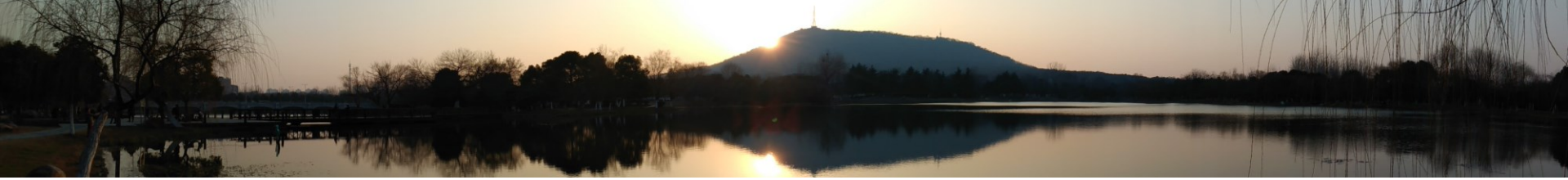


模式识别

中国科学技术大学 汪增福

- 第一章 绪论
- 第二章 统计模式识别中的几何方法
- 第三章 统计模式识别中的概率方法
- 第四章 分类器的错误率
- 第五章 统计模式识别中的聚类方法
- 第六章 结构模式识别中的句法方法
- 第七章 总结

第五章 统计模式识别中的聚类方法

- 本章主要内容

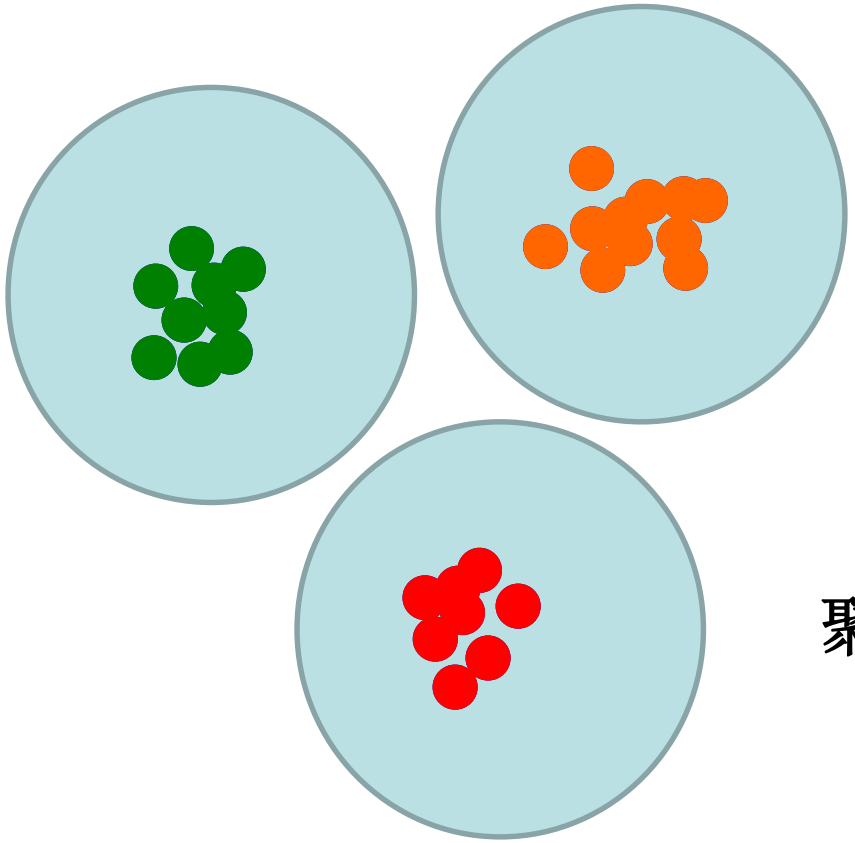
主要讨论类别属性未知情况下训练样本的分类问题。

- 问题概述
- 聚类准则
- 基于分裂的聚类算法
- 基于合并的聚类算法
- 动态聚类算法
- 近邻函数值准则聚类算法
- 最小张树聚类算法

§ 5.1 问题概述

- 有监督分类
- 无监督分类　无教师分类

如何分类？ ➡ 利用样本在几何上所表现出的相似性

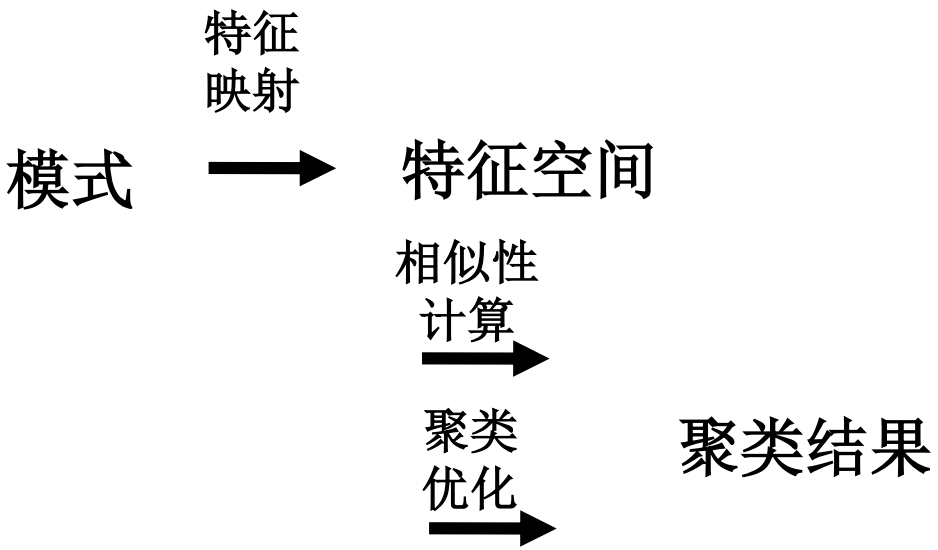


聚类分析

§5.1 问题概述

聚类分析

聚类方法：对于给定特征的两个样本，依照相似性测度计算其相似性，若相似性的度量值大于给定的阈值，则判它们属于同一个类别，否则判它们属于不同的类别。



§5.1 问题概述

聚类分析

两个要素

- 特征
- 相似性测度

一个准则

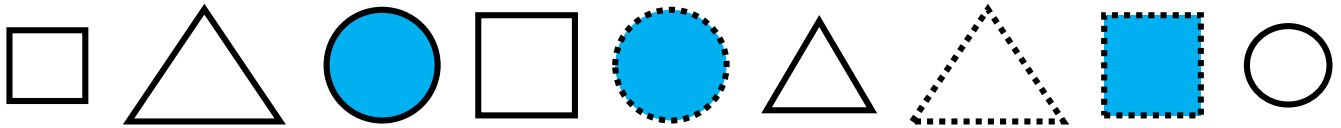
聚类准则

特征选择是基础
相似度计算是依据
聚类优化是关键

} 相辅相成的三位一体

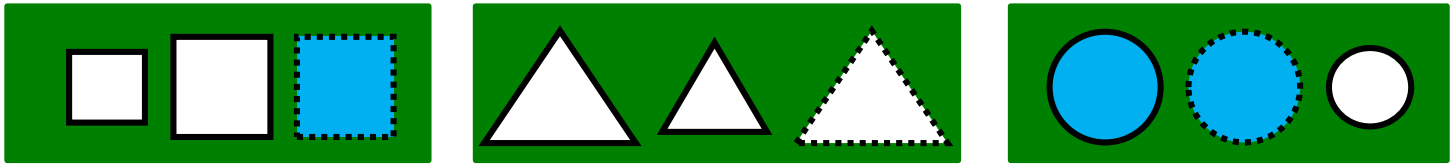
§ 5.1 问题概述

特征选择

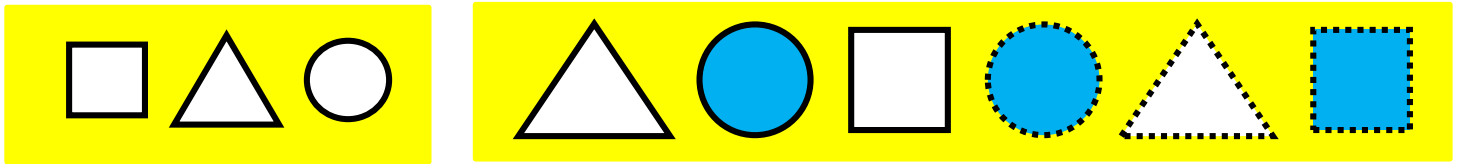


所选特征

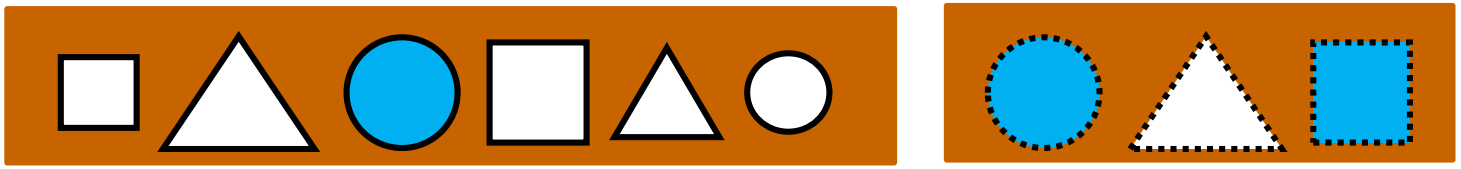
形状



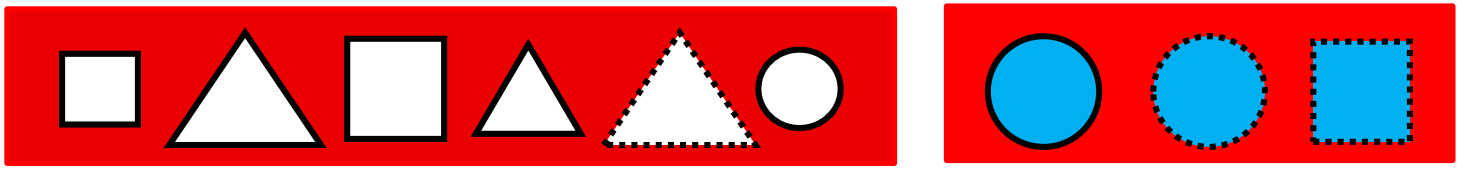
尺寸



线种



颜色

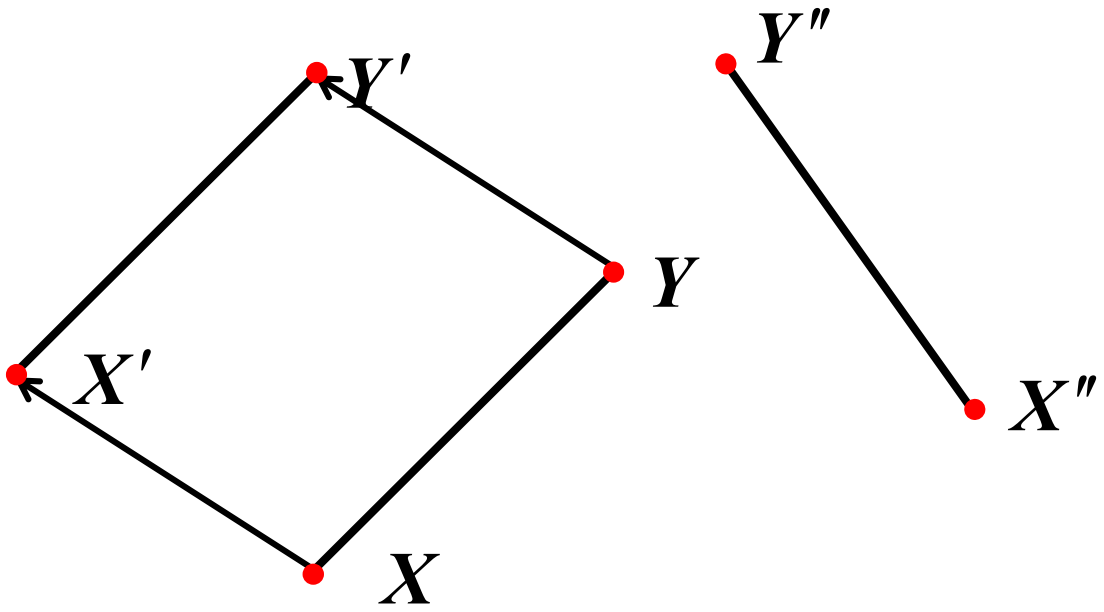


§5.1 问题概述

相似性测度

不变特性 { 平移 / 旋转 / 尺度

距离测度：欧氏距离



§ 5.1 问题概述

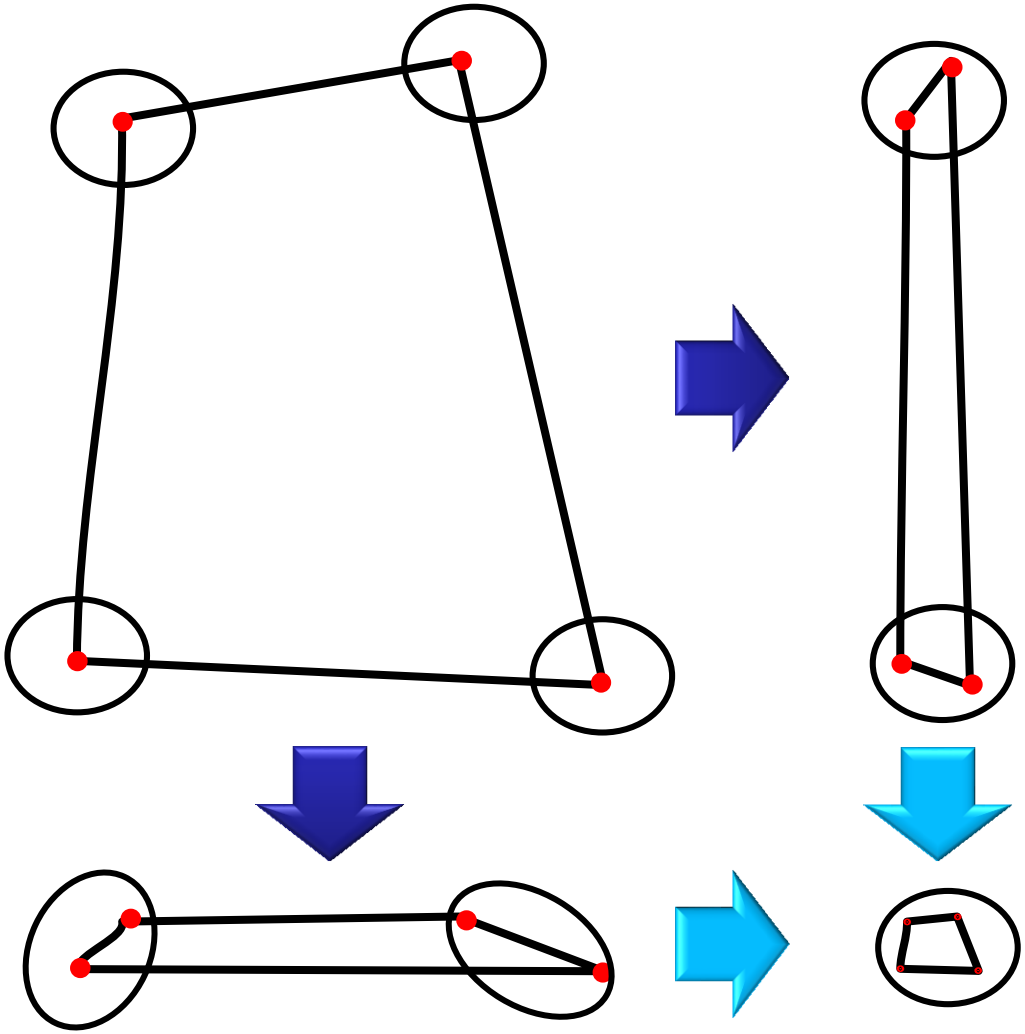
量纲

量纲（physical dimension）是指物理量的基本属性。物理学的研究可以定量地描述各种物理现象，描述中所采用的各类物理量之间有着密切的关系，即它们之间具有确定的函数关系。为了准确地描述这些关系，物理量可分为基本量和导出量，一切导出量均可从基本量中导出，由此建立了整个物理量之间函数关系，这种关系通常称为量制。以给定量制中基本量量纲的幂的乘积表示某量量纲的表达式，称为量纲式或量纲积。它定性地表达了导出量与基本量的关系，对于基本量而言，其量纲为其自身。在物理学发展的历史上，先后曾建立过各种不同的量制：CGS量制、静电量制、高斯量制等。1971年后，国际上普遍采用了国际单位制（简称SI），选定了由7个基本量构成的量制，导出量均可用这7个基本量导出。7个基本量的量纲分别用长度L、质量M、时间T、电流I、温度Θ、物质的量N和光强度J表示。

量纲是表征物理量的性质，单位是表征物理量大小或数量的标准。

§5.1 问题概述

量纲和单位对分类的影响



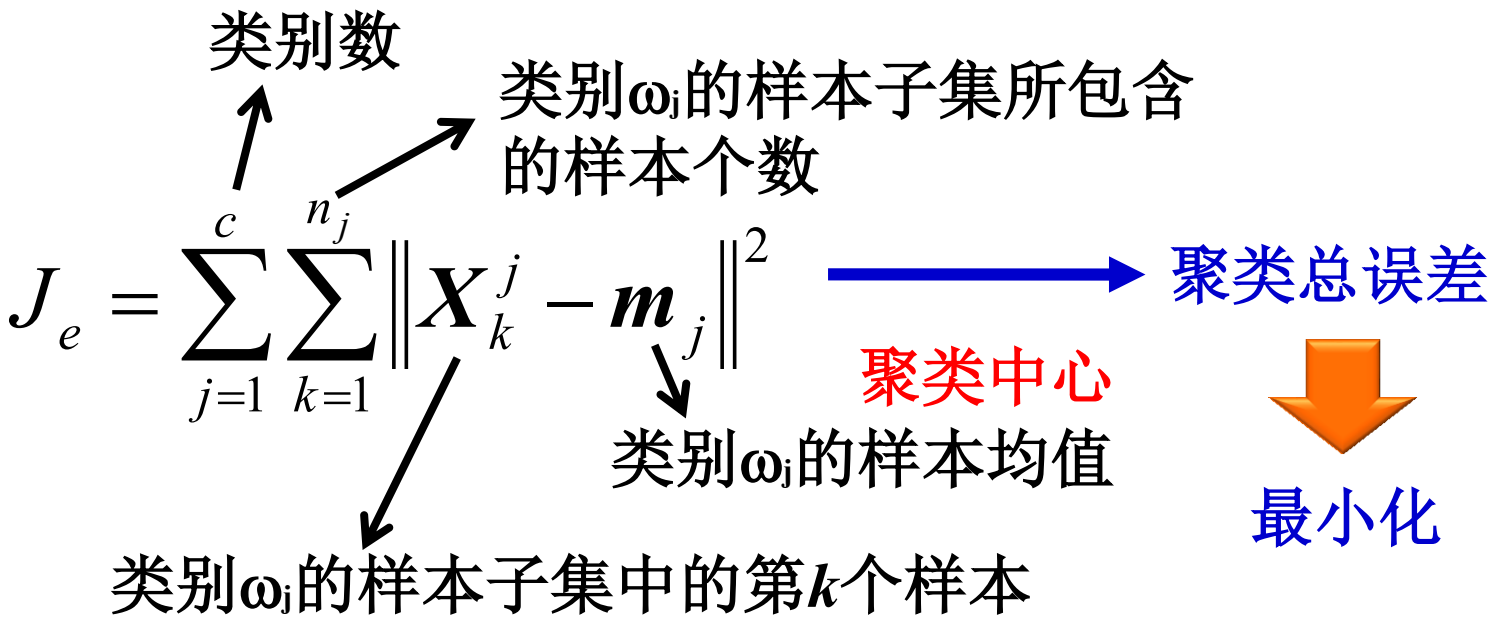
§5.2 聚类准则

好的聚类准则的标准

- 把属于同一个类别的样本聚在一起
- 把不属于同一个类别的样本分离开

常用聚类准则函数

误差平方和准则函数J_e



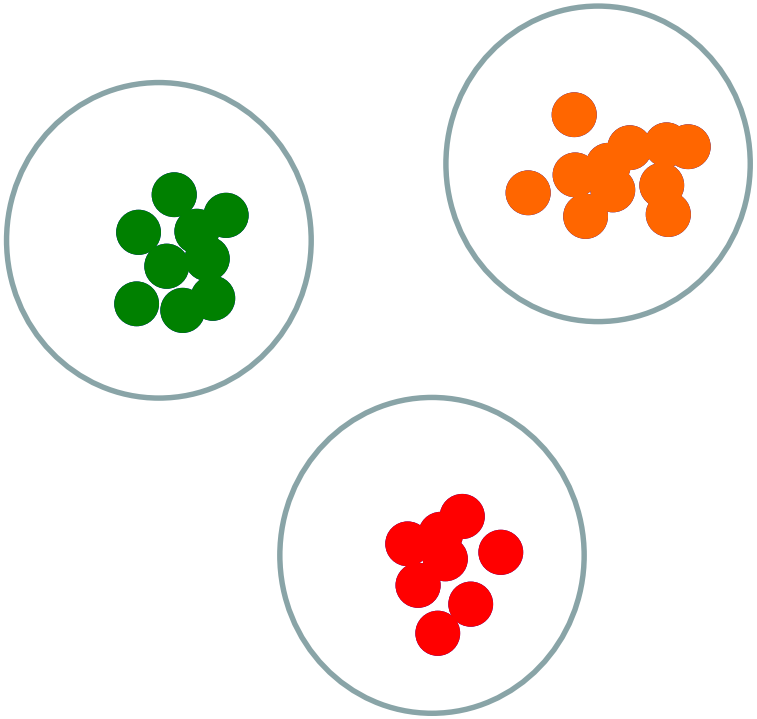
§5.2 聚类准则

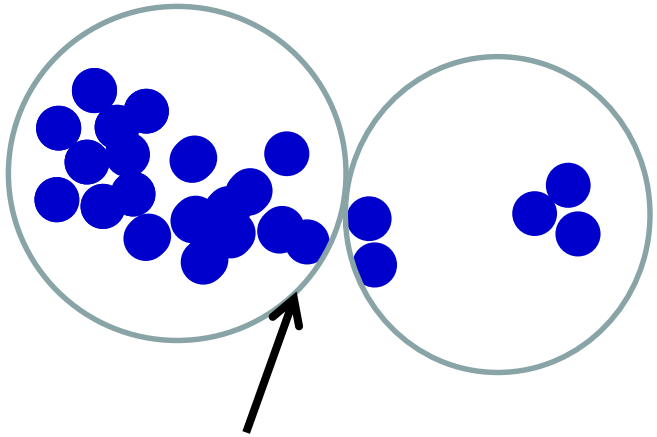
常用聚类准则函数

误差平方和准则函数J_e

适用范围：

1. 同类样本分布相对密集；
2. 各类别所包含样本数相差不大、类间距离较大。





样本数较多的类别发生分裂

§5.2 聚类准则

常用聚类准则函数

加权平均平方距离和准则函数J_l

类别ω_j发生的先验概率

$$J_l = \sum_{j=1}^{c} P_j D_j^*$$

聚类总误差

最小化

类别ω_j的类内平均平方距离和

$$D_j^* = \frac{1}{C_{n_j}^2} \sum_{\substack{k.l=1 \\ l<k}}^{n_j} \left\| \boldsymbol{X}_j^k - \boldsymbol{X}_j^l \right\|^2$$

$$J_l = \frac{1}{n} \sum_{j=1}^{c} n_j D_j^*$$

§5.2 聚类准则

常用聚类准则函数

类间平方距离和准则函数J_b

全体样本的均值向量

$$J_{b1}=\sum_{j=1}^{c}(\boldsymbol{m}_j-\boldsymbol{m})^T(\boldsymbol{m}_j-\boldsymbol{m})$$

总的类间距离

$$J_{b2}=\sum_{j=1}^{c}P_j(\boldsymbol{m}_j-\boldsymbol{m})^T(\boldsymbol{m}_j-\boldsymbol{m})$$

$$=\frac{1}{n}\sum_{j=1}^{c}n_j(\boldsymbol{m}_j-\boldsymbol{m})^T(\boldsymbol{m}_j-\boldsymbol{m})$$

最大化

§ 5.2 聚类准则

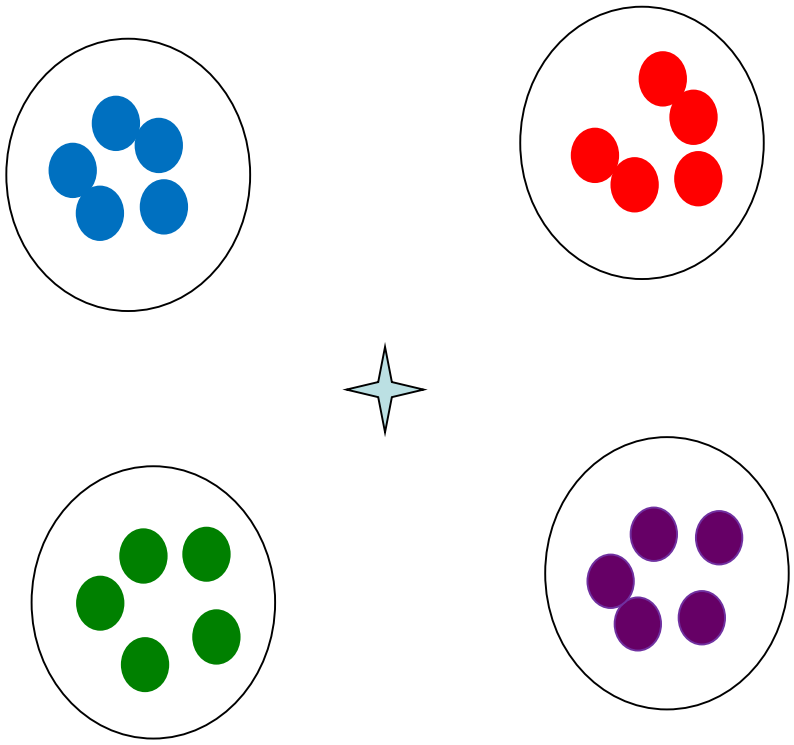
常用聚类准则函数

平均类间平方距离准则函数J_b

$$J_{b3}=\frac{1}{C_c^2}\sum_{\substack{k.l=1\\ l<k}}^{c}(\boldsymbol{m}_k-\boldsymbol{m}_l)^T(\boldsymbol{m}_k-\boldsymbol{m}_l)$$

