Online Algorithms in Machine Learning

Proseminar von
Prof. Dr. Rolf Klein, Dr. Elmar Langetepe, Dipl. Inform. Thomas Kamphans
im Wintersemester 00/01

von Alexandra Scherbart
04. Dezember 2000

Inhalt:

I. Einfuehrung

Die Bereiche Online-Algorithmen und Machine Learning beschäftigen sich beide mit dem Problem, Entscheidung in der Gegenwart zu treffen, nur basierend auf dem Wissen über die Vergangenheit.
Wir beginnen mit einem Problem, welches schon ausführlich in der Literatur des Maschinellen Lernens behandelt worden ist: "predicting from expert advice".
Das Problem der Prognose durch Expertenbefragung wurde zuerst von Littlestone und Warmuth, DeSantis, Markowsky und Wegman, und Vovk studiert. Der WMA (samt Theorem 1 und 2) geht auf Littlestone und Warmuth zurück. Verschiedene Variationen und Erweiterungen dieses Algorithmus lassen sich in der Literatur finden.

Ziel:

Entwicklung von Programmen, die ihre Ausführung durch Erfahrung oder durch Lernen verbessern können.

Wie können Computer programmiert werden, so daß sie ”lernen” ?

Definition: Lernendes System
Ein System wird als lernend bezeichnet, wenn es in der Lage ist, unbekannte Eigenschaften eines Prozesses oder seiner Umgebung durch schrittweises und/oder wiederholtes Handeln und Beobachten zu erfassen.

Die dadurch gewonnene Erfahrung wird benutzt, um Vorhersagen, Klassifikationen und Entscheidungen durchzuführen, damit ein vorgegebenes optimales Systemverhalten oder Leistungssteigerung des Systems erreicht werden kann.
(Fu, 1964; Saridis, 1977)

Vorhersage nach Expertenrat

Wir fangen mit einem einfachen, intuitiven Problem an. Ein lernender Algorithmus hat die Aufgabe das Wetter für jeden Tag vorherzusagen, ob es an diesem Tag regnen wird oder nicht. Um diese Vorhersage zu treffen, erhält der Algorithmus als Eingabe den Rat von n Wetterexperten. Jeden Tag gibt jeder dieser Wetterexperten seine eigene Vorhersage ab (ja oder nein) und der Algorithmus nutzt diese Information, nur die ja/nein - Bits der Experten für Regen oder keinen Regen, um seine eigene Vorhersage zu machen. Nachdem der Algorithmus seine Entscheidung getroffen hat, wird ihm das tatsächlich eingetroffenen Ereignis mitgeteilt.

Wir setzen voraus, daß Vorhersagen vom Typ {0,1} sind.

Die Folge der Ereignisse, in der der Algorithmus
(1) eine Eingabe der n Experten erhält vom Typ $\{0,1\}^n$,
(2) seine eigene Vorhersage trifft, und dann
(3) die korrekte Antwort erhält,
nennen wir einen Versuch.

Da keine Annahmen über die Qualität oder Unabhängigkeit der Experten gemacht werden, können wir nicht auf ein besonders gutes Ergebnis in unseren Vorhersagen hoffen. Wir wollen aber trotzdem alle Expertenvorhersagen berücksichtigen.
Der lernende Algorithmus soll die Durchführung seiner Aufgabe mit jeder Versuchsdurchführung verbessern. Sein Lernziel ist es, sich zu "merken", welcher Experte gute Vorhersagen liefert und welcher nicht, d.h. die Experten in „bessere“ und „schlechtere“ einzuteilen.
Wir wollen uns dann am besten Experten orientieren und garantieren, daß wir zu jeder Zeit nicht wesentlich schlechter als der bis dahin beste Experte sind.
Die Leistung des Algorithmus soll daran gemessen werden, daß die Differenz zwischen der Anzahl an Fehlern des Algorithmus und der Fehleranzahl des besten Experten minimal ist.

