第5章

聚类分析

视频讲解

【学习目标】

- 理解聚类的基本概念；
- 掌握距离计算的不同方式；
- 掌握聚类的不同方法。

将物理或抽象对象的集合分成由类似的对象组成的多个类的过程被称为聚类。由聚类所生成的簇是一组数据对象的集合，这些对象与同一个簇中的对象彼此相似，与其他簇中的对象相异。根据计算簇之间的距离来解决分类问题，一般称之为聚类分析。

5.1 聚类的基本概念

“物以类聚，人以群分”，在自然科学和社会科学中，存在着大量的分类问题。例如，市场营销中的市场细分和客户细分问题。大型购物网站收集到客户人口特征、消费行为和喜好方面的数据，并希望对这些客户进行特征分析。可以从客户分类入手，根据客户的年龄、职业、收入、消费金额、消费频率、喜好等方面进行单变量或多变量的客户分组。这种分组是极为常见的客户细分方式，但存在的不足是客户群划分带有明显的主观色彩，需要有丰富的行业经验才能得到比较合理或理想的客户细分，否则得到的分组可能无法充分反映和展现客户的特点，主要表现在：同一客户细分段中的客户在某些特征方面并不相似，而不同客户细分段中的客户在某些特征方面却又很相似。因此，这种客户细分并没有真正起到划分客户群的作用。为解决该问题，希望从数据自身出发，充分利用数据进行客户的客观分组，使诸多特征有相似性的客户被分在同一组，而不相似的客户被区分到另一些组中。聚类分析则是这样一种方法。

聚类分析又称群分析，它是研究（样品或指标）分类问题的一种多元统计分析方法。聚类分析起源于分类学，但是聚类不等于分类。聚类与分类的不同在于，聚类所要求划分的类是未知的。聚类分析内容非常丰富，有系统聚类法、有序样品聚类法、动态聚类法、模糊聚类法、图论聚类法、聚类预报法等。聚类分析能够将一批样本或（变量）数据依据其诸多特征，按照性质上的亲疏程度在没有先验知识的情况下进行自动分类，产生多个分类结果。类内部的个体在特征上具有相似性，不同类间个体特征的差异性较大。

理解聚类分析的关键是理解两个要点："没有先验知识"和"亲疏程度"。为此，可以先看一个例子。如表 5-1 所示，是同一批客户对经常光顾的五座超市在购物环境和服务质量两方面的平均评分。现希望根据这批数据将五座超市分类。

表 5-1　超市的客户评分数据

编号	购物环境/分	服务质量/分
A 超市	73	68
B 超市	66	64
C 超市	84	82
D 超市	91	88
E 超市	94	90

很明显，根据表 5-1 中的数据，若将它们分成两类，则 A 超市和 B 超市是一类，C 超市、D 超市、E 超市是另一类；若将它们分成三类，则 A 超市和 B 超市是一类，D 超市和 E 超市是一类，C 超市单独一类。得到如此分类结果的原因是：在两方面的评分中，A 超市和 B 超市分数较为接近，D 超市和 E 超市较为接近。A 超市和 E 超市之所以没有被分在一起，是由于它们分数相差较远。可见，这种分类结果是在没有指定任何分类标准下完全由样本数据出发而形成的分类。

聚类分析的分类思想与上述分类是一致的。所谓 "没有先验知识"是指没有事先指定分类标准；所谓"亲疏程度"是指在各变量（特征）取值上的总体差异程度。聚类分析基于此实现数据自动分类。

5.2 "亲疏程度"的衡量与计算

在聚类分析中，衡量个体之间的"亲疏程度"是极为重要的，它将直接影响最终的聚类结果。衡量"亲疏程度"一般有两个角度：第一，个体间的相似程度；第二，个体间的差异程度。衡量个体间的相似程度通常可以采用简单相关系数或等级相关系数等；个体间的差异程度通常通过某种距离来测度，以下着重讨论个体间的差异程度。

为定义个体间的距离，应先将每个样本数据看成 k 维空间上的一个点。例如，可将表 5-1 中五个超市样本看成 $k=2$ 的二维空间上的五个点，也就是看成由购物环境和服务质量两个变量构成的二维平面上的五个点，并由此定义某种距离，计算五个点彼此间的"亲疏程度"。通常，点与点之间距离越小，意味着它们越"亲密"，越有可能聚成一类。点与点之间距离越大，意味着它们越"疏远"，越有可能分别属于不同的类。

个体间距离的定义会受 k 个变量类型的影响。由于变量类型一般有定距型和非定距型之分，使得个体间距离的定义也因此不同。

5.2.1 定距型变量个体间的距离计算

如果涉及的 k 个变量都是定距型变量，那么个体间距离的定义通常有以下几种方式。

1. 欧氏距离

欧氏距离(Euclidean Distance)，也称欧几里得度量(Euclidean Metric)，是一个通常采用的距离定义，指在 m 维空间中两个点之间的真实距离，或者向量的自然长度(即该点到原点的距离)。在二维和三维空间中的欧氏距离就是两点之间的实际距离。

在二维空间上，欧氏距离的计算公式为：

$$\rho=\sqrt{(x_2-x_1)^2+(y_2-y_1)^2} \tag{5-1}$$

$$|X|=\sqrt{x_2^2+y_2^2} \tag{5-2}$$

其中，ρ 为点(x_1,y_1)与点(x_2,y_2)之间的欧式距离；$|X|$为点(x_2,y_2)到原点的欧式距离。

在三维空间上，公式为：

$$\rho=\sqrt{(x_2-x_1)^2+(y_2-y_1)^2+(z_2-z_1)^2} \tag{5-3}$$

$$|X|=\sqrt{x_2^2+y_2^2+z_2^2} \tag{5-4}$$

在 n 维空间上，公式为：

$$d(x,y)=\sqrt{(x_2-x_1)^2+(y_2-y_1)^2+\cdots+(x_n-y_n)^2}=\sqrt{\sum_{i=1}^{n}(x_i-y_i)^2} \tag{5-5}$$

2. 曼哈顿距离

想象某人在曼哈顿要从一个十字路口开车到另外一个十字路口，驾驶距离是两点间的直线距离吗？显然不是，除非他能穿越大楼。其实际驾驶距离就是曼哈顿距离(Manhattan Distance)，而这也是曼哈顿距离名称的来源。曼哈顿距离也称为城市街区距离(City Block Distance)。

曼哈顿距离是指两点在南北方向上的距离加上在东西方向上的距离。对于一个具有正南正北、正东正西方向规则布局的城镇街道，从一点到达另一点的距离正是在南北方向上驾驶的距离加上在东西方向上驾驶的距离，因此，曼哈顿距离又称为出租车距离。当坐标轴变动时，点间的距离就会不同。曼哈顿距离示意图在早期的计算机图形学中，屏幕是由像素构成的，是整数，点的坐标也一般是整数，原因是浮点运算很昂贵、很慢而且有误差，如果直接使用 AB 的欧氏距离，则必须要进行浮点运算；如果使用 AC 和 CB，则只要计算加减法即可，这就大大提高了运算速度，而且不管累计运算多少次，都不会有误差。

在二维平面上，两点 $a(x_1,y_1)$ 与 $b(x_2,y_2)$ 间的曼哈顿距离为：

$$d(a,b)=|x_1-x_2|+|y_1-y_2| \tag{5-6}$$

两个 n 维向量 $\boldsymbol{a}(x_{11},x_{12},\cdots,x_{1n})$ 与 $\boldsymbol{b}(x_{21},x_{22},\cdots,x_{2n})$ 间的曼哈顿距离为：

$$d(\boldsymbol{a},\boldsymbol{b})=\sum_{k=1}^{n}|x_{1k}-x_{2k}| \tag{5-7}$$

要注意的是，曼哈顿距离依赖坐标系统的转度，而非系统在坐标轴上的平移或映射。它有如下数学性质：

(1) 非负性：$d(i,j)\geqslant 0$，距离是一个非负的数值；

(2) 同一性：$d(i,i)=0$，对象到自身的距离为 0；

(3) 对称性：$d(i,j)=d(j,i)$，距离是一个对称函数；

(4) 三角不等式：$d(i,j)\leqslant d(i,k)+d(k,j)$，从对象 i 到对象 j 的直接距离不会大于途经的任何其他对象 k 的距离和。

3. 切比雪夫距离

在国际象棋玩法中，国王走一步能够移动到相邻的 8 个方格中的任意一个。那么国王从格子 (x_1,y_1) 走到格子 (x_2,y_2) 最少需要多少步？读者可以自己走走试试。你会发现最少步数总是 $\max(|x_2-x_1|+|y_2-y_1|)$ 步。有一种类似的距离度量方法叫切比雪夫距离(Chebyshev Distance)。