平均类间平方距离

最大化



§5.2 聚类准则

常用聚类准则函数

离散度准则函数

类内离散度矩阵

$$\boldsymbol{S}_j = \frac{1}{n_j}\sum_{k=1}^{n_j}(\boldsymbol{X}_k^j - \boldsymbol{m}_j)(\boldsymbol{X}_k^j - \boldsymbol{m}_j)^T \qquad j = 1,2,...c$$

总的类内离散度矩阵

$$\boldsymbol{S}_w = \sum_{j=1}^{c} P_j \boldsymbol{S}_j$$

类间离散度矩阵

$$\boldsymbol{S}_b = \sum_{j=1}^{c} P_j(\boldsymbol{m}_j - \boldsymbol{m})(\boldsymbol{m}_j - \boldsymbol{m})^T$$

总的离散度矩阵

$$\boldsymbol{S}_t = \frac{1}{n}\sum_{k=1}^{n}(\boldsymbol{X}_k - \boldsymbol{m})(\boldsymbol{X}_k - \boldsymbol{m})^T$$

三者之间有何关系？

§ 5.2 聚类准则

常用聚类准则函数

离散度准则函数

三个离散度矩阵之间的关系　　$\boldsymbol{S}_t = \boldsymbol{S}_w + \boldsymbol{S}_b$

$$\boldsymbol{S}_t = \frac{1}{n}\sum_{k=1}^{n}(\boldsymbol{X}_k - \boldsymbol{m})(\boldsymbol{X}_k - \boldsymbol{m})^T$$

$$= \frac{1}{n}\sum_{j=1}^{c}\sum_{k=1}^{n_j}(\boldsymbol{X}_k^j - \boldsymbol{m})(\boldsymbol{X}_k^j - \boldsymbol{m})^T$$

$$= \sum_{j=1}^{c}\frac{n_j}{n}\frac{1}{n_j}\sum_{k=1}^{n_j}(\boldsymbol{X}_k^j - \boldsymbol{m})(\boldsymbol{X}_k^j - \boldsymbol{m})^T$$

$$= \sum_{j=1}^{c}P_j[\frac{1}{n_j}\sum_{k=1}^{n_j}(\boldsymbol{X}_k^j - \boldsymbol{m})(\boldsymbol{X}_k^j - \boldsymbol{m})^T]$$

§5.2 聚类准则

常用聚类准则函数

离散度准则函数

三个离散度矩阵之间的关系 $\boldsymbol{S}_t = \boldsymbol{S}_w + \boldsymbol{S}_b$

$$\boldsymbol{S}_t = \sum_{j=1}^{c} P_j [\frac{1}{n_j} \sum_{k=1}^{n_j} (\boldsymbol{X}_k^j - \boldsymbol{m})(\boldsymbol{X}_k^j - \boldsymbol{m})^T]$$

$$= \sum_{j=1}^{c} P_j [\frac{1}{n_j} \sum_{k=1}^{n_j} (\boldsymbol{X}_k^j - \boldsymbol{m}_j - (\boldsymbol{m} - \boldsymbol{m}_j))(\boldsymbol{X}_k^j - \boldsymbol{m}_j - (\boldsymbol{m} - \boldsymbol{m}_j))^T]$$

$$= \sum_{j=1}^{c} P_j [\frac{1}{n_j} \sum_{k=1}^{n_j} (\boldsymbol{X}_k^j - \boldsymbol{m}_j)(\boldsymbol{X}_k^j - \boldsymbol{m}_j)^T + (\boldsymbol{m}_j - \boldsymbol{m})(\boldsymbol{m}_j - \boldsymbol{m})^T]$$

$$= \sum_{j=1}^{c} P_j S_j + \sum_{j=1}^{c} P_j (\boldsymbol{m}_j - \boldsymbol{m})(\boldsymbol{m}_j - \boldsymbol{m})^T$$

$$= \boldsymbol{S}_w + \boldsymbol{S}_b$$

§5.2 聚类准则

常用聚类准则函数

离散度准则函数

三个离散度矩阵之间的关系

仅和样本全体的分布有关，
而和聚类结果无关。

此消彼长
相互依存
相互制约

$$\boldsymbol{S}_t = \boldsymbol{S}_w + \boldsymbol{S}_b$$

与聚类结果直接关联
两者的总和保持不变
不便直接进行评估

§5.2 聚类准则

常用聚类准则函数

离散度准则函数：基于迹的准则函数

方阵的迹:方阵主对角线元素之和。 方阵A的迹记为 $T_r[A]$

$$T_r[\boldsymbol{S}_j]=\frac{1}{n_j}\sum_{k=1}^{n_j}\left\|\boldsymbol{X}_k^j-\boldsymbol{m}_j\right\|^2$$

沿各坐标轴方向的分量方差之和

其值越小, 类内样本聚集程度越高

$T_r[\boldsymbol{S}_w]$ 其值反映了同类样本的聚集程度

$T_r[\boldsymbol{S}_b]$ 其值反映了不同类样本的分离程度

优化准则函数	优化目标
$J_{tw}=T_r[\boldsymbol{S}_w]$	最小化
$J_{tb}=T_r[\boldsymbol{S}_b]$	最大化

§5.2 聚类准则

常用聚类准则函数

离散度准则函数：基于行列式的准则函数

优化准则函数　　优化目标

$J_{dw} = |\boldsymbol{S}_w|$　　最小化

$J_{db} = |\boldsymbol{S}_b|$　　最大化

$$J_{t1} = T_r[\boldsymbol{S}_w^{-1}\boldsymbol{S}_b] \qquad J_{d1} = |\boldsymbol{S}_w^{-1}\boldsymbol{S}_b|$$

$$J_{t2} = T_r[\boldsymbol{S}_w^{-1}\boldsymbol{S}_t] \qquad J_{d2} = |\boldsymbol{S}_w^{-1}\boldsymbol{S}_t|$$

§5.2 聚类准则

常用聚类准则函数

离散度准则函数：基于特征值的准则函数

$$\boldsymbol{AX} = \lambda \boldsymbol{X}$$

对非奇异的d阶方阵而言，有：

1. A的d个特征值之和等于A的迹
2. A的d个特征值之积等于A的行列式

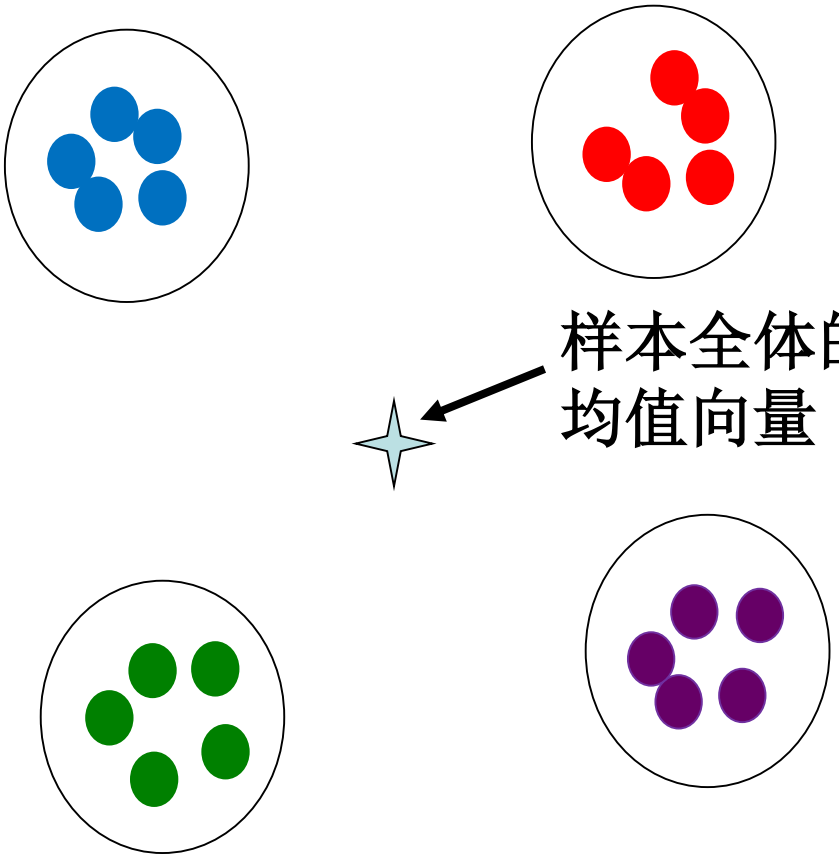
若非奇异的d阶方阵$\boldsymbol{S}_w^{-1}\boldsymbol{S}_b$的特征值依次为$\lambda_1, \lambda_2, \ldots, \lambda_d$，则

$$J_{t1} = \sum_{i=1}^{d} \lambda_i \qquad J_{t2} = \sum_{i=1}^{d} (1+\lambda_i) \qquad J_{d1} = \prod_{i=1}^{d} \lambda_i \qquad J_{d2} = \prod_{i=1}^{d} (1+\lambda_i)$$

均可作为准则函数。

§ 5.2 聚类准则

影响聚类效果的主要因素：类别数的指定



$$\boldsymbol{S}_j = \frac{1}{n_j}\sum_{k=1}^{n_j}(\boldsymbol{X}_k^j - \boldsymbol{m}_j)(\boldsymbol{X}_k^j - \boldsymbol{m}_j)^T$$

$$\boldsymbol{S}_w = \sum_{j=1}^{c} P_j \boldsymbol{S}_j$$

$$\boldsymbol{S}_b = \sum_{j=1}^{c} P_j(\boldsymbol{m}_j - \boldsymbol{m})(\boldsymbol{m}_j - \boldsymbol{m})^T$$

§5.3 基于分裂的聚类算法

聚类算法通常表现为一个迭代过程。

影响聚类算法性能优劣的几个因素

- 聚类中心的选择与更新
- 聚类策略和聚类准则的选择
- 控制阈值和类别数的设置

聚类算法分类

- 增类聚类算法
- 减类聚类算法
- 动态聚类算法

§5.3 基于分裂的聚类算法

简单增类聚类算法

Step1. 读入训练样本集 $\mathscr{X} = \{\boldsymbol{X}_1, \boldsymbol{X}_2, ..., \boldsymbol{X}_N\}$；

Step2. 执行初始化操作：

置第一个聚类中心 $\boldsymbol{Z}[1] = \boldsymbol{X}_1$ ；

置下一个聚类中心 $\boldsymbol{Z}_{next} =$ 空；

置阈值$\boldsymbol{T}=$ 足够大的值；

初始化样本处理状态数组 $\boldsymbol{P}_{ed}(i) = 0, \quad i = 1,2,...,N$；

置类别计数器 $C_s = 1$；

置已处理样本计算器 $n = 0$ 。

§5.3 基于分裂的聚类算法

简单增类聚类算法（续1）

Step3. *for* $X_i \in \mathcal{X}, i = 1,2,...N$ {

if $P_{ed}(i) == 0$ {

计算 $d(X_i, Z[j]), j = 1,2,...,C_s$;

if $\underset{j}{Minimize}\ d(X_i, Z[j]) \le T$ {

根据最近邻聚类准则将 X_i 划分到距离最近的类别中，并置 $P_{ed}(i) = 1$ 和 $n++$ 。

}

else {

if $Z_{next} ==$ 空 {

$Z_{next} = X_i$ ，去**Step4**。

}

}

}

}

§5.3 基于分裂的聚类算法

简单增类聚类算法（续2）

Step4. *if* $n < N$ *{*

$Z[++C_s] = Z_{zext}$ ；

Z_{next} = 空 ；

返回**step3**。

}

else {

考察聚类结果；

若满意，则算法结果；

否则，修改初始化参数，返回**step3**。

}

§5.3 基于分裂的聚类算法

简单增类聚类算法小结

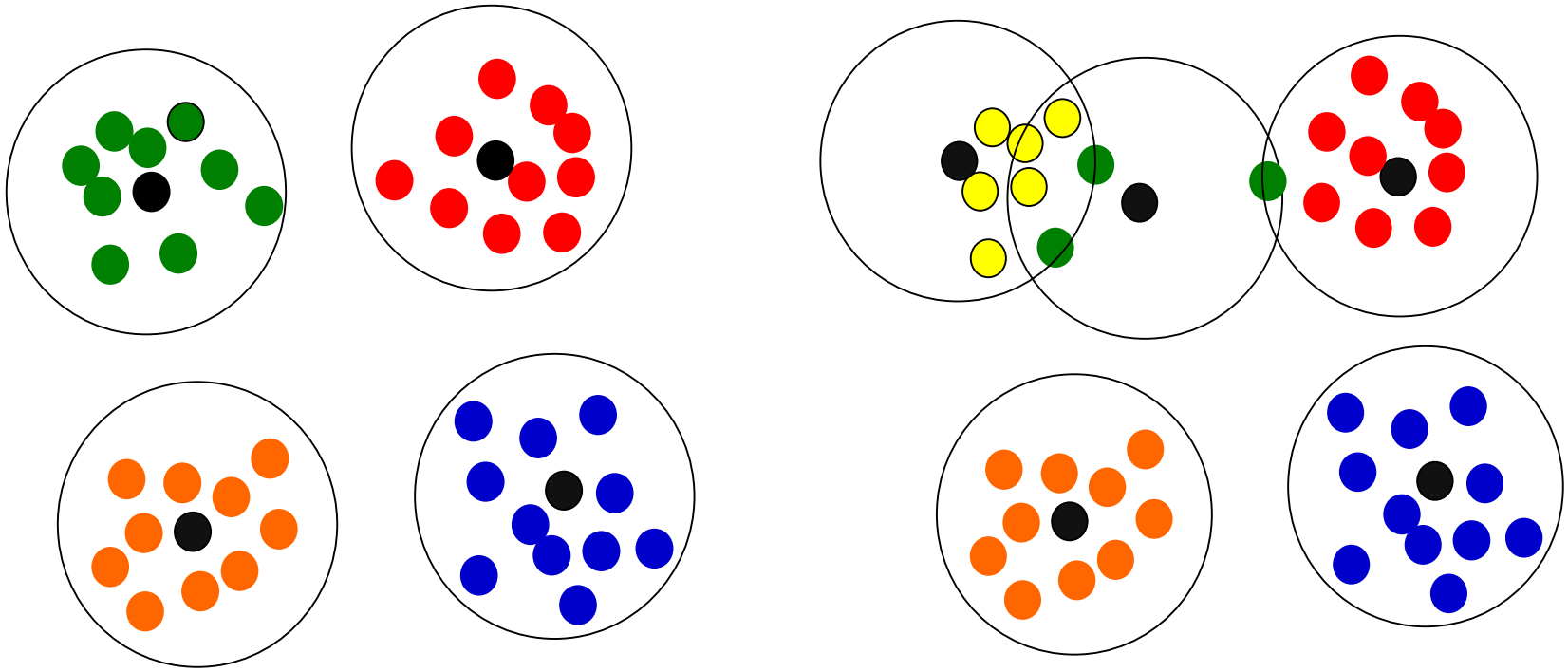
- 算法简单
- 聚类结果好坏依赖于聚类中心和判决阈值的选择
- 聚类中心的选择具有很大的随机性
- 距离阈值的设定也存在不确定性

结论

- 成员相同、排序不同的样本集合在相同距离阈值的设定条件下可能给出不同的聚类结果；
- 成员相同、排序也相同的样本集合在不同距离阈值的设定条件下可能给出不同的聚类结果。

§5.3 基于分裂的聚类算法

简单增类聚类算法小结



§5.3 基于分裂的聚类算法

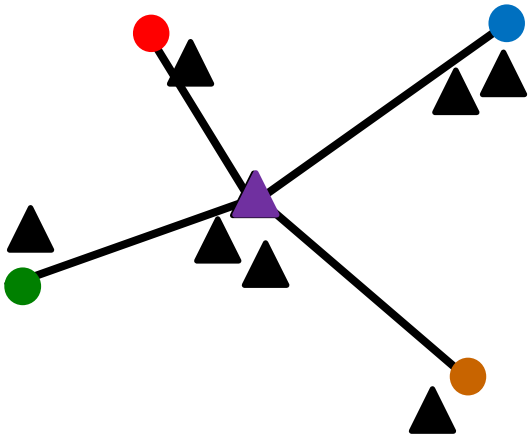
改进的简单增类聚类算法

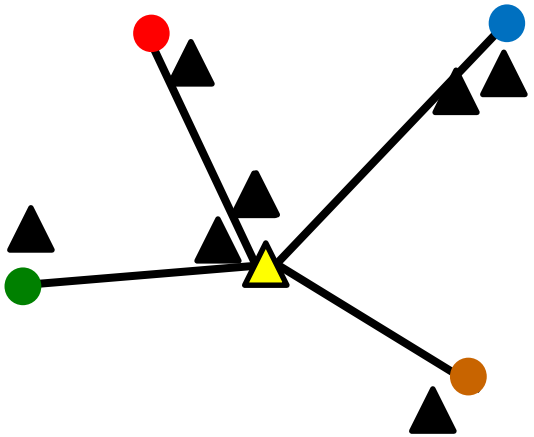
改进之处：聚类中心的选择

- ✓ 第一个聚类中心任选；
- ✓其后，按照最大最小距离准则选择下一个聚类中心直到满足条件的聚类中心被选出为止。

最大最小距离

$$d_{MM} = \underset{i}{Maximize}\ \{\underset{j}{Minimize}\ d(\boldsymbol{X}_i, \boldsymbol{Z}[j])\} \qquad i = 1,2,\ldots,N \qquad j = 1,2,\ldots,C_s$$





§5.3 基于分裂的聚类算法

改进的简单增类聚类算法

Step1. 读入训练样本集 $\mathcal{X} = \{\boldsymbol{X}_1, \boldsymbol{X}_2, ..., \boldsymbol{X}_N\}$；

Step2. 执行初始化操作：

置第一个聚类中心 $\boldsymbol{Z}[1] = \boldsymbol{X}_1$ ；

置类别计数器 $C_s = 1$；

设置控制参数θ, 0< θ <1；

$\boldsymbol{Z}[2] = \underset{i}{Maximize}\ d(\boldsymbol{X}_i, \boldsymbol{Z}[1])$ ， $C_s{+}{+}$ ；

$d_{12} = d(\boldsymbol{Z}[1], \boldsymbol{Z}[2])$ 。

§5.3 基于分裂的聚类算法

改进的简单增类聚类算法（续1）

Step3. 寻找新的聚类中心直至终止条件被满足。

Step3-1. 计算 $d_{MM} = \underset{i}{Maximize}\ \{\underset{j}{Minimize}\ d(\boldsymbol{X}_i, \boldsymbol{Z}[j])\}$

$i = 1,2,\ldots,N \quad j = 1,2,\ldots,C_s$

Step3-2. 若 $d_{MM} > \theta d_{12}$ ，则 $\boldsymbol{Z}[++C_s] = \boldsymbol{X}_{d_{MM}}$ ，返回**Step3-1**。

否则，继续下一步骤。

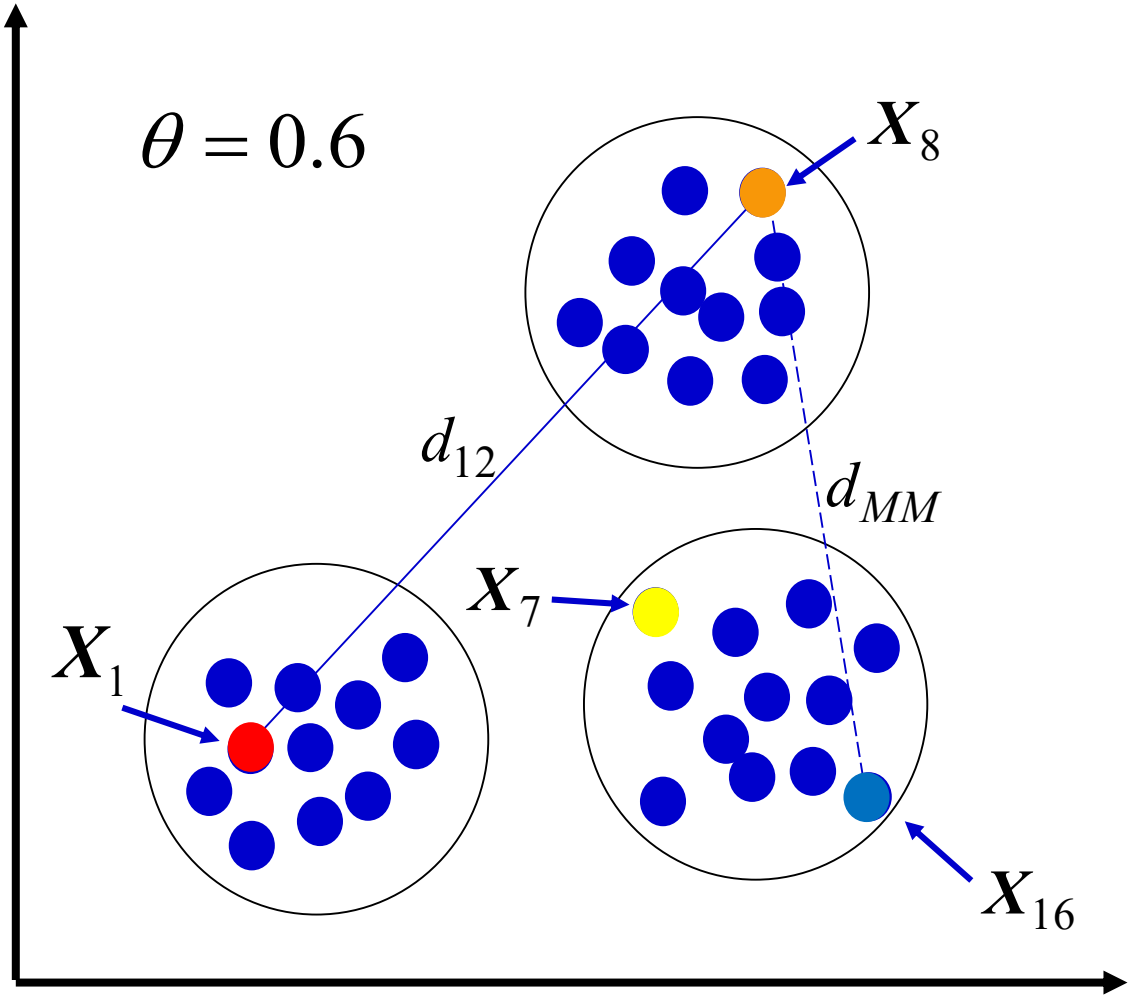
Step4. 根据最近邻准则分类剩余的 样本。

Step5. 考察聚类结果。若满意，则算法结果；

否则，返回**Step2**，修改参数,重新聚类。

§5.3 基于分裂的聚类算法

改进的简单增类聚类算法：举例



§5.4 基于合并的聚类算法

特点：类别数由多到少；

终止条件：聚类后的最小类间距离是否大于给定的阈值。

减类聚类算法

Step1. 读入训练样本集 $\mathcal{X}=\{\boldsymbol{X}_1,\boldsymbol{X}_2,...,\boldsymbol{X}_N\}$；

Step2. 执行初始化操作：

置类别计数器 $C_s=N$；

置迭代计数器 $k=0$ ；

设置类间距离阈值T=足够大的距离值；

For i =1 to C_s**{**

$\mathcal{X}_i^k=\{\boldsymbol{X}_i\}$

}

§5.4 基于合并的聚类算法

减类聚类算法（续1）

Step3. 构造矩阵 $D^k=[D_{pq}]_{C_s\times C_s}$；其中，$D_{pq}$为现行第***p***和第***q***两个类别的类间距离。

Step4. 求 $D^k=[D_{pq}]_{C_s\times C_s}$ 的最小元素 $D_{ij}=\underset{p,q}{Minimize}\ D_{pq}$；

若 $D_{ij}>T$，则算法结果；

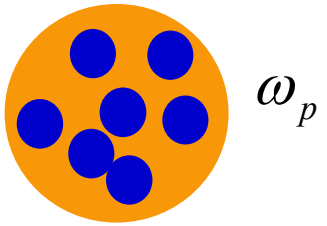
否则，将 D_{ij} 对应的第***i*** 和第***j*** 两个类别合并，并整理合并后的分类结果，使重新排序后的类别序号仍是连续的：$\mathcal{X}_1^{k+1}$，$\mathcal{X}_2^{k+1}$，…，$\mathcal{X}_{C_s-1}^{k+1}$。

