Wir möchten einen Lernalgorithmus entwerfen

Welche Fragen sind zu klären? (1/3)

1. Wie kann **problem-spezifisches Wissen** umgesetzt werden?
   - Welche **Feature-Funktion**

     $$\phi : X \to \mathbb{R}^n$$

     sollten wir einsetzen?
   - Mit welcher **Hypothesenklasse** $\mathcal{H}$ sollten wir arbeiten?
     - ⋆ Z.B. welche Architektur für neuronale Netzwerke?
   - Oder sollten wir besser auf mehrere Klassen $\mathcal{H}_1, \ldots, \mathcal{H}_r$ setzen?
     - ⋆ Z.B. die Größen der Netze variieren?
2. **Beispielkomplexität**:
   - Wie viele Beispiele werden benötigt?
     - ⋆ Im Zweifel: So viele wie möglich, mindestens aber ein Vielfaches der Freiheitsgrade von $\mathcal{H} \Longrightarrow$ **kein Overfitting**.
     - ⋆ Und wenn der Trainingsfehler zu groß ist?
       Training misslungen oder zu wenige Parameter $\Longrightarrow$ **Underfitting**.
   - Wie verschaffen wir uns möglichst viele, gute Beispiele?

Welche Fragen sind zu klären? (2/3)

3. **Lernregel**:
   - Welche Loss-Funktion soll benutzt werden?
     - ⋆ Quadratischer Loss?
     - ⋆ Maximum-Likelihood Schätzer?
   - Soll eine Form der Regularisierung gewählt werden?

$$\text{minimiere}_{w \in \mathbb{R}^n} \left(\text{Loss}_D(w) + \alpha \cdot \langle w, w \rangle \right).$$

   (Bevorzuge „einfache“ Hypothesen.)

4. **Algorithmische Komplexität**:
   - Kann man ERM-Hypothesen effizient berechnen?
     - ⋆ Wahrscheinlich nicht, aber kann man Hypothesen mit kleinem Trainingsfehler berechnen?
   - Kann Lernerfolg mit Ensemble-Methoden verbessert werden?
     - ⋆ Ensemble-Methoden (wie Bagging oder Boosting) werden später beschrieben.

Welche Fragen sind zu klären? (3/3)

5. **Fehlerbestimmung**:
   - Der **wahre Fehler** $\text{Loss}_D(h)$: **Validierung** ✓
   - der **Approximationsfehler**

$$\min_{h \in \mathcal{H}} \text{Loss}_D(h) \; \lightning$$

   - und der **minimale Trainingsfehler**

$$\min_{h \in \mathcal{H}} \text{Loss}^S(h) \; \lightning$$

   für Hypothesenklasse $\mathcal{H}$, Verteilung D, Beispielmenge S.

Validierung

Modellauswahl

? Verschiedene rivalisierende Hypothesen

$$h_1, \ldots, h_r$$

stehen zur Auswahl. Welcher Ansatz ist besser?

! Versuche den wahren Fehler für jeden Ansatz möglichst scharf vorauszusagen.

Validierung: Hold-Out-Set

1. $h_1, \ldots, h_r$ seien binäre Hypothesen über einer Menge X potentieller Beispiele.
2. D sei eine unbekannte Verteilung über X.
3. $\ell_1, \ldots, \ell_r : X \times \{0, 1\} \to [0, 1]$ seien Loss-Funktionen.

Wie groß ist der wahre Fehler $\text{Loss}_{D,i}(h_i)$?

