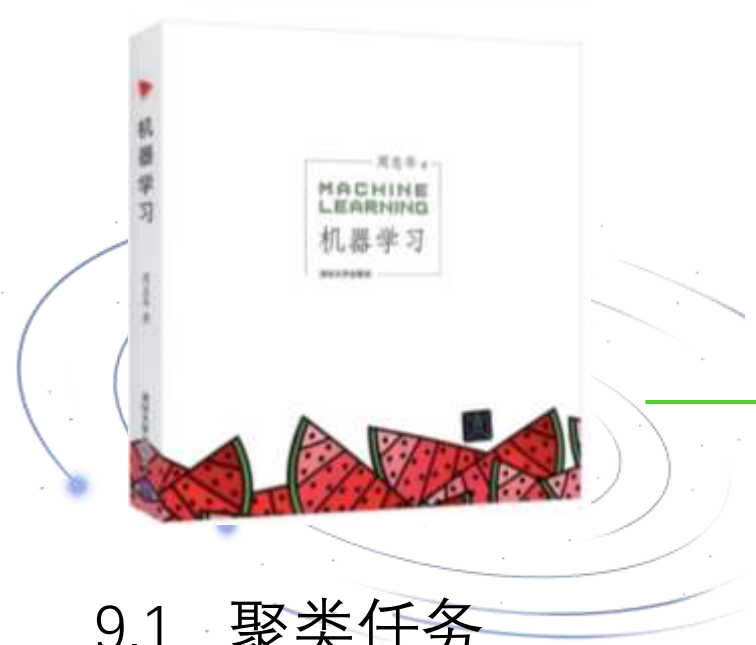


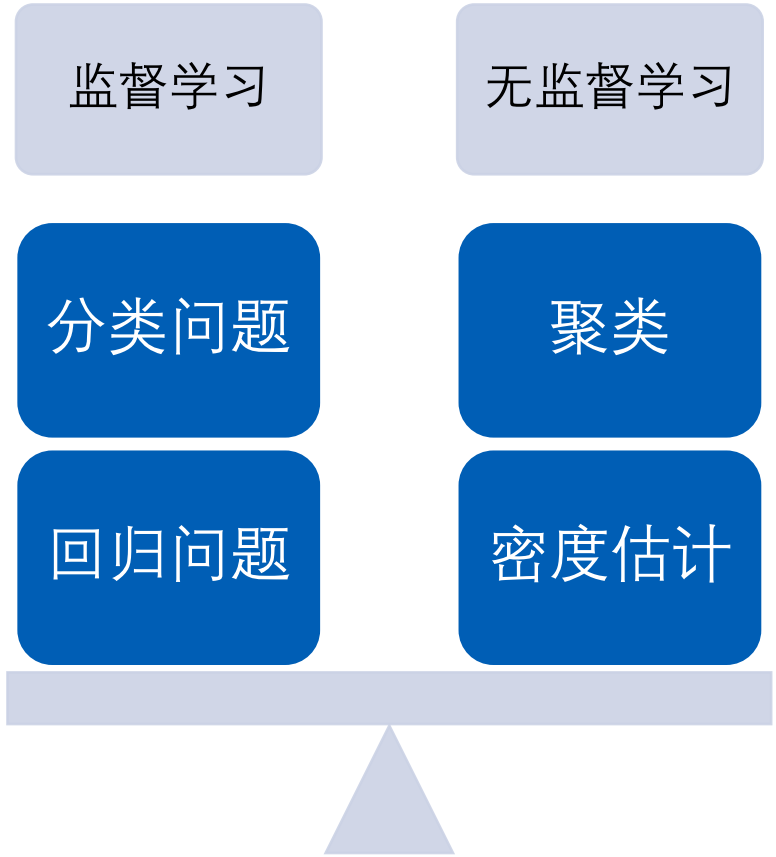
聚类

从“西瓜书”开始

9.1 聚类任务
9.2 性能度量
9.3 距离计算
9.4 原型聚类
9.5 聚类密度
9.6 层次聚类

潘贤润
2024/09/29

9.1 聚类任务



聚类试图将数据集中的样本划分为若干个通常是不相交的子集，每个子集称为一个“簇”(cluster).

9.2 性能度量

- **validity index**
  - 物以类聚
    - 簇内相似度（intra-cluster similarity）高
    - 簇间相似度（inter-cluster similarity）低

聚类性能度量大致有两类. 一类是将聚类结果与某个“参考模型”(reference model)进行比较, 称为“外部指标”(external index); 另一类是直接考察聚类结果而不利用任何参考模型, 称为“内部指标”(internal index).

9.2.1 外部指标

对数据集 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_m\}$, 假定通过聚类给出的簇划分为 $\mathcal{C}=\{C_1,C_2,\ldots,C_k\}$, 参考模型给出的簇划分为 $\mathcal{C}^*=\{C_1^*,C_2^*,\ldots,C_s^*\}$. 相应地, 令 $\boldsymbol{\lambda}$ 与 $\boldsymbol{\lambda}^*$ 分别表示与 $\mathcal{C}$ 和 $\mathcal{C}^*$ 对应的簇标记向量. 我们将样本两两配对考虑, 定义

$$a=|SS|,\quad SS=\{(\boldsymbol{x}_i,\boldsymbol{x}_j)\mid \lambda_i=\lambda_j,\lambda_i^*=\lambda_j^*,i<j\}, \tag{9.1}$$

$$b=|SD|,\quad SD=\{(\boldsymbol{x}_i,\boldsymbol{x}_j)\mid \lambda_i=\lambda_j,\lambda_i^*\neq\lambda_j^*,i<j\}, \tag{9.2}$$

$$c=|DS|,\quad DS=\{(\boldsymbol{x}_i,\boldsymbol{x}_j)\mid \lambda_i\neq\lambda_j,\lambda_i^*=\lambda_j^*,i<j\}, \tag{9.3}$$

$$d=|DD|,\quad DD=\{(\boldsymbol{x}_i,\boldsymbol{x}_j)\mid \lambda_i\neq\lambda_j,\lambda_i^*\neq\lambda_j^*,i<j\}, \tag{9.4}$$

其中集合 SS 包含了在 $\mathcal{C}$ 中隶属于相同簇且在 $\mathcal{C}^*$ 中也隶属于相同簇的样本对, 集合 SD 包含了在 $\mathcal{C}$ 中隶属于相同簇但在 $\mathcal{C}^*$ 中隶属于不同簇的样本对, ……由于每个样本对 $(\boldsymbol{x}_i,\boldsymbol{x}_j)$ $(i<j)$ 仅能出现在一个集合中, 因此有 $a+b+c+d=m(m-1)/2$ 成立.

