Bayes'sches Lernen: Übersicht

- Bayes'sches Theorem
- MAP, ML Hypothesen
- MAP Lernen
- Minimum Description Length Principle
- Bayes'sche Klassifikation
- Naive Bayes Lernalgorithmus

2 Zielrichtungen der Bayes'schen Methoden

Bereitstellen von praktischen Lernalgorithmen:

- Naive Bayes
- Bayes'sche Netze
- Kombiniere Wissen (a priori-Wahrscheinlichkeiten) und beobachtete Daten
- Erfordert a priori-Wahrscheinlichkeiten

Bereitstellen eines konzeptuellen Modells

- „Standard“ zum Vergleich mit anderen Lernalgorithmen
- Zusätzliche Einsichten in Occam's Razor

Bayes'sches Theorem

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

- $P(h)$ = a priori Wahrscheinlichkeit der Hypothese h
- $P(D)$ = a priori Wahrscheinlichkeit der Trainingsdaten D
- $P(h|D)$ = Wahrscheinlichkeit von h gegeben D
- $P(D|h)$ = Wahrscheinlichkeit von D gegeben h

Auswahl von Hypothesen

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

Suchen wahrscheinlichste Hypothese gegeben die Traingsdaten

Maximum a posteriori Hypothese h_{MAP}:

$$\begin{aligned} h_{MAP} = \arg\max_{h \in H} P(h|D) \quad &= \arg\max_{h \in H} \frac{P(D|h)P(h)}{P(D)} \\ &= \arg\max_{h \in H} P(D|h)P(h) \end{aligned}$$

Unter der Annahme $P(h_i) = P(h_j)$ kann man weiter vereinfachen und wählt die *Maximum likelihood* (ML)-Hypothese:

$$h_{ML} = \arg\max_{h_i \in H} P(D|h_i)$$

Grundlegende Formeln für Wahrscheinlichkeiten

- *Produktregel*: Wahrscheinlichkeit $P(A \wedge B)$ der Konjunktion zweier Ereignisse A und B:
$$P(A \wedge B) = P(A|B)P(B) = P(B|A)P(A)$$
- *Summenregel*: Wahrscheinlichkeit $P(A \vee B)$ der Disjunktion zweier Ereignisse A und B:
$$P(A \vee B) = P(A) + P(B) - P(A \wedge B)$$
- *Theorem der totalen Wahrscheinlichkeiten*: Wenn die Ereignisse $A_1, \ldots, A_n$ sich gegenseitig ausschließen und $\sum_{i=1}^{n} P(A_i) = 1$, dann
$$P(B) = \sum_{i=1}^{n} P(B|A_i)P(A_i)$$

Brute Force MAP-Hypothesen-Lerner

1. Für jede Hypothese h in H, berechne a posteriori Wahrscheinlichkeit

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

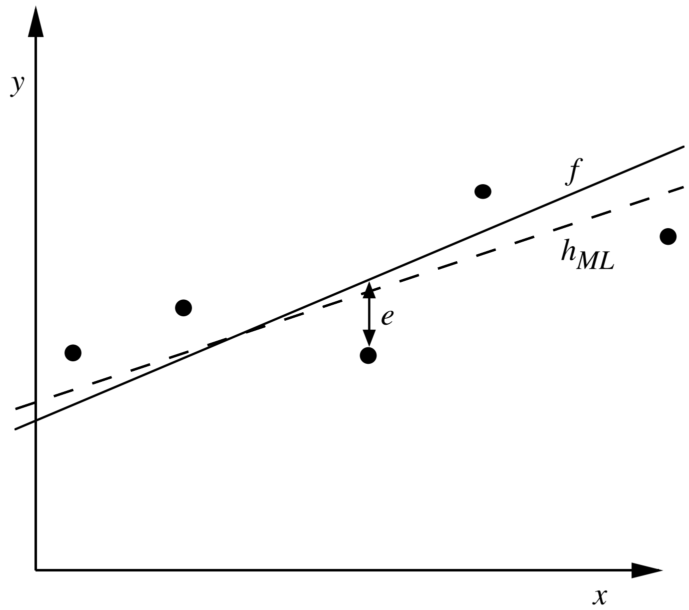
2. Gib Hypothese h_{MAP} mit höchster a posteriori Wahrscheinlichkeit aus

$$h_{MAP} = \operatorname*{argmax}_{h \in H} P(h|D)$$

Beispielanwendung: Lernen einer reelwertigen Funktion

Betrachte reelwertige Zielfunktion f
Trainingsbeispiele sind $\langle x_i, d_i \rangle$,
wobei die d_i verrauscht sind

- $d_i = f(x_i) + e_i$
- e_i ist Zufallsvariable (Noise) die unabhängig voneinander für jedes x_i bezüglich einer Normalverteilung mit Mittelwert=0 gezogen werden



