezüglich der Klassenordnung in ihre Vorhersage miteinbezogen werden. Es ist zu überprüfen, ob diese Optimierung zu einer Leistungsverbesserung führt. In diesem Kapitel wird die Funktionsweise des erweiterten Verfahrens vorgestellt und analysiert. Um die beiden Klassifizierer besser voneinander zu unterscheiden wird die Bezeichnung Round-Robin Klassifizierer explizit für den paarweisen Klassifizierer, welcher ohne Klassenordnung arbeitet, benutzt und die Bezeichnung paarweiser geordneter Klassifizierer für die an die Ordnung angepasste Version.

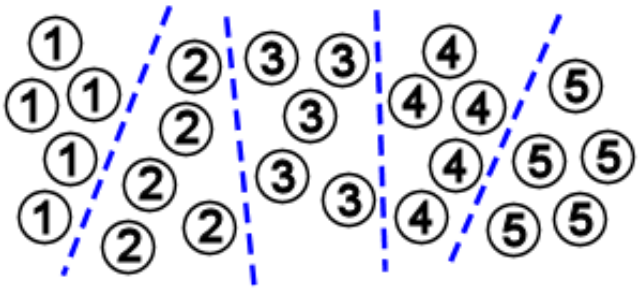

Abbildung 4.1.: Multiklassenproblem mit Klassenordnung

4.1. Probleme des Round-Robin Klassifizierers bei geordneten Klassenwerten

Zunächst einmal soll darauf eingegangen werden, in welchen Fällen die Anwendung des Round-Robin Klassifizierers problematisch sein kann. Der Round-Robin Klassifizierer lernt für jedes vorhandene Paar von Klassen $<i, j>$ mit $i = 1, \ldots, c-1$ und $j = i+1, \ldots, c$ einen Klassifizierer. Somit wird ein c Klassen Problem in $c(c-1)/2$ binäre Probleme aufgeteilt, anhand dieser die jeweiligen Klassifizierer trainiert werden. Für den Klassifizierer $K(i, j)$ werden lediglich die Instanzen der Klasse i als positiv und die Instanzen der Klasse

j als negativ zugeordnet. Die Instanzen der anderen Klassen werden von diesem Klassifizierer nicht beachtet. Demnach wird eine vorhandene Klassenordnung grundsätzlich ignoriert. Soll nun einer Instanz der Klasse x mit $x \neq i, j$ ein Wert zugeordnet werden, so findet gezwungenermaßen eine Falschklassifizierung statt. Da die Klasse x jedoch während der Trainingsphase für $K(i, j)$ nicht berücksichtigt wurde, kann es durchaus vorkommen, dass die weiter entferntere Klasse gewählt wird.

Die Klasse x lässt sich eindeutig in die Klassenordnung platzieren, enweder ist sie kleiner i, grösser j oder liegt zwischen i und j. Für den Fall, dass x kleiner i ist, ist die Vorhersage i zu bevorzugen, da $x < i < j$ und somit i einen kleineren Abstand zu x hat als j. Für den Fall, dass x grösser j ist, ist entsprechend die Vorhersage j zu bevorzugen, da dann $i < j < x$ gilt. Liegt x zwischen i und j so kann keine Aussage darüber getroffen werden welche Vorhersage besser ist.

Der Abstand zwischen der vorhergesagten und der tatsächlichen Klasse wird als Ordnungsdistanz bezeichnet. Je niedriger die Ordnungsdistanz ist, desto genauer ist die Vorhersage. Ordnungsdistanz 0 stellt eine korrekte Klassifizierung dar.

Definition 4.1 (Ordnungsdistanz) *Sei $V = \{v_1, v_2, v_3, \ldots, v_c\}$ die Menge der Klassen mit der Ordnung $v_1 < v_2 < v_3 < \ldots < v_c$. v_a stellt die tatsächliche Klasse einer Instanz dar. v_p stellt die von einem Klassifizierer vorhergesagte Klasse dar. Die Ordnungsdistanz $x = |v_a - v_p|$ stellt den Abstand zwischen der tatsächlichen Klasse und der vorhergesagten Klasse dar.*

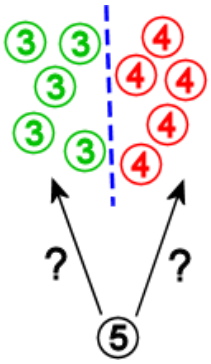

Abbildung 4.2.: Schwierigkeiten des Round-Robin Klassifizierers bei Problemen mit Klassenordnung

Das Problem soll anhand einer Grafik nochmal veranschaulicht werden. Abbildung 4.1 zeigt ein 5 Klassen Problem, indem eine Klassenordnung vorherrscht. Die Klassenordnung ist in der Abblidung durch die Zahlen gekennzeichnet. Die kleinste Klasse wird als 1 und die größte Klasse als 5 dargestellt. In dem Beispiel gilt somit die Ordnung $1 < 2 < 3 < 4 < 5$.

Wird jetzt, wie in Abbildung 4.2 dargestellt, der Round-Robin Klassifizierer $K(3, 4)$ trainiert so werden die Instanzen der Klasse 3 als positiv und Instanzen der Klasse 4 als

negativ eingestuft. Da nur anhand dieser beiden Klassen trainiert wird ist der Klassifizierer während der Entscheidungsphase nicht in der Lage Informationen bzgl. der Ordnung zu berücksichtigen. Es kann passieren, dass eine neue Instanz, deren tatsächliche Klasse 5 ist und somit der nächstgrößere Nachbar von 4 ist, von diesem Klassifizierer fälschlicherweise den Wert 3 zugeordnet bekommt. Die so enstandene Klassifikation mit Ordnungsdistanz 2 ist schlechter als die andere Alternative, welche eine genauere Ordnungsdistanz von 1 erzeugt hätte.

4.2. Aufbau

Im folgenden soll die Funktionsweise des paarweise geordneten Klassifizierers vorgestellt werden. Das Ziel ist es bei einer Falschklassifizierung die Klasse mit der niedrigsten Ordnungsdistanz auszusuchen. Diese Optimierung soll letzendlich zu einer Erhöhung der Genauigkeit führen. Wie auch bei der einfachen paarweisen Methode wird für jedes Klassenpaar ein Klassifikator $K(i, j)$ erlernt. Somit entstehen auch hier $c(c-1)/2$ Klassifikatoren die trainiert werden. Neben den eigentlichen beiden Klassen i und j werden nun auch weitere Klassen berücksichtigt. Die Instanzen, von denen sich bestimmen lässt, dass sie in der Klassenordnung näher an der Klasse i sind als an Klasse j werden als positiv eingestuft und für den umgekehrten Fall, d.h. die Instanzen näher an der Klasse j werden als negativ eingestuft.

Definition 4.2 (paarweise geordnete Binärisierung) *Sei ein geordnetes Klassenproblem mit n Klassen gegeben. Sei* $V = \{v_1, v_2, v_3, \ldots, v_c\}$ *die Menge der Klassen mit Ordnung* $v_1 < v_2 < v_3 < \ldots < v_c$. *Die paarweise geordnete Binärisierung wandelt ein geordnetes Klassenproblem in* $c(c-1)/2$ *binäre Klassenprobleme* $<i, j>$ *mit* $i = 1, \ldots, c-1$, $j = i+1, \ldots, c$ *und* $i \neq j$ *um. Für jedes binäre Klassenproblem wird ein Klassifizierer* $K(i, j)$ *gelernt. Alle Instanzen der Klassen x, mit* $x \leq i$, *werden als positiv trainiert und alle Instanzen der Klassen y, mit* $y \geq j$ *werden als negativ trainiert.*

Es ist zu beachten, dass die Abstände zwischen den einzelnen Klassen nicht bekannt sind. Es wird nicht davon ausgegangen, dass diese gleich verteilt sind. Für Instanzen, die zwischen der Klasse i und j liegen kann daher keine ordnungsrelevante Aussage getroffen werden.

Anhand des vorangegangenen Beispiels wird der erwartete Vorteil des paarweise geordneten Klassifizierers veranschaulicht. In Abbildung 4.3 ist der geordnete paarweise Klassifizierer für das Klassenpaar $<1, 4>$ dargestellt. Wie man erkennen kann werden in der Trainingsphase nicht nur die direkten Instanzen der Klassen 1 und 4 berücksichtigt, sondern auch die Instanzen der Klasse grösser 4, in diesem Falle ist dies lediglich die Klasse 5. Die Trainingsinstanzen der Klasse 5 werden nun, ebenso wie die der Klasse 4, als negativ trainiert. Da Klasse 1 schon die kleinste Klasse aus der Klassenmenge ist, werden neben Instanzen der Klasse 1 keine zusätzlichen Beispiele als positiv trainiert. Soll nun eine neue Instanz, die den tatsächlichen Klassenwert 5 hat, klassifiziert werden, so ist der Klassifizierer $K(1, 4)$ in der Lage die Klasse mit der niedrigeren Ordnungs-

distanz vorherzusagen. Da $K(1,4)$ zusätzlich mit den Instanzen der Klasse 5 trainiert wurde kann die genauere Vorhersage der Klasse 4 getroffen werden.

Es sei darauf hingewiesen, dass $K(1,4)$ keine ordnungsrelevanten Aussagen bzgl. den Klassen 2 und 3 treffen kann. Da diese beiden Klassen sich zwischen 1 und 4 befinden und die einzelnen Klassenabstände unbekannt sind, kann nicht gesagt werden, welche näher an der Klasse 1 oder 4 liegt. Auch wenn weniger Klassen zwischen 1 und 2 liegen als zwischen 2 und 4 kann es trotzdem durchaus sein, dass der tatsächliche Abstand d zwischen 1 und 2 grösser ist als der zwischen 2 und 4. Es gilt nicht $d(1,2) = d(2,3) = d(3,4)$. Die Abstände sind unbekannt und können unterschiedlich gross sein. Daher wird dieser Fall nicht in Betracht gezogen.

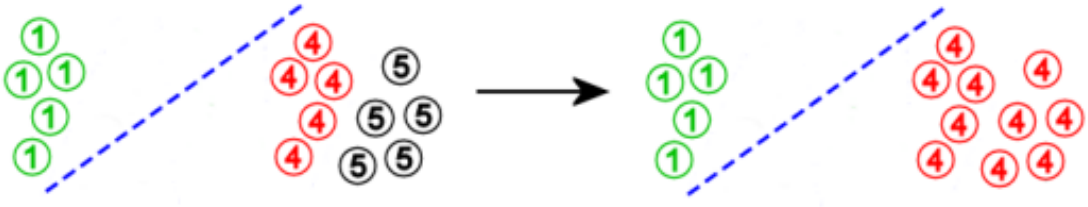

Abbildung 4.3.: Trainingsphase des Klassifizierers $K(1,4)$ - die Instanzen der Klasse 5 werden ebenso als Klasse 4 trainiert

Abbildung 4.4 zeigt den Klassifizierer $K(3,4)$. Die Klassen kleiner 3 sind 1 und 2, die Klasse grösser 4 ist 5. In diesem Fall werden während der Trainingsphase alle vorhandenen Klassen in Betracht gezogen. Das Beispiel zeigt wie viel aufwendiger der paarweise geordnete Klassifizierer unter Umständen werden kann. Im nächsten Kapitel wird genauer darauf eingegangen.

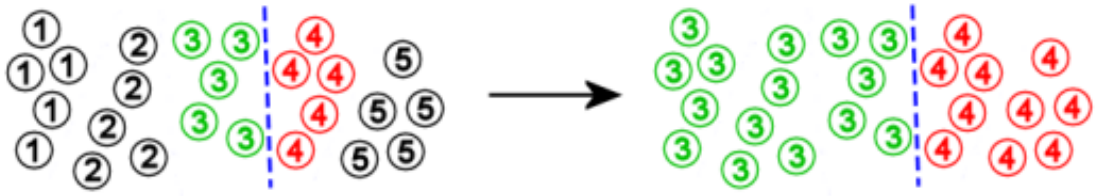

Abbildung 4.4.: Trainingphase des Klassifizierers $K(3,4)$ - durch die Hinzunahme der weiteren Intanzen wird die Trainingsmenge sehr groß

Als Dekodierungsart wird wie auch beim Round-Robin Klassifizier das *voting* benutzt. Jeder der binären Klassifizierer gibt eine Stimme ab welche Klasse zu bevorzugen ist. Die Klasse mit den meisten Stimmen wird letzendlich zugewiesen.

4.3. Laufzeit

In diesem Abschnitt wird der Aufwand des paarweise geordneten Verfahrens analysiert. Hierfür bedarf es einer eingehenden Untersuchung der Anzahl der benötigten Trainingsinstanzen. Da das geordnete Verfahren eine Erweiterung des herkömmlichen Round-Robin Verfahren ist soll zunächst näher auf den Aufwand des Round-Robin Verfahrens eingegangen werden.

Der Round-Robin Klassifikator wandelt ein Problem mit c Klassen in $c(c-1)/2$ binäre Probleme um. Ein binäres Problem besteht aus dem Klassenpaar $<i,j>$. Um zu ermitteln, wieviele Instanzen insgesamt für die Trainingsphase benötigt werden, betrachtet man die einzelnen Klassifizierer. Die Instanzen einer Klasse i werden einmal gegen jede andere Klasse gepaart. Somit taucht eine Instanz genau einmal in jeder der $c-1$ Klassenpaare auf. Da der ursprüngliche Trainingsdatensatz aus n Beispielen besteht ist die gesamte Anzahl der Instanzen $(c-1) \cdot n$.

Das paarweise geordnete Verfahren im Vergleich hierzu, wandelt auch ein c Klassen Problem in $c(c-1)/2$ binäre Probleme um. Jedoch besteht hier ein binäres Problem nicht nur aus den beiden Beispielen des Klassenpaares $<i,j>$, sondern weitere ordnungsrelevante Klassen werden in der Trainingsphase berücksichtigt, was zu einer erhöhten Anzahl der Trainingsinstanzen führt. Wieviele Instanzen dadurch genau benötigt werden, soll im folgenden genau ermittelt werden. Dabei kann man zwischen den beiden Fällen unterscheiden, dass alle Klassen die gleiche Anzahl von Instanzen besitzen oder eben nicht. Zunächst soll die Berechnung des einfachereren Falls, dass die Instanzen gleichverteilt sind, vorgestellt werden.

4.3.1. Klassen besitzen gleiche Instanzanzahl

Ausgangsproblematik ist, dass jede Klasse aus gleich vielen Instanzen besteht. Unter den Klassenpaaren existieren $c-1$ Klassenpaare, welche direkt miteinander benachbart sind $(<v_1,v_2>, <v_2,v_3>, \ldots, <v_{c-1},v_c>)$.

Da der Klassifizierer $K(i,j)$ die Instanzen kleiner Klasse als i und größer Klasse als j miteinbezieht besteht der Trainingsdatensatz, für diese Art von Problemen, aus der Menge aller Trainingsinstanzen n. Bei $c-1$ solchen Klassifizieren kommt man auf eine Gesamtanzahl von $(c-1) \cdot n$ Instanzen.

Als nächstes sind die Klassenpaare zu untersuchen, welche genau zwei Klassen voneinander entfernt sind $(<v_1,v_3>, <v_2,v_4>, \ldots, <v_{c-2},v_c>)$. Solche Paare tauchen $(c-2)$-mal auf. Unter Einbeziehung der Instanzen kleinerer und grösserer Klassen, besteht die Trainingsmenge der entsprechenden Klassifizierer $K(i,j)$ aus $\frac{c-1}{c} \cdot n$ Trainingsinstanzen. Bei $c-2$ dieser Klassifizierer kommt man insgesamt also auf $(c-2) \cdot \frac{c-1}{c} \cdot n$ Instanzen.

Für die restlichen Klassenpaare kann man analog verfahren. $c-3$ Klassenpaare mit zwei Klassen Abstand nutzen insgesamt $\frac{c-2}{c}$ vom kompletten Trainingsdatensatz n. Insgesamt dann: $(c-3) \cdot \frac{c-2}{c} \cdot n$.

Dieser Zusammenhang lässt sich mittels einer Summenformel folgendermassen darstellen:

$$
\begin{aligned}
&\sum_{i=1}^{c-1} \frac{i}{c} n \cdot (c-i) \cdot 2 \\
=&\sum_{i=1}^{c-1} 2n \cdot \left(i - \frac{i^2}{c} \right) \\
=&2n \cdot \left(\frac{(c-1)c}{2} - \frac{1}{c} \cdot \frac{(c-1)c(2c-1)}{6} \right) \\
=&2n \cdot \left(\frac{(c-1)\,(3c-(2c-1))}{6} \right) \\
=&\frac{n}{3} \cdot (c-1)\,(c+1) \\
=&\frac{n}{3} \cdot \left(c^2-1\right)
\end{aligned}
$$

In Abbildung 4.5 ist dazu nochmal ein Beispiel für einen Datensatz mit sechs Klassen zu sehen. Jede Zeile stellt einen der 15 Klassifizierer dar. Jede Klasse besitzt die gleiche Anzahl von $\frac{n}{6}$ Instanzen.

K_1, K_6, K_{10}, K_{13} und K_{15} nutzen den kompletten Trainingsdatensatzes n: $5 \cdot n$
K_2, K_7, K_{11} und K_{14} nutzen 5/6 des Trainingsdatensatzes n: $4 \cdot 5/6 \cdot n$
K_3, K_8 und K_{12} nutzen 4/6 des Trainingsdatensatzes n: $3 \cdot 4/6 \cdot n$
K_4 und K_9 nutzen 3/6 des Trainingsdatensatzes n: $2 \cdot 3/6 \cdot n$
K_5 nutzt 2/6 des Traininsdatensatzes n: $1 \cdot 2/6 \cdot n$

$$
\begin{aligned}
\text{Als Summe erhält man somit : } \quad & 5 \cdot n + 4 \cdot \frac{5}{6} n + 3 \cdot \frac{4}{6} n + 2 \cdot \frac{3}{6} n + 1 \cdot \frac{2}{6} n \\
&= \frac{35}{3} n = \frac{n}{3} \cdot \left(6^2 - 1\right)
\end{aligned}
$$

4.3.2. Klassen besitzen unterschiedliche Instanzanzahl

Kommen die Instanzen in jeder Klasse nicht gleich häufig vor, erweist sich die Berechnung der verwendeten Trainingsinstanzen als schwieriger. Anders als bei dem vorherigen Fall kann man nicht davon ausgehen, dass man jede Klasse als einen Anteil $\frac{1}{c}$ betrachten kann und diese entsprechend aufsummiert. Stattdessen muss überprüft werden, wie oft jede Klasse tatsächlich auftaucht. In der folgenden Berechnung wird auch gezeigt, dass bei dem paarweisen geordneten Klassifizierer die "äusserern Klassen" häufiger auftauchen, als die "Inneren". Dies beruht auf der Tatsache, dass diese Klassen häufiger als Nebenklassen auftauchen. Wie sich die Verteilung der Klassen im Hinblick auf die Klassengröße verhält wird im Folgenden aufgeführt.