在数学中，切比雪夫距离是向量空间中的一种度量，两个点之间的距离定义是其各坐标数值差绝对值的最大值。从数学的观点来看，切比雪夫距离是由一致范数(Uniform Norm)(或称为上确界范数)所衍生的度量，也是超凸度量(Injective Metric Space)的一种。

二维平面两点 $a(x_1,y_1)$ 与 $b(x_2,y_2)$ 间的切比雪夫距离为：

$$d(a,b)=\max(|x_1-x_2|+|y_1-y_2|) \tag{5-8}$$

两个 n 维向量 $\boldsymbol{a}(x_{11},x_{12},\cdots,x_{1n})$ 与 $\boldsymbol{b}(x_{21},x_{22},\cdots,x_{2n})$ 间的切比雪夫距离为：

$$d(\boldsymbol{a},\boldsymbol{b})=\max(|x_{1i}-x_{2i}|) \tag{5-9}$$

这个公式的另一种等价形式是：

$$d(\boldsymbol{a},\boldsymbol{b})=\lim_{k\to\infty}\left(\sum_{i=1}^{n}|x_{1i}-x_{2i}|^k\right)^{1/k} \tag{5-10}$$

4. 闵氏距离

闵氏距离又称闵可夫斯基距离，它不是一种距离，而是一组距离的定义。闵氏距离出自闵氏空间理论，该理论中的空间指狭义相对论中由一个时间维和三个空间维组成的时空，为俄裔德国数学家闵可夫斯基(H. Minkowski，1864－1909)最先表述。而在该空间中的距离根据维数的不同有不同的表示形式，故闵氏距离为根据维数不同定义的一组距离的定义。

1) 闵氏距离的定义

两个 n 维变量 $\boldsymbol{a}(x_{11},x_{12},\cdots,x_{1n})$ 与 $\boldsymbol{b}(x_{21},x_{22},\cdots,x_{2n})$ 间的闵氏距离定义为：

$$d(\boldsymbol{a},\boldsymbol{b})=\sqrt[p]{\sum_{k=1}^{n}|x_{1k}-x_{2k}|^{p}} \tag{5-11}$$

其中：p 是一个变参数。

当 $p=1$ 时，就是曼哈顿距离；

当 $p=2$ 时，就是欧氏距离；

当 $p\to\infty$ 时，就是切比雪夫距离。

根据变参数的不同，闵氏距离可以表示一类的距离。

2）闵氏距离的缺点

简单来说，闵氏距离的缺点主要有两个：一是将各个分量的量纲(Scale)，也就是“单位”当作相同的看待了；二是没有考虑各个分量的分布(期望、方差等)可能是不同的。

举个例子：二维样本(身高，体重)，其中身高范围是 150～190cm，体重范围是 50～60kg，有三个样本：a(180,50)、b(190,50)、c(180,60)。那么 a 与 b 之间的闵氏距离(无论是曼哈顿距离、欧氏距离或切比雪夫距离)等于 a 与 c 之间的闵氏距离，但是身高的 10cm 真的等价于体重的 10kg 吗？因此用闵氏距离来衡量这些样本间的相似度很有问题。

5. 标准化欧氏距离

标准化欧氏距离(Standardized Euclidean Distance)是针对简单欧氏距离的缺点而做的一种改进方案。标准欧氏距离的思路是针对数据各维分量的分布不一致情况将各个分量“标准化”到均值、方差相等。假设样本集 X 的均值(Mean)为 m，标准差(Standard Deviation)为 s，那么 X 的“标准化变量”(标准化变量的数学期望为 0，方差为 1)表示为：

标准化后的值＝(标准化前的值－分量的均值)/分量的标准差

即

$$X^{*}=\frac{X-m}{s}$$

经过简单的推导就可以得到两个 n 维向量 $a(x_{11},x_{12},\cdots,x_{1n})$ 与 $b(x_{21},x_{22},\cdots,x_{2n})$ 间的标准化欧氏距离的公式为：

$$d(a,b)=\sqrt{\sum_{k=1}^{n}\left(\frac{x_{1k}-x_{2k}}{s_k}\right)^{2}} \tag{5-12}$$

如果将方差的倒数看成是一个权重，这个公式可以看成是一种加权欧氏距离(Weighted Euclidean Distance)。

6. 马氏距离

如图 5-1 所示，有两个正态分布的总体，它们的均值分别为 a 和 b，但方差不一样，则图中的 A 点离哪个总体更近？或者说 A 有更大的概率属于谁？通过对比 A 与均值 a、b 的垂直距离，A 点离均值 a 的垂直距离小于离均值 b 的垂直距离，所以，A 属于左边总体的概率更大，尽管 A 与 a 的欧式距离远一些。这就是马氏距离(Mahalanobis Distance)的直观解释。

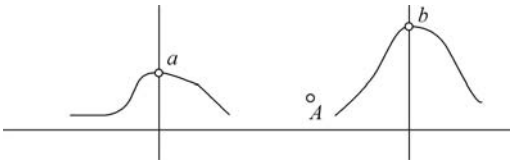

图 5-1　马氏距离示意图

马氏距离是基于样本分布的一种距离，其物理意义就是在规范化的主成分空间中的欧氏距离。所谓规范化的主成分空间就是利用主成分分析对一些数据进行主成分分解，再对所有主成分分解轴做归一化，形成新的坐标轴。由这些坐标轴组成的空间就是规范化的主成分空间。

马氏距离定义如下。

有 M 个样本向量 $\boldsymbol{X}_1,\boldsymbol{X}_2,\cdots,\boldsymbol{X}_m$，协方差矩阵记为 $\boldsymbol{S}$，均值记为向量 $\boldsymbol{\mu}$，则其中样本向量 $\boldsymbol{X}$ 到 $\boldsymbol{\mu}$ 的马氏距离表示为：

$$d(\boldsymbol{X},\boldsymbol{\mu})=\sqrt{(\boldsymbol{X}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{S}^{-1}(\boldsymbol{X}-\boldsymbol{\mu})} \tag{5-13}$$

而其中向量 $\boldsymbol{X}_i$ 与 $\boldsymbol{X}_j$ 之间的马氏距离定义为：

$$d(\boldsymbol{X}_i,\boldsymbol{X}_j)=\sqrt{(\boldsymbol{X}_i-\boldsymbol{X}_j)^{\mathrm{T}}\boldsymbol{S}^{-1}(\boldsymbol{X}_i-\boldsymbol{X}_j)} \tag{5-14}$$

若协方差矩阵是单位矩阵(各个样本向量之间独立同分布)，则公式就变为：

$$d(\boldsymbol{X}_i,\boldsymbol{X}_j)=\sqrt{(\boldsymbol{X}_i-\boldsymbol{X}_j)^{\mathrm{T}}(\boldsymbol{X}_i-\boldsymbol{X}_j)} \tag{5-15}$$

也就是欧氏距离；若协方差矩阵是对角矩阵，则公式就变成了标准化欧氏距离。

马氏距离的特点：

(1) 量纲无关，排除变量之间相关性的干扰。

(2) 马氏距离的计算是建立在总体样本的基础上的，如果拿同样的两个样本，放入两个不同的总体中，最后计算得出的两个样本间的马氏距离通常是不相同的，除非这两个总体的协方差矩阵碰巧相同。

