Artificial Intelligence ——总结

Yanghui Rao
Assistant Prof., Ph.D
School of Data and Computer Science,
Sun Yat-sen University
raoyangh@mail.sysu.edu.cn

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

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

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

Probability

- Exercise: 假设有一盒骰子，里面有4面的（点数为1、2、3、4），6面的、8面的、12面的、20面的均匀骰子各1个。如果我随机从盒子中选一个骰子，投掷它得到了点数5。那么我选中的骰子为4面、6面、8面、12面、20面的概率各是多少?

答案: **0, 0.392, 0.294, 0.196, 0.118**

Probability

- There are two random variables X and Y. Which of the following is always true?
  - A. $\sum_X P(X|Y) = 1$
  - B. $\sum_Y P(X|Y) = 1$
  - C. All of the above
  - D. None of the above
- Is the statement True or False? Entropy of a discrete random variable is always non-negative.

Probability

- There are two random variables X and Y. Which of the following is always true? (Answer: A)
  - A. $\sum_X P(X|Y) = 1$
  - B. $\sum_Y P(X|Y) = 1$
  - C. All of the above
  - D. None of the above
- Is the statement True or False? Entropy of a discrete random variable is always non-negative. (Answer: True)

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 $\qquad\qquad$ True

$\forall x\ (F(x) \Leftrightarrow F(h)) \quad \not\Leftrightarrow \quad \forall x\ F(x) \Leftrightarrow F(h)$

Exercise

- Fill in the following truth table:

P	Q	$(P \Rightarrow Q) \wedge (Q \Rightarrow P)$	$(\neg P \vee Q) \Leftrightarrow (P \Rightarrow Q)$
True	True		
True	False		
False	True		
False	False		

- If we represent "... is hot" by H(…), and represent "fire" by f, what are the values of "$H(f) \Rightarrow \forall x\ H(x)$" and "$\exists x\ (H(f) \Leftrightarrow H(x))$"?

Exercise

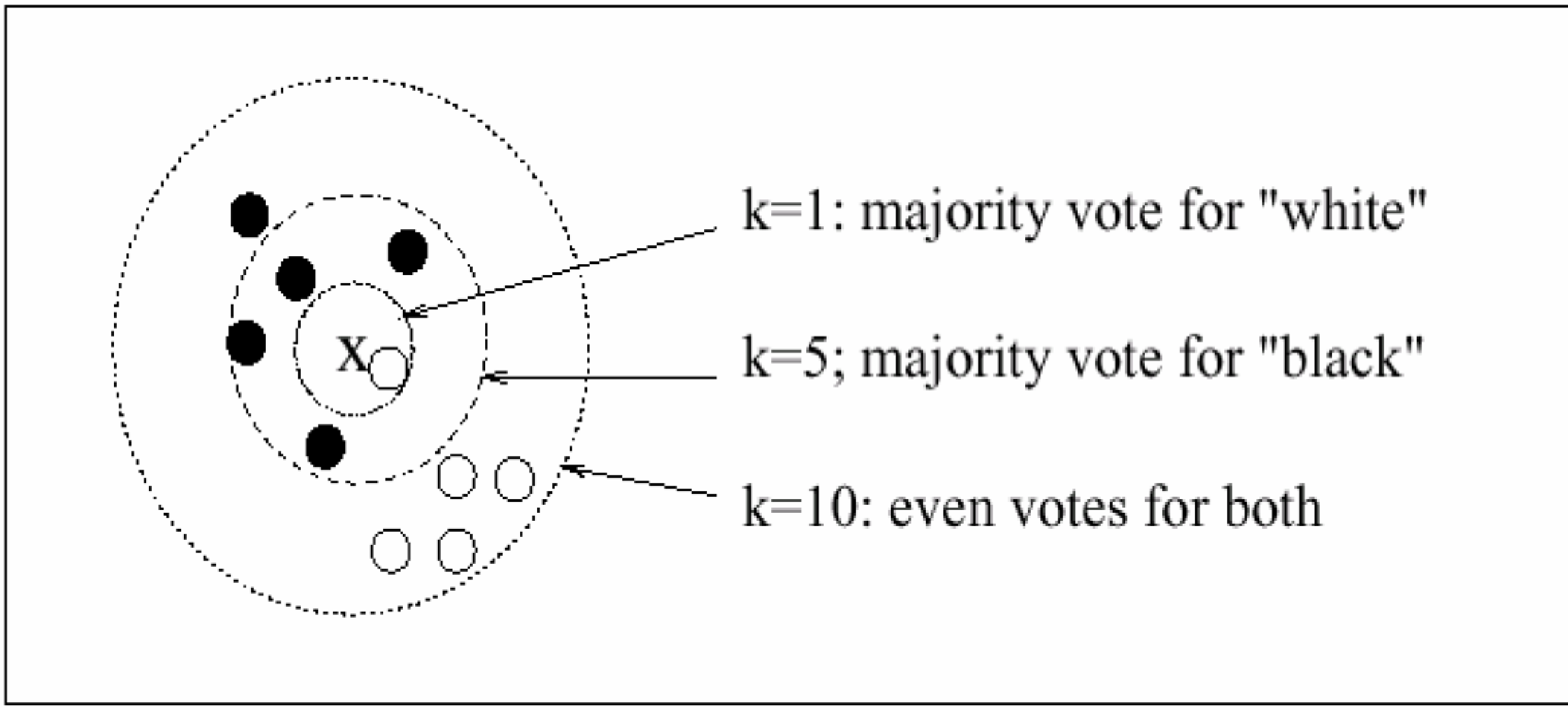
- Fill in the following truth table:

