Online Lernen: Die Themen

(a) Das **Online-Spiel**: In jeder Runde präsentiert ein Lehrer ein Beispiel, das ein Schüler klassifiziert. Nach wie vielen Runden hat der Schüler das unbekannte Zielkonzept gelernt?

- Charakterisierung der minimalen Rundenzahl.
- Zusammenhang zur VC-Dimension.
- Der Zusammenhang zwischen Online- und PAC-Algorithmen.

(b) Wichtige algorithmische Ideen:

- Der **Halbierungs-Algorithmus**,
- **Weighted-Majority**: Auswahl von Experten,
- **Winnow**: Relevante Eigenschaften schnell bestimmen!
- der **Perzeptron-Algorithmus**: das Online-Lernen von Halbräumen.

Das Online-Spiel

Das Online-Spiel

1. Das Spiel beginnt in Runde $t = 1$.
2. Solange das Spiel noch nicht beendet ist:
   (a) Der Lehrer präsentiert ein Beispiel x_t.
   (b) Der Schüler gibt eine Klassifizierung an.

   /* Der Schüler ist an keine Hypothesenklasse gebunden. */

   - ⋆ Ist die Klassizierung richtig, ist das Spiel beendet: Der Schüler hat gewonnen.
   - ⋆ Bei einer falschen Antwort (x_t ist ein **Gegenbeispiel**), erhält der Schüler die richtige Klassifikation.

   (d) Setze $t := t + 1$.

- Der Schüler möchte möchte so schnell wie möglich gewinnen.
- Der Lehrer ist bösartig.

Gegenbeispiel-Komplexität

(a) Sei A ein Algorithmus, mit dem der Schüler Beispiele klassifizieren kann. Dann ist

$$\textbf{Gegenbeispiel}_A(\mathcal{C}) := m,$$

falls bei Klassifikation der Beispiele durch A

- $(*)$ jedes Konzept in $\mathcal{C}$ nach höchstens m Gegenbeispielen für jeden Lehrer gelernt wird
- $(*)$ und mindestens m Gegenbeispiele für irgendeinen Lehrer und irgendein Konzept benötigt werden.

(b) Es ist

$$\textbf{Gegenbeispiel}(\mathcal{C}) := m,$$

falls es einen Algorithmus A mit $\text{Gegenbeispiel}_A(\mathcal{C}) = m$ gibt und falls $\text{Gegenbeispiel}_B(\mathcal{C}) \geqslant m$ für jeden Algorithmus B gilt.

Gegenbeispiel-Komplexität: Konzeptklassen

(a) Sei $X = \{1, \ldots, n\}$. Die Konzeptklasse $\mathbf{1}_n$ bestehe aus allen

„Einermengen“ $\{i\}$ für $1 \leqslant i \leqslant n$.

- Es ist Gegenbeispiel$(\mathbf{1}_n) = 1$:
  Der Schüler behauptet, dass x_1 ein negatives Beispiel ist :-))

(b) Ein Online-Algorithmus für $\mathbf{MONOM}_n$:

- Der Schüler klassifiziert anfänglich mit der Hypothese

$$x_1 \wedge \neg x_1 \wedge x_2 \wedge \neg x_2 \wedge \cdots \wedge x_n \wedge \neg x_n.$$

- Der Schüler erhält ein Gegenbeispiel und arbeitet mit der maximal-konsistenten Hypothese weiter.
- Der Lehrer kann nur positive Gegenbeispiele geben. Es ist

$$\text{Gegenbeispiel}(\mathbf{MONOM}_n) \leqslant n + 1.$$

Gegenbeispiele und VC-Dimension

Für jede Konzeptklasse $\mathcal{C}$ gilt

$$\text{VC}(\mathcal{C}) \leqslant \text{Gegenbeispiel}(\mathcal{C}).$$

1. Sei $s = \text{VC}(\mathcal{C})$ und $\mathcal{C}$ zertrümmere die Menge $S = \{x_1, \ldots, x_s\}$.
2. $\Longrightarrow$ **jede** Teilmenge von S ist ein Konzept.
3. $\Longrightarrow \text{Gegenbeispiel}(\mathcal{C}) \geqslant s$. □

Eine erste Konsequenz:

$$n \leqslant \text{Gegenbeispiel}(\textbf{MONOM}_n) \leqslant n + 1.$$

Endliche Konzeptklassen

Die Konzeptklasse $\mathcal{C}$ bestehe aus endlich vielen Konzepten. Dann folgt

$$\text{Gegenbeispiel}(\mathcal{C}) \leqslant \log_2 |\mathcal{C}|.$$

1. Der Schüler führt den **Halbierungs-Algorithmus** aus
   - und klassifiziert mit der Mehrheitshypothese

$$x \in M \;:\Longleftrightarrow\; \Big|\{c \in \mathcal{C} : x \in c\}\Big| \geqslant \Big|\{c \in \mathcal{C} : x \notin c\}\Big|.$$

   - Jedes falsch klassifizierende Konzept $c \in \mathcal{C}$ wird aus $\mathcal{C}$ entfernt.
2. Ein Gegenbeispiel eliminiert mindestens die Hälfte aller Konzepte.

Insbesondere: $\text{Gegenbeispiel}_{\text{Halbierung}}(\mathcal{C}) \leqslant \log_2 |\mathcal{C}|$.

Wie viele Gegenbeispiele?
Eine exakte Charakterisierung

Wie arbeitet ein Lehrer?

(a) Der Lehrer beginnt mit einem ersten Beispiel $b_\epsilon := x_1$.

(b) Wenn der Lehrer gerade Beispiel b_w präsentiert, dann
- zeigt er Beispiel b_{w0} bei einer negativen Klassifizierung von b_w durch den Schüler
- bzw b_{w1} bei einer positiven Klassifizierung.

(c) Die Strategie des Lehrers wird durch einen **vollständigen** binären, geordneten Baum $\mathcal{B}$ modelliert:
- Innere Knoten w sind mit den Beispielen b_w markiert
- und Kanten mit der Antwort des Schülers.

! Es muss zu jedem Weg W ein Konzept $c \in \mathcal{C}$ geben, so dass c **konsistent** mit den Klassifizierungen von W ist.