(3) 在计算马氏距离的过程中，要求总体样本数大于样本的维数，否则得到的总体样本协方差矩阵的逆矩阵不存在，这种情况下，用欧式距离计算即可。

视频讲解

7. 夹角余弦

几何中夹角余弦(Cosine)可用来衡量两个向量方向的差异，机器学习中借用这一概念来衡量样本向量之间的差异。

(1) 在二维空间中，向量 $\boldsymbol{A}(x_1,y_1)$ 与向量 $\boldsymbol{B}(x_2,y_2)$ 的夹角余弦公式为：

$$\cos\theta=\frac{x_1x_2+y_1y_2}{\sqrt{x_1^2+y_1^2}\sqrt{x_2^2+y_2^2}} \tag{5-16}$$

(2) 两个 n 维样本点 $a(x_{11},x_{12},\cdots,x_{1n})$ 与 $b(x_{21},x_{22},\cdots,x_{2n})$ 的夹角余弦为：

$$\cos\theta=\frac{a\cdot b}{|a||b|} \tag{5-17}$$

类似的，对于两个 n 维样本点 $a(x_{11},x_{12},\cdots,x_{1n})$ 和 $b(x_{21},x_{22},\cdots,x_{2n})$，可以使用

类似于夹角余弦的概念来衡量它们间的相似程度，即：

$$\cos\theta = \frac{\sum_{k=1}^{n} x_{1k}x_{2k}}{\sqrt{\sum_{k=1}^{n} x_{1k}^2}\sqrt{\sum_{k=1}^{n} x_{2k}^2}} \tag{5-18}$$

夹角余弦取值范围为[－1，1]。夹角余弦越大表示两个向量的夹角越小，夹角余弦越小表示两向量的夹角越大。当两个向量的方向重合时，夹角余弦取最大值1；当两个向量的方向完全相反时，夹角余弦取最小值－1。

5.2.2　计数变量个体间的距离计算

如果涉及的变量是计数(Count)的非连续变量，那么个体间距离的定义通常有以下几种方式。

1. 卡方距离

两个个体(x,y)间卡方(Chi-Square)距离的数学定义为：

$$\mathrm{CHISQ}(x,y) = \sqrt{\sum_{i=1}^{k}\frac{(x_i - E(x_i))^2}{E(x_i)} + \sum_{i=1}^{k}\frac{(y_i - E(y_i))^2}{E(y_i)}} \tag{5-19}$$

其中：x_i是个体x的第i个变量的变量值(频数)；y_i是个体y的第i个变量的变量值(频数)；$E(x_i)$和$E(y_i)$分别为期望频数。

例5.1：表5-2给出了某市各企业职工文化程度的数据资料，试计算甲、乙两企业间的卡方距离。

表5-2　不同企业的职工文化程度

企业	文化程度			合计
	高中及以上	初中	小学及以下	
甲	44 (46)	36 (42)	140 (132)	220
乙	60 (58)	60 (54)	160 (168)	280
合计	104	96	300	500

说明：表格中括号里的数字表示期望频数。

解：根据表格里的数据可以计算甲、乙两企业之间的卡方距离如下：

$$\sqrt{\frac{(44-46)^2}{46} + \frac{(36-42)^2}{42} + \frac{(140-132)^2}{132} + \frac{(60-58)^2}{58} + \frac{(60-54)^2}{54} + \frac{(160-168)^2}{168}} \approx 1.595$$

卡方距离越大，个体与变量取值有显著关系，个体间变量取值差异性较大。

2. Phi方距离

两个个体(x,y)间Phi方(Phi-Square)距离的数学定义为：

$$\mathrm{PHISQ}(x,y)=\sqrt{\frac{\sum_{i=1}^{k}\frac{(x_i-E(x_i))^2}{E(x_i)}+\sum_{i=1}^{k}\frac{(y_i-E(y_i))^2}{E(y_i)}}{n}} \tag{5-20}$$

其中，x_i 是个体 x 的第 i 个变量的变量值(频数)；y_i 是个体 y 的第 i 个变量的变量值(频数)；$E(x_i)$和 $E(y_i)$分别为期望频数；n 为总频数。

例 5.2：根据表 5-2 给出的数据资料，试计算甲、乙两企业间的 Phi 方距离。

解：甲、乙两企业间的 Phi 方距离为：

$$\sqrt{\frac{\frac{(44-46)^2}{46}+\frac{(36-42)^2}{42}+\frac{(140-132)^2}{132}+\frac{(60-58)^2}{58}+\frac{(60-54)^2}{54}+\frac{(160-168)^2}{168}}{500}}\approx 0.0714$$

5.2.3 二值变量个体间的距离计算

如果个体的 k 个变量都是二值变量，则个体之间的距离测度将基于一个如表 5-3 所示的 2×2 的列联表。该表是根据原始数据转换而来的两个体取值的交叉列联表。表中，$a+b+c+d$ 等于变量的总个数；a 为两个体取值都为 1 的变量个数；b 为个体 x 取值为 0 而个体 y 取值为 1 的变量个数；c 为个体 x 取值为 1 而个体 y 取值为 0 的变量个数；d 为两个个体取值都是 0 的变量个数。显然，$a+d$ 的比重描述了两个体之间的相似程度，而 $b+c$ 的比重反映了两个个体之间的差异程度。

表 5-3 两个个体列联表

x、y 简单匹配系数		个体 x	
		1	0
个体 y	1	a	b
	0	c	d

1. 简单匹配系数

简单匹配系数重点考查两个个体的差异性，其数学定义为：

$$S(x,y)=\frac{b+c}{a+b+c+d} \tag{5-21}$$

由式(5-21)可知，简单匹配系数排除了同时拥有或同时不拥有某特征的频数，反映了两个个体间的差异程度。

例 5.3：表 5-4 所示是三位病人的临床检查数据，其中 Y 和 P 表示阳性，N 表示阴性。根据表格数据分析哪两位病人可能得了同样的病。

表 5-4 三位病人临床检查数据

姓名	性别	发烧	咳嗽	检查 1	检查 2	检查 3	检查 4
Jack	M	Y	N	P	N	N	N
Mary	F	Y	N	P	N	P	N
Jim	M	Y	P	N	N	N	N

解：首先给 Y 和 P 值赋值为 1，N 赋值为 0。然后分别计算两两之间的简单匹配系数：

$$S(\text{Jack},\text{Mary})=\frac{0+1}{2+0+1+3}=\frac{1}{6}$$

$$S(\text{Jack},\text{Jim})=\frac{1+1}{1+1+1+3}=\frac{2}{6}$$

$$S(\text{Jim},\text{Mary})=\frac{1+2}{1+2+1+2}=\frac{3}{6}$$

根据以上计算发现：Jack 和 Mary 之间的简单匹配系数最小，可见他们之间的差异度最小，有可能得了同一种病。

2. 雅科比系数

二元属性是一种标称属性，只有两个类别或状态：0 或 1，其中 0 通常表示该属性不出现，而 1 表示出现。如果两种状态对应于 true 和 false，二元属性又称布尔属性。例如，倘若属性 smoker 表示患者对象，1 表示患者抽烟，0 表示患者不抽烟。

如果一个二元属性是对称的，那么它的两种状态具有同等价值并且携带相同的权重，即对某个结果来说，应该用 0 或 1 编码并无偏好（例如，属性 gender 的两种状态男和女）。

如果一个二元属性是非对称的，那么其状态的结果不是同等重要的。如艾滋病病毒（HIV）化验的阳性和阴性结果。为方便计算，我们将用 1 对最重要的结果（通常是稀有的）编码（例如，HIV 阳性），而另一个用 0 编码（例如，HIV 阴性）。给定两个不对称的二元变量，两个都取值 1 的情况（正匹配）被认为比两个都取值 0 的情况（负匹配）更有意义。因此，这样的二元变量经常被认为好像只有一个状态。基于这样变量的相似度被称为非恒定的相似度。对于非恒定的相似度，最著名的评价系数是雅科比系数（Jaccard coefficient），在它的计算中，负匹配的数目被认为是不重要的，因此被忽略。