Ziehe eine Menge V von v klassifizierten Beispiele gemäß $D \Longrightarrow$
Mit Wahrscheinlichkeit mindestens $1 - \delta$ gilt

$$\text{für alle } i \leqslant r\text{:} \quad \left|\text{Loss}^V(h_i) - \text{Loss}_{D,i}(h_i)\right| \leqslant \sqrt{\frac{2\ln(\frac{2r}{\delta})}{v}}$$

Der Fehler auf der Validierungsmenge

Mit der Hoeffding-Ungleichung folgt für jedes i $(1 \leqslant i \leqslant r)$

$$\begin{aligned} & \text{prob}\Big[\,\Big|\,\text{Loss}^V(h_i) - \text{Loss}_{D,i}(h_i)\Big| \geqslant \beta\,\Big] \\ = \quad & \text{prob}\Big[\,\Big|\frac{1}{v}\cdot\sum_{j=1}^{v}\ell_i(h_i,x_j,y_j) - \mathbb{E}[\,\ell_i(h_i,x,y)\,]\Big| \geqslant \beta\,\Big] \leqslant 2e^{-v\beta^2/2}. \end{aligned}$$

Die Forderung

$$r\cdot 2e^{-v\beta^2/2} \overset{!}{\leqslant} \delta$$

wird durch $\beta \geqslant \sqrt{\frac{2\ln(\frac{2r}{\delta})}{v}}$ erfüllt. □

Modellauswahl: k-fache Kreuzvalidierung

A sei ein Lernalgorithmus für r Hypothesenklassen $\mathcal{H}_1, \ldots, \mathcal{H}_r$.

1. A fordert A eine Menge S von s klassifizierten Beispielen an und zerlegt S in k disjunkte Teilmengen $S_1, \ldots, S_k$ der Größe s/k.
2. Für jedes $j \in \{1, \ldots, k\}$ bestimmt A eine Hypothese $h_{i,j} \in \mathcal{H}_i$ mit der Beispielmenge $S \setminus S_j$.
   - Schätze den wahren Fehler $\beta_{i,j}$ von $h_{i,j}$ mit Validierungsmenge S_j.
3. A schätzt den wahren in $\mathcal{H}_i$ erreichbaren Fehler als Durchschnitt

$$\beta_i := \frac{1}{k} \cdot \sum_{j=1}^{k} \beta_{i,j}$$

   und wählt die Hypothesenklasse $\mathcal{H}_j$ mit kleinstem β_j.
4. A bestimmt die finale Hypothese $h \in \mathcal{H}_j$ mit Trainingsmenge S.

Experimentelle Erfahrung: **gut**, mathematische Begründung: ?

Nicht-binäre Lernprobleme

Objekterkennung

Für Mengen X und Y ist eine Funktion

$$h : X \to Y$$

(mit $|Y| > 2$) zu lernen.

(a) In der Ziffererkennung ist $Y = \{0, 1, \ldots, 9\}$.
Für eine Pixeldarstellung x benutze z.B. die Loss-Funktion

$$\ell(h, x, y) = \begin{cases} 0 & h(x) = y, \\ 1 & h(x) \neq y \end{cases}$$

wenn y die gewünschte Interpretation der Pixeldarstellung x ist.

(b) In der einfachen linearen Regression bewerte $h(x) := ax + b$ mit

$$\ell(h, x, y) = (a \cdot x + b - y)^2,$$

wenn y der Zielwert ist.

One-versus-All

Sei $\mathcal{H}$ eine Hypothesenklasse über X und $Y = \{1, \ldots k\}$.

1. $S = \{(x_1, y_1), \ldots, (x_s, y_s)\}$ ist die Menge klassifizierter Beispiele.
2. Für jedes i $(1 \leqslant i \leqslant k)$:
   (a) Bestimme die Menge S_i:
      - ★ Wenn $y_j = i$, dann ist x_j ein positives Beispiel
      - ★ und ansonsten ein negatives Beispiel.

   (b) Lerne eine binäre Hypothese h_i mit Hilfe von S_i.
3. Gib die Hypothese $h : X \to \{1, \ldots, k\}$ aus, wobei

$$h(x) = i \iff \underbrace{i \text{ ist minimal mit}} \quad h_i(x) = 1.$$

Besser: Konfidenz-Gewichtung

Willkürliche Wahl bei mehreren möglichen Klassen!

