ZK 2021

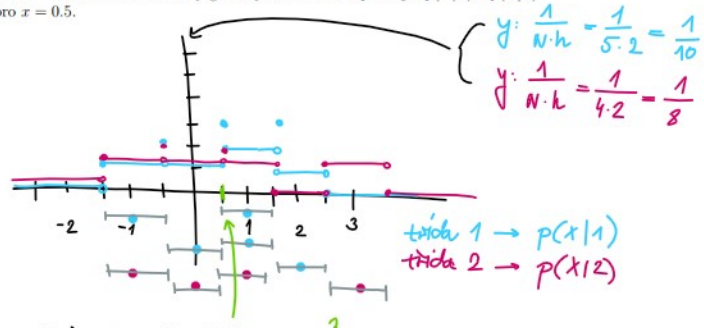
neděle 15. ledna 2023 14:40

1. (6b) Parzenova okna a Neyman-Pearson.
Mějme dvě třídy označené 1 a 2. Trénovací množina pro třídu 1 je $X_1 = \{1, -1, 1, 2, 0\}$, pro třídu 2 je $X_2 = \{-1, 0, 1, 3\}$.

   (a) (3b) Z dat X_1 odhadněte podmíněnou hustotu $p(x \mid 1)$ metodou Parzenových oken. Analogicky odhadněte hustotu $p(x \mid 2)$ pro druhou třídu. Použijte jádrovou funkci $K(x, y) = k(x - y)$, kde $k(z)$ je definovaná takto:

$$k(z) = 1/h \quad \text{pro } |z| \le h/2\,,$$
$$k(z) = 0 \quad \text{pro } |z| > h/2\,,$$

   kde $h = 2$. Odhadnuté hustoty graficky znázorněte. Spočítejte $p(x \mid 1)$ a $p(x \mid 2)$ pro $x = 0.5$.



   (b) (3b) Třída 2 představuje nebezpečí. Zkonstruujte Neyman-Pearsonovu strategii takovou, aby přehlédnuté nebezpečí bylo $< 40\%$. Pro nalezenou strategii spočítejte chyby klasifikace $\varepsilon_1, \varepsilon_2$ pro obě třídy 1 a 2.

---

2. (8b) Adaboost.

   (a) (3b) Vysvětlete metodu Adaboost tak, aby ji podle Vašeho popisu kterýkoli Váš kolega dokázal naimplementovat.

AdaBoost učící algoritmus vybere z velké množiny slabých klasifikátorů H malou podmnožinu klasifikátorů a jejich váhy, tak aby výsledný (silný) klasifikátor byl přesný

- Metoda Adaboost vybírá v množiny slabých klasifikátorů
- Chce zkonstruovat silný klasifikátor z těch slabých

Algoritmus:

- Vstup:
  - trénovací množina $T = \{(x_1, y_1), \dots, (x_n, y_n)\}$; $x_i \in X$, $y_i \in (-1, 1)$
  - počet iterací T
  - set slabých klasifikátorů $\mathcal{H}$

0) Inicializace vah dat na 1/N (kde N je počet dat) $D_1(i) = 1/n \quad i = 1, \dots, n$

Pro iterace t = 1 do T:

1) Výběr slabého klasifikátoru s nejmenším errorem
   - --> Minimalizuju chybu

$$h_t = \underset{h_j \in \mathcal{H}}{\operatorname{argmin}}\ \varepsilon_j \quad \rightarrow \quad \varepsilon_j = \sum_{i=1}^{n} D_t(i)\, \mathbb{1}[y_i \neq h_j(x_i)]$$

$[\![y_i \neq h_j(x_i)]\!] \rightarrow$ True = 1, False = 0

2) Když chyba té iterace ε_t je větší než 1/2 --> KONEC

$$\text{if } \varepsilon_t \geq 0.5 \rightarrow \text{STOP}$$

3) Nastavit váhu klasifikatoru $\alpha_t = \frac{1}{2} \ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$

If $\varepsilon_t = 0.5 \rightarrow$ stop

3) Nastavit váhu klasifikatoru $\alpha_t = \frac{1}{2} \ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$

4) Updatnout váhy dat $D_{t+1}(i) = \frac{D_t(i) \cdot \exp(-\alpha_t h_t(x_i) y_i)}{Z_t}$

- Výstup:
  - silný klasifikátor

$H(x) = \text{sign}(F(x)) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$

Z_t ↳ normalizační faktor, aby byla distribuce (tj. aby se sčítalo do 1)

What is time complexity of training the Adaboost algotithm?

