Artificial Intelligence & Machine Learning and Pattern Recognition ——Summary

Yanghui Rao
Assistant Prof., Ph.D
School of Mobile Information Engineering,
Sun Yat-sen University
raoyangh@mail.sysu.edu.cn

k-Means (exercise)

- Given the following 6 points with 2 attributes:

  A: (1, 3), B: (2, 1), C: (2, 2), D: (3, 5), E: (4, 4), F: (3, 3).
- a) We need to group all 6 points into three clusters. Suppose initially we assign B, D and E as the prototype of the first, second and third cluster respectively. Use the k-Means algorithm to find the three clusters and their respective centroids after the first iteration.
- b) If the initial class label of A, D and E is “C1”, the initial class label of B, C and F is “C2”, use the k-Means algorithm to find the two clusters and their respective centroids until convergence.

k-Means (answer)

- a) After the first iteration:

The first cluster is {A, B, C}, and its centroid is (5/3, 2).

The second cluster is {D}, and its centroid is (3, 5).

The third cluster is {E, F}, and its centroid is (3.5, 3.5).

- b) Initially, the first cluster “C1” is {A, D, E}, and its centroid is (8/3, 4).

The second cluster “C2” is {B, C, F}, and its centroid is (7/3, 2).

After the first iteration, the first cluster “C1” is {D, E, F}, and its centroid is (10/3, 4).

The second cluster “C2” is {A, B, C}, and its centroid is (5/3, 2).

Then, the k-Means algorithm is convergence.

DBSCAN (exercise)

- We consider the following 6 data points:

p1: (5, 9), p2: (5, 8), p3: (3, 8), p4: (1, 2), p5: (2, 1), p6: (4, 4).

The distance function is Euclidean distance.

- Find the clusters in this data set based on DBSCAN, with Eps=2 and Minpts=3. Identify the core points, border points and noise points.

DBSCAN (answer)

- The neighborhood of each point is as follows:

N(p1)={p1, p2}, N(p2)={p1, p2, p3}, N(p3)={p2, p3}, N(p4)={p4, p5}, N(p5)={p4, p5}, N(p6)={p6}. Thus,

- The core point is p2.
- The border points are p1, p3.
- The noise points are p4, p5, p6.

Probability

- **Product rule:**
  $$P(A,B)=P(A)P(B|A)=P(B)P(A|B)$$
  $$P(A,B_1,B_2,B_3)=P(A)P(B_1|A)P(B_2|A,B_1)P(B_3|A,B_1,B_2)$$

- **Sum rule:** $P(A)=P(A,B)+P(A,B^c)$

$$P(A)=\sum_{i=1}^{n}P(A,B_i)$$

$$=\sum_{i=1}^{n}P(A|B_i)P(B_i)$$

$$\sum_G P(G|L)=1$$

Truth Tables

- Truth tables are used to define logical connectives and to determine when a complex sentence is true given the values of the symbols in it
- Note that $\Rightarrow$ is a logical connective, so $P \Rightarrow Q$ is a logical sentence and has a truth value, i.e., is either true or false

Truth tables for the five logical connectives

P	**Q**	**$\neg$ P**	**P $\wedge$ Q**	**P $\vee$ Q**	**P $\Rightarrow$ Q**	**P $\Leftrightarrow$ Q**
False	False	True	False	False	True	True
False	True	True	False	True	True	False
True	False	False	False	True	False	False
True	True	False	True	True	True	True