置 $k++$和 C_s--，返回**Step3**。

§5.4 基于合并的聚类算法

常用类间距离

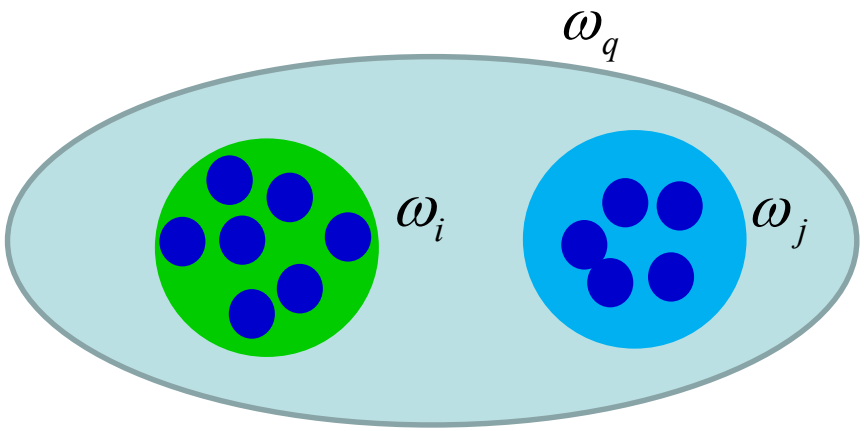
1. 最小距离



$$D_{ij} = \underset{u,v}{Minimize}\ \{d_{uv}\}$$

$$d_{uv} = d(\boldsymbol{X}_u, \boldsymbol{X}_v)$$

$$\boldsymbol{X}_u \in \omega_i, \boldsymbol{X}_v \in \omega_j$$



递推计算

$$D_{pq} = Minimize\ (D_{pi},\ D_{pj})$$

$$D_{pq} = \underset{u,v}{Minimize}\ \{d_{uv}\} \quad (\boldsymbol{X}_u \in \omega_p, \boldsymbol{X}_v \in \omega_q)$$

$$= \underset{u,v}{Minimize}\ \{d_{uv}\} \quad (\boldsymbol{X}_u \in \omega_p, \boldsymbol{X}_v \in \omega_i \text{ or } \boldsymbol{X}_v \in \omega_j)$$

$$= \text{Minimize}\ (\underset{u,v}{Minimize}\ \{d_{uv}\}, \underset{u,w}{Minimize}\ \{d_{uw}\})\ \ (\boldsymbol{X}_u \in \omega_p, \boldsymbol{X}_v \in \omega_i, \boldsymbol{X}_w \in \omega_j)$$

$$= \text{Minimize}\ (D_{pi}, D_{pj})$$

§5.4 基于合并的聚类算法

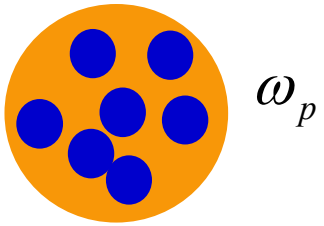
常用类间距离

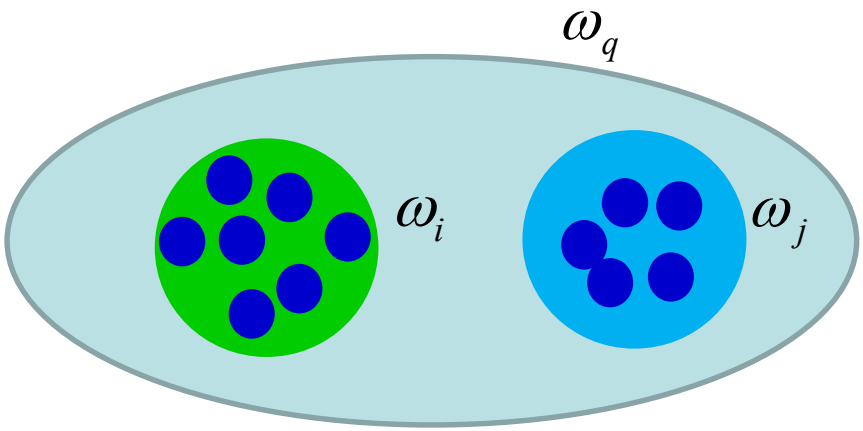
2. 最大距离

$$D_{ij} = \underset{u,v}{Maximize}\ \{d_{uv}\}$$

$$d_{uv} = d(\boldsymbol{X}_u, \boldsymbol{X}_v)$$

$$\boldsymbol{X}_u \in \omega_i, \boldsymbol{X}_v \in \omega_j$$





递推计算

$$D_{pq} = Maximize\ (D_{pi},\ D_{pj})$$

$$D_{pq} = \underset{u,v}{Maximize}\ \{d_{uv}\} \quad (\boldsymbol{X}_u \in \omega_p, \boldsymbol{X}_v \in \omega_q)$$

$$= \underset{u,v}{Maximize}\ \{d_{uv}\} \quad (\boldsymbol{X}_u \in \omega_p, \boldsymbol{X}_v \in \omega_i \text{ or } \boldsymbol{X}_v \in \omega_j)$$

$$= \text{Maximize}\ (\underset{u,v}{Maximize}\ \{d_{uv}\}, \underset{u,w}{Maximize}\ \{d_{uw}\}) \quad (\boldsymbol{X}_u \in \omega_p, \boldsymbol{X}_v \in \omega_i, \boldsymbol{X}_w \in \omega_j)$$

$$= \text{Maximize}\ (D_{pi}, D_{pj})$$

§5.4 基于合并的聚类算法

常用类间距离

3. 类中心距离

$$X_q = \frac{1}{2}(X_i + X_j)$$

X_i ω_i 的类中心位置向量

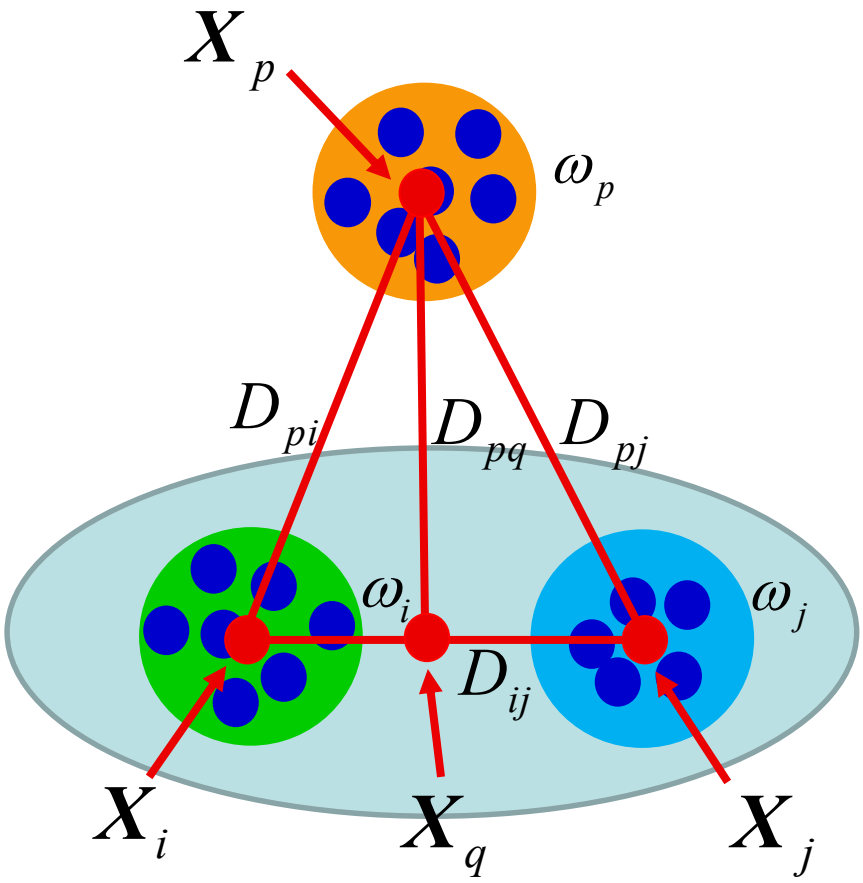
X_j ω_j 的类中心位置向量

X_q ω_q 的类中心位置向量

递推计算

$$D_{pq} = [\frac{1}{2}D_{pi}^2 + \frac{1}{2}D_{pj}^2 - \frac{1}{4}D_{ij}^2]^{\frac{1}{2}}$$

三角形中线计算公式



§5.4 基于合并的聚类算法

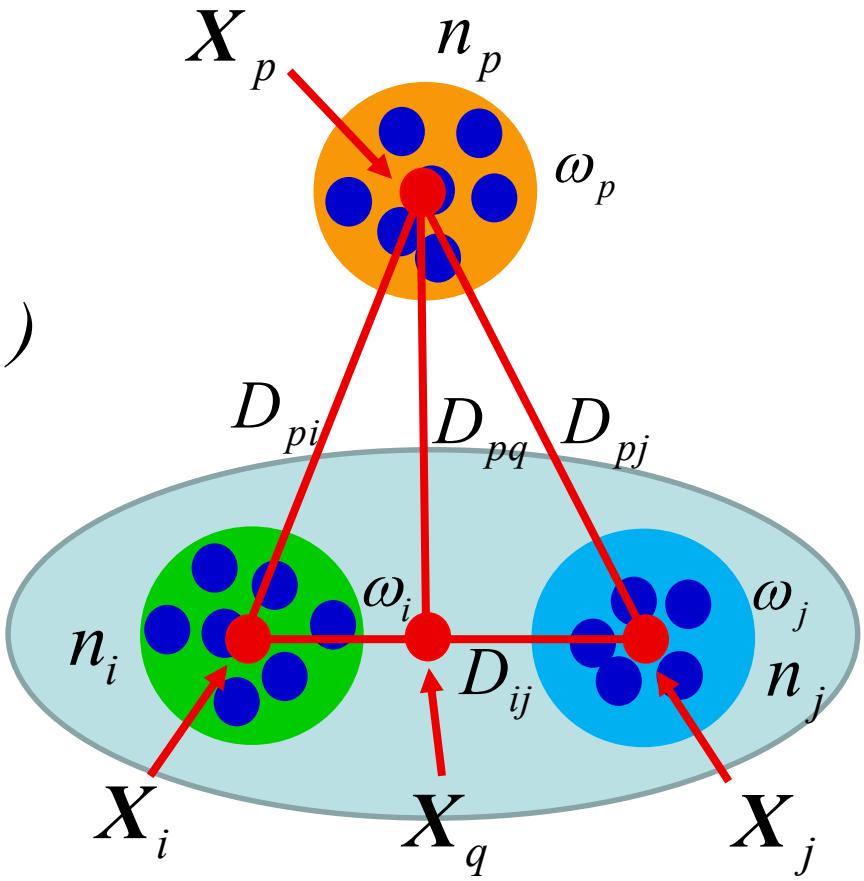
常用类间距离

4. 修正的类中心距离

$$X_q = \frac{1}{n_i + n_j}(n_i X_i + n_j X_j)$$

$$D_{pq} = [(X_p - X_q)^T (X_p - X_q)]^{\frac{1}{2}}$$

$$D_{pq} = [\frac{n_i}{n_i + n_j} D_{pi}^2 + \frac{n_j}{n_i + n_j} D_{pj}^2 - \frac{n_i n_j}{(n_i + n_j)^2} D_{ij}^2]^{\frac{1}{2}}$$



§ 5.4 基于合并的聚类算法

$$D_{pq} = [(X_p - X_q)^T (X_p - X_q)]^{\frac{1}{2}}$$

$$D_{pq}^2 = (X_p - \frac{1}{n_i + n_j}(n_i X_i + n_j X_j))^T (X_p - \frac{1}{n_i + n_j}(n_i X_i + n_j X_j))$$

$$= (\frac{n_i}{n_i + n_j}(X_p - X_i) + \frac{n_j}{n_i + n_j}(X_p - X_j))^T (\frac{n_i}{n_i + n_j}(X_p - X_i) + \frac{n_j}{n_i + n_j}(X_p - X_j))$$

$$= \frac{n_i^2}{(n_i + n_j)^2}(X_p - X_i)^T (X_p - X_i) + \frac{n_i n_j}{(n_i + n_j)^2}(X_p - X_i)^T (X_p - X_j)$$

$$+ \frac{n_i n_j}{(n_i + n_j)^2}(X_p - X_j)^T (X_p - X_i) + \frac{n_j^2}{(n_i + n_j)^2}(X_p - X_j)^T (X_p - X_j)$$

$$= \frac{n_i^2}{(n_i + n_j)^2} D_{pi}^2 + \frac{n_i n_j}{(n_i + n_j)^2}(X_p - X_i)^T (X_p - X_i + (X_i - X_j))$$

$$+ \frac{n_i n_j}{(n_i + n_j)^2}(X_p - X_j)^T (X_p - X_j - (X_i - X_j)) + \frac{n_j^2}{(n_i + n_j)^2} D_{pj}^2$$

$$= \frac{n_i^2}{(n_i + n_j)^2} D_{pi}^2 + \frac{n_i n_j}{(n_i + n_j)^2} D_{pi}^2 + \frac{n_i n_j}{(n_i + n_j)^2}(X_p - X_i)^T (X_i - X_j)$$

$$- \frac{n_i n_j}{(n_i + n_j)^2}(X_p - X_j)^T (X_i - X_j) + \frac{n_i n_j}{(n_i + n_j)^2} D_{pj}^2 + \frac{n_j^2}{(n_i + n_j)^2} D_{pj}^2$$

$$= \frac{n_i}{n_i + n_j} D_{pi}^2 - \frac{n_i n_j}{(n_i + n_j)^2}(X_i - X_j)^T (X_i - X_j) + \frac{n_j}{n_i + n_j} D_{pj}^2$$

$$= \frac{n_i}{n_i + n_j} D_{pi}^2 + \frac{n_j}{n_i + n_j} D_{pj}^2 - \frac{n_i n_j}{(n_i + n_j)^2} D_{ij}^2$$

递推公式证毕。

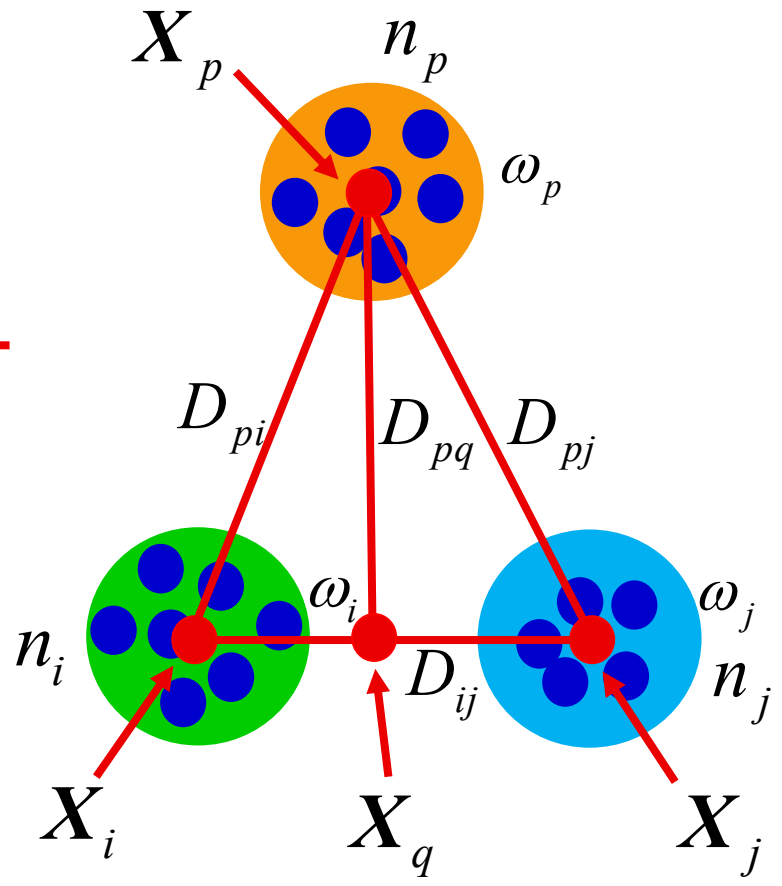
§5.4 基于合并的聚类算法

常用类间距离

5. 类平均距离

$$D_{ij}=[\frac{1}{n_i n_j}\sum_{u\in\omega_i,v\in\omega_j}d_{uv}^2]^{\frac{1}{2}}$$

$$D_{pq}=[\frac{n_i}{n_i+n_j}D_{pi}^2+\frac{n_j}{n_i+n_j}D_{pj}^2]^{\frac{1}{2}}$$



$$D_{pq}=[\frac{1}{n_p n_q}\sum_{u\in\omega_p,v\in\omega_q}d_{uv}^2]^{\frac{1}{2}}$$

$$=[\frac{1}{n_p(n_i+n_j)}(\sum_{u\in\omega_p,v\in\omega_i}d_{uv}^2+\sum_{u\in\omega_p,v\in\omega_j}d_{uv}^2)]^{\frac{1}{2}}$$

$$=[\frac{n_i}{n_i+n_j}\frac{1}{n_p n_i}\sum_{u\in\omega_p,v\in\omega_i}d_{uv}^2+\frac{n_j}{n_i+n_j}\frac{1}{n_p n_j}\sum_{u\in\omega_p,v\in\omega_j}d_{uv}^2)]^{\frac{1}{2}}$$

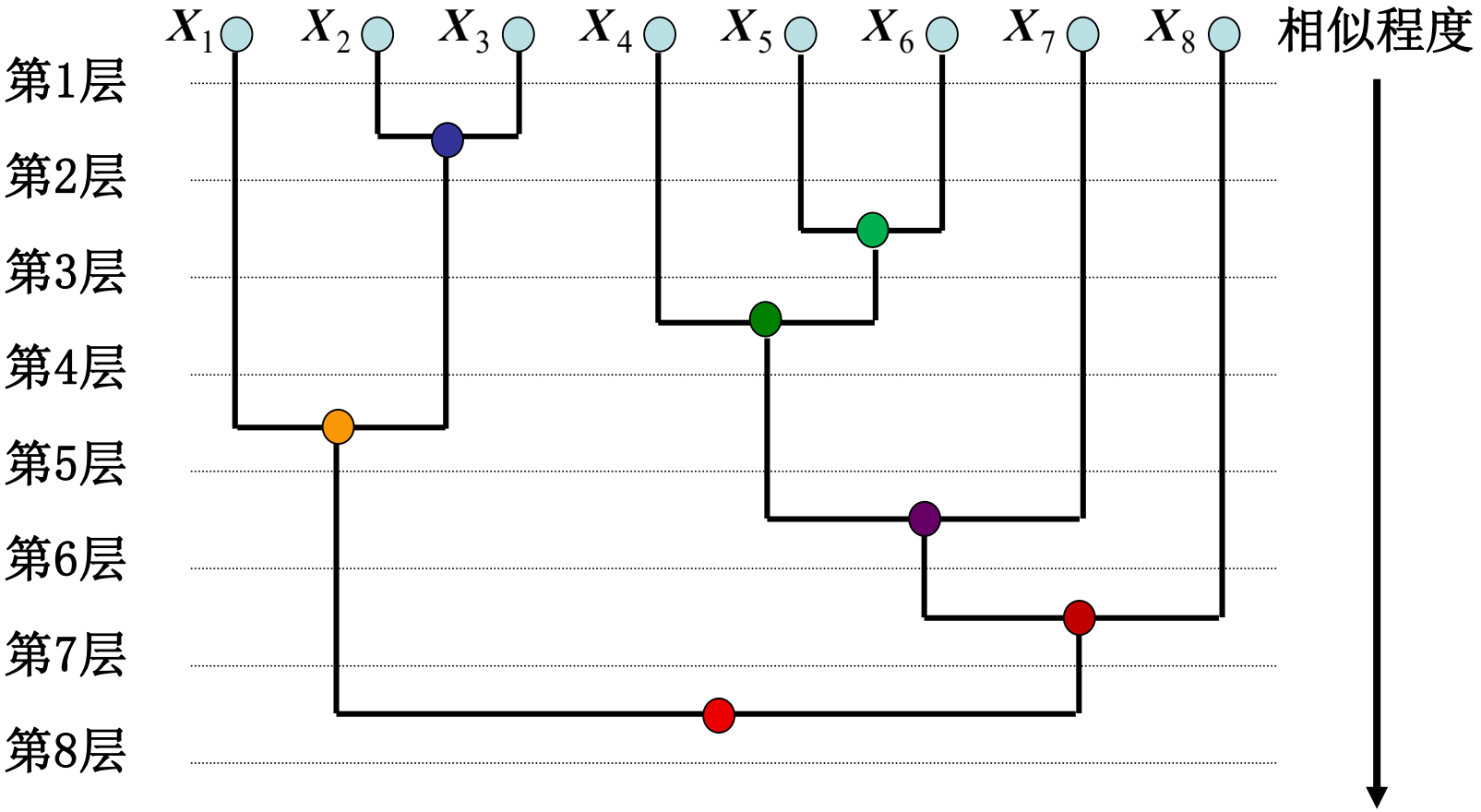
$$=[\frac{n_i}{n_i+n_j}D_{pi}^2+\frac{n_j}{n_i+n_j}D_{pj}^2]^{\frac{1}{2}}$$

递推公式证毕。

§5.4 基于合并的聚类算法

设置阈值T充分大

减类聚类算法特例：层次聚类

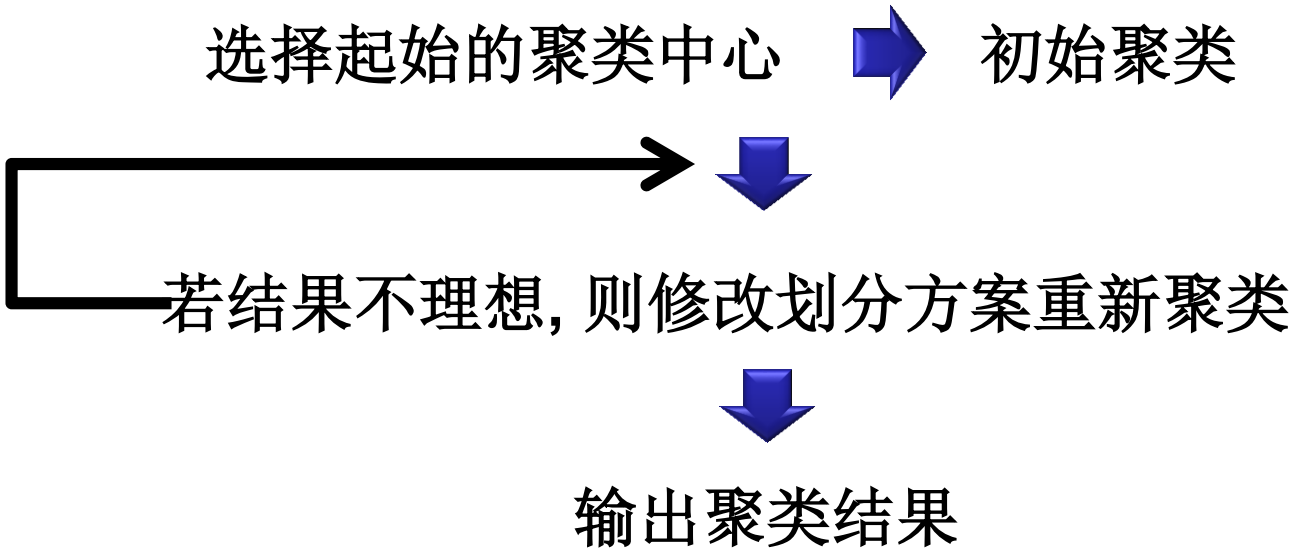


类别数　样本数　层数

$$c = N - K + 1$$

§5.5 动态聚类算法

基本思路



- C均值聚类算法
- ISODATA算法
- 基于核相似度量的动态聚类算法

§5.5 动态聚类算法

C均值聚类算法（I）

Step1. 读入训练样本集 $\mathcal{X}=\{\boldsymbol{X}_1,\boldsymbol{X}_2,\ldots,\boldsymbol{X}_N\}$；

Step2. 执行初始化操作：

设置期望的类别数C；

置迭代计数器 $k=0$；

任选C个样本作为初始聚类中心，例如：$\boldsymbol{Z}_j^k=\boldsymbol{X}_j, j=1,2,\ldots,C$

Step3. 计算 $d(\boldsymbol{X}_l,\boldsymbol{Z}_j^k),\quad l=1,2,\ldots N, j=1,2,\ldots C$。

判断:若 $d(\boldsymbol{X}_l,\boldsymbol{Z}_i^k)=\underset{j=1,2,\ldots C}{Minimize}\ \{d(\boldsymbol{X}_l,\boldsymbol{Z}_j^k)\}$，则 $\boldsymbol{X}_l\in\omega_i^k$ 。