All-Pairs: Jeder gegen jeden

1. $S = \{(x_1, y_1), \ldots, (x_s, y_s)\}$ ist eine Menge klassifizierter Beispiele.
2. Für alle i, j $(1 \leqslant i < j \leqslant k)$:
   (a) Bestimme die Menge $S_{i,j}$ der Beispiele in S, die entweder mit i oder mit j klassifiziert sind.
   (b) Ein binäre Hypothese $h_{i,j}$ wird mit Hilfe der Beispielmenge $S_{i,j}$ gelernt.
3. Gib die Hypothese $h : X \to \{1, \ldots, k\}$ aus, wobei

$$h(x) = i \Leftrightarrow \; i \text{ hat für } x \text{ die meisten Siege zu verzeichnen.}$$

Was ist im nicht-binären Lernen anders?

Verschiedene ERM-Lernalgorithmen haben möglicherweise völlig unterschiedliche Beispielzahlen.

1. Wie üblich sei **X** die Menge der Beispiele. $\mathcal{P}_{\text{endlich}}(X)$ bestehe aus allen Teilmengen $Z \subseteq X$, sodass Z oder $X \setminus Z$ endlich ist.
2. $\mathbf{Y} = \mathcal{P}_{\text{endlich}}(X) \cup \{*\}$ ist die Menge aller Klassifizierungen.
3. Hypothesen sind die Funktionen $f_Z : X \to Y$ für $Z \in \mathcal{P}_{\text{endlich}}(X)$ mit

$$f_Z(x) = \begin{cases} Z & x \in Z, \\ * & x \notin Z. \end{cases}$$

4. Die Verteilung D erzeuge nur Beispiele, die mit einer Hypothese f_Z konsistent sind. Benutze den 0-1 Loss.

Um welche Menge Z geht es? Was ist die richtige Hypothese f_Z?

Zwei ERM-Algorithmen

Beispiel $(x, *)$ besagt, dass x nicht zur gesuchten Menge gehört,
Beispiel (x, Z) besagt, dass Z die gesuchte Menge ist.
$\Longrightarrow$ Nur der Fall $S = \{(x_1, *), \ldots, (x_s, *)\}$ ist interessant.

1. Algorithmus A_1 gibt die Hypothese $f_\emptyset$ aus und setzt damit auf die **kleinste konsistente** Menge.
2. Algorithmus A_2 setzt auf die **größte konsistente** Menge und gibt die Ausgabe

$$f_{X \setminus \{x_1, \ldots, x_s\}}.$$

A_1 wie auch A_2 bestimmen ERM-Hypothesen, denn ihre Hypothesen sind konsistent mit S. Wie viele Beispiele benötigen A_1 und A_2?

Wie viele Beispiele benötigt A_1?

Für A_1 genügen $\frac{1}{\varepsilon} \ln \frac{1}{\delta}$ Beispiele, um mit Wahrscheinlichkeit $\geqslant 1 - \delta$ einen Fehler $\leqslant \varepsilon$ zu erreichen.

Warum? Angenommen, die richtige Antwort ist f_Z.

1. Der Fehler der Antwort $f_\emptyset$ ist nur dann zu groß, wenn

$$\text{prob}_D[\,\{(x, Z) \,:\, x \in Z\}\,] \geqslant \varepsilon.$$

2. Die Wahrscheinlichkeit, keine Elemente aus Z in s Versuchen zu ziehen, ist aber höchstens

$$\underbrace{(1-\varepsilon)^s \leqslant e^{-\varepsilon \cdot s}}_{\text{Mutter aller Ungleichungen}} \overset{!}{\leqslant} \delta.$$

3. Setze $s = \frac{1}{\varepsilon} \ln \frac{1}{\delta}$. □

Wie viele Beispiele benötigt A_2?

Um mit Wahrsch $\geqslant 1 - \delta$ einen Fehler $\leqslant \varepsilon$ zu erreichen, benötigt A_2

$$\Omega(\frac{1}{\varepsilon} \cdot \left(|X| + \ln(\frac{1}{\delta}) \right))$$

klassifizierte Beispiele.

1. O.B.d.A. ist $S = \{(x_1, *), \ldots, (x_s, *)\}$.
2. Sei $Z = \emptyset$ die zu lernende Menge.
3. Dann ist $\mu = \text{prob}_D[\, x \notin \{x_1, \ldots, x_s\}\,]$ der Fehler von A_2.
   - Wähle D als Gleichverteilung.