以上简单匹配系数主要衡量对称的二元属性值间的差异性，而雅科比系数用来衡量非对称二元属性值之间的差异性。其数学定义为：

$$J(x,y)=\frac{b+c}{a+b+c} \tag{5-22}$$

由式(5-22)可知，雅科比系数排除了同时拥有或同时不拥有某特征的频数，反映了两个个体间的差异程度，但它也排除了两个个体同时为 0 的频数，一般认为 0 的状态对研究的意义不显著。

例如，可以对例 5.3 数据计算相应雅科比系数。

$$J(\text{Jack},\text{Mary})=\frac{0+1}{2+0+1}=\frac{1}{3}$$

$$J(\text{Jack},\text{Jim})=\frac{1+1}{1+1+1}=\frac{2}{3}$$

$$J(\text{Jim},\text{Mary})=\frac{1+2}{1+2+1}=\frac{3}{4}$$

根据以上计算发现：Jack 和 Mary 之间的简单匹配系数最小，可见他们之间的差异

度最小，有可能得了同一种病，与例 5.3 得出的结论一样。

5.2.4 其他个体间的距离计算

1. 汉明距离

汉明距离(Hamming Distance)表示两个(相同长度)字对应位不同的数量。汉明距离是以理查德·卫斯里·汉明的名字命名的。在信息论中，两个等长字符串之间的汉明距离是两个字符串对应位置的不同字符的个数。换句话说，它就是将一个字符串变换成另外一个字符串所需要替换的字符个数。例如：

- 1011101 与 1001001 之间的汉明距离是 2。
- 2143896 与 2233796 之间的汉明距离是 3。
- "toned"与"roses"之间的汉明距离是 3。

一般将汉明距离应用在信息编码中，为了增强容错性，应使编码间的最小汉明距离尽可能大。

另一个重要概念是汉明重量，它是字符串相对于同样长度的零字符串的汉明距离，也就是说，它是字符串中非零的元素个数。对于二进制字符串来说，就是 1 的个数，所以 11101 的汉明重量是 4。因此，如果向量空间中的元素 a 和 b 之间的汉明距离等于它们汉明重量的差 $a-b$，则表明向量 $\boldsymbol{a}$ 和 $\boldsymbol{b}$ 之间不同元素的个数为 $\boldsymbol{a}-\boldsymbol{b}$。例如，假设元素 $\boldsymbol{a}$ 为 $[2,2,1,f,0]$，$\boldsymbol{b}$ 为 $[2,2,1,f,1]$，$\boldsymbol{a}$ 的汉明重量为 4，$\boldsymbol{b}$ 的汉明重量为 5，$\boldsymbol{a}$ 和 $\boldsymbol{b}$ 汉明重量的差为 1，而两者的汉明距离也为 1，由此可以得出两个向量的不同元素个数为 1。

汉明重量分析在信息论、编码理论、密码学等领域都有应用。例如，在信息编码过程中，为了增强容错性，应使编码间的最小汉明距离尽可能大。但是，如果要比较两个不同长度的字符串，不仅要进行替换，而且要进行插入与删除的运算，在这种场合下，通常使用更加复杂的编辑距离等算法。

2. 相关系数与相关距离

相关系数(Correlation Coefficient)是衡量随机变量 X 与 Y 相关程度的一种方法，相关系数的取值范围是$[-1,1]$。相关系数的绝对值越大，则表明 X 与 Y 相关度越高。当 X 与 Y 线性相关时，相关系数取值为 1(正线性相关)或-1(负线性相关)。

相关系数计算公式为：

$$\rho_{XY}=\frac{\mathrm{Cov}(X,Y)}{\sqrt{D(X)}\sqrt{D(Y)}}=\frac{E((X-EX)(Y-EY))}{\sqrt{D(X)}\sqrt{D(Y)}} \tag{5-23}$$

其中：$\mathrm{Cov}(X,Y)$为 X、Y 之间的协方差；$D(X)$、$D(Y)$分别为 X、Y 的方差；E 为均值。

相关距离(Correlation Distance)的计算公式为：

$$D_{XY}=1-\rho_{XY} \tag{5-24}$$

3. 信息熵

信息熵(Information Entropy)并不属于一种相似性度量。信息熵是衡量分布的混乱

程度或分散程度的一种度量。分布越分散(或者说分布越平均),信息熵就越大;分布越有序(或者说分布越集中),信息熵就越小。

信息熵的计算公式为:

$$\text{Entropy}(X)=\sum_{i=1}^{n}-p_i\log_2 p_i \tag{5-25}$$

其中,n 为样本集 X 的分类数;p 为 X 中第 i 类元素出现的概率。

信息熵越大表明样本集 X 分类越分散,信息熵越小则表明样本集 X 分类越集中。当 X 中 n 个分类出现的概率一样大时(都是 $1/n$),信息熵取最大值 $\log_2(n)$。当 X 只有一个分类时,信息熵取最小值 0。

5.3　聚类的方法

5.3.1　K-均值聚类算法


视频讲解

1. 基本概念

K-均值聚类算法(K-Means Clustering Algorithm)是一种迭代求解的聚类分析算法,其步骤是随机选取 K 个对象作为初始的聚类中心,然后计算每个对象与各个种子聚类中心之间的距离,把每个对象分配给距离它最近的聚类中心,如图 5-2 所示。聚类中心以及分配给它们的对象就代表一个聚类。每分配一个样本,聚类的聚类中心会根据聚类中现有的对象被重新计算。这个过程将不断重复直到满足某个终止条件。终止条件可以是没有(或最小数目)对象被重新分配给不同的聚类、没有(或最小数目)聚类中心再发生变化,以及误差平方和局部最小。

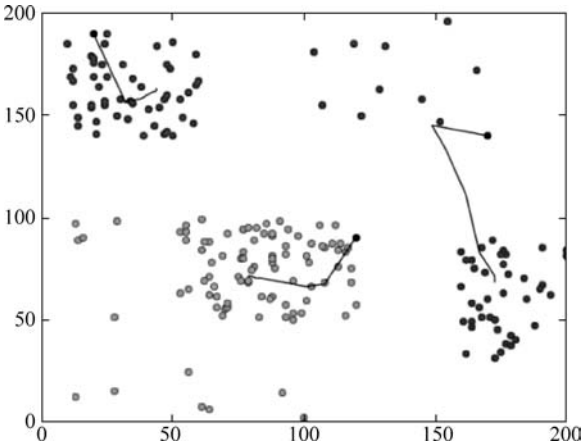

图 5-2　聚类示意图

“聚”是一个将数据集中在某些方面相似的数据成员进行分类组织的过程,聚类就是一种发现这种内在结构的技术,聚类技术经常被称为无监督学习。

K-均值聚类是最著名的划分聚类算法,因简洁和高效的特点使得它成为所有聚类算

法中最广泛使用的。给定一个数据点集合和需要的聚类数目 k（k 由用户指定），K-均值算法可根据某个距离函数反复把数据分入 k 个聚类中。

K-均值聚类算法是一种基本的已知聚类类别数的划分算法。它是很典型的基于距离的聚类算法，采用距离作为相似性的评价指标，即认为两个对象的距离越近，其相似度就越大。该算法认为簇是由距离靠近的对象组成的，因此把得到紧凑且独立的簇作为最终目标。它是使用欧氏距离度量的（简单理解就是两点间直线距离，欧氏距离只是将这个距离定义得更加规范化，扩展到 N 维而已）。它可以处理大数据集，且高效。它的输入自然是数据集和类别数。聚类结果是划分为 k 类的 k 个数据集。根据聚类结果的表达方式又可以分为硬 K-均值（Hard Clusting Method，HCM）算法、模糊 K-均值算法（Fuzzy Clusting Method，FCM）和概率 K-均值算法（Probability Clusting Method，PCM）。