P	Q	$(P \Rightarrow Q) \wedge (Q \Rightarrow P)$	$(\neg P \vee Q) \Leftrightarrow (P \Rightarrow Q)$
True	True	True	True
True	False	False	True
False	True	False	True
False	False	True	True

- If we represent "... is hot" by H(…), and represent "fire" by f, what are the values of "$H(f) \Rightarrow \forall x\, H(x)$" and "$\exists x\, (H(f) \Leftrightarrow H(x))$"? (Answer: False, True)

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

Exercise

- What is the meaning of "k" for the k-Nearest Neighbor (i.e., k-NN) and the k-Means clustering algorithm?
- If using k-NN for classification, what is the predicted class label when "x = 5"? Is there any difference if based on City Block, Euclidean, or Supremum distance?

Note: Given a testing sample x, if there are multiple training samples' distances are the nearest, k-NN classifier will use the mode (众数) of the class labels of all nearest training samples as the predicted class label of x

X	Y
2	-
3.3	-
3.2	-
3.1	+
5	?

X	Y
2	+
3.3	-
3.2	-
3.1	+
5	?

Exercise

- What is the meaning of “k” for the k-Nearest Neighbor (i.e., k-NN) and the k-Means clustering algorithm?
  - Answer: a) The parameter “k” means the number of neighbors used to classify test examples for the k-NN. b) The parameter “k” specifies the number of clusters for the k-Means.

- Given a testing sample x, if there are multiple training samples’ distances are the nearest, k-NN classifier will use the mode (众数) of the class labels of all nearest training samples as the predicted class label of x.
  - Answer: There is no difference if based on those distance measures. (1) Left table. The predicted class label is “-”; (2) Right table. The predicted class label is “-” for k=1, 2, 3 and “Unknown” for k > 3.

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

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

- $H(D \mid A=\text{"}age\text{"})$=0.694

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

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

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
- $H(D \mid A=\text{"age"})$=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

$$g(D,A) = H(D) - H(D \mid A)$$

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:

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

Exercise

- We consider the training examples shown in the following table for a binary classification problem.
  - Calculate the respective changes in the Gini index value when a_1 and a_2 are used for partitioning the training set.
  - Calculate the respective changes in the classification (training) error when a_1 and a_2 are used for partitioning the training set.

a_1	a_2	a_3	Target Class
T	T	1	+
T	T	6	+
T	F	5	-
F	F	4	+
F	T	7	-
F	T	3	-
F	F	8	-
T	F	7	+
F	T	5	-

Exercise

- (1) The original Gini index is $1-(\frac{4}{9})^2-(\frac{5}{9})^2=0.494$

  After splitting on a_1, the Gini index becomes

  $$\frac{4}{9}[1-(\frac{3}{4})^2-(\frac{1}{4})^2]+\frac{5}{9}[1-(\frac{1}{5})^2-(\frac{4}{5})^2]=0.344$$

  As a result, the change in Gini index is

  $\triangle G(a_1) = 0.494 - 0.344 = 0.15.$

  After splitting on a_2, the Gini index becomes

  $$\frac{5}{9}[1-(\frac{2}{5})^2-(\frac{3}{5})^2]+\frac{4}{9}[1-(\frac{2}{4})^2-(\frac{2}{4})^2]=0.489$$

  As a result,

  $\triangle G(a_2) = 0.494 - 0.489 = 0.005.$

Exercise

- (2) The original classification error is $1-\max(\frac{4}{9},\frac{5}{9})=\frac{4}{9}$

  After splitting on a_1, the classification error becomes

  $$\frac{4}{9}[1-\max(\frac{3}{4},\frac{1}{4})]+\frac{5}{9}[1-\max(\frac{1}{5},\frac{4}{5})]=\frac{2}{9}$$

  As a result, the change in classification error is

  $\triangle E(a_1) = 4/9 - 2/9 = 2/9.$

  After splitting on a_2, the classification error becomes

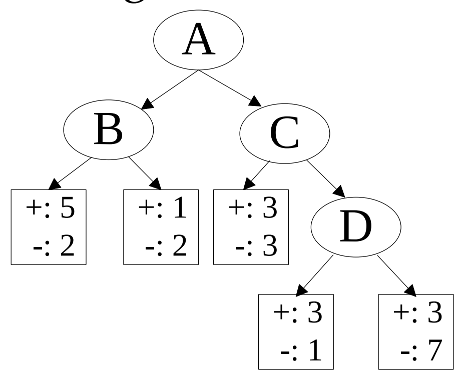
  $$\frac{5}{9}[1-\max(\frac{2}{5},\frac{3}{5})]+\frac{4}{9}[1-\max(\frac{2}{4},\frac{2}{4})]=\frac{4}{9}$$