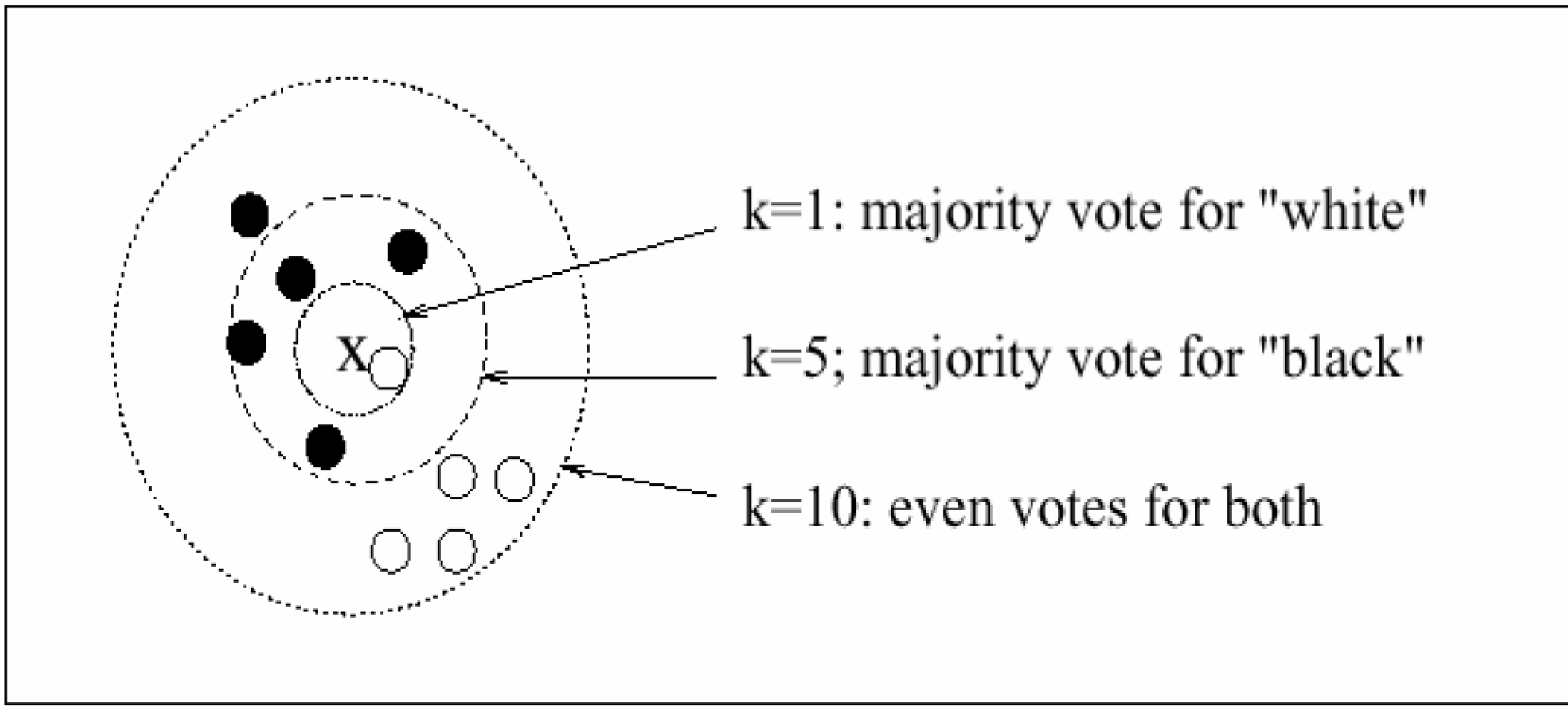
Quantifier Scope

- If a quantifier Q is followed by (, then the scope of Q is to the matched)
  - $\underline{\forall x\ (F(x) \Leftrightarrow F(h))}$
- If a quantifier Q is not followed by (or another quantifier, then the scope of Q is to the first connective
  - $\underline{\forall x\ F(x)} \Leftrightarrow F(h)$
- If a quantifier $Q1$ is followed by another quantifier $Q2$, then the scope of $Q1$ is to the scope of $Q2$
  - $\underline{\forall x\ \underline{\exists y\ R(x, y)}}$
- F: … can fly
- h: human being

False		True
$\forall x\ (F(x) \Leftrightarrow F(h))$	$\not\Leftrightarrow$	$\forall x\ F(x) \Leftrightarrow F(h)$

k-Nearest Neighbor

k-NN using a majority voting scheme



Naïve Bayesian Classifier

- This can be derived from Bayes' theorem

$$P(C_i \mid \mathbf{X}) = \frac{P(\mathbf{X} \mid C_i) P(C_i)}{P(\mathbf{X})}$$

- Since $P(\mathbf{X})$ is constant for all classes, only

$$P(C_i \mid \mathbf{X}) \propto P(\mathbf{X} \mid C_i) P(C_i)$$

  needs to be maximized
- $P(C_i)$ can be obtained from training set s_i/s

Derivation

- **Assumption**: attributes are conditionally independent (i.e., no dependence relation between attributes): $$P(\mathbf{X} \mid C_i) = \prod_{k=1}^{n} P(x_k \mid C_i)$$
- This greatly reduces the computation cost: Only counts the class distribution
- If A_k is categorical, $P(x_k|C_i) = s_{ik}/s_i$, count the distribution
- If A_k is continuous-valued, $P(x_k|C_i)$ can be computed based on Gaussian distribution

Information Gain (ID3)

