Kooperative Klassifikation
Seminar Medizinische Bildverarbeitung
in Hellenthal vom 28.1.1999 – 29.1.1999

von Udo Sorges

Inhaltsverzeichnis

1 Einleitung **2**
1.1 Einführung in die Objektklassifikation . . . 2
1.2 Kombination von Klassifikatoren . . . 3

2 Kombinationsregeln **4**
2.1 Bayessche Entscheidungsregel für minimale Fehlerrate . . . 4
2.2 Produkt- und Summenregel . . . 5
2.3 Weitere Kombinationsregeln . . . 7
2.3.1 Die Max-Regel . . . 7
2.3.2 Die Min-Regel . . . 8
2.3.3 Die Median-Regel . . . 8
2.3.4 Die Majority-Vote Regel . . . 8
2.4 Experimente mit den Kombinationsregeln . . . 9
2.5 Einfluß von Schätzfehlern . . . 10

3 Bagging und Boosting **12**
3.1 Bagging . . . 12
3.2 Boosting . . . 12
3.3 Experimente mit Bagging, Boosting und C4.5 . . . 13
3.3.1 Verfeinerung von AdaBoost . . . 15

4 Fazit **16**

Zusammenfassung

In dieser Arbeit wird die Idee der Kombination von Klassifikatoren im Bereich der Mustererkennung vorgestellt. Zum einen werden verschiedene Kombinationsregeln, die die Outputs mehrerer Einzelklassifikatoren koppeln, theoretisch hergeleitet. Zum anderen werden zwei Verfahren, Bagging und Boosting, vorgestellt, die nicht nur Einzelklassifikatoren kombinieren, sondern diese selbst konstruieren. In verschiedenen Experimenten zeigt sich die Leistungsfähigkeit dieser Verfahren. Prinzipiell kann durch die Kombination mehrerer Klassifikatoren eine Verbesserung erzielt werden, eine generelle Leistungssteigerung ist aber nicht garantiert.

Keywords: Klassifikation, Kombinationsregeln, Bagging, Boosting

1 Einleitung

Im Bereich der Klassifikation von Mustern, einem Teilgebiet der Mustererkennung, ist das Ziel die Entwicklung eines Klassifikators, der für die zu bewältigende Aufgabe die beste Leistung erbringt. Hierzu wird im Allgemeinen ein einzelnes Klassifikationsverfahren angewendet, das auf die bestehende Aufgabe hin spezialisiert wird. Die Idee einzelne Klassifikatoren zu kombinieren und eine gemeinsame Entscheidung herbeizuführen, soll nun hier vorgestellt werden. Der Vorteil, den man sich von dieser Methode erhofft, ist eine Leistungssteigerung gegenüber der Nutzung von Einzelklassifikatoren.

Bevor das Prinzip der kooperativen Klassifikation vorgestellt wird, gibt es eine kurze, ganz allgemeine Einführung zum Thema Objektklassifikation. In Kapitel 2 wird dann der theoretische Hintergrund einiger Kombinationsregeln beleuchtet. Diese Regeln werden in der Praxis oft rein intuitiv angewendet, ohne daß sich der Nutzer darüber viele Gedanken macht. Im 3. Kapitel werden die zwei Verfahren Bagging und Boosting gezeigt, die Einzelklassifikatoren nicht nur gemeinsam über eine Klassenzugehörigkeit bestimmen lassen, sondern diese Einzelklassifikatoren auch selbst konstruieren. Die Leistungsfähigkeit der verschiedenen Kombinationsverfahren läßt sich aus einigen durchgeführten Experimenten ablesen und zeigt zum Teil tatsächlich eine Verbesserung der Klassifikationsergebnisse.

1.1 Einführung in die Objektklassifikation

Im Allgemeinen werden bei einer Klassifikation Objekte entsprechend ihrer Eigenschaften verschiedenen Objektgruppen zugeordnet. Jede Gruppe repräsentiert eine Menge von ähnlichen Eigenschaften. Niemann [1] definiert den Begriff der Klassifikation wie folgt:

> *Eine Klassifikation ist die Zuordnung eines Objekts/Musters zu genau einer von mehreren möglichen Klassen. Die Rückweisung eines Musters, also die Zuordnung zu keiner der Klassen, ist zulässig.*

Diese Definition ist nur eine von vielen möglichen Umschreibungen für die Objektklassifikation. Die Zulässigkeit einer Rückweisung ist beispielsweise ein sehr umstrittener Punkt. Oft wird diese Vorgehensweise untersagt und als Fehlklassifikation interpretiert. Generell erfolgt die Klassifikation eines Objekts oder Musters Z durch einen sogenannten Klassifikator, ein Verfahren, das über die Zugehörigkeit des Objekts zu einer Klasse ω_k entscheidet.

$$x \longrightarrow \boxed{\text{Klassifikator}} \longrightarrow \omega_k$$

Als Input erhält der Klassifikator einen sogenannten Merkmalsvektor x, der die Eigenschaften bzw. Merkmale des Musters repräsentiert. Anhand dieses Merkmalsvektors entscheidet das Klassifikationsverfahren darüber, zu welcher von m möglichen Klassen $\omega_1, ..., \omega_m$ das Objekt zugeordnet wird. Die Klassenzugehörigkeit eines Musters kann vom Klassifikator entweder relativ oder absolut angegeben werden. D.h. der Klassifikator gibt entweder eine Wahrscheinlichkeit für die Zugehörigkeit eines Objekts zu jeder Klasse oder er liefert als Output eine Klassenbezeichnung selbst.

Der Merkmalsvektor besteht aus verschiedenen Merkmalen bzw. Eigenschaften des Objekts. Diese Merkmale spannen den Merkmalsraum auf, in dem jedes Objekt durch einen Punkt beschrieben werden kann. Die Klassen werden durch Gruppierungen im Merkmalsraum bestimmt, zu denen durch ein Ähnlichkeitsmaß eine Zugehörigkeit definiert ist. Ein Klassifikator ist also eine Diskriminantenfunktion, die ein Muster Z bzw. dessen Merkmalsvektor x auf eine der möglichen Klassen $\omega_1, ..., \omega_m$ abbildet.

Um die obigen Ausführungen über die Grundlagen der Objektklassifikation zu verdeutlichen sind nun einige Anwendungsbeispiele aufgeführt. Die vermutlich bekannteste Problematik der Mustererkennung in Zusammenhang mit der Klassifikation von Objekten ist wohl die Erkennung von handgeschriebenen Adressen auf Postsendungen. Insbesondere die Verarbeitung der Postleitzahlen zeigt den Bezug zur Klassifikation. Hier sind die Objekte gerade die Ziffern 0 bis 9 und diese sollen in die Klasse 0 bis Klasse 9 klassifiziert werden. Weiter Anwendungsbereiche sind die Handschriftenerkennung ganz allgemein oder auch die Erkennung (isoliert) gesprochener Worte, wobei bei Letzterem die zu klassifizierenden Objekte die Audiosignale der Worte sind und diese Klassen zugeordnet werden sollen, die gerade den Worten entsprechen. Im Bereich der Medizin kann man sagen, daß jede Diagnose, die ein Arzt stellt eine Klassifikation ist. Der Patient mit seinen Krankheitssymptomen, die den Merkmalen entsprechen, soll in eine der Klassen *krank* oder *gesund* eingeordnet werden, wobei die Klasse *krank* differenzierter betrachtet werden kann, also in verschiedene Krankheiten unterteilt. Eine konkrete Anwendung in der Medizin ist beispielsweise ein *Diagnoseunterstützungssystem*.

1.2 Kombination von Klassifikatoren

Nachdem das Grundprinzip der Klassifikation von Objekten verdeutlicht wurde, soll nun die Idee der Kombination von Klassifikatoren vorgestellt werden. Zunächst allerdings noch einen Blick auf die schon erwähnte traditionelle Vorgehensweise bei der Entwicklung eines Klassifikators.

Bei der Entwicklung eines Klassifikators für eine bestehende Aufgabe werden in der Regel mehrere verschiedene Klassifikatoren entworfen, aus denen dann der mit der besten Leistung ausgewählt wird. Im Detail sieht es so aus, daß es für das zu lösende Klassifikationsproblem eine Menge von Trainingsdaten gibt, d.h. es sind diverse Muster vorhanden, repräsentiert durch je einen Merkmalsvektor, mit denen die unterschiedlichen Klassifikationsverfahren trainieren werden. Um einen Leistungsvergleich zwischen den Verfahren durchzuführen, werden alle entwickelten und trainierten Klassifikatoren getestet. Mit Hilfe einer Menge von Testdaten wird die Fehlerrate der einzelnen Verfahren ermittelt. Schließlich wird das Klassifikationsverfahren mit dem besten Klassifikationsergebnis, also das Verfahren mit der niedrigsten Fehlerrate, zur Lösung der Klassifikationsaufgabe ausgewählt. Ein letzter Schritt ist eventuell noch möglich, nämlich die Verbesserung des ausgewählten Klassifikators. Diese Optimierung ist allerdings meist sehr aufwendig im Verhältnis zum Leistungszuwachs.