1. Ein solcher Baum $\mathcal{B}$ heißt **konsistent** mit $\mathcal{C}$.
2. **Tiefe**($\mathcal{C}$) ist die **maximale** Tiefe eines mit $\mathcal{C}$ konsistenten Baums.
3. Tiefe($\mathcal{C}$) $\leqslant$ Gegenbeispiel($\mathcal{C}$) ✓

Tiefe($\mathcal{C}$) $\geqslant$ Gegenbeispiel($\mathcal{C}$)

Ein pfiffiger Schüler:

1. Setze $i = 1$ und $\mathcal{C}_i := \mathcal{C}$.
2. Wiederhole, solange der Schüler noch nicht gewonnen hat:
   (a) Wenn der Lehrer das Beispiel x vorlegt, dann
   (b) bestimme $\mathcal{C}_i^{(0)} := \{c \in \mathcal{C}_i : x \notin c\}$ und $\mathcal{C}_i^{(1)} := \{c \in \mathcal{C}_i : x \in c\}$.
   (c) Wenn Tiefe($\mathcal{C}_i^{(0)}$) $\geqslant$ Tiefe($\mathcal{C}_i^{(1)}$), dann antworte mit Klassifizierung $a = 0$ und sonst mit Klassifizierung $a = 1$.
   (d) Ist die Antwort falsch, dann setze $\mathcal{C}_{i+1} := \mathcal{C}_i^{(1-a)}$ und $i = i + 1$.

Sei Tiefe($\mathcal{C}$) $= t$.

- Wenn Tiefe($\mathcal{C}_i^{(0)}$) $<$ Tiefe($\mathcal{C}_i$) oder Tiefe($\mathcal{C}_i^{(1)}$) $<$ Tiefe($\mathcal{C}_i$), dann sind wir fertig: Ein Gegenbeispiel senkt die Tiefe um mindestens 1.
- Also ist Tiefe($\mathcal{C}_i^0$) $=$ Tiefe($\mathcal{C}_i^1$) $=$ Tiefe($\mathcal{C}_i$). Seien $B^{(0)}, B^{(1)}$ tiefste mit $\mathcal{C}_i^{(0)}$ bzw $\mathcal{C}_i^{(1)}$ konsistente Bäume $\Longrightarrow$ Tiefe($\mathcal{C}$) $= t + 1$ ↯

Zusammenfassung

(a) Für alle Konzeptklassen gilt

$$\text{VC}(\mathcal{C}) \leqslant \text{Tiefe}(\mathcal{C}) = \text{Gegenbeispiel}(\mathcal{C}).$$

(b) Für Konzeptklassen mit endlich vielen Konzepten folgt zusätzlich

$$\begin{aligned} \text{VC}(\mathcal{C}) \leqslant \text{Tiefe}(\mathcal{C}) &= \text{Gegenbeispiel}(\mathcal{C}) \\ &\leqslant \text{Gegenbeispiel}_{\text{Halbierung}}(\mathcal{C}) \leqslant \log_2 |\mathcal{C}| \end{aligned}$$

Die Unterschiede zwischen

- „VC-Dimension und Gegenbeispielzahl“ sowie zwischen
- „Gegenbeispielzahl und der logarithmierten Konzeptzahl“

können unbeschränkt groß werden.

Auswahl von Experten

Die Auswahl von Experten

! n Experten geben in mehreren Runden Ja/Nein Empfehlungen ab

wie etwa zum Kauf oder Verkauf einer Aktie.

! Nach jeder Runde wird mitgeteilt, welche Empfehlung richtig ist.

? Gibt es eine Strategie, die fast mit dem besten Experten mithält?

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

Eine erste Anwendung: Lerne eine unbekannte Threshold-Funktion

$$\text{sign}(\sum_{i=1}^{n} w_i x_i)$$

Interpretiere die Variablen x_i als Experten.

Weighted-Majority, die deterministische Variante

Weighted-Majority: Die deterministische Variante

Es gelte $0 < \beta < 1$ für den „Bestrafungsparameter“ β.

1. Setze $w_i = 1$ für alle Experten i.
2. Wenn Experte i die Empfehlung $x_i \in \{\text{Ja}, \text{Nein}\}$ abgibt, dann treffe die Entscheidung „**Ja**“, falls

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

   und ansonsten treffe die Entscheidung „**Nein**“.
3. Nach Erhalt der richtigen Entscheidung: Hat Experte i eine falsche Empfehlung abgegeben, dann bestrafe ihn mit der Setzung

$$w_i = \beta \cdot w_i.$$

   Ansonsten lasse das Gewicht w_i unverändert.

Analyse

(1) Sei W_t das Gesamtgewicht aller Experten zu 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:

$$W_{t+1} \leqslant \frac{W_t}{2} + \beta \cdot \frac{W_t}{2} = \frac{1+\beta}{2} W_t.$$

(3) Bei f falschen Entscheidungen bis zum Zeitpunkt t, ist

$$W_t \leqslant n \cdot (\frac{1+\beta}{2})^f.$$

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

$$\beta^{f_{\text{opt}}} \leqslant W_t \leqslant n \cdot (\frac{1+\beta}{2})^f.$$

Das Resultat

Es ist $\beta^{f_{\text{opt}}} \leqslant W_t \leqslant n \cdot (\frac{1+\beta}{2})^f$ und nach Logarithmierung:

$$f_{\text{opt}} \log_2 \beta \leqslant \log_2 n + f \log_2 \frac{1+\beta}{2}.$$

(a) Sei f_{opt} die Anzahl der Fehlentscheidungen des besten Experten und f die Anzahl der Fehlentscheidungen von Weighted-Majority.

(b) Für n Experten und $0 < \beta < 1$ gilt

$$f \leqslant \frac{f_{\text{opt}} \cdot \log_2 \frac{1}{\beta} + \log_2 n}{\log_2 \frac{2}{1+\beta}}.$$

Nach logarithmischer „Aufwärmzeit“ wird die Prognosekraft des besten Experten erreicht bis auf den Faktor $\log_2(\frac{1}{\beta}) / \log_2(\frac{2}{1+\beta}) \geqslant 2$.

Die Methode der Analyse

Wir haben die **Potentialmethode** angewandt: Wir haben beobachtet wie schnell sich

das Potential W_t

im Vergleich zum Gewicht des besten Experten verringert.