- Class label: buy_computer="yes/no"
- 用字母D表示类标签，字母A表示每个属性
- $H(D)$=0.940

14个训练样本中，9个买了电脑

$$H(D) = -\frac{9}{14}\log_2\frac{9}{14} - (1-\frac{9}{14})\log_2(1-\frac{9}{14})$$

- $H(D|A="age")$=0.694

$$H(D|A="age") = \frac{5}{14}\times\left(-\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5}\right)$$

$$+\frac{4}{14}\times\left(-\frac{4}{4}\log_2\frac{4}{4} - \frac{0}{4}\log_2\frac{0}{4}\right) + \frac{5}{14}\times\left(-\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5}\right)$$

Information Gain (ID3)

- Class label: buy_computer="yes/no"
- Compute the mutual information (互信息) between D and each attribute A
- $H(D)$=0.940
- $H(D|A=\text{"age"})$=0.694
- $g(D,A=\text{"age"})$=0.246
- $g(D,A=\text{"income"})$=0.029
- $g(D,A=\text{"student"})$=0.151
- $g(D,A=\text{"credit\_rating"})$=0.048

"age"这个属性的条件熵最小（等价于信息增益最大），因而首先被选出作为根节点

$$g(D,A) = H(D) - H(D|A)$$

Information Gain Ratio (C4.5)

- $\text{GainRatio}_A(D) = \text{Gain}_A(D) / \text{SplitInfo}_A(D)$

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right)$$

- $\text{GainRatio}_{A=\text{“}income\text{”}}(D) = ?$

$$SplitInfo_{A=\text{"}income\text{"}}(D)$$
$$= -\frac{4}{14} \times \log_2\left(\frac{4}{14}\right) - \frac{6}{14} \times \log_2\left(\frac{6}{14}\right) - \frac{4}{14} \times \log_2\left(\frac{4}{14}\right)$$
$$= 0.926$$

- $\text{GainRatio}_{A=\text{“}income\text{”}}(D) = 0.029/0.926 = 0.031$

Gini Index (CART)

- D has 9 samples in buys_computer = "yes" and 5 in "no"

$$gini(D) = 1 - (\frac{9}{14})^2 - (\frac{5}{14})^2 = 0.459$$

- The attribute *income* partitions D into 10 in D_1: {medium, high} and 4 in D_2

$$gini_{income \in \{\text{medium,high}\}}(D) = \frac{10}{14} gini(D_1) + \frac{4}{14} gini(D_2)$$

$$= \frac{10}{14}\left(1 - (\frac{6}{10})^2 - (\frac{4}{10})^2\right) + \frac{4}{14}\left(1 - (\frac{1}{4})^2 - (\frac{3}{4})^2\right)$$

$$= 0.450 = gini_{income \in \{low\}}(D)$$

Decision Tree

- But how can we compute the gini index, information gain of an attribute that is **continuous-valued**?
  - Given v values of A, then v-1 possible splits are evaluated. For example, the midpoint between the values a_i and a_{i+1} of A is $(a_i + a_{i+1})/2$

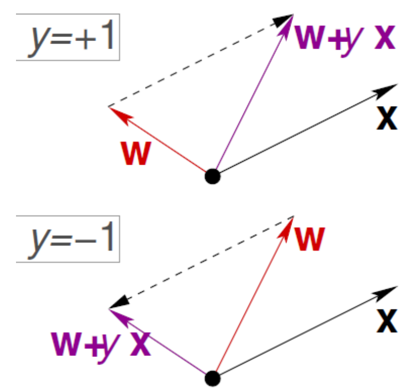
Incorporating model complexity

- In the case of a decision tree, let
  - L be the number of leaf nodes.
  - n_l be the l-th leaf node.
  - $m(n_l)$ be the number of training records classified by n_l.
  - $r(n_l)$ be the number of misclassified records by n_l.
  - $\zeta(n_l)$ be a penalty term associated with the node n_l.
- The resulting error e_c of the decision tree can be estimated as follows:

$$e_c = \frac{\sum_{l=1}^{L} \left(r(n_l) + \zeta(n_l) \right)}{\sum_{l=1}^{L} m(n_l)}$$