Ein anderes Konzept steht hinter der Idee der Kombination von Klasifikatoren. Anstatt einen einzelnen Klassifikator über die Klassenzugehörigkeit entscheiden zu lassen, sollen mehrere Klassifikatoren gemeinsam eine Entscheidung fällen. Man erhofft sich davon den Vorteil, daß durch die Kombination ganz verschiedener Einzelklassifikatoren eine bessere Leistung erbracht werden kann als dies der einzelne Klassifikator kann. In Abbildung 1 ist das Konzept noch einmal bildlich dargestellt.

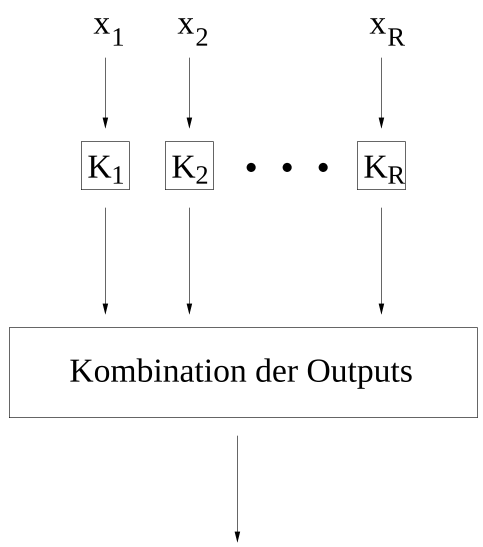

Abbildung 1: Konzept der Kombination von Klassifikatoren

Es gibt R verschiedene Klassifikatoren $K_1, ..., K_R$, jeder erhält als Input einen Merkmalsvektor x_i, mit $i = 1, ..., R$. Diese Merkmalsvektoren können alle gleich sein, d.h. jeder Einzelklassifikator bekommt denselben Input. Vorstellbar ist dies beispielsweise bei einem *k-nearest neighbor* Klassifikator. Hierbei erhält jeder Einzelklassifikator dieselbe Repräsentation eines Musters, hat also gleiche Startbedingungen, nur der Parameter k variiert von Einzelklassifikator zu Einzelklassifikator. Ein anderes Beispiel für gleiche Repräsentationen als Input ist ein *neural-network* Klassifikator, hierbei liegt die Veränderbarkeit der Einzelklassifikatoren in einer Menge von verschiedenen Gewichtungen, die aus dem Training entstanden sind. Geht man von unterschiedlichen Repräsentationen der Muster für jeden Einzelklassifikator aus, so braucht nicht das einzelne Klassifikationsverfahren variiert werden. Der Vorteil hier ist, daß durch den Ansatz verschiedener Repräsentationen ganz unterschiedliche Einzelklassifikationsverfahren gewählt werden können. Für jeden Klassifikator kann sich so ein völlig unterschiedlicher Merkmalsraum ergeben. Abbildung 1 zeigt wie das Verfahren der Kombination funktioniert. Jeder Einzelklassifikator entscheidet sich anhand seines Inputs, dem Merkmalsvektor x_i für den i-ten Klassifikator, für eine der möglichen Klassen. Die R Outputs der einzelnen Klassifikatoren werden dann zu einer finalen Entscheidung kombiniert. Alle Klassifikatoren entscheiden also gemeinsam über die Klassenzugehörigkeit eines Musters. Die Art und Weise, wie die einzelnen Outputs schließlich kombiniert werden können, ist vielfältig. Eine einfache Kombinationsregel wäre beispielsweise das Aufsummieren der einzelnen Stimmen für eine Klasse und eine finale Entscheidung für die Klasse mit den meisten Stimmen. Diese Art der Kombination ist nur eine von vielen Möglichkeiten, die im Folgenden untersucht wird.

2 Kombinationsregeln

Für die Kombination einzelner Klassifikatoren gibt es eine Reihe von Regeln die in der Praxis meist rein intuitiv angewendet werden. Die Kombinationsregeln der Praxis sind also eher Heuristiken als mathematisch bewiesen. Hier soll nun der mathematische Hintergrund beleuchtet werden. Die bekanntesten dieser Regeln sind die Summen-, Produkt-, sowie Maximum- und Minimum-Regel. Josef Kittler [2] suchte für diese Regeln eine theoretische Herleitung auf Basis der *Bayesschen Entscheidungsregel für minimale Fehlerrate*. Er zeigte damit welche Annahmen gemacht werden müssen, wenn man diese Regeln aus der Praxis anwendet. Erstaunliches zeigten die Experimente mit diesen Regeln, die trotz der einschränkenden Voraussetzungen, die für eine theoretische Herleitung oft getroffen werden müssen, hervorragende Leistung erbrachten. Bevor die theoretische Herleitung hier erläutert wird, soll zuerst die Bayessche Entscheidungsregel vorgestellt und auf das Problem der Kombination von Klassifikatoren, d.h. Entscheidung unter Berücksichtigung mehrerer Inputvektoren, projiziert werden.

2.1 Bayessche Entscheidungsregel für minimale Fehlerrate

Zum leichteren Verständnis wird hier zuerst die Bayessche Entscheidungsregel für einen Klassifikator erläutert. Man geht aus von einem Mustererkennungsproblem, bei dem ein Muster Z einer von m möglichen Klassen $(\omega_1, ..., \omega_m)$ zugewiesen werden soll. Dieses Muster Z wird durch einen Merkmalsvektor x repräsentiert. Anhand der Merkmale des Musters wird nun die Klassenzugehörigkeit zu den verschiedenen Klassen bestimmt. Für jede Klasse erhält man so eine *a-posteriori* Wahrscheinlichkeit $P(\omega_k|x)$, die Wahrscheinlichkeit für die Klasse ω_k gegeben den Merkmalsvektor x. Mittels dieser Wahrscheinlichkeit fällt die Bayessche Regel eine Entscheidung für eine Klasse.

$$\begin{aligned} Muster Z \rightarrow \omega_j,\ wenn \\ P(\omega_j|x) = \max_k P(\omega_k|x) \end{aligned} \tag{1}$$

Die Bayessche Entscheidungsregel (1) besagt, daß das Muster Z derjenigen Klasse ω_j zugewiesen wird, für die die a-posteriori Wahrscheinlichkeit dieser Interpretation maximal ist. Mit Hilfe des *Satz von Bayes* ($P(\omega_k|x) = \frac{p(x|\omega_k)P(\omega_k)}{p(x)}$) lassen sich die a-posteriori Wahrscheinlichkeiten der einzelnen Klassen berechnen. Vernachlässigt man den Nenner in dieser Formel, der unabhängig von k und somit konstant ist, sich also nur normierent auswirkt und keinen Einfluß auf die Maximierungsentscheidung hat, so läßt sich die Bayessche Entscheidungsregel schreiben als:

$$\begin{aligned} Muster Z &\rightarrow \omega_j,\ wenn \\ P(\omega_j)p(x|\omega_j) &= \max_k P(\omega_k)p(x|\omega_k) \end{aligned} \tag{2}$$

Somit weist die Entscheidungsregel (2) ein Muster Z der Klasse ω_j zu, für die das Produkt aus *a-priori* Wahrscheinlichkeit einer Klasse und *klassenbedingter* Wahrscheinlichkeit maximal ist. Die klassenbedingte Wahrscheinlichkeit $p(x|\omega_k)$ modelliert die Verteilung der Klassen im Merkmalsraum. Für eine Klassifikationsaufgabe ergibt sich nun ein Problem, da $P(\omega_k)$ und $p(x|\omega_k)$ unbekannt sind. Als Ansatz werden hier Modelle für diese beiden Wahrscheinlichkeiten gewählt, deren Parameter aus den Trainingsdaten geschätzt werden. Ein Beispiel wäre für die klassenbedingte Wahrscheinlichkeit, also das Modell der Klassen im Merkmalsraum, eine *Gaußverteilung* zu wählen und den Erwartungswert und die Varianz zu schätzen. Die a-priori Wahrscheinlichkeit einer Klasse ω_k ist dann deren relative Häufigkeit im Training.