基于式(9.1)~(9.4)可导出下面这些常用的聚类性能度量外部指标:

- Jaccard 系数(Jaccard Coefficient, 简称 JC)

$$\mathrm{JC}=\frac{a}{a+b+c}.$$

- FM 指数(Fowlkes and Mallows Index, 简称 FMI)

$$\mathrm{FMI}=\sqrt{\frac{a}{a+b}\cdot\frac{a}{a+c}}.$$

- Rand 指数(Rand Index, 简称 RI)

$$\mathrm{RI}=\frac{2(a+d)}{m(m-1)}.$$

9.2.2 内部指标

考虑聚类结果的簇划分 $\mathcal{C}=\{C_1,C_2,\ldots,C_k\}$, 定义

$$\operatorname{avg}(C)=\frac{2}{|C|(|C|-1)}\sum_{1\leqslant i<j\leqslant|C|}\operatorname{dist}(\boldsymbol{x}_i,\boldsymbol{x}_j)\,, \tag{9.8}$$

$$\operatorname{diam}(C)=\max_{1\leqslant i<j\leqslant|C|}\operatorname{dist}(\boldsymbol{x}_i,\boldsymbol{x}_j)\,, \tag{9.9}$$

$$d_{\min}(C_i,C_j)=\min_{\boldsymbol{x}_i\in C_i,\boldsymbol{x}_j\in C_j}\operatorname{dist}(\boldsymbol{x}_i,\boldsymbol{x}_j)\,, \tag{9.10}$$

$$d_{\mathrm{cen}}(C_i,C_j)=\operatorname{dist}(\boldsymbol{\mu}_i,\boldsymbol{\mu}_j)\,, \tag{9.11}$$

其中, $\operatorname{dist}(\cdot,\cdot)$ 用于计算两个样本之间的距离; $\boldsymbol{\mu}$ 代表簇 C 的中心点 $\boldsymbol{\mu}=\frac{1}{|C|}\sum_{1\leqslant i\leqslant|C|}\boldsymbol{x}_i$. 显然, $\operatorname{avg}(C)$ 对应于簇 C 内样本间的平均距离, $\operatorname{diam}(C)$ 对应于簇 C 内样本间的最远距离, $d_{\min}(C_i,C_j)$ 对应于簇 C_i 与簇 C_j 最近样本间的距离, $d_{\mathrm{cen}}(C_i,C_j)$ 对应于簇 C_i 与簇 C_j 中心点间的距离.

基于式(9.8)~(9.11)可导出下面这些常用的聚类性能度量内部指标:

- DB 指数(Davies-Bouldin Index, 简称 DBI)

$$\mathrm{DBI}=\frac{1}{k}\sum_{i=1}^{k}\max_{j\neq i}\left(\frac{\operatorname{avg}(C_i)+\operatorname{avg}(C_j)}{d_{\mathrm{cen}}(\boldsymbol{\mu}_i,\boldsymbol{\mu}_j)}\right).$$

↓内紧 ↑间疏 ↓

- Dunn 指数(Dunn Index, 简称 DI)

$$\mathrm{DI}=\min_{1\leqslant i\leqslant k}\left\{\min_{j\neq i}\left(\frac{d_{\min}(C_i,C_j)}{\max_{1\leqslant l\leqslant k}\operatorname{diam}(C_l)}\right)\right\}.$$

↑间疏 ↑ ↓内紧

显然, DBI 的值越小越好, 而 DI 则相反, 值越大越好.

9.3 距离计算

非负性: $\mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j) \geqslant 0$;

同一性: $\mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j) = 0$ 当且仅当 $\boldsymbol{x}_i = \boldsymbol{x}_j$;

对称性: $\mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \mathrm{dist}(\boldsymbol{x}_j, \boldsymbol{x}_i)$;

直递性: $\mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j) \leqslant \mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{x}_k) + \mathrm{dist}(\boldsymbol{x}_k, \boldsymbol{x}_j)$.

连续属性和存在序关系的离散属性都可以直接参与计算，因为它们都可以反映一种程度，我们称其为“有序属性”

Value Difference Metric

$$\mathrm{VDM}_p(a,b) = \sum_{i=1}^{k} \left| \frac{m_{u,a,i}}{m_{u,a}} - \frac{m_{u,b,i}}{m_{u,b}} \right|^p .$$

- 最常用的距离度量“闵可夫斯基距离”（Minkowski distance）:

$$\mathrm{dist}_{\mathrm{mk}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \left(\sum_{u=1}^{n} |x_{iu} - x_{ju}|^p \right)^{\frac{1}{p}} .$$

$$\mathrm{MinkovDM}_p(\boldsymbol{x}_i, \boldsymbol{x}_j) = \left(\sum_{u=1}^{n_c} |x_{iu} - x_{ju}|^p + \sum_{u=n_c+1}^{n} \mathrm{VDM}_p(x_{iu}, x_{ju}) \right)^{\frac{1}{p}} . \quad (9$$