Perceptron Learning Algorithm

- Difficult: the set of $h(\mathbf{x})$ is of infinite size
- Idea: start from some initial weight vector $\mathbf{w}_{(0)}$, and "correct" its mistakes on D
- For t = 0, 1,…
  - find a mistake of $\mathbf{w}_{(t)}$ called $(\mathbf{x}_{n(t)}, y_{n(t)})$
    $$sign\left(\mathbf{w}_{(t)}^{\mathrm{T}}\mathbf{x}_{n(t)}\right) \neq y_{n(t)}$$
  - (try to) correct the mistake by
    $$\mathbf{w}_{(t+1)} \leftarrow \mathbf{w}_{(t)} + y_{n(t)}\mathbf{x}_{n(t)}$$
  - until no more mistakes
- Return last $\mathbf{W}$ (called $\mathbf{W}_{\mathrm{PLA}}$)



Perceptron Learning Algorithm

- Only if there exists an hyperplane that correctly classifies the data, the Perceptron procedure is guaranteed to converge; furthermore, the algorithm may give different results depending on the order in which the elements are processed, indeed several different solutions exist.

Logistic Regression Model

- Gradient Decent (梯度下降)
  - Calculate the gradient vector
  - Update the weighting in the opposite direction of the gradient vector at each surface point
- Repeat: $\tilde{\mathbf{w}}_{new}^{(j)} = \tilde{\mathbf{w}}^{(j)} - \eta \dfrac{\partial C(\tilde{\mathbf{w}})}{\partial \tilde{\mathbf{w}}^{(j)}}$

$$= \tilde{\mathbf{w}}^{(j)} - \eta \sum_{i=1}^{n} \left[\left(\frac{e^{\tilde{\mathbf{w}}^{\mathrm{T}} \tilde{\mathbf{x}}_i}}{1 + e^{\tilde{\mathbf{w}}^{\mathrm{T}} \tilde{\mathbf{x}}_i}} - y_i \right) \tilde{\mathbf{x}}_i^{(j)} \right]$$

- Until convergence

Neural Network

Propagate the inputs forward

- Given a unit j in a hidden or output layer, the net input, I_j, to unit j is $I_j = \sum_i w_{ij} O_i + \theta_j$

  where w_{ij} is the weight of the connection from unit i in the previous layer to unit j; O_i is the output of unit i from the previous layer; and θ_j is the bias of the unit.

- Given the net input I_j to unit j, then O_j, the output of unit j, is computed as $O_j = \dfrac{1}{1+e^{-I_j}}$

Backpropagate the error

- For a unit k in the output layer, the error Err_k is computed by

  $$Err_k = O_k(1-O_k)(T_k - O_k)$$

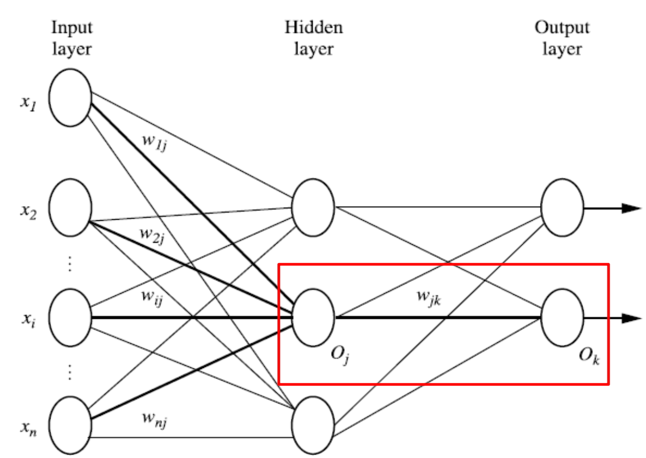
- The error of a hidden layer unit j is

  $$Err_j = O_j(1-O_j)\sum_k Err_k w_{jk}$$

- Weights are updated by

  $$w_{jk} = w_{jk} + \eta Err_k O_j$$

  $$\theta_k = \theta_k + \eta Err_k$$



Contact

- 成绩公布
  - 成绩会陆续公布在QQ群：413433008
  - 有任何疑问，请发送邮件至 raoyangh@mail.sysu.edu.cn

Contact

- 目前在我们研究团队中，以及对我们研究方向感兴趣的本届同学：
  - 2015级研究生：罗茂权、郑文杰、梁伟明、赵施宇
  - 2013级本科生：王耀威、詹雪莹、陈慧均、李祥圣、庞健辉、莫碧云、黄国燕、卢宇沣、罗锦涛、于济玮、陈樑源、李声涛、胡泽杰、梁倩乔、李建立、曹启正、黄行昌、林东定、张煜昊、刘爽、刘健。。。

- 欢迎志同道合的同学们互相讨论、共同进步！

谢谢大家！