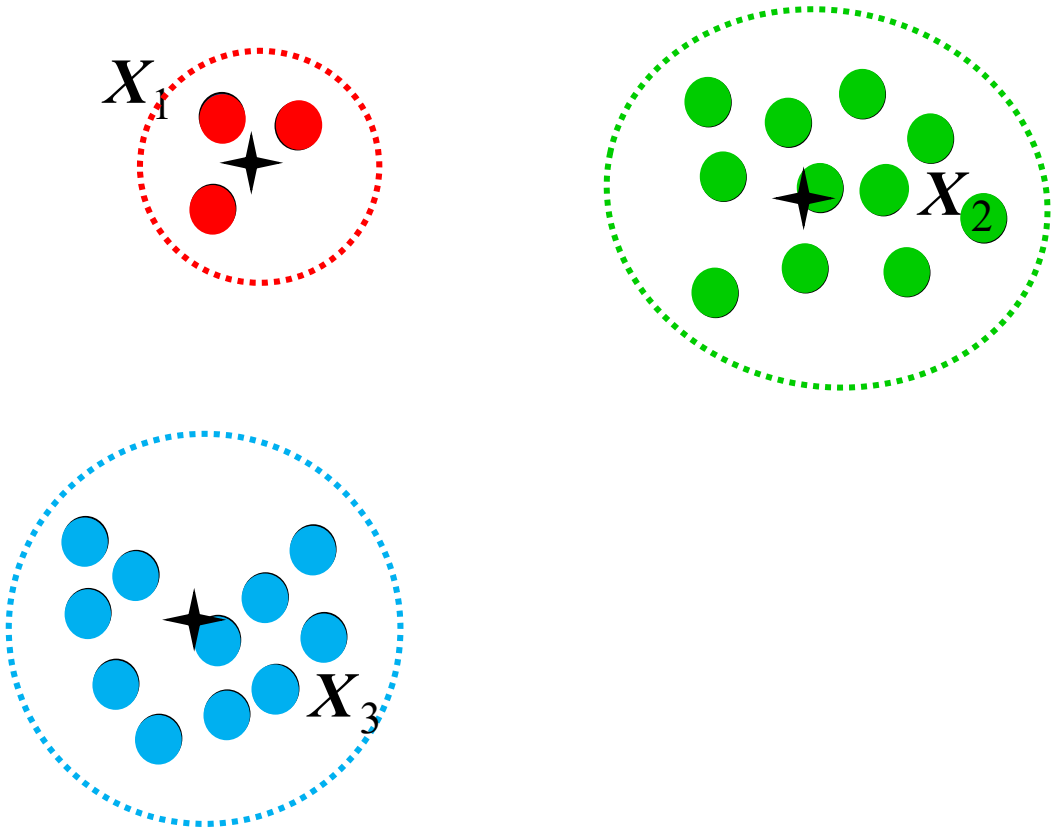
Step4. 计算 $\boldsymbol{Z}_j^{k+1}=\dfrac{1}{n_j}\sum_{l=1}^{n_j}\boldsymbol{X}_{j,l}^k\qquad j=1,2,\ldots,C$

若 $\boldsymbol{Z}_j^{k+1}\neq\boldsymbol{Z}_j^k, j=1,2,\ldots C$，则 $k++$,返回**Step3**；

否则，输出聚类结果，算法结束。

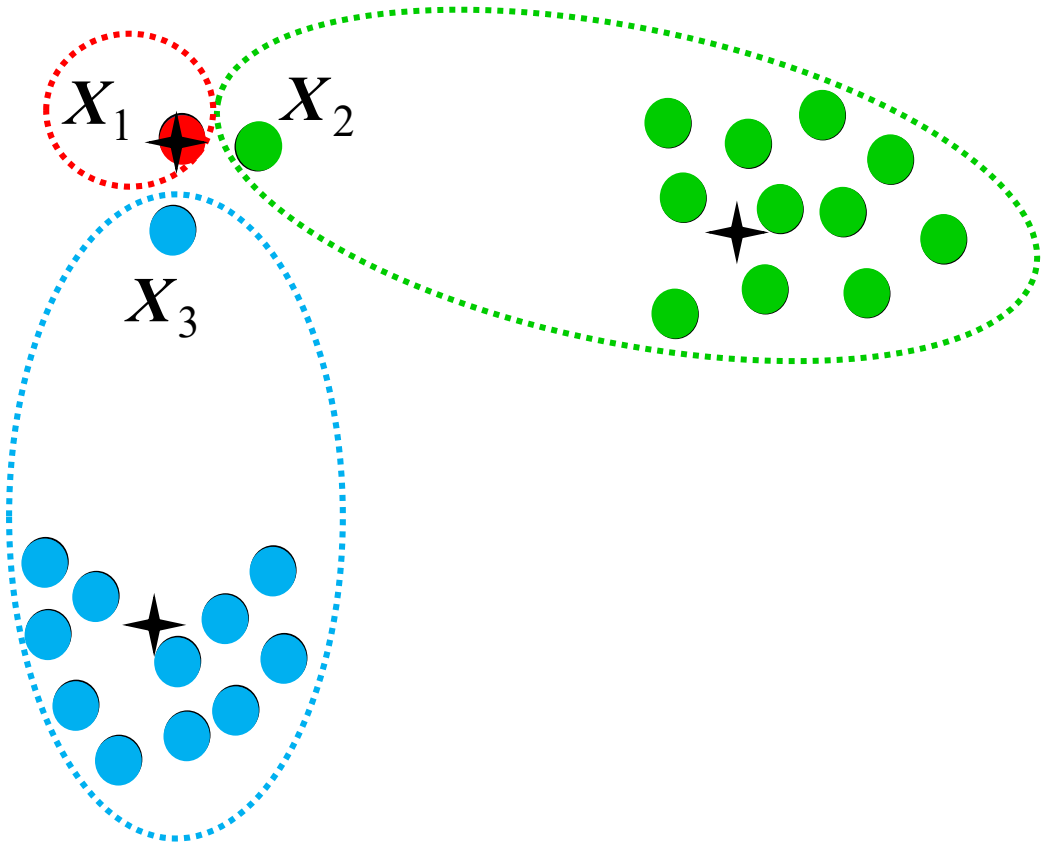
§5.5 动态聚类算法

C均值聚类算法（I）举例



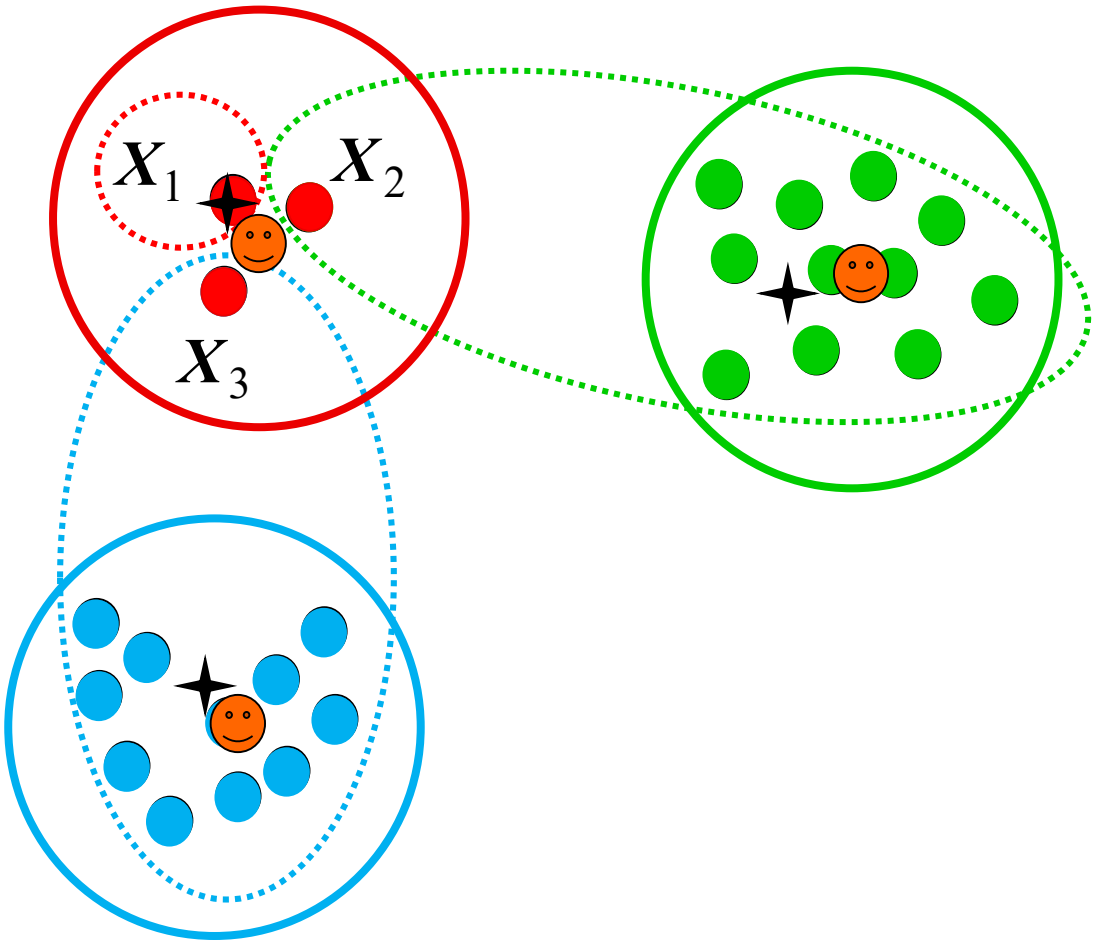
§5.5 动态聚类算法

C均值聚类算法（I）举例



§ 5.5 动态聚类算法

C均值聚类算法（I）举例



§5.5 动态聚类算法

C均值聚类算法（II）

基本动机：让样本在类别间移动使总误差平方和不断减少

设在第k次迭代中，$X_{i,l}^{k} \in \mathcal{X}_i^k \longrightarrow \mathcal{X}_j^k$

则两个样本子集分别变为

$$\begin{cases} \mathcal{X}_i^{k+1} = \mathcal{X}_i^k - \{X_{i,l}^k\} \\ \mathcal{X}_j^{k+1} = \mathcal{X}_j^k + \{X_{i,l}^k\} \end{cases}$$

$$\begin{aligned} Z_i^{k+1} &= \frac{1}{n_i - 1}\left(\sum_{m=1}^{n_i} X_{i,m}^k - X_{i,l}^k\right) \\ &= \frac{n_i}{n_i - 1} Z_i^k - \frac{1}{n_i - 1} X_{i,l}^k \\ &= Z_i^k + \frac{1}{n_i - 1}(Z_i^k - X_{i,l}^k) \end{aligned}$$

$$\begin{aligned} Z_j^{k+1} &= \frac{1}{n_j + 1}\left(\sum_{m=1}^{n_j} X_{j,m}^k + X_{i,l}^k\right) \\ &= \frac{n_j}{n_j + 1} Z_j^k + \frac{1}{n_j + 1} X_{i,l}^k \\ &= Z_j^k - \frac{1}{n_j + 1}(Z_j^k - X_{i,l}^k) \end{aligned}$$

C均值聚类算法（II）

迭代前后，两个类别类内误差平方和分别为

$$J_{e,i}^{k}=\sum_{m=1}^{n_i}\left\|\boldsymbol{X}_{i,m}^{k}-\boldsymbol{Z}_{i}^{k}\right\|^{2}$$

$$J_{e,j}^{k}=\sum_{m=1}^{n_j}\left\|\boldsymbol{X}_{j,m}^{k}-\boldsymbol{Z}_{j}^{k}\right\|^{2}$$

$$J_{e,i}^{k+1}=\sum_{m=1,m\neq l}^{n_i}\left\|\boldsymbol{X}_{i,m}^{k}-\boldsymbol{Z}_{i}^{k+1}\right\|^{2}=\sum_{m=1}^{n_i}\left\|\boldsymbol{X}_{i,m}^{k}-\boldsymbol{Z}_{i}^{k+1}\right\|^{2}-\left\|\boldsymbol{X}_{i,l}^{k}-\boldsymbol{Z}_{i}^{k+1}\right\|^{2}$$

$$J_{e,j}^{k+1}=\sum_{m=1}^{n_j}\left\|\boldsymbol{X}_{j,m}^{k}-\boldsymbol{Z}_{j}^{k+1}\right\|^{2}+\left\|\boldsymbol{X}_{i,l}^{k}-\boldsymbol{Z}_{j}^{k+1}\right\|^{2}$$

§ 5.5 动态聚类算法

C均值聚类算法（II）

$$J_{e,i}^{k+1}=\sum_{m=1,m\neq l}^{n_i}\left\|\boldsymbol{X}_{i,m}^{k}-\boldsymbol{Z}_i^{k+1}\right\|^2=\sum_{m=1}^{n_i}\left\|\boldsymbol{X}_{i,m}^{k}-\boldsymbol{Z}_i^{k+1}\right\|^2-\left\|\boldsymbol{X}_{i,l}^{k}-\boldsymbol{Z}_i^{k+1}\right\|^2$$

$$=\sum_{m=1}^{n_i}\left\|\boldsymbol{X}_{i,m}^{k}-(\boldsymbol{Z}_i^{k}+\frac{1}{n_i-1}(\boldsymbol{Z}_i^{k}-\boldsymbol{X}_{i,l}^{k}))\right\|^2-\left\|\boldsymbol{X}_{i,l}^{k}-(\boldsymbol{Z}_i^{k}+\frac{1}{n_i-1}(\boldsymbol{Z}_i^{k}-\boldsymbol{X}_{i,l}^{k}))\right\|^2$$

$$=\sum_{m=1}^{n_i}\left\|(\boldsymbol{X}_{i,m}^{k}-\boldsymbol{Z}_i^{k})-\frac{1}{n_i-1}(\boldsymbol{Z}_i^{k}-\boldsymbol{X}_{i,l}^{k}))\right\|^2-\left\|\frac{n_i}{n_i-1}(\boldsymbol{X}_{i,l}^{k}-\boldsymbol{Z}_i^{k})\right\|^2$$

$$=\sum_{m=1}^{n_i}((\boldsymbol{X}_{i,m}^{k}-\boldsymbol{Z}_i^{k})-\frac{1}{n_i-1}(\boldsymbol{Z}_i^{k}-\boldsymbol{X}_{i,l}^{k}))^T((\boldsymbol{X}_{i,m}^{k}-\boldsymbol{Z}_i^{k})-\frac{1}{n_i-1}(\boldsymbol{Z}_i^{k}-\boldsymbol{X}_{i,l}^{k}))-\left\|\frac{n_i}{n_i-1}(\boldsymbol{Z}_i^{k}-\boldsymbol{X}_{i,l}^{k}))\right\|^2$$

$$=\sum_{m=1}^{n_i}(\boldsymbol{X}_{i,m}^{k}-\boldsymbol{Z}_i^{k})^T(\boldsymbol{X}_{i,m}^{k}-\boldsymbol{Z}_i^{k})-2\frac{1}{n_i-1}\sum_{m=1}^{n_i}(\boldsymbol{X}_{i,m}^{k}-\boldsymbol{Z}_i^{k})^T(\boldsymbol{Z}_i^{k}-\boldsymbol{X}_{i,l}^{k})$$

$$+\frac{1}{(n_i-1)^2}\sum_{m=1}^{n_i}(\boldsymbol{X}_{i,l}^{k}-\boldsymbol{Z}_i^{k})^T(\boldsymbol{X}_{i,l}^{k}-\boldsymbol{Z}_i^{k})-\frac{n_i^2}{(n_i-1)^2}\left\|\boldsymbol{X}_{i,l}^{k}-\boldsymbol{Z}_i^{k}\right\|^2$$

§ 5.5 动态聚类算法

C均值聚类算法（II）

为0

$$J_{e,i}^{k+1}=\sum_{m=1}^{n_i}(\boldsymbol{X}_{i,m}^k-\boldsymbol{Z}_i^k)^T(\boldsymbol{X}_{i,m}^k-\boldsymbol{Z}_i^k)-2\frac{1}{n_i-1}\sum_{m=1}^{n_i}(\boldsymbol{X}_{i,m}^k-\boldsymbol{Z}_i^k)^T(\boldsymbol{Z}_i^k-\boldsymbol{X}_{i,l}^k)$$

$$+\frac{1}{(n_i-1)^2}\sum_{m=1}^{n_i}(\boldsymbol{X}_{i,l}^k-\boldsymbol{Z}_i^k)^T(\boldsymbol{X}_{i,l}^k-\boldsymbol{Z}_i^k)-\frac{n_i^2}{(n_i-1)^2}\left\|\boldsymbol{X}_{i,l}^k-\boldsymbol{Z}_i^k\right\|^2$$

$$=\sum_{m=1}^{n_i}\left\|\boldsymbol{X}_{i,m}^k-\boldsymbol{Z}_i^k\right\|^2+\frac{n_i}{(n_i-1)^2}\left\|\boldsymbol{X}_{i,l}^k-\boldsymbol{Z}_i^k\right\|^2-\frac{n_i^2}{(n_i-1)^2}\left\|\boldsymbol{X}_{i,l}^k-\boldsymbol{Z}_i^k\right\|^2$$

$$=J_{e,i}^k-\frac{n_i}{n_i-1}\left\|\boldsymbol{X}_{i,l}^k-\boldsymbol{Z}_i^k\right\|^2$$

$$\Delta J_{e,i}=J_{e,i}^{k+1}-J_{e,i}^k=-\frac{n_i}{n_i-1}\left\|\boldsymbol{X}_{i,l}^k-\boldsymbol{Z}_i^k\right\|^2$$

$$J_{e,j}^{k+1}=J_{e,j}^k+\frac{n_j}{n_j+1}\left\|\boldsymbol{X}_{i,l}^k-\boldsymbol{Z}_j^k\right\|^2 \qquad \Delta J_{e,j}=J_{e,j}^{k+1}-J_{e,j}^k=\frac{n_j}{n_j+1}\left\|\boldsymbol{X}_{i,l}^k-\boldsymbol{Z}_j^k\right\|^2$$

§5.5 动态聚类算法

C均值聚类算法（II）

$$\Delta J_{e,i} = J_{e,i}^{k+1} - J_{e,i}^{k} = -\frac{n_i}{n_i - 1}\left\| \boldsymbol{X}_{i,l}^{k} - \boldsymbol{Z}_i^{k} \right\|^2$$

$$\Delta J_{e,j} = J_{e,j}^{k+1} - J_{e,j}^{k} = \frac{n_j}{n_j + 1}\left\| \boldsymbol{X}_{i,l}^{k} - \boldsymbol{Z}_j^{k} \right\|^2$$

类间移动所引起的总误差平方和变化为

$$\Delta J_e = (J_{e,i}^{k+1} + J_{e,j}^{k+1}) - (J_{e,i}^{k} + J_{e,j}^{k}) = (J_{e,i}^{k+1} - J_{e,i}^{k}) + (J_{e,j}^{k+1} - J_{e,j}^{k})$$

$$= -[\frac{n_i}{n_i - 1}\left\| \boldsymbol{X}_{i,l}^{k} - \boldsymbol{Z}_i^{k} \right\|^2 - \frac{n_j}{n_j + 1}\left\| \boldsymbol{X}_{i,l}^{k} - \boldsymbol{Z}_j^{k} \right\|^2]$$

大于0

最大化！

§5.5 动态聚类算法

C均值聚类算法（II）

Step1. 读入训练样本集 $\mathcal{X}=\{\boldsymbol{X}_1,\boldsymbol{X}_2,\ldots,\boldsymbol{X}_N\}$；

Step2. 执行初始化操作：

设置期望的类别数C；

置迭代计数器 $k=0$；

任选C个样本作为初始聚类中心，例如：$\boldsymbol{Z}_j^k=\boldsymbol{X}_j, j=1,2,\ldots,C$

Step3. 计算 $d(\boldsymbol{X}_l,\boldsymbol{Z}_j^k), \quad l=1,2,\ldots N, j=1,2,\ldots C$。

判断:若 $d(\boldsymbol{X}_l,\boldsymbol{Z}_i^k)=\underset{j=1,2,\ldots C}{Minimize}\{d(\boldsymbol{X}_l,\boldsymbol{Z}_j^k)\}$，则 $\boldsymbol{X}_l\in\omega_i^k$ 。

计算 $\boldsymbol{Z}_j^{k+1}=\frac{1}{n_j}\sum_{l=1}^{n_j}\boldsymbol{X}_{j,l}^k \qquad j=1,2,\ldots,C$

$$J_e^k=\sum_{j=1}^{C}J_{e,j}^k=\sum_{j=1}^{C}\sum_{m=1}^{n_j}\left\|\boldsymbol{X}_{j,m}^k-\boldsymbol{Z}_j^k\right\|^2$$

§5.5 动态聚类算法

C均值聚类算法（II）

Step4. 求

$$\underset{i,j,l}{\text{Maximize}}\left(\frac{n_i}{n_i-1}\left\|\boldsymbol{X}_{i,l}^{k}-\boldsymbol{Z}_{i}^{k}\right\|^2-\frac{n_j}{n_j+1}\left\|\boldsymbol{X}_{i,l}^{k}-\boldsymbol{Z}_{j}^{k}\right\|^2\right)$$

$$Subject\ to\quad i=1,2,\ldots C\quad j=1,2,\ldots C\quad l=1,2,\ldots n_i$$

若 $\frac{n_i}{n_i-1}\left\|\boldsymbol{X}_{i_{max},l_{max}}^{k}-\boldsymbol{Z}_{i_{max}}^{k}\right\|^2-\frac{n_j}{n_j+1}\left\|\boldsymbol{X}_{i_{max},l_{max}}^{k}-\boldsymbol{Z}_{j_{max}}^{k}\right\|^2\le 0$

