聚类模型

1 基础知识

1.1 聚类模型

原型聚类：“原型”指样本空间中具有代表性的点，原型聚类亦称“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。

1.2 评价指标

评价指标有内部指标（数据集没有标签）和外部指标（数据集有标签）。

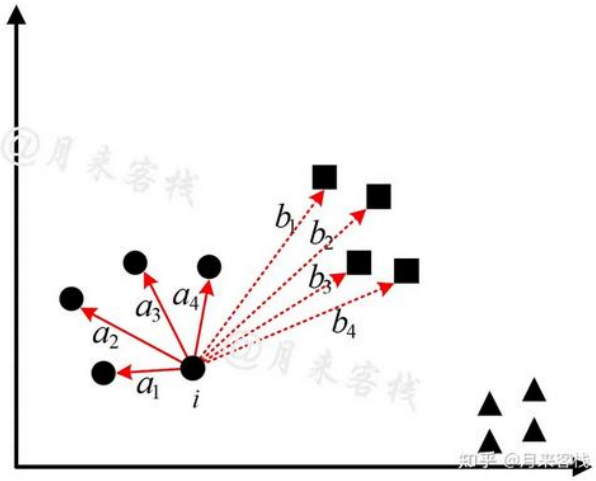
1.2.1 内部指标

由于聚类的目标是使同一簇中的样本相似，而不同簇中的对象不同，因此大多数内部验证都基于以下两个标准：

内聚度：同一簇中对象的紧密程度。内聚度可以用不同的方法来衡量，比如使用每个簇内点的方差，或者计算它们之间的平均成对距离。

分离度：一个簇与其他簇的区别或分离程度。分离度的例子包括簇中心之间的成对距离或不同簇中对象之间的成对最小距离。

（1）轮廓系数 Silhouette Coefficient



样本 x_i 的轮廓系数公式为：

$$s(x_i)=\frac{b(x_i)-a(x_i)}{\max\{a(x_i),b(x_i)\}}$$

其中，$a(x_i)$ 表示样本 x_i 到簇中每个样本欧氏距离的均值，$b(x_i)$ 表示样本 x_i 到最近簇中每个样本欧氏距离的均值的内聚度，计算公式为

$$a(x_i)=\frac{1}{4}\sum_{i=1}^{n}a_i, b(x_i)=\frac{1}{4}\sum_{i=1}^{n}b_i$$

这里需要注意的一点是，对于同一个簇中的每个样本点来说，距离自己最近的簇可能并不是同一个；同时，在寻找距离当前样本点最近的簇结构时，计算的是当前样本点到各个簇中心的最短距离，而不是计算当前样本点所在簇的簇中心到其它每个簇中心的最短距离。

对整个数据集中样本的轮廓系数求平均，得到整个聚类的轮廓系数：

$$s=\sum_{i=1}^{m}s(x_i)$$

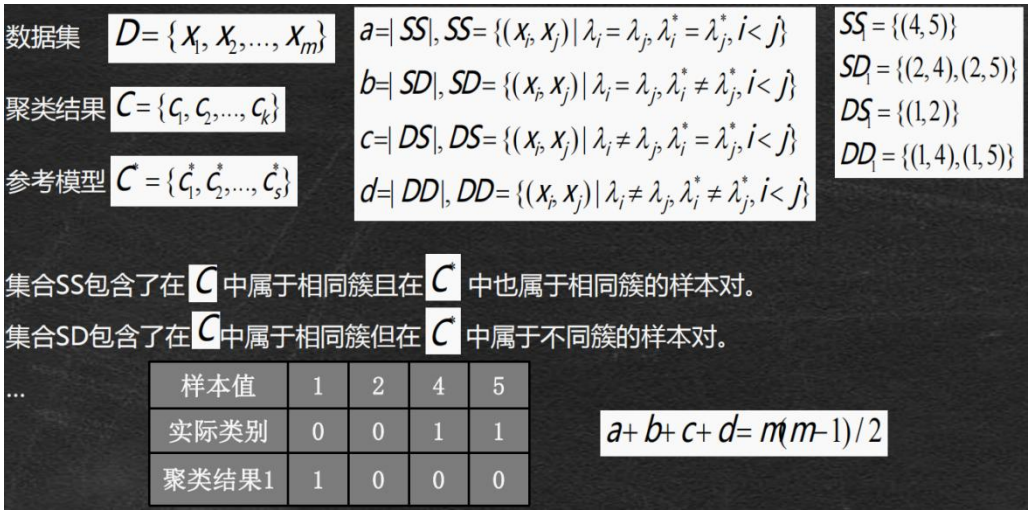
轮廓系数的取值范围在[-1,1]，对样本轮廓系数来说，值越高表示该点与自己的聚类匹配得越好，与邻近的聚类匹配得越差。对平均轮廓系数来说，越接近 1 意味着聚类质量越好，聚类紧凑且分离良好。

