On-line Lernen: Die Themen

(1) Experten unterschiedlicher Qualität geben Empfehlungen ab.

Können wir die Qualität des besten Experten annähernd erreichen, auch wenn sich die Qualität eines Experten mit der Zeit ändert?

(2) Eine sehr große Zahl von Eigenschaften ist gegeben, aber nur eine sehr kleine Zahl ist von Relevanz:

Wie können wir die relevanten Eigenschaften schnell bestimmen?

(3) Wir erhalten Gegenbeispiele für jede Hypothese:

Die allgemeine Frage: Wann können wir das Zielkonzept nach möglichst wenigen Gegenbeispielen exakt rekonstruieren?

Der Perzeptron-Algorithmus: Wieviel Gegenbeispiele werden benötigt, wenn wir versuchen negative und positive Beispiele linear voneinander zu trennen?

Wir beginnen mit der Auswahl von Experten.

Die Auswahl von Experten

- n Experten geben in mehreren Runden Ja/Nein Empfehlungen (wie etwa Kauf oder Verkauf einer Aktie) ab.
- Nach jeder Runde wird mitgeteilt, welche Empfehlung richtig ist.
- Können wir einen Algorithmus entwickeln, dessen Anzahl an Fehlentscheidungen nur unwesentlich größer ist als die des besten Experten?

Warum nicht einfach in jeder Runde den bisher besten Experten wählen?

(1) Konstruiere eine Menge von n Experten, so dass sich diese Strategie immer irrt, obwohl der beste Experte nach t Runden höchstens t/n Fehler macht.

(2) Katastrophale Strategie: Keine einzige richtige Entscheidung, obwohl die Experten nur mit Wahrscheinlichkeiten $1/n$ irren.

Der **weighted Majority** Ansatz

(1) Setze $w_i = 1$ für alle Experten i.
/* Zu Anfang besitzen allen Experten dasselbe Gewicht. */

(2) Wenn Experte i die Empfehlung $x_i \in \{\text{Ja}, \text{Nein}\}$ abgibt, dann treffe die Entscheidung „Ja“, falls

$$\sum_{i, x_i = \text{Ja}} w_i \geq \sum_{i, x_i = \text{Nein}} w_i$$

und ansonsten treffe die Entscheidung „Nein“.
/* Wir folgen der Mehrheit. */

(3) Nach Erhalt der richtigen Entscheidung: Wenn Experte i eine falsche Empfehlung abgegeben hat, dann setze $w_i = w_i/2$.
Ansonsten lasse das Gewicht w_i unverändert.
/* „Heftige“ Bestrafung falscher Entscheidungen */

Analyse

(1) Sei W_t das Gesamtgewicht aller Experten vor Beginn von Runde t. Dann ist anfänglich $W_1 = n$.

(2) Wenn die Entscheidung in Runde t falsch ist, dann haben sich Experten mit einem Gesamtgewicht von mindestens $W_t/2$ geirrt:

$$W_{t+1} \leq \frac{1}{2} \cdot \frac{W_t}{2} + \frac{W_t}{2} = \frac{3}{4} W_t.$$

(3) Wenn wir f falschen Entscheidungen bis zum Zeitpunkt t treffen, ist $W_t \leq n \cdot (\frac{3}{4})^f$.

(4) Wenn der beste Experte i bis zum Zeitpunkt t genau f_{opt} Fehler gemacht hat, dann ist $w_i = 2^{-f_{\text{opt}}}$ und als Konsequenz

$$(\frac{1}{2})^{f_{\text{opt}}} \leq W_t \leq n \cdot (\frac{3}{4})^f, \quad \text{bzw } (\frac{4}{3})^f \leq 2^{f_{\text{opt}}} \cdot n.$$

Das Resultat

Da $(\frac{4}{3})^f \leq 2^{f_{\text{opt}}} \cdot n$:

Wenn der Weighted Majority Algorithmus bei n Experten bisher f falsche und der beste Experte f_{opt} falsche Entscheidungen getroffen haben, dann gilt

$$f \leq \frac{1}{\log_2(4/3)} \cdot (f_{\text{opt}} + \log_2 n).$$

(1) Nach logarithmischer „Aufwärmzeit“ erreichen wir die Prognosekraft des besten Experten bis auf den Faktor $\frac{1}{\log_2(4/3)} \approx 2,41$.

(2) Können wir bei „entsprechend verlängerter Aufwärmzeit“ die Prognosekraft des besten Experten fast exakt erreichen?

Eine randomisierte Version von **Weighted Majority**

Ein allgemeineres Modell

Die Empfehlung eines Experten i zum Zeitpunkt t bewerten wir mit einer „Note“ c_i^t auf einer Skala von 0 (sehr gut) bis 1 (sehr schlecht).

Unser Algorithmus:

(1) Setze $w_i = 1$ für alle Experten i.

(2) Wähle einen Experten zufällig, wobei Experte i die Wahrscheinlichkeit

$$p_i = \frac{w_i}{\sum_{k=1}^{n} w_k}$$

erhält und übernimm seine Entscheidung.

(3) Berechne neue Gewichte: Setze $w_i = w_i(1 - \varepsilon \cdot c_i^t)$.
Kommentar: Die von der Rundenzahl t unabhängige Konstante ε wird aus dem Intervall $]0, 1/2[$ gewählt. Beachte $\varepsilon \cdot c_i^t \in [0, 1/2[$.

Analyse I

(1) w_i^t sei das Gewicht von Experte i zu Beginn von Runde t. Die Gewichtssumme aller Experten vor Runde t ist $W_t = \sum_{i=1}^n w_i^t$.

(2) $K_t = \sum_{i=1}^n \frac{w_i^t}{W_t} \cdot c_i^t$ ist die erwartete Benotung unserer Entscheidung zum Zeitpunkt t.

(3) Wie berechnet sich W_{t+1} aus W_t?

$$\begin{aligned} W_{t+1} &= \sum_{i=1}^n w_i^t \cdot (1 - \varepsilon \cdot c_i^t) = \sum_{i=1}^n w_i^t - \varepsilon \cdot \sum_{i=1}^n w_i^t \cdot c_i^t \\ &= W_t - \varepsilon \cdot K_t \cdot W_t = W_t \cdot (1 - \varepsilon \cdot K_t) \end{aligned}$$

Wir expandieren und erhalten $W_{T+1} = W_1 \cdot \Pi_{t \leq T}(1 - \varepsilon \cdot K_t)$.