2. 具体步骤

K-均值聚类算法的流程如图 5-3 所示，算法的具体步骤如下：

（1）首先确定一个 k 值，即希望将数据集经过聚类得到 k 个集合。

（2）从数据集中随机选择 k 个数据点作为质心。

（3）对数据集中的每个点，计算其与每个质心的距离（如欧式距离），离哪个质心近，就划分到那个质心所属的集合。

（4）把所有数据归好集合后，共有 k 个集合。然后重新计算每个集合的质心。

（5）如果新计算出来的质心和原来的质心之间的距离小于某一个设置的阈值（表示重新计算的质心的位置变化不大，趋于稳定，或者说收敛），就可以认为聚类已经达到期望的结果，算法终止。

（6）如果新质心和原质心距离变化很大，需要迭代步骤（3）～（5）。

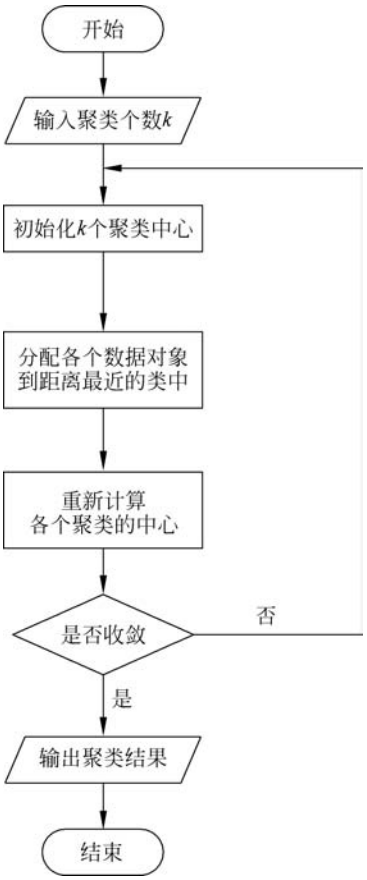

图 5-3　K-均值算法流程图

3. 特点

K-均值算法有如下优点。

（1）原理比较简单，容易实现，收敛速度快。

（2）当结果簇是密集的，而簇与簇之间区别明显时，它的效果较好。

（3）主要需要调参的参数仅仅是簇数 k。

K-均值算法有如下缺点。

（1）k 值需要预先给定，很多情况下 k 值的估计是非常困难的。

（2）K-均值算法对初始选取的质心点是敏感的，不同的随机种子点得到的聚类结果

完全不同，对结果影响很大。

（3）对噪音和异常点比较敏感。

（4）采用迭代方法，可能只能得到局部的最优解，而无法得到全局的最优解。

4. 实例

坐标系上有 6 个点，如表 5-5 和图 5-4 所示。

表 5-5　6 个点数据

数据点	X 坐标值	Y 坐标值
P_1	0	0
P_2	1	2
P_3	3	1
P_4	8	8
P_5	9	10
P_6	10	7

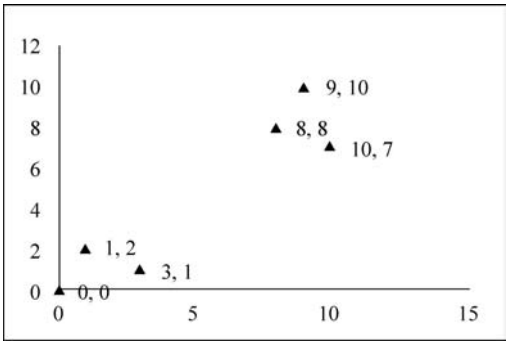

图 5-4　6 个点在坐标轴上

计算过程：

(1)首先令 k 等于 2，随机选择两个点 P_1 和 P_2。

（2）通过勾股定理计算剩余点分别到这两个点的距离，如表 5-6 所示。

表 5-6　剩余点到 P_1 和 P_2 的距离

数据点	数据点到 P_1 的距离	数据点到 P_2 的距离
P_3	3.16	2.24
P_4	11.3	9.22
P_5	13.5	11.3
P_6	12.2	10.3

（3）第一次分组后结果：

组 A：P_1

组 B：P_2、P_3、P_4、P_5、P_6

(4) 分别计算 A 组和 B 组的质心：

A 组质心还是

$$P_1=(0,0)$$

B 组新的质心坐标为：

$$P_{2'}=((1+3+8+9+10)/5,(2+1+8+10+7)/5)=(6.2,5.6)$$

(5) 再次计算每个点到质心的距离，如表 5-7 所示。

表 5-7 其余点到质心的距离

数据点	数据点到质心 P_1 的距离	数据点到质心 $P_{2'}$ 的距离
P_2	2.24	6.3246
P_3	3.16	5.6036
P_4	11.3	3
P_5	13.5	5.2154
P_6	12.2	4.0497

(6) 第二次分组结果：

组 A：P_1、P_2、P_3

组 B：P_4、P_5、P_6

(7) 再次计算质心：

$P_{1'}=(1.33,1)$

$P_{2''}=(9,8.33)$

(8) 再次计算每个点到质心的距离，如表 5-8 所示。

表 5-8 其余点到新质心的距离

数据点	数据点到质心 $P_{1'}$ 的距离	数据点到质心 $P_{2''}$ 的距离
P_1	1.4	12
P_2	0.6	10
P_3	1.4	9.5
P_4	47	1.1
P_5	70	1.7
P_6	56	1.7

(9) 第三次分组结果：

组 A：P_1、P_2、P_3

组 B：P_4、P_5、P_6

可以发现，第三次分组结果和第二次分组结果一致，说明已经收敛，聚类结束。

5.3.2 K-中间值聚类算法

1. 基本概念

K-均值算法存在一个问题，就是当数据中出现了某些数据偏离整体数据很远时，会

给算术平均值带来不利影响。例如，某公司有五个人的年薪是 5 万元，但是有另外一个人的年薪高达 100 万元，那么年薪中间值会是 5 万元(能代表公司的年薪情况)，而平均值达到了 20 万元(完全不能代表公司薪资情况)。这种问题当然也会在 K-均值算法中发生。如果数据有类似情况，采用 K-均值算法得到的结果不是很理想，一个解决办法就是使用 K-中间值(K-Medians)算法代替 K-均值算法，二者算法相似，只是用中值代替平均值，这样可以滤掉数据异常的影响。另外，在计算效率上也会比平均值法更高效。因此，K-中间值是 K-均值的一种变体，是用数据集的中位数而不是均值来计算数据的中心点。

K-中间值算法的优势是使用中位数来计算中心点不受异常值的影响；缺点是计算中位数时需要对数据集中的数据进行排序，速度相对比 K-均值算法较慢。

2. 具体步骤

K-中间值算法的基本步骤如下所示。

(1) 选取初始的中心点的个数。

(2) 计算剩余点的距离到初始的中心点的距离。

(3) 将步骤(2)中剩余点根据不同中心点的距离进行分类，距离相同中心点较近的剩余点归为一类。

(4) 用曼哈顿距离重新计算中心点。

(5) 重复(3)、(4)两个步骤，直到中心点不会变化为止。

3. 实例

有 10 个点：1.(3,8)、2.(3,6)、3.(3,4)、4.(4,5)、5.(4,7)、6.(5,1)、7.(5,5)、8.(7,3)、9.(7,5)、10.(8,5)。将这 10 个点划分为两个类。首先，选取两个初始的中心点为 3 号和 6 号。然后，用曼哈顿距离公式为它们进行划分，得到的结果如表 5-9 所示。

表 5-9 第一次聚类结果

点集合	到中心点 1 的距离	到中心点 2 的距离	类别
1. (3,8)	4	9	1
2. (3,6)	2	7	1
4. (4,5)	2	5	1
5. (4,7)	4	7	1
7. (5,5)	3	4	1
8. (7,3)	5	4	2
9. (7,5)	5	6	1
10. (8,5)	6	7	1

经过第一次的迭代发现：1、2、3、4、5、7、9、10 是一个类，6、8 是另一个类。在坐标轴上的结果如图 5-5 所示。

对第一类点集重新排列：(3,8)、(3,6)、(3,4)、(4,5)、(4,7)、(5,5)、(7,5)、(8,5)。对横坐标排序之后的中位数是 4，对纵坐标排序之后的中位数是 5，这个时候第一类的中心点就变成了(4,5)。