1.2.2 外部指标

外部指标有 Jaccard 系数，FM 指数，Rand 指数，值在[0,1]之间，越大越好。

给定数据集 $D=\{x_1,...,x_m\}$，聚类的簇划分为 $C=\{c_1,...,c_k\}$，真实的簇划分为 $C^*=\{c_1^*,...,c_s^*\}$，令 λ,λ^* 分别表示 C,C^* 对应的簇标记向量，将样本两两配对考虑，定义

$$a=|SS|, SS=\{(x_i,x_j)\mid \lambda_i=\lambda_j, \lambda_i^*=\lambda_j^*, i<j\}$$
$$b=|SD|, SD=\{(x_i,x_j)\mid \lambda_i=\lambda_j, \lambda_i^*\neq\lambda_j^*, i<j\}$$
$$c=|DS|, DS=\{(x_i,x_j)\mid \lambda_i\neq\lambda_j, \lambda_i^*=\lambda_j^*, i<j\}$$
$$d=|DD|, DD=\{(x_i,x_j)\mid \lambda_i\neq\lambda_j, \lambda_i^*\neq\lambda_j^*, i<j\}$$



基于上式可导出下面这些常用的聚类性能度量外部指标：

$$JC=\frac{a}{a+b+c},FMI=\sqrt{\frac{a}{a+b}\cdot\frac{a}{a+c}},RI=\frac{2(a+d)}{m(m-1)}$$

2 原型聚类 KMeans

2.1 距离度量相似性

聚类的目的是尽可能让相似的样本归为一类，KMeans 采用距离来度量样本之间的相似性，将距离近的归为一类。

对连续型特征，通过以下距离来度量样本间的相似性：

距离	定义式	说明
绝对值距离	$d_{ij}(1)=\sum_{k=1}^{p}\lvert x_{ik}-x_{jk}\rvert$	一维空间下进行的距离计算
欧式距离	$d_{ij}(2)=\sqrt{\sum_{k=1}^{p}(x_{ik}-x_{jk})^2}$	二维空间下进行的距离计算
闵可夫斯基距离	$d_{ij}(q)=[\sum_{k=1}^{p}(x_{ik}-x_{jk})^q]^{1/q},q>0$	q 维空间下进行的距离计算
切比雪夫距离	$d_{ij}(\infty)=\max_{1\le k\le p}\lvert x_{ik}-x_{jk}\rvert$	q 取无穷大时的闵可夫斯基距离
Lance 距离	$d_{ij}(L)=\sum_{k=1}^{p}\frac{\lvert x_{ik}-x_{jk}\rvert}{x_{ik}+x_{jk}}$	削弱极端值的影响能力

对离散型特征，如出行方式{飞机，火车，轮船}，采用 VDM（Value Difference Metric）距离来度量样本之间的相似性：

令 $m_{u,a}$ 表示在特征 u 上取值为 a 的样本数，$m_{u,a,i}$ 表示在第 i 个样本簇中在特征 u 上取值为 a 的样本数，k 为样本簇数，则特征 u 上两个离散值 a 和 b 之间的 VDM 距离为：

$$VDM_q(a,b)=\sum_{i=1}^{k}\left|\frac{m_{u,a,i}}{m_{u,a}}-\frac{m_{u,b,i}}{m_{u,b}}\right|^q$$

对混合性特征，采用上述距离结合的方式来度量样本之间的相似性：

如将闵可夫斯基距离和 VDM 结合即可处理混合属性，假定有 n_c 个有序属性，$n-n_c$ 个无序属性，不失一般性，令有序属性排列在无序属性之前，则

$$MinkovDM_p(x_i,x_j)=(\sum_{u=1}^{n_c}\lvert x_{iu}-x_{ju}\rvert^q+\sum_{u=n_c+1}^{n}VDM_p(x_{iu},x_{ju}))^{1/q}$$

2.2 算法伪代码

给定样本集 $D=\{x_1,x_2,...,x_m\}$，假设 K 均值针对聚类所得簇划分为 $C=\{C_1,...,C_k\}$，最小化平方误差：

$$E=\sum_{i=1}^{k}\sum_{x\in C_i}\| x-\mu_i \|_2^2 ,$$

其中 $\mu_i=\frac{1}{|C_i|}\sum_{x\in C_i} x$ 是簇 C_i 的均值向量，直观来看，最小误差刻画了簇内样本围绕簇均值向量的紧密程度，越小则簇内相似性越高。但是最小化该误差并不容易，找到它的最优解需要考虑样本集 D 中所有可能的簇划分，这是一个 NP 难问题，因此 K 均值采用了贪心策略，通过迭代优化来近似求解。