Soviel zur Bayesschen Entscheidungsregel für einen einzelnen Klassifikator bzw. einen Merkmalsvektor als Input. Um diese Entscheidungsregel auf die Problematik der Kombination von Klassifikatoren anzuwenden, müssen die Inputs der Einzelklassifikatoren gleichzeitig betrachtet werden. Geht man wiederum von R Klassifikatoren aus, die jeweils einen unterschiedlichen Merkmalsvektor x_i, $i = 1, ..., R$ verarbeiten müssen, so verändert sich das Bayessche Theorem zu:

$$\begin{aligned} Muster Z &\rightarrow \omega_j,\ wenn \\ P(\omega_j|x_1, ..., x_R) &= \max_k P(\omega_k|x_1, ..., x_R) \end{aligned} \tag{3}$$

Es wird nun eine Entscheidung für eine Klasse aufgrund der gleichzeitigen Betrachtung aller Inputvektoren getroffen. Um die a-posteriori Wahrscheinlichkeiten zu berechnen kann man eine entsprechend angepaßte Formel des Satz von Bayes anwenden:

$$P(\omega_k|x_1, ..., x_R) = \frac{p(x_1, ..., x_R|\omega_k)P(\omega_k)}{p(x_1, ..., x_R)} \tag{4}$$

Betrachtet man den Nenner, so kann auch hier derselbe Ansatz wie bei Bayes für einen Klassifikator gemacht werden. Der Nenner wird vernachlässigt, da er keinen Einfluß auf die Maximumentscheidung hat, und man erhält aus der Form (3) die Entscheidungsregel (5), die besagt, daß man das Muster Z derjenigen Klasse ω_j zuweist, für die das Produkt aus a-priori und klassenbedingter Verbundwahrscheinlichkeit maximal ist.

$$\begin{aligned} Muster Z &\rightarrow \omega_j,\ wenn \\ P(\omega_j)p(x_1, ..., x_R|\omega_j) &= \max_k P(\omega_k)p(x_1, ..., x_R|\omega_k) \end{aligned} \tag{5}$$

Man erkennt auf den ersten Blick die Problematik an dieser Stelle. Die Merkmalsvektoren $x_1, ..., x_R$ werden hier simultan betrachtet, was eine Abschätzung äußerst schwierig macht. Um wieder auf eine einfache Entscheidungsregel, wie die Bayessche Regel für einen einzelnen Klassifikator, zu gelangen muß versucht werden die Verbundwahrscheinlichkeiten aufzubrechen.

Der Versuch die Verbundwahrscheinlichkeiten zu entkoppeln führt zu einigen stark einschränkenden Annahmen, aus denen sich allerdings die beiden Grundregeln, die Summen- und die Produktregel, herleiten lassen. Weitere Regeln gehen aus diesen beiden Regeln hervor und wie es sich zeigen wird erreichen sie zum Teil eine hervorragende Leistung in durchgeführten Experimenten.

2.2 Produkt- und Summenregel

Eine erste Annahme die gemacht wird, um die Verbundwahrscheinlichkeiten aufzubrechen, ist die statistische Unabhängigkeit der Merkmalsvektoren $x_1, ..., x_R$. Die Verwendung unterschiedlicher Repräsentationen eines Musters kann einen solchen Fall der Unabhängigkeit erfüllen. Setzt man die statistische Unabhängigkeit als gegeben voraus, so läßt sich die klassenbedingte Verbundwahrscheinlichkeit schreiben als

$$p(x_1, ..., x_R|\omega_k) = \prod_{i=1}^{R} p(x_i|\omega_k) \tag{6}$$

wobei $p(x_i|\omega_k)$ das Klassenmodell der i-ten Repräsentation ist. Wendet man den Term (6) auf die klassenbedingten Verbundwahrscheinlichkeiten in der Entscheidungsregel (5) an, so ergibt sich eine entkoppelte Variante der Regel.

$$\begin{aligned} Muster Z &\rightarrow \omega_j,\ wenn \\ P(\omega_j)\prod_{i=1}^{R} p(x_i|\omega_j) &= \max_k P(\omega_k)\prod_{i=1}^{R} p(x_i|\omega_k) \end{aligned} \tag{7}$$

Man sieht sofort, daß sich in dieser Entscheidungsregel keine (schwer zu schätzenden) Verbundwahrscheinlichkeiten befinden. Schreibt man diese Formel wieder durch Terme der a-posteriori Wahrscheinlichkeiten der Einzelklassifikatoren, so erhält man aus (7) eine Regel, die als *Produktregel* bezeichnet wird.

$$\begin{aligned} MusterZ &\rightarrow \omega_j,\ wenn \\ P^{-(R-1)}(\omega_j)\prod_{i=1}^{R}P(\omega_j|x_i) &= \max_k P^{-(R-1)}(\omega_k)\prod_{i=1}^{R}P(\omega_k|x_i) \end{aligned} \tag{8}$$

Die Entscheidungsregel (8) kombiniert die Hypothesen der Einzelklassifikatoren durch eine Produktregel. Unter der Annahme gleicher a-priori Wahrscheinlichkeiten aller Klassen, wird aus (8) die in der Praxis angewandten Kombinationsregel erkennbar. In Tabelle 1 ist ein Beispiel für diese Produktregel aufgeführt. Es werden jeweils die Outputs aller Einzelklassifikatoren für eine Klasse multipliziert und schließlich eine Entscheidung für die Klasse mit dem größten Wert gefällt. Man kann allerdings schon hier einen markanten Nachteil der Produktregel erkennen. Liefert wenigstens ein Klassifikator einen Output für eine Klasse nahe Null oder gleich Null, beeinflußt dies das gesamte Klassifikationsergebnis negativ. Das Produkt der Outputs für eine solche Klasse wird, selbst wenn die restlichen Werte größer sind, so klein, daß die Maximierungsentscheidung eine andere Klasse liefert.

Ausgehend von der Entscheidungsregel (8) läßt sich eine weitere Regel herleiten, die Summenregel. Macht man hier die Annahme, daß die a-posteriori Wahrscheinlichkeiten, die die einzelnen Klassifikatoren produzieren, nicht gravierend von den a-priori Wahrscheinlichkeiten abweichen, so kann man schreiben:

$$P(\omega_k|x_i) = P(\omega_k)(1+\delta_{ki}), \quad \delta_{ki} \ll 1 \tag{9}$$

Diese Annahme ist natürlich eine sehr strenge Einschränkung. Denn die Behauptung, die a-posteriori Wahrscheinlichkeiten würden nicht gravierend von den a-priori Wahrscheinlichkeiten abweichen, bedeutet im Grunde nichts anderes, als daß die Muster selbst bzw. deren Merkmale keinen Einfluß auf die Klassifikation haben. In diesem Fall würden die Objekte nur in Abhängigkeit der relativen Häufigkeiten klassifiziert. Nur wenn die Muster so stark verrauscht sind, daß aus ihnen keine zusätzliche Information heraus zu holen ist, wird diese Annahme gerechtfertigt. Zur Herleitung der Summenregel muß diese Annahme allerdings gemacht werden. Ersetzen wir in der Produktregel (8) die a-posteriori Wahrscheinlichkeiten durch (9), mit der Bedingung $\delta_{ki} \ll 1$, so ergibt sich

$$P^{-(R-1)}(\omega_k)\prod_{i=1}^{R}P(\omega_k|x_i) = P(\omega_k)\prod_{i=1}^{R}(1+\delta_{ki}) \tag{10}$$

Multipliziert man das Produkt auf der rechten Seite von (10) aus und vernachlässigt alle Terme höherer Ordnung, so erhält man die Summe $1+\delta_{k1}+...+\delta_{kR}$ bzw. $1+\sum_{i=1}^{R}\delta_{ki}$. Damit läßt sich die rechte Seite von (10) annähernd schreiben als

$$P(\omega_k)\prod_{i=1}^{R}(1+\delta_{ki}) = P(\omega_k) + P(\omega_k)\sum_{i=1}^{R}\delta_{ki} \tag{11}$$

Unter Verwendung von (11) und einigen weiteren Umformungsschritten, die dazu führen, daß in der Summe wieder der Term $1+\delta_{ki}$ auftaucht, der wiederum durch (9) substituiert werden kann, kann schließlich die *Summenregel* hergeleitet werden.

$$\begin{aligned} MusterZ &\rightarrow \omega_j,\ wenn \\ (1-R)P(\omega_j) + \sum_{i=1}^{R}P(\omega_j|x_i) &= \max_k \left[(1-R)P(\omega_k) + \sum_{i=1}^{R}P(\omega_k|x_i)\right] \end{aligned} \tag{12}$$

Die Entscheidungsregel (12) summiert die Hypothesen der einzelnen Klassifikatoren auf. Vernachlässigt man die a-priori Wahrscheinlichkeiten, so entsteht die Kombinationsregel, die man in der Praxis intuitiv einsetzen würde. Das Beispiel in Tabelle 1 zeigt die Funktionsweise der Regel. Die Outputs der Klassifikatoren für eine Klasse werden aufsummiert und daraus die finale Entscheidung durch die Maximierung erzielt. Der gravierende Nachteil der Produktregel, bei Outputs nahe Null ein Klassifikationsergebnis zu verzerren, tritt hier nicht auf.