Zur Verdeutlichung soll die Berechnung der verwendeten Trainingsinstanzen anhand eines Beispiels verfolgt werden. Als Beispiel dient wieder die in Abbildung 4.5 gezeigte

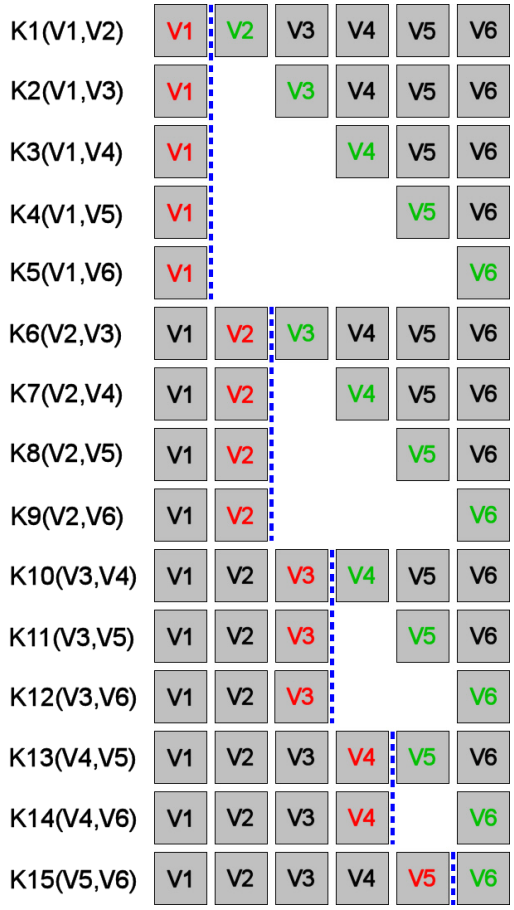

Abbildung 4.5.: Komplette Trainingsmenge eines paarweise geordneten Klassifizierers für ein 6 Klassen Problem

Menge mit sechs geordneten Klassenwerten. Im ersten Schritt gibt es, wie auch beim vorherigen Ansatz, die 5 (entspricht $c-1$ für den allgemeinen Fall) Klassifizierer, von denen jeder Gebrauch von allen Instanzen des Trainingsdatensatzes macht. Abbildung 4.3.2 zeigt die fünf daraus resultierenden vollständigen Trainingsinstanzmengen in hellgrau.

Für die weiteren Fälle werden jeweils die Trainingsinstanzen von zwei Klassifizieren zusammengefasst, sodass man jeweils eine vollständige Menge von Trainingsinstanzen n und weitere restliche Klassen erhält. Im nächsten Schritt betrachtet man sich die Klassifikatoren K_2, K_3 und K_4, in denen die Instanzen der Klasse v_1 alleine als negative Beispiele trainiert werden. Diese kombiniert man mit den Klassifikatoren K_9, K_{12} und K_{14}, in denen die Instanzen der Klasse v_6 alleine als positive Beispiele trainiert werden. Die Trainingsinstanzen der sechs Klassifikatoren lassen sich so miteinander kombinieren, dass man drei (entspricht $c-3$ für den allgemeinen Fall) komplette Datensätze erhält. Als Rest von Klassifikator K_9, K_{12} und K_{14} bleiben die Randklassen v_1 und v_6. Zusammen mit den Klassifizier K_5, welcher ebenso aus den Randklassen v_1 und v_6 besteht,

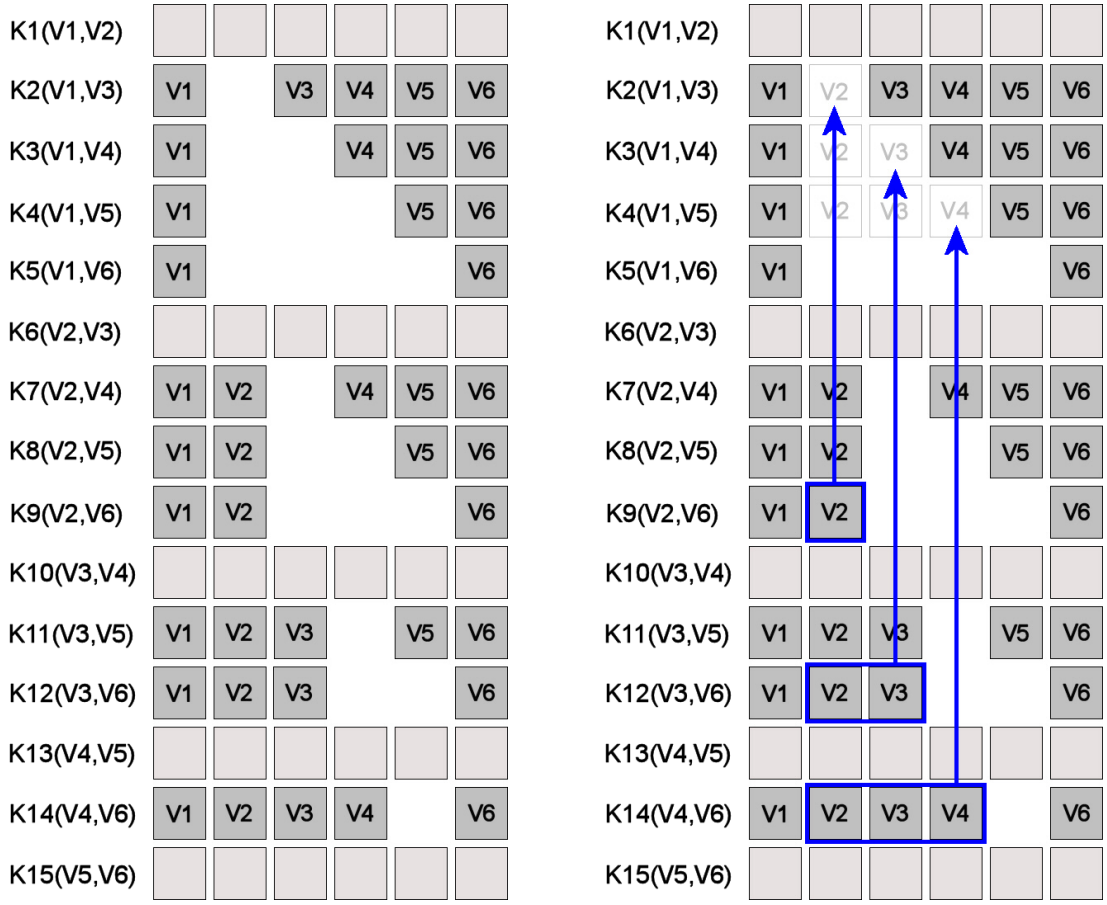

Abbildung 4.6.: Die Trainingsmengen der Klassifikatoren werden so kombiniert, dass man wieder vollständige Trainingsmengen n erhält.

ergeben das jeweils viermal (entspricht $c-2$ für den allgemeinen Fall) die Instanzen der Randklassen v_1 und v_6.

Als nächstes lassen sich auf die gleiche Weise Klassifkator K_7 und K_{11} kombinieren. Man erhält eine (entspricht $c-5$ für den allgemeinen Fall) vollständige Trainingsinstanzmenge n und einmal die äußersten vier Klassen des Datensatzes. Zusammen mit Klassifikator K_8, welcher ebenso aus den äußersten vier Klassen besteht, kommt man auf insgesamt zweimal (entspricht $c-4$ für den allgemeinen Fall) die Instanzen der äußersten vier Klassen.

Für die allgemeine Berechnung der Trainingsinstanzen lässt sich folgende Formel aufstellen:

$$\sum_{i=1}^{c-1}\left(2\cdot\left(\frac{i}{2}-\left\lfloor\frac{i}{2}\right\rfloor\right)\cdot(i\cdot n)+2\cdot\left(\frac{i+1}{2}-\left\lfloor\frac{i+1}{2}\right\rfloor\right)\cdot(c-i)\cdot\sum_{j=1}^{i/2}(\#(v_i)+\#(v_{c+1-j}))\right)$$

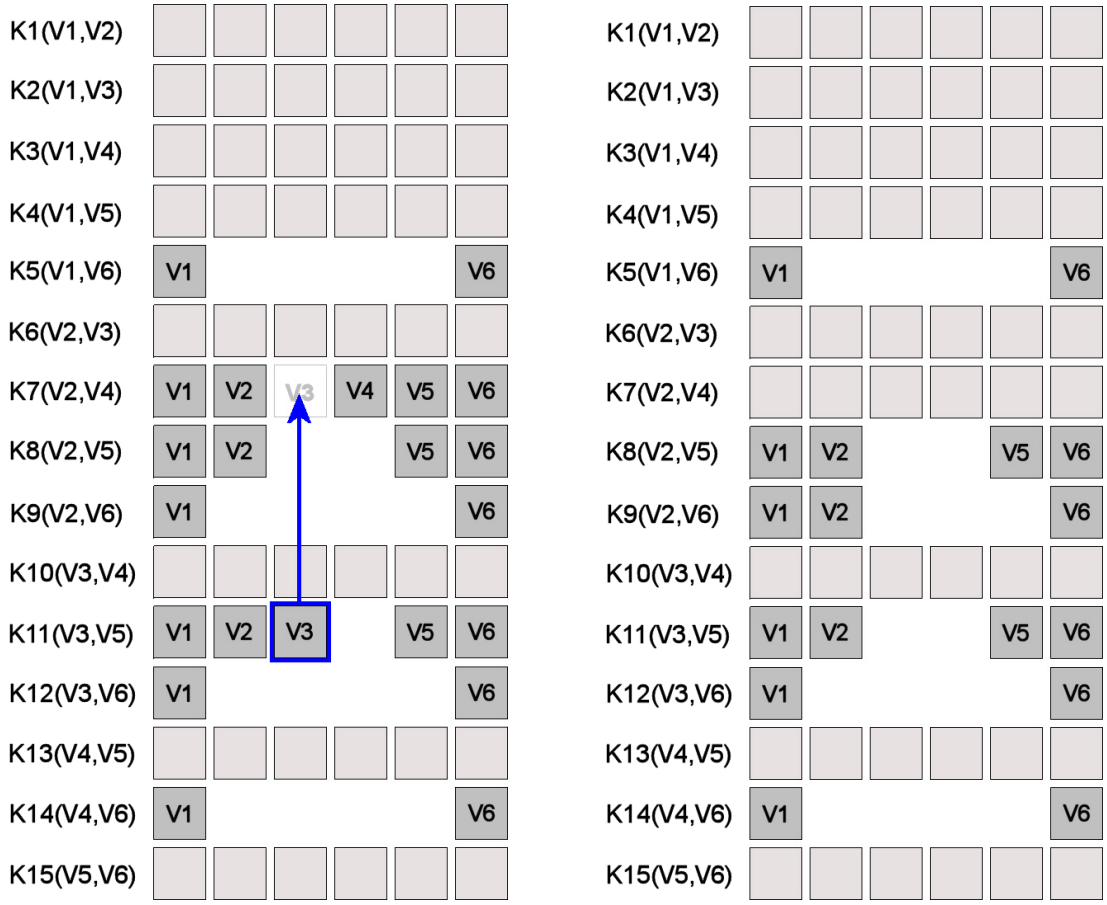

Abbildung 4.7.: Am Ende bleiben gewisse Randklassen übrig, die sich nicht mehr zusammenfassen lassen. Je mehr Klassen ein Problem hat, desto höher ist die Anzahl der Randklassen.

Die erste Hälfte beschreibt wieviele vollständige Instanzmengen n erzeugt werden. Wie man an diesem Beispiel sehen kann, entsprechen diese der ungeraden Anzahl zwischen 1 und $c-1$. In dem Beispiel kommt man auf insgesamt $9 \cdot n$, die sich aus den drei Werten 5 (im allgemeinen $c-1$), 3 (im allgemeinen $c-3$) und 1 (im allgemeinen $c-5$) zusammensetzen. Durch den Faktor $2 \cdot \left(\frac{i}{2} - \left\lfloor \frac{i}{2} \right\rfloor\right)$ werden nur die ungeraden Zahlen in Betracht gezogen. Die zweite Hälfte beschreibt die erzeugten Randklassen. Die Funktion $\#(v_x)$ bestimmt dabei die Anzahl der Instanzen, die die Klasse v_x enthält. Angefangen mit den Äußersten beiden Klassen v_1 und v_6, werden für jede gerade Zahl zwischen 1 und $c-1$ die jeweils nächst äußeren hinzugefügt. Jede dieser Randklassen tauchen $(c-i)$-mal auf. Der Faktor $2 \cdot \left(\frac{i+1}{2} - \left\lfloor \frac{i+1}{2} \right\rfloor\right)$ in der zweiten Hälfte beachtet nur die geraden Zahlen für die Berechnung der Randklassen.

4.3.3. Praktische Vergleiche der Laufzeit

Anhand der benötigten Trainingsintanzen erkennt man, dass die Trainingsphase des paarweise geordneten Klassifizierers sehr viel aufwendiger ist als die des Round-Robin Klassifizierers. Verhält es sich beim Round-Robin Klassifizierer derart, dass eine lineare Anzahl von $(c-1)n$ Instanzen benötigt wird, so werden vom paarweisen geordneten Ansatz eine quadratische Anzahl, von etwa rund $\frac{n}{3}\left(c^2-1\right)$ Instanzen, verwendet.

Diese Größenordnung spiegelt sich auch in den Laufzeiten der Experimente wider. Die Abbildung 4.8 zeigt ein Beispiel für die beiden Laufzeiten im Vergleich. Die Kurven stellen die Zeit in CPU-Sekunden in Abhängigkeit von der Klassenanzahl dar. Als Datensatz diente eine modifizierte Version von *MVArtificial* mit einem Naive Bayes Basislerner. Weiterhin ist anzumerken, dass die Laufzeitunterschiede durch Anwendung von komplexeren Basislernen, wie der Support Vector Machine, noch größer werden. Tabelle 4.1 veranschaulicht die durchschnittliche Laufzeiten, die durch die Experimente in dieser Arbeit ermittelt wurden. Es wurden vier verschiedene Basislerner bei drei unterschiedlichen Klassengrößen verwendet. Der Aufbau der Experimente wird im nächsten Kapitel ausführlich erläutert.

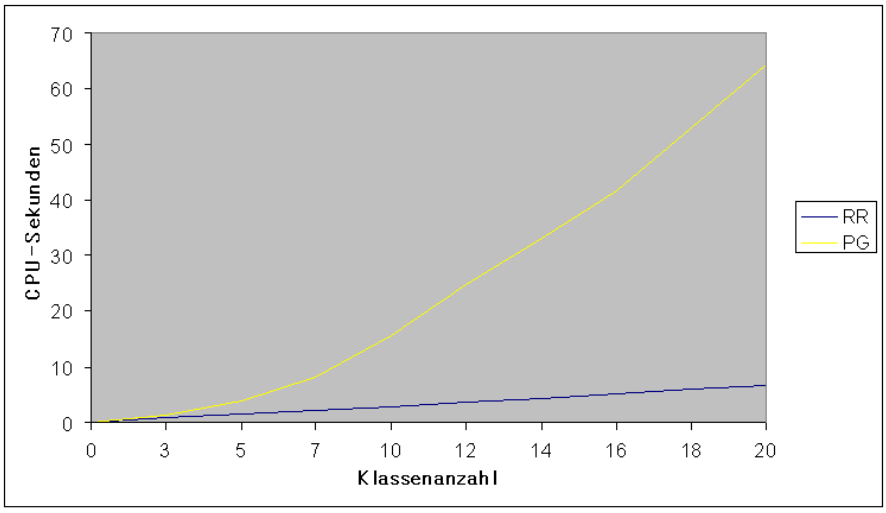

Abbildung 4.8.: Laufzeit der Klassifizierer in Abhängigkeit von der Klassenanzahl. Die blaue Kurve stellt die Laufzeit des Round-Robin Klassifizierers dar, die gelbe die des paarweise geordneten Klassifizierers.

4.4. Falschklassifizierungen

Bei Datensätzen ohne Klassenordnung ist es für die Evaluierung hinreichend nur die Genauigkeit eines Klassifizierers zu betrachten. Entweder wird eine Instanz richtig klassifiziert oder sie wird einer falschen Klasse zugeordnet. Welchen Wert diese falsche Klasse hat war für diese Art der Betrachtung nicht relevant. Ist dagegen in einem Datensatz eine

	Round-Robin Klassifizierer	paarweise geordneter Klassifizierer
3 Klassen		
Naive Bayes	0,42	0,63
J48	4,67	6,72
JRip	41,99	77,92
SMO	58,68	88,83
5 Klassen		
Naive Bayes	0,66	1,62
J48	6,86	15,42
JRip	48,52	170,58
SMO	51,53	169,93
10 Klassen		
Naive Bayes	1,10	5,25
J48	9,80	48,21
JRip	43,56	377,77
SMO	44,64	499,09

Tabelle 4.1.: Vergleich der Laufzeit beider Verfahren in CPU-Sekunden

Ordnung innerhalb der Klassenwerte vorhanden, reicht in vielen Fällen die Genauigkeit alleine nicht mehr aus, um Aussagen bezüglich der Performanz zu treffen. Fehler sind in solchen Fällen nicht einfach vorhanden oder nicht vorhanden, sondern tauchen in verschiedenen Größen auf. Je geringer der Abstand von der vorhergesagten Klasse zur tatsächlichen Klasse ist, desto besser ist solch eine fehlerhafte Vorhersage zu bewerten.

Wie in Abschnitt 3.2 erwähnt wurde, zeigen die Regressionsdatensätze auch eine gewisse Art von Ordnung auf. Bei Regressionslernern ist die Messung der Genauigkeit oder Fehlerrate als Maß für die Performanz ungeeignet. Stattdessen gibt es alternative Messwerte, wie z.B. den *mean-squared error*, *root mean-squared error*, *mean absolute error* usw. Die Regressionslerner werden dabei mittels der durchschnittlichen Abweichung bewertet.

Diese Art des Fehlermaßes kann man auch auf die paarweise geordnete Klassifikation übertragen. Als Mittelwert wird in dieser Arbeit der aus der Regression bekannten *mean absolute error* benutzt, der sich folgendermaßen zusammensetzt:

$$mean\ absolute\ error : \frac{|p_1 - a_1| + \ldots + |p_n - a_n|}{n}$$

p_i gibt dabei den vorhergesagten Wert und a_i den tatsächlichen Wert an. n entspricht der Anzahl der Instanzen. Um diese Formel auch auf ordinale Klassenwerte anwenden zu können, ist eine Umwandlung in numerische Werte notwendig. Dazu werden die Klassenwerte in feste, ganz zahlige Zahlen umgewandelt. Der kleinste Wert v_1 der Klassenordnung $V = \{v_1, v_2, v_3, \ldots, v_c\}$ bekommt die Zahl 1 zugewiesen und der jeweils nachfolgend grössere Klassenwert eine um 1 grössere Zahl.

$|p_i - a_i|$ gibt in diesen Fall an, um wieviele Klassen die vorhergesagte Klasse für die Instanz i von der tatsächlichen Klasse von i entfernt ist. Sie entspricht der Ordnungsdi-

stanz. Eine korrekt klassifizierte Intanz erhält somit eine Ordnungsdistanz von 0, eine um eine Klasse falsch klassifizierte Instanz eine Ordnungsdistanz von 1, usw. Alle Klassifizierungabstände werden summiert und durch die komplette Anahl der Instanzen dividiert. Als Resultat erhält man den *mean absolute error*, welcher in diesem Fall angibt, um wieviele Klassen im Durchschnitt eine vorhergesagte Instanz von der tatsächlichen Instanz abweicht.