输入: 样本集 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_m\}$;
聚类簇数 k.
过程:
1: 从 D 中随机选择 k 个样本作为初始均值向量 $\{\boldsymbol{\mu}_1,\boldsymbol{\mu}_2,\ldots,\boldsymbol{\mu}_k\}$
2: **repeat**
3: 令 $C_i=\varnothing\ (1\leqslant i\leqslant k)$
4: **for** $j=1,2,\ldots,m$ **do**
5: 计算样本 $\boldsymbol{x}_j$ 与各均值向量 $\boldsymbol{\mu}_i\ (1\leqslant i\leqslant k)$ 的距离: $d_{ji}=\|\boldsymbol{x}_j-\boldsymbol{\mu}_i\|_2$;
6: 根据距离最近的均值向量确定 $\boldsymbol{x}_j$ 的簇标记: $\lambda_j=\arg\min_{i\in\{1,2,\ldots,k\}} d_{ji}$;
7: 将样本 $\boldsymbol{x}_j$ 划入相应的簇: $C_{\lambda_j}=C_{\lambda_j}\bigcup\{\boldsymbol{x}_j\}$;
8: **end for**
9: **for** $i=1,2,\ldots,k$ **do**
10: 计算新均值向量: $\boldsymbol{\mu}_i'=\frac{1}{|C_i|}\sum_{\boldsymbol{x}\in C_i}\boldsymbol{x}$;
11: **if** $\boldsymbol{\mu}_i'\neq\boldsymbol{\mu}_i$ **then**
12: 将当前均值向量 $\boldsymbol{\mu}_i$ 更新为 $\boldsymbol{\mu}_i'$
13: **else**
14: 保持当前均值向量不变
15: **end if**
16: **end for**
17: **until** 当前均值向量均未更新
输出: 簇划分 $\mathcal{C}=\{C_1,C_2,\ldots,C_k\}$

图 9.2 k 均值算法

其中，第 1 行对均值向量进行初始化，在第 4-8 行与第 9-16 行依次对当前簇划分和均值向量迭代更新，若迭代更新后聚类结果保持不变，则在第 18 行将当前簇划分结果返回。

2.3 优缺点分析

- 优点是对当距离可以反映出样本之间的区别时，对球状簇聚类效果好。
- 缺点是不能处理非球状簇，不同尺寸和不同密度的簇；对离群点，噪声点敏感。
- 为了避免对离群点敏感，可以牺牲聚类时间来实现，这就是 K-Medoids：在更新簇中心时，在各类别内选取到其余样本距离之和最小的样本作为新的类中心。

2.4 西瓜数据集举例

表 9.1 西瓜数据集 4.0

编号	密度	含糖率	编号	密度	含糖率	编号	密度	含糖率
1	0.697	0.460	11	0.245	0.057	21	0.748	0.232
2	0.774	0.376	12	0.343	0.099	22	0.714	0.346
3	0.634	0.264	13	0.639	0.161	23	0.483	0.312
4	0.608	0.318	14	0.657	0.198	24	0.478	0.437
5	0.556	0.215	15	0.360	0.370	25	0.525	0.369
6	0.403	0.237	16	0.593	0.042	26	0.751	0.489
7	0.481	0.149	17	0.719	0.103	27	0.532	0.472
8	0.437	0.211	18	0.359	0.188	28	0.473	0.376
9	0.666	0.091	19	0.339	0.241	29	0.725	0.445
10	0.243	0.267	20	0.282	0.257	30	0.446	0.459

假定聚类簇数 $k=3$, 算法开始时随机选取三个样本 $\boldsymbol{x}_6$, $\boldsymbol{x}_{12}$, $\boldsymbol{x}_{27}$ 作为初始均值向量, 即

$$\boldsymbol{\mu}_1=(0.403;0.237),\ \boldsymbol{\mu}_2=(0.343;0.099),\ \boldsymbol{\mu}_3=(0.532;0.472)\ .$$

考察样本 $\boldsymbol{x}_1=(0.697;0.460)$, 它与当前均值向量 $\boldsymbol{\mu}_1,\boldsymbol{\mu}_2,\boldsymbol{\mu}_3$ 的距离分别为 $0.369, 0.506,\ 0.166$, 因此 $\boldsymbol{x}_1$ 将被划入簇 C_3中. 类似的, 对数据集中的所有样本考察一遍后, 可得当前簇划分为

$$C_1=\{\boldsymbol{x}_5,\boldsymbol{x}_6,\boldsymbol{x}_7,\boldsymbol{x}_8,\boldsymbol{x}_9,\boldsymbol{x}_{10},\boldsymbol{x}_{13},\boldsymbol{x}_{14},\boldsymbol{x}_{15},\boldsymbol{x}_{17},\boldsymbol{x}_{18},\boldsymbol{x}_{19},\boldsymbol{x}_{20},\boldsymbol{x}_{23}\};$$

$$C_2=\{\boldsymbol{x}_{11},\boldsymbol{x}_{12},\boldsymbol{x}_{16}\};$$

$$C_3=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\boldsymbol{x}_3,\boldsymbol{x}_4,\boldsymbol{x}_{21},\boldsymbol{x}_{22},\boldsymbol{x}_{24},\boldsymbol{x}_{25},\boldsymbol{x}_{26},\boldsymbol{x}_{27},\boldsymbol{x}_{28},\boldsymbol{x}_{29},\boldsymbol{x}_{30}\}.$$

于是, 可从 C_1、C_2、C_3 分别求出新的均值向量

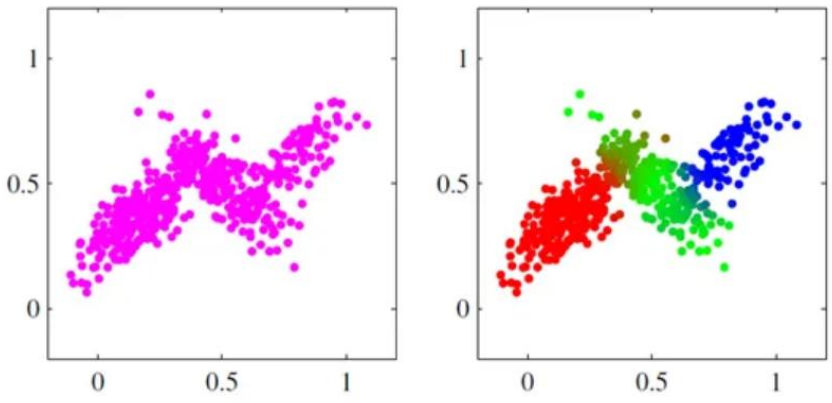
$$\boldsymbol{\mu}_1'=(0.473;0.214),\ \boldsymbol{\mu}_2'=(0.394;0.066),\ \boldsymbol{\mu}_3'=(0.623;0.388)\ .$$