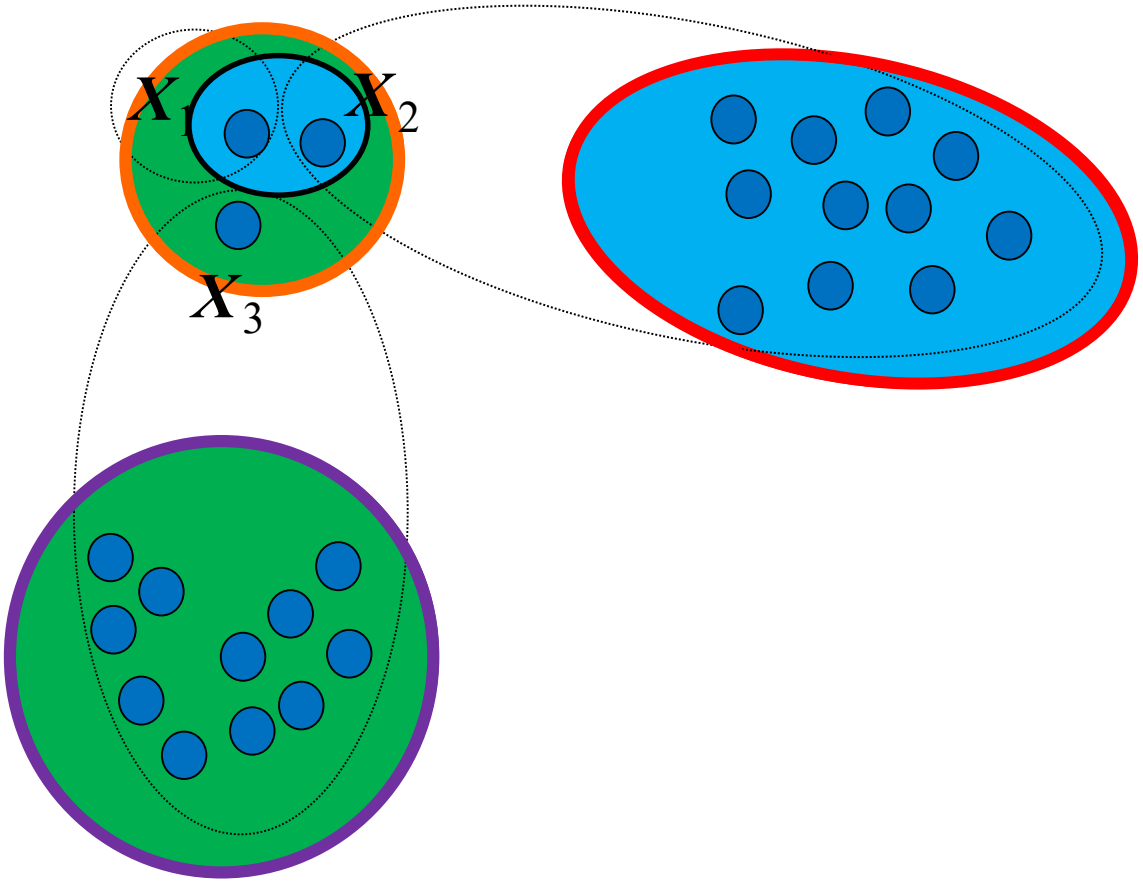
则输出已聚类结果，算法结束；

否则移动 $\boldsymbol{X}_{i_{max},l_{max}}^{k}$ 至第 j_{max} 类，并更新相关参数；

置$k++$,返回**Step4.**。

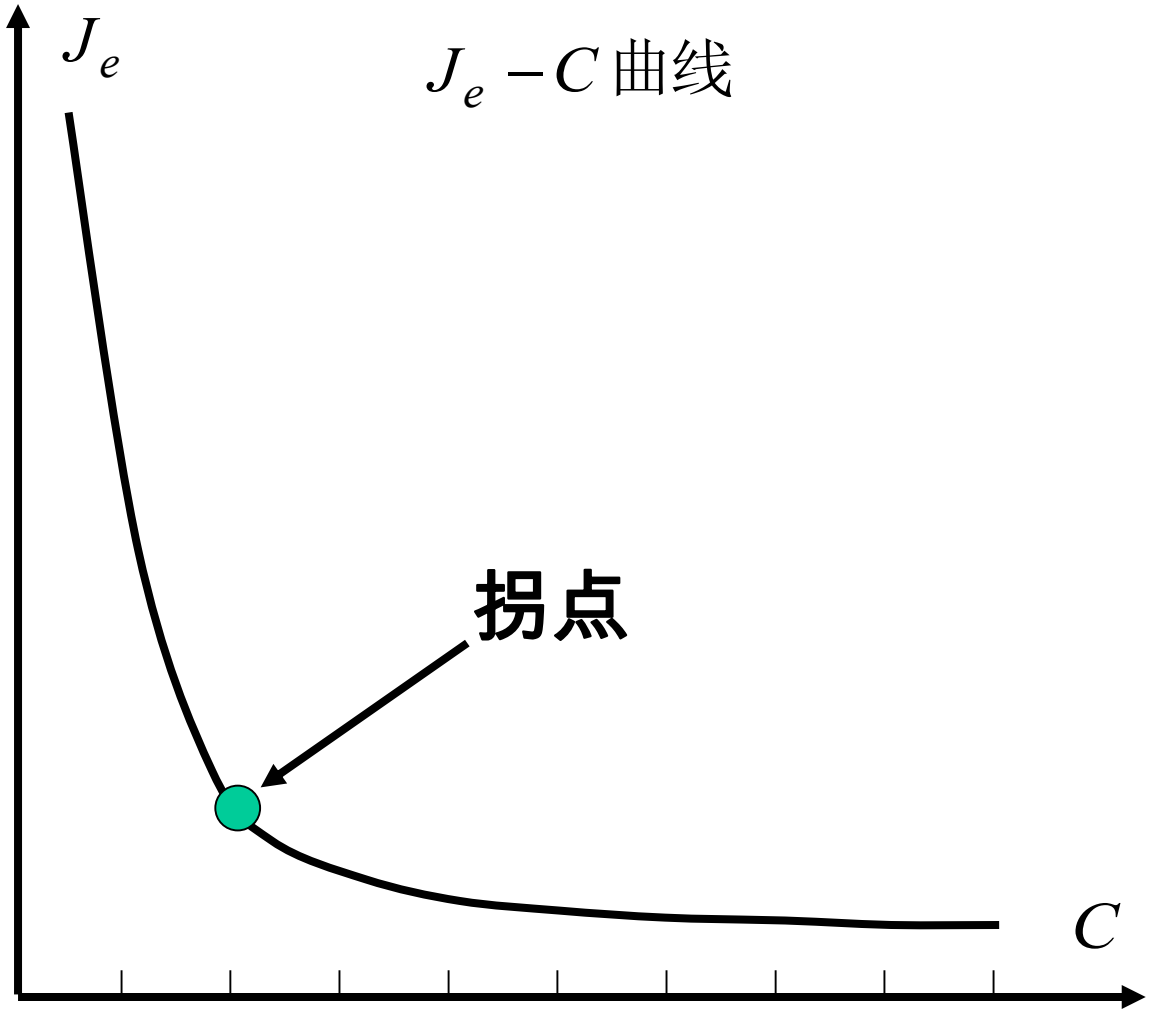
§5.5 动态聚类算法

C均值聚类算法（II）举例



§5.5 动态聚类算法

C均值聚类算法中类别数的选择



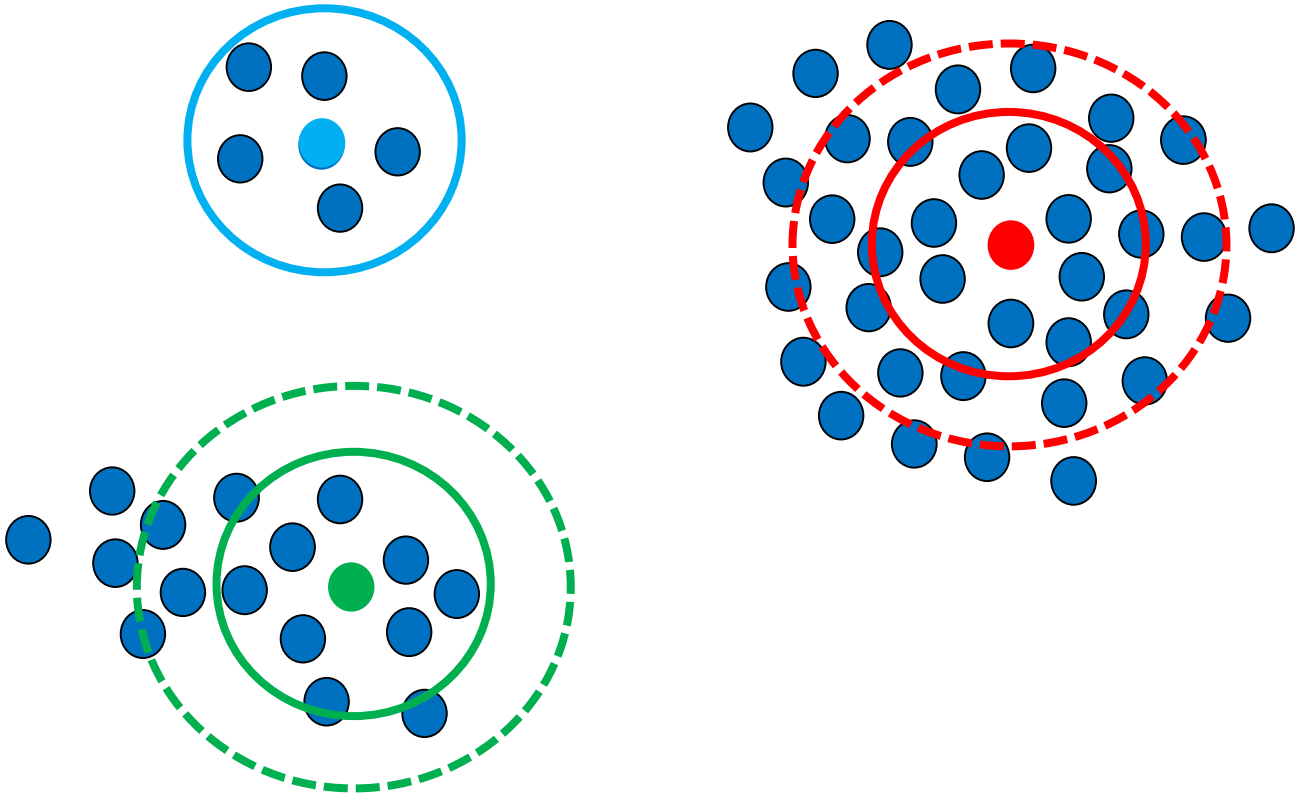
§5.5 动态聚类算法

初始聚类中心的选择

1. 根据过往的经验选择
2. 根据随机分类得到的类别中心选择
3. 根据样本的密度分布选择
4. 根据最大最小距离选择
5. 根据层次聚类法的结果选择

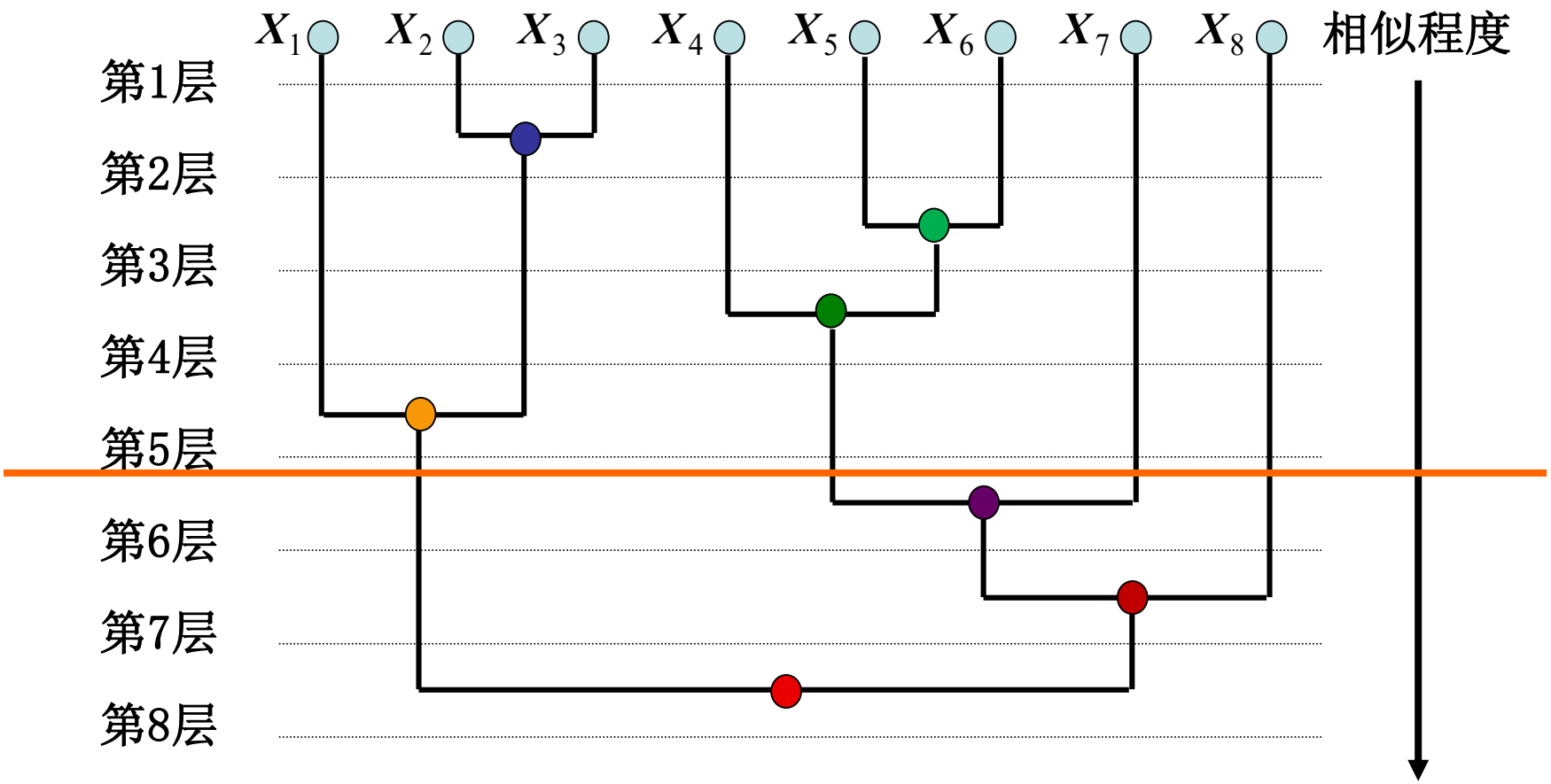
§ 5.5 动态聚类算法

- 根据样本的密度分布选择初始聚类中心



§5.5 动态聚类算法

- 根据层次聚类法的结果选择初始聚类中心



类别数　样本数　层数

$$c = N - K + 1$$

§5.5 动态聚类算法

ISODATA算法（迭代自组织数据分析算法）

ISODATA：Iterative Self-Organizing Data Analysis Technique Algorithm

算法特点：

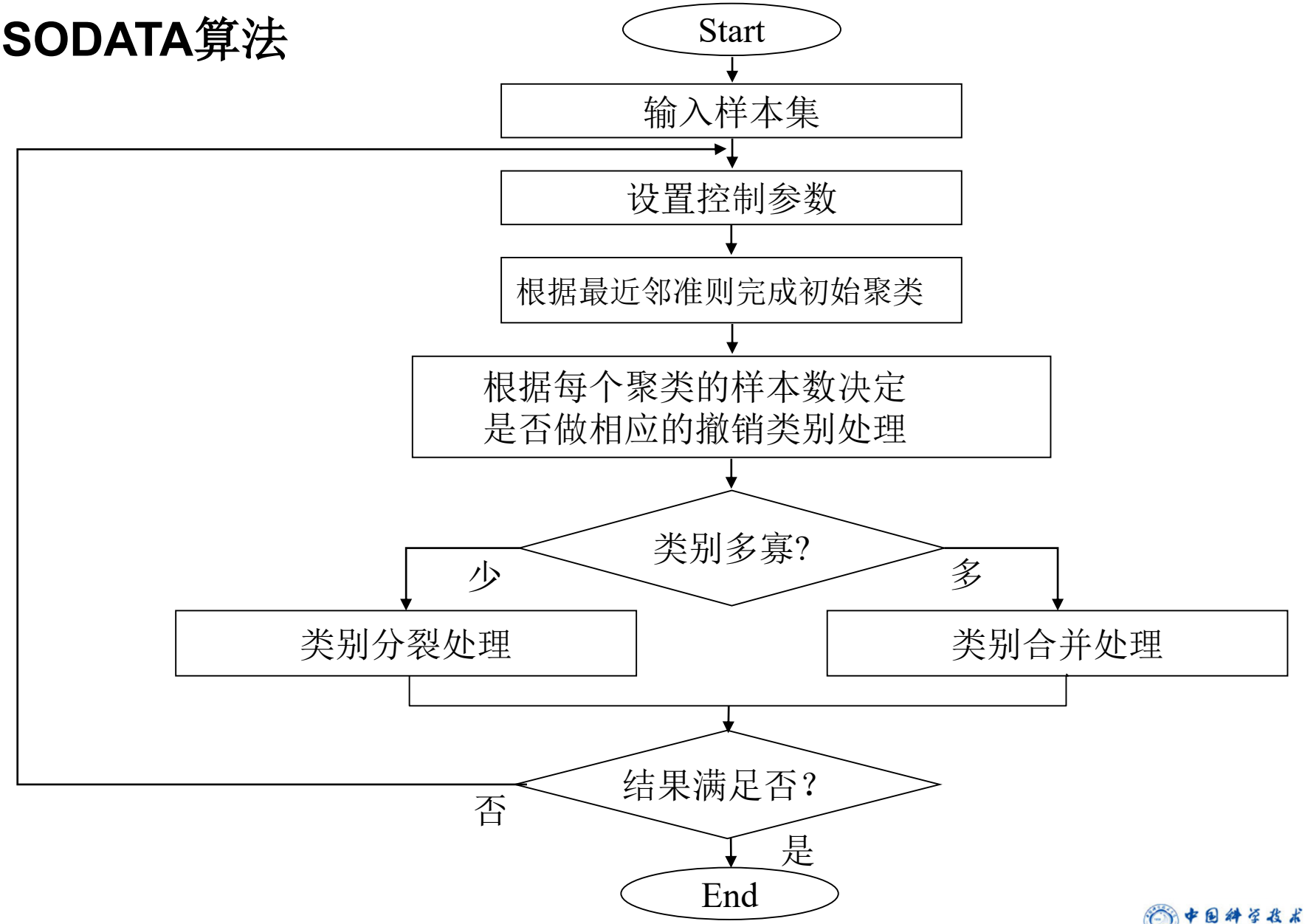
- 迭代运算
- 自组织能力 → ✓聚类数非固定，在指定范围内变化；✓合并、分裂和撤销等处理
- 试探性的操作和人机交互功能

若干标记

C	期望的聚类数
N_s	每个聚类中应包含的最少样本数
T_σ	所允许的类别标准偏差分量的上限
T_d	任意两个类别的聚类中心之间应具有的最小距离
N_m	在一次迭代运算中所允许的最大合并次数
$I_{\max}$	预设的最大迭代次数

§ 5.5 动态聚类算法

ISODATA算法



§5.5 动态聚类算法

ISODATA算法步骤

Step1. 输入样本集：$\mathcal{X}=\{\boldsymbol{X}_1,\boldsymbol{X}_2,...,\boldsymbol{X}_N\}$

Step2. 完成初始化操作：对诸控制参数进行赋值。

置迭代计数器 $k=1$；

任选C个样本作为初始聚类中心，例如：$\boldsymbol{Z}_j^k=\boldsymbol{X}_j, j=1,2,...,C$

置类别计算器 $C_s=C$。

Step3. 计算 $d(\boldsymbol{X}_l,\boldsymbol{Z}_j),\quad l=1,2,...,N, j=1,2,...,C$。

判断:若 $d(\boldsymbol{X}_l,\boldsymbol{Z}_i)=\underset{j=1,2,...C}{Minimize}\ \{d(\boldsymbol{X}_l,\boldsymbol{Z}_j)\}$，则 $\boldsymbol{X}_l\in\omega_i^k$。

Step4. 对每一个样本子集,做如下处理。

若 $n_j<N_s$，则撤销该样本子集及其聚类中心 $\boldsymbol{Z}_j$，并将子集中所有样本按最近邻准则并入到剩余各类别中，置 $C_s=C_s-1$。

按序号连续原则对所有现行类别重新排序。

§5.5 动态聚类算法

ISODATA算法步骤

Step5. 对每一个聚类，计算其类内平均距离：

$$\overline{D}_j = \frac{1}{n_j}\sum_{m=1}^{n_j} d(\boldsymbol{X}_{j,m}, \boldsymbol{Z}_j), \qquad j = 1,2,\ldots C_s$$

Step6. 计算整个样本集的类内平均距离。

$$\overline{D} = \frac{1}{N}\sum_{j=1}^{C_s} n_j \overline{D}_j$$

Step7. 判断是否进行类别的分裂或合并处理。

若 $C_s \le C/2$，则转**Step 8.**，做分裂处理。

若 $C_s \ge 2C$，则转**Step 11.**，做合并处理。

若 $C/2 \le C_s \le 2C$，则引入某种随机处理机制，选择性地进行分裂或合并处理。

§5.5 动态聚类算法

ISODATA算法步骤

Step8. 计算每个聚类的标准偏差：

$$\boldsymbol{\sigma}_j = ((\boldsymbol{\sigma}_j)_1, (\boldsymbol{\sigma}_j)_2, \ldots, (\boldsymbol{\sigma}_j)_d)^T, \quad j = 1,2,\ldots C_s$$

$$(\boldsymbol{\sigma}_j)_i = \sqrt{\frac{1}{n_j}\sum_{m=1}^{n_j}((\boldsymbol{X}_{j,m})_i - (\boldsymbol{Z}_j)_i)^2}$$

Step9. 求每个聚类的标准偏差分量的最大值

$$(\boldsymbol{\sigma}_j)_{i_{max}} = \underset{i}{Maximize}\{(\boldsymbol{\sigma}_j)_i\}, \quad j = 1,2,\ldots C_s$$

Step10. 对每一个聚类，根据其标准偏差分量最大值的取值情况，并结合其它条件，确定是否执行分裂处理。

Step11-Step14. 略。

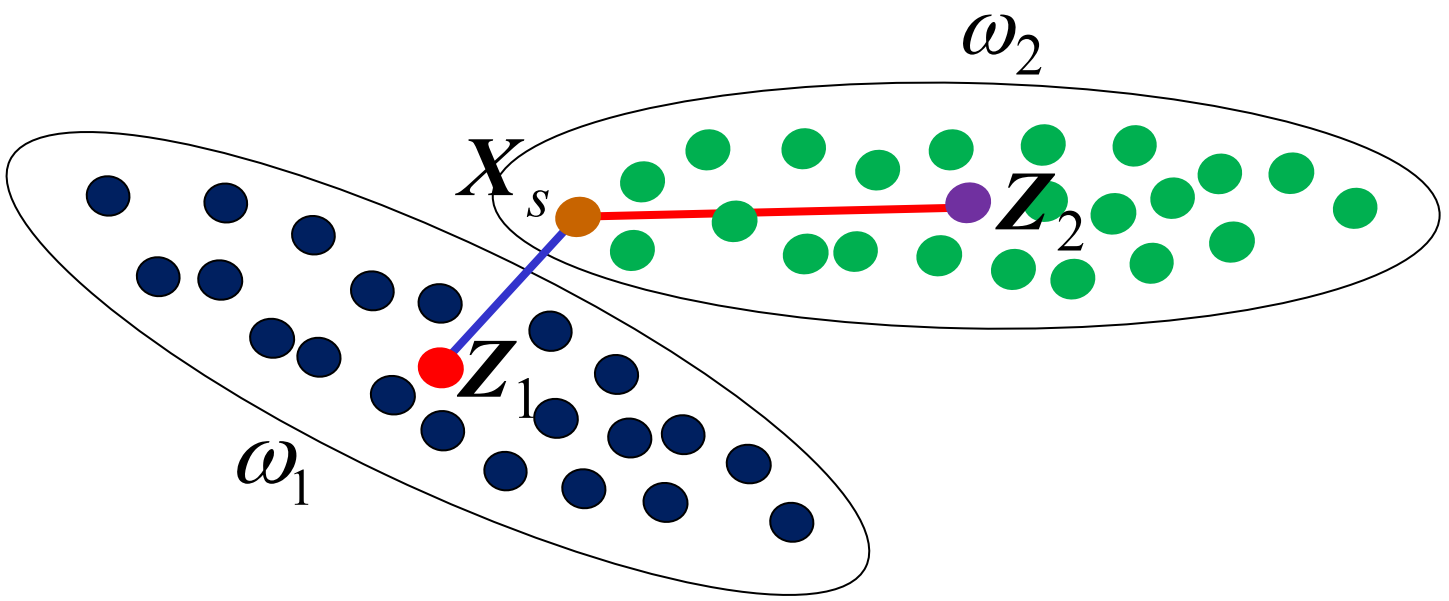
§ 5.5 动态聚类算法

基于样本和核的相似性度量的动态聚类算法

- C-均值聚类算法
- ISODATA算法

- 用聚类中心代表一个类别
- 用最近邻准则进行聚类

未充分考虑样本集的内部结构



§ 5.5 动态聚类算法

基于样本和核的相似性度量的动态聚类算法

如何刻画一个类别？

$K_j = K(X, V_j)$

与 ω_j 有关的一个参数集

以 V_j 为参数集的分类模型 ：函数，或样本子集等。

$$K_j = K(X, \hat{\mu}_j, \hat{\Sigma}_j) = \frac{1}{(2\pi)^{\frac{d}{2}} \left|\hat{\Sigma}_j\right|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}(X-\hat{\mu}_j)^T \hat{\Sigma}_j^{-1}(X-\hat{\mu}_j)\right\}$$

$$K_j = K(X, \hat{\mu}_j) = (X-\hat{\mu}_j)^T \hat{\Sigma}^{-1}(X-\hat{\mu}_j)$$

如何借助核函数完成分类任务？

引入样本和核的相似性度量

若 $\Delta(X_m, K_i) = \underset{j}{Minimize}\ \{\Delta(X_m, K_j)\}$，则 $X_m \in \omega_i$ 。

§5.5 动态聚类算法

基于样本和核的相似性度量的动态聚类算法

Step1. 读入训练样本集 $\mathcal{X}=\{\boldsymbol{X}_1,\boldsymbol{X}_2,\ldots,\boldsymbol{X}_N\}$；

Step2. 执行初始化操作：

设置期望的类别数C；

置迭代计数器 $k=0$；

$$K_j=\frac{1}{n_j}\sum_{l=1}^{n_j}\boldsymbol{X}_{j,l}^k,\ j=1,2,\ldots,C$$
$$\Delta(\boldsymbol{X}_{\mathrm{m}},K_j)=d(\boldsymbol{X}_{\mathrm{m}},K_j),$$
$$\mathrm{m}=1,2,\ldots N,\ j=1,2,\ldots,C。$$

任选C个样本作为初始聚类中心：$\boldsymbol{Z}_j^k,\ j=1,2,\ldots,C$ 。

按最近邻准则划分样本集，确定初始核 $K_j,\ j=1,\ldots C$。

Step3. 计算各样本与已有聚类核之间的相似性：

$$\Delta(\boldsymbol{X}_{\mathrm{m}},K_j),\quad \mathrm{m}=1,2,\ldots N,\ j=1,2,\ldots,C。$$

若 $\Delta(\boldsymbol{X}_{\mathrm{m}},K_{\mathrm{i}})=\underset{j=1,2,\ldots C}{Minimize}\ \{\Delta(\boldsymbol{X}_{\mathrm{m}},K_j)\}$，则 $\boldsymbol{X}_m\in\omega_i$。

Step4. 依据聚类结果更新各类别的核，记为 $\tilde{K}_j,\mathrm{j}=1,\ldots,C$。

若 $\tilde{K}_j\neq K_j,j=1,\ldots C$，则 $K_j=\tilde{K}_j,j=1,\ldots C$，$k++$，返回**Step3**；

否则，输出聚类结果，算法结束。

§ 5.5 动态聚类算法

常用的核函数

- 正态核函数

$$\mathrm{K}_j = K(\boldsymbol{X},\boldsymbol{V}_j) = \frac{1}{(2\pi)^{\frac{d}{2}}\left|\hat{\boldsymbol{\Sigma}}_{\mathrm{j}}\right|^{\frac{1}{2}}}\exp\left\{-\frac{1}{2}(\boldsymbol{X}-\hat{\boldsymbol{\mu}}_{\mathrm{j}})^T\hat{\boldsymbol{\Sigma}}_{\mathrm{j}}^{-1}(\boldsymbol{X}-\hat{\boldsymbol{\mu}}_{\mathrm{j}})\right\}$$

$$\hat{\boldsymbol{\mu}}_j = = \frac{1}{n_j}\sum_{l=1}^{n_j}\boldsymbol{X}_{j,l}$$

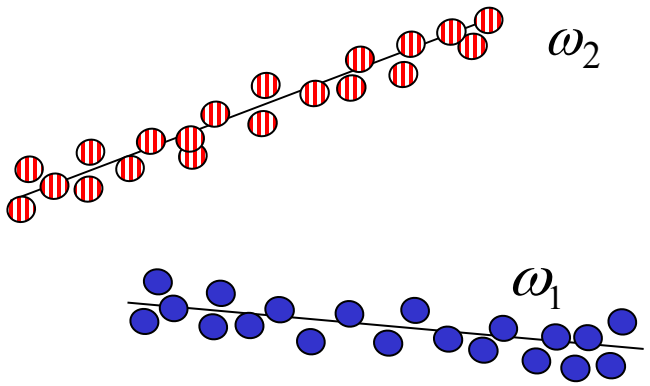
$$\hat{\boldsymbol{\Sigma}}_j = \frac{1}{n_j}\sum_{l=1}^{n_j}(\boldsymbol{X}_l-\hat{\boldsymbol{\mu}}_j)(\boldsymbol{X}_l-\hat{\boldsymbol{\mu}}_j)^T$$

$$\Delta(\boldsymbol{X},\mathrm{K}_j) = \frac{1}{2}(\boldsymbol{X}-\hat{\boldsymbol{\mu}}_{\mathrm{j}})^T\hat{\boldsymbol{\Sigma}}_{\mathrm{j}}^{-1}(\boldsymbol{X}-\hat{\boldsymbol{\mu}}_{\mathrm{j}}) + \frac{1}{2}\log\left|\hat{\boldsymbol{\Sigma}}_{\mathrm{j}}\right|$$