Weighted-Majority, die randomisierte Variante

Weighted-Majority: Wir würfeln

Bewerte die **komplexe** Empfehlung eines Experten i zum Zeitpunkt t mit der „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. Setze $t := 1$ und wähle $\varepsilon \in]0, 1/2[$.
2. Wähle den Experten i mit Wahrscheinlichkeit

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

   und übernimm seine Entscheidung.
3. Berechne neue Gewichte: Setze

$$w_i = w_i(1 - \varepsilon \cdot c_i^t).$$

   Setze $t := t + 1$.
   Kommentar: Beachte $\varepsilon \cdot c_i^t \in [0, 1/2[$.

Analyse (1/3)

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

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

(4) Beachte $\log(1 - x) \leqslant -x$ und

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

Erhalten wir schlechte Noten, sinkt das Gesamtgewicht.

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

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

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

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

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

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

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

Das Ergebnis

- Es ist $-(\varepsilon + \varepsilon^2) \cdot \sum_{t \leqslant T} c_i^t \leqslant \ln n - \sum_{t \leqslant T} \varepsilon K_t$.
- 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

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

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

Weighted-Majority: Weitere Anwendungen

1. Im Lernen monotoner Threshold-Funktionen $\text{sign}(\langle w, x \rangle)$ entsprechen die Variablen x_i den Experten.
   - Siehe später den **Winnow-Algorithmus**.
2. „**AdaBoost**“ implementiert Weighted-Majority.
   - Die Ensemble-Methode baut eine bessere Hypothese aus guten Hypothesen.
3. Der **Universal Portfolio Algorithmus** von Cover ist eine weitere Implementierung von Weighted-Majority:
   - Portfolios, bestehend aus Finanzanlagen, stehen im Wettbewerb miteinander.
   - Der Markt legt die Gewichte der Portfolios fest.

Constant Rebalanced Portfolios, Der Universal Portfolio Algorithmus

Constant Rebalanced Portfolio: Die CRP-Methode

Ein Vermögen wird über mehrere Anlagezeitpunkte auf m Anlagemöglichkeiten verteilt: Anlage i erhält stets den Anteil p_i.

Zwei Aktien mit $p_1 = p_2 = 0.5$: Die erste Aktie bewegt sich nicht, 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 Anlagezeitpunkt 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 (über die verschiedenen Anlagemöglichkeiten) sollte gewählt werden?
- **Risiko-Minimierung**:
  Der Universal Portfolio Algorithmus versucht die

  erwartete Vermögensentwickung

  des CRP-Ansatzes zu erreichen.
  - Arbeiten mit N „charakteristischen“ Verteilungen.
  - Zu jedem Anlagezeitpunkt: Rebalanciere ein Portfolio nach seiner Verteilung.
  - Gelder dürfen nicht zwischen Portfolios transferiert werden.

Der Universal Portfolio Algorithmus und Weighted-Majority

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

Wie gut ist der Universal Portfolio Algorithmus?

Es möge m Anlagemöglichkeiten und n Anlageperioden geben.

- opt_n sei das Vermögen des **besten** Portfolios und
- e_n das **erwartete** Vermögen nach n Anlageperioden $\Longrightarrow$

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

- Für $n >> m$ ist der durchschnittliche relative Verlustfaktor pro Anlageperiode (gegenüber der optimalen Strategie) 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 $\lim_{n\to\infty} a_n = \lim_{n\to\infty} b_n$.

Was passiert nach Steuern und Gebühren?

Der Winnow-Algorithmus für Disjunktionen

Der Winnow Algorithmus für monotone Disjunktionen

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

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

wird als Anfangshypothese verwendet.

(2) Wiederhole, solange es Gegenbeispiele gibt.

- Wenn x ein positives Gegenbeispiel ist: Verdopple w_i, falls $x_i = 1$.
  /* x_i könnte ein „relevantes Attribut“ sein. */
- Wenn x ein negatives Gegenbeispiel ist: Setze $w_i = 0$, falls $x_i = 1$.
  /* Das nicht relevante Attribut x_i wird eliminiert. */

Die Analyse (1/3)

b_+ und b_- sind die Anzahlen positiver und negativer Gegenbeispiele. Dann ist

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

(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 aktuelle Gesamtgewicht.

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

Die Analyse (2/3)

Das Zielkonzept besitze genau k Literale.

Warum ist die Anzahl positiver Gegenbeispiele klein?

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

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

(3) Es gibt ein Literal x_i mit

$$\frac{b_+}{k} \leqslant \log_2 w_i \leqslant \log_2 t + 1.$$

Die Analyse (3/3)

Was wissen wir?

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

(1) Also ist

$$b_+ \leqslant k \cdot (\log_2 t + 1).$$

Es gibt höchstens $k \cdot (\log_2 t + 1)$ positive Gegenbeispiele.

(2) Wieviele negative Gegenbeispiele kann es geben? Es ist

$$b_- \leqslant \frac{n}{t} + b_+ \leqslant \frac{n}{t} + k \cdot (\log_2 t + 1) = 2 + k \cdot \log_2 n$$

.

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

Weitere Anwendungen

Annahme: Alle Beispiele liegen in $\{0,1\}^n$.

(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 Negationen, nämlich der Disjunktionen.

(2) DNFs 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 $N = O((2n)^k)$ möglichen Monomen mit k Literalen.
- Ist das Zielkonzept eine Disjunktion von s Monomen, benötigt Winnow „nur“ $O(s \cdot \log_2((2n)^k)) = O(s \cdot k \cdot \log_2 n)$ Gegenbeispiele.

Diskussion

- Für die Konzeptklasse **k-DNF**$_n$ werden höchstens $O(s \cdot k \log_2 n)$ Gegenbeispielen benötigt, falls das Zielkonzept nur aus s Monomen besteht.
- Allerdings muss die Laufzeit $n^{O(k)}$ investiert werdenen, da alle Monome mit höchstens k Literalen als neue Eingaben auftreten.

Der Versuch einer weitreichenden Verallgemeinerung:

Lerne eine unbekannte, monotone Threshold-Funktion

$$\text{sign}(\sum_{i=1}^{n} w_i x_i)$$

mit nicht-negativen Koeffizienten.