II . Weighted Majority Algorithm
-simple version-

Idee: Wir wollen bei der Vorhersage „bessere“ und „schlechtere“ Experten berücksichtigen. Der Algorithmus soll die Experten für jeden Fehler „bestrafen“, falls ihre Prognose von der korrekten Antwort abweicht.

Dieser Algorithmus behält eine Liste von Gewichten $w_1,...,w_n$, für jeden Experten einzeln und der Algorithmus sagt das Ereignis mit der höchsten Gesamtgewichtung der Expertenmeinungen vorher.

1. Alle n Experten werden mit der Gewichtung $w_1,...,w_n = 1$ initialisiert.
   Die Gewichte aller Experten werden zunächst mit 1 initialisiert, da bei Versuchsbeginn kein Wissen über die Güte der Experten bekannt ist und alle gleich gut sind.

2. Gegeben: $\{x_1,...,x_n\}$ (= Menge an Vorhersagen der Experten)
   Im zweiten Schritt des Algorithmus erfolgt die Ausgabe der Vorhersage des Ereignisses mit der höchsten Gesamtgewichtung.

$$\begin{cases} 1 & \text{wenn } \sum\limits_{i:x_i=1} w_i \geq \sum\limits_{i:x_i=0} w_i \\ 0 & \text{sonst} \end{cases}$$

3. Ist die Prognose erfolgt, erhält der Algorithmus Auskunft über die korrekte Antwort. Das Gewicht jedes falsch geschätzten Experten wird mit ½ multipliziert, um diesen bei der nächsten Durchführung nur noch zur Hälfte zu berücksichtigen. Gehe zu Schritt 2.

$$\begin{cases} w_i \leftarrow w_i/2 & \text{wenn } x_i \neq l \\ w_i & \text{wenn } x_i = l \end{cases}$$

Beispiel für WMA:

Prognosen	Gewichte				$\sum$ Y	$\sum$ N	Prognose Algorithmus	korrekte Antwort
Y Y Y N	1	1	1	1	3	1	Y	Y
Y N N Y	1	1	1	0,5	1.5	2	N	Y
Y N N N	1	0,5	0,5	0,5	1	1,5	N	N
N Y N Y	0,5	0,5	0,5	0,5	1	1	Y	N
	0,5	0,25	0,5	0,25				

Theorem 1
Die Anzahl der Fehler M des oben beschriebenen Algorithmus beträgt nie mehr als **2,41 (m +log n)**, wobei m die Anzahl der Fehler ist, die der beste Experte bis dahin gemacht hat.

Beweis:
W bezeichne die Gesamtgewichtung aller Experten, also initialisiere W = n. Wenn der Algorithmus einen Fehler macht, bedeutet das, daß mindestens die Hälfte der Gesamtgewichtung der Experten inkorrekt vorhergesagt hat, und deshalb wird in Schritt 3 des WMA das Gesamtgewicht um mind. den Faktor ¼ reduziert. Bei M Fehlern des Algorithmus ergibt sich für die Gesamtgewichtung:

$$W \leq n \cdot (3/4)^M$$

Auf der anderen Seite, wenn der beste Experte m Fehler gemacht hat, beträgt seine Gewichtung $w = (1/2)^m$.

$$W \geq w \Rightarrow \quad 1/2^{\,m} \leq n \cdot (3/4)^M$$

$$2^m \geq (1/n)(4/3)^M$$

$$m \geq -\lg(n) + M \lg(4/3)$$

$$M \leq \frac{1}{\log(4/3)} (m + \log n)$$

$$M \leq 2{,}41\,(m + \log n) \quad \blacksquare$$

Verbesserung des Weighted-Majority-Algorithm(simple version):
Bei der ersten Version des WMA trat der worst case ein, wenn die Gewichte zu 50/50 verteilt waren, bzw. wenn wenig mehr als die Hälfte des Gesamtgewichts falsch vorhersagte, veranlaßte dies den Algorithmus einen Fehler zu machen.