	Einzelklassifikatoren				Kombinationsregeln			
	K_1	K_2	K_3	K_4	$\prod$	$\sum$	max	min
Klasse ω_1	0,7	0,7	0,6	0	0	2,0	0,7	0
Klasse ω_2	0,2	0,2	0,3	0,4	0,0048	1,1	0,4	0,2
Klasse ω_3	0,1	0,1	0,1	0,6	0,0006	0,9	0,6	0,1
			$\max\limits_k\{...\}$		0,0048	2,0	0,7	0,2
			Klasse		ω_2	ω_1	ω_1	ω_2

Tabelle 1: Beispiel für die Anwendung der Kombinationsregeln

Bevor aus den beiden gezeigten Kombinationsregeln weitere Regeln entwickelt werden, noch ein paar Worte zu den Annahmen. Diese Annahmen sind für viele Situationen unrealistisch, aber es gibt einige Anwendungen, bei denen diese zutreffen. Man kann diese Annahmen auch als akzeptables Modell für eine sehr viel komplexere Realität ansehen. Und schließlich darf nicht vergessen werden, daß diese Regeln, die aus den Voraussetzungen hergeleitet werden, in der Praxis ständig eingesetzt werden. D.h. bei der Anwendung der Regeln in der Praxis werden diese Annahmen unbewußt, mit letztlich guten Resultaten, eingesetzt. Die Untersuchung der Anfälligkeit der Kombinationsregeln auf Schätzfehler wird zeigen, warum beispielsweise die Summenregel trotz allem hervorragende Ergebnisse liefern kann.

2.3 Weitere Kombinationsregeln

Die Produktregel (8) und die Summenregel (12) bilden die Basis für die Kombination von Klassifikatoren. Aus diesen beiden Grundregel lassen sich viele allgemein gebräuchliche Kombinationsstrategien ableiten. Für die weiteren Regeln, die hergeleitet werden sollen, wird die folgende Relation verwendet.

$$\prod_{i=1}^{R} P(\omega_k|x_i) \leq \min_{i=1}^{R} P(\omega_k|x_i) \leq \frac{1}{R}\sum_{i=1}^{R} P(\omega_k|x_i) \leq \max_{i=1}^{R} P(\omega_k|x_i) \tag{13}$$

Durch die Relation (13) lassen sich die obere und untere Grenze der Produkt- und der Summenregel abschätzen. Unter der Berücksichtigung, daß die a-posteriori Wahrscheinlichkeiten im Intervall [0,1] liegen, gilt die obige Abschätzung. Denn damit ist das Produkt der a-posteriori Wahrscheinlichkeiten immer kleiner oder gleich dem Minimum dieser Werte, das Minimum ist wiederum immer kleiner oder gleich dem Mittelwert der a-posteriori Werte und der Mittelwert ist immer kleiner oder gleich dem Maximum der Reihe. Diese Abschätzung, sowie die beiden Grundregeln bilden den Ausgangspunkt für die nächsten Kombinationsregeln.

2.3.1 Die Max-Regel

Ausgehend von der Summenregel (12) und der Abschätzung der Summe durch das Maximum in (13) kann die *Maximumregel* aufgestellt werden.

$$\begin{aligned} Muster Z &\rightarrow \omega_j,\ wenn \\ (1-R)P(\omega_j) + R\max_{i=1}^{R} P(\omega_j|x_i) &= \max_k \left[(1-R)P(\omega_k) + R\max_{i=1}^{R} P(\omega_k|x_i)\right] \end{aligned} \tag{14}$$

Woraus sich unter der Annahme gleicher a-priori Wahrscheinlichkeiten aller Klassen die einfache Regel

$$\begin{aligned} Muster Z &\rightarrow \omega_j,\ wenn \\ \max_{i=1}^{R} P(\omega_j|x_i) &= \max_k \max_{i=1}^{R} P(\omega_k|x_i) \end{aligned} \tag{15}$$

ergibt. Ein Beispiel für die Maximumregel ist in Tabelle 1 gegeben.

2.3.2 Die Min-Regel

Die *Minimumregel* wird abgeleitet von der Produktregel (8) unter Berücksichtigung der Abschätzung des Produkts durch das Minimum in (13). Es ergibt sich

$$\begin{aligned} Muster Z &\rightarrow \omega_j,\ wenn \\ P^{-(R-1)}(\omega_j) \min_{i=1}^{R} P(\omega_j|x_i) &= \max_k P^{-(R-1)}(\omega_k) \min_{i=1}^{R} P(\omega_k|x_i) \end{aligned} \tag{16}$$

Auch hier ergibt sich unter Annahme gleicher a-priori Wahrscheinlichkeiten eine vereinfachte Formel:

$$\begin{aligned} Muster Z &\rightarrow \omega_j,\ wenn \\ \min_{i=1}^{R} P(\omega_j|x_i) &= \max_k \min_{i=1}^{R} P(\omega_k|x_i) \end{aligned} \tag{17}$$

Ein Beispiel zu dieser Regel ist in Tabelle 1 gegeben. Zur Verdeutlichung dieser Regel sei gesagt, daß es sich um eine vorsichtige Schätzung handelt, bei der für jede Klasse der minimale Output aller Klassifikatoren genommen wird und über diese Minima das Maximum gebildet wird. Diese Regel wird auch als Minimax-Regel oder *Pessimisten*-Regel bezeichnet.

Die Minimum- und Maximumregeln sind von den übrigen Kombinationsregel insofern unterschiedlich, als daß sie nicht wirklich alle Outputs der Einzelklassifikatoren kombinieren, sondern nur einen Output-Wert herausfiltern für die Entscheidung einer Klasse. Genau genommen handelt es sich hier also nicht um eine Kombination, sondern um eine Selektion.

2.3.3 Die Median-Regel

Betrachtet man die Summenregel (12) unter der Annahme gleicher a-priori Wahrscheinlichkeiten, so kann diese einfach in eine Formel für die *durchschnittlichen* a-posteriori Wahrscheinlichkeiten aller Klassen überführt werden.

$$\begin{aligned} Muster Z &\rightarrow \omega_j,\ wenn \\ \frac{1}{R}\sum_{i=1}^{R} P(\omega_j|x_i) &= \max_k \frac{1}{R}\sum_{i=1}^{R} P(\omega_k|x_i) \end{aligned} \tag{18}$$

Diese Entscheidungsregel (18) weist also ein Muster Z derjenigen Klasse ω_j zu, für die der Mittelwert bzw. das arithmetische Mittel über alle Outputs der Klassifikatoren einer Klasse maximal ist. Nun ist jedoch allseits bekannt, das die Mittelwertbildung über eine Reihe von Werten nicht stabil ist. D.h. es reicht aus wenn nur ein einziger Ausreißer vorhanden ist, um den Mittelwert stark zu verfälschen. Ein robusteres Verfahren ist dagegen die Bildung des Medianwertes. Und somit ergibt sich aus der 'Mittelwertregel' die *Medianregel*.

$$\begin{aligned} Muster Z &\rightarrow \omega_j,\ wenn \\ \operatorname*{med}_{i=1}^{R} P(\omega_j|x_i) &= \max_k \operatorname*{med}_{i=1}^{R} P(\omega_k|x_i) \end{aligned} \tag{19}$$

2.3.4 Die Majority-Vote Regel

Die *Majority-Vote Regel* ist eine Regel, bei der die Klassenzugehörigkeit nicht durch Wahrscheinlichkeiten gegeben ist, sondern die Outputs eines Klassifikators werden umgewandelt in eine Stimme für genau eine Klasse.

$$\Delta_{ki} = \begin{cases} 1 & \text{wenn } P(\omega_k|x_i) = \max_{j=1}^{m} P(\omega_j|x_i) \\ 0 & \text{sonst} \end{cases} \tag{20}$$

Δ_{ki} ist eine binäre Funktion, die jeden Klassifikator zwingt eine Stimme für einen Klassifikator abzugeben. Die Stimme erhält die Klasse für die der Output des Einzelklassifikators maximal ist. Ausgehend von der Summenregel und der Umwandlung durch (20) ergibt sich

$$\begin{aligned} Muster Z &\rightarrow \omega_j,\ wenn \\ \sum_{i=1}^{R} \Delta_{ji} &= \max_k \sum_{i=1}^{R} \Delta_{ki} \end{aligned} \tag{21}$$

Die rechte Seite von (21) summiert die Stimmen aller Klassifikatoren für eine Klasse ω_k auf. Die finale Entscheidung über die Klassenzugehörigkeit eines Objekts wird schließlich durch die Bildung des Maximums über alle Klassen gefällt, d.h. die Klasse mit den meisten Stimmen 'gewinnt'.

2.4 Experimente mit den Kombinationsregeln

Um nun die Leistungsfähigkeit der hergeleiteten Kombinationsregeln zu überprüfen müssen diese nicht nur untereinander verglichen werden sondern auch der Leistung verschiedener Einzelklassifikatoren gegenüber gestellt werden. Dazu wurden zwei Experimente durchgeführt, eine Identitätsprüfung bei Personen und eine Handschriftenerkennung.