Threshold-Funktionen $\text{sign}(\sum_{i=1}^{n} w_i x_i + t)$

Analytische Geometrie (1/2)

1. (Vektornotation) Für $w = (w_1, \ldots, w_n), x = (x_1, \ldots, x_n) \in \mathbb{R}^n$ gilt

$$\langle w, x \rangle := w^T \cdot x = \sum_{i=1}^{n} w_i x_i.$$

2. $H_{w,t} := \{x : \langle w, x \rangle + t = 0\}$ ist eine **Hyperebene**.
3. Eine **Drehung** des $\mathbb{R}^n$ wird durch eine **orthogonale Matrix** A beschrieben: Es ist

$$A^T \cdot A = \mathrm{Id}.$$

4. Eine Konsequenz:

$$\langle A \cdot x, A \cdot y \rangle = x^T \cdot A^T \cdot A \cdot y = \langle x, y \rangle.$$

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

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

   - Eine Konsequenz: Die Ungleichung von Cauchy und Schwartz

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

6. Die Länge der Projektion eines Vektors x auf einen Vektor w ist

$$\left| \frac{\langle w, x \rangle}{||w||} \right|.$$

7. Sei $x_0 \in H_{w,t}$. Der Abstand zwischen $x \in \mathbb{R}^n$ und $H_{w,t}$ stimmt überein mit der Länge der Projektion von $x - x_0$ auf $\frac{w}{||w||}$, d.h. mit

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

Der Margin (1/2)

Für Def $\subseteq \mathbb{R}^n$ ist $f : \text{Def} \to \{-1, 1\}$ die unbekannte Zielfunktion.
Des Weiteren ist

$$H_{w,t} = \{y \,:\, \langle w, y\rangle + t = 0\}.$$

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

$$\text{Margin}_f(x, w, t) := f(x) \cdot (\langle w, x\rangle + t\ ).$$

- $\text{Margin}_f(x, w, t) \leqslant 0 \iff \text{sign}(\langle w, x\rangle + t) \neq f(x)$.
  $(\ H_{w,t}$ klassifiziert Beispiel x richtig $\iff \text{Margin}_f(x, w, t) > 0.\ )$

(b) Der Abstand zwischen dem Vektor x und der Hyperebene $H_{w,t}$ ist

$$\left|\frac{\text{Margin}_f(x, w, t)}{\|w\|}\right|.$$

Der Margin (2/2)

(c) Sei $S \subseteq \mathbb{R}^n$ eine Menge von Beispielen. Dann ist

$$\text{Margin}_f(S, w, t) := \inf_{x \in S} \text{Margin}_f(x, w, t)$$

der Margin der Hyperebene $H_{w,t}$ auf der Beispielmenge S.

$H_{w,t}$ trennt positive von negativen Beispielen $\iff \text{Margin}_f(S, w, t) > 0$.

(d) Der Margin von S wird definiert durch

$$\text{Margin}_f(S) := \sup_{w,t} \left\{ \frac{\text{Margin}_f(S, w, t)}{\|w\|} \ : \ \text{Margin}_f(S, w, t) > 0 \right\}.$$

$\text{Margin}_f(S)$ ist der

größtmögliche minimale Abstand

eines Beispiels $x \in S$ zu einer trennenden Hyperebene.

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 $S = \text{Def} = \{0,1\}^n$:

- Die Hyperebene

$$\langle w, x \rangle - n + \frac{1}{2}$$

trennt $x = (1, \ldots, 1)$ von allen anderen Vektoren in $\{0,1\}^n$.

- Für alle $x \in S$ folgt

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

- und als Konsequenz

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

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

Vereinfachungen

Statt eine allgemeine Threshold-Funktion

$$\text{sign}(\langle x, w \rangle + t)$$

zu lernen, genügt es

(a) Threshold-Funktionen der Form $\text{sign}(\langle x, u \rangle)$ zu lernen
(Übersetze Beispiele $x \in \mathbb{R}^n$ in geeignete Beispiele $x' \in \mathbb{R}^{n+1}$.)

(b) **monotone Threshold-Funktionen** $\text{sign}(\langle x, u \rangle)$ zu lernen.
– $\text{sign}(\langle x, u \rangle)$ ist monoton, wenn $u \geqslant 0$.–
(Übersetze Beispiele $x \in \mathbb{R}^n$ in geeignete Beispiele $x' \in \mathbb{R}^{2n}$.)

Winnow lernt monotone Threshold-Funktionen

Winnow wird sanft

(a) Für monotone *Disjunktionen* bestraft Winnow irrelevante Attribute ($x_i = 1$ für ein negatives Gegenbeispiel) drakonisch ($w_i = 0$).

- **Katastrophales** Vorgehen, wenn sich die Zielfunktion ein wenig von einer Disjunktion unterscheidet.

(b) Soll eine monotone *Threshold-Funktion* gelernt werden, dann gibt es keine Unterscheidung in

relevant und irrelevant.

- Behandle positive und negative Beispiele gleichrangig.

Winnow für monotone Threshold-Funktionen

Die unbekannte Threshold-Funktion $f : \text{Def} \to \{-1, 1\}$ ist zu lernen.

1. Für $w^{(0)} = (\frac{1}{n}, \ldots, \frac{1}{n})$ verwende die Hypothese

$$h(x) = \text{sign}(\langle w^{(0)}, x \rangle).$$

   η ist eine positive reelle Zahl und $t := 0$.
2. Wiederhole, solange es ein Gegenbeispiel $x^{(t)}$ gibt:
   (a) Setze $Z_t = \sum_{i=1}^{n} w_i^{(t)} \cdot exp^{\eta \cdot f(x^{(t)}) \cdot x_i^{(t)}}$ und für $i = 1, \ldots n$

$$w_i^{(t+1)} := \frac{w_i^{(t)} \cdot exp^{\eta \cdot f(x^{(t)}) \cdot x_i^{(t)}}}{Z_t}.$$

   /* Gegenbeispiele werden „im Exponenten addiert bzw subtrahiert“. */
   /* $w^{(t+1)}$ ist eine Wahrscheinlichkeitsverteilung. */
   (b) $h(x) = \text{sign}(\langle w^{(t+1)}, x \rangle)$ ist die neue Hypothese. Setze $t := t + 1$.