Analyse II

Es ist $W_{T+1} = W_1 \cdot \Pi_{t \leq T}(1 - \varepsilon \cdot K_t)$.

(4) Beachte $\ln(1-x) \leq -x$ und

$$\ln W_{T+1} \quad = \quad \ln W_1 + \sum_{t \leq T} \ln(1 - \varepsilon \cdot K_t) \leq \ln n - \sum_{t \leq T} \varepsilon K_t$$

Eine schlechte erwartete Benotung unserer Entscheidungen drückt das Gesamtgewicht.

(5) Ein Experte mit guter Benotung hingegen stärkt das Gesamtgewicht: Für Experte i gilt

$$W_{T+1} \geq w_i^{T+1} = \Pi_{t \leq T}(1 - \varepsilon \cdot c_i^t).$$

Wir möchten wieder logarithmieren, um unsere erwartete Benotung mit der Benotung eines besten Experten vergleichen zu können: Beachte $\ln(1-x) \geq -x - x^2$ für $x \in [0, 1/2]$.

Analyse III

Es ist $W_{T+1} \geq w_i^{T+1} = \Pi_{t \leq T}(1 - \varepsilon \cdot c_i^t)$ und
$\ln(1-x) \geq -x - x^2$ für $x \in [0, 1/2]$.

(6) Nach Logarithmierung folgt wegen $0 \leq c_i^t \leq 1$:

$$\begin{aligned} \ln W_{T+1} &\geq \sum_{t \leq T} \ln(1 - \varepsilon \cdot c_i^t) \geq -\sum_{t \leq T} (\varepsilon \cdot c_i^t + (\varepsilon \cdot c_i^t)^2) \\ &\geq -\sum_{t \leq T} (\varepsilon \cdot c_i^t + \varepsilon^2 \cdot c_i^t) = -(\varepsilon + \varepsilon^2) \cdot \sum_{t \leq T} c_i^t. \end{aligned}$$

(7) Wir vergleichen die untere mit der oberen Gewichtsschranke:

$$-(\varepsilon + \varepsilon^2) \cdot \sum_{t \leq T} c_i^t \leq \ln W_{T+1} \leq \ln n - \sum_{t \leq T} \varepsilon K_t.$$

Das Ergebnis

- Es ist $-(\varepsilon + \varepsilon^2) \cdot \sum_{t \leq T} c_i^t \leq \ln n - \sum_{t \leq T} \varepsilon K_t$.
- Nach Division durch ε folgt:

Wenn K_{opt} die Gesamtbenotung des besten Experten ist und K_t die erwartete Benotung unserer Entscheidung in Runde t ist, dann folgt

$$\sum_{t \leq T} K_t \leq (1 + \varepsilon) \cdot K_{\text{opt}} + \frac{\ln n}{\varepsilon}.$$

- Wir erreichen die Benotung des besten Experten bis auf den Faktor $1 + \varepsilon$, wenn wir die „Aufwärmzeit“ $\frac{\ln n}{\varepsilon}$ erlauben.

Constant Rebalanced Portfolio

- Ein Vermögen V wird über m Anlagemöglichkeiten verteilt: Anlage i erhält das Vermögen $p_i \cdot V$.
- Zu jedem Anlagezeitpunkt: Anlage i erhält den Anteil $p_i \cdot V'$, wenn V' das gegenwärtige Vermögen ist.

CRP: Zwei Aktien mit $p_1 = p_2 = 0.5$

Wir arbeiten mit zwei Aktien, wobei sich die erste Aktie nicht bewegt, während sich die zweite Aktie stets erst halbiert und dann verdoppelt.

- Das Vermögen nach einem Anlagezeitpunkt sinkt auf $V \cdot (1/2 + (1/2) \cdot (1/2)) = 3 \cdot V/4$
- und steigt nach dem zweiten Anlagezeitpounkt auf $V \cdot (3/8 + 2 \cdot 3/8) = 9 \cdot V/8$.
- Vor dem Anlagetermin $2n + 1$ ist das Vermögen damit exponentiell auf $(\frac{9}{8})^n \cdot V$ angewachsen!

Der Universal Portfolio Algorithmus

- Welche Verteilung p sollte gewählt werden?
- Risiko-Minimierung: Der Universal Portfolio Algorithmus versucht die erwartete Vermögensentwickung des CRP-Ansatzes zu erreichen.
  - Wir arbeiten mit einer großen Anzahl N verschiedener charakteristischer Verteilungen.
  - Zu jedem Anlagezeitpunkt wird jedes Portfolio nach seiner Verteilung rebalanciert.
  - Es dürfen keine Gelder zwischen den verschiedenen Portfolios transferiert werden.

Der Universal Portfolio Algorithmus und Weighted Majority

- Anfänglich erhält jedes ausgewählte Portfolio das Gewicht $\frac{V}{N}$.
- Der Markt akualisiert die Gewichte.
- Die Gewichte geben den Erfolg wieder.

Wie gut ist der Universal Portfolio Algorithmus?

Wir arbeiten mit m Anlagemöglichkeiten über einem Zeitraum von n Anlageperioden.
opt_n sei das optimale Vermögen und e_n das erwartete Vermögen nach n Anlageperioden. Dann gilt:

$$e_n \geq \frac{\mathrm{opt}_n}{(n+1)^{m-1}}.$$

- Der durchschnittliche relative Verlustfaktor pro Anlageperiode (gegenüber der optimalen Strategie) ist höchstens

$$\left((n+1)^{m-1}\right)^{1/n} \approx \left(n^{m-1}\right)^{1/n} = n^{(m-1)/n} = 2^{\frac{(m-1)\cdot \log_2 n}{n}} \approx 1.$$

- Wächst die optimale CRP-Strategie um den Faktor a_n^n für $a_n > 1$, dann wächst das erwartete Vermögen um den Faktor b_n^n mit $b_n > 1$ und $\lim_{n\to\infty} a_n = \lim_{n\to\infty} b_n$.