Ist das ERM-Prinzip in Frage gestellt?

Es gibt Mengen X, Y, eine Hypothesenklasse $\mathcal{H}$ über X und Y sowie **ERM-Algorithmen** A_1 und A_2 so dass

(a) A_1 jedes Konzept $h \in \mathcal{H}$ mit Wahrscheinlichkeit mindestens $1 - \delta$ und Fehler höchstens ε für

$$\frac{1}{\varepsilon} \ln \frac{1}{\delta}$$

Beispiele lernt.

(b) Wenn $|X| = \infty$: Es gibt aber ein Konzept, das von A_2 mit Fehler $\varepsilon < \frac{1}{2}$ nur bei unbeschränkt vielen Beispielen lernbar ist.

Für binäre ERM-Hypothesen ist asymptotisch die fast gleiche Zahl von Beispielen hinreichend und notwendig.

Nicht-binäres Lernen: Was tun?

$\mathcal{H}$ ist **symmetrisch**, wenn für alle Hypothesen $h \in \mathcal{H}$ und für alle Permutationen $\pi : Y \to Y$ auch $\pi \circ h$ zu $\mathcal{H}$ gehört.

(a) Viele Hypothesenklassen von praktischem Interesse sind symmetrisch.

(b) Es gilt: Für symmetrisches $\mathcal{H}$ mit **Natarajan- Komplexität** $N(\mathcal{H})$ nimmt jede Hypothese h nur $\mathcal{O}(N(\mathcal{H}))$ Werte an.
- $N(\mathcal{H})$ ist eine Variante der VC-Dimension.
- Die VC-Dimension ist nur für binäre Lernprobleme geeignet.

(c) Ein **guter** ERM-Lerner A wählt a priori eine beliebige Teilmenge $Y' \subseteq Y$ mit $|Y'| = \mathcal{O}(N(\mathcal{H}))$.
- und bestimmt für Beispielmenge $S = \{(x_1, y_1), \ldots, (x_s, y_s)\}$ eine konsistente Hypothese $h \in \mathcal{H}$ mit Wertebereich $Y' \cup \{y_1, \ldots, y_s\}$.
- Wegen Symmetrie ist Existenz von h gesichert.

(d) Algo A_1 hat $Y' = \{*\}$ gewählt, während A_2 sich nicht einschränkt.

Ähnliche Ergebnisse werden auch für nicht-symmetrisches $\mathcal{H}$ erwartet.

Der Nearest-Neighbor Algorithmus

k-Nearest-Neighbor

Die **Eingabe**:

- Eine Trainingsmenge $S = \{ (x_1, y_1), \ldots, (x_s, y_s) \}$ mit Vektoren $x_1, \ldots, x_s \in \mathbb{R}^N$.
- Eine Distanzfunktion $d : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}$ mit den Eigenschaften
  - $d(u, v) = 0 \iff u = v$,
  - $d(u, v) = d(v, u)$ und
  - $d(u, w) \leqslant d(u, v) + d(v, w)$.

Sei x ein nicht-klassifiziertes Beispiel.

1. Bestimme die k zu x nächstliegenden Beispiele $x_{i_1}, \ldots, x_{i_k} \in S$, wobei die Distanz gemäß d zu messen ist.
2. Klassifiziere x mit dem häufigsten Wert y in $\{ y_{i_1}, \ldots, y_{i_k} \}$.
   - Wenn die Zielfunktion f „hinreichend glatt“ ist: Setze

$$y := \frac{y_1 + \cdots + y_k}{k}.$$

Eine statistische Analyse (1/2)