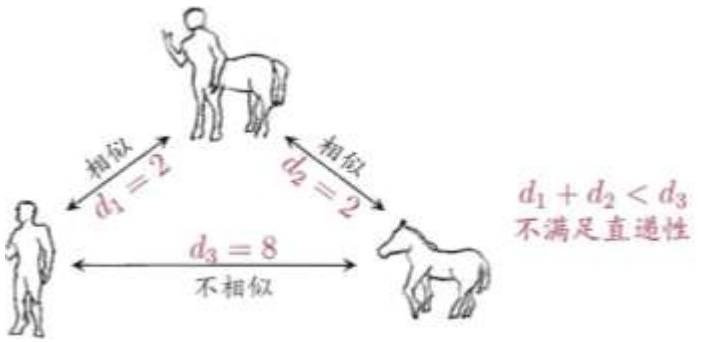
- 当p=1时，即曼哈顿距离（Manhattan distance）:

$$\mathrm{dist}_{\mathrm{man}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = ||\boldsymbol{x}_i - \boldsymbol{x}_j||_1 = \sum_{u=1}^{n} |x_{iu} - x_{ju}| .$$

- 当p=2时，即欧氏距离（Euclidean distance）:

$$\mathrm{dist}_{\mathrm{ed}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = ||\boldsymbol{x}_i - \boldsymbol{x}_j||_2 = \sqrt{\sum_{u=1}^{n} |x_{iu} - x_{ju}|^2} .$$

非度量距离:



ISS
smFISH
Hamming distance

9.4 原型聚类

原型聚类也称基于原型的聚类，此类算法假设聚类结构能通过一组原型刻画，在现实聚类中极为常用。

- 原型是指样本空间中具有代表性的点

算法过程：

- 一般流程为先对原型进行初始化，然后对原型进行迭代更新求解。
- 采用不同的原型表示、不同的求解方式，将产生不同的算法。

- K-Means便是基于簇中心来实现聚类
- 混合高斯聚类则是基于簇分布来实现聚类

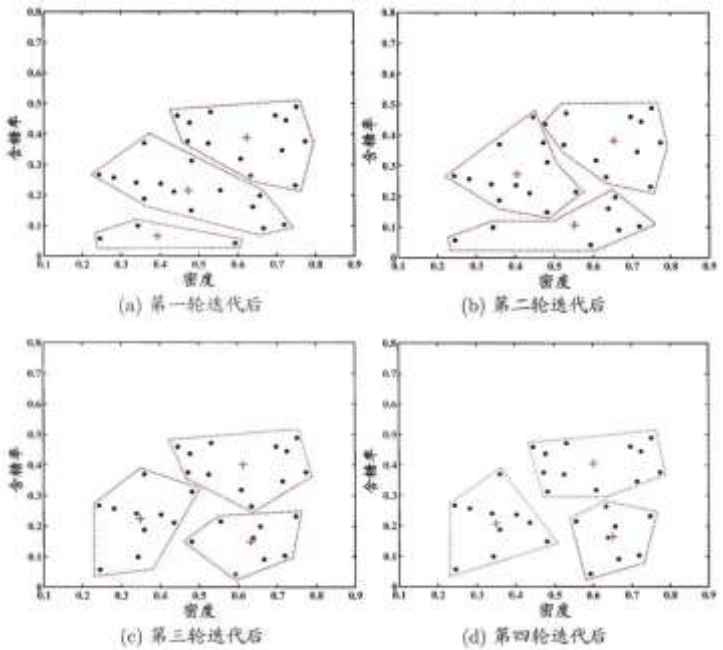
9.4.1 k均值聚类

首先随机指定类中心，根据样本与类中心的远近划分类簇，接着重新计算类中心，迭代直至收敛。

给定样本集 $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$，“$k$ 均值” (k-means)算法针对聚类所得簇划分 $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ 最小化平方误差

$$E = \sum_{i=1}^{k} \sum_{\boldsymbol{x} \in C_i} \|\boldsymbol{x} - \boldsymbol{\mu}_i\|_2^2, \tag{9.24}$$

其中 $\boldsymbol{\mu}_i = \frac{1}{|C_i|} \sum_{\boldsymbol{x} \in C_i} \boldsymbol{x}$ 是簇 C_i 的均值向量. 直观来看, 式(9.24) 在一定程度上刻画了簇内样本围绕簇均值向量的紧密程度, E 值越小则簇内样本相似度越高.



(a) 第一轮迭代后　(b) 第二轮迭代后　(c) 第三轮迭代后　(d) 第四轮迭代后

输入: 样本集 $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$;
聚类簇数 k.

过程:

```
从 D 中随机选择 k 个样本作为初始均值向量 {μ_1, μ_2, ..., μ_k}
repeat
    令 C_i = ∅ (1 ⩽ i ⩽ k)
    for j = 1, 2, ..., m do
        计算样本 x_j 与各均值向量 μ_i (1 ⩽ i ⩽ k) 的距离: d_ji = ||x_j − μ_i||_2;
        根据距离最近的均值向量确定 x_j 的簇标记: λ_j = arg min_{i∈{1,2,...,k}} d_ji;
        将样本 x_j 划入相应的簇: C_{λ_j} = C_{λ_j} ∪ {x_j};
    end for
    for i = 1, 2, ..., k do
        计算新均值向量: μ'_i = (1/|C_i|) Σ_{x∈C_i} x;
        if μ'_i ≠ μ_i then
            将当前均值向量 μ_i 更新为 μ'_i
        else
            保持当前均值向量不变
        end if
    end for
until 当前均值向量均未更新
```

输出: 簇划分 $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$

9.4.2 学习向量量化

LVQ使用样本真实类标记辅助聚类，首先LVQ根据样本的类标记，从各类中分别随机选出一个样本作为该类簇的原型，从而组成了一个**原型特征向量组**，接着从样本集中随机挑选一个样本，计算其与原型向量组中每个向量的距离，并选取距离最小的原型向量所在的类簇作为它的划分结果，再与真实类标比较。