Im ersten Experiment, der Identitätsprüfung von Personen, standen drei verschiedene Datenquellen zur Verfügung. Zum einen wurden von diversen Personen Aufnahmen der Gesichtsfront gemacht. Dazu kamen Aufnahmen des Gesichtsprofils und jeweils eine Stimmprobe. Für jede Datenquelle gab es einen Einzelklassifikator, der anhand der entsprechenden Daten klassifizierte. Dem gegenüber standen in diesem Experiment die Kombinationsregeln, die jeweils die drei Einzelklassifikatoren koppelten und eine gemeinsame Entscheidung herbei führten. In Tabelle 2 sind die Ergebnisse der Klassifikationsaufgabe aufgeführt. Neben dem Klassifikationsverfahren steht die erreichte Fehlerrate, also die relative Zahl der Fehlklassifikationen.

Verfahren	Fehlerrate (%)
Gesichtsfront	12,2
Gesichtsprofil	8,5
Sprachprobe	1,4
Summenregel	0,7
Produktregel	1,4
Maxregel	12,2
Medianregel	1,2
Minregel	4,5

Tabelle 2: Experiment: Identitätsprüfung

Das beste Ergebnis der Einzelklassifikatoren erzielte der 'Sprach'-Klassifikator mit einer Fehlerrate von 1,4%. Dieser Wert soll als Referenzwert zum Vergleich mit den Kombinationsregeln dienen. Die Produktregel zeigt die gleiche Leistungfähigkeit in diesem Vergleich, was heißt, daß in diesem Fall der Aufwand der Kombination keinen Vorteil bringt. Kombinationsregeln, die wirklichen eine Leistungssteigerung gegenüber dem besten Einzelklassifikator erzielen sind Medianregel, aber vor allem die Summenregel. Mit einer Fehlerrate von nur 0,7% ist diese nur halb so groß wie der Referenzwert.

Diese hervorragende Leistung der Summenregel zeigt sich auch im zweiten Experiment, bei dem auf einer Datenbasis handgeschriebener Postleitzahlen die Ziffern erkannt und klassifiziert werden sollten. Die Daten stammen von der CEDAR-CDROM, die am *Center of Excellence for Document Analysis and Recognition* der Universität von New York, Buffalo, erstellt wurde. Die Datenbasis besteht aus Ziffern, die von *dead-letter*-Umschlägen des U.S. Postal Service gescannt wurden und umfaßte in diesem Experiment eine Trainingsmenge von 18.468 Mustern und eine Testmenge von 2.213 Mustern. Als Einzelklassifikatoren wurden vier ganz verschiedene Verfahren eingesetzt. Zum einen wurde ein 'struktureller' Klassifikator, ein Klassifikator der das Muster in Primitive zerlegt und dann die Struktur in einem Entscheidungsbaum klassifiziert. Auf die anderen Einzelklassifikatoren, ein Gauß-Klassifikator, ein Neural-Network Klassifikator und ein auf einem Hidden-Markov-Modell basierender Klassifikator, soll an dieser Stelle nicht näher eingegangen werden. Die einzelnen Hypothesen dieser vier Klassifikatoren werden danach durch die Kombinationsregeln zu einer gemeinsamen Entscheidung kombiniert.

Die Ergebnisse des Vergleichs zwischen den vier Einzelklassifikatoren und den verschiedenen Kombinationen zeig die Tabelle 3. Der beste Einzelklassifikator ist das Hidden-Markov-Modell, dessen Fehlerrate von 5,2% als Referenzwert heran gezogen wird. Im Vergleich dazu die Kombinationsregeln, bei denen die Produktregel am schlechtesten abgeschnitten hat mit einer Fehlerrate von 15,3%. Diese schlechte Leistung ist, wie schon erwähnt, mit der Anfälligkeit dieser Regel bei Outputs der Einzelklassifikatoren nahe oder gleich Null zu erklären. Ein Nachteil der hier deutlich zu erkennen ist. Sehr gut ist auch hier wieder die Summenregel, die mit einer Fehlerrate von 1,9% deutlich besser als der Referenzwert der Einzelklassifikatoren ist. Beste Leistung erzielte die Medianregel mit 1,8% fehlklassifizierten Objekten.

Verfahren	Fehlerrate (%)
Strukturell	9,2
Gauss	6,1
NeuralNet	6,8
HMM	5,2
MajorityVote	2,1
Summenregel	1,9
Produktregel	15,3
Maxregel	6,1
Medianregel	1,8
Minregel	14,0

Tabelle 3: Experiment: Handschriftenerkennung

Die Experimente zeigen, daß es durchaus sinnvoll ist Klassifikatoren zu kombinieren, denn es kann dadurch eine Verbesserung der Leistung gegenüber dem Einsatz einzelner Klassifikationsverfahren erzielt werden. Überraschend ist die hervorragende Leistung der Summenregel, die Regel, die unter den strengsten Einschränkungen entwickelt wurde. Um dieses Phenomen zu erklären wird im folgenden Abschnitt die Anfälligkeit der Summen- und Produktregel auf Schätzfehler untersucht.

2.5 Einfluß von Schätzfehlern

Bei der Entwicklung der Kombinationsregeln wurde angenommen, daß die a-posteriori Wahrscheinlichkeiten $P(\omega_k|x_i)$ als Outputs der Einzelklassifikatoren korrekt berechnet wurden. Tatsächlich ist es aber so, daß diese Wahrscheinlichkeit nur eine Schätzung ist, die einen gewissen Fehler beinhaltet. Bezeichnet man die Schätzung der a-posteriori Wahrscheinlichkeit mit $\hat{P}(\omega_k|x_i)$ und den Schätzfehler, also der Fehler, um den die tatsächliche Wahrscheinlichkeit von der Schätzung abweicht, mit e_{ki}, so läßt sich die Schätzung schreiben als

$$\hat{P}(\omega_k|x_i) = P(\omega_k|x_i) + e_{ki} \tag{22}$$

Mit diesen geschätzten Wahrscheinlichkeiten arbeiten die Kombinationsregeln. Welche Auswirkungen die Schätzfehler auf die Produkt- und die Summenregel haben, soll nun untersucht werden.

Geht man von der Produktregel (8) aus und setzt die Formel (22) ein, so ergibt sich

$$\begin{aligned} MusterZ \quad &\rightarrow \quad \omega_j, \ wenn \\ P^{-(R-1)}(\omega_j) \prod_{i=1}^{R} [P(\omega_j|x_i) + e_{ji}] \quad &= \quad \max_k P^{-(R-1)}(\omega_k) \prod_{i=1}^{R} [P(\omega_k|x_i) + e_{ki}] \end{aligned} \tag{23}$$

die Produktregel unter Berücksichtigung des Schätzfehlers. Um nun eine Art Fehlerfaktor zu erhalten, zieht man die a-posteriori Wahrscheinlichkeiten aus der Klammer. Unter der Voraussetzung, daß der Fehler $e_{ki} \ll P(\omega_k|x_i)$ und $P(\omega_k|x_i) \neq 0$ ist, kann man den Term nun als Produkt schreiben:

$$\prod_{i=1}^{R} [P(\omega_k|x_i) + e_{ki}] = \left[\prod_{i=1}^{R} P(\omega_k|x_i)\right] \prod_{i=1}^{R} \left[1 + \frac{e_{ki}}{P(\omega_k|x_i)}\right] \tag{24}$$

Die rechte Seite läßt sich, wie bei der Herleitung der Summenregel, durch eine Summe abschätzen.

$$\prod_{i=1}^{R} [P(\omega_k|x_i) + e_{ki}] = \left[\prod_{i=1}^{R} P(\omega_k|x_i)\right] \left[1 + \sum_{i=1}^{R} \frac{e_{ki}}{P(\omega_k|x_i)}\right] \tag{25}$$

Schließlich erhält man somit die *Produkt-Regel* beeinflußt durch einen Fehlerfaktor:

$$\begin{aligned} &MusterZ \rightarrow \omega_j, wenn \\ &\left[P^{-(R-1)}(\omega_j) \prod_{i=1}^{R} P(\omega_j|x_i)\right] \left[1 + \sum_{i=1}^{R} \frac{e_{ji}}{P(\omega_j|x_i)}\right] = \\ &\max_k \left[P^{-(R-1)}(\omega_k) \prod_{i=1}^{R} P(\omega_k|x_i)\right] \left[1 + \sum_{i=1}^{R} \frac{e_{ki}}{P(\omega_k|x_i)}\right] \end{aligned} \tag{26}$$

Vergleicht man (8) mit (26), erkennt man, daß jede Hypothese einer Klasse ω_k beeinflußt wird durch den Faktor

$$1 + \sum_{i=1}^{R} \frac{e_{ki}}{P(\omega_k|x_i)}. \tag{27}$$

Bevor auf den Fehlerfaktor genauer eingegangen wird, zuerst noch die Untersuchung der Summenregel. Für die Summenregel ergibt sich ein ähnliches Bild. In (28) ist die Summenregel beeinflußt durch den Fehler e_{ki} dargestellt.

$$\begin{aligned} Muster Z &\rightarrow \omega_j, wenn \\ (1-R)P(\omega_j) + \sum_{i=1}^{R} [P(\omega_j|x_i) + e_{ji}] &= \max_k \left\{ (1-R)P(\omega_k) + \sum_{i=1}^{R} [P(\omega_k|x_i) + e_{ki}] \right\} \end{aligned} \tag{28}$$