  As a result,

  $\triangle E(a_2) = 4/9 - 4/9 = 0.$

Exercise

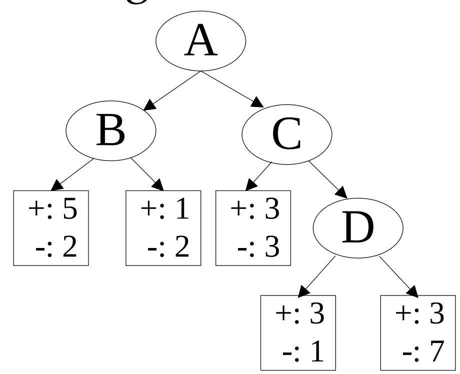
- Consider the following decision tree with four nodes A, B, C, D and five leaf nodes:



- What is the original Gini index value of the data set?
- What is the value of the penalty term for each leaf node if the generalization error is 0.5?
- Suppose a penalty term of 1 is assigned to each leaf node, estimate the generalization error if the sub-tree associated with node C is pruned and replaced with a leaf node.

Exercise

- Consider the following decision tree with four nodes A, B, C, D and five leaf nodes:



- What is the original Gini index value of the data set? **0.5**

$$1-\left(\frac{1}{2}\right)^2-\left(\frac{1}{2}\right)^2=0.5$$

- What is the value of the penalty term for each leaf node if the generalization error is 0.5? **1**

$$\frac{10+1\times 5}{30}=0.5$$

- Suppose a penalty term of 1 is assigned to each leaf node, estimate the generalization error if the sub-tree associated with node C is pruned and replaced with a leaf node. **0.5**

$$\frac{12+1\times 3}{30}=0.5$$

Linear Regression

- Gradient descent solution?

$$n^{-1}\sum_{i=1}^{n}(y_i - w_0 - w_1x_i) = 0$$

$$n^{-1}\sum_{i=1}^{n}x_i(y_i - w_0 - w_1x_i) = 0$$

$$Q(w_0, w_1) = \min_{w_0, w_1}\sum_{i=1}^{n}(y_i - w_0 - w_1x_i)^2$$

$$\partial Q(w_0, w_1)/\partial w_0 = 0$$

$$-2\sum_{i=1}^{n}(y_i - w_0 - w_1x_i) = 0$$

$$\partial Q(w_0, w_1)/\partial w_1 = 0$$

$$-2\sum_{i=1}^{n}x_i(y_i - w_0 - w_1x_i) = 0$$

Perceptron Learning Algorithm

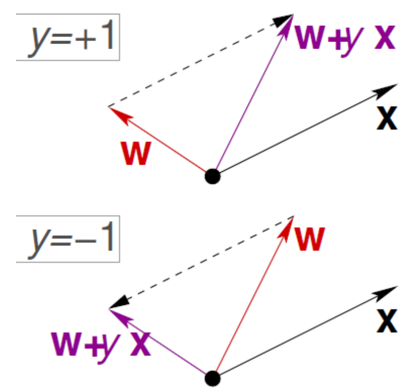
- Difficult: the set of $h(\mathbf{x})$ is of infinite size
- Idea: start from some initial weight vector $\mathbf{w}_{(0)}$, and "correct" its mistakes on D
- For t = 0, 1,…
  - find a mistake of $\mathbf{w}_{(t)}$ called $(\mathbf{x}_{n(t)}, y_{n(t)})$

    $$sign\left(\mathbf{w}_{(t)}^{\mathrm{T}}\mathbf{x}_{n(t)}\right) \neq y_{n(t)}$$

  - (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.

Exercise

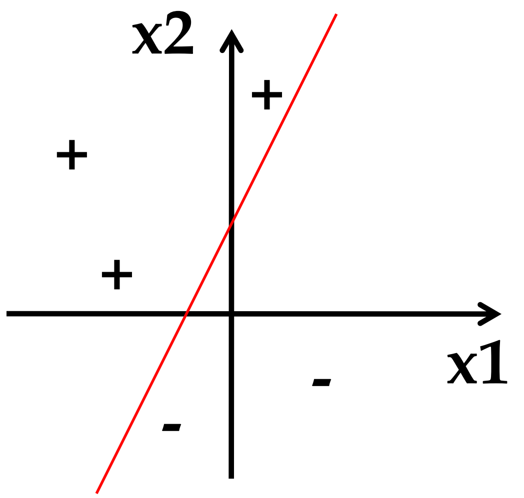
- What are the values of weights w0, w1, and w2 for the perceptron whose decision surface is illustrated in the Figure? Assume the surface crosses the x1 axis at -1, and the x2 axis at 2.

