Machine Learning 1

Henrik Tscherny

20. Dezember 2021

Inhaltsverzeichnis

1 Supervised learning

Ziel: Finde in einer Funktionsfamilie, z.B. lineare Funktionen, eine Funktion z.B. $f = 5x + 3$ welche eine gewichtete Summe über input-output-Paaren minimiert, dabei sollte die Funktion die gegebenen Daten bestmöglich beschreiben ohne dabei zu komplex zu sein

Beispiel:

- $g : \Theta \to Y^X$ -> Ordnet jeder Messung zu ob gesund oder krank

  $$g_\theta(x_s) = \begin{cases} 0, & \sum_{j=1}^{n} \Theta_j x_j > 0 \\ 1, & sonnst \end{cases}$$, für ein Sample $x_s \in \mathbb{R}^m$

  - $x \in X = \mathbb{R}^n$: Inputs -> Messung mit m Messpunkten
  - $y \in Y = \{0, 1\}$: Labels -> gesund = 1, krank = 0
  - $\theta \in \Theta = \mathbb{R}^n$: Parameter -> z.B. Koeffizienten einer linearen Funktion

Die Qualität einer gelernten Funktion kann u.a. durch folgende Größen beschrieben werden:

- **Accuracy** (Verhältnis der korrekten Voraussagen zur Gesamtanzahl)
  $\frac{L(0,0)+L(1,1)}{|S|}$
- **Error ratio** (Verhältnis der falschen Voraussagen zu Gesamtanzahl)
  $\frac{L(0,1)+L(0,1)}{|S|}$
- **Precision** (Von den positiv getesteten, wie viele sind tatsächlich positiv) False-Positive-Rate
  $\frac{L(1,1)}{L(1,0)+L(1,1)}$
- **Recall/Sensitivity** (Von den tatsächlich positiven, wie viele wurden tatsächlich positiv getestet) False-Negative-Rate
  $\frac{L(1,1)}{L(0,1)+L(1,1)}$

1.1 Formal

Optimieren einer Funktionsfamilie $g : \Theta \to \{0,1\}^X$, damit dies einfacher ist, optimieren einer Relaxation $f : \Theta \to \mathbb{R}^X$. Sei L eine Loss-function $L : \mathbb{R} \times \{0,1\} \to \mathbb{R}_0^+$ welche g im Bezug zu f definiert.
$\forall\theta \in \Theta \forall x \in X : g_\theta(x) \in \underset{\hat{y}\in\{0,1\}}{\operatorname{argmin}} L(f_\theta(x), \hat{y})$
Bsp. Loss-function:

0-1-Loss: $L = \begin{cases} 0, & sample = label \\ 1, & sonnst \end{cases}$, der Loss ist 0% wenn das Label stimmt

Definiere des Weiteren:

- S: Samples
- X: Attributspace
- $x : S \to X$: bildet ein konkretes Sample mit seinen Attributen ab
- $y : S \to \{0,1\}$ gibt einem konkreten Sample ein Label

Das Tupel $T = (S, X, x)$ nennt man **unlabled data**
Das Tupel $T = (S, X, x, y)$ nennt man **labeled data**

Damit die Komplexität der zu lernenden Funktion begrenzt wird, führen wir zusätzlich noch einen **Regularizer**. Komplexität kann in diesem Fall z.B. die Anzahl der Koeffizienten oder die Länge einer Formel bemessen werden. Der Einfluss des Regularizers wird durch einen Parameter λ gesteuert

$R : \Theta \to \mathbb{R}_0^+$ und $\lambda \in \mathbb{R}_0^+$
Das **supervised learning problem** kann dann wie folgt formuliert werden:
$\inf_{\theta \in \Theta} \ \lambda R(\theta) + \frac{1}{|S|} \sum_{s \in S} L(f_\theta(x_s), y_s)$

- der Regularizer wird durch λ gewichtet
- es wird die Summe der individuellen Loss-Werte minimiert
- die Loss-Summe wird über die Anzahl der Samples Normalisiert, das macht man, damit man lediglich die Parameter übertragen muss, wenn man das Model weitergibt
- Da der Regularizer zum Gesamtloss addiert wird, wird versucht diesen Term ebenfalls möglichst klein zu halten