Winnow und Weighted-Majority

Wenn Def $= \{-1, 1\}^n$:

1. Die Hypothesen von Winnow entsprechen einem (gewichteten) Mehrheitsentscheid.
2. Für eine monotone Disjunktion multipliziert Winnow das Gewicht
   - eines „**inkorrekten**“ Bits x_i, d.h. $f(x) \cdot x_i < 0$ mit $\exp^{-\eta}$ und
   - eines „**korrekten**“ Bits x_i, d.h. $f(x) \cdot x_i > 0$ mit $\exp^{\eta}$.
   - Alternativ: Bestrafe alle inkorrekten Bits mit $\exp^{-2\eta}$.

Wenn Bitpositionen Experten entsprechen ist

$$\textbf{Winnow} = \textbf{Weighted-Majority}$$

für den Beispielraum Def $= \{-1, 1\}^n$.

Die Analyse (1/4)

Einige Vorbereitungen:

1. Winnow benutzt die **1-Norm**

$$\|x\|_1 = \sum_{i=1}^{n} |x_i|.$$

   Wenn $x \geqslant 0$, dann ist $\frac{x}{\|x\|_1}$ eine Wahrscheinlichkeitsverteilung.

2. Fordere für alle Beispiele x: $\|x\|_\infty \leqslant R$ mit der **Maximum-Norm**

$$\|x\|_\infty := \max_{i=1}^{n} |x_i|.$$

3. Wähle einen Gewichtsvektor $v \in \mathbb{R}^n_{\geqslant 0}$, so dass der „**$L_1$-Margin**“

$$\min_t \frac{f(x^{(t)}) \cdot \langle v, x^{(t)} \rangle}{\|v\|_1} =: \rho$$

   auf den Beispielen $x^{(t)}$ möglichst groß wird.

1. Das **Potential** zum Zeitpunkt t wird definiert durch

$$\Phi_t := \sum_{i=1}^{N} \frac{v_i}{\|v\|_1} \ln \frac{v_i/\|v\|_1}{w_i^{(t)}}.$$

   - Φ_t ist die *Kullback-Leibler Divergenz* und misst den Abstand zwischen den Verteilungen $v/\|v\|_1$ und $w^{(t)} \Longrightarrow \Phi_t \geqslant 0$.
   - Φ_t fällt, wenn sich $w^{(t)}$ der Verteilung v annähert.

2. Wie stark fällt das Potential, d.h. wie groß ist $\Phi_{t+1} - \Phi_t$?

$$\begin{aligned}
\Phi_{t+1} - \Phi_t &= \sum_{i=1}^{n} \frac{v_i}{\|v\|_1} \ln \frac{w_i^t}{w_i^{t+1}} = \sum_{i=1}^{n} \frac{v_i}{\|v\|_1} \ln \frac{Z_t}{\exp^{\eta \cdot f(x^{(t)}) \cdot x_i^{(t)}}} \\
&= \ln Z_t - \eta \cdot \sum_{i=1}^{n} \frac{v_i}{\|v\|_1} \cdot f(x^{(t)}) \cdot x_i^{(t)} \\
&\leqslant \ln \left(\sum_{i=1}^{n} w_i^{(t)} \cdot \exp^{\eta \cdot f(x^{(t)}) \cdot x_i^{(t)}} \right) - \eta \cdot \rho
\end{aligned}$$

$$\begin{aligned}
\Phi_{t+1} - \Phi_t &\leqslant \ln\left(\sum_{i=1}^{n} w_i^{(t)} \cdot \exp^{\eta \cdot f(x^{(t)}) \cdot x_i^{(t)}}\right) - \eta \cdot \rho \\
&= \ln \mathbb{E}_{w^{(t)}}\left[\exp^{\eta \cdot f(x^{(t)}) \cdot x^{(t)}}\right] - \eta \cdot \rho \\
&\leqslant \ln\left[\exp^{\eta^2 \cdot (2R)^2/8}\right] - \eta \cdot \rho = \eta^2 R^2/2 - \eta \cdot \rho.
\end{aligned}$$

3. Die letzte Ungleichung folgt aus Hoeffding's Lemma:
   Für $t \in \mathbb{R}$ ist $\mathbb{E}[\exp^{tX}] \leqslant \exp^{\frac{t^2(b-a)^2}{8}}$, falls $\mathbb{E}[X] \leqslant 0$ und $a \leqslant X \leqslant b$.
4. Als Konsequenz

$$\Phi_{T-1} - \Phi_0 = \sum_{t=0}^{T-2} \Phi_{t+1} - \Phi_t \leqslant (T-1) \cdot (\eta^2 R^2/2 - \eta \cdot \rho).$$

5. Die Kullback-Leibler Divergenz ist nicht negativ $\Longrightarrow \Phi_{T-1} \geqslant 0$.
6. Wie groß ist das anfängliche Potential?

$$\Phi_0 = \sum_{i=1}^{n} \frac{v_i}{\|v\|_1} \cdot \ln \frac{v_1/\|v\|_1}{1/n} = \ln n + \sum_{i=1}^{n} \frac{v_i}{\|v\|_1} \cdot \ln \frac{v_i}{\|v\|_1} \leqslant \ln n.$$

7. Also folgt

$$-\ln n \leqslant -\Phi_0 \leqslant \Phi_{T-1} - \Phi_0 \leqslant (T-1) \cdot (\eta^2 R^2/2 - \eta \cdot \rho).$$

8. Wähle $\eta := \rho/R^2 \Longrightarrow$

$$T - 1 \leqslant \frac{2R^2}{\rho^2} \cdot \ln n.$$

Winnow: Das Ergebnis (1/2)

Die Threshold-Funktion f sei zu lernen.

(a) $x^{(0)}, \ldots, x^{(T-1)} \in \mathbb{R}^n$ seien Gegenbeispielen für Winnow.

(b) Es gelte $\|x^{(t)}\|_\infty \leqslant R$ für $0 \leqslant t < T$.

(c) $v \in \mathbb{R}^n_{\geqslant 0}$ sei ein beliebiger Vektor mit

$$0 < \rho \leqslant \frac{f(x^{(t)}) \cdot \langle v, x^{(t)} \rangle}{\|v\|_1}.$$

$\Longrightarrow$ Winnow benötigt höchstens

$$T \leqslant 1 + \frac{2R^2}{\rho^2} \cdot \ln n$$

Gegenbeispiele.