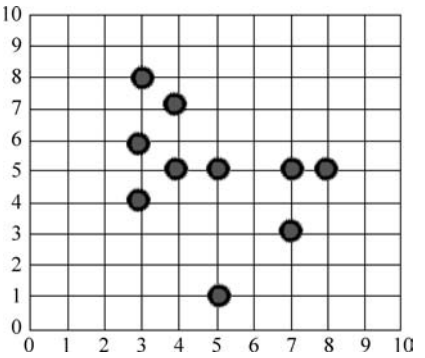

图 5-5 第一次聚类后结果

第二类的点集是 (5,1)和(7,3),中心点就是(6,2)。

重新计算新的聚类,结果如表 5-10 所示。

表 5-10 重新计算聚类结果

点集合	到中心点 1 的距离	到中心点 2 的距离	类别
1. (3,8)	4	9	1
2. (3,6)	2	7	1
3. (3,4)	2	5	1
4. (4,5)	0	5	1
5. (4,7)	2	7	1
6. (5,1)	5	2	2
7. (5,5)	1	4	1
8. (7,3)	5	2	2
9. (7,5)	3	4	1
10. (8,5)	4	5	1

得到的结果还是一样的,迭代结束。

得到了聚类的结果。

5.3.3 均值漂移聚类算法

1. 基本概念

均值漂移(Mean Shift)是一种通用的聚类算法,它的基本原理是:对于给定的一定数量样本,任选其中一个样本,以该样本为中心点划定一个圆形区域,求取该圆形区域内样本的质心,即密度最大处的点,再以该点为中心继续执行上述迭代过程,直至最终收敛。均值漂移聚类是基于滑动窗口的算法,来找到数据点的密集区域。这是一个基于质心的算法,通过将中心点的候选点更新为滑动窗口内点的均值来完成定位每个组/类的中心点,然后对这些候选窗口的相似窗口进行去除,最终形成中心点集及相应的分组。

2. 具体步骤

均值漂移聚类算法的具体步骤如下。

(1) 确定滑动窗口半径 r,以随机选取的中心点 C 和半径为 r 的圆形滑动窗口开始滑动。均值漂移聚类算法类似一种爬山算法,它在每次迭代中都会向密度更高的区域移动,直到收敛。

(2) 每次滑动到新的区域,都会计算滑动窗口内的均值来作为中心点,滑动窗口内的点的数量为窗口内的密度。在每次的移动中,窗口会向密度更高的区域移动。

(3) 移动窗口,计算窗口内的中心点以及窗口内的密度,直到在窗口内没有方向可以容纳更多的点,即一直移动到圆内密度不再增加为止。

(4) 步骤(1)~(3)会产生很多个滑动窗口,当多个滑动窗口重叠时,保留包含最多点的窗口,然后根据数据点所在的滑动窗口进行聚类。

3. 特点

- 优点:不同于 K-均值算法,均值漂移聚类算法不需要我们知道有多少类/组,该算法相比于 K-均值算法,受均值影响较小。
- 缺点:窗口半径 r 的大小选择对结果的影响较大。

5.3.4 基于密度的聚类算法

1. 基本概念

基于密度的聚类算法(Density-Based Spatial Clustering of Applications with Noise, DBSCAN)与划分和层次聚类方法不同,它将簇定义为密度相连的点的最大集合,能够把具有足够高密度的区域划分为簇,并可在噪声的空间数据库中发现任意形状的聚类。

以下是基于密度的聚类算法中的几个重要定义。

- E 邻域:给定对象半径为 E 内的区域称为该对象的 E 邻域。
- 核心对象:如果给定对象 E 邻域内的样本点数大于或等于 MinPts(给定的最小点数),则称该对象为核心对象。
- 直接密度可达:对于样本集合 D,如果样本点 q 在 p 的 E 邻域内,并且 p 为核心对象,那么对象 q 从对象 p 直接密度可达。
- 密度可达:对于样本集合 D,给定一串样本点 $p_1, p_2, \cdots, p_n, p=p_1, q=p_n$,假如对象 p_i 从 p_{i-1} 直接密度可达,那么对象 q 从对象 p 密度可达。
- 密度相连:存在样本集合 D 中的一点 o,如果对象 o 到对象 p 和对象 q 都是密度可达的,那么 p 和 q 密度相连。

可以发现,密度可达是直接密度可达的传递闭包,并且这种关系是非对称的。密度相连是对称关系。基于密度的聚类算法的目的是找到密度相连对象的最大集合。

例如:假设半径 $E=3$,MinPts=3,点 p 的 E 邻域中有点$\{m, p, p_1, p_2, o\}$,点 m 的 E 邻域中有点$\{m, q, p, m_1, m_2\}$,点 q 的 E 邻域中有点$\{q, m\}$,点 o 的 E 邻域中有点$\{o, p, s\}$,点 s 的 E 邻域中有点$\{o, s, s_1\}$。

那么核心对象有 p、m、o、s(q 不是核心对象,因为它对应的 E 邻域中点数量等于 2,小于 MinPts=3);点 m 从点 p 直接密度可达,因为 m 在 p 的 E 邻域内,并且 p 为核心

对象；点 q 从点 p 密度可达，因为点 q 从点 m 直接密度可达，并且点 m 从点 p 直接密度可达；点 q 到点 s 密度相连，因为点 q 从点 p 密度可达，并且 s 从点 p 密度可达。

如图 5-6 所示的点是分布在样本空间的众多样本，现在的目标是把这些在样本空间中距离相近的聚成一类。我们发现 A 点附近的点密度较大，且各点根据一定的规则在周围移动，最终收纳了 A 附近的 5 个点，并标记为同样的颜色定为同一个簇。其他没有被收纳的根据一样的规则成簇。

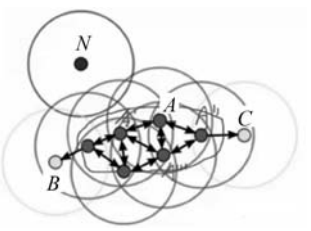

图 5-6　基于密度的聚类示意图

形象来说，我们可以认为这是系统在众多样本点中随机选中一个，围绕这个被选中的样本点画一个圆，规定这个圆的半径以及圆内最少包含的样本点，如果在指定半径内有足够多的样本点在内，那么这个圆圈的圆心就转移到这个内部样本点，继续去圈附近其他的样本点。等到这个移动的圆圈发现所圈住的样本点数量少于预先指定的值，就停止了。我们称最开始那个点为核心点，如 A，停下来的那个点为边界点，如 B、C，没被圈到的那个点为离群点，如 N。[①]

2. 具体步骤

(1) 首先确定半径 r 和 minPts，从一个没有被访问过的任意数据点开始，查看在以这个点为中心、r 为半径的圆内包含的点的数量是否大于或等于 minPts，如果大于或等于 minPts，则该点被标记为中心点(Central Point)，反之则会被标记为噪声点(Noise Point)。

(2) 重复(1)的步骤，如果一个噪声点存在于某个中心点为半径的圆内，则这个点被标记为边缘点，反之仍为噪声点。重复步骤(1)，直到所有的点都被访问过。

3. 特点

- 优点：不需要知道簇的数量，能够处理环形、不规则形状等(如图 5-7 所示)，弥补了 K-均值算法的缺陷。
- 缺点：需要确定半径 r 和 minPts。


图 5-7　环形或不规则形状点集

① 参考：https://blog.csdn.net/huacha__/article/details/81094891

5.3.5 高斯混合模型聚类算法

1. 基本概念

高斯混合模型(Gaussian Mixture Model,GMM)是聚类算法的一种,它相比较 5.3.2 节描述的 K-均值算法的最大优点是能对如图 5-7 所示的图形作出正确的判断。

K-均值算法是确切给出每个样本被分配到某一个簇,称为硬分配;而高斯混合模型则是给出每个样本被分配到每个簇的概率,最后从中选取一个最大的概率对应的簇作为该样本被分配到的簇,称为软分配。