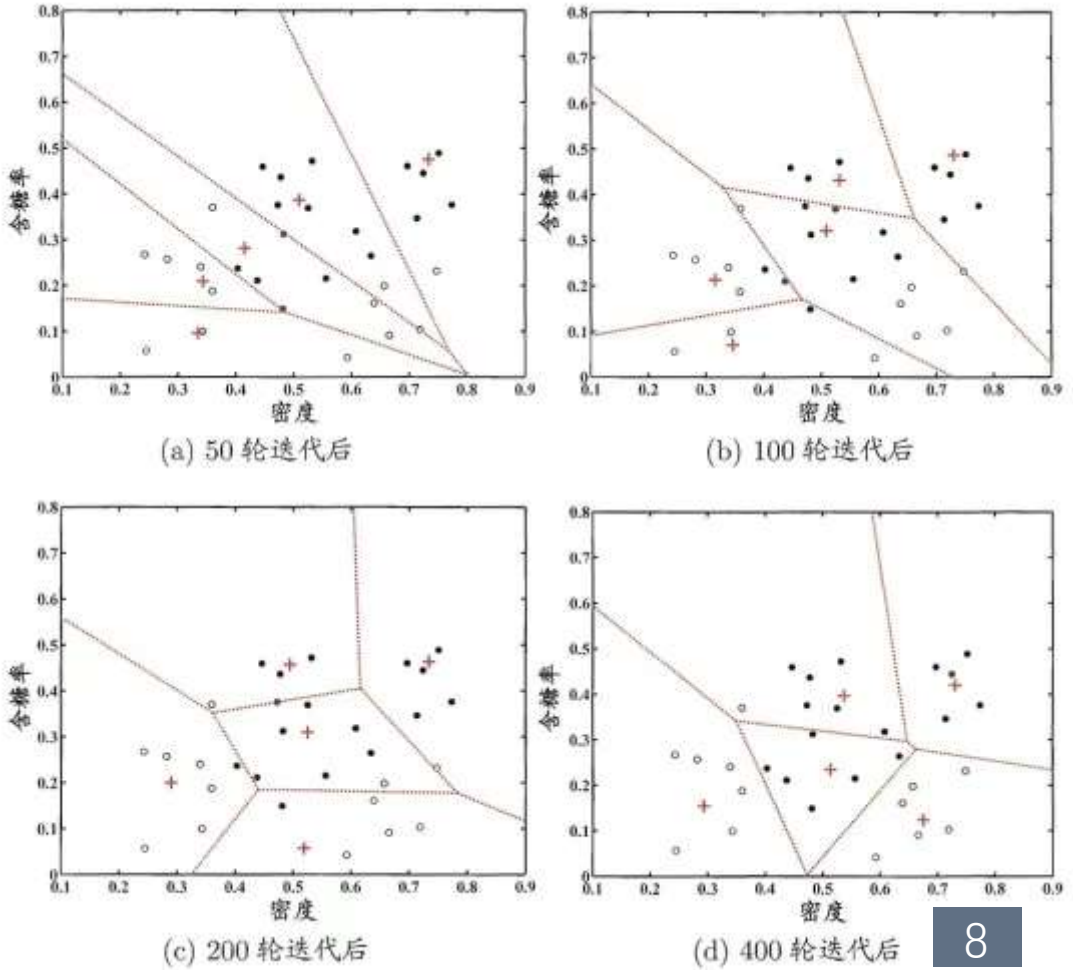
输入： 样本集 $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$;
原型向量个数 q，各原型向量预设的类别标记 $\{t_1, t_2, \ldots, t_q\}$;
学习率 $\eta \in (0, 1)$.

过程：

初始化一组原型向量 $\{\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_q\}$
repeat
  从样本集 D 随机选取样本 $(\boldsymbol{x}_j, y_j)$;
  计算样本 $\boldsymbol{x}_j$ 与 $\boldsymbol{p}_i$ $(1 \leqslant i \leqslant q)$ 的距离: $d_{ji} = \|\boldsymbol{x}_j - \boldsymbol{p}_i\|_2$;
  找出与 $\boldsymbol{x}_j$ 距离最近的原型向量 $\boldsymbol{p}_{i^*}$, $i^* = \arg\min_{i \in \{1,2,\ldots,q\}} d_{ji}$;
  if $y_j = t_{i^*}$ **then**
    $\boldsymbol{p}' = \boldsymbol{p}_{i^*} + \eta \cdot (\boldsymbol{x}_j - \boldsymbol{p}_{i^*})$ 让更新的 p_i^* 接近 x_j
  else
    $\boldsymbol{p}' = \boldsymbol{p}_{i^*} - \eta \cdot (\boldsymbol{x}_j - \boldsymbol{p}_{i^*})$
  end if
  将原型向量 $\boldsymbol{p}_{i^*}$ 更新为 $\boldsymbol{p}'$
until 满足停止条件 样本与 p_i 距离不大于其与 p' 的距离

$$\|\boldsymbol{p}' - \boldsymbol{x}_j\|_2 = \|\boldsymbol{p}_{i^*} + \eta \cdot (\boldsymbol{x}_j - \boldsymbol{p}_{i^*}) - \boldsymbol{x}_j\|_2 = (1-\eta) \cdot \|\boldsymbol{p}_{i^*} - \boldsymbol{x}_j\|_2 .$$

输出： 原型向量 $\{\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_q\}$ 中心点，而不是每个簇的样本



(a) 50 轮迭代后　(b) 100 轮迭代后

(c) 200 轮迭代后　(d) 400 轮迭代后

9.4.3 高斯混合聚类

对n维样本空间中的随机向量x，若x服从高斯分布，其概率密度函数为：

$$p(x) = \frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)}$$

其中μ是n维均值向量,Σ是n*n的协方差矩阵。也可将概率密度函数记作$p(x|\mu,\Sigma)$

$$p_{\mathcal{M}}(\boldsymbol{x}) = \sum_{i=1}^{k} \alpha_i \cdot p(\boldsymbol{x} \mid \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \ ,$$