Ein besseres Ergebnis als oben kann durch Modifizierung des Algorithmus auf zwei Arten erfolgen:
1. Randomisierung: Anstatt der Vorhersage des Outputs mit der höchsten Gesamtgewichtung, betrachtet man nun die Gewichtungen als Wahrscheinlichkeiten, und sagt jedes Ereignis hervor als Wahrscheinlichkeit proportional zu seinem Gewicht.
2. Die zweite Veränderung betrifft die Ersetzung des Faktors ½ durch einen anderen ß, dessen Wert später festgelegt wird.

Der Vorteil der zufallsbedingten Näherung ist, daß der schlimmste Fall abgeschwächt wird. Zuvor war im schlimmsten Fall leicht mehr als die Hälfte des Gesamtgewichtes inkorrekt vorhergesagt, was zur Folge hatte, daß der Algorithmus einen Fehler machte und jetzt nur noch das Gesamtgewicht um ¼ reduzierte. Nun gibt es eine 50/50 - Chance, daß der Algorithmus in diesem Fall richtig vorhersagt.

III . The Weighted Majority Algorithm
-Randomized Version-

1. Initialisierung der w_i (i=1,...,n) mit 1.

2. Gegeben: $\{x_1,...,x_n\}$
   Output x_i mit Wahrscheinlichkeit w_i/W, wobei $W=\Sigma_i w_i$

3. $\begin{cases} w_i \leftarrow w_i \cdot ß & \text{wenn } x_i \neq l \\ w_i & \text{wenn } x_i = l \end{cases}$

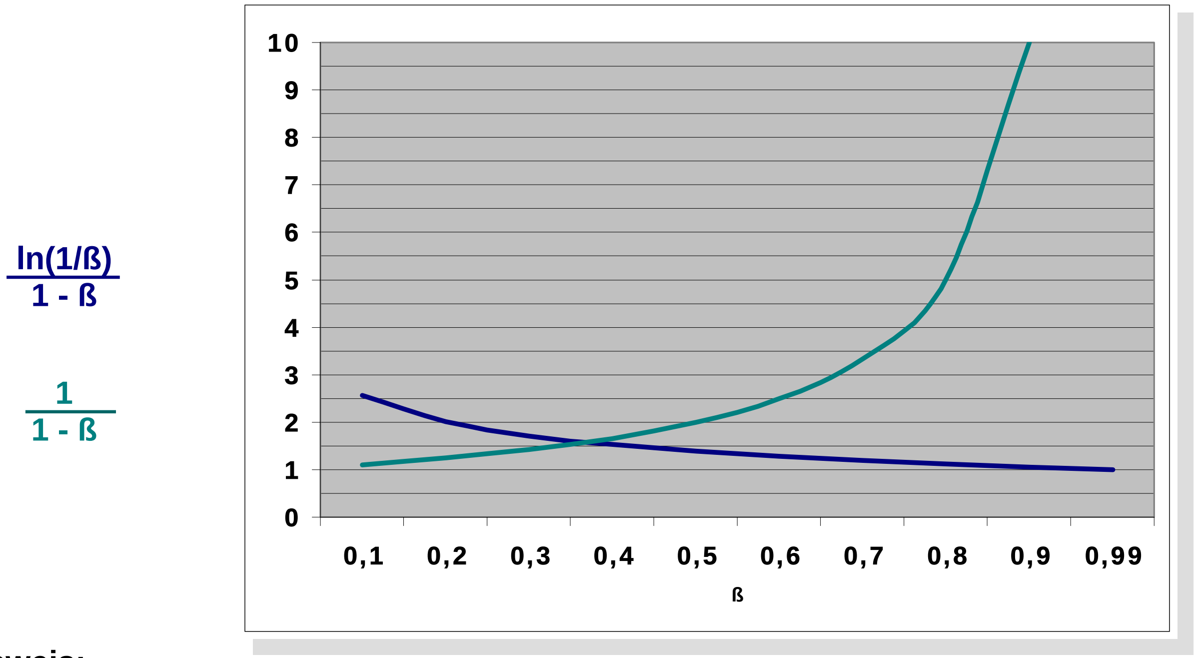
Beispiel für RWMA (ß=0,5):