§ 5.5 动态聚类算法

常用的核函数

- 主轴核函数



$$K_j = K(\boldsymbol{X}, \boldsymbol{V}_j) = \boldsymbol{U}_j{}^T \boldsymbol{X}$$

$$\boldsymbol{U}_j = (\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots \boldsymbol{u}_{d_j})$$

$$\hat{\Sigma}_j \boldsymbol{u}_k = \lambda_k \boldsymbol{u}_k, k = 1,2,\ldots,d_j$$

$$\lambda_1 > \lambda_1 > \Lambda > \lambda_{d_j}$$

$\hat{\Sigma}_j, \quad j = 1,2.$

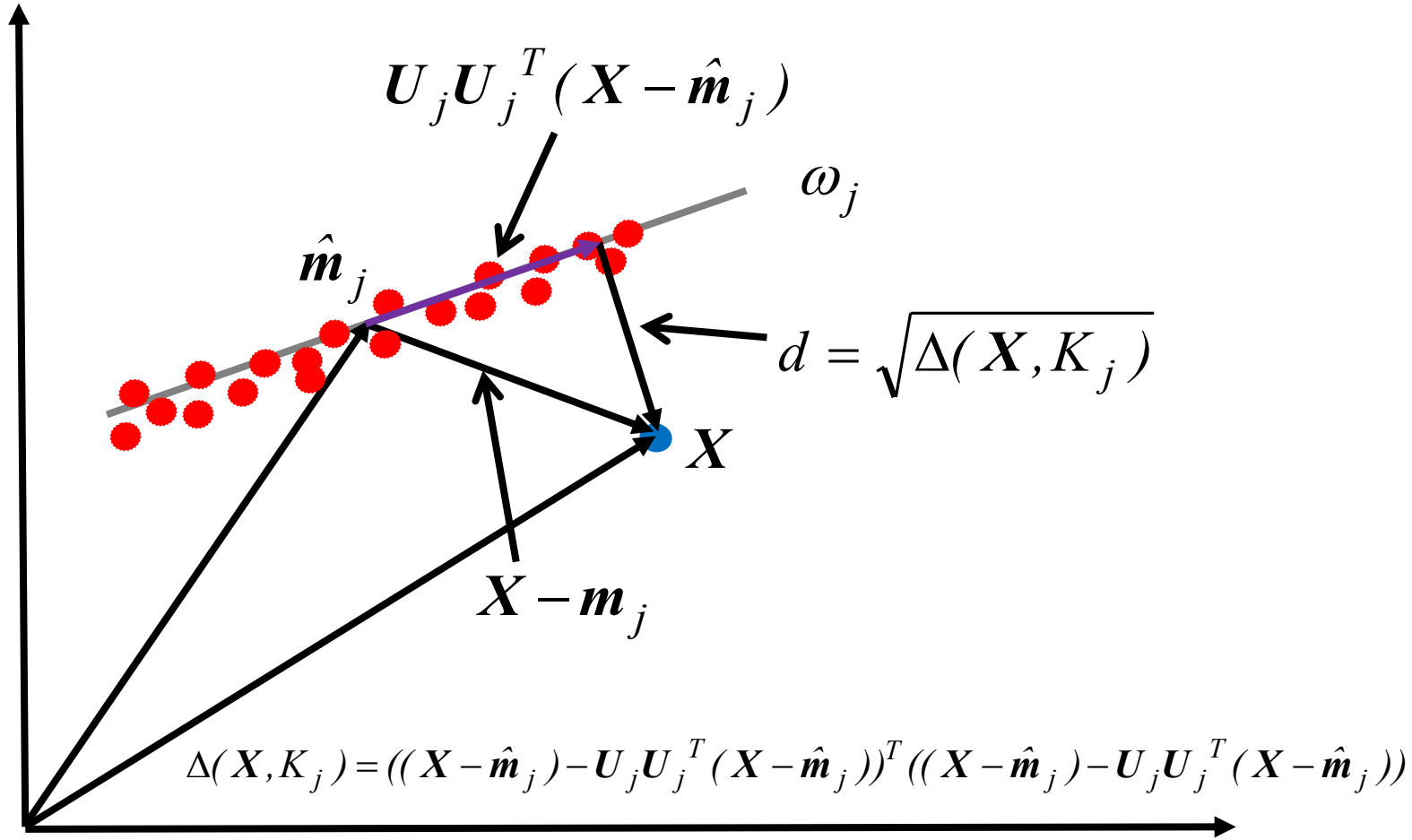
部分主轴系统　　规格化特征向量

其所有列向量张成一个特征子空间
给出了样本在特征子空间上的投影

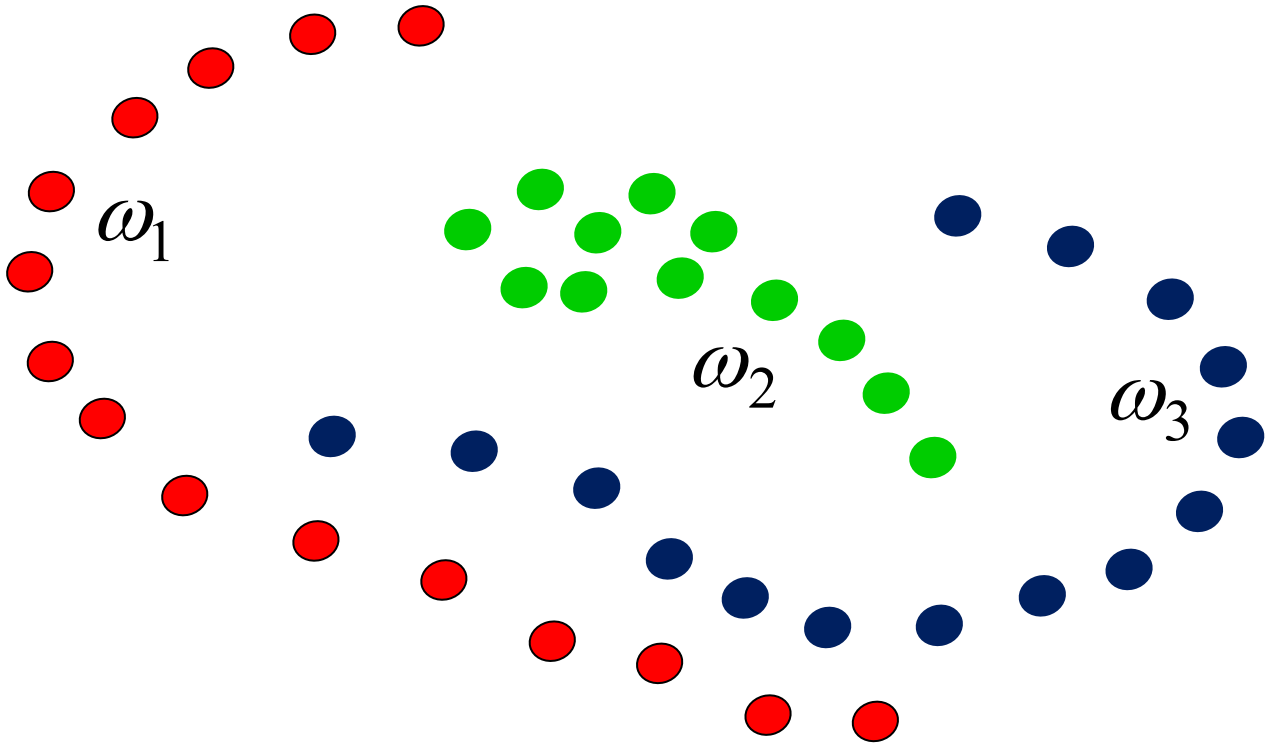
§5.5 动态聚类算法

常用的核函数

- 主轴核函数



§5.6 近邻函数值准则聚类算法



当核不能简单表示时，聚类算法将遭遇困难。

§5.6 近邻函数值准则聚类算法

近邻系数和近邻函数值

考虑任意两个样本 X_i 和 X_k ：

若 X_i 是 X_k 的第 I 个近邻，则称 X_i 对 X_k 的近邻系数为 I ；

若 X_k 是 X_i 的第 k 个近邻，则称 X_k 对 X_i 的近邻系数为 k 。

- 两个样本 X_i 和 X_k 的近邻函数值 $\alpha_{ik} = I + K - 2$
- 近邻函数值 ⟶ 连接损失

  为避免仅包含一个样本的聚类，规定每个样本到自身的连接损失为 $2N$ 。

$$\underset{i,k;i\neq k}{Maximize}\ \alpha_{ik} = (N-1)+(N-1)-2 = 2N-4$$

- $$\gamma_{pq} = \underset{X_i\in\omega_p, X_k\in\omega_q}{Minimize}\ \alpha_{ik}$$

  最小连接损失 $\gamma_{pq_{min}} = \underset{q, q\neq p}{Minimize}\ \gamma_{pq}$

§5.6 近邻函数值准则聚类算法

近邻系数和近邻函数值

若 $\gamma_{pq_{min}} = \underset{q,q\neq p}{Minimize} \ \gamma_{pq}$

问：ω_p 和 $\omega_{q_{min}}$ 什么关系？ 距离 ω_p 最近的类别为 $\omega_{q_{min}}$ 。

类内样本间连接损失的最大值

$$Max \ \gamma_p = \underset{\boldsymbol{X}_i\in\omega_p}{Maximize} \ \underset{\boldsymbol{X}_k\in\omega_p}{Minimize} \ \alpha_{ik}$$

$$Max \ \gamma_{q_{min}} = \underset{\boldsymbol{X}_i\in\omega_p}{Maximize} \ \underset{\boldsymbol{X}_k\in\omega_{q_{min}}}{Minimize} \ \alpha_{ik}$$

类内损失的定量描述

$$L_{\omega_p} = \sum_{\boldsymbol{X}_i\in\omega_p} \underset{\boldsymbol{X}_k\in\omega_p}{Minimize} \ \alpha_{ik}$$

$$L_w = \sum_{p=1}^{c} L_{\omega_p}$$ 总的类内损失

§ 5.6 近邻函数值准则聚类算法

类间损失的定量描述　　类内样本间的最大连接损失

类间样本间的最小连接损失

$$L_{\omega_p,\omega_{q_{min}}} = \begin{cases} -[(\gamma_{pq_{min}} - Max\ \gamma_p) + (\gamma_{pq_{min}} - Max\ \gamma_{q_{min}})] & \text{若} \begin{cases} \gamma_{pq_{min}} > Max\ \gamma_p \\ \gamma_{pq_{min}} > Max\ \gamma_{q_{min}} \end{cases} \\ \gamma_{pq_{min}} + Max\ \gamma_p & \text{若} \begin{cases} \gamma_{pq_{min}} \le Max\ \gamma_p \\ \gamma_{pq_{min}} > Max\ \gamma_{q_{min}} \end{cases} \\ \gamma_{pq_{min}} + Max\ \gamma_{q_{min}} & \text{若} \begin{cases} \gamma_{pq_{min}} > Max\ \gamma_p \\ \gamma_{pq_{min}} \le Max\ \gamma_{q_{min}} \end{cases} \\ \gamma_{pq_{min}} + Max\ \gamma_p + Max\ \gamma_{q_{min}} & \text{若} \begin{cases} \gamma_{pq_{min}} \le Max\ \gamma_p \\ \gamma_{pq_{min}} \le Max\ \gamma_{q_{min}} \end{cases} \end{cases}$$

在后续处理中考虑合并。

其值大小反映了现行聚类结果的合理性。

总的类间损失　$L_b = \sum_{p=1}^{c} L_{\omega_p,\omega_{q_{min}}}$

§5.6 近邻函数值准则聚类算法

近邻函数值准则函数 $J_N = L_w + L_b$ ➡ 最小化

Step1. 读入 $\mathcal{X} = \{\boldsymbol{X}_1, \boldsymbol{X}_2, ..., \boldsymbol{X}_N\}$，并计算 $\boldsymbol{D} = [d_{ij}]$。

Step2. 计算近邻系数矩阵 $\boldsymbol{M} = [m_{ij}]$。

Step3. 计算近邻函数值矩阵 $\boldsymbol{A} = [\alpha_{ij}]$。 （$\alpha_{ij} = m_{ij} + m_{ji} - 2$）

Step4. 找出每行中的最小元素，并把该最小元素所对应的两个样本点连接起来，形成初始聚类 $\omega_p, p = 1,2,...c$ 。

Step5. 对每一个聚类 $\omega_p, p = 1,2,...c$，确定在近邻函数值意义下与之“相距”最近的类别 $\omega_{q_{min}}$，并计算

$$Max\ \gamma_p = \underset{\boldsymbol{X}_i \in \omega_p}{Maximize}\ \underset{\boldsymbol{X}_k \in \omega_p}{Minimize}\ \alpha_{ik}$$

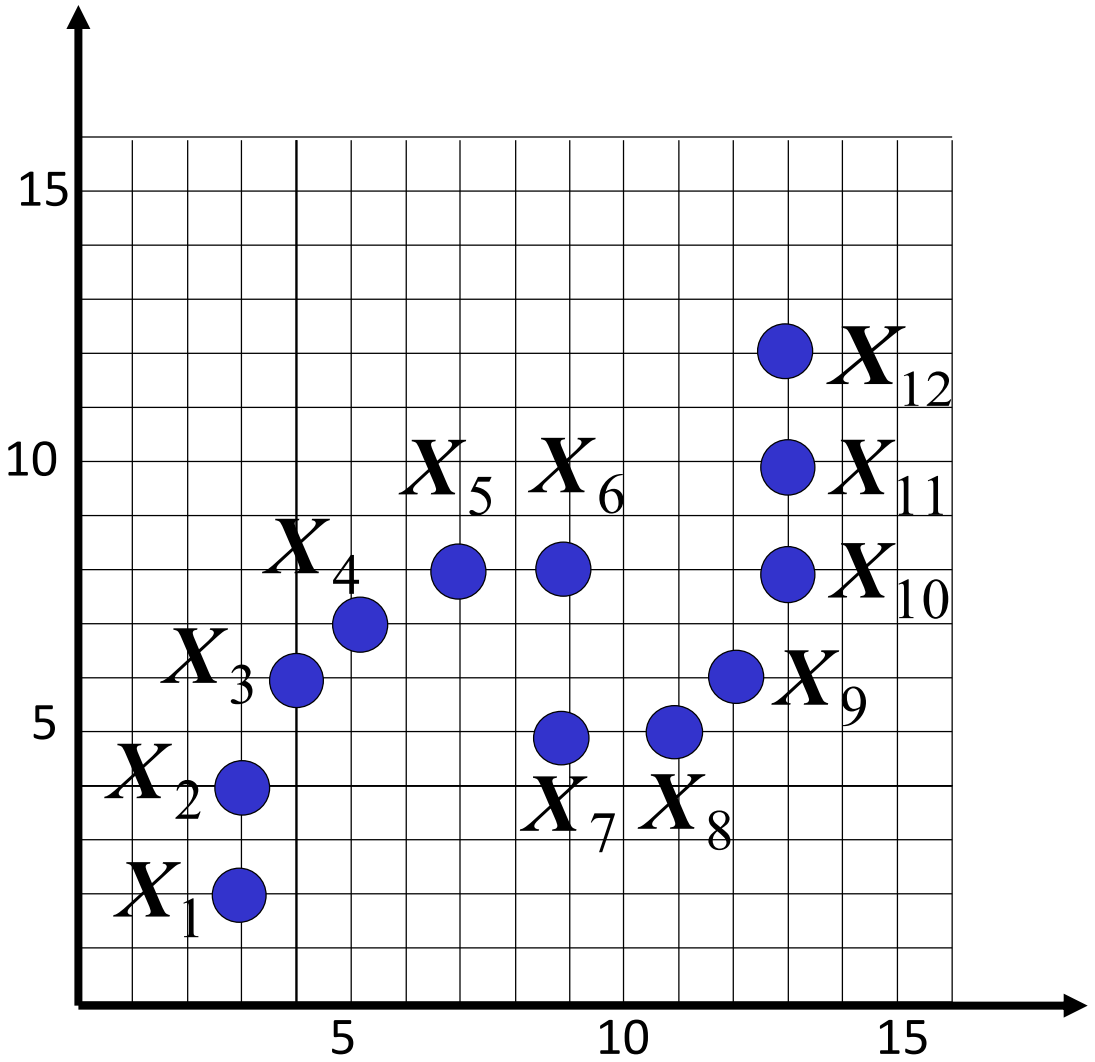
$$Max\ \gamma_{q_{min}} = \underset{\boldsymbol{X}_i \in \omega_p}{Maximize}\ \underset{\boldsymbol{X}_k \in \omega_{q_{min}}}{Minimize}\ \alpha_{ik}$$

若 $\gamma_{pq_{min}} \le Max\ \gamma_p (or \le Max\ \gamma_{q_{min}})$，则将两者合并。

Step6. 计算 J_N，若其值较上一次迭代没有减少，则算法结束；否则，返回步骤**<5>**。

§ 5.6 近邻函数值准则聚类算法

近邻函数值准则聚类算法举例



§ 5.6 近邻函数值准则聚类算法

近邻函数值准则聚类算法举例

	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}
X_1	24	0	4	7	13	14	12	14	16	16	17	18
X_2	0	24	2	4	6	12	10	13	15	15	16	17
X_3	4	2	24	0	4	11	10	12	15	15	16	18
X_4	7	4	0	24	2	7	8	11	14	14	15	16
X_5	13	6	4	2	24	0	5	6	9	11	11	12
X_6	14	12	11	7	0	24	2	4	5	6	7	9
X_7	12	10	10	8	5	2	24	1	4	9	13	15
X_8	14	13	12	11	6	4	1	24	0	4	7	11
X_9	16	15	15	14	9	5	4	0	24	2	5	9
X_{10}	16	15	15	14	11	6	9	4	2	24	0	4
X_{11}	17	16	16	15	11	7	13	7	5	0	24	0
X_{12}	18	17	18	16	12	9	15	11	9	4	0	24

$\omega_1=\{X_1,X_2\}$ $\omega_2=\{X_3,X_4\}$ $\omega_3=\{X_5,X_6\}$ $\omega_4=\{X_7,X_8,X_9\}$ $\omega_5=\{X_{10},X_{11},X_{12}\}$

§ 5.6 近邻函数值准则聚类算法

近邻函数值准则聚类算法举例

	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}
X_1	24	0	4	7	13	14	12	14	16	16	17	18
X_2	0	24	2	4	6	12	10	13	15	15	16	17
X_3	4	2	24	0	4	11	10	12	15	15	16	18
X_4	7	4	0	24	2	7	8	11	14	14	15	16
X_5	13	6	4	2	24	0	5	6	9	11	11	12
X_6	14	12	11	7	0	24	2	4	5	6	7	9
X_7	12	10	10	8	5	2	24	1	4	9	13	15
X_8	14	13	12	11	6	4	1	24	0	4	7	11
X_9	16	15	15	14	9	5	4	0	24	2	5	9
X_{10}	16	15	15	14	11	6	9	4	2	24	0	4
X_{11}	17	16	16	15	11	7	13	7	5	0	24	0
X_{12}	18	17	18	16	12	9	15	11	9	4	0	24