Das **separation problem** ist definiert durch:
$\inf_{\theta \in \Theta} R(\theta)$
$\forall s \in S : f_\theta(x_s) = y_s$

- finden des minimalen Regularizers
- alle Daten sind korrekt gelabeled

Das **bounded separability problem** lautet:
$R(\theta) \leq m$
$\forall s \in S : \ f_\theta(x_s) = y_s$

- finden eines Regularizers welcher die Komplexität für jeden Parameter unter einer Schranke m hält

Das **inference problem** (Anwenden des trainierten Modells) kann nun wie folgt formuliert werden:
$\min_{y' \in \{0,1\}^S} \sum_{s \in S} L(\hat{f}(x_s), y'_s) = \sum_{s \in S} \min_{y' \in \{0,1\}^S} L(\hat{f}(x_s), y'_s)$

2 Disjunktive Normalformen (DNF'S)

Probleme können ebenfalls als logische Gleichungen interpretiert werden, diese Gleichungen können dann in eine DNF (Mit oder verbundene und-Terme) umgeformt werden.

2.1 Formal

- $\Gamma = \{(V_0, V_1) \in 2^V \times 2^V | V_0 \cap V_1 = \emptyset\}$
  Jede Variable kann entweder negiert oder nicht-negiert vorkommen
- $\Theta = 2^\Gamma$
- $\forall x \in \{0,1\}^V \ : \ f_\theta(x) = \bigwedge_{(V_0,V_1)\in\theta} \prod_{v\in V_0}(1 - x_v) \prod_{v\in V_1} x_v$
  Definition einer DNF, veroderte negierte und nicht-negierte Variablen

Beispiel $\{(\emptyset, \{v_1, v_2\}), (\{v_1\}\{v_3\})\} = \theta \in \Theta \ \rightarrow f_\theta(x) = x_{v_1}x_{v_2} \vee (1 - x_{v_1})x_{v_3}$

Des Weiteren können Regularizer für DNF's definiert wenn, um deren Komplexität zu bemessen:

- $R_d(\theta) = \max_{(V_0,V_1)\in\theta}(|V_0| + |V_1|)$
  Tiefe der Formel, e.g. Anzahl der Variablen des längsten und-Terms
- $R_l(\theta) = \sum_{(V_0,V_1)\in\theta}(|V_0| + |V_1|)$
  Länge der Formel, e.g. Gesamtanzahl der Variablen in der Gesamtformel (auch doppelte zählen)

Beispiel $\theta = \{(\emptyset, \{0\}), (\{0\}, \{3\}), (\{0,3\}, \{1,2\}))\}$
$\rightarrow f_\theta(x) = x_0 \vee (1 - x_0)x_3 \vee (1 - x_0)(1 - x_3)x_1x_2 \rightarrow R_l(\theta) = 7, \ R_d(\theta) = 4$

Das **Supervised learning problem of DNF's** kann wie folgt formuliert werden:

$\min_{\theta\in\Theta} R(\theta)$
$\forall s \in S \ : \ f_\theta(x_s) = y_s$

- Der Unterschied ist lediglich, dass ein min statt eines inf benutzt wird
- min: kleinstes Element einer Menge und muss in der Menge selbst liegen
- inf: größte untere Schranke, muss nicht in der Menge selbst liegen, es müssen nur alle Elemente kleiner sein Des Weiteren erhält man das **bounded depth/length DNF Problem** indem man den Regularizer aus dem bounded sparability problem mit R_d / R_l austauscht

Das bounded length/depth DNF problem ist NP-hard
Beweis durch Reduktion des set cover problems auf das bounded length/depth DNF problem (Haussler):
Was ist ein Set-Cover:

- Sei S eine eine Menge
- Sei $\Sigma \subseteq 2^S,\ \emptyset \notin \Sigma$ ein Cover, gdw. $\bigcup_{U \in \Sigma} U = S$
- $m \in \mathbb{N}$
- Die Entscheidung ob ein $\Sigma' \subseteq \Sigma$ existiert, s.d. $|\Sigma'| \leq m$ nennt man das **set cover problem** (S, Σ, m)