$\alpha_i > 0$，混合系数 $\sum_{i=1}^{k} \alpha_i = 1.$

α_i 为选择第 i 个混合成分的概率;

对于训练集 $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$, 现在要把 m 个样本划分为 k 个簇, 即认为训练集 D 的样本是根据 k 个不同的多元高斯分布加权而得的高斯混合模型生成的。

9.4.3 高斯混合聚类

$$p_{\mathcal{M}}(z_j = i \mid \boldsymbol{x}_j) = \frac{P(z_j = i) \cdot p_{\mathcal{M}}(\boldsymbol{x}_j \mid z_j = i)}{p_{\mathcal{M}}(\boldsymbol{x}_j)}$$

$$= \frac{\alpha_i \cdot p(\boldsymbol{x}_j \mid \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)}{\sum_{l=1}^{k} \alpha_l \cdot p(\boldsymbol{x}_j \mid \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)} .$$

全概率公式

$$\lambda_j = \underset{i \in \{1,2,\ldots,k\}}{\arg\max} \ \gamma_{ji} .$$

极大似然估计

$$LL(D) = \ln\left(\prod_{j=1}^{m} p_M(x_j)\right)$$

$$= \sum_{j=1}^{m} \ln\left(\sum_{i=1}^{k} \alpha_i \cdot p\left(x_j | \mu_i, \Sigma_i\right)\right)$$

令 $\frac{\partial LL(D)}{\partial \mu_i} = 0$ 得 $\mu_i = \frac{\sum_{j=1}^{m} \gamma_{ji} x_j}{\sum_{j=1}^{m} \gamma_{ji}}$

令 $\frac{\partial LL(D)}{\partial \Sigma_i} = 0$ 得 $\Sigma_i = \frac{\sum_{j=1}^{m} \gamma_{ji} (x_j - \mu_i)(x_j - \mu_i)^T}{\sum_{j=1}^{m} \gamma_{ji}}$

拉格朗日乘子 $\alpha_i = \frac{1}{m} \sum_{j=1}^{m} \gamma_{ji}$

《南瓜书》P114

9.4.3 高斯混合聚类

输入: 样本集 $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$; m个
高斯混合成分个数 k.

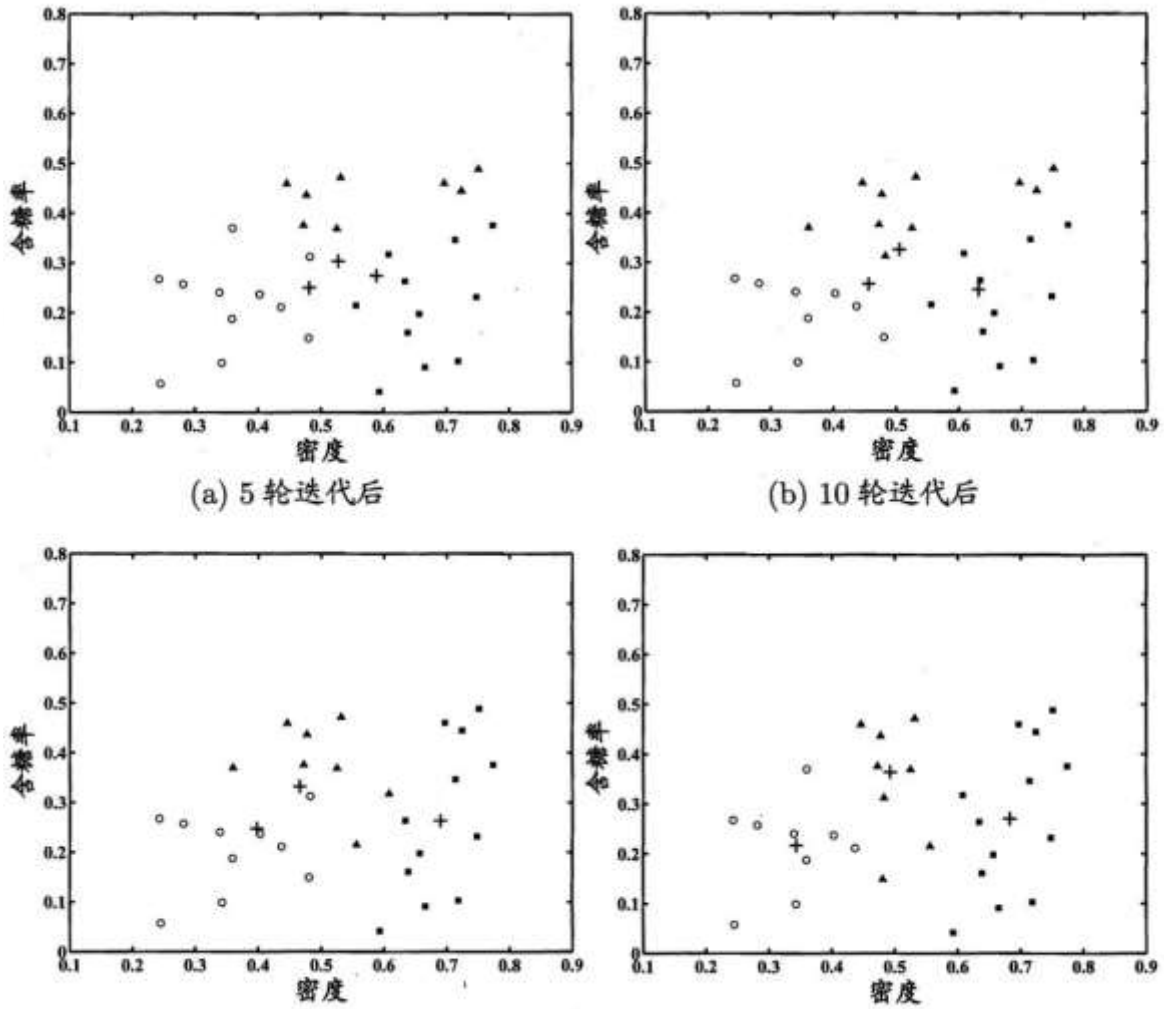
EM算法:
(期望最大算法)