更新当前均值向量后, 不断重复上述过程, 如图 9.3 所示, 第五轮迭代产生的结果与第四轮迭代相同, 于是算法停止, 得到最终的簇划分.

3 原型聚类 GMM

3.1 原理介绍

高斯混合模型 GMM 是一种概率模型，它假定实例是由多个参数未知的高斯分布线性组合而成，通过 EM 算法求解，实例属于哪一个高斯分布并不知道，因此 E 步先判断这些实例属于哪一个高斯分布（类别），M 步利用极大似然估计更新每一类的未知参数。

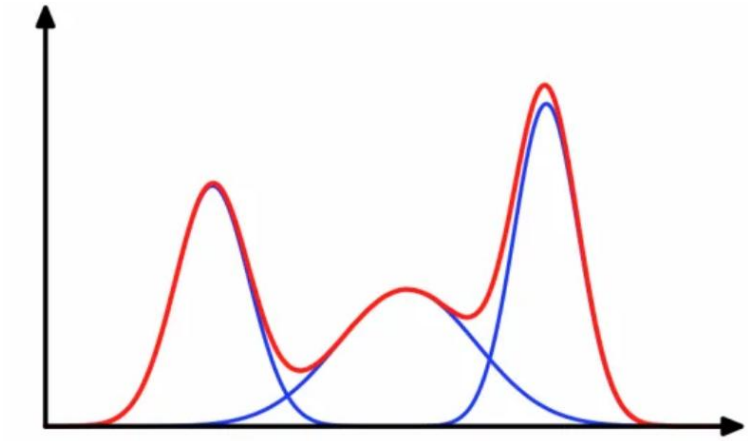


高斯混合模型具有如下形式的概率分布：

$$P(X=x\mid\theta)=\sum_{k=1}^{K}\alpha_k N(X=x\mid\mu_k,\Sigma_k),\sum_{k=1}^{K}\alpha_k=1$$

其中，μ_k,Σ_k 为均值向量和协方差，维度取决于特征的个数。

分布具体表现为：



红色的曲线是三个一元高斯分布的线性组合(具有很好的可解释性)，高斯分布的线性组合不再有那么大的局限性，通过构造复杂的概率密度函数，调节各参数，理论上几乎所有连续概率密度都可以被高斯混合模型以任意精度近似。

3.2 算法伪代码

算法第 1 行对高斯混合分布的模型参数进行初始化；然后，在第 2 -12 行基于 EM 算法对模型参数进行迭代更新，若 EM 算法的停止条件满足(例如已达到最大迭代轮数，或似然函数 LL (D)增长很少甚至不再增长)，则在第 14-17 行根据高斯混合分布确定簇划分，在第 18 行返回最终结果.

输入: 样本集 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_m\}$;
高斯混合成分个数 k.
过程:
1: 初始化高斯混合分布的模型参数 $\{(\alpha_i,\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)\mid 1\leqslant i\leqslant k\}$
2: **repeat**
EM 算法的 E步.
3: **for** $j=1,2,\ldots,m$ **do**
4: 根据式(9.30)计算 $\boldsymbol{x}_j$ 由各混合成分生成的后验概率, 即 $\gamma_{ji}=p_{\mathcal{M}}(z_j=i\mid\boldsymbol{x}_j)\ (1\leqslant i\leqslant k)$
5: **end for**
EM 算法的 M步.
6: **for** $i=1,2,\ldots,k$ **do**
7: 计算新均值向量: $\boldsymbol{\mu}_i'=\frac{\sum_{j=1}^{m}\gamma_{ji}\boldsymbol{x}_j}{\sum_{j=1}^{m}\gamma_{ji}}$;
8: 计算新协方差矩阵: $\boldsymbol{\Sigma}_i'=\frac{\sum_{j=1}^{m}\gamma_{ji}(\boldsymbol{x}_j-\boldsymbol{\mu}_i')(\boldsymbol{x}_j-\boldsymbol{\mu}_i')^{\mathrm{T}}}{\sum_{j=1}^{m}\gamma_{ji}}$;
9: 计算新混合系数: $\alpha_i'=\frac{\sum_{j=1}^{m}\gamma_{ji}}{m}$;
10: **end for**
11: 将模型参数 $\{(\alpha_i,\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)\mid 1\leqslant i\leqslant k\}$ 更新为 $\{(\alpha_i',\boldsymbol{\mu}_i',\boldsymbol{\Sigma}_i')\mid 1\leqslant i\leqslant k\}$
12: **until** 满足停止条件
例如达到最大迭代轮数.
13: $C_i=\varnothing\ (1\leqslant i\leqslant k)$
14: **for** $j=1,2,\ldots,m$ **do**
15: 根据式(9.31)确定 $\boldsymbol{x}_j$ 的簇标记 λ_j;
16: 将 $\boldsymbol{x}_j$ 划入相应的簇: $C_{\lambda_j}=C_{\lambda_j}\bigcup\{\boldsymbol{x}_j\}$
17: **end for**
输出: 簇划分 $\mathcal{C}=\{C_1,C_2,\ldots,C_k\}$