Versucht man die a-posteriori Wahrscheinlichkeiten aus der Summe heraus zu ziehen, sodaß auch hier ein Fehlerfaktor ekennbar wird, ergibt sich einfach

$$\begin{aligned} & Muster Z \rightarrow \omega_j, wenn \\ & (1-R)P(\omega_j) + \left[\sum_{i=1}^{R} P(\omega_j|x_i)\right] \left[1 + \frac{\sum_{i=1}^{R} e_{ji}}{\sum_{i=1}^{R} P(\omega_j|x_i)}\right] = \\ & \max_k \left\{ (1-R)P(\omega_k) + \left[\sum_{i=1}^{R} P(\omega_k|x_i)\right] \left[1 + \frac{\sum_{i=1}^{R} e_{ki}}{\sum_{i=1}^{R} P(\omega_k|x_i)}\right] \right\} \end{aligned} \tag{29}$$

Der Vergleich von (12) mit (29) zeigt, daß jede Hypothese durch den Faktor

$$1 + \frac{\sum_{i=1}^{R} e_{ki}}{\sum_{i=1}^{R} P(\omega_k|x_i)} \tag{30}$$

beeinflußt wird. Die beiden Fehlerfaktoren von Produkt- und Summenregel sollen nun miteinander verglichen und analysiert werden. Im Folgenden sind beide Faktoren noch einmal aufgeführt. Der Fehlerfaktor der Produktregel steht links, der der Summenregel auf der rechten Seite.

$$\left[1 + \sum_{i=1}^{R} \frac{e_{ki}}{P(\omega_k|x_i)}\right] \qquad\qquad \left[1 + \frac{\sum_{i=1}^{R} e_{ki}}{\sum_{i=1}^{R} P(\omega_k|x_i)}\right]$$

Der Fehlerfaktor der Produktregel zeigt, daß jeder Fehler e_{ki} durch den Faktor $1/P(\omega_k|x_i)$ verstärkt wird, da die a-posteriori Wahrscheinlichkeiten zwischen 0 und 1 liegen. In der Summenregel werden die Fehler gedämpft. Die a-posteriori Wahrscheinlichkeiten im Fehlerfaktor werden hier zuerst aufaddiert und ergeben meist einen Wert größer 1. Dadurch wird die Summe der Fehler hier gedämpft.

Es zeigt sich also, daß die Summenregel weniger anfällig auf Schätzfehler reagiert, im Gegensatz dazu wird bei der Produktregel der Fehler verstärkt. Dies ist zumindest teilweise eine Erklärung für die guten Leistungen, die die Summenregel in den Experimenten lieferte. Die Summenregel ist also nicht nur ein sehr einfaches und oft rein intuitiv verwendetes Kombinationsverfahren für Einzelklassifikatoren, sondern es ist auch noch sehr robust.

3 Bagging und Boosting

Die im vorigen Kapitel hergeleiteten Regeln kombinieren bestehende Klassifikatoren und leiten aus deren einzelnen Outputs eine gemeinsame Entscheidung herbei. Einen anderen Ansatz zeigen die beiden folgenden Verfahren, Bagging und Boosting, die die Einzelklassifikatoren während der Trainingsphase selbst konstruieren. Das gemeinsame Grundprinzip beider Algorithmen ist eine Manipulation der Trainingsdaten. Ziel dieses Verfahrens ist die Konstruktion möglichst unterschiedlicher Einzelklassifikatoren, deren Einzelhypothesen dann durch eine *Voting*-Methode kombiniert werden. Sowohl die Manipulation der Trainingsmenge als auch die Kombination der Einzelstimmen wird bei Bagging und Boosting auf verschiedene Arten erreicht. Beiden gemeinsam ist allerdings die Voraussetzung eines instabilen Lernsystems, d.h. geringe Änderungen der Trainingsmenge führen zu unterschiedlichen Klassifikatoren.

3.1 Bagging

Die Manipulation der Trainingsmenge erfolgt bei Bagging durch ziehen von Stichproben mit Zurücklegen. Das Ergebnis dieser Stichproben sind verschiedene Mengen von Trainingsdaten, da einige Daten doppelt und andere überhaupt nicht enthalten sind. Jede dieser unterschiedlichen Mengen wird dann zum Training der Einzelklassifikatoren benutzt.

Im Detail sieht das Bagging-Verfahren nach Breiman [3] so aus, daß man von einer Menge von N Instanzen oder Mustern, die jeweils einer von m möglichen Klassen angehören, und einem Lernsystem, das den Einzelklassifikator mittels der Trainingsdaten konstruiert, ausgeht. Um nun mehrere unterschiedliche Klassifikatoren zu erzeugen muß die Trainingsmenge so manipuliert werden, daß jeder Einzelklassifikator durch eine andere Ausprägung der Merkmale trainiert wird. Um eine Anzahl von T unterschiedlichen Einzelklassifikatoren zu konstruieren sind gerade T Durchläufe notwendig. Bei jedem Durchlauf $t = 1, ..., T$ wird aus der ursprünglichen Menge von Trainingsdaten eine Stichprobe mit Zurücklegen vom Umfang N gezogen, d.h. die entstehende Trainingsmenge beinhaltet genauso viele Muster wie das Original, nur daß einige Instanzen mehrfach vorkommen können und andere überhaupt nicht enthalten sind. Mit diesen Daten wird dann der Einzelklassifikator K_t des t-ten Durchlaufs trainiert. Dadurch, daß alle Trainingsmengen unterschiedlich sind und Instanzen mehrfach auftreten, entstehen verschiedene Ausprägungen einzelner Merkmale und somit unterschiedliche Einzelklassifikatoren. Die Outputs der einzelnen Klassifikatoren werden dann durch ein Voting-Verfahren zu einer finalen Entscheidung kombiniert. D.h. jeder Einzelklassifikator gibt eine Stimme $K_t(x)$ ab und der kombinierende Klassifikator K^* liefert die Klasse mit den meisten Stimmen.

Mit diesem Verfahren kann man auf Basis einer kleinen Menge von Trainingsdaten und nur einem Lernsystem, also nur einer Methode zur Klassifikation, ein System verwirklichen, das verschiedene Einzelklassifikatoren konstruiert und die Entscheidung über eine Klassenzugehörigkeit in Abhängigkeit mehrerer Hypothesen fällt. In Kapitel 3.3 zeigen die Versuche, daß dieses Verfahren im Mittel eine geringere Fehlerrate besitzt, also besser klassifiziert als der einzelne Klassifikator.

Im Gegensatz zu Bagging, das die Trainingsmenge mit Hilfe des Zufalls manipuliert, versucht das Boosting-Verfahren durch verschiedene Gewichtungen der Daten unterschiedliche Trainingsmengen für die einzelnen Klassifikatoren zu generieren.

3.2 Boosting

Das Grundprinzip des Boosting-Verfahrens ist mit dem des Bagging identisch. Prinzipiell wird versucht auf Basis einer Menge von Trainingsdaten und einem Lernsystem mehrere Einzelklassifikatoren zu konstruieren und eine gemeinsame Entscheidung für die Klassenzugehörigkeit eines Objekts zu finden. Die Manipulation der Trainingsmenge wird hier durch eine veränderbare Gewichtung jeder Instanz erreicht. Während des Trainings eines Einzelklassifikators wird das Gewicht einer jeden fehlklassifizierten Instanz erhöht, sodaß im nächsten Durchlauf diese Daten mehr Einfluß im Training des folgenden Einzelklassifikators haben. Soll nun ein neues Muster x klassifiziert werden, so bilden die unterschiedlichen Klassifikatoren durch ein Voting-Verfahren eine gemeinsame finale Entscheidung. Die Stimmen der Einzelklassifikatoren besitzen allerdings eine Gewichtung in Abhängigkeit ihrer Leistungsfähigkeit, also ihrer Fehlerrate auf den Trainingsdaten.