过程:

初始化高斯混合分布的模型参数 $\{(\alpha_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \mid 1 \leqslant i \leqslant k\}$
repeat
  for $j = 1, 2, \ldots, m$ **do**
    根据式(9.30)计算 $\boldsymbol{x}_j$ 由各混合成分生成的后验概率, 即
    $\gamma_{ji} = p_{\mathcal{M}}(z_j = i \mid \boldsymbol{x}_j)\ (1 \leqslant i \leqslant k)$ E步 →估计参数 (期望步)
  end for
  for $i = 1, 2, \ldots, k$ **do**
    计算新均值向量: $\boldsymbol{\mu}_i' = \frac{\sum_{j=1}^{m} \gamma_{ji} \boldsymbol{x}_j}{\sum_{j=1}^{m} \gamma_{ji}}$;
    计算新协方差矩阵: $\boldsymbol{\Sigma}_i' = \frac{\sum_{j=1}^{m} \gamma_{ji} (\boldsymbol{x}_j - \boldsymbol{\mu}_i')(\boldsymbol{x}_j - \boldsymbol{\mu}_i')^{\mathrm{T}}}{\sum_{j=1}^{m} \gamma_{ji}}$;
    计算新混合系数: $\alpha_i' = \frac{\sum_{j=1}^{m} \gamma_{ji}}{m}$;
  end for
  将模型参数 $\{(\alpha_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \mid 1 \leqslant i \leqslant k\}$ 更新为 $\{(\alpha_i', \boldsymbol{\mu}_i', \boldsymbol{\Sigma}_i') \mid 1 \leqslant i \leqslant k\}$
until 满足停止条件
$C_i = \varnothing\ (1 \leqslant i \leqslant k)$ 每个簇集合为空
for $j = 1, 2, \ldots, m$ **do**
  根据式(9.31)确定 $\boldsymbol{x}_j$ 的簇标记 λ_j
  将 $\boldsymbol{x}_j$ 划入相应的簇: $C_{\lambda_j} = C_{\lambda_j} \bigcup \{\boldsymbol{x}_j\}$
end for

(极大步)
M步 →调整参数

输出: 簇划分 $C = \{C_1, C_2, \ldots, C_k\}$



(a) 5 轮迭代后　(b) 10 轮迭代后

(c) 20 轮迭代后　(d) 50 轮迭代后

9.5 密度聚类

密度聚类也称为“基于密度的聚类” (density-based clustering)。此类算法假设聚类结构能通过样本分布的紧密程度来确定。通常情况下，密度聚类算法从样本密度的角度来考察样本之间的可连接性，并基于可连接样本不断扩展聚类簇来获得最终的聚类结果。

DBSCAN算法：

ϵ-邻域：对 $\boldsymbol{x}_j \in D$，其 ϵ-邻域包含样本集 D 中与 $\boldsymbol{x}_j$ 的距离不大于 ϵ 的样本，即 $N_\epsilon(\boldsymbol{x}_j) = \{\boldsymbol{x}_i \in D \mid \mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j) \leqslant \epsilon\}$；

圈内的是△的邻域

核心对象(core object)：若 $\boldsymbol{x}_j$ 的 ϵ-邻域至少包含 $MinPts$ 个样本，即 $|N_\epsilon(\boldsymbol{x}_j)| \geqslant MinPts$，则 $\boldsymbol{x}_j$ 是一个核心对象；6>3，△是一个核心对象

密度直达(directly density-reachable)：若 $\boldsymbol{x}_j$ 位于 $\boldsymbol{x}_i$ 的 ϵ-邻域中，且 $\boldsymbol{x}_i$ 是核心对象，则称 $\boldsymbol{x}_j$ 由 $\boldsymbol{x}_i$ 密度直达；□由△密度直达

密度可达(density-reachable)：对 $\boldsymbol{x}_i$ 与 $\boldsymbol{x}_j$，若存在样本序列 $\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_n$，其中 $\boldsymbol{p}_1 = \boldsymbol{x}_i$，$\boldsymbol{p}_n = \boldsymbol{x}_j$ 且 $\boldsymbol{p}_{i+1}$ 由 $\boldsymbol{p}_i$ 密度直达，则称 $\boldsymbol{x}_j$ 由 $\boldsymbol{x}_i$ 密度可达；

通过多次直达即可达

密度相连(density-connected)：对 $\boldsymbol{x}_i$ 与 $\boldsymbol{x}_j$，若存在 $\boldsymbol{x}_k$ 使得 $\boldsymbol{x}_i$ 与 $\boldsymbol{x}_j$ 均由 $\boldsymbol{x}_k$ 密度可达，则称 $\boldsymbol{x}_i$ 与 $\boldsymbol{x}_j$ 密度相连.

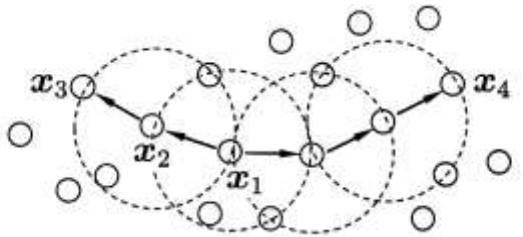


图 9.8 DBSCAN 定义的基本概念($MinPts = 3$)：虚线显示出 ϵ-邻域，$\boldsymbol{x}_1$ 是核心对象，$\boldsymbol{x}_2$ 由 $\boldsymbol{x}_1$ 密度直达，$\boldsymbol{x}_3$ 由 $\boldsymbol{x}_1$ 密度可达，$\boldsymbol{x}_3$ 与 $\boldsymbol{x}_4$ 密度相连.