Dieser Wert stellt neben der reinen Genauigkeit eine weitere Anschauungsweise dar, unter der die unterschiedlichen Klassifizierer miteinander verglichen werden können. Zur Veranschaulichung dient ein medizinisches Diagnosesystem, etwa wie in [2] vorgestellt. Ein Patient wird untersucht und anschließend wird eine Diagnose gestellt. Das Diagnoseergebnis stellt den Ausgangswert dar. Es gibt unterschiedliche Krankheiten zur Auswahl, die sich kategorisieren lassen, inwiefern sie negativen Einfluss auf das Wohlbefinden eines Patienten haben. Diese stellen die Klassenwerte dar, die sich dem Wohlbefinden nach ordnen lassen. Wird nun bei einem Vergleich zwischen Round-Robin und paarweisen geordneten Verfahren bei dem ersteren eine höhere Genauigkeit ermittelt, so bedeutet dies nicht, dass das Round-Robin Verfahren auch immer eine genauere Diagnose stellt. Zwar wird häufiger die richtige Diagnose gestellt, jedoch kann es bei Fehldiagnosen vorkommen, dass eine komplett falsche Krankheit zugeordnet wird, z.B. einen schwer kranken Patienten als gesund zu erklären. Der paarweise geordnete Ansatz hingegen würde bei Fehldiagnosen wenigstens einigermaßen verwandte Krankheiten mit einer ähnlichen "Wohlbefindesstufe" diagnostizieren. Welchen Einfluss Genauigkeit und mean absolute error haben gilt es somit für das jeweilige Problem zu bestimmen.

Nachfolgend wird der Unterschied zwischen ungeordneten und geordneten Problemen anhand eines praktischen Beispiels aus den Experimenten verdeutlicht. Es sei der Datensatz *Bank* $32NH$ mit zehn Klassenwerten gegeben. Die Klassen A, B, C, D, E, F, G, H, I, J sind aufsteigend in der Reihenfolge angegeben. Insgesamt besteht das Problem aus 8192 Instanzen. Die 10-fold Cross-Validation hat ergeben, dass der Round-Robin Klassifizierer eine höhere Genauigkeit von 32.62% besitzt als der geordnete paarweise Klassifizierer mit 27.66%.

Bank $32NH - SMO - Round\text{-}Robin$

Correctly Classified Instances	2672	32.6172 %
Incorrectly Classified Instances	5520	67.3828 %

Bank $32NH - SMO - paarweise\ geordnet$

Correctly Classified Instances	2266	27.6611 %
Incorrectly Classified Instances	5926	72.3389 %

Zusätzlich wird nun der mean absolute error beider Verfahren verglichen. Dazu betrachten wir uns die Konfusionsmatrix. Eine Konfusionsmatrix, wie in den Tabellen 4.2 und 4.3 dargestellt, gibt neben den korrekt klassifizierten Instanzen auch an, welcher konkrete Wert den falsch klassifizierten Instanzen zugeordnet wurde. In den Zeilen sind alle Instanzen einer Klasse angegeben, sowohl die als richtig und die falsch eingestuften

Instanzen. Die Spalten geben an, wievielen Instanzen dieser bestimmten Klasse welcher Wert zugewiesen wurde. Somit repräsentiert die Diagonale der Matrix die Genauigkeit mit den richtig klassifizierten Instanzen. Je weiter entfernt ein Feld von der Diagonale ist, desto grösser ist die Abweichung von der tatsächlichen Klasse.

A	B	C	D	E	F	G	G	I	J	<– classified as
1512	15	22	37	20	16	14	14	9	3	A
518	17	49	48	33	16	10	12	20	3	B
405	17	53	68	49	28	46	22	22	16	C
305	14	45	89	76	62	59	36	27	12	D
214	8	40	74	99	103	73	48	45	22	E
162	8	37	71	84	115	80	78	59	31	F
117	9	27	48	80	115	103	89	75	63	G
70	6	16	41	42	99	109	110	143	89	H
56	4	12	16	22	52	103	123	165	173	I
21	1	3	4	8	19	40	70	150	409	J

Tabelle 4.2.: Konfusionsmatrix des Round Robin Klassifizierer - sehr viele Instanzen wurde fälschlicherweise als A klassifiziert

A	B	C	D	E	F	G	H	I	J	<– classified as
797	444	206	112	43	30	23	7	0	0	A
107	234	158	118	52	25	18	12	2	0	B
55	159	171	123	90	64	38	17	9	0	C
26	100	132	151	127	101	59	16	13	0	D
20	53	97	143	158	121	81	40	11	2	E
11	38	70	125	134	145	103	73	26	0	F
6	13	47	78	137	170	137	87	47	4	G
10	14	20	38	84	138	176	149	89	7	H
3	12	23	24	41	85	161	188	167	22	I
0	2	8	6	15	41	90	144	262	157	J

Tabelle 4.3.: Konfusionsmatrix des paarweise geordneter Klassifizierer - Die Werte liegen eher auf der Diagonalen

Aus den Konfusionsmatrizen 4.2 und 4.3 wird ersichtlich, dass bei dem Round Robin Klassifizierer häufiger grössere Klassenabstände zwischen den vorhergesagten und tatsächlichen Klassenwert exisitieren. Wenn man den mean absolute error für beide Klassifizierer berechnet machen sich die Unterschiede bemerkbar.

$$mean\ absolute\ error - Round\text{-}Robin\ Klassifizierer$$
$$(2031 \cdot 1 + 1378 \cdot 2 + 859 \cdot 3 + 525 \cdot 4 + 308 \cdot 5 + 199 \cdot 6 + 127 \cdot 7 + 69 \cdot 8 + 24 \cdot 9)/8192$$
$$= 13855/8192 = 1.691$$

$$mean\ absolute\ error - paarweise\ geordneter\ Klassifizierer$$
$$(2745 \cdot 1 + 1680 \cdot 2 + 843 \cdot 3 + 358 \cdot 4 + 163 \cdot 5 + 93 \cdot 6 + 39 \cdot 7 + 5 \cdot 8 + 0 \cdot 9)/8192$$
$$= 11752/8192 = 1.435$$

Die bedeutet die vom paarweisen geordneten Klassifizierer getroffene Vorhersage ist im Schnitt etwa um 0.256 Klassen näher an der tatsächlichen Klasse als die Vorhersage des Round-Robin Klassifizierers.

Eine andere Möglichkeit, um diesen Sachverhalt darzustellen, ist es, die Daten der Konfusionsmatrix in einen Graphen zu übertragen. Die Anzahl der Instanzen wird auf die y-Achse und die Klassenwerte, in geordneter Reihenfolge, auf die x-Achse abgebildet. Für jede einzelne Klasse wird eine seperate Kurve erstellt. Bei einem Klassifizierer, der die Ordnung beachtet, sollte die daraus resultierende Kurve annähernd eine Normalverteilung sein. Das heißt, dass die Anzahl der korrekt klassifizierten Klassen am höchsten ist und mit zunehmender Abweichung von der korrekten Klasse die Anzahl stetig abnimmt. In Abbildung 4.9 ist ein theoretisches Beispiel dargestellt, wie eine ideale normalverteilte Klasse aussehen könnte. Die Kurve stellt die Klassenverteilung für eine Klasse E dar. Die meissten Instanzen werden korrekt als E klassifiziert. Je weiter entfernt eine Klasse von E ist, desto weniger Instanzen werden dieser zugeordnet.

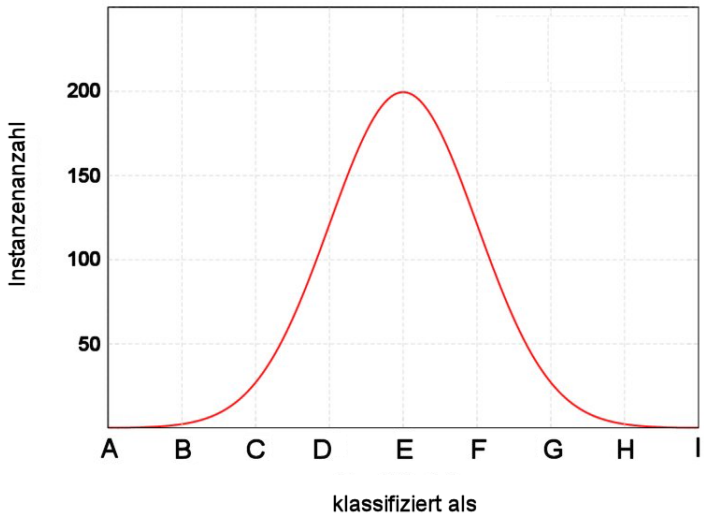

Abbildung 4.9.: Beispiel für eine theoretische Klassenverteilung; im Idealfall sollte diese einer Normalverteilung gleichen

Wenn man diese Darstellungsart auf die beiden praktisch ermittelten Ergebnisse des *Bank* 32*NH* Datensatzes anwendet, erhält man die Abbildung 4.10 und 4.11. In 4.11 sieht man, dass ein sehr hoher Anteil der Instanzen der Klasse *A* (1512 Instanzen) korrekt klassifiziert wurde (steile dunkelblaue Linie, links im Bild). Dies ist auch einer der Gründe für den höheren Genauigkeitswert; ein sehr hoher Anteil dieser Klasse wurde richtig klassifziert. Jedoch ist auch zu erkennen, dass sehr viele der anderen Klassen fälschlicherweise als *A* klassifiziert wurden. In Tabelle 4.10 sieht man, dass beim paarweise geordneten Ansatz ein geringerer Anteil von *A* (797 Instanzen) richtig klassifiziert wurde. Die Kurven gleichen dafür annähernd einer Normalverteilung. Der mean absolute error wird bei den Auswertungen der Experimente im nächsten Kapitel mit berücksichtigt.

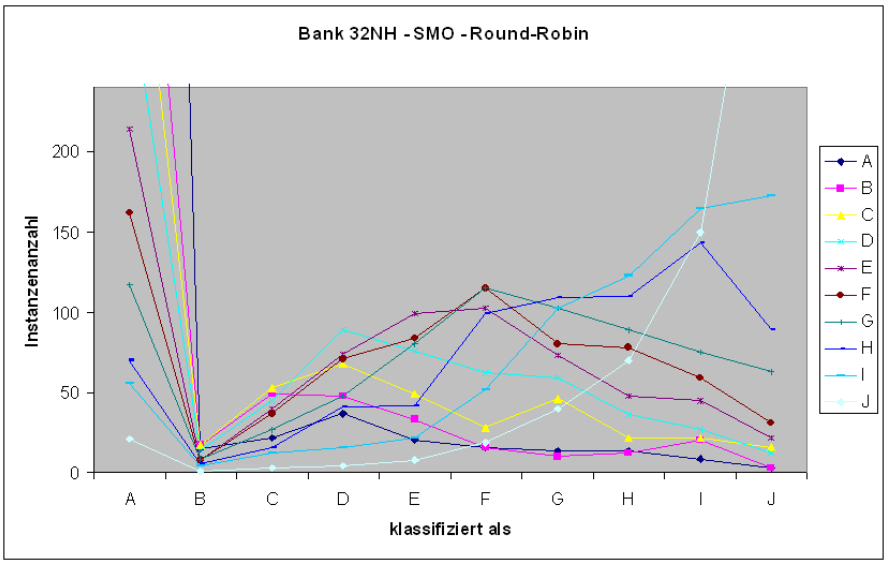

Abbildung 4.10.: Grafisch dargestellte Klassenverteilung des Round-Robin Klassifizierers - Es wird zwar ein hoher Anteil von A richtig klassifiziert, jedoch werden auch viele andere Klassen fälschlicherweise als A klassifiziert.

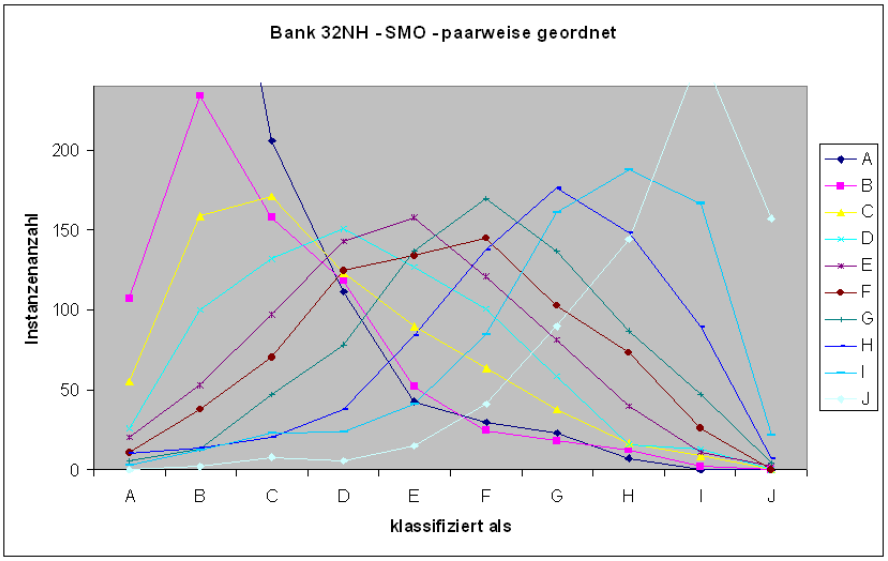

Abbildung 4.11.: Grafisch dargestellte Klassenverteilung des paarweise geordneten Klassifizierers - Die Klassenverteilung ähnelt eher einer Normalverteilung. Es finden weniger drastische Falschklassifizierungen statt.

5. Experimente

5.1. Testumgebung

Um zu testen, ob die Anwendung des paarweise geordneten Klassifizieres zu einer Verbesserung der Ergebnisse führt, wurden Experimente basierend auf realen Datensätzen durchgeführt. Als Testumgebung diente das Weka Framework [15] (Waikato Environment for Knowledge Analysis). Weka ist ein von der University of Waikato entwickeltes Programm zur Anwendung von maschinellen Lernverfahren. Weka ist eine in Java geschriebene freie Software, nach der allgemeinen öffentlichen GNU-Lizenz. Das Programm umfasst eine Reihe von Tools unter anderem zur Pre-Processing von Daten, Klassifikation, Regression, Clustering, Assoziationsregelnlernen und Visualisierung. Das Programm eignet sich sehr gut dazu, um neue Lernmodelle zu entwickeln und diese einzubinden. Die Experimente, die im Rahmen dieser Arbeit durchgeführt wurden, wurden auf Weka Version 3.5 ausgeführt. Die bereits bestehende Klasse $MultiClassClassifier.java$ aus Weka wurde um die Funktionsweise des paarweise geordneten Klassifizierers erweitert.

5.1.1. Datensätze

Die Datensätze stammen aus der UCI Repository [11]. Die UCI Repository ist eine von der University of California, Irvine gepflegte Sammlung von Datensätzen, die für die empirische Auswertung von maschinellen Lernverfahren verwendet werden. Die Sammlung enthält Datensätze von realen Problemen aus der Praxis. Da jedoch keine Datensätze mit Klassenordnung zur Verfügung standen, wurden stattdessen vorhandene Regressionsdatensätze umgewandelt. Mittels des *equal frequency binning* wurden die numerischen Zielwerte in geordnete Klassen zusammengefasst. Bei diesem Diskretisierungsverfahren wird der (unendliche) numerische Zielwertebereich in feste n Intervalle aufgeteilt. Das *equal frequency binning* sorgt außerdem dafür, dass jedes dieser Intervalle aus annähernd gleich vielen Instanzen besteht.

Als Beispiel für solch eine Zerlegung dient der aus Tabelle 3.1 bekannte Regressiondatensatz $CPU\ Performance$. Der Zielwert ist hier numerisch, er gibt die relative Leistung eines CPUs an. Nun sollen mittels *equal frequency binning* alle Instanzen dieses Datensatzes auf drei Klassen abgebildet werden. Es werden Intervalle bestimmt, die die Menge der Zielwerte in drei Teilmengen aufteilt. Die Grenzen werden dabei so gewählt, dass jede Menge aus gleich viel Instanzen besteht. Solch eine Teilmenge wird auch als *bin* bezeichnet. Abbildung 5.1 zeigt wie so eine Umformung aussehen kann. Die hier gewählten

Grenzen für die Klassen lauten:

$$-\infty < A \leq 64$$
$$74 < B \leq 381$$
$$381 < B < +\infty$$

Als Resultat erhählt man drei norminale Klassenwerte mit der Ordnung $A < B < C$.

MYCT	MMIN	MMAX	CACH	CHMIN	CHMAX	PRP
320	512	2000	4	1	3	21
320	256	3000	4	1	3	22
320	256	6000	0	1	6	28
320	256	5000	4	1	6	27
143	1500	6300	0	5	32	35
200	512	16000	0	4	32	64
50	2620	10480	30	12	24	74
25	1310	2620	131	12	24	102
64	5240	20970	30	12	24	136
56	5240	20970	30	12	24	138
140	2000	32000	32	1	20	175
140	2000	32000	32	1	54	181
23	16000	32000	64	16	32	381
30	16000	32000	256	16	24	603
23	16000	64000	64	16	32	749
30	8000	64000	96	12	176	919
30	8000	64000	128	12	176	978
23	32000	64000	128	32	64	1238

MYCT	MMIN	MMAX	CACH	CHMIN	CHMAX	PRP
320	512	2000	4	1	3	A
320	256	3000	4	1	3	A
320	256	6000	0	1	6	A
320	256	5000	4	1	6	A
143	1500	6300	0	5	32	A
200	512	16000	0	4	32	A
50	2620	10480	30	12	24	B
25	1310	2620	131	12	24	B
64	5240	20970	30	12	24	B
56	5240	20970	30	12	24	B
140	2000	32000	32	1	20	B
140	2000	32000	32	1	54	B
23	16000	32000	64	16	32	C
30	16000	32000	256	16	24	C
23	16000	64000	64	16	32	C
30	8000	64000	96	12	176	C
30	8000	64000	128	12	176	C
23	32000	64000	128	32	64	C

Abbildung 5.1.: Beispiel einer Diskretisierung mittels *equal frequency binning*

Tabelle 5.1.1 stellt die 29 Datensätze, die für die Auswertungen benutzt wurden, mit ihren Eigenschaften dar. Die Spalten gliedern sich auf in Name des Datensatzes, Anzahl der Instanzen, Gesamtanzahl der Attribute, Anzahl an numerischer Attributen und Anzahl an nominaler Attribute. Um den Zusammenhang zwischen Klassengrösse und Genauigkeit zu analysieren, wurden aus jedem Regressionsdatensatz drei verschiedene Versionen mit unterschiedlichen Bingrößen erstellt. Die hierfür gewählten Intervalle bestehen aus jeweils drei, fünf und zehn Klassen. Insgesamt wurden so 87 verschiedene Datensätze erstellt. Diese Datensätze wurden in der Form auch für die Experimente in der Arbeit von Frank und Hall [15] genutzt.