Winnow: Das Ergebnis (2/2)

Für die Funktion $f : \text{Def} \to \{-1, 1\}$ definiere

$$\text{Margin}_{f,1}(\text{Def}) := \sup_{v \in \mathbb{R}^n} \inf_{x \in \text{Def}} \frac{f(x) \cdot \langle v, x \rangle}{\|v\|_1}$$

als den **L_1-Margin** (von f auf Def).

Es gelte $\|x\|_\infty \leqslant R$ für alle Vektoren $x \in \text{Def} \Longrightarrow$
Winnow lernt eine mit f äquivalente Klassifizierung nach höchstens

$$2 \cdot \left(\frac{R}{\text{Margin}_{f,1}(\text{Def})} \right)^2 \cdot \ln n$$

Gegenbeispielen.

Winnow: Eine Anwendung

Beispiele werden aus der Menge $\text{Def} := \{0,1\}^n \times \{1\}$ gewählt.

Die monotone Disjunktion $\alpha = x_{i_1} \vee \cdots \vee x_{i_k}$ sei zu lernen.

1. Für alle $x \in \{0,1\}^n$ ist

$$\alpha(x) \text{ ist wahr } \iff \langle v, x \rangle + v_{n+1} \geqslant 0$$

   mit $v_{n+1} = -\frac{1}{2}$, ($v_j = 1$ gdw. $j \in \{i_1, \ldots, i_k\}$) und $v_j = 0$ sonst.
2. Für $x \in \{0,1\}^n$ ist $\|x\|_\infty \leqslant 1$.
3. $\|v\|_1 = k + 1/2$ und $\langle v, x \rangle \geqslant 1/2$ gilt für alle 0-1 Vektoren $\Longrightarrow$
   - $\rho = \frac{1}{2} \cdot \frac{1}{(k+1/2)} = \frac{1}{2k+1}$ und $R = 1$.

$T \leqslant 2(2k+1)^2 \cdot \log_2 n$ Gegenbeispiele genügen für das Lernen monotoner Disjunktionen mit k Literalen.

Die Winnows

Eine monotone Disjunktionen mit k Literalen ist zu lernen.

(a) Für den „drakonischen“ Winnow genügen

$$T \leqslant 2 + 2k \cdot \log_2 n$$

Gegenbeispiele,

(b) während der „fehlertolerante“ Winnow bis zu

$$T \leqslant 2(2k+1)^2 \cdot \log_2 n$$

Gegenbeispiele benötigt.

$\Longrightarrow$ Fehlertoleranz und größere Ausdrucksstärke haben ihren Preis.

Der Perzeptron-Algorithmus

Der Perzeptron-Algorithmus

Lerne die unbekannte Threshold-Funktion $f : \text{Def} \to \{-1, 1\}$ mit

$$f(x) = \text{sign}(\langle v, x \rangle).$$

1. $h(x) = \text{sign}(\langle w^{(0)}, x \rangle)$ mit $w^{(0)} = 0$ ist die Anfangshypothese. Setze $t = 0$.
2. Wiederhole, solange es ein Gegenbeispiel $x^{(t)}$ gibt:
   (a) Setze
   $$w^{(t+1)} = w^{(t)} + f(x^{(t)}) \cdot x^{(t)}.$$
   /* Gegenbeispiele werden addiert bzw subtrahiert. */
   (b) $h(x) = \text{sign}(\langle w^{(t+1)}, x \rangle)$ ist die aktuelle Hypothese. Setze $t = t + 1$.

- Winnow ist ein *multiplikatives*,
- Perzeptron ein *additives* Lernverfahren.

Das Resultat

Hält der Perzeptron-Algorithmus immer? **Nein!**

(a) Für irgendein $v \in \mathbb{R}^n$ mit $\|v\| = 1$ gelte

$$0 < \rho \leqslant f(x) \cdot \langle v, x \rangle$$

für alle $x \in \text{Def}$.

(b) $x^{(0)}, \ldots, x^{(T-1)} \in \mathbb{R}^n$ seien Gegenbeispiele mit $\|x^{(t)}\| \leqslant R$.

$\Longrightarrow$ der Perzeptron-Algorithmus benötigt höchstens

$$T \leqslant \frac{R^2}{\rho^2}$$

Gegenbeispiele.

Analyse: Vollständige Trennbarkeit

Die Analyse (1/2)

Aus $T \cdot \rho \leqslant \sqrt{T \cdot R^2}$ folgt $T^2 \cdot \rho^2 \leqslant T \cdot R^2$ und damit die Behauptung.

$$
\begin{aligned}
T \cdot \rho &\leqslant \langle\, v, \sum_{t=0}^{T-1} f(x^{(t)}) \cdot x^{(t)} \,\rangle \\
&\leqslant \| \sum_{t=0}^{T-1} f(x^{(t)}) \cdot x^{(t)} \| && \|v\| = 1, \langle a, b \rangle \leqslant \|a\| \cdot \|b\| \\
&= \| \sum_{t=0}^{T-1} (w^{(t+1)} - w^{(t)}) \| = \|w^{(T)}\| && \|w^{(0)}\| = 0 \\
&= \sqrt{\sum_{t=0}^{T-1} \left(\|w^{(t+1)}\|^2 - \|w^{(t)}\|^2 \right)} && \|w^{(0)}\| = 0
\end{aligned}
$$

Die Analyse (2/2)

$$
\begin{aligned}
T \cdot \rho &\leqslant \sqrt{\sum_{t=0}^{T-1} \left(\|w^{(t+1)}\|^2 - \|w^{(t)}\|^2 \right)} \\
&= \sqrt{\sum_{t=0}^{T-1} \left(\|w^{(t)} + f(x^{(t)}) \cdot x^{(t)}\|^2 - \|w^{(t)}\|^2 \right)} \\
&= \sqrt{\sum_{t=0}^{T-1} \left(\underbrace{2 f(x^{(t)}) \cdot \langle x^{(t)}, w^{(t)} \rangle}_{\leqslant 0} + \|x^{(t)}\|^2 \right)} \\
&\leqslant \sqrt{\sum_{t=0}^{T-1} \|x^{(t)}\|^2} = \sqrt{T \cdot R^2} \quad \checkmark
\end{aligned}
$$

Perzeptron und Margin

Sei $f : \text{Def} \to \{-1, 1\}$ eine unbekannte Threshold-Funktion.