简单来理解DBSCAN是：找出一个核心对象所有密度可达的样本集合形成簇

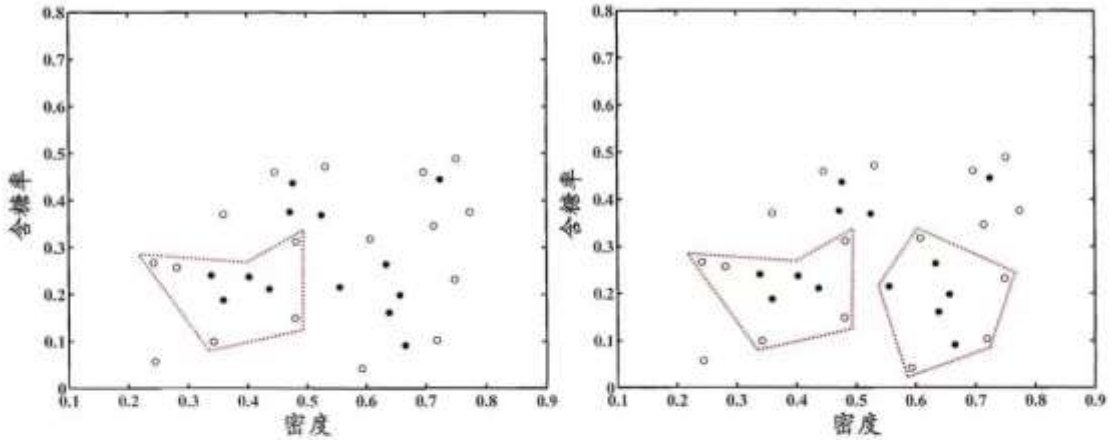
9.5 密度聚类

输入: 样本集 $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$;
邻域参数 $(\epsilon, MinPts)$.

过程:

初始化核心对象集合: $\Omega = \varnothing$
for $j = 1, 2, \ldots, m$ **do**
  确定样本 $\boldsymbol{x}_j$ 的 ϵ-邻域 $N_\epsilon(\boldsymbol{x}_j)$;
  if $|N_\epsilon(\boldsymbol{x}_j)| \geqslant MinPts$ **then**
    将样本 $\boldsymbol{x}_j$ 加入核心对象集合: $\Omega = \Omega \bigcup \{\boldsymbol{x}_j\}$
  end if
end for
初始化聚类簇数: $k = 0$
初始化未访问样本集合: $\Gamma = D$
while $\Omega \neq \varnothing$ **do**
  记录当前未访问样本集合: $\Gamma_{\text{old}} = \Gamma$;
  随机选取一个核心对象 $\boldsymbol{o} \in \Omega$, 初始化队列 $Q = < \boldsymbol{o} >$;
  $\Gamma = \Gamma \setminus \{\boldsymbol{o}\}$;
  while $Q \neq \varnothing$ **do**
    取出队列 Q 中的首个样本 $\boldsymbol{q}$;
    if $|N_\epsilon(\boldsymbol{q})| \geqslant MinPts$ **then**
      令 $\Delta = N_\epsilon(\boldsymbol{q}) \bigcap \Gamma$;
      将 Δ 中的样本加入队列 Q;
      $\Gamma = \Gamma \setminus \Delta$;
    end if
  end while
  $k = k + 1$, 生成聚类簇 $C_k = \Gamma_{\text{old}} \setminus \Gamma$;
  $\Omega = \Omega \setminus C_k$
end while

输出: 簇划分 $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$



(a) 生成聚类簇 C_1　(b) 生成聚类簇 C_2

(c) 生成聚类簇 C_3　(d) 生成聚类簇 C_4

9.6 层次聚类

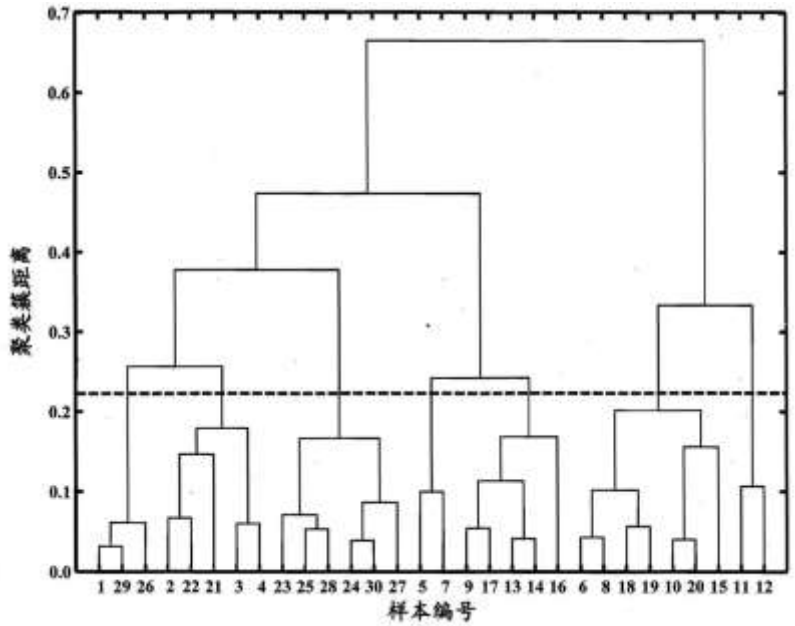
层次聚类在不同层次对数据集进行划分，从而形成树形的聚类结构。数据集划分既可采用“自底向上”的聚合策略，也可采用“自顶向下”的分拆策略。

AGNES算法（自底向上的层次聚类算法）：首先，将样本中的每一个样本看做一个初始聚类簇，然后在算法运行的每一步中找出距离最近的两个聚类簇进行合并，该过程不断重复，直到达到预设的聚类簇的个数。

最小距离：$d_{\min}(C_i, C_j) = \min_{\boldsymbol{x} \in C_i, \boldsymbol{z} \in C_j} \operatorname{dist}(\boldsymbol{x}, \boldsymbol{z})$，最近样本