Die Auswahl von Spezialisten

Wir arbeiten mit Spezialisten:

- Jeder Spezialist gibt eine Ja/Nein Empfehlung ab oder enthält sich.
- Unser Ziel: So gut zu sein, wie eine beste Menge E von Spezialisten:
  - Die Menge E erhält für jeden sich irrenden Spezialisten einen Strafpunkt.
  - Enthalten sich alle Spezialisten in E erhält E insgesamt einen Strafpunkt.

Unser Ziel ist sehr ambitioniert, da wir jetzt mit einer besten Menge von Spezialisten mithalten möchten:

Beste Spezialisten aus verschiedensten Fachgebieten können sehr viel stärker als Experten sein.

Der Winnow Algorithmus

(1) Setze $w_1 = \cdots = w_n = 1$.

(2) Wenn eine Entscheidung zu treffen ist, dann gibt jeder Spezialist i bekannt, ob er sich enthält $(x_i = 0)$ oder nicht $(x_i = 1)$.

(2a) Winnow enthält sich, falls

$$\sum_{i=1}^{n} w_i \cdot x_i < n$$

gilt. Die Gewichte aller Spezialisten, die sich *nicht* enthalten haben, werden verdoppelt.

(2b) Sonst ist $\sum_{i=1}^{n} w_i \cdot x_i \geq n$ und Winnow trifft eine Mehrheitsentscheidung: Wenn Spezialist i die Empfehlung $y_i \in \{\text{Ja}, \text{Nein}\}$ abgibt, dann entscheidet Winnow auf „Ja“, wenn

$$\sum_{i, x_i = 1, y_i = \text{Ja}} w_i \cdot x_i \geq \sum_{i, x_i = 1, y_i = \text{Nein}} w_i \cdot x_i$$

und ansonsten auf „Nein“. Die Gewichte aller Spezialisten, die die falsche Entscheidung unterstützt haben, werden halbiert.

Das Ergebnis

Sei E eine beliebige Teilmenge von insgesamt n Spezialisten. Dann macht Winnow nach T Runden höchstens

$$5 \cdot \text{Strafe}_T(E) + 5 \cdot |E| \cdot \lceil \log_2 n \rceil + 4$$

Fehler, wobei ein Fehler entweder eine Enthaltung oder eine falsche Entscheidung ist.

(1) Unsere Strafpunktzahl ist durch das 5-fache der Strafpunktzahl von E beschränkt: Aber die Aufwärmzeit $5 \cdot |E| \cdot \lceil \log_2 n \rceil + 4$ wächst mit der Größe von E.

(2) To winnow: Den Spreu vom Weizen trennen.

Der Winnow Algorithmus isoliert relativ schnell die erfolgreichen Spezialisten.

Analyse: Die Enthaltungen von Winnow

- Für $i \in E$ ist f_i die Anzahl falscher Entscheidungen nach T Runden.
- Alle Spezialisten in E enthalten sich in genau e Runden.
- Wie groß ist e^*, die Anzahl der Enthaltungen von Winnow?

(1) Winnow enthält sich nicht, wenn sich mindestens ein „Schwergewicht“ (mit $w_i \geq n$) nicht enthält.

(2) Jeder Spezialist $i \in E$, der sich mindestens $f_i + \log_2 n$ Mal **nicht** enthält, während sich Winnow jedesmal enthält, wird zu einem Schwergewicht.

> Sein Gewicht, unter Einrechnung der Halbierungen nach Fehlentscheidungen, ist auf n angestiegen.

(3) In $e^* - e$ Enthaltungsschritten von Winnow wird sich aber mindestens ein Spezialist aus E nicht enthalten.

Analyse: Die Fehlentscheidungen von Winnow

$$\begin{aligned} e^* &\leq e + \sum_{i \in E} f_i + |E| \cdot \lceil \log_2 n \rceil \\ &= \text{Strafe}_T(E) + |E| \cdot \lceil \log_2 n \rceil. \end{aligned}$$

(4) Warum? Wenn e^* größer als die rechte Seite ist, dann blockiert ein Schwergewicht in E eine Enthaltung von Winnow.

(5) Wie groß ist die Anzahl der Fehlentscheidungen von Winnow?

- Sei W_t die Summe aller Gewichte vor Runde t.
- Es ist $W_1 = n$ und $W_t > 0$.
- Wenn Winnow sich in Runde t enthält, dann ist $\sum_{i=1}^{n} w_i \cdot x_i < n$. Nach Verdopplung für Nicht-Enthaltungen ist $W_{t+1} \leq W_t + n$.
- Bei einer Fehlentscheidung von Winnow ist $\sum_{i=1}^{n} w_i x_i \geq n$. Das Gesamtgewicht der sich irrenden Spezialisten ist mindestens $n/2$. Nach Halbierung folgt $W_{t+1} \leq W_t - \frac{n}{4}$.

Gesamtanalyse

- Es ist $e^* \leq \text{Strafe}_T(E) + |E| \cdot \lceil \log_2 n \rceil$.
- Enthält sich Winnow in Runde t, dann ist $W_{t+1} \leq W_t + n$.
- Bei einer Fehlentscheidung von Winnow in Runde t ist $W_{t+1} \leq W_t - \frac{n}{4}$.

(1) Sei f^* die Anzahl der Fehlentscheidungen von Winnow. Dann ist

$$f^* \leq 4e^* + 4.$$

(2) Also ist die Anzahl der Enthaltungen und Fehlentscheidungen von Winnow durch

$$e^* + f^* \leq 5e^* + 4 \leq 5 \cdot \text{Strafe}_T(E) + 5 \cdot |E| \cdot \lceil \log_2 n \rceil + 4$$

beschränkt.

On-line Lernen von Konzeptklassen

Wir versuchen ein Konzept $c \in \mathcal{C}$ mit Hilfe von Hypothesen $h \in \mathcal{H}$ zu lernen.
Ein Lehrer gibt uns (mgl. wenig informative) **Gegenbeispiele**.

Die Lernsituation:

(1) Bestimme eine Hypothese $h \in \mathcal{H}$.
(2) Solange $h \neq c$:
   (a) fordere ein Gegenbeispiel x (mit $x \in c \oplus h$) an und
   (b) bestimme eine neue Hypothese h.