Die verwendeten Binärisierungsarten beinhalten neben der paarweise geordneten (PG) und Round Robin (RR) auch die one-against-all (1VA), die geordnete Binärisierung von Frank und Hall (FH) sowie die direkte Anwendung des Basislerners auf Multiklassenprobleme. Als Basislerner dienten vier verschiedene Modelle: der Entscheidungsbaumlerner J48, der Regellerner JRip, die Support Vector Machine SMO und der Naive Bayes Lerner NB. Jeder der Basislerner, auch die spezielle Version des Support Vector Machine SMO, unterstützt eine direkte Anwendung auf Multiklassenprobleme. Der Umfang der eingesetzten Methoden und Verfahren wurde möglichst groß gewählt, um genau analysieren zu können von welchen Faktoren eine Verbesserung bzw. Verschlechterung abhängig ist. Untersucht wurden unter anderem die Genauigkeit, Trainingszeit und der mean absolute error.

Dataset	Instances	Attributes	Numeric	Nominal
2D Planes	40768	11	10	1
Abalone	4177	9	7	2
Ailerons	13750	41	40	1
Auto MPG	398	8	4	4
Auto Price	159	16	15	1
Bank 8FM	8192	9	8	1
Bank 32NH	8192	33	32	1
Boston Housing	506	14	12	2
California Housing	20640	9	8	1
CPU Small	8192	13	12	1
CPU Act	8192	22	21	1
Delta Ailerons	7129	6	5	1
Delta Elevators	9517	7	6	1
Diabetes	43	3	2	1
Elevators	16599	19	18	1
Friedman Artificial	40768	11	10	1
House 8L	22784	9	8	1
House 16H	22784	17	16	1
Kinematics	8192	9	8	1
Machine CPU	209	7	6	1
MV Artificial	40768	11	7	4
Pole Telecom	15000	49	48	1
Pumadyn 8NH	8192	9	8	1
Pumadyn 32H	8192	33	32	1
Pyrimidines	74	28	27	1
Servo	167	5	0	5
Stocks	950	10	9	1
Triazines	186	61	60	1
Wisconsin Breast Cancer	194	33	32	1

Tabelle 5.1.: Die für die Experimente verwendeten Datensätze

5.1.2. 10-fold Cross-Validation

Die ermittelten Genauigkeitsabgeschätzungen sind das gerundete Ergebnis von zehn unabhängigen Testläufen einer 10-fold Cross-Validation. Diese Methode dient zur Evaluierung von Modellen. Eine Teilmenge des Datensatzes dient dabei zum Training des Lernverfahrens, die andere zum Testen. Bei der 10-fold Cross-Validation wird ein Datensatz in 10 disjunkte Mengen aufgeteilt. Neun der zehn Teilmengen dienen als Trainingssatz und die restliche als Testsatz. Dieser Vorgang wird zehmal wiederholt. Jedesmal dient eine andere der zehn Teilmengen als Testsatz. Jede Instanz taucht somit genau einmal im Testsatz und die restlichen neunmal im Trainingssatz auf. Aus allen zehn Läufen wird dann die durchschnittliche Genauigkeit bestimmt. Es gibt die Möglichkeit diesen ganzen Vorgang wiederum mehrmals zu wiederholen, mit anderen Aufteilungen der disjunkten Mengen. Für vorliegende Experimente wurde die einmalige 10-fold Cross-Validation durchgeführt.

5.1.3. Gepaarter t-Test

Die paarweise geordnete Binärisierung soll mit den anderen Binärisierungsarten verglichen werden. Dazu können die durch 10-fold Cross-Validation ermittelten Genauigkeits-Werte miteinander verglichen werden. Das Verfahren mit den größeren Genauigkeitswert wird dann als besser bewertet. Dies reicht in vielen praktischen Anwendungen aus. Jedoch kann es sein, dass die Unterschiede lediglich durch Einschätzungsfehler zustande gekommen sind. In einigen Situationen ist es wichtig zu bestimmen, ob ein auf ein bestimmtes Problem angewandtes Schema wirklich besser ist als ein anderes. Bei einem neuen Lernalgorithmus sollte gezeigt werden, dass die beobachteten Veränderungen nicht einfach eine Zufallserscheinung des Einschätzungsvorgangs sind.

Der gepaarte t-Test ist ein statistischer Test, der überprüft, ob die Unterschiede signifikant verschieden sind. Es wird ermittelt, ob der durchschnittliche Wert eines Verfahrens signifikant grösser oder kleiner als der Durchschnittswert eines anderen Verfahren ist. Da beide Verfahren auf den selben Datensatz angewendet werden spricht man von einen gepaarten Test.

Durch Anwendung der 10-fold Cross-Validation erhält man eine Reihe von Stichproben $x_1, x_2, \ldots x_{10}$, die Gebrauch von einem Verfahren machen und eine zweite Reihe von Stichproben $y_1, y_2, \ldots y_{10}$, die von einem anderen Verfahren Gebrauch machen. Wie im vorherigen Abschnitt erklärt wurde, nutzt jeder Schritt der Cross-Validation eine andere Teilmenge des Datensatzes. Für beide Verfahren werden die gleichen Cross-Validation Teilmengen gewählt, so dass x_i und y_i das Ergebnis einer identischen Cross-Validatin Aufteilung sind.

Es soll bestimmt werden, ob der Durchschnitt $\hat{x}$ der ersten Stichproben signifikant anders von den Durchschnitt $\hat{y}$ der zweiten Stichproben ist. Wenn es genug Stichproben gibt, ist der Durchschnitt $\hat{x}$ normalverteilt und hat einen Ertwartungswert μ und die Standardabweichung $\frac{\sigma}{\sqrt{n}}$. Da die Standardabweichung jedoch unbekannt ist, muss diese anhand der Menge von Stichproben abgeschätzt werden. Die empirische Standardabweichung lautet somit:

$$s = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \hat{x})^2}{n-1}}$$

Da die Standardabweichung nur eine Abschätzung ist hat dieser Fall keine Normalverteilung. Stattdessen hat es eine so genannte Studentsche t-Verteilung. Das bedeutet, dass dementsprechend eine Tabelle von Konfidenzintervallen für die Studentsche t-Verteilung benutzt werden muss. Die zu verwendete Testprüfgröße lautet:

$$t = \sqrt{n}\frac{\hat{x} - \mu}{s}$$

Für die Experimente in dieser Arbeit wurde eine Irrtumswahrscheinlichkeit von 5% gewählt. Ist der Wert t größer als die entsprechend berechnete Vertrauensgrenze, so wird die Nullhypothese, dass die Durchschnitte gleich sind, abgelehnt. Es wird gefolgert, dass ein signifikanter Unterschied zwischen den beiden Lernverfahren vorherrscht.

5.2. Ergebnisse

Im Anhang A sind die Ergebnisse der 10-fold Cross-Validation im einzelnen zusammengefasst. Die Tabellen auf der oberen Hälfte geben die durchschnittliche Genauigkeit an. Die Zeilen listen die Datensätze auf und die Spalten stellen den Durchschnittswert der einzelnen Binärisierungsarten dar. Für den t-Test wurde jedes Verfahren in Bezug auf den paarweise geordneten Klassifzierer verglichen. Ein $++$ oder $--$ gibt an, ob der jeweilige Wert signifikant besser oder schlechter als der Wert des paarweise geordneten Klassifzierers war. Ein einfaches $+$ oder $-$ gibt ohne Berücksichtigung der Signifikanz an, ob die Genauigkeit des jeweiligen Verfahren höher oder niedriger als die des paarweise geordneten Klassifizieres war. Sind die untersuchten Werte identisch, so steht ein $=$ hinter dem Ergebnis.

In der ersten Spalte sind Ergebnisse des paarweise geordneten Verfahrens angegeben. Die zweite Spalte enthält die Ergebnisse der direkten Anwendung des jeweiligen Lernverfahrens ohne Binärisierung. Wie auch in der Arbeit von Frank und Hall wurde die one-against-all Biärisierung angewendet, um sicher zu stellen, dass die Unterschiede in der Performanz nicht nur durch die Binärisierung zustande gekommen sind. Dieser Wert ist in Spalte drei zu finden. Der eigentlich interessante Wert, der des Round-Robin Verfahrens, befindet sich in der vierten Spalte. Zudem wurde der ordinale Klassifizierer von Frank und Hall in Spalte fünf in Betracht gezogen.

Zudem fassen die Gewinn/Verlust-Tabellen in der unteren Hälfte von Anhang A, zusammen wie oft ein Verfahren im Vergleich zu einem anderen Verfahren besser bzw. schlechter war. Der Wert in einer Zelle gibt an, wie oft das Verfahren in der Spalte besser war als das Verfahren in der Zeile. Der in Klammern gesetzte Wert gibt an wieoft es davon signifikant besser war.

Im Anhang B sind die Ergebnisse der mean absolute error Auswertungen zu finden. Verglichen wurde das paarweise geordnete Verfahren mit dem Round-Robin Verfahren. Ein $+$ oder $-$ gibt an, ob der mean absolute error des Round-Robin Verfahrens vergleichsweise höher oder niederiger war. Ein $=$ steht bei Gleichheit der beiden Werten. Ein t-Test wurde hier nicht durchgeführt.

5.2.1. Genauigkeit

Es ist zu überprüfen, ob die Unterschiede in der Genauigkeit von der Anzahl der Klassen abhängig sind. Eine angemessene Annahme ist es, dass die Performanzunterschiede mit der Anzahl der Klassen zunimmt. Um dies zu überprüfen wurden die Verfahren auf den gleichen Datensatz mit jeweils drei verschiedenen Diskretisierungen der Zielwerte eingesetzt. Es wurden vier verschiedene Typen von Lernverfahren verwendet - Entscheidungsbaumlerner J48, Regellerner JRip, Support Vector Machine SMO und Naive Bayes Lerner NB.

Zunächst werden die Ergebnisse für drei Klassen aus Anhang A.1 in Betracht gezogen. Man sieht, dass bei dieser geringen Klassenanzahl die Berücksichtigung der Klassenordnung nicht zu einer Verbesserung gegenüber dem Round-Robin Verfahren führt. Verglichen mit den J48, ist J48-PG auf 5 Datensätzen signifikant besser und auf keinem

signifikant schlechter. Nimmt man die Signifikanz außer Betracht, so gewinnt J48-PG gegen J48 auf 22 Datensätzen und verliert auf 7. Eine Verbesserung ist hier zu erkennen. Die Unterschiede zwischen J48-PG und J48-RR machen sich nur wenig bemerkbar. Es gibt keine signifikanten Unterschiede. Auch unter Mitberücksichtigung der nicht signifikanten Fälle kann das Gewinn/Verlust-Verhältnis von 15/12 keinen eindeutigen Rückschluss über eine Verbesserung geben.

	J48-PG	J48	J48-RR
J48-PG	-	7 (0)	12 (0)
J48	22 (5)	-	20 (5)
J48-RR	15 (0)	13 (1)	-

	JRip-PG	JRip	JRip-RR
JRip-PG	-	9 (0)	17 (3)
JRip	19 (7)	-	20 (9)
JRip-RR	10 (0)	11 (0)	-

Tabelle 5.2.: Gewinn/Verlust-Tabelle für J48 und JRip bei Datensätzen mit 3 Klassen

Bei JRip lässt sich ein ähnlicher Zusammenhang feststellen. JRip-PG ist 7-mal signifikant besser als JRip und kein einziges Mal signifikant schlechter. Bei Betrachtung ohne Signifikanz beträgt hier das Gewinn/Verlust-Verhältnis 19/9. Bei einem Vergleich mit JRip-RR deutet das Ergebnis eher auf eine Verschlechterung hin. Kein einziges Mal ist JRip-PG signifikant besser als JRip-RR, dafür aber 3-mal signifikant schlechter. Ohne Berücksichtigung der Signifkanz erhält man ein Gewinn/Verlust-Verhältnis von 10/17. Tabelle 5.2 stellt die Gewinn/Verlust-Tabelle, mit den signifikanten Werten in den Klammern, dar.

Bei den anderen beiden Lernverfahren schneidet das paarweise geordnete Verfahren schlechter ab. Verglichen mit SMO-RR beträgt das Gewinn/Verlust-Verhältnis 3/25, bzw. 2/8 bei der signifikanten Unterscheidung. Dies ist ein deutlicher Verlust des SMO-PG. Es ist zu beobachten, dass es auch bei einer Gegenüberstellung des SMO-PG und des unmodifizierten SMO zu einer Niederlage für SMO-PG kommt.

Auf ein ähnliches Resultat kommt man bei dem Naive Bayes Lernverfahren. Auch hier ist NB-PG signifikant schlechter, sowohl im Vergleich zu NB-RR als auch zu NB. Tabelle 5.3 zeigt die Gewinn/Verlust-Aufteilung für SMO und NB.

	SMO-PG	SMO	SMO-RR
SMO-PG	-	20 (5)	25 (8)
SMO	6 (2)	-	21 (3)
SMO-RR	3 (2)	6 (0)	-

	NB-PG	NB	NB-RR
NB-PG	-	25 (15)	24 (15)
NB	4 (1)	-	9 (0)
NB-RR	5 (1)	10 (0)	-

Tabelle 5.3.: Gewinn/Verlust-Tabelle für SMO und NB bei Datensätzen mit 3 Klassen

Diese Ergebnisse spiegeln sich so auch in den insgesamten Durchschnittswerten wider. Tabelle 5.4 zeigt den gemittelten Wert aller Genauigkeitsabschätzungen von Datensätzen mit 3 Klassen.

Als nächstes werden die Ergebnisse für 5 Klassen aus Anhang A.2 in Betrag gezogen. Bei J48 lässt sich auch hier, wie bei dem Fall für 3 Klassen, keine Verbesserung feststellen. J48-PG hat gegenüber J48-RR kein Mal gewonnen und 3-mal verloren bei einem Vergleich mit Signifikanz, bzw. 15-mal gewonnen und 14-mal verloren bei einem Vergleich ohne

	paarweise geordnet	unmodifiziert	Round-Robin
J48	71.25	70.84	71.09
JRip	72.14	70.35	73.07
SMO	67.43	69.72	70.49
NB	61.59	63.53	63.34

Tabelle 5.4.: Durchschnittliche Genauigkeit der Verfahren bei Datensätzen mit 3 Klassen

Signifikanz. Dies deutet darauf hin, dass das J48-PG eher schlechter abschneidet als J48-RR. Der Vorteil gegenüber den unmodifizierten J48 ist gestiegen. 12-mal ist J48-PG signifikant besser als J48.

Auch die Performanz des JRip ist durch die Aufteilung in 5 Klassen nur minimal gestiegen. Zwar ist die Anzahl, in denen JRip-PG gegenüber JRip-RR gewonnen hat auf 11 gestiegen (davon jetzt 1-mal signifikant), aber immer noch verliert es mit 18-mal (davon wieder 3-mal signifikant) genauso oft. Dies ermöglicht die Schlussfolgerung, dass JRip-PG nicht zu einer Verbesserung, sondern eher zu einer kleinen Verschlechterung gegenüber JRip-RR führt. Im Vergleich mit JRip ist der Gewinnanteil weiter gestiegen. Das Gewinn/Verlust-Verhältnis beträgt nun 13/0 im signifikanten Fall und 25/4 insgesamt. Die wichtigsten Ergebnisse aus dem Anhang sind in Tabelle 5.5 zusammengefasst.

	J48-PG	J48	J48-RR
J48-PG	-	4 (1)	14 (3)
J48	25 (12)	-	25 (13)
J48-RR	15 (0)	3 (0)	-

	JRip-PG	JRip	JRip-RR
JRip-PG	-	4 (0)	18 (3)
JRip	25 (13)	-	28 (16)
JRip-RR	11 (1)	2 (0)	-

Tabelle 5.5.: Gewinn/Verlust-Tabelle für J48 und JRip bei Datensätzen mit 5 Klassen

Tabelle 5.6 zeigt die Gewinnaufteilung für SMO und NB. Bei der Support Vector Machine ist anzumerken, dass SMO-PG im Vergleich mit den Fall von 3 Klassen noch öfter verliert. Die Anzahl der Verluste gegenüber dem unmodifizierten SMO steigt auf 14 und beim SMO-RR auf 18 signifikante Fälle. Die beiden Male die SMO-PG, im Fall von 3 Klassen, noch signifikant gewonnen hatte fällt hier auf 0.

Beim Naive Bayes Verfahren bleibt die Performanz des NB-PG praktisch unverändert. Es gibt keine nennenswerte Unterschiede zu dem Fall mit 3 Klassen.

	SMO-PG	SMO	SMO-RR
SMO-PG	-	21 (14)	25 (18)
SMO	8 (0)	-	20 (8)
SMO-RR	4 (0)	7 (0)	-

	NB-PG	NB	NB-RR
NB-PG	-	23 (15)	25 (16)
NB	5 (1)	-	13 (1)
NB-RR	6 (1)	10 (1)	-

Tabelle 5.6.: Gewinn/Verlust-Tabelle für SMO und NB bei Datensätzen mit 5 Klassen

In Tabelle 5.7 sind die Durchschnittswerte aller Auswertungen für die Aufteilung in 5 Klassen angegeben. Gibt es bei J48 und JRip nur einen geringen Unterschied zwischen

dem paarweise geordneten und Round-Robin Verfahren, so sieht man bei SMO und NB deutliche Unterschiede in den ermittelten Genauigkeiten.

	paarweise geordnet	unmodifiziert	Round-Robin
J48	58.51	56.94	59.19
JRip	58.69	55.02	59.93
SMO	53.42	56.39	56.83
NB	45.34	48.36	48.57

Tabelle 5.7.: Durchschnittliche Genauigkeit der Verfahren bei Datensätzen mit 5 Klassen

Hinsichtlich der Aufteilung in 10 Klassen aus Anhang A.3 kommt es auch hier nicht zu einer Verbesserung. Bei Auswertung der Resultate ist hier zu beachten, dass aufgrund ihres Umfangs, die in zehn Klassen aufgeteilte Versionen der Datensätze *Ailerons* und *Friedman* nicht getestet werden konnten. Die dafür benötigte Speicherkapazität ging über die in der Testumgebung zur Verfügung stehenden hinaus.

Tabelle 5.8 fasst die wichtigsten Ergebnisse zusammen. J48-PG schneidet nun einmal signifikant besser ab als J48-RR, dafür fallen die allgemeinen Gewinne von 15 auf 12. Auch nimmt die Anzahl der signifikanten Verluste von 3 auf 5 zu. War es beim Anstieg von 3 auf 5 Klassen so, dass die Performanz des J48-PG gegenüber dem J48 gestiegen ist, so ist solch eine Entwicklung nicht bei einer weitereren Erhöhung der Klassenanzahl zu beobachten. Die Anzahl der Gewinne des J48-PG, auf Datensätzen mit 10 Klassen, fällt minimal von 25 auf 23 (12-mal davon signifikant).

Es ist zu beobachten, dass es im Vergleich des JRip-PG mit den JRip-RR nicht zu der erwarteten Verbesserung kommt. Die Anzahl der Gewinne nimmt nur minimal um 4 auf insgesamt 15 zu, keiner davon ist signifikant. Dagegen steigt die Verlustanzahl um 3 auf insgesamt 6 signifikante Fälle.