最大距离：$d_{\max}(C_i, C_j) = \max_{\boldsymbol{x} \in C_i, \boldsymbol{z} \in C_j} \operatorname{dist}(\boldsymbol{x}, \boldsymbol{z})$，最远样本

平均距离：$d_{\text{avg}}(C_i, C_j) = \dfrac{1}{|C_i||C_j|} \sum_{\boldsymbol{x} \in C_i} \sum_{\boldsymbol{z} \in C_j} \operatorname{dist}(\boldsymbol{x}, \boldsymbol{z})$ 的每样本



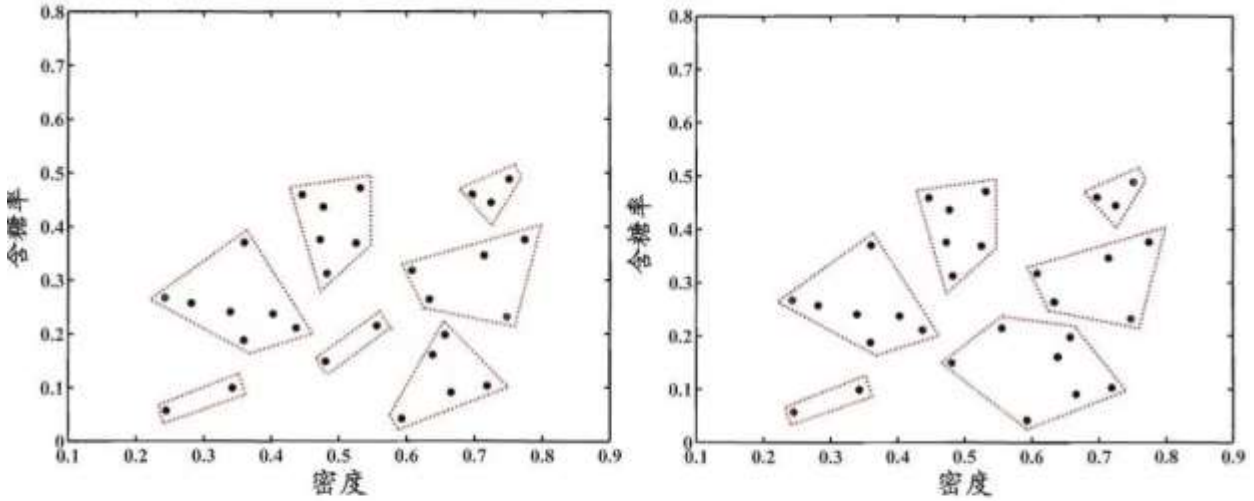
9.6 层次聚类

输入: 样本集 $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$;
聚类簇距离度量函数 d;
聚类簇数 k.

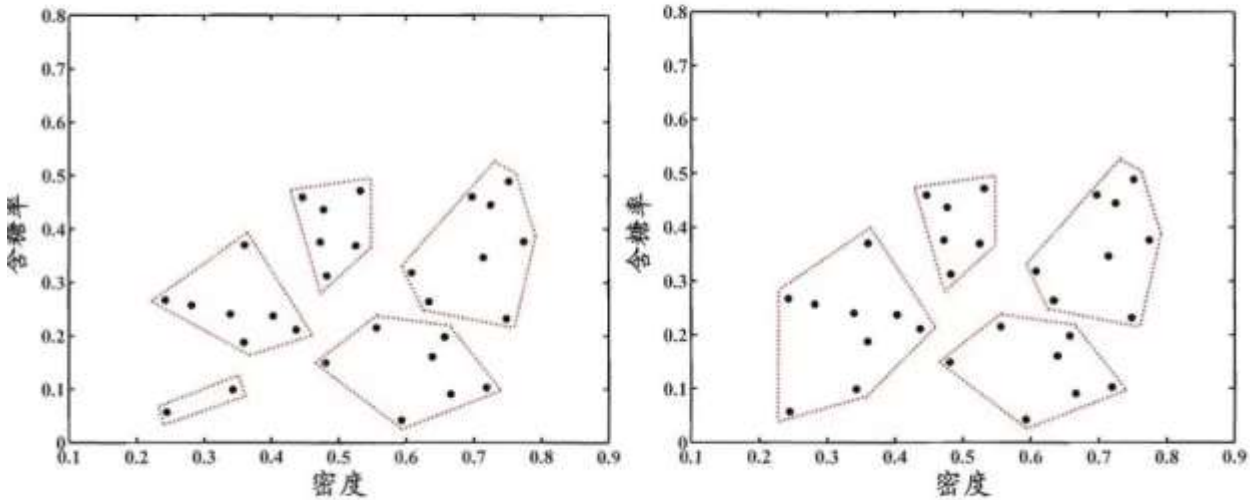
过程:

```
for j = 1, 2, ..., m do
    C_j = {x_j}
end for
for i = 1, 2, ..., m do
    for j = 1, 2, ..., m do
        M(i, j) = d(C_i, C_j);
        M(j, i) = M(i, j)
    end for
end for
设置当前聚类簇个数: q = m
while q > k do
    找出距离最近的两个聚类簇 C_i* 和 C_j*;
    合并 C_i* 和 C_j*: C_i* = C_i* ∪ C_j*;
    for j = j* + 1, j* + 2, ..., q do
        将聚类簇 C_j 重编号为 C_{j-1}
    end for
    删除距离矩阵 M 的第 j* 行与第 j* 列;
    for j = 1, 2, ..., q - 1 do
        M(i*, j) = d(C_i*, C_j);
        M(j, i*) = M(i*, j)
    end for
    q = q - 1
end while
```

输出: 簇划分 $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$



(a) 聚类簇数 $k = 7$　　(b) 聚类簇数 $k = 6$



(c) 聚类簇数 $k = 5$　　(d) 聚类簇数 $k = 4$

感谢倾听

See you next.