使用高斯混合模型做聚类首先假设数据点是呈高斯分布的,相比于 K-均值算法假设数据点是圆形的,高斯分布(椭圆形)给出了更多的可能性。有两个参数用来描述簇的形状:均值和标准差。所以这些簇可以采取任何形状的椭圆形,因为在 x,y 方向上都有标准差。因此,每个高斯分布被分配给单个簇。

2. 具体步骤

(1) 选择簇的数量(与 K-均值算法类似)并随机初始化每个簇的高斯分布参数(均值和标准差)。也可以先观察数据给出一个相对精确的均值和标准差。

(2) 给定每个簇的高斯分布,计算每个数据点属于每个簇的概率。一个点越靠近高斯分布的中心就越可能属于该簇。

(3) 基于这些概率来计算高斯分布参数,使得数据点的概率最大化,可以使用数据点概率的加权来计算这些新的参数,权重就是数据点属于该簇的概率。

(4) 重复步骤(2)~(3)直到结果变化不大。

3. 特点

(1) 高斯混合模型聚类算法使用均值和标准差,簇可以呈现出椭圆形而不是仅仅限制于圆形。K-均值聚类算法是高斯混合模型聚类算法的一个特殊情况,是方差在所有维度上都接近于 0 时簇就会呈现出圆形。

(2) 高斯混合模型聚类算法是使用概率,所以一个数据点可以属于多个簇。例如数据点 X 可以有 20%的概率属于 A 簇,80%的概率属于 B 簇。也就是说高斯混合模型聚类算法可以支持混合资格。

5.3.6 层次聚类算法

层次聚类算法分为两类:凝聚法和分裂法。

1. 凝聚法

1) 基本概念

凝聚层次聚类(Hierarchical Agglomerative Clustering,HAC)是自下而上的一种聚类算法。凝聚层次聚类首先将每个数据点视为一个单一的簇,然后计算所有簇之间的距

离来合并簇，直到所有的簇聚合成为一个簇为止。

图 5-8 所示为凝聚层次聚类的一个实例。

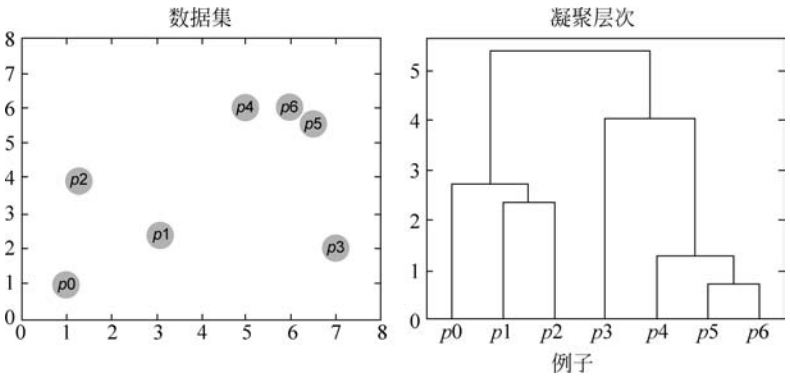

图 5-8　凝聚层次聚类实例

2）具体步骤

凝聚层次聚类算法的具体步骤如下所示。

（1）首先将每个数据点视为一个单一的簇。

（2）计算两两类簇之间的距离，找到距离最小的两个类簇 $c1$ 和 $c2$。

（3）合并类簇 $c1$ 和 $c2$ 为一个类簇。

（4）重复步骤(2)～(3)直到所有的数据点合并完成，或者达到终止条件。

其中簇间距离的计算方法有以下几种。

（1）取两个类中距离最近的两个样本的距离作为这两个集合的距离，也就是说，最近两个样本之间的距离越小，这两个类之间的相似度就越大。

（2）取两个集合中距离最远的两个点的距离作为两个集合的距离。

（3）把两个集合中点的两两距离全部放在一起求平均值，相对也能得到合适一些的结果。

（4）取两两距离的中值，与取均值相比更加能够解除个别偏离样本对结果的干扰。

（5）把两个集合中点的两两距离全部放在一起求和，然后除以两个集合中的元素个数。

（6）求每个集合的中心点(即将集合中所有元素的对应维度相加然后再除以元素个数得到的一个向量)，然后用中心点代替集合再去求集合间的距离。

3）特点

- 优点：①不需要知道有多少个簇。②对于距离度量标准的选择并不敏感。
- 缺点：效率低。

4）实例

举个例子：在平面上有 6 个点：$p0(1,1)$，$p1(1,2)$，$p2(2,2)$，$p3(4,4)$，$p4(4,5)$，$p5(5,6)$，对这 6 个点进行聚类，步骤如下。

（1）首先将所有的样本看作是一个类簇，这样可以得到初始的类簇有 6 个，分别为 $c1(p0)$，$c2(p1)$，$c3(p2)$，$c4(p3)$，$c5(p4)$，$c6(p5)$。

(2) 计算两两类簇间的最小距离。

(3) 合并具有最小簇间距离的类簇 $c1$ 和 $c2$，得到新的聚类结果 $c1(p0,p1)$，$c3(p2)$，$c4(p3)$，$c5(p4)$，$c6(p5)$。

(4) 若是要求聚成 5 个类别的话，到这里就可以结束了。但是如果设定了一个阈值 f，要求若存在距离小于阈值 f 的两个类簇时则将两个类簇合并并且继续迭代，则需回到步骤(2)继续迭代从而得到新的聚类结果。

2. 分裂法

1) 基本概念

分裂法[①]指的是初始时将所有的样本归为一个类簇，然后依据某种准则进行逐渐的分裂，直到达到某种条件或者达到设定的分类数目。

2) 具体步骤

分裂法的具体步骤如下：

(1) 将样本集中所有的样本归为一个类簇。

(2) 在同一个类簇(记为 c)中计算两两样本之间的距离，找出距离最远的两个样本 a，b。

(3) 将样本 a、b 分配到不同的类簇 $c1$ 和 $c2$ 中。

(4) 计算原类簇(c)中剩余的其他样本点和 a、b 的距离，若是 $\mathrm{dis}(a)<\mathrm{dis}(b)$，则将样本点归到 $c1$ 中，否则归到 $c2$ 中。

(5) 重复步骤(2)～(4)，直到聚类的数目没有变化或者达到设定的循环次数。

3) 特点

- 优点：①距离和规则的相似度容易定义，计算限制较少；②无须预定类的数量且可以发现类的层次关系。
- 缺点：算法很可能聚类成链状。

4) 实例

在平面上有 6 个点：$p0(1,1)$，$p1(1,2)$，$p2(2,2)$，$p3(4,4)$，$p4(4,5)$，$p5(5,6)$，对这 6 个点进行聚类，步骤如下。

(1) 将所有的点归为一个类簇 $c(p0,p1,p2,p3,p4,p5)$。

(2) 在类簇 c 中计算它们的距离(简单的欧式距离)可以得到如表 5-11 所示的结果。

表 5-11　类簇距离计算结果

dis	$p0$	$p1$	$p2$	$p3$	$p4$	$p5$
$p0$	0	1	sqrt(2)	sqrt(18)	5	sqrt(41)
$p1$	1	0	1	sqrt(13)	sqrt(18)	sqrt(32)
$p2$	sqrt(2)	1	0	sqrt(8)	sqrt(13)	5
$p3$	sqrt(18)	sqrt(13)	sqrt(8)	0	1	sqrt(5)
$p4$	5	sqrt(18)	sqrt(13)	1	0	sqrt(2)
$p5$	sqrt(41)	sqrt(32)	5	sqrt(5)	sqrt(2)	0

① 参考：https://blog.csdn.net/u012500237/article/details/65437525

由表 5-11 可以看出距离最远的两个点为 $p0$ 和 $p5$。

(3) 将 $p0$ 分配到类簇 $c1$，将 $p5$ 分配到类簇 $c2$。

(4) 查表可以看出，剩余的点中 $p1$ 和 $p2$ 与 $p0$ 的距离小，所以将它们两个归到类簇 $c1$ 中；$p3$ 和 $p4$ 与 $p5$ 的距离小，所以将它们两个归到类簇 $c2$ 中。这样就得到了新的聚类，结果 $c1=(p1,p2,p3)$，$c2=(p3,p4,p5)$。