The time complexity of training the Adaboost algorithm is typically O(MNT), where M is the number of data samples, N is the number of features, and T is the number of iterations or rounds of boosting. This is because in each round, the algorithm needs to iterate over all the data samples to update the weights, and in total, there are T rounds of boosting. The number of features does not generally affect the time complexity.

(b) (1b) Popište časovou náročnost učení.

- O(N*M*T)
- N ... počet dat
- M ... počet klasifikátorů
- T .... Počat iterací
- Záléží na rychlosti a klasfikátoru a na jejich počtu

What is time complexity of classification of the Adaboost algotithm?

The time complexity of classifying a new sample using the Adaboost algorithm is typically O(T*M), where T is the number of iterations or rounds of boosting, and M is the number of weak classifiers in the final ensemble. This is because in order to classify a new sample, the algorithm needs to make a prediction using each of the weak classifiers in the ensemble and combine their predictions using a weighted majority vote. The number of weak classifiers is usually determined by the number of iteration of boosting, so it's generally directly proportional to T.

(c) (1b) Diskutujte časovou náročnost klasifikace.

- O(N*M)
- N ... počet dat
- M ... počet slabých klasifikátorů
- Podle složitosti jednotlivých klasifikátoru

(d) (1b) Napište, jestli a jak je možné chod algoritmu zrychlit při učení/klasifikaci.

- Zmenšení trénovací množiny T
- Zvolit rychlejší (efektivnější) klasifikátory
- Zmenšení počtu iterací
- Paralelizace (klasifikovat na vícero CPU)

Highlight advantages of Adaboost with respect to SVM, Logistic regression, the nearest neighbour and neural network learning and classification methods.

Adaboost has several advantages over other learning and classification methods such as SVM, logistic regression, nearest neighbor, and neural networks:

1. Adaboost is a simple and easy-to-implement algorithm, which does not require a lot of computational resources.
2. Adaboost is versatile, and can be used with a wide range of weak classifiers, such as decision stumps, decision trees, and neural networks.
3. Adaboost is a robust algorithm that can handle datasets with a large number of features and can handle noisy or mislabeled data.
4. Adaboost can achieve high classification accuracy by combining a large number of weak classifiers.
5. Adaboost is less prone to overfitting than other algorithms such as neural networks.
6. Adaboost can be used to solve both binary and multi-class classification problems.
7. Adaboost is less sensitive to the initial conditions than other algorithms such as SVM and Logistic Regression.
8. Adaboost can be combined with other algorithms to improve their performance, for example with Random Forests or Decision Trees.

(e) (2b) Popište výhody Adaboostu ve srovnání s SVM, logistickou regresí, nejbližším sous dem a neuronovými sítěmi (zvažte jak fázi učení, tak i fázi klasifikace.)

- Výhody celkově:
  - Velmi jednoduché na implementaci
  - Je to lineární klasifikátor
  - Výsledek konverguje do lagaritmu likelihood ratia
  - Je víceúčelový - dá se použít na různé typy slabých klasfikátorů
  - Poradí si s šumem

---

3. (3b) Maximálně věrohodný odhad parametrů. Normální rozdělení je dáno jako

$$p(\mathbf{x}) = (2\pi)^{-\frac{d}{2}} \det(\mathbf{C})^{-\frac{1}{2}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathsf{T}}\mathbf{C}^{-1}(\mathbf{x}-\boldsymbol{\mu})},$$

kde $\mathbf{x}, \boldsymbol{\mu} \in \mathbb{R}^d$, $\mathbf{C} \in \mathbb{R}^{d\times d}$, $\mathbf{C} \succeq 0$. Mějme N nezávislých pozorování $(x_i)_{i=1}^N$ z tohoto rozdělení. Kovarianční matice $\mathbf{C}$ je známá. Najděte maximálně věrohodný odhad parametru $\boldsymbol{\mu}$.