- Answer:

  w0 =

  w1 =

  w2 =

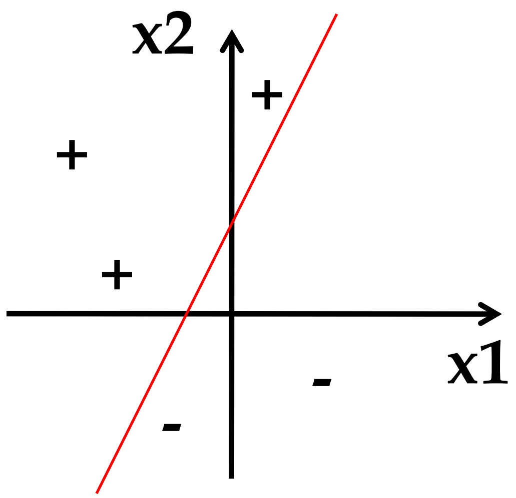


Exercise

- What are the values of weights w0, w1, and w2 for the perceptron whose decision surface is illustrated in the Figure? Assume the surface crosses the x1 axis at -1, and the x2 axis at 2.

  The surface crosses (-1, 0) and (0, 2)

  One surface: $-1 - x1 + 0.5 \cdot x2 = 0$



- Answer:

  $w0 = -1 \cdot C$

  $w1 = -1 \cdot C$

  $w2 = 0.5 \cdot C$

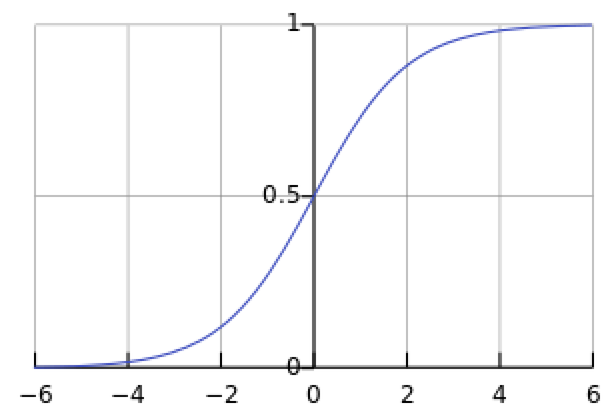
  (where $C > 0$)

Logistic Regression Model

- The logistic distribution constrains the estimated probabilities to lie between 0 and 1.
- The estimated probability $p(y=1|\mathbf{X})$ is:

$$p = \frac{1}{1+e^{-w_0-\sum_{j=1}^{d} w_j x_j}} = \frac{e^{w_0+\sum_{j=1}^{d} w_j x_j}}{1+e^{w_0+\sum_{j=1}^{d} w_j x_j}}$$

$$= \frac{1}{1+e^{-\tilde{\mathbf{w}}^{\mathrm{T}}\tilde{\mathbf{x}}}} = \frac{e^{\tilde{\mathbf{w}}^{\mathrm{T}}\tilde{\mathbf{x}}}}{1+e^{\tilde{\mathbf{w}}^{\mathrm{T}}\tilde{\mathbf{x}}}}$$



- if you let $w_0+\sum_{j=1}^{d} w_j x_j=0$, then $p = 0.5$
- as $w_0+\sum_{j=1}^{d} w_j x_j$ gets really big, p approaches 1 **PLA ?**
- as $w_0+\sum_{j=1}^{d} w_j x_j$ gets really small, p approaches 0

Logistic Regression Model

- The likelihood function is $\prod_{i=1}^{n}(p_i)^{y_i}(1-p_i)^{1-y_i}$
- We want to maximize the log likelihood:

$$L(\tilde{\mathbf{W}}) = \sum_{i=1}^{n}\left(y_i \log p_i + (1-y_i)\log(1-p_i)\right)$$

$$= \sum_{i=1}^{n}\left(y_i \log\frac{p_i}{1-p_i} + \log(1-p_i)\right)$$

$$= \sum_{i=1}^{n}\left(y_i \tilde{\mathbf{W}}^{\mathrm{T}}\tilde{\mathbf{X}}_i - \log(1+e^{\tilde{\mathbf{W}}^{\mathrm{T}}\tilde{\mathbf{X}}_i})\right)$$

$$\frac{\partial L(\tilde{\mathbf{W}})}{\partial \tilde{\mathbf{W}}} = \sum_{i=1}^{n}\left[\left(y_i - \frac{e^{\tilde{\mathbf{W}}^{\mathrm{T}}\tilde{\mathbf{X}}_i}}{1+e^{\tilde{\mathbf{W}}^{\mathrm{T}}\tilde{\mathbf{X}}_i}}\right)\tilde{\mathbf{X}}_i\right]$$

- It is equal to minimize the cost function

$$C(\tilde{\mathbf{W}}) = -L(\tilde{\mathbf{W}}) = -\sum_{i=1}^{n}\left(y_i \log p_i + (1-y_i)\log(1-p_i)\right)$$