图 9.6 高斯混合聚类算法

对于聚类数的选择，可以采用贝叶斯信息准则（BIC）和赤池信息准则（AIC）

$$BIC = \log(m)p - 2\log(\hat{L}), AIC = 2p - 2\log(\hat{L})$$

其中，m 是实例个数，p 是模型参数数量，$\hat{L}$ 是似然函数最大值。

注意：BIC 选择的模型往往比 AIC 选择的模型更简单（参数更少），但往往不太拟合数据（对于较大的数据集尤其如此）。

3.3 优缺点分析

• 优点概率建模：高斯混合模型能够对数据进行概率建模，即将数据看作是由多个高斯分布组成的混合模型。这使得它能够刻画数据点属于不同聚类的概率分布，而不仅仅是将数据点分配到确定的聚类中。
• 优点适用于具有潜在概率分布的数据：当数据的生成过程可以被概率模型所描述时，高斯混合模型是一种有效的聚类算法。例如，当数据来自于不同的高斯分布或者近似服从高斯分布时，可以使用高斯混合模型进行聚类。
• 缺点对数据分布的假设要求较高：高斯混合模型假设数据点服从多个高斯分布，并且每个聚类的分布都是高斯分布。如果数据的实际分布与该假设不符，则可能导致聚类结果不佳。
• 缺点对初始参数的选择敏感：高斯混合模型需要事先指定聚类的数量以及每个高斯分布的初始参数（如均值和协方差矩阵）。不合适的初始参数选择可能导致模型无法收敛或得到不准确的聚类结果。
• 缺点对大规模数据集计算复杂度较高：高斯混合模型的计算复杂度较高，尤其是在处理大规模数据集时。因为它涉及到对每个数据点计算概率和迭代参数估计过程。

4 密度聚类 DBSCAN

4.1 原理介绍

将密度大的样本归为一类，需要提前确定两个参数，扫描半径 ε 和最少包含点数 *MinPts*，基于这两个参数，确定如下三类样本点：

（1）核心点：以该样本点为中心，ε 邻域（以 ε 为半径的范围）内的样本点数量大于或等于 *MinPts* 个。

（2）边缘点：以该样本点为中心，ε 邻域内的样本点数量小于 *MinPts* 个，但其中包含核心点。

（3）离群点：以该样本点为中心，ε 邻域内的样本点数量小于 *MinPts* 个，且不包含核心点。

还需要定义三类样本点之间的关系：

（1）密度直达：对于核心点 p_1 和样本点 p_2，如果 p_2 在 p_1 的 ε 邻域内，就称 p_2 是从 p_1 的“密度直达”。

（2）密度可达：如果不是“密度直达”，但可以通过多个“密度直达”，就称是“密度可达”。

（3）密度相连：如果不是“密度可达”，但可以通过多个“密度可达”，就称是“密度相连”。

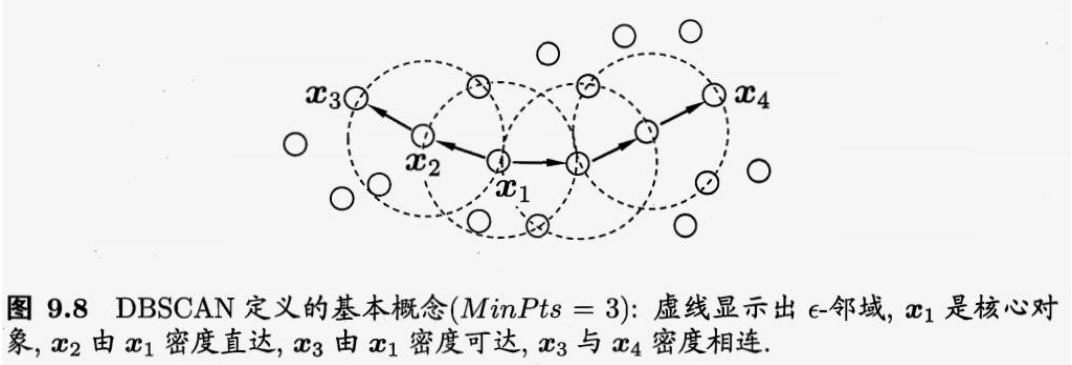


图 9.8 DBSCAN 定义的基本概念($MinPts = 3$): 虚线显示出 ϵ-邻域, x_1 是核心对象, x_2 由 x_1 密度直达, x_3 由 x_1 密度可达, x_3 与 x_4 密度相连.