Unser Ziel

Rekonstruiere c nach möglichst wenigen Gegenbeispielen exakt.

Wieviele Gegenbeispiele?

Wir setzen

$$\text{Gegenbeispiel}_{\mathcal{H}}(\mathcal{C}) = m$$

(1) falls es einen Lernalgorithmus gibt, der jedes Konzept in $\mathcal{C}$ nach höchstens m Gegenbeispielen exakt rekonstruiert und

(2) falls jeder Lernalgorithmus für mindestens ein Zielkonzept in $\mathcal{C}$ und für mindestens eine Folge von Gegenbeispielen mindestens m Gegenbeispiele anfordern muss.

(3) Alle Lernalgorithmen müssen mit Hypothesen $h \in \mathcal{H}$ arbeiten.

Wir können Hypothesen $h \in \mathcal{H}$ als Experten auffassen: Der Weighted Majority Algorithmus wie auch der Winnow Algorithmus sollten im Folgenden eine wichtige Rolle spielen.

MONOM_n

(1) Wir präsentieren als erstes die widersprüchliche Hypothese

$$h \equiv x_1 \wedge \neg x_1 \wedge x_2 \wedge \neg x_2 \wedge \cdots \wedge x_n \wedge \neg x_n$$

und entfernen stets nur die einem Beispiel widersprechenden Literale.

(2) Das erste Gegenbeispiel entfernt genau n Literale aus h.

(3) Der Lehrer kann nur positive Gegenbeispiele geben, denn unsere Hypothese ist stets „maximal konsistent“.

(4) Nach jedem positiven Gegenbeispiel wird mindestens ein Literal entfernt.

Da wir mit $2n$ Literalen beginnen:
$\text{Gegenbeispiel}_{\text{MONOM}_N}(\text{MONOM}_n) \leq n$.

Gegenbeispiele und VC-Dimension

Ist das jetzt ein guter Algorithmus? Wie groß muss die Anzahl der Gegenbeispiele mindestens sein?

(1) Die Konzeptklasse sei $\mathcal{C}$ und es gelte $\mathrm{VC}(\mathcal{C}) = s$.

(2) Die Beispielmenge S werde zertrümmert.

- Jede Teilmenge von S tritt als Durchschnitt von S und einem Konzept aus $\mathcal{C}$ auf.
- Wenn der Lehrer nur Gegenbeispiele aus S wählt, dann bleibt mit jedem Gegenbeispiel das Schicksal der noch nicht genannten Beispiele aus S offen:

Für jede Hypothesenklasse $\mathcal{H}$ ist

$$\mathrm{Gegenbeispiel}_{\mathcal{H}}(\mathcal{C}) \geq \mathrm{VC}(\mathcal{C}).$$

Der Weighted Majority Ansatz

(1) Fasse die Konzepte in $\mathcal{C}$ als Experten auf:
- Zu Anfang erhält jedes Konzept $c \in \mathcal{C}$ das Gewicht $w_c = 1$.
- Wir präsentieren in jedem Schritt die Mehrheitshypothese M mit

$$x \in M \Leftrightarrow \sum_{c \in \mathcal{C},\, x \in c} w_c \geq \sum_{c \in \mathcal{C},\, x \notin c} w_c.$$

- Nach einem Gegenbeispiel x bestrafen wir jedes falsch klassifizierende Konzept $c \in \mathcal{C}$ mit dem neuen Gewicht $w_c = 0$. Die Gewichte der sonstigen Konzepte bleiben unverändert.

(2) $\log_2 |\mathcal{C}|$ Gegenbeispiele sind ausreichend, da jedes Gegenbeispiel mindestens die Hälfte aller verbleibenden Konzepte eliminiert.

Für eine endliche Konzeptklasse $\mathcal{C}$ gibt es eine Hypothesenklasse $\mathcal{H}$ mit

$$\text{Gegenbeispiel}_{\mathcal{H}}(\mathcal{C}) \leq \log_2 |\mathcal{C}|.$$

Monotone Disjunktionen

- Wir versuchen, den Winnow Algorithmus für das Lernen monotoner Disjunktionen

$$x_{i_1} \vee \cdots \vee x_{i_k}$$

einzusetzen, wenn k, die Anzahl der relevanten Attribute klein ist.

- Bisher haben wir bis zu n Gegenbeispiele benötigt, wenn aus n Attributen auszuwählen ist.

Unser Ziel

Höchstens $2 + 2k \cdot \log_2 n$ Gegenbeispiele, wenn das Zielkonzept k Attribute aus n Attributen auswählt.

Der Zusammenhang zu Winnow:

Fasse die positiven Literale als Spezialisten auf.

Der Winnow Algorithmus für monotone Disjunktionen

(1) Setze $t = \frac{n}{2}$ und $w_1 = \cdots = w_n = 1$. Die Mehrheitsfunktion

$$m(x_1, \ldots, x_n) = 1 \quad \Leftrightarrow \quad \sum x_i = \sum w_i x_i \geq t = \frac{n}{2}$$

wird als Anfangshypothese verwendet.

(2) Bei einem positiven Gegenbeispiel x (also einem Beispiel x das zum Zielkonzept gehört, für das aber $\sum w_i x_i < t$ gilt):

Verdopple w_i genau dann, wenn $x_i = 1$.

/* Winnow belohnt „Nicht-Enthaltungen“. */

Bei einem negativen Gegenbeispiel x (also einem Beispiel x das nicht zum Zielkonzept gehört, für das aber $\sum w_i x_i \geq t$ gilt):

Setze $w_i = 0$ genau dann, wenn $x_i = 1$.

/* Winnow bestraft eine „Fehlentscheidung“ diesmal drakonisch. */

Die Analyse I

Sei b_+ die Anzahl positiver und sei b_- die Anzahl negativer Gegenbeispiele. Dann ist

$$n + b_+ \cdot t - b_- \cdot t \geq 0.$$

(1) Bei einem positiven Gegenbeispiel wird das Gesamtgewicht der zu positiven Literalen gehörenden Gewichte verdoppelt.

Aber deren Gesamtgewicht hat t nicht erreicht: Ein positives Gegenbeispiel erhöht das Gesamtgewicht um höchstens t.