Cross-entropy

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

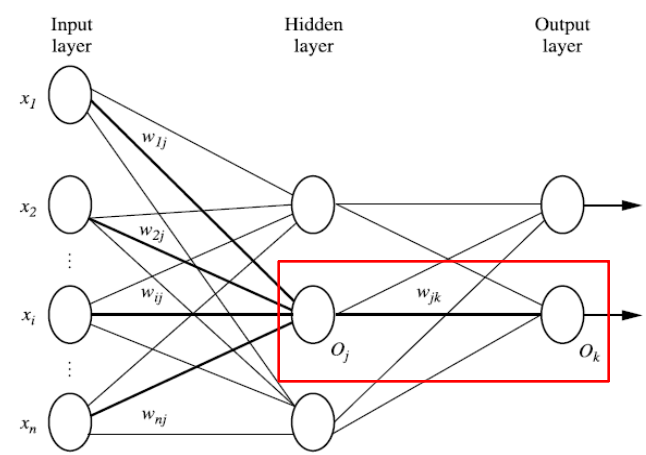
- For a unit k in the output layer, the error Err_k is computed by

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

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

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



Neural Network

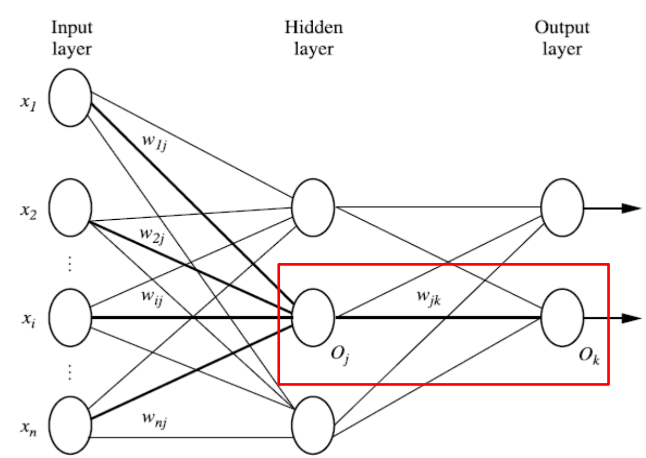
- Minimize the error of node O_k
- We define it as $E = \frac{1}{2}e^2 = \frac{1}{2}(T - O)^2$
- To adjust weight w_{jk}, we first calculate the partial derivation of E on w_{jk}

$$\frac{\partial E}{\partial w_{jk}} = \frac{\partial E}{\partial e} \times \frac{\partial e}{\partial O_k} \times \frac{\partial O_k}{\partial w_{jk}}$$

$$= -(e) \times \left(O_k(1 - O_k)\right) \times \left(O_j\right)$$

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

- and then use the "gradient decent"



Apriori Algorithm

- 自连接: 用 L_{k-1}自连接得到C_k
- 修剪: 一个k-项集，如果他的一个k-1项集（他的子集 ）不是频繁的，那他本身也不可能是频繁的。
- pseudo code:

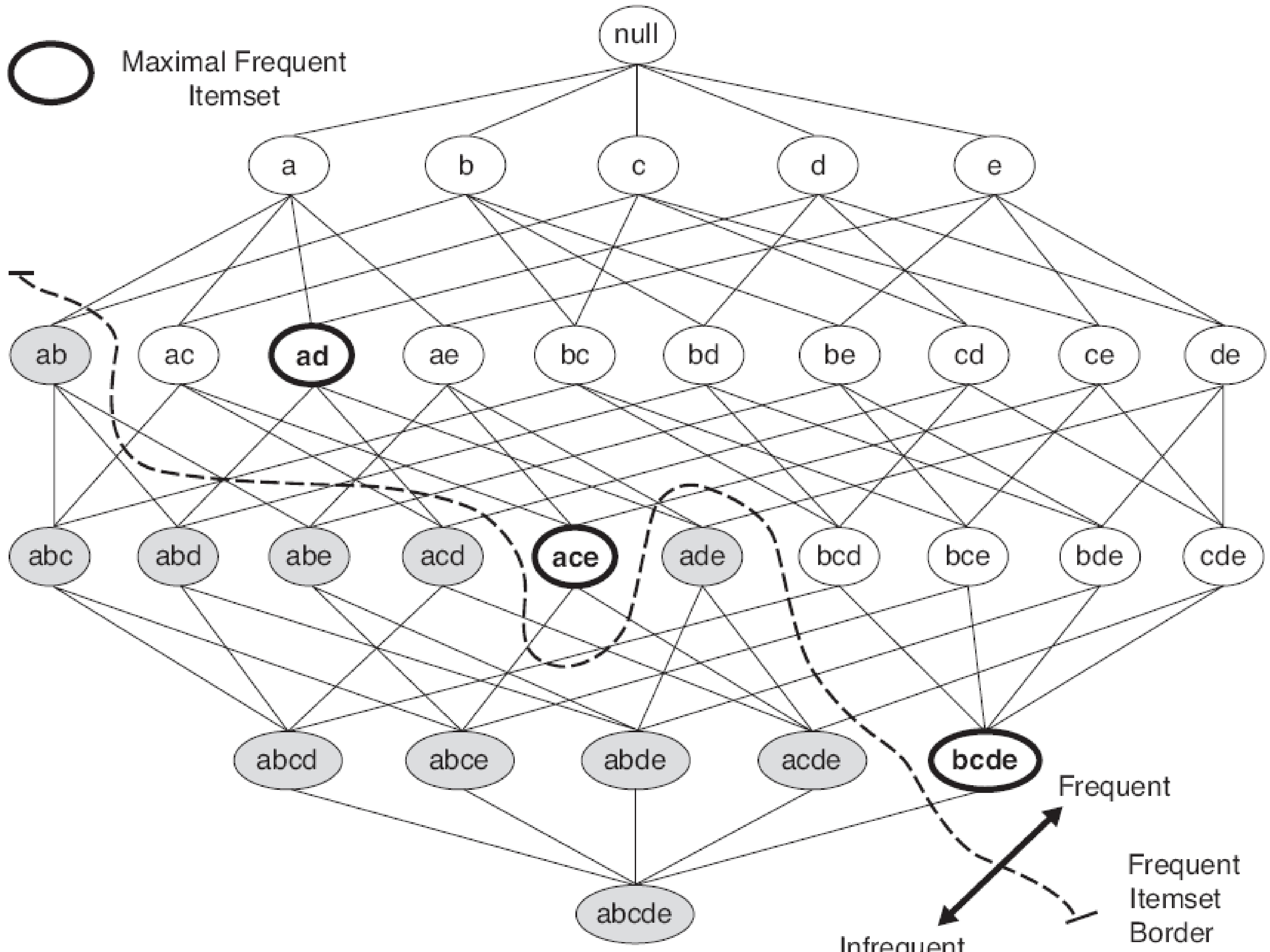
C_k: Candidate itemset of size k
L_k : frequent itemset of size k