Die Maximum-Likelihood-Hypothese h_{ML} ist nun genau diejenige, die die Summe der Quadrate der Fehler minimiert:

$$h_{ML} = \arg\min_{h \in H} \sum_{i=1}^{n} (d_i - h(x_i))^2$$

Warum?

$$
\begin{aligned}
h_{ML} &= \underset{h\in H}{\operatorname{argmax}}\, p(D|h) \\
&= \underset{h\in H}{\operatorname{argmax}} \prod_{i=1}^{n} p(d_i|h) \\
&= \underset{h\in H}{\operatorname{argmax}} \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{d_i - h(x_i)}{\sigma}\right)^2} \\
&= \underset{h\in H}{\operatorname{argmax}} \sum_{i=1}^{n} \ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2}\left(\frac{d_i - h(x_i)}{\sigma}\right)^2 \\
&= \underset{h\in H}{\operatorname{argmax}} \sum_{i=1}^{n} -\frac{1}{2}\left(\frac{d_i - h(x_i)}{\sigma}\right)^2 \\
&= \underset{h\in H}{\operatorname{argmax}} \sum_{i=1}^{n} -\left(d_i - h(x_i)\right)^2 \\
&= \underset{h\in H}{\operatorname{argmin}} \sum_{i=1}^{n} \left(d_i - h(x_i)\right)^2
\end{aligned}
$$

Minimum Description Length Principle

Occam's Razor: wähle kleinste Hypothese

MDL: bevorzuge Hypothese h, die folgendes minimiert:

$$h_{MDL} = \operatorname*{argmin}_{h \in H} L_{C_1}(h) + L_{C_2}(D|h)$$

wobei $L_C(x)$ die Beschreibungslänge von x unter Kodierung C ist

---

Beispiel: H = Entscheidungsbäume, D = Labels der Traingsdaten

- $L_{C_1}(h)$ ist # Bits zum Beschreiben des Baums h
- $L_{C_2}(D|h)$ ist # Bits zum Beschreiben von D gegeben h
  - Anmerkung: $L_{C_2}(D|h) = 0$ falls alle Beispiele korrekt von h klassifiziert werden. Es müssen nur die Ausnahmen kodiert werden.
- h_{MDL} wägt Baumgröße gegen Traingsfehler ab

Minimum Description Length Principle

$$\begin{aligned} h_{MAP} &= \arg\max_{h \in H} P(D|h)P(h) \\ &= \arg\max_{h \in H} \log_2 P(D|h) + \log_2 P(h) \\ &= \arg\min_{h \in H} -\log_2 P(D|h) - \log_2 P(h) \end{aligned} \quad (1)$$

Interessanter Fakt aus der Kodierungstheorie:

> Die optimale (kürzeste) Kodierung für ein Ereignis mit Wahrscheinlichkeit p benötigt $-\log_2 p$ Bits.

Interpretiere (1):

- $-\log_2 P(h)$: Größe von h bei optimaler Kodierung
- $-\log_2 P(D|h)$: Größe von D gegeben h bei optimaler Kodierung

$\longrightarrow$ wähle Hypothese die folgendes minimiert:

$$length(h) + length(\text{misclassifications})$$

Klassifikation neuer Instanzen

Bis jetzt haben wir die *wahrscheinlichste Hypothese* für gegebene Daten D gesucht (d.h., h_{MAP})

Gegeben neue Instanz x, was ist die wahrscheinlichste *Klassifikation*?

- $h_{MAP}(x)$ ist es nicht unbedingt!!!

Beispiel:

- Betrachte 3 Hypothesen und gegebene Daten D:
$$P(h_1|D) = 0,4;\ P(h_2|D) = 0,3;\ P(h_3|D) = 0,3$$
- Gegeben sei neue Instanz x,
$$h_1(x) = +,\ h_2(x) = -,\ h_3(x) = -$$
- Was ist $h_{MAP}(x)$, was ist wahrscheinlichste Klassifikation von x?

Bayes'sche optimale Klassifikation

Bayes'sche optimale Klassifikation:

$$\arg\max_{v_j \in V} \sum_{h_i \in H} P(v_j|h_i)P(h_i|D)$$

Für unser Beispiel:

$$\begin{array}{lll} P(h_1|D) = 0,4; & P(-|h_1) = 0; & P(+|h_1) = 1 \\ P(h_2|D) = 0,3; & P(-|h_2) = 1; & P(+|h_2) = 0 \\ P(h_3|D) = 0,3; & P(-|h_3) = 1; & P(+|h_3) = 0 \end{array}$$

Deshalb:

$\sum_{h_i \in H} P(+|h_i)P(h_i|D) = 0,4$ $\qquad$ $\sum_{h_i \in H} P(-|h_i)P(h_i|D) = 0,6$

Gibbs Klassifikation

Bayes'sche Klassifikation optimal, aber teuer bei vielen Hypothesen

Gibbs Algorithmus:

1. Wähle zufällig eine Hypothese h bezüglich $P(h|D)$
2. Benutze h zur Klassifikation

Überraschung: Sei ein Zielkonzept zufällig bezüglich $\mathcal{D}$ aus H gewählt. Dann:

$$E[error_{Gibbs}] \leq 2 \cdot E[error_{BayesOptimal}]$$

Naive Bayes Klassifikation

Neben Entscheidungsbäumen, Neuronalen Netzen, Nearest Neighbour eine der am meisten eingesetzten Lernmethoden.