	J48-PG	J48	J48-RR
J48-PG	-	4 (0)	15 (5)
J48	23 (12)	-	19 (13)
J48-RR	12 (1)	8 (1)	-

	JRip-PG	JRip	JRip-RR
JRip-PG	-	1 (0)	11 (6)
JRip	26 (20)	-	26 (18)
JRip-RR	15 (0)	1 (0)	-

Tabelle 5.8.: Gewinn/Verlust-Tabelle für J48 und JRip bei Datensätzen mit 10 Klassen

Tabelle 5.9 stellt die Ergebnisse für SMO und NB dar. Auch bei Datensätzen mit 10 Klasen verlieren die paarweise geordneten Verfahren deutlich sowohl gegen die unmodifzierten SMO und NB als auch gegen SMO-RR und NB-RR. Die Ergebnisse der Gewinn/Verlust-Tabellen sind sehr ähnlich wie bei dem Fall mit 5 Klassen.

In den durchschnittlichen Genauigkeitswerten aus Tabelle 5.10 spiegelt sich das Resultat wider. Bei SMO und NB schneidet das paarweise geordnete Verfahren deutlich schlechter ab. Bei J48 und JRip gibt es nur einen minimalen Unterschied zwischen dem paarweise geordneten und Round-Robin Verfahren.

Zusammenfassend lässt sich sagen, dass die Einführung des paarweise geordneteten Verfahren nicht zu einer Verbesserung des Ergebnisses geführt hat. Bei J48 und JRip

	SMO-PG	SMO	SMO-RR
SMO-PG	-	17 (13)	24 (16)
SMO	10 (1)	-	20 (8)
SMO-RR	3 (0)	7 (0)	-

	NB-PG	NB	NB-RR
NB-PG	-	23 (14)	23 (14)
NB	3 (1)	-	8 (0)
NB-RR	4 (1)	13 (0)	-

Tabelle 5.9.: Gewinn/Verlust-Tabelle für SMO und NB bei Datensätzen mit 10 Klassen

	paarweise geordnet	unmodifiziert	Round-Robin
J48	40.30	38.90	40.67
JRip	39.98	31.54	39.78
SMO	34.25	36.57	38.33
NB	27.40	32.08	31.99

Tabelle 5.10.: Durchschnittliche Genauigkeit der Verfahren bei Datensätzen mit 10 Klassen

sind die Ergebnisse des paarweise geordneten Verfahren ähnlich mit dem des Round-Robin Verfahrens. Das paarweise geordnete Verfahren schneidet minimal schlechter ab.

Als Erklärung für eine sinkende Genauigkeit wäre denkbar, dass durch die Hinzunahme von zusätzlichen Beispielen in die Trainingsmenge, die gelernten Klassifizier stark verallgemeinern. Als ein Beispiel dient ein Datensatz mit 10 Klassen, die sich der Klassenordnung nach $\{A, B, C, D, E, F, G, H, I, J\}$ aufreihen lassen. Klassifizier $K(A, B)$ enthält als Trainingsmenge für Klasse A die Instanzen der Klasse A und für Klasse B alle Instanzen grösser B, also alle restlichen Instanzen. Es wäre denkbar, dass dadurch die Vorhersage für Klasse B so verallgemeinert wird, dass während Klassifikationsphase Instanzen der Klasse A, fälschlicherweise die Klasse B zugewiesen bekommen.

Bei SMO und NB schneidet die paarweise geordnete Variante deutlich schlechter ab als die Round-Robin Variante. Interessant ist es zu beobachten, dass bei diesen Lernverfahren die paarweise geordnete Variante auch schlechter als die unmodifizierten Versionen SMO und NB abschneidet. Vergleichbar schlechte Resultate liefert hier die ordinale Klassifikation von Frank und Hall ab. Auch die unterliegt leistungsmäßig bei SMO und NB sowohl der Round-Robin Variante als auch den unmodifizierten Versionen. Dies lässt die Frage offen, ob diese beiden Lernverfahren ungeeignet für die Hinzunahme von Informatin bzgl. der Klassenordnung sind.

5.2.2. Mean absolute error

In diesem Abschnitt wird der mean absolute error zwischen dem paarweise geordneten Klassifizierer und dem Round-Robin Klassifizierer verglichen. Wie in Kapitel 4.4 beschrieben wurde, ist der mean absolute error ein Maß für die durchschnittliche Abweichung eines Klassifizierers. Der Wert gibt an um wieviele Klassen eine Vorhersage im Durchschnitt von der tatsächlichen Klasse entfernt ist. Die Zielsetzung ist es einen möglichst niedrigen mean absolute error zu erhalten.

Eine Annahme hier ist, dass durch die Berücksichtigung der Klassenordnung, der paar-

weise geordnetete Klassifizierer einen geringeren mean absolute error hat als der Round-Robin Klassifzierer. Mit wachsendender Klassenanzahl sollten die Unterschiede deutlicher werden. Dazu wurden die Datensätze wieder mit 3, 5 und 10 Klassen erzeugt. Auf einen statistischen Test, wie beim Vergleich der Genauigkeit, wurde hier verzichtet.

Hierzu liegen folgende Ergebnisse der Datensätze mit 3 Klassen vor. Der mean absolute error von J48-PG ist auf 19 Datensätzen niedriger und auf 6 Datensätzen höher als der des J48-RR. Bei JRip-PG ist der Wert 17-mal besser und 9-mal schlechter.

Bei SMO und NB ist das Ergebnis nicht so positiv. Bei SMO-PG ist die durchschnittliche Abweichung 12-mal besser und 13-mal schlechter als die von SMO-RR. Bei NB-PG ist das Ergebnis nur 6-mal besser und 22-mal schlechter als NB-RR.

	paarweise geordnet	Round-Robin
J48	0.311	0.320
JRip	0.303	0.307
SMO	0.337	0.334
NB	0.443	0.428

Tabelle 5.11.: Durchschnittlicher mean absolute error bei Datensätzen mit 3 Klassen

Tabelle 5.11 zeigt den durchschnittlichen mean absolute error für die Datensätze mit 3 Klassen. Man sieht, dass die paarweise geordnete Version von J48 und JRip besser abschneidet als die entsprechenden Round-Robin Versionen. Der Unterschied ist jedoch sehr gering,

Es folgt der Vergleich der Ergebnisse der Datensätze mit 5 Klassen. Abgesehen vom Naive Bayes Verfahren schneidet die paarweise geordnete Variante auch hier besser ab. Die Unterschiede sind im Gegensatz zum 3-Klassen-Fall grösser. Bei 22 Datensätzen hat J48-PG einen geringeren mean absolute error und bei 5 einen höheren als J48-RR. Beim JRip-PG ist der Wert 21-mal geringer und 6-mal höher. Auch SMO-PG schneidet diesmal 18-mal besser und 11-mal schlechter als SMO-RR ab.

NB-PG erreicht weiterhin kein besser Ergebnis als NB-RR - 11-mal ist der durchschnittliche Error niedriger und 16-mal höher.

Tabelle 5.12 fasst die durchschnittlichen mean absolute error Werte zusammen. Im Gegensatz zum 3-Klassen-Fall sind die Unterschiede nun grösser. So beträgt er bei J48 z.B. nun im Schnitt 0.029 $(0.566 - 0.537)$ Klassen , also etwas über 5 %.

	paarweise geordnet	Round-Robin
J48	0.537	0.566
JRip	0.538	0.558
SMO	0.547	0.610
NB	0.808	0.764

Tabelle 5.12.: Durchschnittlicher mean absolute error bei Datensätzen mit 5 Klassen

Durch die Hinzunahme von weiteren Klassen wächst der Unterschied im mean absolute error weiter an. Bei Datensätzen mit 10 Klassen hat J48-PG bei 28 Datensätzen eine

geringere Durchschnittsabweichung. Auch JRip-PG schneidet gut ab. An dieser Stelle sei anzumerken, dass es bei JRip aufgrund der Komplexität des Verfahrens und dem Umfang der Datensätze nicht möglich war, alle Auswertungen auf der gegebenen Testumgebung durchzuführen. Es wurden nur 23 Fälle untersucht, wobei JRip-PG bei allen außer einem einen niedrigeren mean absolute error hatte.

Auch füf SMO-PG ist das Ergebnis positiv. Die Anzahl der Datensätze auf denen es schlechter abschneidet ist auf 6 gefallen. Bei NB-PG ist weiterhin keine Verbesserung festzustellen. Tabelle 5.13 stellt die durchschnittlichen Werte des mean absolute error bei Datensätzen mit 10 Klassen dar.

	paarweise geordnet	Round-Robin
J48	1.037	1.157
JRip	1.077	1.221
SMO	1.158	1.296
NB	1.766	1.618

Tabelle 5.13.: Durchschnittlicher mean absolute error bei Datensätzen mit 10 Klassen

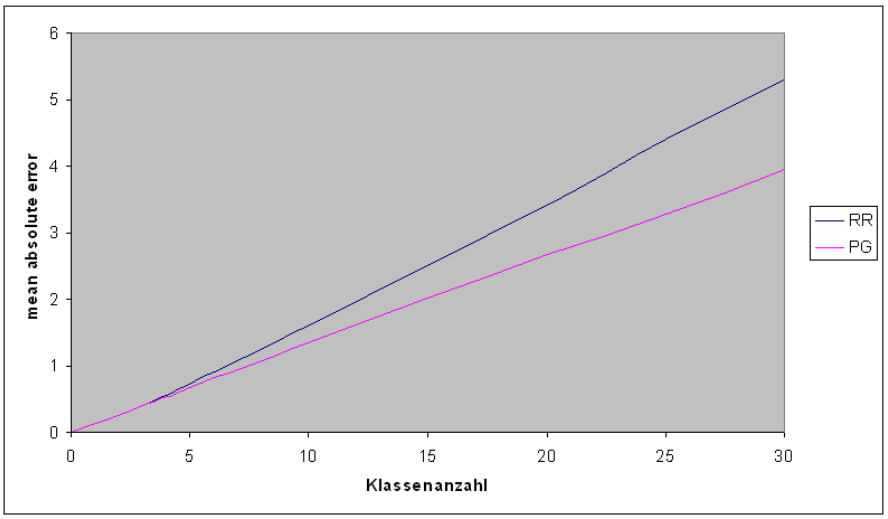

Abbildung 5.2.: Mean absolute error in Abhängigkeit von der Klassenanzahl

Abgesehen vom Naive Bayes Verfahren schneidet der paarweise geordnete Klassifizierer, in Bezug auf den mean absolute error, also besser ab als der Round Robin Klassifizierer. Mit zunehmender Klassenanzahl wird der Unterschied größer. Abbildung 5.2 zeigt für den Datensatz *Kinematics* den mean absolute error in Abhängigkeit von der Klassenanzahl. Als Basislerner diente der Entscheidungsbaumlerner J48. So beträgt z.B. der mean absolute error von J48-RR bei 30 Klassen 5.288 im Vergleich zu 3.951 bei J48-PG. Dies bedeutet hier wäre eine vom J48-PG getroffene Vorhersage im Schnitt etwa 1.337 (5.288 − 3.951) Klassen näher zur tatsächlichen Klasse als die des J48-RR.

6. Zusammenfassung und Ausblick

In der vorliegenden Arbeit wurde ein Verfahren vorgestellt, welches den Round-Robin Klassifzierer erweitert, so dass dieser in der Lage sein sollte Informationen bzgl. der Klassenordnung in seiner Vorhersage zu berücksichtigen. Die Vermutung, dass solch eine Erweiterung zu einer Verbesserung der Genauigkeit führen könnte, wurde durch die Experimente nicht bestätigt. Zwar gibt es einige Datensätze in denen das paarweise geordnete Verfahren besser abschneidet, jedoch ist dies bei der Mehrheit der Datensätze nicht der Fall. Für die Basislerner J48 und JRip lässt sich sagen, dass die Genauigkeit des paarweise geordneten Verfahrens minimal unter der des Round-Robin Klassifizierers liegt.

Ein Grund dafür könnte darin liegen, dass durch die Hinzunahme von zu vielen verschiedenen Klassen in die Trainingsmenge, die so gelernten Klassifizierer zu sehr verallgemeinern. Ein mögliches Szenario dazu wäre, dass für ein großes Multiklassenproblem die Trainingsmenge von i nur aus den Instanzen der einen Klasse i besteht und die Trainingsmenge von j aus den Instanzen der restlichen Klassen, welche alle größer i sind. Durch die Menge der unterschiedlichen Klassen ist es durchaus möglich, dass der so trainierte Klassifizierer $K(i, j)$ in der Klassifikationphase allen Instanzen der Wert j zuweist.

Unter Betracht eines anderen Performanzmaßes lässt sich jedoch eine Verbesserung erkennen. Der durchschnittliche mean absolute error des paarweise geordneten Verfahrens liegt unter dem des Round-Robin Klassifizierers. Der Unterschied nimmt mit wachsender Klassenanzahl stetig zu. Dies bedeutet für die Klassifizierung, dass im Schnitt der Abstand zwischen der vorhergesagten Klasse und tatsächlichen Klasse geringer ist. Es ist für ein konkretes Problem zu entscheiden, welche Auswirkungen eine Falschklassifizierung hat und welches Verfahren dadurch zu bevorzugen ist.

Weiterhin ist zu beobachten, dass durch die Hinzunahme von zusätzlichen Instanzen der Aufwand der Trainingsphase des paarweise geordneten Verfahrens stark zunimmt. Eine Möglichkeit diesem entgegen zu steuern wäre, dass nur eine begrenzte Anzahl der nächstgelegenen Nachbarklassen hinzugenommen wird. Anstelle dass man für einen Klassifizierer $K(i, j)$ zusätzlich alle Instanzen kleiner Klasse i als i und Instanzen größer Klasse j als j trainiert werden, könnte man die Menge auf die s nächsten Klassen, mit $1 \leq s \leq c - 1$, begrenzen. Es wäre auch denkbar, dass die Wahl eines kleinen Wert für s zu einer Verbesserung der Genauigkeit führt, da ein auf die Weise trainierter Klassifizierer $K(i, j)$ weniger verallgemeinert. Solch eine Vorhergehensweise wurde in der Arbeit von Cardoso und Costa [1] gewählt, welche erst kurz vor Fertigstellung dieser Arbeit veröffentlicht wurde.

Ein weiteres Gebiet, in dem die Informationen bzgl. der Klassenordnung einbezogen werden könnten, ist das des Rankings. Manche Klassifizierer sind in der Lage nicht nur die wahrscheinlichste Klasse zurückzugeben, sondern eine der Wahrscheinlichkeit nach

sortierte Liste der Klassen. Hier könnten ebenso, die für die Klassenordnung relevanten Instanzen in die Trainingsmenge hinzugefügt werden.

Neben den aufgezeigten Möglichkeiten der Modifikation des Verfahrens bleiben weitere offene Fragen. Alle Auswertungen wurden auf Datensätzen ausgeführt, in denen die Klassenordnung künstlich erzeugt wurde. Es wäre interessant zu sehen, wie sich das Verfahren bei Datensätzen mit einer realen Klassenordnung verhält. Solche Auswertungen wurden hier jedoch nicht durchgeführt, da derartige Datensätze nicht zur Verfügung standen.

Eine weitere Frage betrifft das Resultat der beiden Lernverfahren Naive Bayes und Support Vector Machine. Hier lag die Genauigkeit des paarweise geordneten Verfahrens sowohl unter der des Round-Robin Klassifizierers als auch unter der der unmodifizierten Versionen der Basislerner. Interessant ist, dass die ordinale Klassifikation von Frank und Hall in diesen Fällen vergleichsweise schlecht abgeschnitten hatte. Es bleibt zu untersuchen, weswegen die in dieser Arbeit vorgestellte Hinzunahme von weiteren ordnungsrelevanten Trainingsinstanzen bei den Implementierungen NB und SMO von Weka einen negativen Effekt haben.

Literaturverzeichnis

[1] J. Cardoso and J. P. Costa. Learning to Classify Ordinal Data: The Data Replication Method. *Journal of Machine Learning Research 8 (2007)* 1393-1429

[2] J. S. Cardoso, J. F. Pinto da Costa, and M. J. Cardoso. Modelling ordinal relations with SVMs: an application to objective aesthetic evaluation of breast cancer conservative treatment. *Neural Networks*, 18:808-817, june-july 2005.

[3] W. W. Cohen and Y. Singer. A simple, fast, and effective rule learner. *In Proceedings of the 16th National Conference on Artificial Intelligence (AAAI-99)*, 335–342, Menlo Park, CA, 1999. AAAI/MIT Press.

[4] E. Frank and M. Hall. A simple approach to ordinal classification. In L. D. Raedt and P. Flach, editors, *Proceedings of the 12th European Conference on Machine Learning (ECML-01)*, pages 145-156, Freiburg, Germany, 2001. Springer-Verlag.

[5] J. Fürnkranz. Round Robin Ensembles. Intelligent Data Analysis 7(5), 2003.

[6] J. Fürnkranz. Round Robin Classification. *Journal of Machine Learning Research*, 2:721–747, March 2002.

[7] R. Herbrich, T. Graepel, and K. Obermayer. Regression Models for Ordinal Data: A Machine Learning Approach. TU Berlin, Tech. Rep. TR-99/03, 1999.

[8] S. Kramer, G. Widmer, B. Pfahringer, and M. DeGroeve. Prediction of Ordinal Classes Using Regression Trees. *International Syposium on Methodologies for Intelligent Systems*, pages 426-434, 2000.

[9] M. J. Mathieson. Ordinal models for neural networks. In A.-P.N Refenes, Y. Abu-Mostafa, and J. Moody, editors, *Neural Networks for Financial Engineering*. World Scientific, Singapore, 1995.

[10] T. M. Mitchell. *Machine Learning.* McGraw Hill, 1997.

[11] D. J. Newman, S. Hettich, C. L. Blake and C. J. Merz. *UCI Repository of machine learning databases.* http://www.ics.uci.edu/~mlearn/MLRepository.html. University of California, Irvine, Dept. of Information and Computer Sciences, 1998.

[12] Park, S.-H. (2006). Effiziente Klassifikation und Ranking mit paarweisen Vergleichen. Diplomarbeit, TU-Darmstadt

[13] J. R. Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann, San Mateo, CA, 1993.

[14] A. Shashua and A. Levin. Ranking with large margin principle: Two approaches. In *Neural Information and Processing Systems (NIPS)*, 2002.

[15] I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques, 2nd Edition*. Morgan Kaufmann, San Francisco, 2005.

A. Vergleich der Genauigkeit

Im folgenden werden die Ergebnisse der Experimente präsentiert. Die Tabellen in der oberen Hälfte geben die Genauigkeitswerte der 10-fold Cross-Validation wider. Ein + oder − hinter einem Wert gibt an, ob das jeweilige Verfahren eine höhere oder niedrigere Genauigkeit erzielt hat als das paarweise geordnete Verfahren. Ein ++ oder −− gibt entsprechend an, ob diese Unterschiede signifikant besser oder schlechter waren. Ein = steht bei Gleichheit der Werte dar.

Die Tabellen in der unteren Hälfte stellen den paarweisen Vergleich aller Verfahren dar. Die Zahl gibt an, wie oft das Verfahren in der Spalte besser war als das Verfahren in der Zeile. Die Zahl in Klammern gibt dabei an wie of es davon signifikant besser war.