将簇定义为与核心点密度相连的点的最大集合。

4.2 算法伪代码

主要步骤如下：

1、从数据集中选择一个未被访问的数据点。

2、如果该数据点是核心点，则以该点为种子开始构建一个聚类簇。

3、通过扩展核心点的邻域，将相邻的核心点和边界点加入到聚类簇中。

4、重复步骤 1-3，直到没有更多的核心点可以被添加到聚类簇中。

5、转到下一个未被访问的数据点，重复上述过程，直到所有数据点都被访问。

算法 1-7 行是确定核心点，10-24 行是以任一核心点为出发点，找出由其密度相连的样本生成聚类簇，直到所有核心点均被访问过为止。

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 $\mathcal{C}=\{C_1,C_2,\ldots,C_k\}$

图 9.9 DBSCAN 算法

4.3 优缺点分析

• 优点不需要预先指定聚类数量：与传统的聚类算法不同，密度聚类算法能够自动发现数据中的聚类，而不需要事先指定聚类的数量。它能够发现任意形状的聚类，包括稀疏和非凸形状的聚类。

• 优点对噪声和异常值具有较好的鲁棒性：密度聚类算法对噪声和异常值具有较好的处理能力，可以将它们识别为单独的簇或者噪声点。

• 缺点对参数的选择敏感：密度聚类算法中有几个关键参数需要用户事先选择，特别是密度参数和邻域半径参数的选择对结果影响较大。选择不合适的参数可能导致聚类结果不理想。

• 对高维数据集效果较差：密度聚类算法在处理高维数据集时可能会受到维度诅咒的影响，因为在高维空间中数据点的密度分布会变得非常均匀，难以找到有效的聚类结构。

4.4 西瓜数据集举例

表 9.1 西瓜数据集 4.0

编号	密度	含糖率	编号	密度	含糖率	编号	密度	含糖率
1	0.697	0.460	11	0.245	0.057	21	0.748	0.232
2	0.774	0.376	12	0.343	0.099	22	0.714	0.346
3	0.634	0.264	13	0.639	0.161	23	0.483	0.312
4	0.608	0.318	14	0.657	0.198	24	0.478	0.437
5	0.556	0.215	15	0.360	0.370	25	0.525	0.369
6	0.403	0.237	16	0.593	0.042	26	0.751	0.489
7	0.481	0.149	17	0.719	0.103	27	0.532	0.472
8	0.437	0.211	18	0.359	0.188	28	0.473	0.376
9	0.666	0.091	19	0.339	0.241	29	0.725	0.445
10	0.243	0.267	20	0.282	0.257	30	0.446	0.459

假设 $\varepsilon = 0.11, MinPts = 5$，首先确定核心点 $\Omega = \{3,5,6,8,9,13,14,18,19,24,25,28,29\}$，然后从 Ω 中随机选择一个核心点，找出由它密度相连的所有点作为一个簇，假设 x_8 被选中，则簇 $C_1 = \{6,7,8,10,12,18,19,20,23\}$，然后将 C_1 中包含的核心点从 Ω 中去除，再从更新后的 Ω 中随机选择核心点来生成下一个簇，重复过程直至 Ω 为空。