$\omega_1 \quad Max \ \gamma_1 = 0$

$\omega_2 \quad Max \ \gamma_2 = 0$

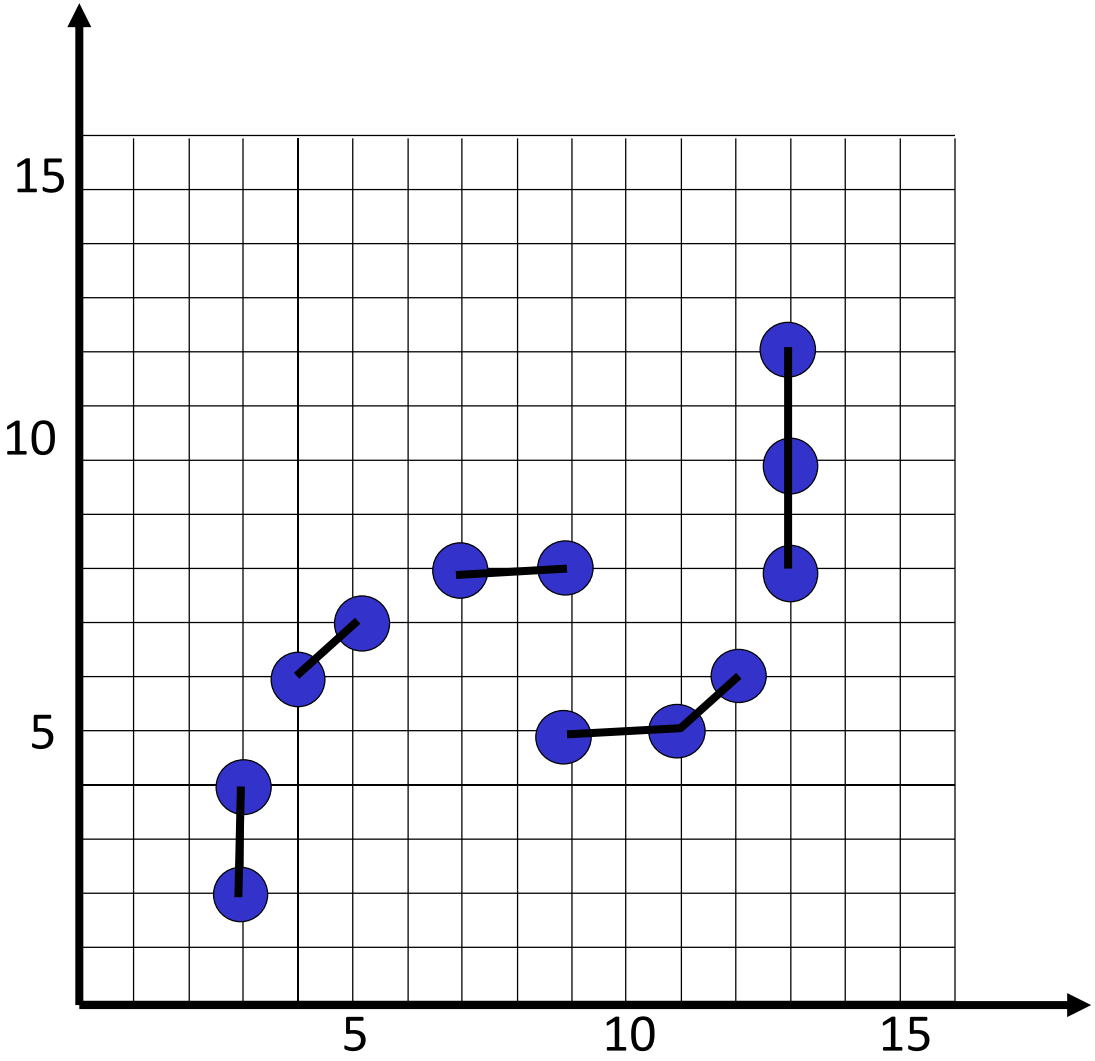
$\omega_3 \quad Max \ \gamma_3 = 0$

$\omega_4 \quad Max \ \gamma_4 = 1$

$\omega_5 \quad Max \ \gamma_5 = 0$

§ 5.6 近邻函数值准则聚类算法

近邻函数值准则聚类算法举例

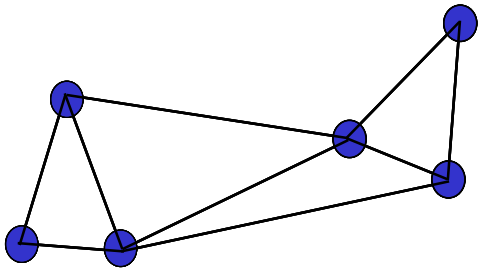


§5.7 最小张树聚类算法

聚类问题 ➡ 图论问题

相关概念简介

- 图



顶点元素集合

$G = \{V, E\}$ ⬅ 二元组

边元素集合

连接V中两个顶点 { 有向 / 无向

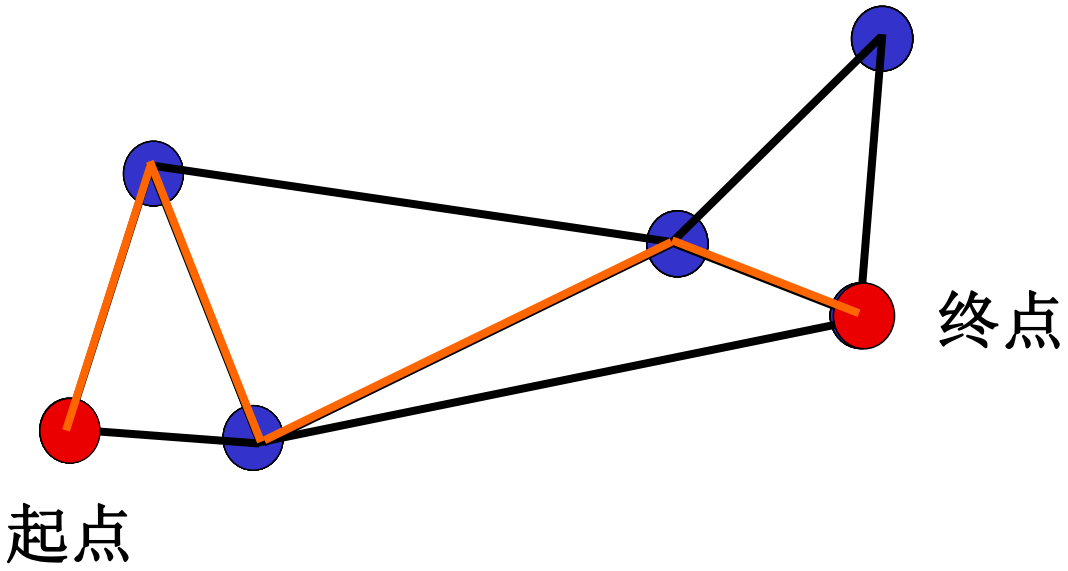
§5.7 最小张树聚类算法

聚类问题 ➡ 图论问题

相关概念简介

- 通路

  由一些边相互连接而成的一个序列



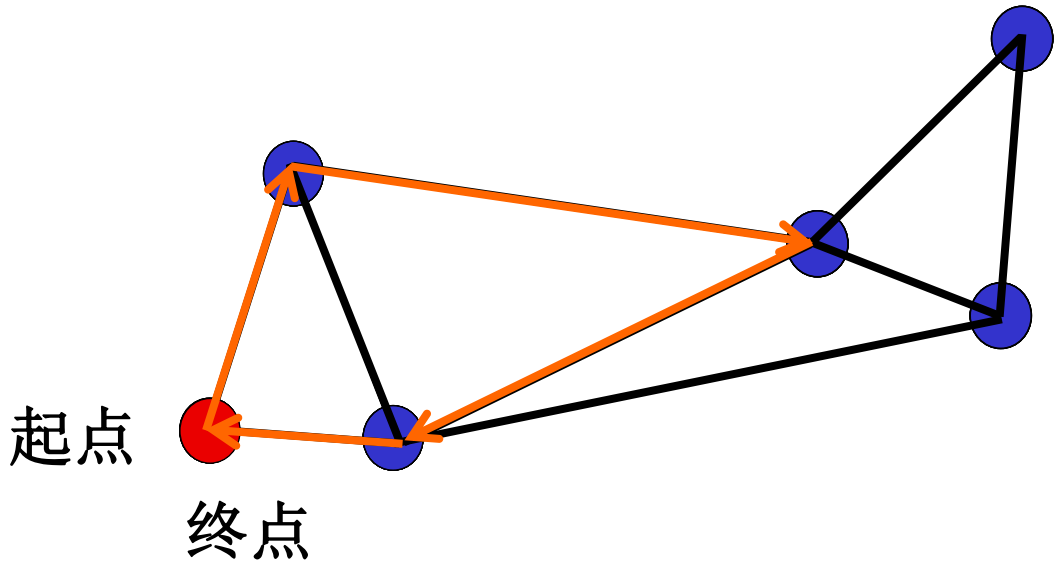
§5.7 最小张树聚类算法

聚类问题 ➡ 图论问题

相关概念简介

- 圈（回路）

  起点与终点为同一个点的通路



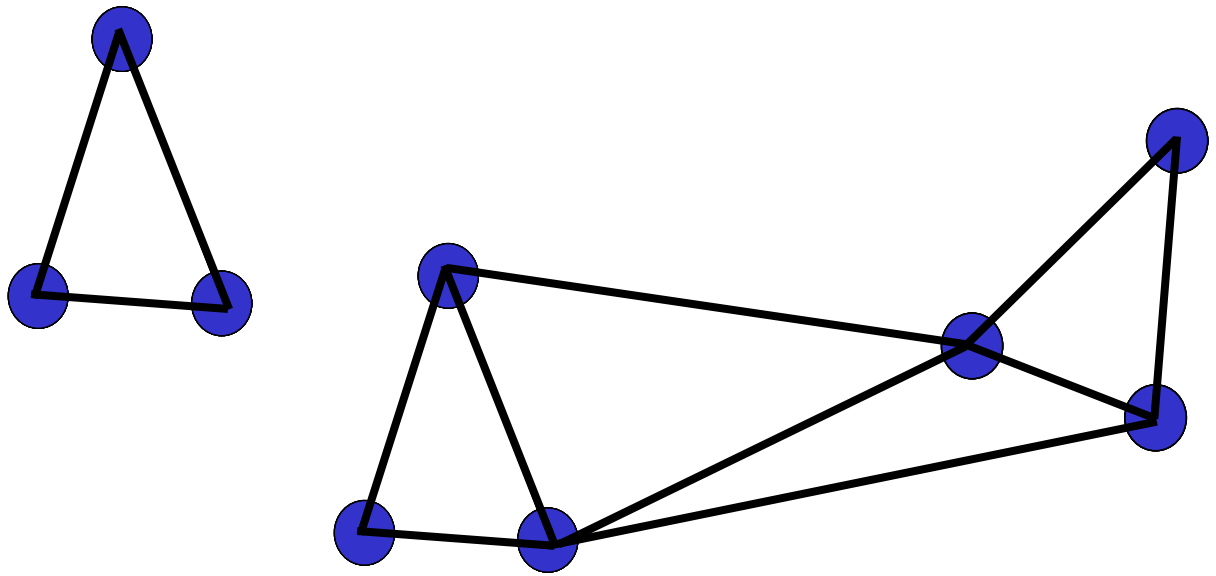
§5.7 最小张树聚类算法

聚类问题 ➡ 图论问题

相关概念简介

- 连通图

  一个图的任意两个不同顶点之间至少存在一条通路



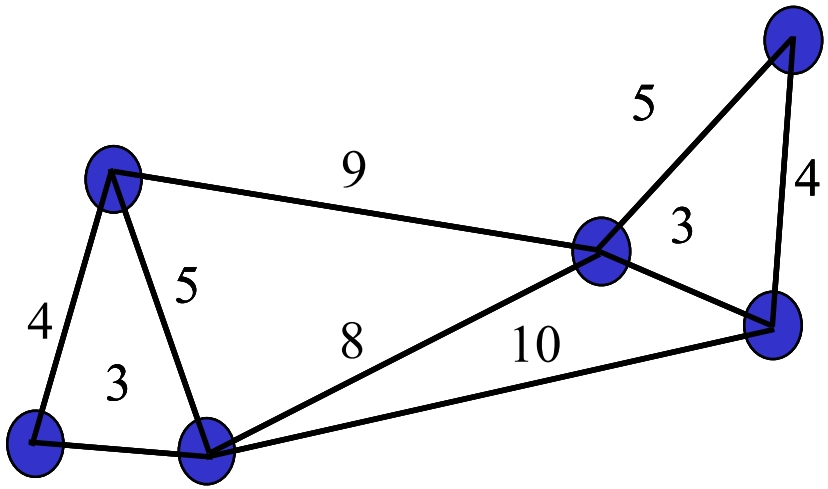
§5.7 最小张树聚类算法

聚类问题 ➡ 图论问题

相关概念简介

- 加权图

  每一条边赋一个权值



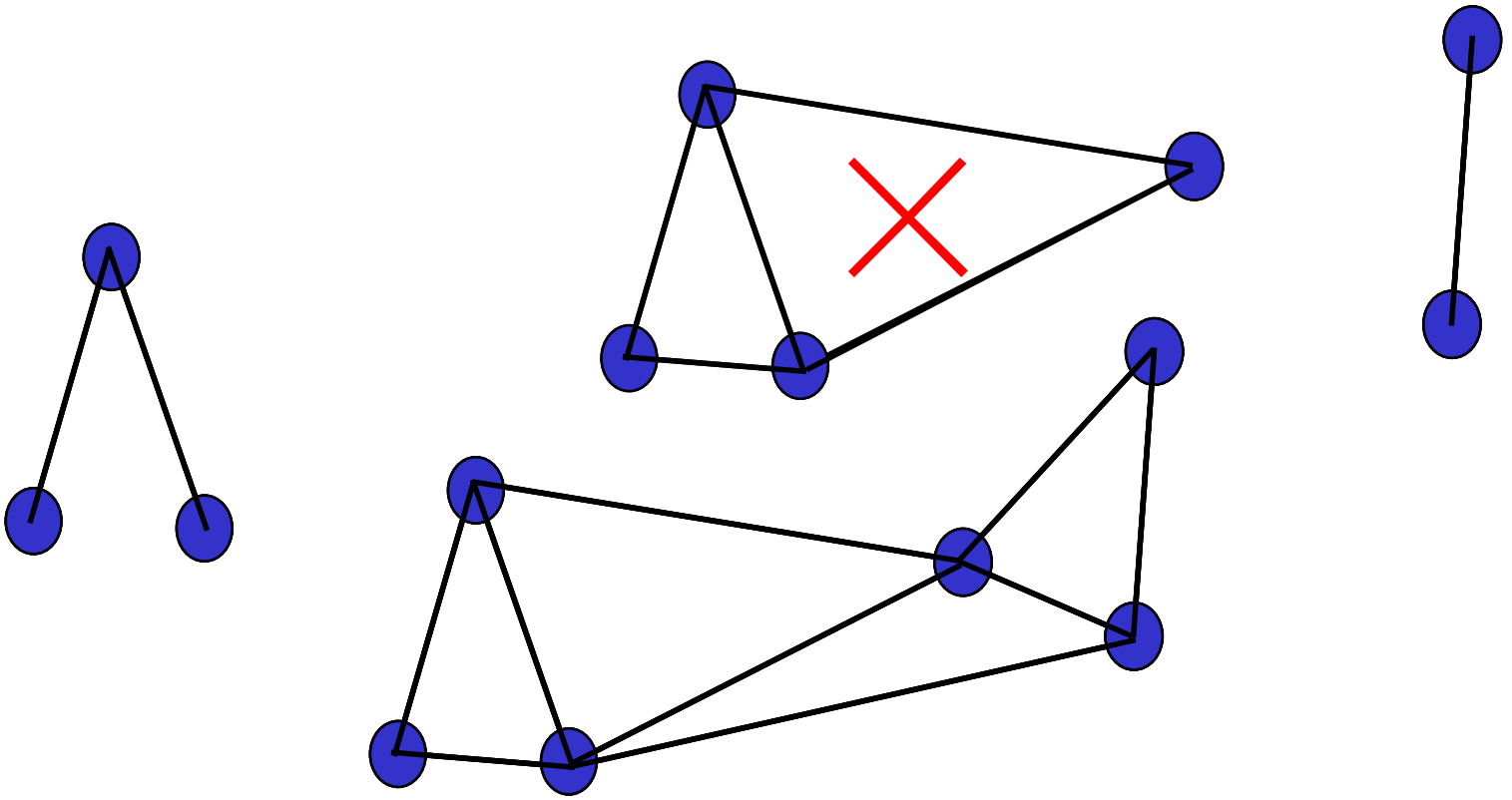
§5.7 最小张树聚类算法

聚类问题 ➡ 图论问题

相关概念简介

- 树

  至少包含两个顶点的连通的、无圈的子图



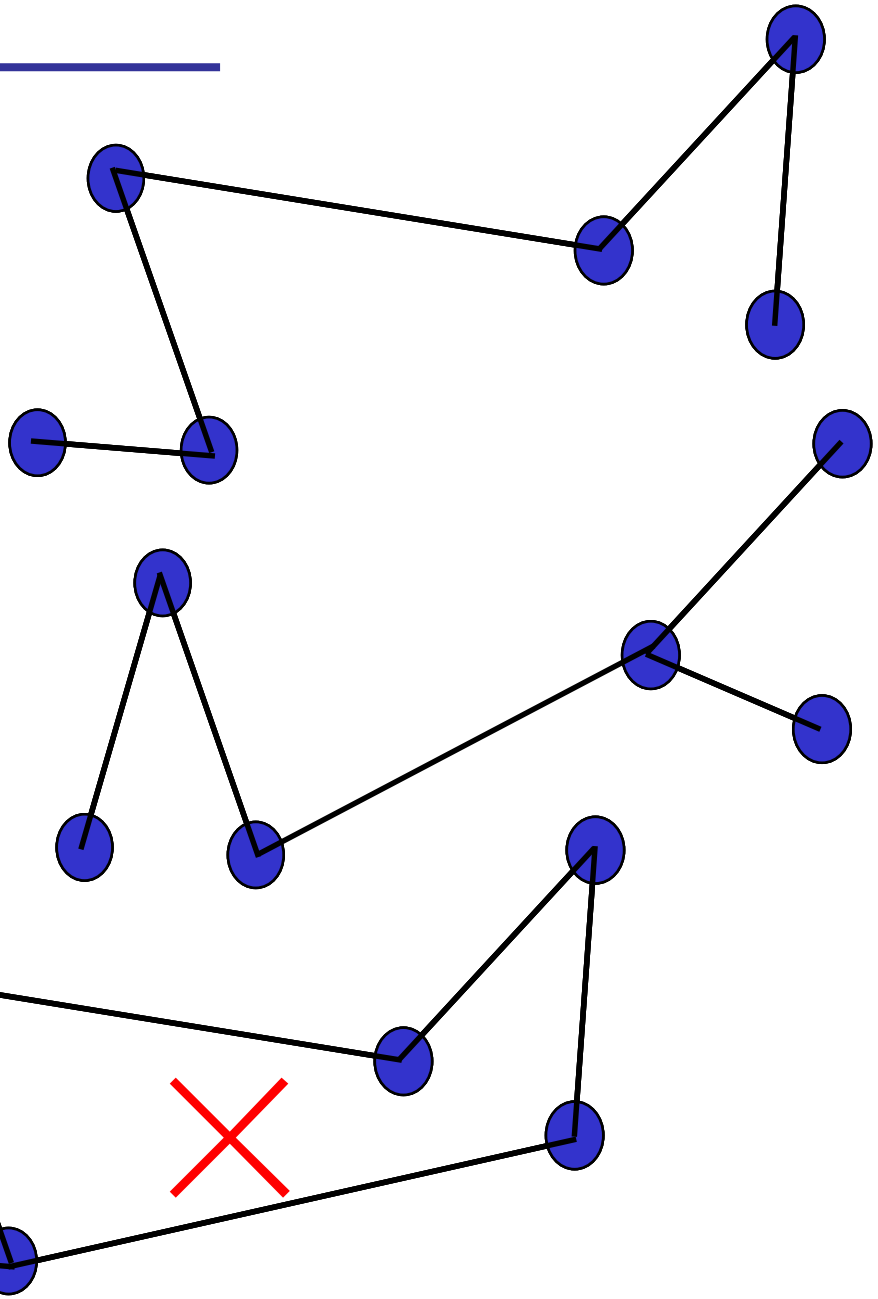
§5.7 最小张树聚类算法

聚类问题 ➡ 图论问题

相关概念简介

- 张树

  包含一个图所有顶点的树



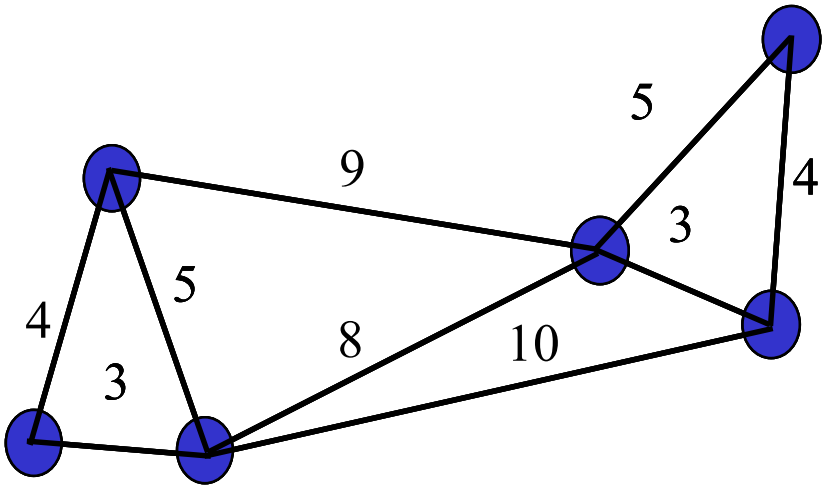
§5.7 最小张树聚类算法

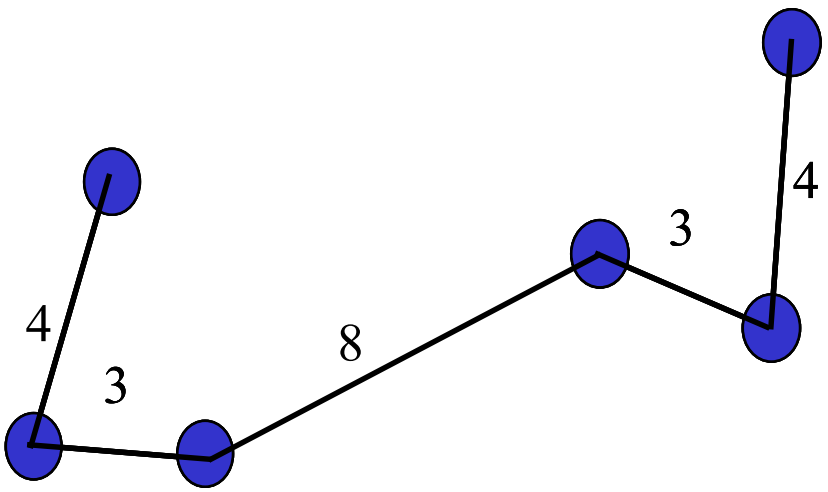
聚类问题 ➡ 图论问题

相关概念简介　　无向加权图

- 最小张树

  边的权值之和最小的张树

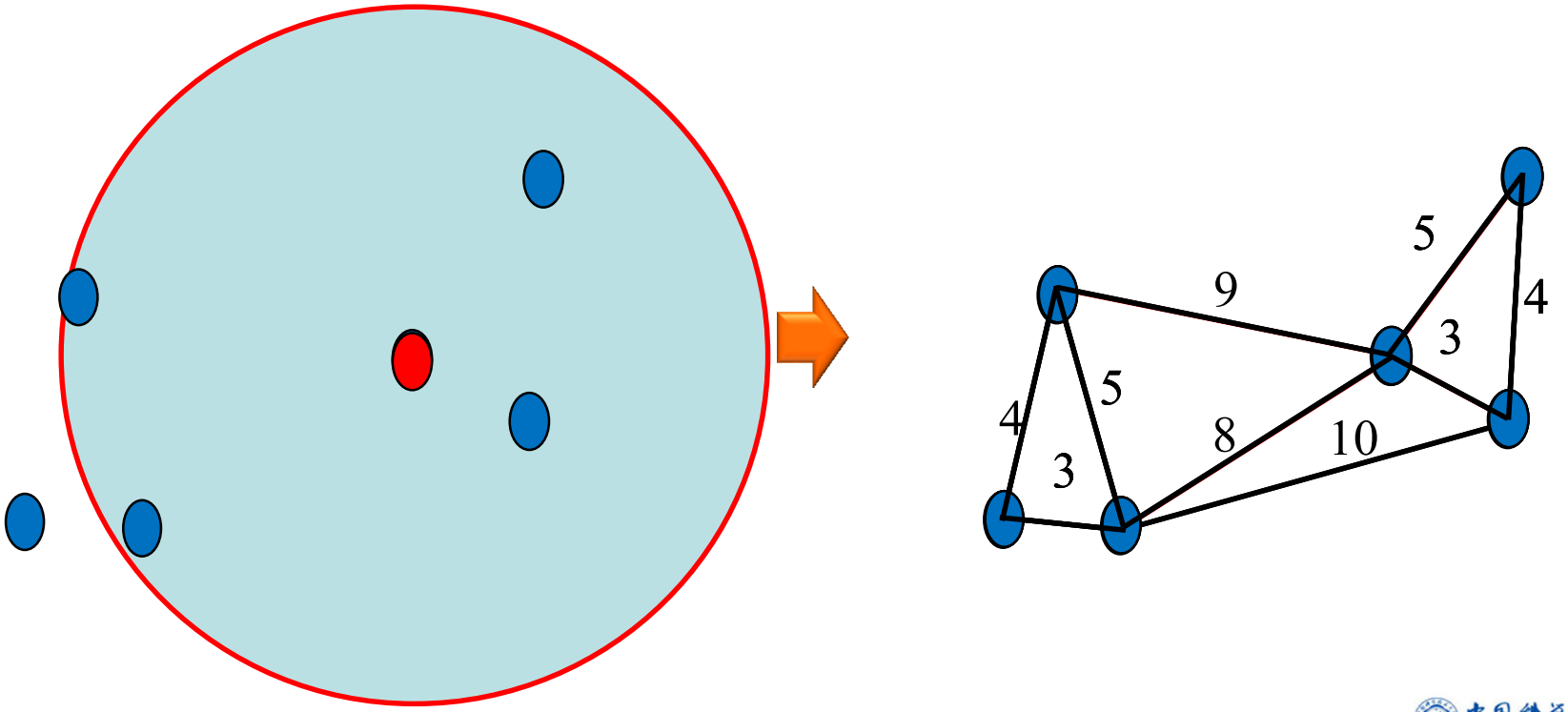




§5.7 最小张树聚类算法

如何得到一个样本集合的无向加权图表示？

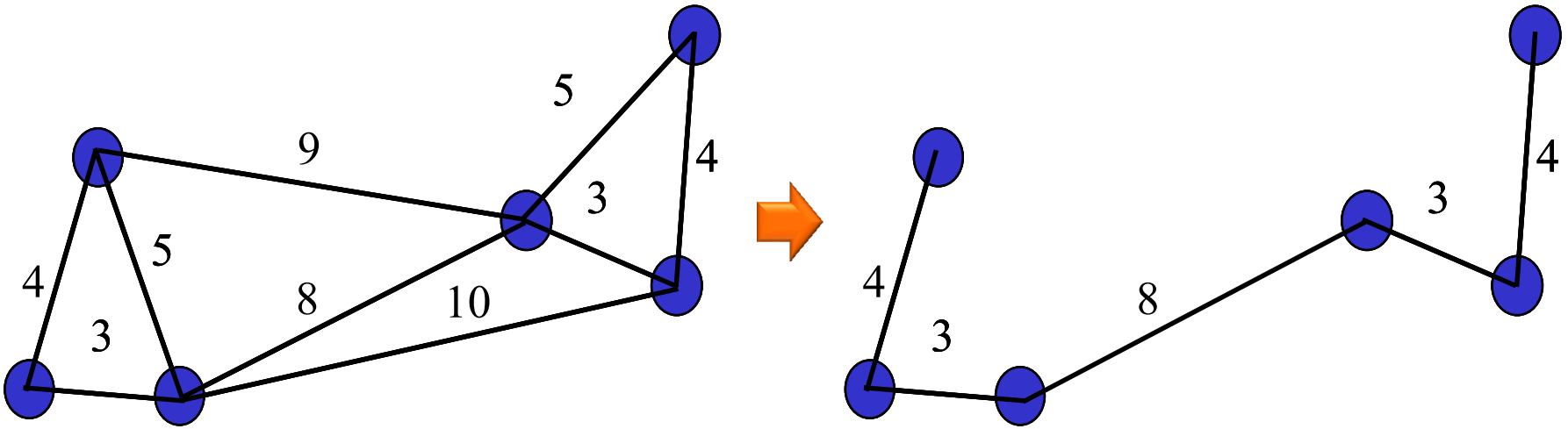
1. 将样本集合中的每个样本作为图的一个顶点；
2. 依据某种方法在顶点之间建立连接，确定具有连接关系的两个顶点之间用一条边相连；
3. 依据距离等相似性测度，为每条边赋一个权值。



§5.7 最小张树聚类算法

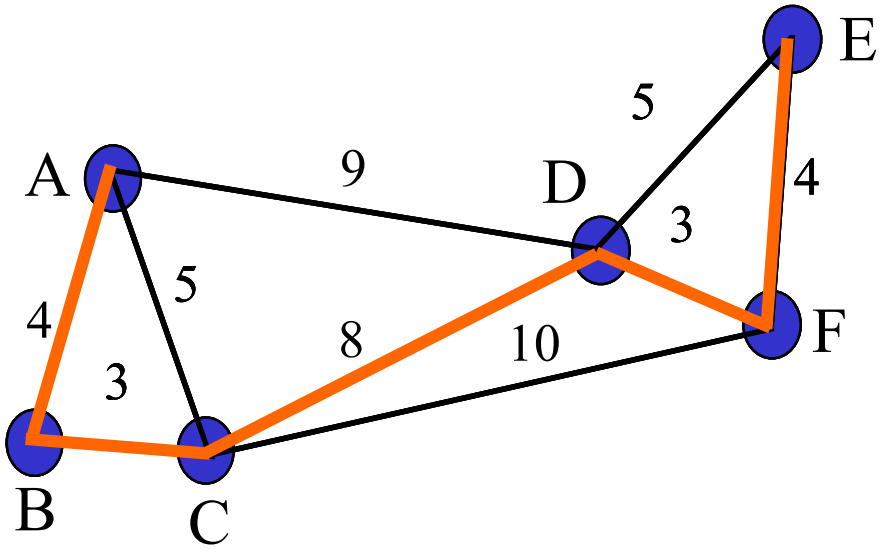
如何获得一个无向加权图的最小张树表示？

1. 输入待处理的加权图；
2. 按权值递增顺序对加权图的边集进行排序；
3. 选择权值最小的边为起始边；
4. 按权值小优先的原则从边集中选择边，使构成的图是连通的、无圈的；若已无边可选，则算法结束，否则重复4。