Bei den Datensätzen *Ailerons* und $Friedman$ mit 10 Klassen hat der in der Testumgebung zur Verfügung stehende Speicher von 1 GB nicht ausgereicht, um die Experimente durchzuführen. Die jeweiligen Felder sind mit einen * markiert.

A.1. 3 Klassen

A.1.1. J48 - 3 Klassen

Datensatz	J48-PG	J48		J48-1VA		J48-RR		J48-FH	
2D Planes	86.69	86.67	−	86.54	−	86.69	=	86.68	−
Abalone	61.57	60.31	−	62.48	+	62.48	+	61.79	+
Ailerons	73.74	72.97	−	71.93	−−	73.16	−	73.88	+
Auto MPG	81.14	80.39	−	83.41	+	81.65	+	80.89	−
Auto Price	86.17	85.58	−	86.79	+	83.67	−	86.79	+
Bank 8FM	86.02	85.79	−	86.06	+	86.38	+	86.04	+
Bank 32NH	56.35	55.65	−	54.20	−−	56.66	+	57.01	+
Boston Housing	77.25	76.07	−	74.86	−	75.67	−	77.05	−
California Housing	78.76	78.84	+	79.08	+	79.35	+	78.91	++
CPU Small	78.05	76.09	−−	77.83	−	77.37	−	78.25	+
CPU Act	80.74	80.62	−	79.97	−	79.82	−	80.90	+
Delta Ailerons	65.37	65.79	+	65.34	−	65.06	−	65.37	=
Delta Elevators	64.24	63.39	−	63.83	−	63.70	−	64.25	+
Diabetes	37.00	44.50	+	37.50	+	41.50	+	37.00	=
Elevators	63.90	62.17	−−	63.87	−	64.29	+	64.46	++
Friedman	80.44	80.16	−	80.18	−	80.36	−	80.57	++
House 8L	70.39	69.32	−−	70.94	+	70.28	−	70.46	+
House 16H	68.92	68.08	−	69.27	+	69.67	+	69.25	++
Kinematics	47.60	43.51	−−	42.85	−−	46.78	−	47.90	+
Machine CPU	72.29	70.86	−	72.29	=	74.19	+	72.29	=
MV Artificial	99.52	99.54	+	99.52	=	99.53	+	99.52	=
Pole Telecom	96.45	96.00	−−	96.32	−	96.44	−	96.46	+
Pumadyn 8NH	66.57	65.92	−	67.63	+	66.49	−	66.60	+
Pumadyn 32NH	78.71	77.94	−	77.60	−−	78.61	−	78.74	++
Pyrimidines	54.11	46.25	−	55.71	+	51.61	−	54.11	=
Servo	76.47	74.78	−	72.32	−	78.90	+	74.74	−
Stocks	90.95	91.37	+	90.95	=	91.16	+	91.05	+
Triazines	51.11	58.65	+	49.42	−	46.17	−	51.61	+
Wisconsin Breast Cancer	35.63	37.26	+	37.18	+	34.00	−	36.18	+

Tabelle A.1.: Ergebnisse der Experimente mit J48 auf Datensätze mit 3 Klassen

	J48-PG	J48	J48-1VA	J48-RR	J48-FH
J48-PG	-	7 (0)	11 (0)	12 (0)	20 (4)
J48	22 (5)	-	14 (4)	20 (5)	21 (5)
J48-1VA	15 (4)	14 (1)	-	18 (3)	16 (5)
J48-RR	15 (0)	13 (1)	11 (1)	-	18 (1)
J48-FH	5 (0)	8 (0)	10 (0)	11 (0)	-

Tabelle A.2.: Paarweiser Vergleich der J48-Methoden auf Datensätze mit 3 Klassen

A.1.2. JRip - 3 Klassen

Datensatz	JRip-PG	JRip		JRip-1VA		JRip-RR		JRip-FH	
2D Planes	86.82	86.62	−	86.55	−−	86.75	−	86.53	−
Abalone	63.73	61.98	−	63.06	−	62.75	−	63.47	+
Ailerons	75.16	71.03	−−	74.85	−	74.76	−	74.88	−
Auto MPG	80.15	81.41	+	79.89	−	81.41	+	80.65	+
Auto Price	83.71	83.67	−	82.42	−	85.58	+	81.88	−
Bank 8FM	87.11	87.01	−	87.46	+	87.00	−	87.01	−
Bank 32NH	61.66	61.16	−	62.41	+	62.22	+	61.89	+
Boston Housing	75.68	75.45	−	76.05	+	76.07	+	73.51	−
California Housing	76.76	74.03	−−	76.99	+	77.86	++	77.01	−
CPU Small	79.28	75.16	−−	79.54	+	78.63	−	78.85	−
CPU Act	81.48	78.31	−−	81.97	+	82.29	+	81.80	+
Delta Ailerons	66.39	65.86	−	66.12	−	65.49	−	66.48	+
Delta Elevators	63.71	63.91	+	63.59	−	63.90	+	63.80	+
Diabetes	48.50	43.50	−	41.50	−	50.50	+	41.00	−
Elevators	64.13	64.66	+	64.26	+	65.65	++	64.41	+
Friedman	81.54	81.59	+	81.82	+	81.43	−	81.67	+
House 8L	71.11	67.86	−−	71.51	+	71.19	+	71.25	+
House 16H	70.76	67.17	−−	70.58	−	71.41	+	70.92	+
Kinematics	45.76	39.34	−−	40.26	−−	47.77	+	46.90	+
Machine CPU	73.71	72.76	−	71.81	−	73.71	=	72.76	−
MV Artificial	99.47	99.46	−	99.52	+	99.48	+	99.51	+
Pole Telecom	95.34	94.61	−	95.41	+	95.62	+	95.82	+
Pumadyn 8NH	67.27	65.95	−	67.70	+	67.05	−	67.00	−
Pumadyn 32NH	81.74	81.47	−	81.71	−	81.15	−	81.45	−
Pyrimidines	42.68	45.27	−	52.86	+	47,68	+	46,07	+
Servo	62.76	70.11	+	66.32	+	79.56	++	65.26	+
Stocks	90.63	91.05	+	89.58	−	90.74	+	91.47	+
Triazines	49.97	50.99	+	46.17	−	51.02	+	52.66	+
Wisconsin Breast Cancer	35.61	38.29	+	35.68	+	35.03	−	37.11	+

Tabelle A.3.: Ergebnisse der Experimente mit JRip auf Datensätze mit 3 Klassen

	JRip-PG	JRip	JRip-1VA	JRip-RR	JRip-FH
JRip-PG	-	9 (0)	15 (0)	17 (3)	18 (0)
JRip	19 (7)	-	14 (1)	20 (9)	18 (8)
JRip-1VA	12 (2)	9 (0)	-	16 (4)	13 (2)
JRip-RR	10 (0)	11 (0)	13 (0)	-	15 (0)
JRip-FH	11 (0)	11 (0)	16 (1)	16 (1)	-

Tabelle A.4.: Paarweiser Vergleich der JRip-Methoden auf Datensätze mit 3 Klassen

A.1.3. SMO - 3 Klassen

Datensatz	SMO-PG	SMO		SMO-1VA		SMO-RR		SMO-FH	
2D Planes	83.21	82.37	−−	68.75	−−	82.36	−−	83.25	++
Abalone	63.04	62.77	−	51.42	−−	63.49	+	63.06	+
Ailerons	76.57	76.59	+	63.56	−−	76.63	+	76.57	=
Auto MPG	78.67	78.17	−	64.37	−−	78.40	−	78.67	=
Auto Price	84.33	84.33	=	64.79	−−	85.58	+	84.33	=
Bank 8FM	88.92	88.93	+	63.05	−−	89.01	+	88.92	=
Bank 32NH	66.03	66.78	+	53.41	−−	66.80	+	66.03	=
Boston Housing	76.07	75.49	−	57.90	−−	76.28	+	76.07	=
California Housing	70.66	70.84	+	57.04	−−	71.22	+	70.66	=
CPU Small	78.70	78.73	+	63.13	−−	79.64	++	78.70	=
CPU Act	82.53	82.60	+	64.01	−−	82.96	+	82.53	=
Delta Ailerons	64.51	64.75	+	51.23	−−	64.62	+	64.53	+
Delta Elevators	64.96	64.89	−	41.45	−−	65.00	+	64.96	=
Diabetes	30.00	42.00	+	29.50	−	49.00	++	30.00	=
Elevators	69.75	70.29	++	56.94	−−	70.52	++	69.79	+
Friedman	72.59	72.68	+	57.33	−−	72.68	+	72.59	=
House 8L	61.81	63.76	++	51.74	−−	63.94	++	61.81	=
House 16H	64.75	66.55	++	52.95	−−	66.79	++	64.76	+
Kinematics	33.61	41.71	++	27.05	−−	41.66	++	33.58	−
Machine CPU	67.98	69.86	+	57.45	−−	75.64	+	67.98	=
MV Artificial	92.44	92.17	−−	78.77	−−	92.21	−−	92.44	=
Pole Telecom	83.31	83.49	+	80.03	−−	83.96	++	83.31	=
Pumadyn 8NH	63.00	64.40	+	55.22	−−	64.37	+	63.00	=
Pumadyn 32NH	33.34	56.03	++	33.34	=	56.07	++	33.34	=
Pyrimidines	52.86	55.89	+	50.71	−	57.50	+	52.86	=
Servo	82.57	84.45	+	77.17	−−	84.45	+	82.57	=
Stocks	90.74	90.74	=	86.11	−−	90.84	+	90.74	=
Triazines	42.87	45.06	+	46.26	+	47.16	+	42.87	=
Wisconsin Breast Cancer	35.58	45.53	+	35.05	−	45.45	+	35.58	=

Tabelle A.5.: Ergebnisse der Experimente mit SMO auf Datensätze mit 3 Klassen

	SMO-PG	SMO	SMO-1VA	SMO-RR	SMO-FH
SMO-PG	-	20 (5)	1 (0)	25 (8)	3 (1)
SMO	6 (2)	-	1 (0)	21 (3)	5 (2)
SMO-1VA	27 (24)	28 (25)	-	29 (26)	27 (24)
SMO-RR	3 (2)	6 (0)	0 (0)	-	3 (2)
SMO-FH	2 (0)	20 (5)	1 (0)	26 (8)	-

Tabelle A.6.: Paarweiser Vergleich der SMO-Methoden auf Datensätze mit 3 Klassen

A.1.4. NB - 3 Klassen

Datensatz	NB-PG	NB	NB-1VA	NB-RR	NB-FH
2D Planes	47.11	55.43 ++	55.33 ++	55.43 ++	55.48 ++
Abalone	57.27	57.70 +	57.84 +	57.84 +	57.22 −
Ailerons	50.39	51.56 ++	51.81 ++	51.69 ++	50.39 =
Auto MPG	72.85	76.38 +	75.86 +	77.13 +	72.85 =
Auto Price	82.46	85.54 +	83.04 +	86.21 +	82.46 =
Bank 8FM	81.96	82.07 +	82.91 −	82.08 +	81.73 −
Bank 32NH	59.23	60.16 ++	60.85 ++	60.16 ++	58.46 −−
Boston Housing	60.86	68.58 ++	67.98 ++	68.77 ++	60.86 =
California Housing	59.20	60.03 ++	59.72 +	60.04 ++	58.41 −−
CPU Small	68.97	71.63 ++	73.54 ++	71.37 ++	68.97 =
CPU Act	71.35	74.61 ++	75.16 ++	74.57 ++	71.35 =
Delta Ailerons	62.07	62.36 +	63.25 ++	62.48 +	62.01 −
Delta Elevators	62.81	62.97 +	63.12 +	63.01 +	62.73 −
Diabetes	51.00	42.50 −	42.50 −	42.50 −	47.50 −
Elevators	53.66	56.12 ++	56.40 ++	56.01 ++	53.65 −
Friedman	75.95	76.04 +	73.51 −−	76.04 +	77.05 ++
House 8L	45.81	46.84 ++	48.43 ++	46.84 ++	46.46 ++
House 16H	57.31	59.26 ++	58.73 +	59.21 ++	57.18 −
Kinematics	37.57	41.86 ++	41.09 ++	41.85 ++	42.83 ++
Machine CPU	72.31	80.90 +	77.98 +	77.07 +	72.31 =
MV Artificial	84.38	86.52 ++	85.08 ++	86.51 ++	83.44 −−
Pole Telecom	57.26	53.45 −−	57.05 −	53.34 −−	57.26 =
Pumadyn 8NH	60.29	63.28 ++	63.40 ++	63.28 ++	63.12 ++
Pumadyn 32NH	51.48	60.07 ++	59.81 ++	60.07 ++	59.69 ++
Pyrimidines	61.79	61.43 −	58.39 −	60.00 −	60.36 −
Servo	65.37	75.48 ++	74.30 ++	75.48 ++	73.79 ++
Stocks	80.63	80.95 +	80.21 −	80.95 +	81.16 +
Triazines	51.52	48.77 −	48.25 −	48.77 −	51.52 =
Wisconsin Breast Cancer	43.34	39.74 −	36.63 −	38.18 −	42.32 −

Tabelle A.7.: Ergebnisse der Experimente mit NB auf Datensätze mit 3 Klassen

	NB-PG	NB	NB-1VA	NB-RR	NB-FH
NB-PG	-	25 (15)	22 (14)	24 (15)	8 (7)
NB	4 (1)	-	11 (4)	9 (0)	7 (2)
NB-1VA	7 (1)	16 (3)	-	16 (3)	9 (2)
NB-RR	5 (1)	10 (0)	12 (5)	-	8 (2)
NB-FH	11 (3)	21 (10)	19 (11)	19 (10)	-

Tabelle A.8.: Paarweiser Vergleich der NB-Methoden auf Datensätze mit 3 Klassen

A.2. 5 Klassen

A.2.1. J48 - 5 Klassen

Datensatz	J48-PG	J48		J48-1VA		J48-RR		J48-FH	
2D Planes	75.49	75.41	−	75.25	+	75.57	+	75.48	−
Abalone	49.08	45.39	−−	48.34	−	48.74	−	48.96	−
Ailerons	58.85	56.18	−−	55.99	−−	58.23	−	58.89	+
Auto MPG	81.14	80.39	−	83.41	+	81.65	+	80.89	−
Auto Price	69.13	54.71	−−	60.92	−	64.08	−	68.50	−
Bank 8FM	73.75	72.81	−	73.33	−	73.68	−	73.69	−
Bank 32NH	38.29	36.22	−−	36.22	−−	38.46	+	38.50	+
Boston Housing	62.67	58.28	−	59.10	−	63.27	+	62.27	−
California Housing	64.63	63.45	−	64.99	+	64.46	−	64.58	−
CPU Small	64.86	62.77	−−	63.71	−	64.58	−	64.98	+
CPU Act	67.58	65.83	−−	66.22	−−	67.09	−	67.82	+
Delta Ailerons	54.33	54.38	+	56.98	++	56.15	+	55.13	+
Delta Elevators	47.07	45.37	−−	47.86	+	48.09	+	48.69	++
Diabetes	21.50	20.50	−	27.00	+	25.00	+	21.50	=
Elevators	48.73	46.08	−−	49.06	+	50.57	++	49.35	++
Friedman	66.45	64.89	−−	64.06	−−	65.83	−	66.51	+
House 8L	53.13	50.16	−−	51.01	−−	52.42	−	53.49	+
House 16H	50.68	49.34	−	50.82	+	52.37	++	51.14	++
Kinematics	47.60	43.51	−−	42.85	−−	46.78	−	47.90	+
Machine CPU	53.52	55.38	−	54.10	+	53.14	−	55.90	+
MV Artificial	99.19	99.14	−	99.22	+	99.19	=	99.20	+
Pole Telecom	94.75	94.61	−	94.37	−	94.66	−	94.82	+
Pumadyn 8NH	49.32	45.56	−−	48.57	−	50.74	+	49.69	+
Pumadyn 32NH	65.53	64.03	−	64.17	−	65.27	−	65.80	+
Pyrimidines	36.25	37.50	+	38.57	+	37.50	+	34.82	−
Servo	58.16	67.72	++	50.96	−	70.04	++	58.16	=
Stocks	87.26	87.16	−	85.16	−	86.21	−	87.37	+
Triazines	37.57	37.49	−	34.80	−	37.02	−	36.96	−
Wisconsin Breast Cancer	20.16	16.95	−	22.66	+	25.74	+	21.18	+

Tabelle A.9.: Ergebnisse der Experimente mit J48 auf Datensätze mit 5 Klassen

	J48-PG	J48	J48-1VA	J48-RR	J48-FH
J48-PG	-	4 (1)	12 (1)	14 (3)	18 (3)
J48	25 (12)	-	19 (6)	25 (13)	26 (13)
J48-1VA	18 (6)	8 (1)	-	22 (8)	22 (7)
J48-RR	15 (0)	3 (0)	7 (0)	-	17 (0)
J48-FH	1 (0)	3 (1)	7 (1)	11 (1)	-

Tabelle A.10.: Paarweiser Vergleich der J48-Methoden auf Datensätze mit 5 Klassen

A.2.2. JRip - 5 Klassen

Datensatz	JRip-PG	JRip	JRip-1VA	JRip-RR	JRip-FH
2D Planes	75.92	75.87 −	75.44 −	75.86 −	75.78 −
Abalone	49.99	48.93 −	48.93 −	50.59 +	50.59 +
Ailerons	59.67	57.05 −−	55.13 −−	59.59 −	59.70 +
Auto MPG	80.15	81.41 +	79.89 −	81.41 +	80.65 +
Auto Price	65.96	56.58 −	62.29 −	64.79 −	64.13 −
Bank 8FM	75.52	73.23 −−	72.51 −−	74.61 −	74.99 −
Bank 32NH	41.38	30.54 −−	36.23 −−	42.66 +	41.93 +
Boston Housing	58.13	60.45 +	58.91 +	61.89 +	61.45 +
California Housing	62.01	50.82 −−	54.12 −−	62.34 +	60.91 −−
CPU Small	65.49	60.13 −−	61.34 −−	66.19 +	65.84 +
CPU Act	69.73	64.22 −−	65.56 −−	69.31 +	69.99 +
Delta Ailerons	54.82	50.22 −−	53.99 −	56.46 +	55.60 +
Delta Elevators	47.49	40.49 −−	40.50 −−	49.34 ++	48.44 +
Diabetes	24.50	31.50 +	27.00 +	36.50 +	27.00 +
Elevators	49.19	48.68 −	48.23 −	51.62 ++	49.51 +
Friedman	67.89	49.08 −−	54.27 −−	66.80 −−	68.03 +
House 8L	53.39	49.32 −−	46.57 −−	53.20 −	53.23 −
House 16H	52.57	49.15 −−	45.24 −−	53.56 ++	53.10 +
Kinematics	45.76	39.34 −−	40.26 −−	47.77 +	46.90 +
Machine CPU	58.38	54.55 −	52.62 −	59.38 +	63.12 +
MV Artificial	99.03	98.87 −	98.77 −−	99.15 +	99.02 −
Pole Telecom	93.54	92.00 −−	92.64 −−	93.75 +	93.51 −
Pumadyn 8NH	49.82	44.09 −−	47.31 −−	50.31 +	50.96 +
Pumadyn 32NH	68.54	66.48 −	65.09 −−	68.31 −	68.53 −
Pyrimidines	38.75	33.04 −	33.39 −	38.57 −	38.57 −
Servo	57.98	57.94 −	59.96 −	68.24 ++	60.55 +
Stocks	85.26	83.68 −	85.37 +	84.63 −	87.68 ++
Triazines	32.13	27.87 −	29.04 −	33.07 +	35.96 +
Wisconsin Breast Cancer	19.05	20.03 +	19.58 +	17.95 −	23.08 +