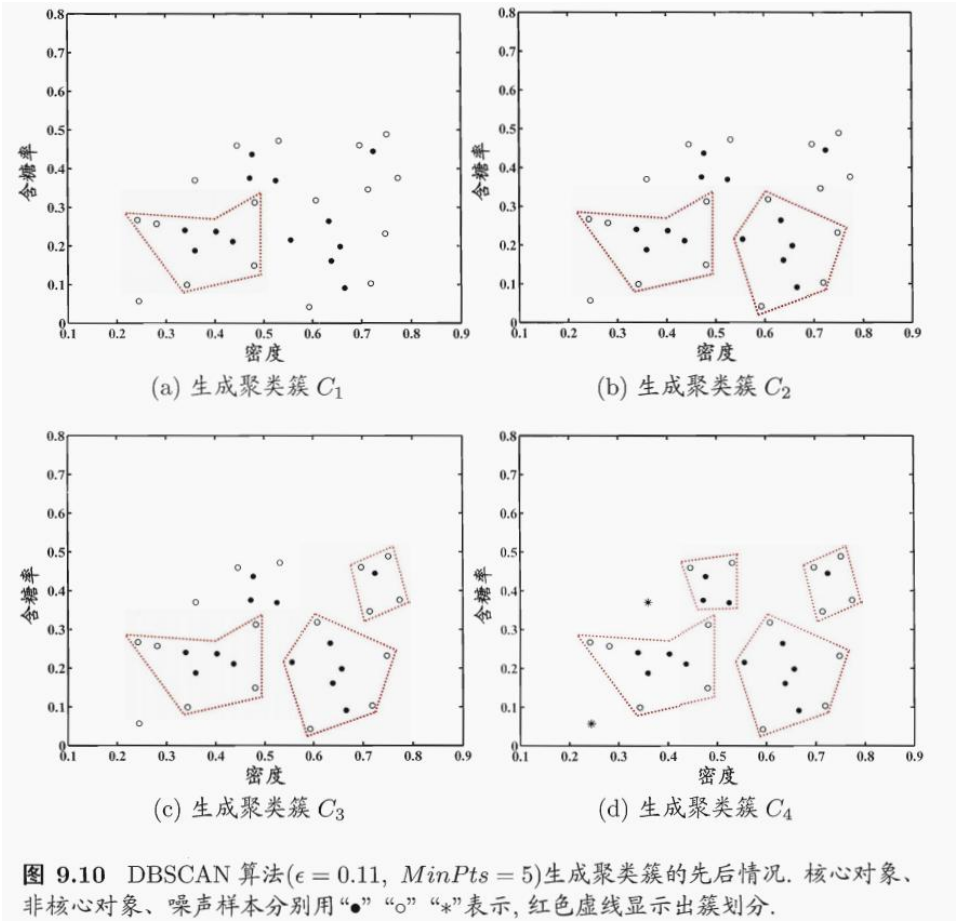


图 9.10 DBSCAN 算法($\epsilon = 0.11$, $MinPts = 5$)生成聚类簇的先后情况. 核心对象、非核心对象、噪声样本分别用“•”“○”“∗”表示, 红色虚线显示出簇划分.

5 层次聚类 AGNES

5.1 原理介绍

首先将距离近的归为一类，归为几类后，类与类之间继续归类，形成一个层次结构树。

核心问题是计算簇之间的距离，采用下述距离计算时，AGNES 算法被相应地称为“单链接”(single-linkage)、“全链接“(complete-linkage) 或“均链接”(average-linkage)算法。

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

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

$$\text{平均距离: } d_{\text{avg}}(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{\boldsymbol{x}\in C_i} \sum_{\boldsymbol{z}\in C_j} \text{dist}(\boldsymbol{x}, \boldsymbol{z}) .$$

通过分裂（自上而下，初始是将所有样本划分为一个簇，再分类为满足要求的簇数）或者凝聚（自下而上，初始是将每一样本作为一个簇，再聚合为满足要求的簇数）方式构建树状图，再根据实际需求横切树状图，获得簇。

5.2 算法伪代码

1 初始化： 将每个数据点视为一个单独的簇。

2 计算相似性： 计算所有簇之间的相似性或距离。常用的距离度量包括欧氏距离、曼哈顿距离、马哈拉诺比斯距离等。

3 合并最相似的簇： 选择最小的相似性度量，将最相似的**两个簇合并成一个新的簇**。

4 更新相似性矩阵： 重新计算合并后新簇与其他簇之间的相似性。

5 重复合并： 重复步骤 3 和 4，直到只剩下一个簇或者达到预定的簇的数量。

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 $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$

图 9.11 AGNES 算法

在第 1 -9 行，算法先对仅含一个样本的初始聚类簇（就是样本本身）和相应

的距离矩阵进行初始化；然后在第 11-23 行，AGNES 不断合并距离最近的聚类簇（14 行中的$i^* < j^*$），并对合并得到的聚类簇的距离矩阵进行更新；上述过程不断重复,直至达到预设的聚类簇数。

5.3 优缺点分析

- 优点不需要预先指定聚类数量：与 K-means 算法不同，层次聚类算法不需要事先指定聚类的数量，能够以层次结构的形式展示聚类结果。
- 优点能够处理非凸形状的簇结构：层次聚类算法能够处理各种形状的簇结构，包括非凸形状的簇，因此对于复杂数据集具有较好的适应性。
- 缺点计算复杂度较高：层次聚类算法的计算复杂度较高，特别是在处理大规模数据集时，会面临较大的计算压力。
- 缺点对初始样本顺序敏感：层次聚类算法对于初始样本的顺序比较敏感，不同的初始顺序可能导致不同的聚类结果。

5.4 西瓜数据集举例

表 9.1 西瓜数据集 4.0

编号	密度	含糖率	编号	密度	含糖率	编号	密度	含糖率
1	0.697	0.460	11	0.245	0.057	21	0.748	0.232
2	0.774	0.376	12	0.343	0.099	22	0.714	0.346
3	0.634	0.264	13	0.639	0.161	23	0.483	0.312
4	0.608	0.318	14	0.657	0.198	24	0.478	0.437
5	0.556	0.215	15	0.360	0.370	25	0.525	0.369
6	0.403	0.237	16	0.593	0.042	26	0.751	0.489
7	0.481	0.149	17	0.719	0.103	27	0.532	0.472
8	0.437	0.211	18	0.359	0.188	28	0.473	0.376
9	0.666	0.091	19	0.339	0.241	29	0.725	0.445
10	0.243	0.267	20	0.282	0.257	30	0.446	0.459