Wann anwendbar:

- Mittlere oder große Traingsmengen
- Attribute sind bedingt unabhängig gegeben die Klassifikation

Erfolgreiche Anwendungsgebiete:

- Diagnose
- Klassifikation von Textdokumenten

Naive Bayes Klassifikation

Ziel $f : X \rightarrow V$, jede Instanz durch Attribute $\langle a_1, a_2 \ldots a_n \rangle$ beschrieben

Wahrscheinlichster Wert von $f(x)$:

$$
\begin{aligned}
v_{MAP} &= \operatorname*{argmax}_{v_j \in V} P(v_j | a_1, a_2 \ldots a_n) \\
&= \operatorname*{argmax}_{v_j \in V} \frac{P(a_1, a_2 \ldots a_n | v_j) P(v_j)}{P(a_1, a_2 \ldots a_n)} \\
&= \operatorname*{argmax}_{v_j \in V} P(a_1, a_2 \ldots a_n | v_j) P(v_j)
\end{aligned}
$$

Annahme von Naive Bayes: $P(a_1, a_2 \ldots a_n | v_j) = \prod_i P(a_i | v_j)$

Naive Bayes Klassifikation: $v_{NB} = \operatorname*{argmax}_{v_j \in V} P(v_j) \prod_i P(a_i | v_j)$

Naive Bayes Algorithmus

Naive_Bayes_Learn(*examples*):

Für jeden Klassifikationswert v_j

$\hat{P}(v_j) \leftarrow$ schätze $P(v_j)$

Für jeden Attributwert a_i jedes Attributs a

$\hat{P}(a_i|v_j) \leftarrow$ schätze $P(a_i|v_j)$

Ergebnis: Tabelle mit geschätzten WKen

Classify_New_Instance(x):

$$v_{NB} = \operatorname*{argmax}_{v_j \in V} \hat{P}(v_j) \prod_{a_i \in x} \hat{P}(a_i|v_j)$$

Naive Bayes: Beispiel

Betrachte *PlayTennis* mit neuer Instanz

$$\langle Outlk = sun, Temp = cool, Humid = high, Wind = strong \rangle$$

Wollen berechnen:

$$v_{NB} = \operatorname*{argmax}_{v_j \in V} P(v_j) \prod_i P(a_i|v_j)$$

$$P(yes)\ P(sun|yes)\ P(cool|yes)\ P(high|yes)\ P(strong|yes) = .005$$

$$P(no)\ P(sun|no)\ P(cool|no)\ P(high|no)\ P(strong|no) = .021$$

$$\rightarrow v_{NB} = no$$

Naive Bayes: Diskussion (1)

Annahme der bedingten Unabhängigkeit ist oft nicht erfüllt

$$P(a_1, a_2 \ldots a_n | v_j) = \prod_i P(a_i | v_j)$$

...aber es funktioniert trotzdem erstaunlich gut. Warum? Abschätzungen für $\hat{P}(v_j|x)$ müssen nicht notwendig korrekt sein, sondern nur

$$\operatorname*{argmax}_{v_j \in V} \hat{P}(v_j) \prod_i \hat{P}(a_i|v_j) = \operatorname*{argmax}_{v_j \in V} P(v_j) P(a_1 \ldots, a_n | v_j)$$

Naive Bayes: Diskussion (2)

Was, wenn aufgrund kleiner Trainingsmengen keines der Trainingsbeispiele mit Klassifikation v_j den Attributwert a_i hat? Dann

$$\hat{P}(a_i|v_j) = 0, \text{ und...}$$

$$\hat{P}(v_j) \prod_i \hat{P}(a_i|v_j) = 0$$

Typische Lösung: m-Abschätzung: $\hat{P}(a_i|v_j) \leftarrow \frac{n_c + mp}{n+m}$

wobei

- n ... Anzahl der Trainingsbeispiele mit $v = v_j$,
- n_c ... Anzahl der Beispiele mit $v = v_j$ und $a = a_i$
- p ... a priori Schätzung für $\hat{P}(a_i|v_j)$
  (z.B. durch Annahme uniformer Verteilung der Attributwerte $\longrightarrow$ $p = \frac{1}{|values(a_i)|}$)
- m ... Gewicht für a priori-Abschätzung p (Anzahl "virtueller" Beispiele)