Tabelle A.11.: Ergebnisse der Experimente mit JRip auf Datensätze mit 5 Klassen

	JRip-PG	JRip	JRip-1VA	JRip-RR	JRip-FH
JRip-PG	-	4 (0)	5 (0)	18 (3)	20 (1)
JRip	25 (13)	-	16 (4)	28 (16)	28 (13)
JRip-1VA	24 (15)	14 (4)	-	29 (16)	30 (18)
JRip-RR	11 (1)	2 (0)	2 (0)	-	21 (2)
JRip-FH	8 (1)	3 (0)	0 (0)	21 (2)	-

Tabelle A.12.: Paarweiser Vergleich der JRip-Methoden auf Datensätze mit 5 Klassen

A.2.3. SMO - 5 Klassen

Datensatz	SMO-PG	SMO		SMO-1VA		SMO-RR		SMO-FH	
2D Planes	67.34	67.62	+	37.18	−−	67.70	+	67.36	+
Abalone	46.71	49.05	++	25.16	−−	49.44	++	46.71	=
Ailerons	60.74	61.41	+	35.83	−−	61.45	+	60.74	=
Auto MPG	78.67	78.17	−	64.37	−−	78.40	−	78.67	=
Auto Price	57.83	52.13	−	34.00	−−	56.50	−	57.83	=
Bank 8FM	78.14	78.00	−	36.67	−−	78.81	++	78.14	=
Bank 32NH	46.00	47.93	++	28.91	−−	48.56	++	46.00	=
Boston Housing	64.25	61.07	−	35.78	−−	61.86	−	64.05	−
California Housing	51.73	52.29	+	30.01	−−	53.45	++	51.73	=
CPU Small	64.53	63.57	−	34.72	−−	65.94	++	64.49	−
CPU Act	70.28	70.17	−	36.51	−−	71.11	++	70.12	−−
Delta Ailerons	50.82	56.74	++	36.89	−−	56.91	++	50.82	=
Delta Elevators	45.79	49.39	++	27.89	−−	49.40	++	45.79	=
Diabetes	21.00	29.50	+	18.50	−	25.00	+	21.00	=
Elevators	51.44	55.18	++	36.41	−−	55.62	++	51.43	−
Friedman	55.22	55.92	++	32.48	−−	55.91	++	55.22	=
House 8L	40.49	46.25	++	29.46	−−	46.56	++	40.48	−
House 16H	36.40	47.65	++	22.29	−−	48.17	++	36.38	−
Kinematics	33.61	41.71	++	27.05	−−	41.66	++	33.58	−
Machine CPU	55.00	51.10	−	32.98	−−	57.31	+	55.00	=
MV Artificial	88.62	91.51	++	42.29	−−	91.49	++	88.62	=
Pole Telecom	80.66	82.49	++	80.01	−	82.78	++	80.66	=
Pumadyn 8NH	33.72	47.47	++	20.00	−−	47.42	++	33.64	−
Pumadyn 32NH	24.43	39.49	++	20.00	−−	39.26	++	24.43	=
Pyrimidines	48.21	48.39	+	31.25	−−	48.57	+	48.21	=
Servo	68.31	74.34	+	59.78	−	74.34	+	64.78	+
Stocks	75.47	79.47	++	54.00	−−	79.68	++	75.47	=
Triazines	30.61	29.53	−	21.49	−	29.53	−	30.09	−
Wisconsin Breast Cancer	23.21	27.79	+	19.58	−	25.16	+	23.21	=

Tabelle A.13.: Ergebnisse der Experimente mit SMO auf Datensätze mit 5 Klassen

	SMO-PG	SMO	SMO-1VA	SMO-RR	SMO-FH
SMO-PG	-	21 (14)	0 (0)	25 (18)	1 (0)
SMO	8 (0)	-	0 (0)	20 (8)	7 (0)
SMO-1VA	29 (24)	29 (27)	-	29 (27)	29 (24)
SMO-RR	4 (0)	7 (0)	0 (0)	-	4 (0)
SMO-FH	10 (1)	22 (14)	0 (0)	25 (18)	-

Tabelle A.14.: Paarweiser Vergleich der SMO-Methoden auf Datensätze mit 5 Klassen

A.2.4. NB - 5 Klassen

Datensatz	NB-PG	NB	NB-1VA	NB-RR	NB-FH
2D Planes	29.96	43.58 ++	43.75 ++	43.58 ++	45.69 ++
Abalone	39.91	41.85 +	43.31 ++	42.04 ++	39.53 −
Ailerons	34.32	35.78 ++	36.64 ++	35.83 ++	34.33 +
Auto MPG	72.85	76.38 +	75.86 +	77.13 +	72.85 =
Auto Price	59.08	56.50 −	54.63 −	57.13 −	59.08 =
Bank 8FM	66.13	67.59 ++	67.54 ++	67.58 ++	65.31 −−
Bank 32NH	41.48	42.19 +	41.94 +	42.18 +	40.36 −−
Boston Housing	41.73	49.05 ++	50.24 ++	49.44 ++	41.73 =
California Housing	41.31	42.09 +	41.29 −	42.10 +	40.50 −−
CPU Small	51.67	58.52 ++	60.22 ++	58.24 ++	51.68 +
CPU Act	53.16	59.77 ++	60.45 ++	59.63 ++	53.16 =
Delta Ailerons	50.41	53.67 ++	54.12 ++	53.77 ++	51.48 ++
Delta Elevators	46.66	47.78 +	48.04 ++	47.95 +	46.95 +
Diabetes	26.50	26.50 =	29.00 +	29.00 +	29.50 +
Elevators	36.40	40.16 ++	41.05 ++	40.21 ++	36.65 ++
Friedman	58.00	58.59 ++	57.09 −−	58.59 ++	60.63 ++
House 8L	29.70	35.63 ++	37.43 ++	35.57 ++	29.93 +
House 16H	40.09	39.25 −	39.54 −	39.22 −	39.98 −
Kinematics	37.57	41.86 ++	41.09 ++	41.85 ++	42.83 ++
Machine CPU	55.55	56.36 +	57.81 +	56.83 +	55.55 =
MV Artificial	71.44	75.32 ++	74.30 ++	75.32 ++	70.42 −−
Pole Telecom	53.73	46.16 −−	52.13 −−	46.11 −−	53.75 +
Pumadyn 8NH	40.83	47.92 ++	48.03 ++	47.91 ++	47.51 ++
Pumadyn 32NH	28.67	43.52 ++	43.40 ++	43.51 ++	42.61 ++
Pyrimidines	40.89	41.79 +	45.71 +	40.54 +	40.89 =
Servo	46.07	54.41 +	55.04 ++	54.41 +	47.21 +
Stocks	57.68	62.95 ++	62.21 ++	62.95 ++	58.11 +
Triazines	34.33	31.08 −	31.61 −	32.69 −	34.33 =
Wisconsin Breast Cancer	28.82	26.21 −	27.76 −	27.24 −	30.34 +

Tabelle A.15.: Ergebnisse der Experimente mit NB auf Datensätze mit 5 Klassen

	NB-PG	NB	NB-1VA	NB-RR	NB-FH
NB-PG	-	23 (15)	22 (17)	25 (16)	16 (7)
NB	5 (1)	-	19 (5)	13 (1)	9 (3)
NB-1VA	7 (2)	10 (4)	-	11 (4)	9 (4)
NB-RR	6 (1)	10 (1)	17 (5)	-	10 (3)
NB-FH	6 (4)	21 (13)	20 (16)	19 (14)	-

Tabelle A.16.: Paarweiser Vergleich der NB-Methoden auf Datensätze mit 5 Klassen

A.3. 10 Klassen

A.3.1. J48 - 10 Klassen

Datensatz	J48-PG	J48	J48-1VA	J48-RR	J48-FH
2D Planes	54.72	52.66 −−	52.02 −−	54.09 −	54.77 +
Abalone	29.09	25.62 −−	25.04 −−	29.69 +	28.97 −
Ailerons	*	*	*	*	*
Auto MPG	38.65	37.94 −	23.10 −−	38.20 −	37.64 −
Auto Price	44.67	38.92 −	35.79 −	35.17 −	50.29 +
Bank 8FM	52.88	49.82 −−	44.09 −−	51.87 −	53.26 +
Bank 32NH	23.94	23.00 −	22.81 −	27.84 ++	23.35 −
Boston Housing	41.50	39.32 −	34.39 −−	43.69 +	42.87 +
California Housing	44.97	42.49 −−	39.64 −−	43.67 −−	44.41 −−
CPU Small	45.57	41.67 −−	40.15 −−	45.73 +	45.53 −
CPU Act	48.99	44.76 −−	45.69 −−	48.40 −	48.93 −
Delta Ailerons	38.76	37.48 −	40.20 +	42.60 ++	39.29 +
Delta Elevators	37.08	33.52 −−	37.76 +	38.78 +	37.64 +
Diabetes	14.50	21.00 +	18.50 +	19.50 +	11.00 −
Elevators	31.19	28.58 −−	29.10 −−	33.00 ++	30.56 −
Friedman	*	*	*	*	*
House 8L	31.69	29.60 −−	28.27 −−	33.96 ++	32.51 ++
House 16H	31.07	28.89 −−	28.15 −−	33.14 ++	30.97 =
Kinematics	25.77	24.30 −	20.35 −−	27.21 +	26.03 +
Machine CPU	36.90	35.40 −	29.69 −	33.90 −	39.81 +
MV Artificial	98.13	98.20 +	97.38 −−	98.02 −	98.17 +
Pole Telecom	91.53	91.29 −	89.46 −−	90.96 −	91.56 +
Pumadyn 8NH	26.87	24.37 −−	16.81 −−	28.17 +	26.87 =
Pumadyn 32NH	46.23	42.96 −−	41.34 −−	45.23 −	45.80 −
Pyrimidines	17.68	22.68 +	14.82 −	18.57 +	19.11 +
Servo	31.76	30.59 −	10.85 −−	34.12 +	31.76 =
Stocks	73.89	73.58 −	73.16 −	71.47 −	74.32 +
Triazines	18.89	17.72 −	12.31 −	18.80 −	22.11 +
Wisconsin Breast Cancer	11.26	13.89 +	10.34 −	12.34 +	9.66 −

Tabelle A.17.: Ergebnisse der Experimente mit J48 auf Datensätze mit 10 Klassen

	J48-PG	J48	J48-1VA	J48-RR	J48-FH
J48-PG	-	4 (0)	3 (0)	15 (5)	14 (1)
J48	23 (12)	-	4 (2)	19 (13)	22 (13)
J48-1VA	24 (17)	23 (10)	-	25 (19)	23(19)
J48-RR	12 (1)	8 (1)	2 (0)	-	12 (3)
J48-FH	12 (1)	5 (0)	4 (0)	15 (5)	-

Tabelle A.18.: Paarweiser Vergleich der J48-Methoden auf Datensätze mit 10 Klassen

A.3.2. JRip - 10 Klassen

Datensatz	JRip-PG	JRip	JRip-1VA	JRip-RR	JRip-FH
2D Planes	55.11	40.72 −−	43.85 −−	54.85 −	55.18 +
Abalone	31.12	24.44 −−	24.42 −−	32.97 +	29.88 −
Ailerons	*	*	*	*	*
Auto MPG	38.70	33.65 −	31.42 −−	35.44 −	39.94 +
Auto Price	43.46	28.92 −	30.79 −	34.54 −	43.38 −
Bank 8FM	53.50	37.65 −−	37.42 −−	51.98 −	54.37 +
Bank 32NH	25.59	23.35 −−	23.16 −−	29.96 ++	25.74 +
Boston Housing	42.29	34.77 −−	31.02 −−	38.53 −	39.90 −
California Housing	41.66	25.72 −−	27.58 −−	41.43 −	41.34 −
CPU Small	46.02	29.30 −−	34.08 −−	45.59 −	46.33 +
CPU Act	48.94	32.31 −−	37.02 −−	49.16 +	50.40 +
Delta Ailerons	39.42	35.22 −−	34.45 −−	43.09 ++	38.48 −
Delta Elevators	36.88	29.49 −−	29.79 −−	39.21 ++	37.43 +
Diabetes	12.00	14.00 +	19.00 +	9.00 −	14.00 +
Elevators	30.45	25.61 −−	23.45 −−	32.15 ++	31.21 +
Friedman	*	*	*	*	*
House 8L	32.00	20.14 −−	22.33 −−	34.16 ++	32.41 +
House 16H	31.88	19.78 −−	21.67 −−	34.24 ++	31.82 −
Kinematics	26.01	17.66 −−	17.30 −−	27.81 +	26.62 +
Machine CPU	36.83	30.14 −	30.67 −	36.86 +	41.17 +
MV Artificial	97.82	97.14 −−	96.82 −−	97.76 −	97.74 −
Pole Telecom	89.27	86.07 −−	87.11 −−	88.90 −	89.41 +
Pumadyn 8NH	28.30	16.76 −−	16.27 −−	27.58 −	27.47 −
Pumadyn 32NH	48.93	35.46 −−	37.26 −−	47.71 −	48.99 +
Pyrimidines	15.00	6.79 −	7.86 −	13.39 −	16.25 +
Servo	28.27	18.05 −−	17.46 −−	31.10 +	30.00 +
Stocks	72.95	66.74 −−	68.95 −	70.74 −	73.58 +
Triazines	15.15	11.81 −	14.47 −	12.98 −	20.38 +
Wisconsin Breast Cancer	11.84	9.79 −	9.79 −	12.82 +	7.66 −

Tabelle A.19.: Ergebnisse der Experimente mit JRip auf Datensätze mit 10 Klassen

	JRip-PG	JRip	JRip-1VA	JRip-RR	JRip-FH
JRip-PG	-	1 (0)	1 (0)	11 (6)	18 (0)
JRip	26 (20)	-	15 (5)	26 (18)	25 (22)
JRip-1VA	26 (20)	11 (2)	-	25 (19)	25 (21)
JRip-RR	15 (0)	1 (0)	2 (0)	-	14 (2)
JRip-FH	9 (0)	1 (0)	2 (0)	13 (4)	-

Tabelle A.20.: Paarweiser Vergleich der JRip-Methoden auf Datensätze mit 10 Klassen

A.3.3. SMO - 10 Klassen

Datensatz	SMO-PG	SMO	SMO-1VA	SMO-RR	SMO-FH
2D Planes	46.84	46.77 −	17.95 −−	46.47 −	46.83 −
Abalone	28.97	29.97 +	10.73 −−	30.91 +	28.97 =
Ailerons	*	*	*	*	*
Auto MPG	36.15	32.90 −	8.31 −−	36.67 +	36.40 +
Auto Price	30.25	23.83 −	13.83 −−	30.88 +	29.58 −
Bank 8FM	51.84	49.43 −−	17.83 −−	59.05 ++	51.84 =
Bank 32NH	27.66	32.42 ++	22.20 −−	32.62 ++	27.66 =
Boston Housing	38.56	34.96 −	13.84 −−	39.10 +	38.76 +
California Housing	27.27	31.11 ++	13.39 −−	31.48 ++	27.27 =
CPU Small	43.90	43.23 −	16.26 −−	46.83 ++	43.90 =
CPU Act	49.39	50.06 +	16.36 −−	51.48 ++	49.39 =
Delta Ailerons	34.59	42.35 ++	9.38 −−	42.53 ++	34.55 −
Delta Elevators	37.59	39.28 ++	6.90 −−	39.41 ++	37.59 =
Diabetes	14.50	14.00 −	11.00 −	11.50 −	14.50 =
Elevators	32.48	35.80 ++	15.74 −−	36.79 ++	32.48 =
Friedman	*	*	*	*	*
House 8L	21.29	26.40 ++	16.20 −−	27.05 ++	21.29 =
House 16H	18.83	27.65 ++	13.88 −−	28.52 ++	18.83 =
Kinematics	16.44	24.22 ++	10.00 −−	24.17 ++	16.46 +
Machine CPU	26.81	25.38 −	15.31 −−	33.00 +	26.81 =
MV Artificial	79.14	86.61 ++	16.85 −−	87.05 ++	79.13 −
Pole Telecom	77.55	81.22 ++	80.03 ++	81.19 ++	77.55 =
Pumadyn 8NH	18.01	27.39 ++	10.00 −−	27.32 ++	17.98 −
Pumadyn 32NH	11.40	23.73 ++	10.00 −−	23.90 ++	11.40 =
Pyrimidines	33.57	23.93 −	14.64 −	25.36 −	32.14 −
Servo	44.34	48.64 +	20.33 −−	47.94 +	44.89 +
Stocks	50.00	57.89 ++	19.79 −−	62.53 ++	49.47 −
Triazines	16.20	12.92 −	11.29 −	16.67 +	16.20 =
Wisconsin Breast Cancer	11.29	15.39 +	9.26 −	14.39 +	11.29 =

Tabelle A.21.: Ergebnisse der Experimente mit SMO auf Datensätze mit 10 Klassen

	SMO-PG	SMO	SMO-1VA	SMO-RR	SMO-FH
SMO-PG	-	17 (13)	0 (0)	24 (16)	4 (0)
SMO	10 (1)	-	0 (0)	20 (8)	10 (1)
SMO-1VA	26 (22)	27 (23)	-	27 (23)	26 (22)
SMO-RR	3 (0)	7 (0)	0 (0)	-	3 (0)
SMO-FH	7 (0)	17 (13)	0 (0)	24 (16)	-

Tabelle A.22.: Paarweiser Vergleich der SMO-Methoden auf Datensätze mit 10 Klassen

A.3.4. NB - 10 Klassen

Datensatz	NB-PG	NB	NB-1VA	NB-RR	NB-FH
2D Planes	16.78	29.16 ++	28.92 ++	29.16 ++	31.93 ++
Abalone	21.57	29.83 ++	30.45 ++	30.09 ++	21.83 +
Ailerons	*	*	*	*	*
Auto MPG	29.91	34.67 +	33.67 +	35.67 +	29.66 −
Auto Price	31.46	35.25 +	32.75 +	33.96 +	30.83 −
Bank 8FM	43.84	44.01 +	43.36 −	44.01 +	41.98 −−
Bank 32NH	28.49	28.99 +	29.42 +	28.98 +	29.06 +
Boston Housing	22.33	33.40 ++	33.80 ++	33.80 ++	22.34 +
California Housing	24.83	24.23 −	23.95 −−	24.23 −	23.54 −−
CPU Small	32.56	39.39 ++	39.76 ++	38.88 ++	32.53 −
CPU Act	32.49	39.47 ++	39.58 ++	39.62 ++	32.49 −
Delta Ailerons	33.02	40.23 ++	40.68 ++	40.19 ++	37.34 ++
Delta Elevators	34.20	39.12 ++	39.38 ++	39.05 ++	36.38 ++
Diabetes	21.00	23.00 +	23.00 +	18.50 −	19.00 −
Elevators	19.31	23.47 ++	23.69 ++	23.30 ++	19.07 −
Friedman	*	*	*	*	*
House 8L	19.55	21.40 ++	22.19 ++	21.31 ++	19.63 +
House 16H	23.56	23.56	23.63 +	23.67 +	22.56 −−
Kinematics	19.24	24.05 ++	23.90 ++	24.01 ++	25.65 ++
Machine CPU	32.02	37.33 +	37.36 +	36.86 +	32.50 +
MV Artificial	49.79	58.94 ++	57.95 ++	58.95 ++	48.64 −−
Pole Telecom	51.77	40.57 −−	48.82 −−	40.37 −−	51.79 +
Pumadyn 8NH	21.00	27.30 ++	27.42 ++	27.28 ++	27.47 ++
Pumadyn 32NH	12.81	25.28 ++	25.04 ++	25.27 ++	24.69 ++
Pyrimidines	20.18	31.07 +	31.07 +	33.75 +	21.61 +
Servo	33.71	38.38 +	37.79 +	38.38 +	36.62 +
Stocks	36.95	48.84 ++	49.16 ++	48.42 ++	37.16 +
Triazines	13.98	9.12 −	9.12 −	10.18 −	13.98 =
Wisconsin Breast Cancer	13.42	15.97 +	14.92 +	15.97 +	14.42 +

Tabelle A.23.: Ergebnisse der Experimente mit NB auf Datensätze mit 10 Klassen

	NB-PG	NB	NB-1VA	NB-RR	NB-FH
NB-PG	-	23 (14)	23 (14)	23 (14)	16 (6)
NB	3 (1)	-	13 (2)	8 (0)	7 (3)
NB-1VA	4 (2)	10 (3)	-	14 (2)	5 (3)
NB-RR	4 (1)	13 (0)	13 (2)	-	8 (3)
NB-FH	9 (4)	21 (12)	22 (11)	20 (13)	-

Tabelle A.24.: Paarweiser Vergleich der NB-Methoden auf Datensätze mit 10 Klassen

B. Vergleich des mean absolute errors

Im folgenden werden die Ergebnisse der mean absolute error Auswertungen präsentiert. Ein + oder − zeigt an, ob der mean absolute error des Round-Robin Verfahrens größer oder kleiner war, als der des paarweise geordneten Verfahrens. Sind beide Werte gleich, wird dies durch ein = markiert. Einige wenige Datensätze konnten aufgrund ihres Umfangs nicht getestet werden. Die entsprechenden Felder sind durch ein * gekennzeichnet.