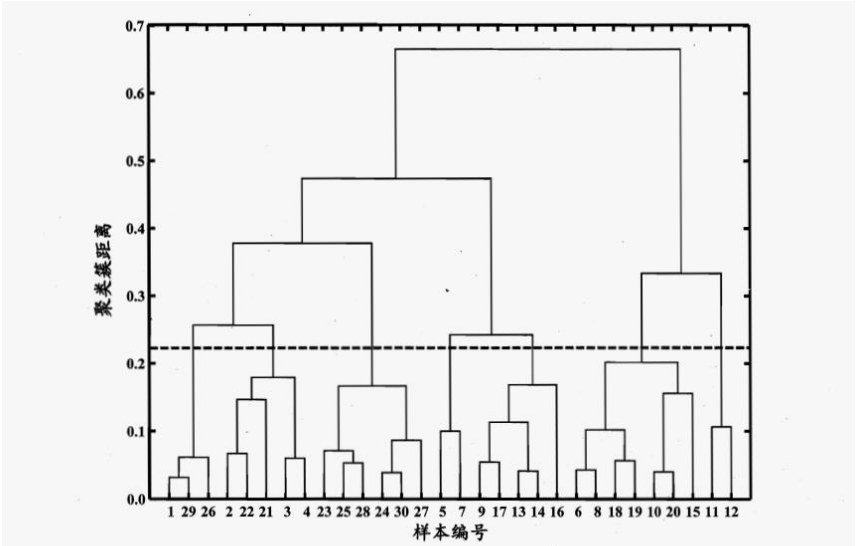


图 9.12 西瓜数据集 4.0 上 AGNES 算法生成的树状图(采用 $d_{\max}$). 横轴对应于样本编号, 纵轴对应于聚类簇距离.

在树状图的特定层次上进行分割, 则可得到相应的簇划分结果. 例如, 以图 9.12 中所示虚线分割树状图, 将得到包含 7 个聚类簇的结果:

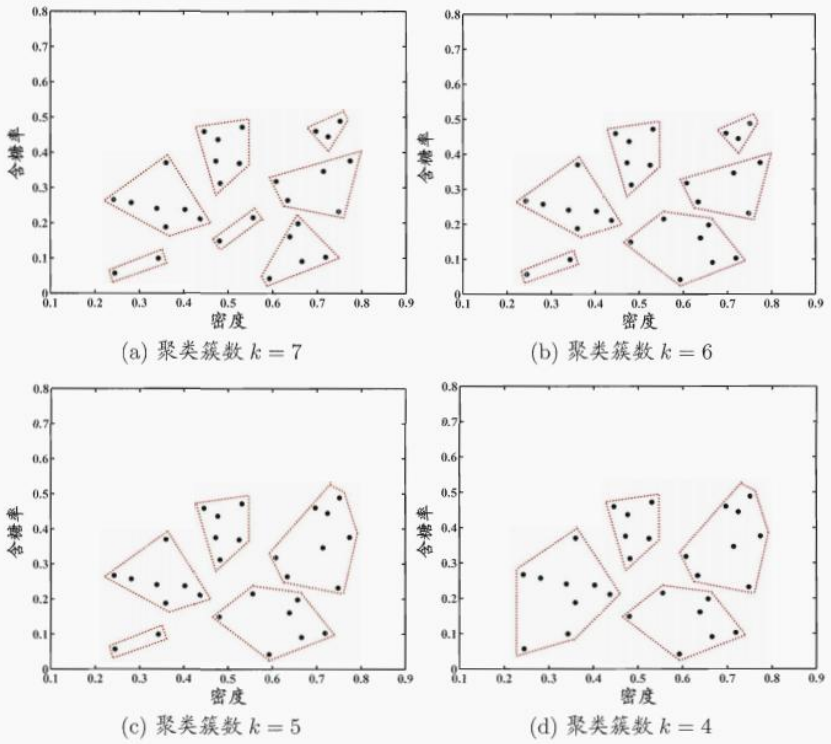
$$C_1 = \{\boldsymbol{x}_1, \boldsymbol{x}_{26}, \boldsymbol{x}_{29}\};\ C_2 = \{\boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4, \boldsymbol{x}_{21}, \boldsymbol{x}_{22}\};$$

$$C_3 = \{\boldsymbol{x}_{23}, \boldsymbol{x}_{24}, \boldsymbol{x}_{25}, \boldsymbol{x}_{27}, \boldsymbol{x}_{28}, \boldsymbol{x}_{30}\};\ C_4 = \{\boldsymbol{x}_5, \boldsymbol{x}_7\};$$

$$C_5 = \{\boldsymbol{x}_9, \boldsymbol{x}_{13}, \boldsymbol{x}_{14}, \boldsymbol{x}_{16}, \boldsymbol{x}_{17}\};\ C_6 = \{\boldsymbol{x}_6, \boldsymbol{x}_8, \boldsymbol{x}_{10}, \boldsymbol{x}_{15}, \boldsymbol{x}_{18}, \boldsymbol{x}_{19}, \boldsymbol{x}_{20}\};$$

$$C_7 = \{\boldsymbol{x}_{11}, \boldsymbol{x}_{12}\}.$$

将分割层逐步提升, 则可得到聚类簇逐渐减少的聚类结果. 例如图 9.13 显示出了从图 9.12 中产生 7 至 4 个聚类簇的划分结果.



(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 $k = 4$

图 9.13 西瓜数据集 4.0 上 AGNES 算法(采用 $d_{\max}$)在不同聚类簇数($k = 7, 6, 5, 4$)时的簇划分结果. 样本点用"•"表示, 红色虚线显示出簇划分.

总结来说，在选择适合的聚类算法时，需要根据实际问题和数据特点综合考虑各种算法的优缺点。对于线性可分的数据，K-means 算法可能是一个简单而有效的选择；对于任意形状和大小的聚类，密度聚类算法如 OPTICS 可能更合适；而对于描述具有潜在概率分布的数据，高斯混合模型（GMM）是一个可行的选择。因此，根据问题的需求和数据的特点，选择最合适的聚类算法非常重要。