(2) Ein Bestrafungsschritt erniedrigt das Gesamtgewicht um mindestens t.

(3) Das anfängliche Gesamtgewicht beträgt n und $n + b_+ \cdot t - b_- \cdot t$ ist eine obere Schranke für das Gesamtgewicht, wenn Winnow terminiert.

(4) Die Behauptung folgt: Das Gesamtgewicht ist stets nicht-negativ.

Die Analyse II

Es kann nicht zu viele negative Gegenbeispiele geben. Aber warum ist auch die Anzahl positiver Gegenbeispiele klein?

(1) Für jedes Gewicht w_i gilt stets $w_i \leq 2t$: Ein Gewicht w_i mit $w_i \geq t$ nimmt nie an einem Belohnungsschritt teil.

(2) Nach b_+ positiven Gegenbeispielen gibt es ein Gewicht w_i mit $\log_2 w_i \geq \frac{b_+}{k}$.
   - Jedes positive Gegenbeispiel verdoppelt das Gewicht von mindestens einem der k Literale des Zielkonzepts.
   - kein Literal des Zielkonzepts wird je bestraft: Es gilt $w_i \geq 2^{b_+/k}$ für mindestens ein Literal x_i des Zielkonzepts.

(3) Es gibt ein Literal x_i mit $\frac{b_+}{k} \leq \log_2 w_i \leq \log_2 t + 1$.

Die Analyse III

Was wissen wir?

- Es ist $n + b_+ \cdot t - b_- \cdot t \geq 0$.
- Es gibt ein Literal x_i mit $\frac{b_+}{k} \leq \log_2 w_i \leq \log_2 t + 1$.

(1) Also ist $b_+ \leq k \cdot (\log_2 t + 1)$ und es gibt höchstens $k \cdot (\log_2 t + 1) = k \cdot \log_2 n$ positive Gegenbeispiele.

(2) Wieviele negative Gegenbeispiele kann es geben?
Es ist $b_- \leq \frac{n}{t} + b_+ \leq \frac{n}{t} + k \cdot (\log_2 t + 1) = 2 + k \cdot \log_2 n$.

Die Anzahl der Gegenbeispiele ist durch $2 + 2k \cdot \log_2 n$ beschränkt.

Weitere Anwendungen

Wir nehmen an, dass alle Beispiele die Länge n haben.

(1) Allgemeine Disjunktionen und Monome mit k Literalen können nach höchstens $O(k \cdot \log_2 n)$ Gegenbeispielen gelernt werden.

- Durch das Hinzufügen der n neuen Literale $x'_1, \ldots, x'_n$ (mit $x'_i = \neg x_i$) werden nicht-monotone Disjunktionen zu monotonen Disjunktionen.
- Das Lernen von Monomen ist äquivalent zum Lernen ihrer Komplemente, nämlich der Disjunktionen.

(2) Formeln in disjunktiver Normalform mit höchstens k Literalen pro Monom und höchstens s Monomen werden nach höchstens $O(s \cdot k \cdot \log_2 n)$ Gegenbeispielen gelernt.

- Benutze statt Eingabe $x = (x_1, \ldots, x_n)$ das Ergebnis von x auf allen möglichen Monomen mit k Literalen.
- Wenn das Zielkonzept eine Disjunktion von s Monomen ist, benötigt Winnow höchstens $O(s \cdot \log_2((2n)^k)) = O(s \cdot k \cdot \log_2 n)$ Gegenbeispiele.

Diskussion

- Wir können also die Konzeptklasse **k-DNF**$_n$ nach höchstens $O(s \cdot k \log_2 n)$ Gegenbeispielen lernen, wenn das Zielmonom nur aus s Monomen besteht.
- Allerdings müssen wir die Laufzeit $n^{O(k)}$ investieren, da alle Monome mit höchstens k Literalen als neue Eingaben auftreten.

(1) Im „**r aus k**“ Problem werden k unbekannte relevante Variablen gesucht. Eine Eingabe x wird akzeptiert, wenn mindestens r der k Variablen wahr sind.

(2) Wir benötigen bisher die Laufzeit $\Omega(n^{O(r)})$.

Aber es geht wesentlich schneller.

Das r aus k Problem

Der angepasste Winnow-Algorithmus:

(1) Der Parameter r sei bekannt, k ist unbekannt. Setze $t = n$, $w_1 = \cdots = w_n = 1$ und $\varepsilon = \frac{1}{2r}$. Die Mehrheitsfunktion $m(x_1, \ldots, x_n) = 1 \quad \Leftrightarrow \quad \sum x_i = \sum w_i x_i \geq t = n$ wird als Anfangshypothese verwendet.

(2) Bei einem positiven Gegenbeispiel x (also einem Beispiel x das zum Zielkonzept gehört, für das aber $\sum w_i x_i < t$ gilt):

Ersetze w_i durch $w_i = (1 + \varepsilon) \cdot w_i$ genau dann, wenn $x_i = 1$.

Bei einem negativen Gegenbeispiel x (also einem Beispiel x das nicht zum Zielkonzept gehört, für das aber $\sum w_i x_i \geq t$ gilt):

Ersetze w_i durch $w_i = w_i/(1 + \varepsilon)$ genau dann, wenn $x_i = 1$.

Das „r aus k“ Problem: Die Analyse I

Es ist $t = n$ und $\varepsilon = \frac{1}{2r}$.

(1) Bei einem positiven Gegenbeispiel wächst das Gesamtgewicht um höchstens $\varepsilon \cdot t$ an, bei einem negativen Gegenbeispiel fällt das Gesamtgewicht um mindestens

$$t - \frac{t}{1+\varepsilon} = t \cdot \frac{\varepsilon}{1+\varepsilon}.$$

(2) Wenn b_+ die Anzahl positiver und b_- die Anzahl negativer Gegenbeispiele ist, dann folgt $n + b_+ \cdot \varepsilon \cdot t - b_- \cdot t \cdot \frac{\varepsilon}{1+\varepsilon} \geq 0$ beziehungsweise

$$b_+ \geq b_- \cdot \frac{1}{1+\varepsilon} - 2r.$$

Das „r aus k“ Problem: Die Analyse II

(3) Keine Variable kann an einem Belohnungsschritt teilnehmen, wenn ihr Wert größer als t ist:

> Für keine Variable ist die Anzahl ihrer Belohnungsschritte größer als die Anzahl ihrer Bestrafungsschritte plus $\log_{1+\varepsilon} t$.