L_1 = {frequent items};
for (k = 1; L_k !=∅; k++) **do begin**
  C_{k+1} = candidates generated from L_k;
  for each transaction t in database do
    increment the count of all candidates in C_{k+1} that are contained in t
  L_{k+1} = candidates in C_{k+1} with *minsup*
  end
return $\cup_k L_k$;

Maximal Frequent Itemsets

- A maximal frequent itemset is defined as a frequent itemset for which none of its immediate supersets are frequent.

Maximal Frequent Itemsets



Closed Frequent Itemsets

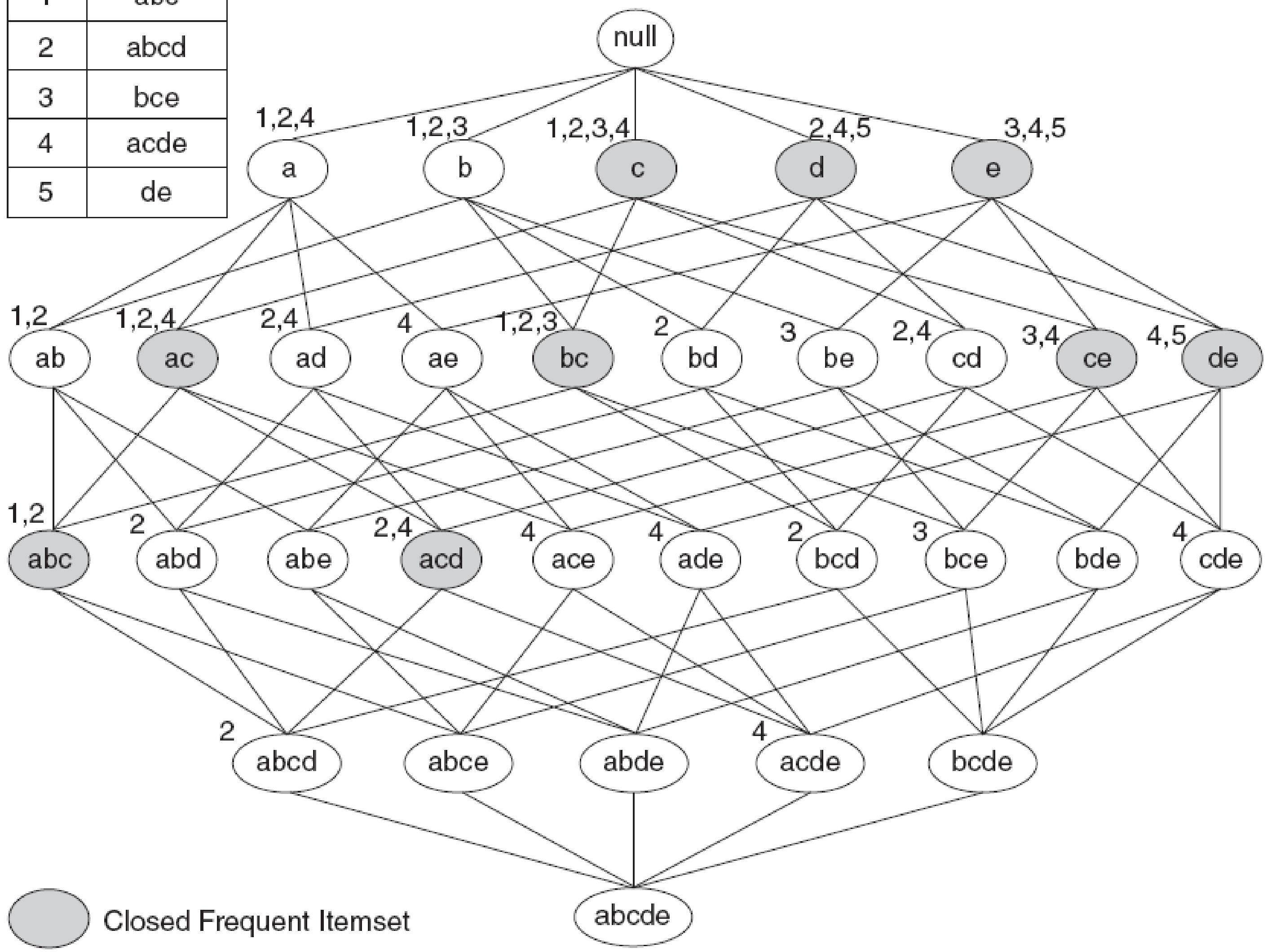
- An itemset X is closed if none of its immediate supersets has exactly the same support count as X.