Es gelte $\|x\| \leqslant R$ für alle Vektoren $x \in \text{Def} \Longrightarrow$
Perzeptron lernt eine äquivalente Klassifizierung nach höchstens

$$\left(\frac{R}{\text{Margin}_f(\text{Def})}\right)^2$$

Gegenbeispielen.

Die Anzahl benötigter Gegenbeispiele hängt **nicht** von der Dimension, sondern nur vom Margin und der maximalen Beispielnorm ab.

Analyse: Partielle Trennbarkeit

Partielle Trennbarkeit: Die Situation

Positive und negative Beispiele sind in vielen Fällen nicht linear trennbar!

- Die Funktion $f : \mathrm{Def} \to \{-1, 1\}$ ist keine Threshold-Funktion: Versuche f mit Threshold-Funktionen approximativ zu bestimmen.
- Holt Perzeptron das Mögliche heraus?
  - Vergleiche Perzeptron mit irgendeiner Threshold-Funktion $\mathrm{sign}(\langle v, x \rangle)$ für eine Folge $x^{(0)}, \ldots, x^{(T-1)}$ von **Beispielen**.
    - Die Folge besteht nicht nur aus Gegenbeispielen.
    - $\mathrm{sign}(\langle v, x \rangle)$ ist mgl. optimal auf die Folge eingestellt.
- Wieviele **Aktualisierungen** benötigt Perzeptron?

Partielle Trennbarkeit: Das Resultat

Die Funktion $f : \text{Def} \to \{-1, 1\}$ ist gegeben.

(a) $x^{(0)}, \ldots, x^{(T-1)} \in \mathbb{R}^n$ seien Beispiele mit $\|x^{(t)}\| \leqslant R$.

(b) Sei $v \in \mathbb{R}^n$ **irgendein** Vektor mit $\|v\| = 1$ und $\rho \in \mathbb{R}_{\geqslant 0}$ sei beliebig. Setze

$$d_t := \max\{\, 0, \rho - f(x_t) \cdot \langle v, x^{(t)} \rangle \,\} \quad \text{sowie} \quad d := (\, d_t \,|\, 0 \leqslant t \leqslant T-1 \,).$$

$\Longrightarrow$ der Perzeptron-Algorithmus führt höchstens

$$\left(\frac{R + \|d\|}{\rho} \right)^2$$

Aktualisierungen für die Beispielfolge durch.

Ist das gut?

Perzeptron führt höchstens $\left(\frac{R+\|d\|}{\rho}\right)^2$ Aktualiserungen durch, wobei

$$d_t = \max\{\,0, \rho - f(x^{(t)}) \cdot \langle v, x^{(t)} \rangle\,\} \text{ sowie } d = (\,d_t \mid 0 \leqslant t \leqslant T-1\,).$$

Wenn $(\|d\|/\rho)^2 = O(\text{Anzahl Klassifikationsfehler von sign}(\langle v, x\rangle))$,

dann $\left(\frac{R+\|d\|}{\rho}\right)^2 \approx \frac{R^2}{\rho^2} + \frac{\|d\|^2}{\rho^2} \approx \frac{R^2}{\rho^2} +$ Klassifikationsfehler von v.

Also erreicht Perzeptron die Leistung von sign($\langle v, x\rangle$) nach einer „Aufwärmphase“ der Länge $\frac{R^2}{\rho^2}$.

Andererseits

Die Bestimmung von Hyperebenen, die eine

(fast-)minimale Anzahl von Beispielen

falsch klassifizieren, führt auf ein NP-hartes Problem.

- Keine Garantie für eine gute Approximation im Allgemeinen :-((
- aber gute Approximation, wenn „fast überall“ ein großer Margin erreichbar ist :-))

Die Analyse (1/3)

Stelle lineare Trennbarkeit her.

Wir transformieren die Beispiele und setzen

$$T(x^{(t)}) := \left(x_1^{(t)}, \ldots, x_n^{(t)}, 0, \ldots 0, \ldots, 0, \underbrace{\Delta}_{\text{Komponente } n+t+1}, 0, \ldots, 0 \right).$$

Der Parameter Δ wird später gesetzt.

Behauptung 1: Perzeptron klassifiziert $T(x^{(t)})$ genauso wie $x^{(t)}$.

1. Perzeptron summiert bzw subtrahiert Gegenbeispiele und dies gilt für $T(x^{(t)})$ wie auch für $x^{(t)}$.
2. Ist $w^{(t)}$ bzw. $\overline{w}^{(t)}$ der Gewichtsvektor vor bzw. nach Transformation der Beispiele, dann folgt

$$\langle w^{(t)}, x^{(t)} \rangle = \langle \overline{w}^{(t)}, T(x^{(t)}) \rangle$$

Wir übersetzen auch v in $\overline{v}/\|\overline{v}\|$ mit

$$\overline{v} := \left(v_1, \ldots, v_n, \frac{f(x^{(0)}) \cdot d_0}{\Delta}, \ldots, \frac{f(x^{(T-1)}) \cdot d_{T-1}}{\Delta} \right).$$

Es ist $\|v\| = 1$ und deshalb folgt $\|\overline{v}\| = \sqrt{1 + \frac{\|d\|^2}{\Delta^2}}$.

$$\begin{aligned}
f(x^{(t)}) \cdot \frac{\langle \overline{v}, T(x^{(t)}) \rangle}{\|\overline{v}\|} &= f(x^{(t)}) \cdot \left(\frac{\langle v, x^{(t)} \rangle}{\|\overline{v}\|} + \Delta \cdot \frac{f(x^{(t)}) \cdot d_t}{\|\overline{v}\| \cdot \Delta} \right) \\
&= f(x^{(t)}) \cdot \frac{\langle v, x^{(t)} \rangle}{\|\overline{v}\|} + \frac{d_t}{\|\overline{v}\|} \\
&\geqslant f(x^{(t)}) \cdot \frac{\langle v, x^{(t)} \rangle}{\|\overline{v}\|} + \frac{\rho - f(x^{(t)}) \cdot \langle v, x^{(t)} \rangle}{\|\overline{v}\|} \\
&= \frac{\rho}{\|\overline{v}\|}
\end{aligned}$$

Die Analyse (3/3)

Die Beispiele $T(x^{(0)}), \ldots, T(x^{(T-1)})$ sind linear trennbar mit Margin $\rho/\|\overline{v}\|$.