Prognosen	Gewichte				W	Prob(Y)	Prob(N)	Prognose Algorithmus	korrekte Antwort
Y Y Y N	1	1	1	1	4	0,75	0,25	Y	Y
Y N N Y	1	1	1	ß	3,5	0,43	0,57	N	Y
Y N N N	1	ß	ß	ß	2,5	0,4	0,6	Y	N
N Y N Y	ß	ß	ß	ß	1,75	0,5	0,5	Y	N
	ß	$ß^2$	ß	$ß^2$					

Theorem 2
Für die Anzahl an Fehlern M bei dem RWM-Algorithmus gilt, wobei m wieder die Anzahl an Fehlern ist, die der beste Experte bis dahin gemacht hat:

$$M \leq \frac{m \cdot \ln(1/ß) + \ln n}{1 - ß}$$

Für ß=½ erhält man folgendes Ergebnis: $M \leq$ **1,39**m + **2**ln n,
für ß=¾ sogar $M \leq$ **1,15**m + **4**ln n
Man kann erreichen, daß der Faktor so nah an 1 ist, wie man möchte, aber ein größeres $ß \in [0,1]$ zieht auch eine größere additive Konstante mit sich. Häufig nutzt man deshalb die typische „guess and double“-Näherung, indem man ß mit einem Wert initialisiert und im Laufe der Versuchsdurchführungen dynamisch anpaßt.



Beweis:
F_i sei jetzt der Teil des Gesamtgewichts der falschen Antworten beim i-ten Versuch.
Nach t Versuchen beträgt die erwartete Anzahl von Fehlern M:

$$M = \sum_{i=1}^{t} F_i$$

Beim i-ten Versuch verändert sich das Gesamtgewicht wie folgt:

$$W \leftarrow W(1-(1-ß)\cdot F_i)$$

Nach t Versuchen beträgt das Gesamtgewicht:

$$W = n \prod_{i=1}^{t} (1-(1-ß) F_i)$$

m sei wieder die Anzahl an Fehlern des besten Experten.
W muß mindestens so groß sein wie das Gewicht des besten Experten:

$$n \prod_{i=1}^{t} (1-(1-ß) F_i) \geq ß^m$$

$$\ln n + \sum \ln(1-(1-ß) F_i) \geq m \cdot \ln ß$$

$$-\ln n - \sum \ln(1-(1-ß) F_i) \leq m \cdot \ln(1/ß) \qquad -\ln(1-x) < x$$

$$-\ln n + (1-ß) \sum F_i \leq m \cdot \ln(1/ß)$$

$$-\ln n + (1-ß) M \leq m \cdot \ln(1/ß)$$

$$\boxed{M \leq \frac{m \cdot \ln(1/ß) + \ln n}{1-ß}}$$

■

Ein großer Vorteil ist, daß die Vorhersagen als „Strategien“ betrachtet werden können. So können z.B. die Experten Programme oder zu bewertende Funktionen darstellen.

Beispiel: Jeder Experte stellt einen Page-Replacement-Algorithmus dar, das System muß entscheiden, welcher Algorithmus benutzt werden soll. Regelmäßig wird der Verlust für die verschiedenen Algorithmen, die das System hätte benutzen können, berechnet und anhand dieser Information entschieden, welcher Algorithmus als nächstes benutzt werden soll.

Eine Erweiterung der Randomized-Version besteht darin, daß z.B. jeder Experte eine reelle Zahl [0,1] als Vorhersage liefert und der Algorithmus genauso eine reelle Zahl als Wahrscheinlichkeit für ein Ereignis liefert.

Quellen:

A. Fiat and G.J. Woeginger
Online Algorithms: The State of the Art, LNCS 1442
Kap. 14 S. 306-325