Beispiel:
Sei $S = \{1,2,3,4,5,6\}$, dann ist ein mögliches Cover $\Sigma = \{\{1,2,3\},\{4,5,6\}\}$ (m=2)

2.2 Beweis der NP-hardness von set-cover

Wir zeigen das set-cover NP-hard ist indem wir es auf bounded length/depth DNF reduzieren (Haussler)

- Sei (S', Σ, m) eine Instanz von set-cover
- Definiere nun **Haussler data** (S, X, x, y) s.d.:
  - $S = S' \cup \{1\}$, wir fügen S ein spezielles distinktes Element 1 hinzu
  - $X = \{0,1\}^\Sigma$
  - wir definieren uns eine Funktion welche angibt ob ein Element in einer Menge vorkommt wie folgt:
    $$\forall s \in S' \, \forall \sigma \in \Sigma : \ x_s(\sigma) = \begin{cases} 0, & s \in \sigma \\ 1, & \text{otherwise} \end{cases}$$
    Beispiel:
    * $S = S' \cup \{1\} = \{2,3\} \cup \{1\} = \{1,2,3\}$
    * $\Sigma = \{\{2\}, ..., \{1,2\}, ..., \{2,3\}\}$
    * $x_2(\{2,3\}) = 0,\ x_3(\{2\}) = 1$
  - $x_1 = 1^\Sigma$, das spezielle Element kommt nirgends vor
  - $y_1 = 1$ und $\forall s \in S' : y_s = 0$, wir definieren das label das spezielle Element 1, für alle anderen 0

- z.z. Lemma: $\bigcup_{\sigma \in \Sigma'} = S' \Leftrightarrow \forall s \in S' : \prod_{\sigma \in \Sigma'} x_s(\sigma) = 0$
  set-cover kann umgeschrieben werden in ein Produkt mittel der Funktion x_s (das Produkt verhält sich wie ein logisches UND)
  - $\bigcup_{\sigma \in \Sigma'} = S'$
  - $\Leftrightarrow \forall s \in S' \exists \sigma \in \Sigma' : \quad s \in \sigma$, für jedes Sample ex. eine TM mit diesem Sample
  - $\Leftrightarrow$ für dieses Sample gilt somit $x_s(\sigma) = 0$, d.h. es ex. ein Sample für die die Funktion 0 ist
  - $\Leftrightarrow$ Existenz kann mittel des logischen UND's repräsentiert werden $\forall s \in \Sigma' : \prod_{\sigma \in \Sigma'} x_s(\sigma) = 0$

- **Beweis NP-hardness**:
  z.z.: $\exists\, \Sigma' \subseteq \Sigma$ von S' mit $|\Sigma'| \leq m \Leftrightarrow \exists\, \theta \in \Theta : R(\theta) \leq m$ und
  $\forall s \in S : f_\theta(x_s) = y_s$
  d.h., es ex. ein Lösung von set-cover mit bound m gdw. es Parameter θ mit Komplexität $\leq$ m gibt und alle Samples korrekt inferred werden

- ($\Rightarrow$)
  - Sei $\Sigma' \subseteq \Sigma$ ein Cover von S mit $\Sigma'| \leq m$
  - Sei $V_0 = \emptyset$, $V_1 = \Sigma'$, d.h wir definieren das Cover als Menge der nicht-negierten Variablen einer DNF
  - $\forall x' \in X : f_\theta(x') = \prod_{\sigma \in \Sigma'} x'(\sigma)$, (DNF mit nur positiven Variablen)
  - siehe Lemma muss dieses Produkt gleich 0 sein $\forall s \in S' : f(x_s) = 0$
  - Laut Definition gilt zudem $f(1^\Sigma) = 1$
  - daraus folgt, dass alle Daten richtig gelabelt wurden $\forall s \in S' : f(x_s) = y_s$
  - Da wir $V_1 = \Sigma'$ und $V_0 = \emptyset$ gesetzt haben, ist der Regularizer auch gleich $R(\theta) = |\Sigma'| \leq m$

- ($\Leftarrow$)
  - Sei $\theta \in \Theta$ s.d. alle Daten richtig inferred werden und $R(\theta) \leq m$
  -