- In other words, X is not closed if at least one of its immediate supersets has the same support count as X.

Closed Frequent Itemsets

TID	Items
1	abc
2	abcd
3	bce
4	acde
5	de

minsup = 40%



Partitional Clustering

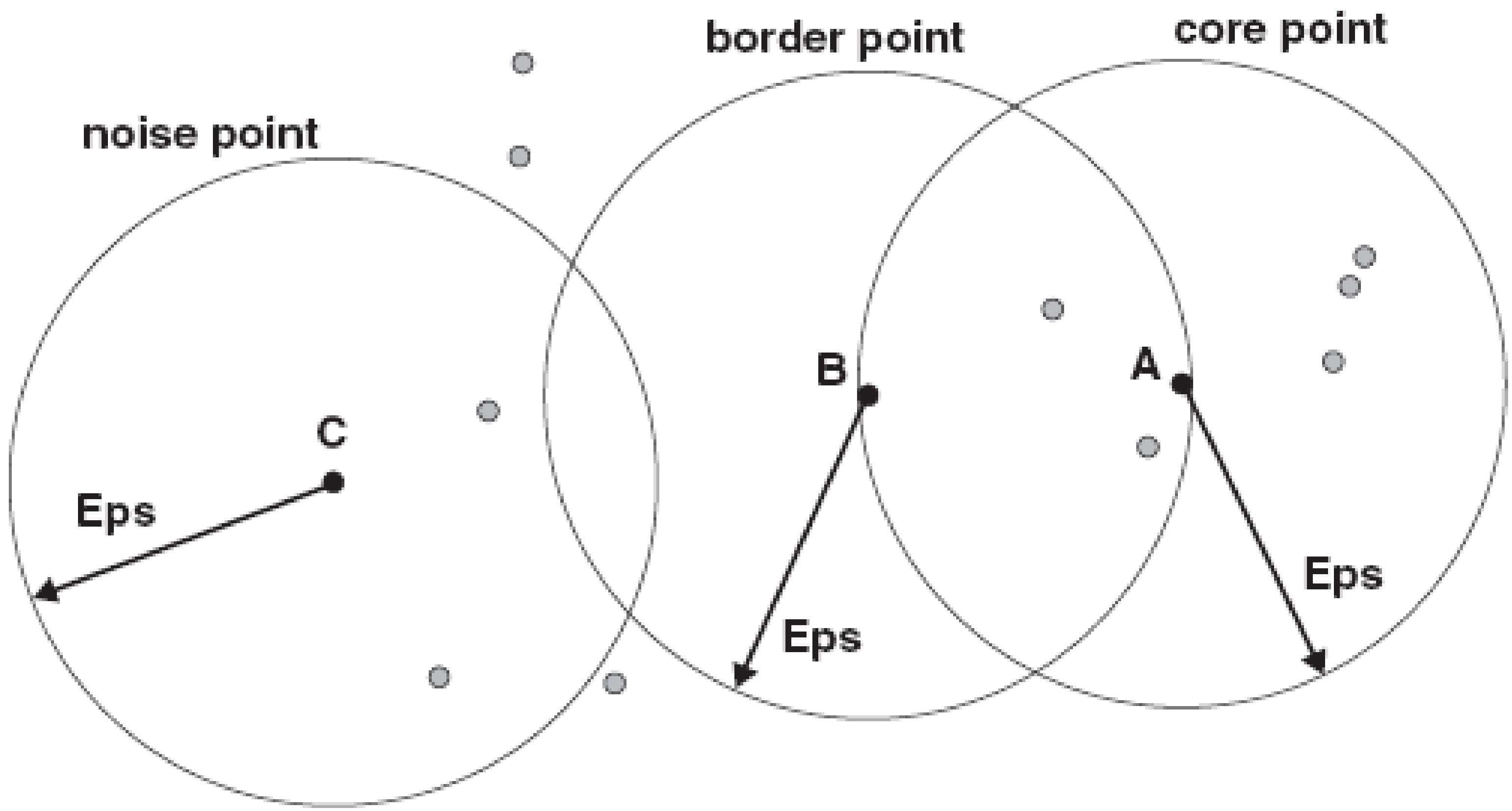
- *k*-Means: Repeat…
  - Choose *k* arbitrary 'centroids'
  - Assign each document to nearest centroid
  - Re-compute centroids