(4) Ein positives Gegenbeispiel erhöht die Gewichte von mindestens r relevanten Variablen, ein negatives Gegenbeispiel erniedrigt höchstens $r - 1$ Gewichte relevanter Variablen.

- Durchschnittlich nimmt eine relevante Variable an mindestens $\frac{r \cdot b_+}{k}$ Belohnungsschritten und an höchstens $\frac{(r-1) \cdot b_-}{k}$ Bestrafungsschritten teil.
- Die Differenz der beiden Durchschnitte ist durch $\log_{1+\varepsilon} t$ beschränkt.

(5) Als Konsequenz: $r \cdot b_+ - (r-1) \cdot b_- \leq k \cdot \log_{1+\varepsilon} t$.

Das „r aus k“ Problem: Die Analyse III

Die beiden wesentlichen Zwischenergebnisse:

- $b_+ \geq b_- \cdot \frac{1}{1+\varepsilon} - 2r$: Die Anzahl der Belohnungsschritte ist nicht sehr viel kleiner als die Anzahl der Bestrafungsschritte.
- $r \cdot b_+ - (r-1) \cdot b_- \leq k \cdot \log_{1+\varepsilon} t$: Die Anzahl der Belohnungsschritte ist nicht sehr viel größer als die Anzahl der Bestrafungsschritte.

(6) Als Konsequenz: $b_+, b_- = O(k \cdot \log_{1+\varepsilon}(t))$.

(7) Es ist $\varepsilon = \frac{1}{2r}$ und deshalb folgt $\log_{1+\varepsilon} t = (\log_{1+\varepsilon} 2) \cdot \log_2 n = \Theta(r \cdot \log_2 n)$.

Winnow benötigt für das „r aus k“ Problem bei n Variablen höchstens $O(r \cdot k \cdot \log_2 n)$ Gegenbeispiele.

Das Perzeptron I

Das Perzeptron besteht aus einem Gatter mit Eingaben $x_1, \ldots, x_n$, das die Thresholdfunktion

$$f(x_1, \ldots, x_n) = \begin{cases} 1 & \sum_{i=1}^{n} w_i x_i + t \geq 0 \\ -1 & \text{sonst.} \end{cases}$$

berechnet.

- Nur „linear trennbare“ boolesche Funktionen lassen sich durch ein Perzeptron berechnen:

  ODER sowie **UND** sind linear trennbar, **XOR** nicht.
- Die Berechnungskraft nimmt stark zu, wenn mehrlagige Perzeptrons (bzw. Threshold-Schaltkreise) betrachtet werden:

  Neuronale Netzwerke wurden „aus“ dem Perzeptron entwickelt.
- Das Perzeptron ist Grundlage der Support-Vektor Maschinen.

Das Perzeptron II

$$f(x_1, \ldots, x_n) = \begin{cases} 1 & \sum_{i=1}^{n} w_i x_i + t \geq 0 \\ -1 & \text{sonst.} \end{cases}$$

sei eine Thresholdfunktion.

- $w_1, \ldots, w_n$ sind die Gewichte von f und t ist sein Schwellenwert.
- In Vektornotation

$$f(x_1, \ldots, x_n) = 1 \Leftrightarrow \langle w, x \rangle + t \geq 0.$$

- Wir haben das Konsistenzproblem für Perzeptrons mit Hilfe der linearen Programmierung gelöst.
- Wir möchten als Nächstes einen einfachen On-line Algorithmus entwerfen.

Der Perzeptron Algorithmus

(0) Eine unbekannte Hyperebene $\langle w, x \rangle + t = 0$ wird gesucht, die alle negativen Beispiele von den positiven Beispielen trennt.

(1) Setze $i = 0$. Weiterhin initialisiere $w^{(0)} = (0, 0, \ldots, 0)$ sowie $t = 0$, d.h. wir akzeptieren zu Anfang alle Eingaben.

(2) WHILE (noch nicht gelernt) DO

(2a) Fordere ein Gegenbeispiel $x^{(i)} = (x_1, \ldots, x_n)$ mit korrekter Klassifizierung b an. /*Es ist $b = 1$ (bzw. $b = -1$), wenn $x^{(i)}$ zu akzeptieren (bzw. zu verwerfen) ist. */

(2b) Setze

$$w^{(i+1)} = w^{(i)} + b \cdot x^{(i)} \quad \text{und} \quad t^{(i+1)} = t^{(i)} + b \cdot R^2.$$

/*Bei einem positiven (bzw. negativen) Gegenbeispiel werden die Gewichte und der Schwellenwert t erhöht (bzw. erniedrigt). */

(2c) $i = i + 1$.

(3) Gib den Gewichtsvektor $w^{(i)}$ und den Schwellenwert $t^{(i)}$ aus.

Diskussion

Annahme: Alle Beispiele besitzen die euklidsche Norm höchstens R.

- Der Perzeptron-Algorithmus ändert seine Gewichte additiv und wird deshalb auch als additives Lernverfahren bezeichnet. Winnow ist hingegen ein multiplikatives Lernverfahren.
- Angenommen, eine Hyperebene trennt die Beispielmenge korrekt. Definiere den **Margin** von H als den minimalen Abstand eines Beispiels von der Hyperebene H.

Die zentrale Frage: Lernt der Perzeptron Algorithmus und wieviel Gegenbeispiele werden benötigt?

Das Lernproblem sollte umso schwieriger sein, je kleiner das **größtmögliche** Margin ist.

Analytische Geometrie

(1) Die Vektoren $x, y \in \mathbb{R}^n$ mögen die Länge Eins haben. Dann

$$\text{Kosinus}(x, y) = \langle x, y \rangle.$$

(2) Wir erhalten als Konsequenz die Ungleichung von Cauchy und Schwartz

$$\langle u, v \rangle \leq ||u|| \cdot ||v||.$$

(3) Der Abstand eines Vektors $x \in \mathbb{R}^n$ von der Hyperebene $\langle w, y \rangle + t = 0$ stimmt überein mit dem Absolutbetrag von

$$\frac{\langle w, x \rangle + t}{||w||}$$

.