1. Es ist $\|T(x^{(t)})\|^2 \leqslant R^2 + \Delta^2$.
2. Die Anzahl der Aktualisierungen ist beschränkt durch

$$\frac{(R^2+\Delta^2)\cdot\|\overline{v}\|^2}{\rho^2} = \frac{(R^2+\Delta^2)\cdot(1+\frac{\|d\|^2}{\Delta^2})}{\rho^2}.$$

3. Wähle $\Delta^2 := R\cdot\|d\|$ ✓

Winnow versus Perzeptron

VC-Dimension und Margin

$$\text{LINEAR}_\rho(\text{Def}) := \{\, f(*) = \text{sign}(\langle w, * \rangle) \;:\; \text{Margin}_f(\text{Def}) \geqslant \rho,\ w \in \mathbb{R}^n \,\}$$

ist die Konzeptklasse der Threshold-Funktionen $f : \text{Def} \to \{-1, 1\}$ mit Margin $\geqslant \rho$ für die Teilmenge $\text{Def} \subseteq \mathbb{R}^n$. Dann folgt

Es gelte $\|x\|_2 \leqslant R$ für alle $x \in \text{Def} \implies$

$$\text{VC}(\text{LINEAR}_\rho(\text{Def})) \leqslant \min\{\, \frac{R^2}{\rho^2},\ n+1 \,\}.$$

(Beweis: Satz 4.2 im Textbuch „Foundations of Machine Learning“)

Wenn $\frac{R^2}{\rho^2} = o(n)$, dann hängt die VC-Dimension

- von der 1-Norm der Beispiele und dem Margin ρ ab,
- nicht aber von der Dimension n.

Wann ist der Perzeptron-Algorithmus gut?

(a) Perzeptron ist fast-optimal, wenn $\mathrm{VC}(\mathrm{LINEAR}_\rho(\mathrm{Def})) \approx \frac{R^2}{\rho^2}$.

- In diesem Fall ist R^2 wie auch ρ^2 höchstens polynomiell in n.
- In allen anderen Fällen: Ersetze Perzeptron durch lineare Programmierung.

(b) Ein Vergleich mit dem Halbierungs-Algorithmus für den Beispielraum $X = \{0,1\}^n$:

- Man kann zeigen, dass es $2^{\Theta(n^2)}$ viele verschiedene Threshold-Funktionen mit *ganzzahligen* Koeffizienten gibt.
  - ★ Die überwältigende Mehrheit dieser Threshold-Funktionen besitzt somit Koeffizienten, die im Absolutbetrag exponentiell groß sind.
  - ★ Der Perzeptron-Algorithmus addiert bzw. subtrahiert Beispiele $\Longrightarrow$ die meisten Threshold-Funktionen erfordern $2^{\Omega(n)}$ Gegenbeispiele!
- Der Halbierungs-Algorithmus lernt jede der $2^{\Theta(n^2)}$ Threshold-Funktionen „locker“ nach $O(n^2)$ Gegenbeispielen.
  - ★ Die Auswertung seiner Hypothesen ist allerdings **sehr aufwändig**.

Winnow versus Perzeptron

$\|x\|_\infty \leqslant \|x\|_2 \leqslant \sqrt{n} \cdot \|x\|_\infty$ und $\|w\|_2 \leqslant \|w\|_1 \leqslant \sqrt{n} \cdot \|w\|_2 \Longrightarrow$
Gegenbeispielzahlen unterscheiden sich maximal um Faktor $n \cdot \log_2 n$.

(a) Winnow reagiert gutmütiger auf Beispiele mit vielen ungefähr gleichgroßen Komponenten.

(b) Wer mag die Zielfunktion $\text{sign}(\langle w, x \rangle)$?

- Winnow mag es, wenn w nur wenige große Komponenten besitzt. Z.B. für Disjunktionen mit k (aus n möglichen Variablen)
  - genügen $O(k \cdot \log_2 n)$ bzw $O(k^2 \cdot \log_2 n)$ Gegenbeispiele für Winnow und
  - sind bis zu $\Omega(k \cdot n)$ Gegenbeispiele für Perzeptron erforderlich.
- Perzeptron mag Gewichtsvektoren mit vielen ungefähr gleichgroßen Komponenten.

Zusammenfassung

Online-Algorithmen: Zusammenfassung

(a) Gegenbeispielzahl:

$$\begin{aligned} \mathrm{VC}(\mathcal{C}) \leqslant \mathrm{Tiefe}(\mathcal{C}) &= \mathrm{Gegenbeispiel}(\mathcal{C}) \\ &\leqslant \mathrm{Gegenbeispiel}_{\mathrm{Halbierung}}(\mathcal{C}) \leqslant \log_2 |\mathcal{C}| \end{aligned}$$

(b) **Weighted-Majority**: Für die Gesamtbenotung K ist

$$K \leqslant (1+\varepsilon) \cdot K_{\mathrm{opt}} + \frac{\ln n}{\varepsilon}$$

- Anwendungen: Winnow, AdaBoost, Universal Portfolio Algorithmus.

(c) **Winnow** und **Perzeptron**: $\leqslant \frac{R^2}{\mathrm{Margin}_f^2(\mathrm{Def})}$ Gegenbeispiele (bis auf logarithmische Faktoren)

- Winnow passt sich der Komplexität der Zielfunktion an.
- Beide, Winnow und Perzeptron sind fehlertolerant.

Pleiten, Pech, Pannen und geniale Ideen

1943 McCulloch und Pitts führen Threshold-Funktionen ein, um Neuronen zu modellieren.

1958 Rosenblatt stellt den Perzeptron-Algorithmus vor.

1969 Minsky und Pappert dämpfen die Euphorie: Das XOR ist nicht durch eine Threshold-Funktion darstellbar.

1960 In der Steuerungstheorie wird die Methode „Backpropagation“ vorgeschlagen.

1986 Rumelhart, Hinton und Williams wenden Backpropagation auf neuronale Netzwerke an.

1995 Einfachere Methoden (Support-Vektor Maschinen) laufen den neuronalen Netzwerken den Rang ab.

2009 Deep Learning ist erfolgreich: Verbesserte Hardware, mehr und bessere Beispiele, verbesserte Software.

Und die Zukunft?