Das grundlegende Boosting-Verfahren wird nun am Beispiel des von Freund und Schapire entwickelten AdaBoost [4] detaillierter betrachtet. Der Algorithmus des AdaBoost sieht wie folgt aus:

Algorithmus:

Input: Trainingsdaten $x_1, ..., x_N$
Initialisiere $D_1(x_i) = \frac{1}{N}$ für alle i
Do for $t = 1, 2, ..., T$:

1. Trainieren des Klassifikators K_t unter Berücksichtigung der Gewichte $D_t(x_i)$
2. Berechne Fehler des Klassifikators K_t: $\varepsilon_t = \sum D_t(x_i)$ für fehlklassifizierten Instanzen x_i
   Wenn $\varepsilon_t \geq 1/2$, dann setze $T = t - 1$ und stoppe Iteration! (Abbruchkriterium)
3. Setze $\beta_t = \dfrac{\varepsilon_t}{1 - \varepsilon_t}$
4. Berechne neue Gewichtung D_{t+1}:

$$D_{t+1}(x_i) = \frac{D_t(x_i)}{c_t} \times \begin{cases} \beta_t & \text{wenn } x_i \text{ korrekt klassifiziert} \\ 1 & \text{sonst} \end{cases}$$

   wobei c_t Renormierungskonstante

Output der finalen Entscheidung:

$$K^*(x) = \underset{k}{\operatorname{argmax}} \sum_{t:K_t(x)=k} \log\left(\frac{1}{\beta_t}\right)$$

Der Algorithmus erhält als Input eine Menge von Trainingsdaten $x_1, ..., x_N$, die jeweils einer Klasse k, von m möglichen Klassen, zugehören. Zu Beginn wird jedem Muster dieselbe Gewichtung zugewiesen, d.h. bei m möglichen Klassen also ein Gewicht von $D_1(x_i) = 1/m$ für alle i, $i = 1, ..., N$. Mit dieser Anfangsgewichtung wird nun für T Einzelklassifikatoren iteriert. In jedem Durchlauf t, $t = 1, ..., T$, wird ein Klassifikator K_t durch das Lernsystem konstruiert, wobei das Gewicht jeder Instanz in den Lernprozeß einfließt. Der Trainingsvorgang liefert neben einem trainierten Einzelklassifikator gleichzeitig auch ein Maß für dessen Leistungsfähigkeit. Der Fehler ε_t des Einzelklassifikators wird berechnet als Summe der Gewichte der fehlklassifizierten Instanzen x_i. Daraus läßt sich das Verhältnis von fehlklassifizierten zu korrekt klassifizierten Objekten ablesen. Dieses Gütemaß β_t wird in zweifacher Weise als Gewichtungsfaktor verwendet, zum einen bei der Neugewichtung der Instanzen für den nächsten Klassifikator und zum anderen als Gewichtung der Stimmen der Einzelklassifikatoren bei der finalen Entscheidung K^*. Bei der Neuberechnung der Gewichte für den nächsten Durchgang $t+1$ wird jede korrekt klassifizierte Instanz mit dem Faktor β_t multipliziert. Da dieser Faktor kleiner als 0.5 ist (Abbruchkriterium) werden die Gewichte der korrekt klassifizierten Instanzen verringert. Durch die Renormierung mit der Konstanten c_t, die gleich der Summe der neuen Gewichte ist, wird somit die Gewichtung der fehlklassifizierten Instanzen vergrößert und diese gehen damit stärker in den Trainingsprozeß des nächsten Einzelklassifikators ein. Nach der Renormierung sind die falsch klassifizierten Instanzen stärker und die korrekt klassifizierten schwächer gewichtet. Diese neu gewichteten Instanzen der Trainingsmenge werden nun als Daten für die Konstruktion des nächsten Klassifikators verwendet. Nachdem alle Einzelklassifikatoren trainiert wurden, wird ein Objekt durch die Kombination aller Outputs klassifiziert. Eine finale Entscheidung wird schließlich durch ein Voting-Verfahren gefällt, bei dem die Stimme jedes Klassifikators mit einem Gewicht von $\log(1/\beta_t)$ einfließt. Je kleiner β_t ist, also je besser der Einzelklassifikator arbeitet, desto größer ist sein Stimmanteil am Gesamtergebnis.

Durch die Gewichtung der Instanzen der Trainingsmenge verändert sich die Form der Daten und es ergeben sich verschiedene Klassifikatoren. Die Kombination der Einzelklassifikatoren führt im Allgemeinen zu einer besseren Leistung als ein einzelner dieser Klassifikatoren, der allein klassifiziert.

3.3 Experimente mit Bagging, Boosting und C4.5

Die Leistungsfähigkeit der beiden hier vorgestellten Verfahren, Bagging und Boosting, wird nun an einem Experiment gezeigt. Diese Versuchsreihe wurde von J.R. Quinlan [5] durchgeführt, der das von ihm entwickelte Klassifikationsverfahren C4.5 [6], ein Klassifikator auf Basis eines Entscheidungsbaums, als Einzelklassifikator eingesetzt und es mit einem BaggedC4.5 und einem BoostedC4.5 verglichen hat. D.h. er wendete das Bagging- und Boosting-Verfahren auf seinen Algorithmen an und verglich die Leistungen der Kombinationsverfahren mit seinem Einzelklassifikator.

In Tabelle 4 sind die Ergebnisse dieses Experimentes aufgeführt. Quinlan benutzte für den Vergleich der Verfahren 27 ganz unterschiedliche Datenbasen aus dem *UCI Machine Learning Repository* (www.ics.uci.edu/~mlearn). Für Bagging und Boosting wurde der Parameter T, der die Anzahl der Durchläufe angibt, auf zehn gesetzt, d.h. beide Verfahren generierten maximal zehn Einzelklassifikatoren, die eine gemeinsame Entscheidung fällen sollten.

	C4.5	BaggedC4.5 vs. C4.5		BoostedC4.5 vs. C4.5		Boosting vs. Bagging
	err(%)	err(%)	ratio	err(%)	ratio	ratio
anneal	7.67	6.25	.814	4.73	.617	.758
audiology	22.12	19.29	.872	15.71	.710	.814
auto	17.66	19.66	1.113	15.22	.862	.774
breast-w	5.28	4.23	.802	4.09	.775	.966
chess	8.55	8.33	.975	4.59	.537	.551
colic	14.92	15.19	1.018	18.83	1.262	1.240
credit-a	14.70	14.13	.962	15.64	1.064	1.107
credit-g	28.44	25.81	.908	29.14	1.025	1.129
diabetes	25.39	23.63	.931	28.18	1.110	1.192
glass	32.48	27.01	.832	23.55	.725	.872
heart-c	22.94	21.52	.938	21.39	.932	.994
heart-h	21.53	20.31	.943	21.05	.978	1.037
hepatitis	20.39	18.52	.908	17.68	.867	.955
hypo	.48	.45	.928	.36	.746	.804
iris	4.80	5.13	1.069	6.53	1.361	1.273
labor	19.12	14.39	.752	13.86	.725	.963
letter	11.99	7.51	.626	4.66	.389	.621
lymphography	21.69	20.41	.941	17.43	.804	.854
phoneme	19.44	18.73	.964	16.36	.842	.873
segment	3.21	2.74	.853	1.87	.583	.684
sick	1.34	1.22	.907	1.05	.781	.861
sonar	25.62	23.80	.929	19.62	.766	.824
soybean	7.73	7.58	.981	7.16	.926	.944
splice	5.91	5.58	.943	5.43	.919	.944
vehicle	27.09	25.54	.943	22.72	.839	.889
vote	5.06	4.37	.864	5.29	1.046	1.211
waveform	27.33	19.77	.723	18.53	.678	.938
average	15.66	14.11	.905	13.36	.847	.930

Tabelle 4: Vergleich von Bagging, Boosting und C4.5

Tabelle 4 zeigt für die drei Verfahren, C4.5, Bagging und Boosting, die erzielte Fehlerrat (in %) auf den 27 Datenbasen, d.h. den Anteil der fehlklassifizierten Instanzen. Weiterhin werden die Leistungsverhältnisse von Bagging zu C4.5 und Boosting zu C4.5 angegeben. Ein Wert kleiner eins weist auf ein besseres Ergebnis des jeweiligen Verfahrens zu seinem Vergleichspartner hin. Die letzte Spalte zeigt den Vergleich zwischen Boosting und Bagging, wobei hier ein Wert kleiner eins auf die bessere Leistung von Boosting verweist.

Es zeigte sich, daß Bagging und Boosting im Mittel über alle 27 Datenbasen deutlich bessere Leistung erbrachten als der Einzelklassifikator C4.5. Bagging verringerte den Klassifikationsfehler von C4.5 um annähernd 10% und war auf 24 der 27 Datenbasen besser als der Einzelklassifikator. Boosting reduzierte den Fehler um etwa 15%, aber war nur auf 21 Datenbasen besser und schlug völlig fehl auf sechs der 27 Mengen.

Beim Vergleich von Bagging und Boosting lieferte Boosting eine geringere Fehlerrate auf 20 der 27 Datenbasen. Aber es zeiget sich auch, daß Boosting unberechenbarer ist und in den Ergebnissen stark schwanken kann. So verschlechterte Boosting die Fehlerrate z.B. auf der *iris* Datenbasis um 36% und um 26% auf *colic*. Dagegen war das schlechteste Ergebnis von Bagging eine Erhöhung des Fehlers um 11% (*auto*).

Boosting erzielte also im Mittel bessere Ergebnisse, ist aber im Gegensatz zu Bagging risikoreicher in der Beständigkeit. Ein Nachteil des Bagging-Verfahrens ist allerdings, daß die Ergebnisse nicht reproduzierbar sind, da die Trainingsmengen per Zufall generiert wurden. Trotz der Ausfälle beider Verfahren sind sie im Durchschnitt besser als der Einzelklassifikator C4.5.