Der Margin

Eine Menge $S \subseteq \mathbb{R}$ sei gegeben ebenso wie eine Hyperebene $\langle w, x \rangle + t = 0$.

(1) Der Margin eines mit b klassifizierten Beispiels $x \in \mathbb{R}^n$ ist

$$\text{Margin}(x, b, w, t) = b \cdot (\langle w, x \rangle + t\ )$$

- Margin(x, b, w, t) ist genau dann negativ, wenn die Thresholdfunktion (w, t) das Beispiel falsch klassifiziert.
- Der Absolutbetrag des Bruchs $\frac{\text{Margin}(x,b,w,t)}{||w||}$ misst den Abstand des Beispiels x von der Hyperebene $(w, x) + t = 0$.

(2) Der Margin Margin(S, f) der Beispielmenge S für eine Klassifikation $f : S \to \{-1, 1\}$ ist der

größtmögliche minimale Abstand eines Beispiels aus S zu einer Hyperebene, die die positiven von den negativen Beispielen trennt.

Der Margin von UND

$$f(x) = 1 \quad \Leftrightarrow \quad x_1 \wedge \cdots \wedge x_n \text{ ist wahr.}$$

(1) Setze $w = (1, \ldots, 1)$ und $S = \{0, 1\}^n$:

- Die Hyperbene $\langle w, x \rangle - n + \frac{1}{2}$ trennt den 1-Vektor von allen anderen Vektoren.
- Für alle $x \in S$ folgt

$$\text{Margin}(x, \text{UND}(x), w, t) \geq \frac{1}{2}$$

- und als Konsequenz

$$\text{Margin}(S, \text{UND}) \geq \frac{1}{2 \cdot ||w||} = \frac{1}{2 \cdot \sqrt{n}}.$$

(2) Die Und-Funktion hat also einen Margin von mindestens $\Omega(\frac{1}{\sqrt{n}})$.

Perzeptron Algorithmus: Die Lerngeschwindigkeit I

Die Beispielmenge S und die Klassifikation f seien gegeben.

S sei in der Kugel mit Radius R um 0 enthalten.

Wenn nur Gegenbeispiele aus der Menge S gewählt werden, dann lernt der Perzeptron-Algorithmus erfolgreich nach höchstens

$$\left(\frac{2 \cdot R}{\text{Margin}(S, f)}\right)^2$$

Gegenbeispielen.

(1) Die UND-Funktion für Beispielmenge $S = \{0, 1\}^n$:

- S ist in der Kugel mit Radius $R = \sqrt{n}$ um 0 enthalten.
- $\text{Margin}(S, \text{UND}) = \Omega(\frac{1}{\sqrt{n}})$.
- Erfolgreiches Lernen nach $O(n^2)$ Gegenbeispielen.
- Aber linear viele Gegenbeispiele reichen für andere Verfahren (Winnow).

Perzeptron Algorithmus: Die Lerngeschwindigkeit II

(2) Angenommen, die zu lernende Thresholdfunktion f besitzt n ganzzahlige Gewichte $w_1, \ldots, w_n$ und einen ganzzahligen Schwellenwert t mit

$$|w_1|, \ldots, |w_n| \leq W.$$

Für die Beispielmenge $S = \{0,1\}^n$ genügen dann $O(n^2 \cdot W^2)$ Gegenbeispiele. Warum?

- Wie klein ist $\text{Margin}(S, f)$?
  In Übungsaufgabe wird $\text{Margin}(S, f) \geq \frac{1}{\sqrt{n} \cdot W}$ gezeigt.
- Da S wieder im Ball mit Radius $R = \sqrt{n}$ enthalten ist, genügen $(\frac{2 \cdot R}{\text{Margin}(S,f)})^2 = O(n^2 \cdot W^2)$ Gegenbeispiele.
- Also wird zum Beispiel die Mehrheitsfunktion $\sum_{i=1}^{n} x_i \geq n/2$ nach höchstens $O(n^2)$ Gegenbeispielen gelernt.

Perzeptron Algorithmus: Die Lerngeschwindigkeit III

(3) Es sei $S = \{0,1\}^n$. Der Perzeptron Algorithmus berechnet Gewichte als Summe und Differenz von Beispielen in S.

Der Perzeptron Algorithmus berechnet stets ganzzahlige Gewichte.

(4) Die Funktion COMP_n vergleicht zwei Binärdarstellungen $x, y \in \{0,1\}^n$:

$$\mathrm{COMP}_n(x,y) = \begin{cases} 1 & \text{falls } \sum_{i=1}^{n} x_i 2^{i-1} \geq \sum_{i=1}^{n} y_i 2^{i-1}, \\ 0 & \text{sonst.} \end{cases}$$

- In einer Übungsaufgabe wird gezeigt, dass jede Implementierung Gewichte der Größe mindestens $\Omega(2^n/n)$ benötigt, wenn die Gewichte ganzzahlig sind.
- Mindestens $\Omega(2^n/n)$ Gegenbeispiele werden benötigt!

Analyse I

Wir heben die Sonderbehandlung der Schwellenwerte t auf und setzen

$$\hat{w} = (w, \frac{t}{R}) \quad \text{und} \quad \hat{x} = (x, R)$$

für einen (alten) Gewichtsvektor w und ein (altes) Beispiel x.

(1) Im Schritt i führt der Perzeptron Algorithmus die Aktualisierung $w^{(i)} = w^{(i-1)} + b_{i-1} \cdot x^{(i-1)}$ und $t^{(i)} = t^{(i-1)} + b_{i-1} \cdot R^2$ durch.

(2) Die Aktualisierung in der neuen Notation ist

$$\begin{aligned} \hat{w}^{(i)} &= (w^{(i)}, \frac{t^{(i)}}{R}) = (w^{(i-1)}, \frac{t^{(i-1)}}{R}) + b_{i-1} \cdot (x^{(i-1)}, R) \\ &= \hat{w}^{(i-1)} + b_{i-1} \cdot \hat{x}^{(i-1)}. \end{aligned}$$

Analyse II

Es ist $\hat{w}^{(i)} = \hat{w}^{(i-1)} + b_{i-1} \cdot \hat{x}^{(i-1)}$.