1. Es ist $X = [0,1]^d$ und $Y = \{0,1\}$:
   Binäre Klassifikation auf dem d-dimensionalen Einheitswürfel.
2. d ist die Euklidsche Distanz $\|x - y\| := \sqrt{\sum_{i=1}^{d}(x_i - y_i)^2}$.
3. $\mathcal{H}$ besteht aus allen Funktion $h : X \to \{0,1\}$.
4. Wir arbeiten mit dem 0-1 Loss $\ell(h,x,y) = \begin{cases} 1 & h(x) \neq y \\ 0 & \text{sonst.} \end{cases}$
5. Für die Verteilung D über $X \times Y$
   - $\text{prob}_D[\, y = 0 \mid x \,]$ ist der Grenzwert der Wahrscheinlichkeit der Klassifikation 0 für beliebig kleine Bälle mit Zentrum x.
   - D sei eine „C-Lipschitz-Verteilung“ auf $X \times Y$, d.h. es gelte

$$\left| \text{prob}_D[\, y = 0 \mid x \,] - \text{prob}_D[\, y' = 0 \mid x' \,] \right| \leqslant C \cdot \|x - x'\|$$

   für alle $x, x' \in X$ mit einer Konstanten $C \Longrightarrow$ naheliegende Beispiele tragen wahrscheinlich die gleiche Klassifikation.

Eine statistische Analyse (2/2)

Die Bayes-Klassifikation $h_{\text{Bayes}} : [0,1]^d \to \{0,1\}$ ist die Hypothese mit dem kleinsten 0-1 Loss für Verteilung D, d.h.

$$\text{Loss}_D(h_{\text{Bayes}}) = \min_h \text{Loss}_D(h).$$

Also setze

$$h_{\text{Bayes}}(x) = 0 :\Longleftrightarrow \Big(\text{prob}_D[\, y = 0 \,|\, x \,] \geqslant \text{prob}_D[\, y = 1 \,|\, x \,]\Big).$$

Sei h_S die Hypothese von 1-Nearest-Neighbor für Beispielmenge S.

D sei C-Lipschitz, dann – siehe Shalev-Shwartz, Ben-David –

$$\text{Loss}_D[\, h_S \,] \leqslant 2 \cdot \text{Loss}_D(h_{\text{Bayes}}) + 4C\sqrt{d} \cdot s^{-1/(d+1)}.$$

Der Fluch der Dimensionen

$$\text{Loss}_D[h_S] \leqslant 2 \cdot \text{Loss}_D(h_{\text{Bayes}}) + 4C\sqrt{d} \cdot s^{-1/(d+1)}.$$

Der erwartete Loss von 1-Nearest-Neighbor ist höchstens doppelt so groß wie der kleinstmögliche Loss, *wenn* $4C\sqrt{d} \cdot s^{-1/(d+1)} = o(1)$.

$$\Longrightarrow \quad s = d^{\Omega(d)}.$$

Was passiert hier?

1. Beschränke dich auf die Beispielmenge

$$X = \{(\frac{a_1}{C}, \ldots, \frac{a_d}{C}) \,:\, 0 \leqslant a_i \leqslant C\}.$$

2. Beliebige Klassifikationen von X sind erlaubt :-((
3. $|X| = (C+1)^d \Longrightarrow$ Beispielmenge muss exponentiell groß sein.

Nearest-Neighbor: Stärken und Schwächen

(a) Nur erfolgversprechend, wenn
- „ähnliche“ Beispiele sich „ähnlich“ verhalten,
- Bildung von wenigen Clustern zu vermuten ist.

(b) Für großes k
- ist die Bestimmung der k nächstliegenden Beispiele aufwändig,
- große Klassen treten häufig auf.

(c) Das Verfahren neigt zum Overfitting:
- Keinerlei Kompression der Beispiele in eine kurze Hypothese.

Die Hoeffding-Ungleichung

$X_1, \ldots, X_n$ seien **unabhängige** Zufallsvariablen, so dass für alle i

$$a_i \leqslant X_i - \mathbb{E}[X_i] \leqslant b_i$$

mit Wahrscheinlichkeit 1 gelte. Dann folgt für alle $\epsilon > 0$

$$\text{prob}\left[\left|\sum_{i=1}^{n}\left(X_i - \mathbb{E}[X_i]\right)\right| \geqslant \varepsilon \cdot n\right] \leqslant 2\exp^{-\frac{2n^2\varepsilon^2}{\sum_{i=1}^{n}(b_i-a_i)^2}}.$$