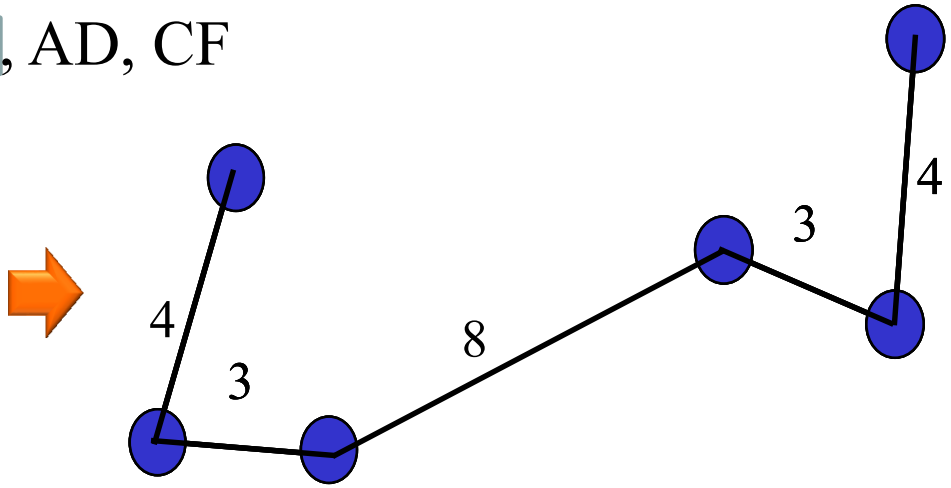
§ 5.7 最小张树聚类算法

获得无向加权图最小张树表示的实例



BC, DF, AB, EF, AC, DE, CD, AD, CF

BC, AB, CD, DF, EF



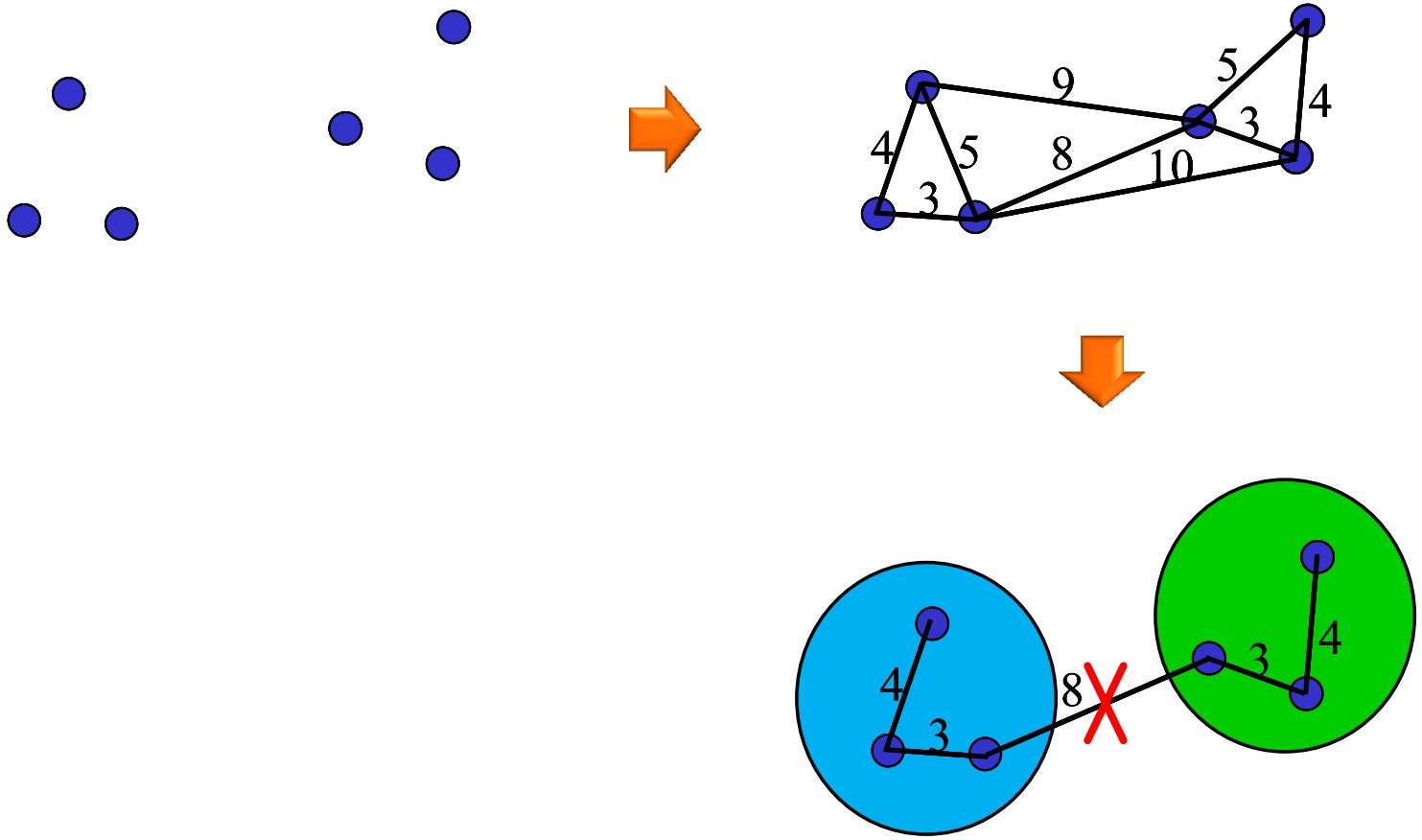
§5.7 最小张树聚类算法

最小张树聚类算法的算法步骤

1. 依据样本集合的加权图表示，生成相应的最小张树；
2. 从最小张树中找到权值最大的一条边，将其从最小张树中去除, 从而将样本集合分割成对应的两个样本子集；
3. 对每一个样本子集，递归地进行步骤 2，不断地将输入样本集合分裂成更多的样本子集, 直至最大边的权值小于给定的阈值为止。

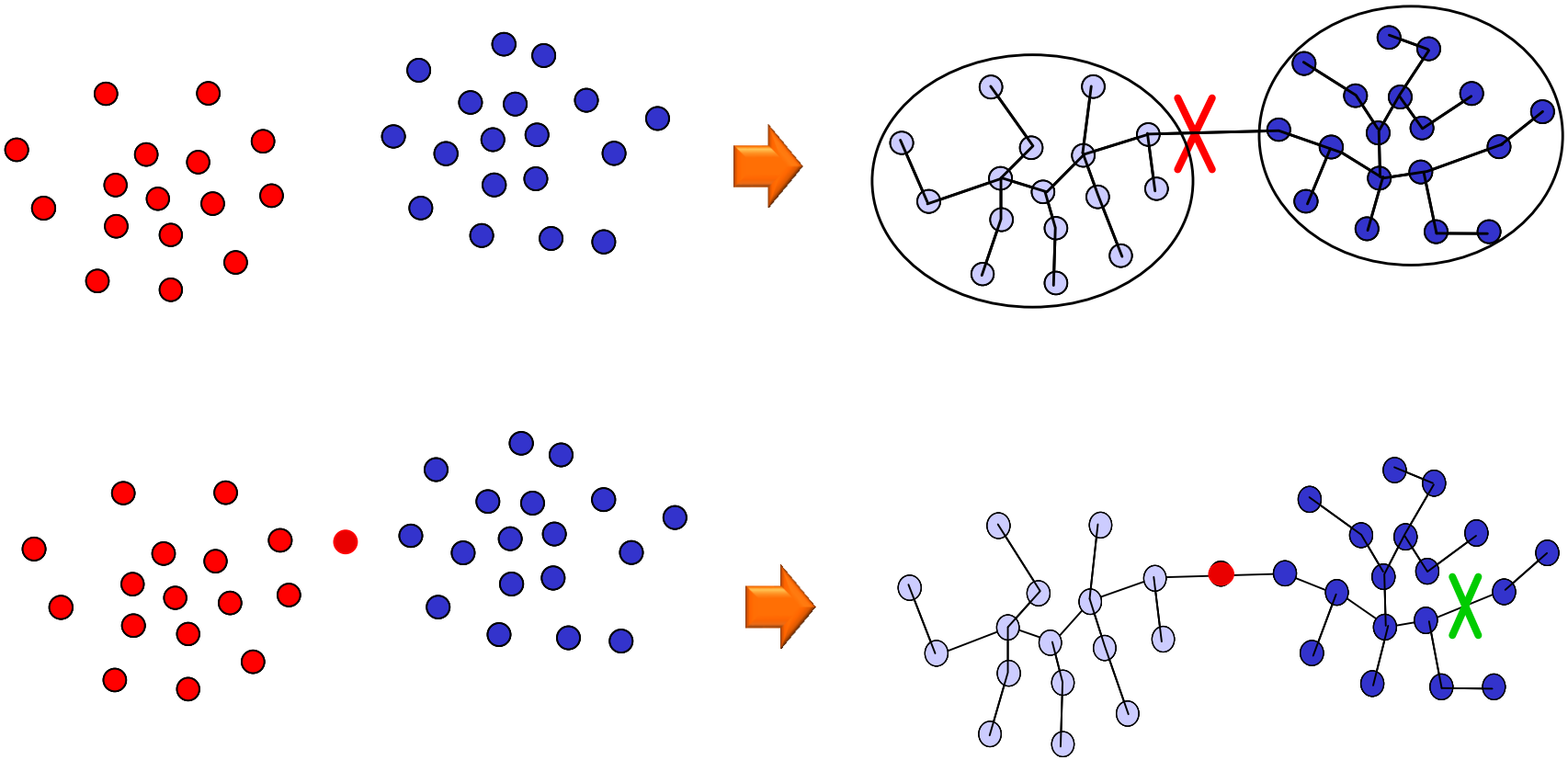
§ 5.7 最小张树聚类算法

最小张树聚类算法的实例



§ 5.7 最小张树聚类算法

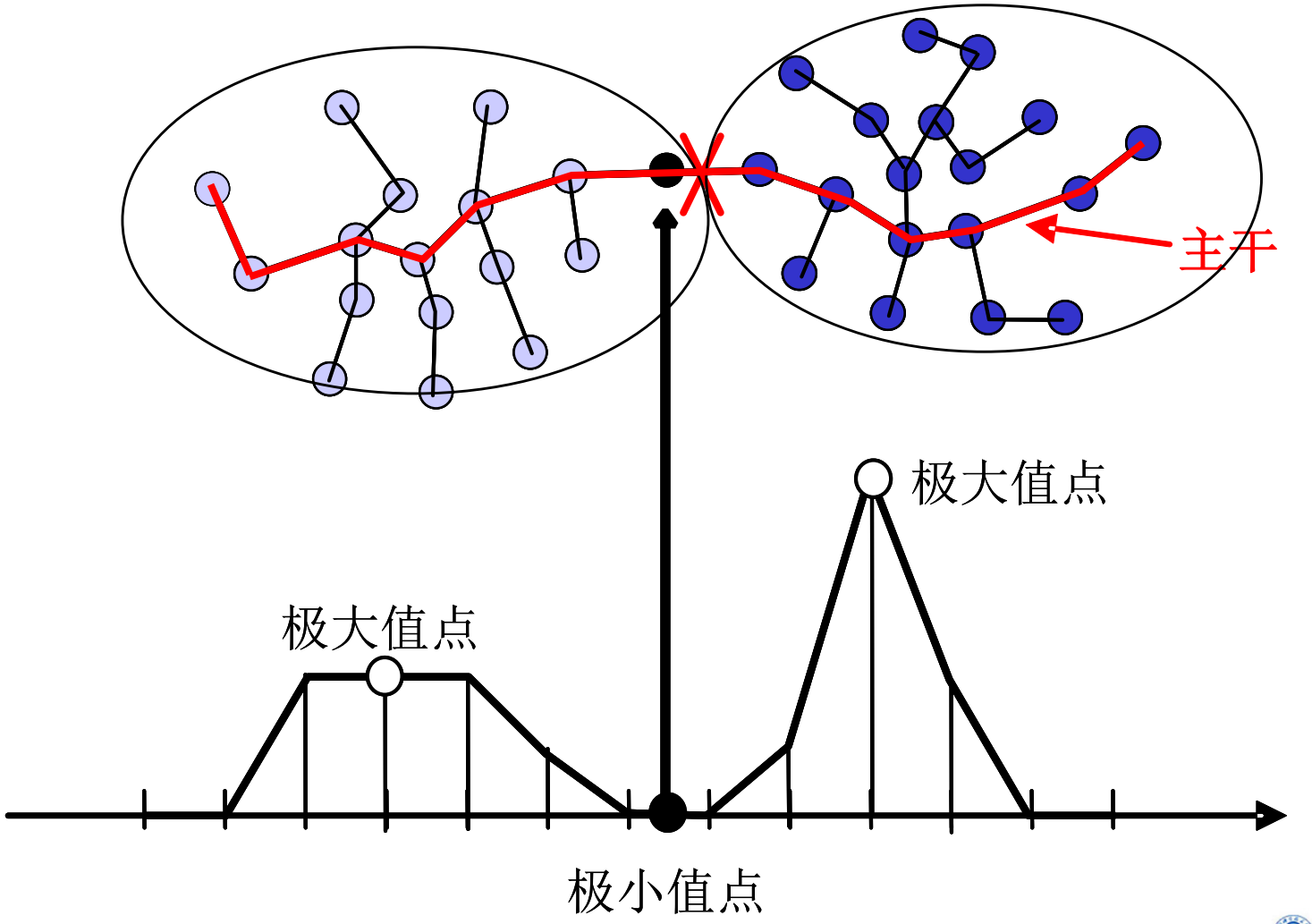
最小张树聚类算法改进



§5.7 最小张树聚类算法

最小张树聚类算法改进

最小张树中最长的一个通路称为树的主干

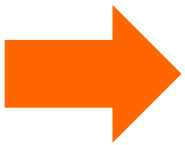


§5.8 新聚类算法选讲

Alex Rodriguez, Alessandro Laio，
Clustering by fast search and find of density peak
Science 344，1492，2014.
DOI: 10.1126/science.1242072

提出背景：

经典的C均值聚类算法通过指定初始聚类中心，并通过迭代计算更新和最终确定聚类中心的方式完成聚类中心的选择。由于每个点都被指派到距离最近的聚类中心，导致其不能对呈非球面分布的样本集进行正确聚类。



对呈任意形状分布的样本集正确聚类

§5.8 新聚类算法选讲

假设：

聚类中心是密度极大值点，即其周围点的密度均比其要低。此外，相比于其他潜在的聚类中心而言，其周围的点距离该聚类中心更近。

定义：

$$\chi(x) = \begin{cases} 1 & if \ x < 0 \\ 0 & otherwise \end{cases}$$

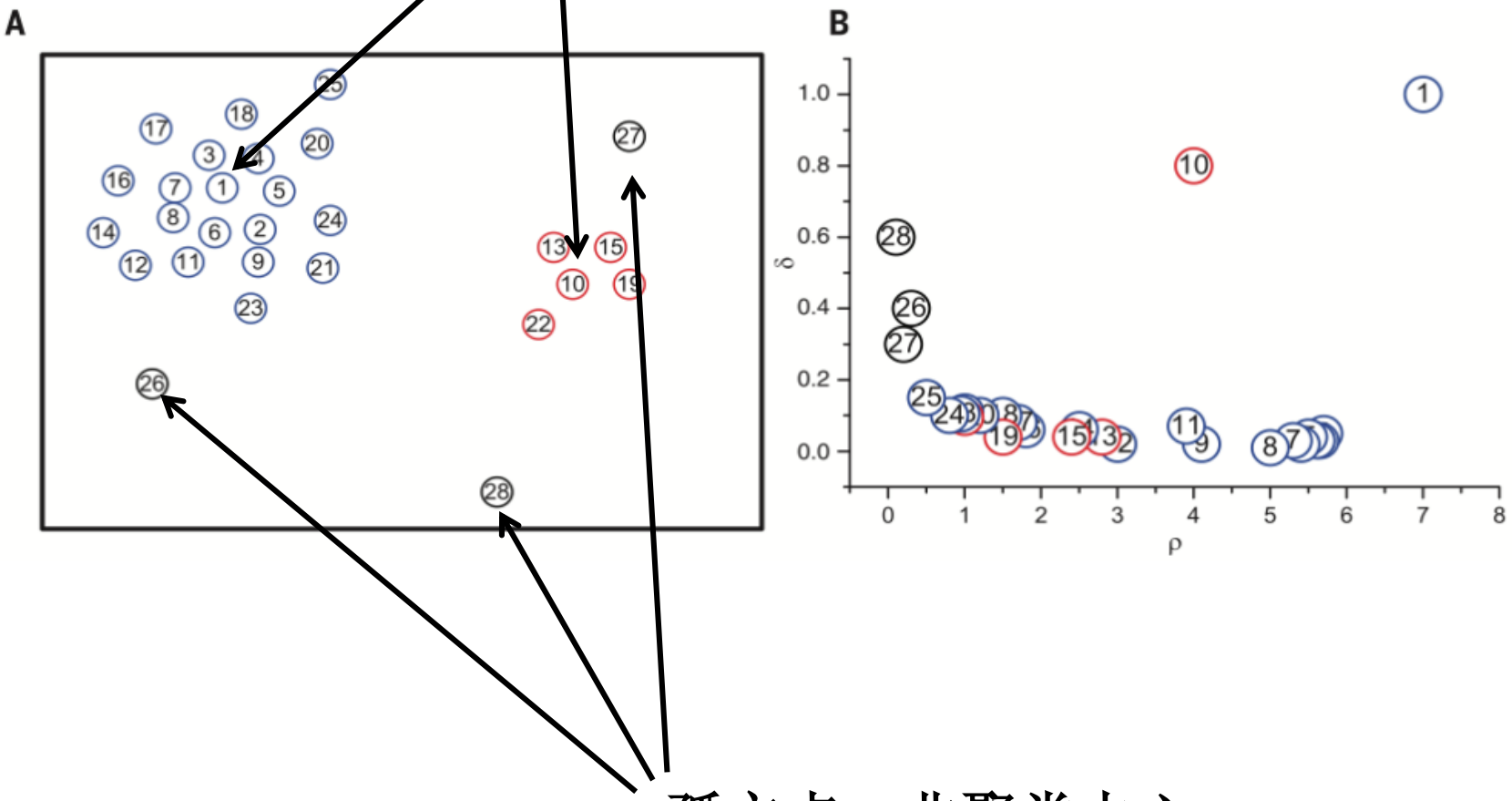
局部密度 $$\rho_i = \sum_j \chi(d_{ij} - d_c)$$

距离 $$\delta_i = \min_{j:\rho_j > \rho_i} (d_{ij})$$

算法思想：首先找出密度的局部最大点，将其作为候选聚类中心。

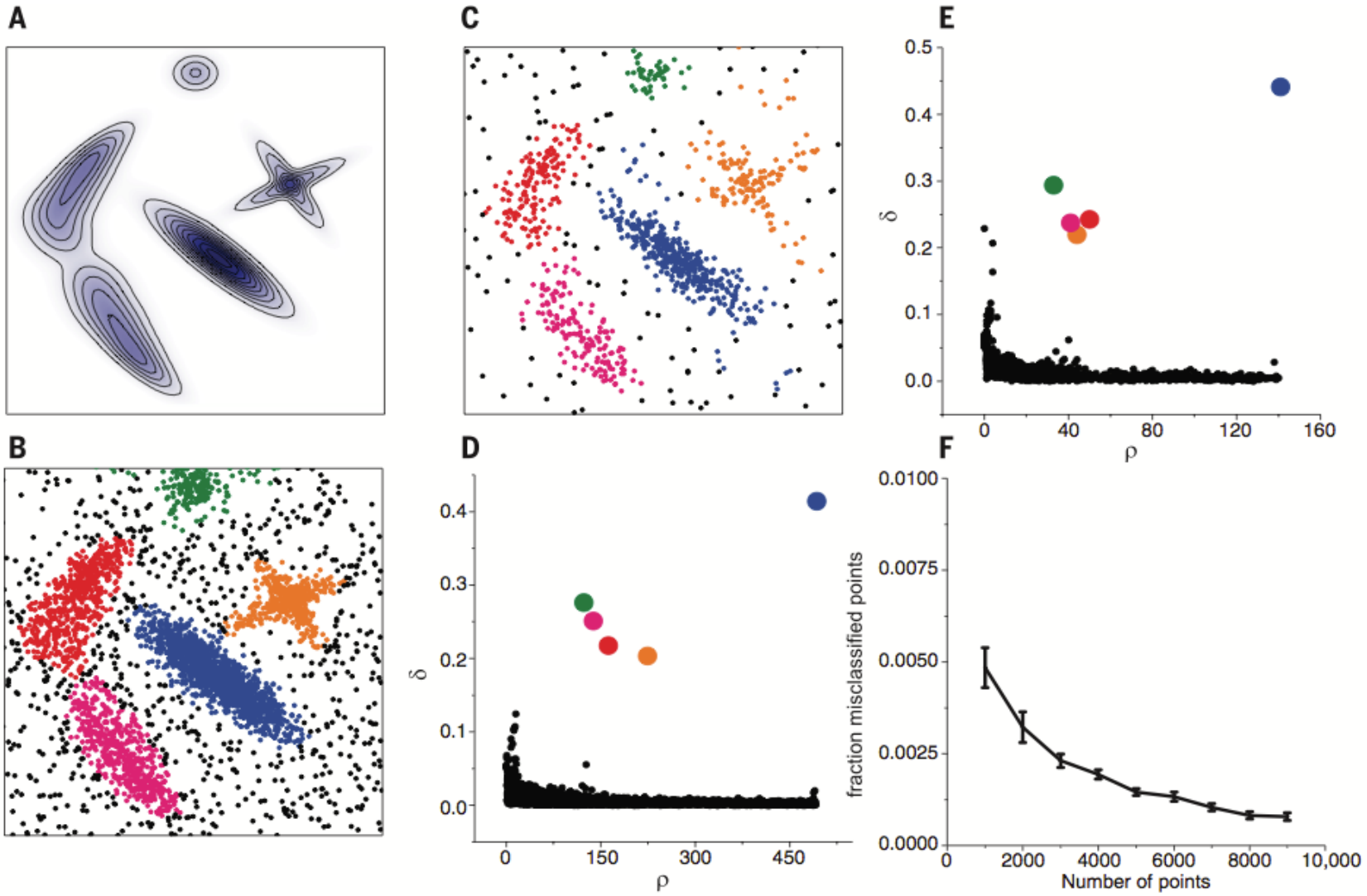
§ 5.8 新聚类算法选讲

候选聚类中心

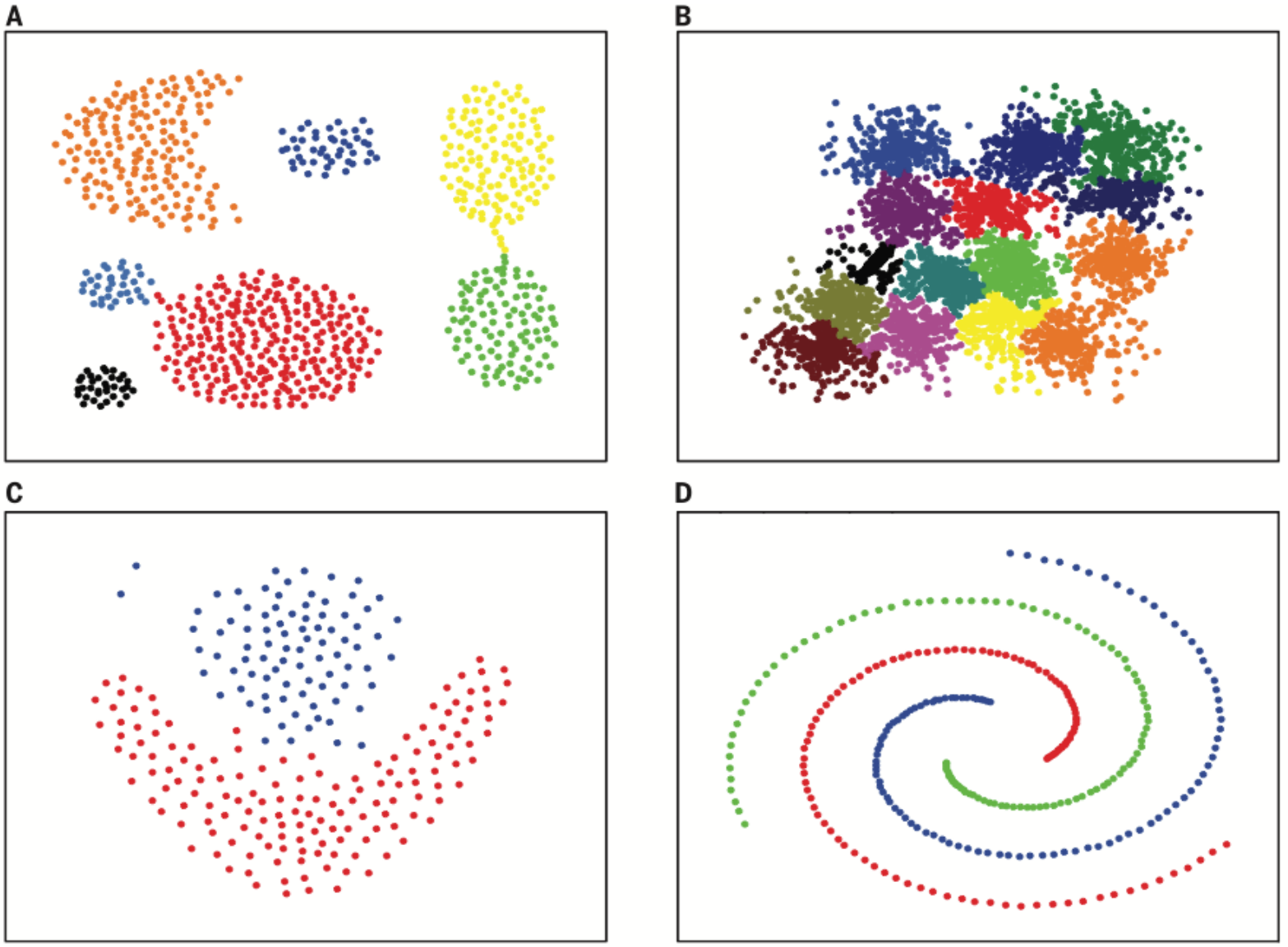


孤立点：非聚类中心

§ 5.8 新聚类算法选讲



§ 5.8 新聚类算法选讲



§ 小结

主要探讨了输入样本集合的非监督聚类问题。

- 基于分裂的聚类算法
- 基于合并的聚类算法
- 动态聚类算法
- 近邻函数值准则聚类算法
- 最小张树聚类算法

若干图片材料取自网络，特此致谢。

谢谢聆听！

中国科学技术大学