maximum likelihood ML

$\hat{\mu} = \arg\max_{\mu} L(\mu) = \arg\max_{\mu} p(\mathcal{X}/\mu)$

$L(\mu) = \prod_{i=1}^{N} p(x_i/\mu)$

$l(\mu) = \ln L(\mu) = \ln\left(\prod_{i=1}^{N} p(x_i/\mu)\right)$

$p(\mathbf{x}) = (2\pi)^{-\frac{d}{2}} \cdot \det(\mathbf{C})^{-\frac{1}{2}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathsf{T}}\mathbf{C}^{-1}(\mathbf{x}-\boldsymbol{\mu})}$, w

$l(\mu) = \sum_{i=1}^{N} \ln(p(x)) = \sum_{i=1}^{N} \ln\left((2\pi)^{-\frac{d}{2}}\right) + \ln\left(\det(C)^{-\frac{1}{2}}\right) + \underbrace{\ln(e^{w})}_{w}$

$\frac{\partial l(\mu)}{\partial \mu} = \sum_{i=1}^{N} 0 + 0 + \frac{\partial w(\mu)}{\partial \mu}$

$w = y^{\mathsf{T}} A y \rightarrow \frac{\partial w}{\partial \mu} = y^{\mathsf{T}}\left(A + A^{\mathsf{T}}\right)$ $\quad \mathbf{x}^T\mathbf{A}\mathbf{x}$ | $\mathbf{x}^T(\mathbf{A}+\mathbf{A}^T)$

$w = -\frac{1}{2} y^T A y \rightarrow \frac{\partial w}{\partial \mu} = y^T (A + A^T)$ $\quad \vec{x}^T \vec{A} \vec{x} \quad | \vec{x}^T (A + A^T)$

$(x-\mu)$ $\quad C^{-1}$ $\quad C \ldots$ kovariační matice $\rightarrow$ čtverec symetrická

$C^T = C$

$$\frac{\partial l(\mu)}{\partial \mu} = \sum_{i=1}^{N} -\frac{1}{2} \cdot (x-\mu)^T \cdot \underbrace{\left(C^{-1} + (C^{-1})^T\right)}_{2C^{-1}} \cdot (-I) = 0$$

$$\sum_{i=1}^{N} + \frac{1}{2} \cdot (x-\mu)^T \cdot 2 C^{-1} \cdot I = 0$$

$$\sum_{i=1}^{N} \underbrace{(x-\mu)^T \cdot C^{-1}}_{} = 0$$

$$\Sigma (x^T - \mu^T) \cdot C^{-1} = 0$$

$$\Sigma x^T C^{-1} \Sigma \mu^T \cdot C^{-1} = 0$$

$$+ \Sigma \mu^T \cdot C^{-1} = + \Sigma x^T C^{-1}$$

$$\mu^T = x^T C^{-1} \cdot C$$

$$\hat{\mu} = \Sigma x_i \quad I$$

---

4. (3b) Výběr klasifikátoru.
Máte k dispozici trénovací data a máte za úkol implementovat jednoduchý klasifikátor. Chcete, aby klasifikátor fungoval co nejlépe. Rozhodl/a jste se, že začnete s metodou K-nejbližšího souseda (K-NN). Vysvětlete, jak vyberete K.

- Záleží co máme za data
- Doporučené mít k liché, abychom předešli nerozhodným počtům v klasifikaci
- Cross-validace
  - Rozdělíme data na trénovací a testovací množinu (např. 70:30)
  - U k-NN crossvalidace na 3 podmnožiny - přidáme další pro výběr správného k
  - Takže na trénovací natrénujeme model, pak pustíme na cross-validační data - vybereme nejúspěšnější k a pak otestujeme na testovacích datech

In cross-validation, instead of splitting the data into two parts, we split it into 3. Training data, cross-validation data, and test data. Here, we use training data for finding nearest neighbors, we use cross-validation data to find the best value of "K" and finally we test our model on totally unseen test data. This test data is equivalent to the future unseen data points.