- **Example of *k*-Means (划分法)**

DBSCAN

- We need to classify a point as being
  - In the interior of a dense region (a core point, 核心点).
  - At the edge of a dense region (a border point, 边界点)
  - In a sparsely occupied region (a noise or background point, 噪音点).
- The concepts of core, border and noise points are illustrated as follows.

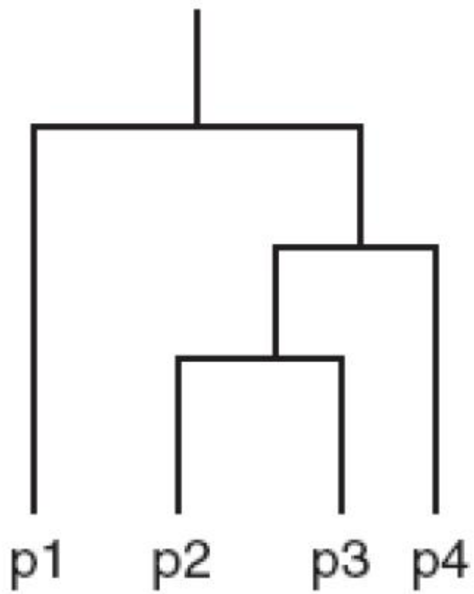
DBSCAN



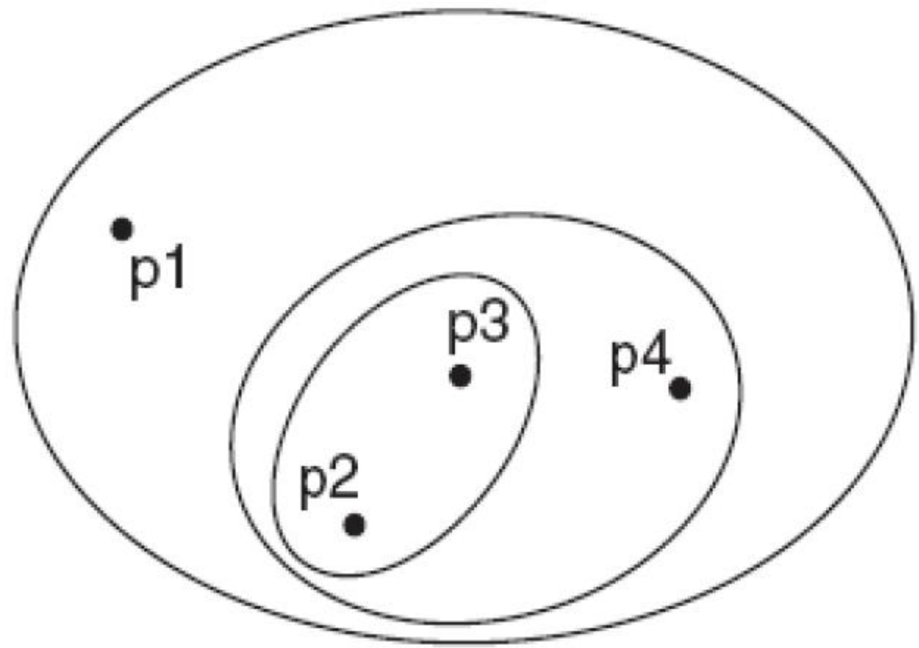
Hierarchical Clustering

- A hierarchical clustering is often displayed graphically using a tree-like diagram called the dendrogram (树状图).
- The dendrogram displays both
  - the cluster-subcluster relationships and
  - the order in which the clusters are merged (agglomerative) or split (divisive).
- For sets of 2-D points, a hierarchical clustering can also be graphically represented using a nested cluster diagram.

Hierarchical Clustering



(a) Dendrogram.



(b) Nested cluster diagram.

Hierarchical Clustering

- Different definitions of cluster distance leads to different versions of hierarchical clustering.

- These versions include
  - Single link (单连接) or MIN
  - Complete link (全连接) or MAX
  - Group average (组平均)

Single Link

- We now consider the single link or MIN version of hierarchical clustering.
- In this case, the distance of two clusters is defined as the minimum of the distance between any two points in the two different clusters.
- This technique is good at handling non-elliptical (非球状的) shapes.

Complete Link

- We now consider the complete link or MAX version of hierarchical clustering.
- In this case, the distance of two clusters is defined as the maximum of the distance between any two points in the two different clusters.
- Complete link tends to produce clusters with globular (球状) shapes.