(1) $x^{(i-1)}$ ist ein Gegenbeispiel: $b_{i-1} \cdot (\langle w^{(i-1)}, x^{(i-1)} \rangle + t^{(i-1)}) < 0$.

(2) Wir erhalten

$$\begin{aligned} b_{i-1} \cdot \langle \hat{w}^{(i-1)}, \hat{x}^{(i-1)} \rangle &= b_{i-1} \cdot (\langle w^{(i-1)}, x^{(i-1)} \rangle + \frac{t^{(i-1)}}{R} \cdot R) \\ &= b_{i-1} \cdot (\langle w^{(i-1)}, x^{(i-1)} \rangle + t^{(i-1)}) < 0. \end{aligned}$$

und $b_{i-1} \cdot \langle \hat{w}^{(i-1)}, \hat{x}^{(i-1)} \rangle < 0$ folgt.

Die Übersetzung von der alten in die neue Notation ist abgeschlossen:
$\hat{w}^{(i)} = \hat{w}^{(i-1)} + b_{i-1} \cdot \hat{x}^{(i-1)}$ und $b_{i-1} \cdot \langle \hat{w}^{(i-1)}, \hat{x}^{(i-1)} \rangle < 0$.

Analyse III

Es gelte $\mathrm{Margin}(x, f(x), w, t) \geq \gamma \geq 0$ für einen Gewichtsvektor w mit $||w|| = 1$ und alle Beispiele x. Dann gilt

$$\langle \hat{w}^{(i)}, \hat{w} \rangle \geq (i+1) \cdot \gamma.$$

(1) Wir führen eine Induktion über i aus:

$$\begin{aligned} \langle \hat{w}^{(i)}, \hat{w} \rangle &= \langle \hat{w}^{(i-1)} + b_{i-1} \cdot \hat{x}^{(i-1)}, \hat{w} \rangle \\ &= \langle \hat{w}^{(i-1)}, \hat{w} \rangle + b_{i-1} \cdot \langle \hat{x}^{(i-1)}, \hat{w} \rangle \\ &\geq \langle \hat{w}^{(i-1)}, \hat{w} \rangle + \gamma, \end{aligned}$$

denn $\mathrm{Margin}(x^{(i-1)}, f(x^{(i-1)}), w, t) = b_{i-1} \cdot \langle \hat{x}^{(i-1)}, \hat{w} \rangle \geq \gamma$ nach Annahme.

(2) Aber $\langle \hat{w}^{(i-1)}, \hat{w} \rangle \geq i \cdot \gamma$ nach Induktionsannahme.

Wie weit sind wir?

Wir wissen $\langle \hat{w}^{(i)}, \hat{w} \rangle \geq (i+1) \cdot \gamma$.

(1) Also ist

$$\langle \frac{\hat{w}^{(i)}}{||\hat{w}^{(i)}||}, \frac{\hat{w}}{||\hat{w}||} \rangle = \text{Kosinus}(\hat{w}^{(i)}, \hat{w}) \geq \frac{(i+1) \cdot \gamma}{||\hat{w}^{(i)}|| \cdot ||\hat{w}||}.$$

(2) Also konvergiert der Kosinus gegen 1, wenn $||\hat{w}^{(i)}|| = o(i)$.

Wie schnell wächst $||\hat{w}^{(i)}||$?

Analyse IV

Es ist stets

$$||\hat{w}^{(i)}||^2 \leq 2 \cdot (i+1) \cdot R^2.$$

(1) Warum? Wir führen wieder eine Induktion über i aus:

$$\begin{aligned}
||\hat{w}^{(i)}||^2 &= ||\hat{w}^{(i-1)}||^2 + 2 \cdot b_{i-1} \cdot \langle \hat{w}^{(i-1)}, \hat{x}^{(i-1)} \rangle + b_{i-1}^2 \cdot ||\hat{x}^{(i-1)}||^2 \\
&\leq ||\hat{w}^{(i-1)}||^2 + b_{i-1}^2 \cdot ||\hat{x}^{(i-1)}||^2 \\
&\quad \text{denn } b_{i-1} \cdot \langle \hat{w}^{(i-1)}, \hat{x}^{(i-1)} \rangle < 0, \\
&= ||\hat{w}^{(i-1)}||^2 + (\, ||x^{(i-1)}||^2 + R^2 \,) \\
&\quad \text{denn } b_{i-1}^2 = 1 \text{ und } \hat{x}^{(i-1)} = (x^{(i-1)}, R), \\
&\leq ||\hat{w}^{(i-1)}||^2 + 2R^2, \quad \text{denn } ||x^{(i-1)}||^2 \leq R.
\end{aligned}$$

(2) Und $||\hat{w}^{(i-1)}||^2 \leq 2 \cdot i \cdot R^2$ nach Induktionsannahme.

Wie weit sind wir?

Wir wissen

- $\langle \hat{w}^{(i)}, \hat{w} \rangle \geq (i+1) \cdot \gamma$ sowie
- $||\hat{w}^{(i)}||^2 \leq 2 \cdot (i+1) \cdot R^2$.

(1) Wir benutzen die Ungleichung $\langle a, b \rangle \leq ||a| \cdot ||b||$ von Cauchy und Schwartz und erhalten

$$(i+1) \cdot \gamma \leq \langle \hat{w}^{(i)}, \hat{w} \rangle \leq ||\hat{w}^{(i)}|| \cdot ||\hat{w}|| \leq \sqrt{2(i+1)} \cdot R \cdot ||\hat{w}||.$$

(2) Wir können $||w|| = 1$ annehmen. Da Beispiele aus dem Ball um 0 mit Radius R gewählt werden, folgt $|t| \leq R$.

(3) Deshalb ist $||\hat{w}||^2 = ||w||^2 + (\frac{t}{R})^2 \leq ||w||^2 + 1 = 2$.

(4) Als Konsequenz:

$$i + 1 \leq 2 \cdot (\frac{R}{\gamma})^2 \cdot ||\hat{w}||^2 \leq (\frac{2R}{\gamma})^2.$$