3.3.1 Verfeinerung von AdaBoost

Es stellt sich nach diesen Ergebnissen die Frage, warum Boosting manchmal versagt?

Freund und Schapire schieben dieses auf das sogenannte *Overfitting* der Klassifikatoren. D.h. durch eine große Zahl an Durchläufen (Freund und Schapire führten Experimente mit einer sehr viel größeren Zahl von Durchläufen durch) werden die Einzelklassifikatoren so komplex, daß sie sich zu sehr an die Trainingsdaten anpassen und somit neue Muster fehlklassifizieren. Quinlan untersuchte diesen Punkt und versuchte den Algoritmus so zu verändern, daß die Iteration nicht nur stoppte wenn ein Einzelklassifikator eine Fehlerrate von Null erreichte, sondern brach die Schleife schon ab, wenn der kombinierte Klassifikator K^* optimale Ergebnisse lieferte. Diese Verbesserung des Verfahrens brachte aber auf den vorhandenen 27 Datenbasen keinen Vorteil. Tatsächlich wurden die Ergebnisse schlechter und es ergab sich eine um 13% höhere durchschnittliche Fehlerrate.

Daraufhin untersuchte Quinlan die Gewichtung der Stimme der einzelnen Klassifikatoren und kam zu der Erkenntnis, daß die starre Struktur der Gewichtung nicht der Leistung dieser Klassifikatoren gerecht wurde. Für seinen Entscheidungsbaum-Klassifikator entwickelte er eine Verfeinerung des Boosting-Algorithmus, bei der die Gewichtung nun abhängig von der zu klassifizierenden Instanz x ist.

Quinlan's Idee ist es die Stimme des t-ten Klassifikators K_t nicht mit dem fixen Wert von $\log(1/\beta_t)$ zu gewichten, sondern aus dem Entscheidungsbaum ein *Konfidenzmaß* abzulesen. Die Konfidenz $H^t(x)$, $H^t(x)\epsilon[0,1]$, einer Hypothese $K_t(x)$ ist ein Gütemaß und macht eine Aussage darüber, wie sicher eine Hypothese ist. Als Beispiel sei hier in Abbildung 2 ein trainierter Entscheidungsbaum aufgeführt.

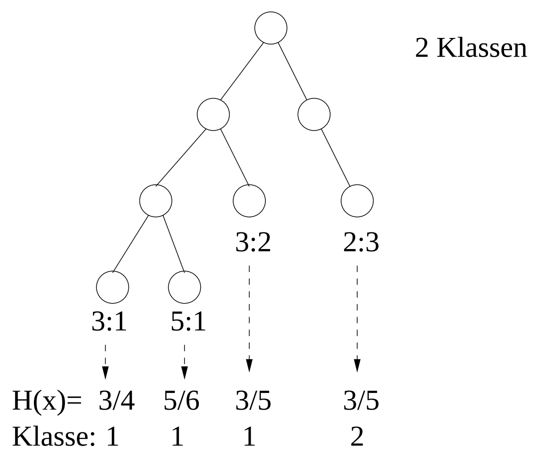

Abbildung 2: Ein Konfidenzmaß in einem trainierten Entscheidungsbaum

Der trainierte Entscheidungsbaum klassifiziert Objekte in eine von zwei möglichen Klassen. Im Training sind beispielsweise im äußersten linken Blatt drei Objekte der Klasse 1 und ein Objekt der Klasse 2 gelandet. Soll der Klassifikator nun über die Klassenzugehörigkeit eines neuen Musters entscheiden und gelangt dieses Muster in das besagte Blatt, so ist die Zuordnung in die Klasse 1 mit einem Konfidenzmaß von 3/4 gegeben. D.h. mit einer Wahrscheinlichkeit von 3/4 gehört das Objekt in die Klasse 1.

Dieses Konfidenzmaß spiegelt die Genauigkeit einer Hypothese eines Einzelklassifikators genauer wieder als die starre Stimmgewichtung durch den Term $\log(1/\beta_t)$, der den Einfluß einer zu klassifizierenden Instanz x nicht berücksichtigt. Quinlan verwendete allerdings statt der Anzahl der in den Blättern gelandeten Trainingsdaten, die Summe der fehlklassifizierten bzw. der korrekt klassifizierten Instanzen eines Blattes. Das Verhältnis dieser beiden Summen zueinander diente somit als Gewichtungsfaktor für die Stimmen der einzelnen Klassifikatoren. Mit dieser Verfeierung erhielt er eine deutliche Verbesserung auf den 27 Datenbasen. Auf 25 der 27 Datenbasen erreichte er so eine geringere Fehlerrate und nur auf einer einzigen ein schlechteres Ergebnis (auf einer Datenbasis ergabt sich keine Veränderung). Die mittlere Fehlerrate sank um annähernd drei Prozent.

Zusammenfassend läßt sich also sagen, daß durch eine kleine Veränderung des Boosting-Verfahrens, die es erlaubt die Gewichtung der Stimme eines Einzelklassifikators je nach Instanz zu variieren, eine Verbesserung erzielt werden kann. Der Wechsel von der Gewichtung der Stimmen durch den fixen Term $\log(1/\beta_t)$ hin zu einer angepaßteren Gewichtung an die jeweilige Instanz mittels eines Konfidenzmaßes führt zu einer geringeren Fehlerrate und macht Boosting zu einem weniger riskanten und stabileren Verfahren.

4 Fazit

Die theoretische Herleitung der verschiedenen Kombinationsregeln für Einzelklassifikatoren anhand der Bayesschen Entscheidungsregel zeigt die Annahmen und Voraussetzungen, die gemacht werden müssen, um solche Regeln, die in der Praxis ohne großes Hinterfragen eingesetzt werden, mathematisch aufzustellen. Die aus den beiden Basisregeln, Produkt- und Summenregel, abgeleitete Max- bzw. Min-Regel, sowie Median und Majority-Vote Regel liefern ähnliche Ergebnisse, wie die Regeln aus denen sie hervorgehen. Ein überraschendes Ergebnis lieferten die durchgeführten Experimente, in denen die Summenregel, die Regel, die unter den einschränkensten Annahmen entwickelt wurde, mit die beste Leistung erbrachte und anderen Kombinationsregel bzw. einzelne Klassifikationsverfahren deutlich übertrumpfte. Erklären läßt sich dies durch die Sensitivitätsanalyse, die die Anfälligkeit der Regeln bezüglich Schätzfehlern untersuchte. Dabei stellte sich heraus, daß die Summenregel robuster gegenüber Schätzfehlern ist, da der Fehlerfaktor gedämpft wird. Dies mag wenigstens zum Teil die sehr gute Performance plausibel machen.

Im Gegensatz zu den Kombinationsregeln, die lediglich bestehende Klassifikatoren kombinieren, konstruieren die beiden Verfahren Bagging und Boosting die Einzelklassifikatoren selbst aus einer einzigen Menge von Trainingsdaten. Um auf dieser Basis verschiedene Klassifikatoren zu entwickeln wird die Trainingsmenge manipuliert. Voraussetzung dafür ist ein instabiles Lernsystem, das bei kleinen Änderungen an den Trainingsdaten unterschiedliche Klassifikatoren erzeugen kann. Bei Bagging wird diese Manipulation per zufälliger Stichprobe mit Zurücklegen durchgeführt, Boosting dagegen gewichtet die Daten unterschiedlich, sodaß sie verschieden stark in den Trainingsprozeß eines Einzelklassifikators eingehen. Die finale Entscheidung wird bei beiden Verfahren durch eine Voting-Methode bestimmt. Die Experimente haben gezeigt, daß Bagging und Boosting einem einzelnen Klassifikator erheblich überlegen sein können. Allerdings zeigte gerade Boosting, daß sich das Konzept des Verfahrens auf einigen Datenbasen auch sehr negativ auswirken kann.

Als Fazit kann man schließlich sagen, daß die Kooperative Klassifikation meist die Fehlerrate senken kann und ein leistungsfähigeres Klassifikationsverfahren hervorbringt. Aber wie das Wort 'meist' schon besagt, ist dies nicht immer der Fall und es kann somit keine generelle Aussage gemacht werden. Die Anwendbarkeit von kombinierenden Klassifikatoren sollte deshalb immer genau überprüft werden.

Literatur

[1] Niemann H.: *Klassifikation von Mustern*, Springer Verlag, Berlin 1983.

[2] Kittler J., Hatef M., Duin R., Matas J.: "On Combining Classifiers", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 20, No. 3, March 1998.

[3] Breiman L.: "Bagging Predictors", Technical Report 421, Dept. of Statistics, Univ. of California at Berkley, 1994.

[4] Freund, Y. and Schapire R.E.: "Experiments with a new boosting algorithm", *Proc. 13th International Conference of Machine Learning*, 1996.

[5] Quinlan J.R.: "Bagging, Boosting and C4.5", *Proceedings, Fourteenth National Conference of Artificial Intelligence*, 1996.

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