B.1. 3 Klassen

B.1.1. J48 - 3 Klassen

Datensatz	J48-PG	J48-RR	
2D Planes	0.133	0.133	=
Abalone	0.416	0.422	+
Ailerons	0.272	0.280	+
Auto MPG	0.201	0.196	−
Auto Price	0.164	0.182	+
Bank 8FM	0.141	0.137	−
Bank 32NH	0.494	0.510	+
Boston Housing	0.245	0.267	+
California Housing	0.223	0.219	−
CPU Small	0.227	0.235	+
CPU Act	0.198	0.207	+
Delta Ailerons	0.349	0.352	+
Delta Elevators	0.360	0.366	+
Diabetes	0.791	0.837	+
Elevators	0.404	0.416	+
Friedman Artificial	0.198	0.199	+
House 8L	0.314	0.320	+
House 16H	0.333	0.332	−
Kinematics	0.394	0.407	+
Machine CPU	0.297	0.278	−
MV Artificial	0.005	0.005	=
Pole Telecom	0.036	0.036	=
Pumadyn 8NH	0.351	0.356	+
Pumadyn 32H	0.216	0.220	+
Pyrimidines	0.500	0.541	+
Servo	0.257	0.263	+
Stocks	0.095	0.095	=
Triazines	0.586	0.672	+
Wisconsin Breast Cancer	0.820	0.799	−

Tabelle B.1.: Mean absolute error bei J48 auf Datensätze 3 Klassen

B.1.2. JRip - 3 Klassen

Datensatz	JRip-PG	JRip-RR	
2D Planes	0.132	0.133	+
Abalone	0.392	0.421	+
Ailerons	0.255	0.262	+
Auto MPG	0.211	0.201	−
Auto Price	0.182	0.164	−
Bank 8FM	0.129	0.130	+
Bank 32NH	0.414	0.426	+
Boston Housing	0.257	0.253	−
California Housing	0.240	0.235	−
CPU Small	0.214	0.222	+
CPU Act	0.190	0.181	−
Delta Ailerons	0.339	0.348	+
Delta Elevators	0.366	0.364	−
Diabetes	0.651	0.674	+
Elevators	0.395	0.399	+
Friedman Artificial	0.186	0.187	+
House 8L	0.306	0.312	+
House 16H	0.310	0.313	+
Kinematics	0.257	0.389	+
Machine CPU	0.287	0.287	=
MV Artificial	0.005	0.005	=
Pole Telecom	0.047	0.044	−
Pumadyn 8NH	0.342	0.351	+
Pumadyn 32H	0.184	0.189	+
Pyrimidines	0.622	0.622	=
Servo	0.395	0.234	−
Stocks	0.099	0.098	−
Triazines	0.570	0.634	+
Wisconsin Breast Cancer	0.799	0.835	+

Tabelle B.2.: Mean absolute error bei JRip auf Datensätze 3 Klassen

B.1.3. SMO - 3 Klassen

Datensatz	SMO-PG	SMO-RR	
2D Planes	0.168	0.182	+
Abalone	0.393	0.405	+
Ailerons	0.238	0.238	=
Auto MPG	0.221	0.224	+
Auto Price	0.176	0.164	−
Bank 8FM	0.111	0.110	−
Bank 32NH	0.360	0.368	+
Boston Housing	0.251	0.251	=
California Housing	0.303	0.302	−
CPU Small	0.219	0.211	−
CPU Act	0.178	0.175	−
Delta Ailerons	0.358	0.358	=
Delta Elevators	0.352	0.352	=
Diabetes	0.791	0.767	−
Elevators	0.321	0.325	+
Friedman Artificial	0.285	0.286	+
House 8L	0.411	0.416	+
House 16H	0.389	0.388	−
Kinematics	0.441	0.472	+
Machine CPU	0.335	0.268	−
MV Artificial	0.076	0.078	+
Pole Telecom	0.171	0.169	−
Pumadyn 8NH	0.394	0.396	+
Pumadyn 32H	0.667	0.615	−
Pyrimidines	0.527	0.500	−
Servo	0.204	0.186	−
Stocks	0.099	0.095	−
Triazines	0.629	0.661	+
Wisconsin Breast Cancer	0.691	0.716	+

Tabelle B.3.: Mean absolute error bei SMO auf Datensätze 3 Klassen

B.1.4. NB - 3 Klassen

Datensatz	NB-PG	NB-RR	
2D Planes	0.617	0.615	−
Abalone	0.518	0.481	−
Ailerons	0.654	0.595	−
Auto MPG	0.281	0.236	−
Auto Price	0.195	0.157	−
Bank 8FM	0.182	0.180	−
Bank 32NH	0.453	0.449	−
Boston Housing	0.462	0.356	−
California Housing	0.472	0.468	−
CPU Small	0.334	0.294	−
CPU Act	0.300	0.258	−
Delta Ailerons	0.384	0.380	−
Delta Elevators	0.375	0.372	−
Diabetes	0.581	0.721	+
Elevators	0.584	0.535	−
Friedman Artificial	0.243	0.243	=
House 8L	0.708	0.689	−
House 16H	0.501	0.471	−
Kinematics	0.454	0.496	+
Machine CPU	0.311	0.254	−
MV Artificial	0.156	0.135	−
Pole Telecom	0.611	0.614	+
Pumadyn 8NH	0.419	0.417	−
Pumadyn 32H	0.565	0.539	−
Pyrimidines	0.459	0.473	+
Servo	0.371	0.281	−
Stocks	0.259	0.254	−
Triazines	0.656	0.661	+
Wisconsin Breast Cancer	0.732	0.778	+

Tabelle B.4.: Mean absolute error bei NB auf Datensätze 3 Klassen

B.2. 5 Klassen

B.2.1. J48 - 5 Klassen

Datensatz	J48-PG	J48-RR	
2D Planes	0.246	0.245	−
Abalone	0.668	0.714	+
Ailerons	0.483	0.508	+
Auto MPG	0.417	0.465	+
Auto Price	0.346	0.409	+
Bank 8FM	0.267	0.271	+
Bank 32NH	0.845	0.908	+
Boston Housing	0.447	0.445	−
California Housing	0.405	0.420	+
CPU Small	0.388	0.398	+
CPU Act	0.350	0.360	+
Delta Ailerons	0.551	0.569	+
Delta Elevators	0.627	0.660	+
Diabetes	1.233	1.279	−
Elevators	0.687	0.734	+
House 8L	0.575	0.604	+
House 16H	0.607	0.607	=
Friedman Artificial	0.352	0.363	+
Kinematics	0.679	0.728	+
Machine CPU	0.550	0.579	+
MV Artificial	0.008	0.008	=
Pole Telecom	0.157	0.156	−
Pumadyn 8NH	0.628	0.659	+
Pumadyn 32H	0.371	0.379	+
Pyrimidines	0.932	0.959	+
Servo	0.497	0.419	−
Stocks	0.134	0.144	+
Triazines	0.882	1.097	+
Wisconsin Breast Cancer	1.253	1.335	+

Tabelle B.5.: Mean absolute error bei J48 auf Datensätze 5 Klassen

B.2.2. JRip - 5 Klassen

Datensatz	JRip-PG	JRip-RR	
2D Planes	0.242	0.242	=
Abalone	0.649	0.679	+
Ailerons	0.463	0.485	+
Auto MPG	0.440	0.455	+
Auto Price	0.384	0.415	+
Bank 8FM	0.250	0.260	+
Bank 32NH	0.766	0.825	+
Boston Housing	0.492	0.451	−
California Housing	0.434	0.451	+
CPU Small	0.372	0.376	+
CPU Act	0.323	0.329	+
Delta Ailerons	0.535	0.557	+
Delta Elevators	0.622	0.640	+
Diabetes	1.140	0.977	−
Elevators	0.685	0.727	+
Friedman Artificial	*	0.349	
House 8L	0.568	0.591	+
House 16H	0.575	0.593	+
Kinematics	0.672	0.725	+
Machine CPU	0.522	0.498	−
MV Artificial	0.010	0.009	−
Pole Telecom	0.148	0.150	+
Pumadyn 8NH	0.623	0.657	+
Pumadyn 32H	0.329	0.334	+
Pyrimidines	0.865	0.973	+
Servo	0.509	0.485	−
Stocks	0.257	0.208	−
Triazines	1.043	1.204	+
Wisconsin Breast Cancer	1.149	1.340	+

Tabelle B.6.: Mean absolute error bei JRip auf Datensätze 5 Klassen

B.2.3. SMO - 5 Klassen

Datensatz	SMO-PG	SMO-RR	
2D Planes	0.343	0.342	−
Abalone	0.677	0.757	+
Ailerons	0.442	0.462	+
Auto MPG	0.377	0.362	−
Auto Price	0.478	0.497	+
Bank 8FM	0.221	0.215	−
Bank 32NH	0.666	0.711	+
Boston Housing	0.407	0.476	+
California Housing	0.565	0.573	+
CPU Small	0.391	0.383	−
CPU Act	0.320	0.312	−
Delta Ailerons	0.558	0.569	+
Delta Elevators	0.614	0.651	+
Diabetes	1.279	1.372	+
Elevators	0.589	0.649	+
Friedman Artificial	0.285	0.286	+
House 8L	0.753	0.797	+
House 16H	0.785	0.768	−
Kinematics	0.861	0.889	+
Machine CPU	0.517	0.512	−
MV Artificial	0.117	0.096	−
Pole Telecom	0.114	0.093	−
Pumadyn 8NH	0.764	0.772	+
Pumadyn 32H	1.049	1.224	+
Pyrimidines	0.770	0.797	+
Servo	0.419	0.359	−
Stocks	0.257	0.208	−
Triazines	0.104	1.204	+
Wisconsin Breast Cancer	1.149	1.340	+

Tabelle B.7.: Mean absolute error bei SMO auf Datensätze 5 Klassen

B.2.4. NB - 5 Klassen

Datensatz	NB-PG	NB-RR	
2D Planes	1.076	1.131	+
Abalone	0.977	0.888	−
Ailerons	1.346	1.147	−
Auto MPG	0.533	0.407	−
Auto Price	0.478	0.478	=
Bank 8FM	0.352	0.337	−
Bank 32NH	0.847	0.874	+
Boston Housing	0.901	0.678	−
California Housing	0.896	0.919	+
CPU Small	0.617	0.481	−
CPU Act	0.571	0.458	−
Delta Ailerons	0.615	0.626	+
Delta Elevators	0.647	0.659	+
Diabetes	1.279	1.372	+
Elevators	1.169	0.992	−
Friedman Artificial	0.453	0.472	+
House 8L	1.379	1.063	−
House 16H	0.910	0.898	−
Kinematics	0.820	0.921	+
Machine CPU	0.636	0.531	−
MV Artificial	0.303	0.261	−
Pole Telecom	0.106	0.143	+
Pumadyn 8NH	0.753	0.803	+
Pumadyn 32H	0.995	1.025	+
Pyrimidines	0.919	0.919	=
Servo	0.713	0.707	−
Stocks	0.582	0.517	−
Triazines	1.172	1.091	−
Wisconsin Breast Cancer	1.381	1.371	−

Tabelle B.8.: Mean absolute error bei NB auf Datensätze 5 Klassen

B.3. 10 Klassen

B.3.1. J48 - 10 Klassen

Datensatz	J48-PG	J48-RR	
2D Planes	0.503	0.516	+
Abalone	1.295	1.420	+
Ailerons	0.862	0.938	+
Auto MPG	0.884	0.930	+
Auto Price	0.736	0.887	+
Bank 8FM	0.545	0.569	+
Bank 32NH	1.716	1.869	+
Boston Housing	0.877	0.953	+
California Housing	0.809	0.899	+
CPU Small	0.755	0.782	+
CPU Act	0.662	0.696	+
Delta Ailerons	0.911	0.956	+
Delta Elevators	0.871	0.923	+
Diabetes	2.093	2.581	+
Elevators	1.217	1.349	+
Friedman	0.689	0.756	+
House 8L	1.201	1.272	+
House 16H	1.201	1.293	+
Kinematics	1.349	1.598	+
Machine CPU	1.048	1.134	+
MV Artificial	0.019	0.020	+
Pole Telecom	0.096	0.111	+
Pumadyn 8NH	1.302	1.500	+
Pumadyn 32H	0.723	0.767	+
Pyrimidines	1.811	1.959	+
Servo	1.222	1.144	−
Stocks	0.283	0.325	+
Triazines	1.839	2.366	+
Wisconsin Breast Cancer	2.562	3.026	+

Tabelle B.9.: Mean absolute error bei J48 auf Datensätze 10 Klassen

B.3.2. JRip - 10 Klassen

Datensatz	JRip-PG	JRip-RR	
2D Planes	0.497	0.503	+
Abalone	1.239	1.370	+
Ailerons	0.842	0.914	+
Auto MPG	0.894	0.992	+
Auto Price	0.767	0.893	+
Bank 8FM	0.529	0.563	+
Bank 32NH	1.607	1.846	+
Boston Housing	0.875	1.049	+
California Housing	*	*	
CPU Small	*	0.674	
CPU Act	*	*	
Delta Ailerons	0.884	0.949	+
Delta Elevators	0.866	0.915	+
Diabetes	2.186	2.581	+
Elevators	1.248	1.392	+
Friedman	*	0.745	
House 8L	*	1.271	
House 16H	*	1.286	
Kinematics	1.328	1.563	+
Machine CPU	0.990	1.153	+
MV Artificial	0.022	0.023	+
Pole Telecom	0.130	0.147	+
Pumadyn 8NH	1.288	1.440	+
Pumadyn 32H	0.670	0.707	+
Pyrimidines	1.757	2.149	+
Servo	1.371	1.192	−
Stocks	0.296	0.346	+
Triazines	2.075	2.527	+
Wisconsin Breast Cancer	2.418	2.861	+

Tabelle B.10.: Mean absolute error bei JRip auf Datensätze 10 Klassen

B.3.3. SMO - 10 Klassen

Datensatz	SMO-PG	SMO-RR	
2D Planes	0.722	0.781	+
Abalone	1.306	1.470	+
Ailerons	0.807	0.836	+
Auto MPG	0.837	0.869	+
Auto Price	0.893	1.025	+
Bank 8FM	0.516	0.448	−
Bank 32NH	1.435	1.691	+
Boston Housing	0.907	1.036	+
California Housing	1.192	1.192	=
CPU Small	0.789	0.810	+
CPU Act	0.660	0.656	−
Delta Ailerons	0.944	1.060	+
Delta Elevators	0.863	0.994	+
Diabetes	2.302	2.512	+
Elevators	1.070	1.173	+
Friedman	1.098	1.173	+
House 8L	1.600	1.918	+
House 16H	1.613	1.822	+
Kinematics	1.729	1.925	+
Machine CPU	1.105	1.096	−
MV Artificial	0.237	0.168	−
Pole Telecom	0.157	0.148	−
Pumadyn 8NH	1.545	1.703	+
Pumadyn 32H	2.202	2.648	+
Pyrimidines	1.270	1.878	+
Servo	0.874	0.892	+
Stocks	0.578	0.422	−
Triazines	2.032	2.484	+
Wisconsin Breast Cancer	2.304	2.747	+

Tabelle B.11.: Mean absolute error bei SMO auf Datensätze 10 Klassen

B.3.4. NB - 10 Klassen

Datensatz	NB-PG	NB-RR	
2D Planes	2.191	2.390	+
Abalone	2.440	1.636	−
Ailerons	3.083	2.228	−
Auto MPG	1.281	0.952	−
Auto Price	1.113	0.893	−
Bank 8FM	0.729	0.730	+
Bank 32NH	1.818	1.970	+
Boston Housing	1.988	1.366	−
California Housing	1.872	2.002	+
CPU Small	1.289	0.994	−
CPU Act	1.232	0.953	−
Delta Ailerons	1.051	1.047	−
Delta Elevators	0.926	0.908	−
Diabetes	2.233	2.395	+
Elevators	2.324	1.933	−
Friedman	0.951	1.097	+
House 8L	2.996	2.631	−
House 16H	2.128	2.002	−
Kinematics	1.692	2.088	+
Machine CPU	1.335	1.086	−
MV Artificial	0.660	0.509	−
Pole Telecom	1.910	1.925	+
Pumadyn 8NH	1.530	1.778	+
Pumadyn 32H	2.051	2.174	+
Pyrimidines	1.959	1.770	−
Servo	1.461	1.228	−
Stocks	1.258	0.878	−
Triazines	2.704	2.473	−
Wisconsin Breast Cancer	2.995	2.876	−

Tabelle B.12.: Mean absolute error bei NB auf Datensätze 10 Klassen