(5) 若是只要求聚类成两个，则这个聚类到此结束，最终聚类结果是($p1$，$p2$，$p3$)和($p3$，$p4$，$p5$)。若要求同一个类中，最大样本距离不大于 sqrt(2)，那么上述的分类结果没有达到要求，则需要返回到步骤(2)继续聚类，因为 $c1$ 中样本的距离都不大于 sqrt(2)，所以不需要再分了；而类簇 $c2$ 中的 dis($p3$，$p5$)=sqrt(5)>sqrt(2)，还需要继续分，$c2$ 最后分聚类成两个类($p3$，$p4$)和($p5$)，这样最终得到了三个类簇($p1$，$p2$，$p3$)、($p3$，$p4$)和($p5$)。

5.3.7 图团体检测算法

1. 基本概念

当样本以及样本之间的关系可以被表示为一个网络或图(Graph)时，可能存在这样的需求：想找出网络中联系比较“紧密”的样本。举个例子，在社交网站中，用户以及用户之间的好友关系可以表示成如图 5-9 所示的无向图，图中的顶点表示每个用户，顶点之间的边表示用户是否为好友关系。

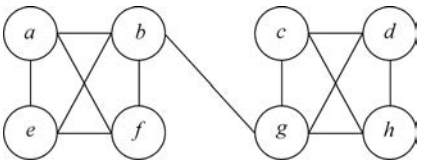

图 5-9 用户好友关系

这个例子涉及的数量规模比较小，直观上可以看出 a、b、e、f 之间的关系比较密切，c、d、g、h 之间的关系比较密切，而这两个集合之间的关系就不那么密切了。但在更大的数据上就不太容易人工找出这些关系密切的集合了，那么如何通过算法把我们的用户按照关系的密切程度划分成一个个集合呢？这就是图团体检测(Graph Community Detection)算法要完成的工作。

图团体通常被定义为一种顶点(Vertice)的子集，每个子集中的顶点相对于网络的其他顶点来说要连接得更加紧密。

2. 具体步骤

(1) 首先初始分配每个顶点到其自己的团体，然后计算整个网络的模块性 M。要求每个团体对(Community Pair)至少被一条单边链接，如果有两个团体融合到了一起，该算法就计算由此造成的模块性改变 ΔM。

(2) 取 ΔM 出现了最大增长的团体对，然后融合，为这个聚类计算新的模块性 M，并记录下来。

(3) 重复步骤(1)和(2)——每次都融合团体对，这样最后得到 ΔM 的最大增益，然后记录新的聚类模式及其相应的模块性分数 M。

3. 特点

图团体检测是限制图论中一个热门的研究领域，这类算法在大部分结构化数据所显示的网络、网状数据都有非常好的性能。缺点是可能会忽略一些小的集群，且只适用于结构化的图模型。

4. 实例

将图 5-9 的社会网络中的顶点进行图团体检测算法，即把顶点按照关系的密切程度划分成一个个集合。

首先，将图 5-9 的图转换为如表 5-12 所示的邻接矩阵的形式。

表 5-12　用户好友关系邻接矩阵

—	a	b	c	d	e	f	g	h
a	0	1	0	0	1	1	0	0
b	1	0	0	0	1	1	1	0
c	0	0	0	1	0	0	1	1
d	0	0	1	0	0	0	1	1
e	1	1	0	0	0	1	0	0
f	1	1	0	0	1	0	0	0
g	0	1	1	1	0	0	0	1
h	0	0	1	1	0	0	1	0

在表 5-12 中，1 表示两用户是好友关系，0 则表示不是好友关系。

顶点的度(Degree)：表示有多少个顶点与其相连，通常标记为 k。

模块性(Modularity)：定义为如下公式，它是衡量图团体划分质量的一种标准，划分得越好，M 的值越大。

$$M=\frac{1}{2L}\sum_{i,j=1}^{N}\left(A_{ij}-\frac{k_ik_j}{2L}\right)\delta(c_i,c_j) \tag{5-26}$$

其中：L 表示图包含的边的数量；N 表示顶点数量；k_i 表示顶点 i 的度，A_{ij} 的值为邻接矩阵中的值；c_i 表示顶点 i 的聚类，δ 则是克罗内克函数(Kronecker-delta Function)。函数的逻辑很简单，两个参数相等则返回 1，不等则返回 0。所以如果顶点 i、j 属于同一聚类，则 $\delta(c_i,c_j)$ 返回 1，不属于同一聚类则 $\delta(c_i,c_j)$ 返回 0。

k_i、k_j 表示顶点 i、j 的度，那么 $k_ik_j/2L$ 表示当该网络是随机分配时顶点 i 和 j 之间的预期边数。当 k_i、k_j 都比较大时，连接顶点 i、j 的边出现的概率就越大；k_i、k_j 任何一个比较小时，或者都小时，连接顶点 i、j 的边出现的概率就越小。

$A_{ij}-k_ik_j/2L$ 就表示了网络的真实结构和随机组合时预期结构之间的差。研究它的值可以发现，当 $A_{ij}=1$ 且 $k_ik_j/2L$ 很小时，其返回的值最高。这意味着，当在顶点 i 和 j 之间存在连接，但是 i、j 之间存在连接的预期又比较小时，得到的值更高。再有，如果把这样的两个顶点分到一个聚类，则能提高网络的模块性，如果分在不同的网络，则并不能提高网络的模块性。

首先把用户随机地划分成两类 a、d、f、g 为一类，b、c、e、h 为另一类，如图 5-10 所示。然后计算这种划分方式下网络的模块性。按照公式，首先计算求和项，计算出每对顶点对模块性的贡献值。根据 δ 函数可知不同类顶点对的贡献值是 0，因此只需要计算同一类的顶点对，结果如表 5-13 所示。

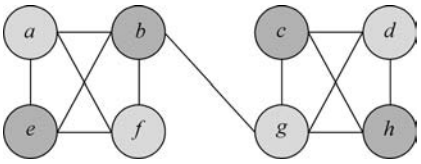

图 5-10　将用户随机分为两类

表 5-13　第一次求和项计算结果

—	$a(3)$	$b(4)$	$c(3)$	$d(3)$	$e(3)$	$f(3)$	$g(4)$	$h(3)$
$a(3)$	0	0	0	$0-3\times3/26=-0.35$	0	$1-3\times3/26=0.65$	$0-3\times4/26=-0.46$	0
$b(4)$	0	0	$0-3\times4/26=-0.46$	0	$1-3\times4/26=0.54$	0	0	$0-3\times4/26=-0.46$
$c(3)$	0	$0-3\times4/26=-0.46$	0	0	$0-3\times3/26=-0.35$	0	0	$1-3\times3/26=0.65$
$d(3)$	$0-3\times3/26=-0.35$	0	0	0	0	$0-3\times3/26=-0.35$	$1-3\times4/26=0.54$	0
$e(3)$	0	$1-3\times4/26=0.54$	$0-3\times3/26=-0.35$	0	0	0	0	$0-3\times3/26=-0.35$
$f(3)$	$1-3\times3/26=0.65$	0	0	$0-3\times3/26=-0.35$	0	0	$0-3\times4/26=-0.46$	0
$g(4)$	$0-3\times4/26=-0.46$	0	0	$1-3\times4/26=0.54$	0	$0-3\times4/26=-0.46$	0	0
$h(3)$	0	$0-3\times4/26=-0.46$	$1-3\times3/26/=0.65$	0	$0-3\times3/26=-0.35$	0	0	0

对矩阵内所有元素求和，除以 $2L$ 得到模块性值：

$M=-1.306/2L=-1.306/26=-0.05$

然后按照观察的结果，根据联系是否密切来划分，得到如图 5-11 所示的结果。

再次计算求和项，结果如表 5-14 所示。

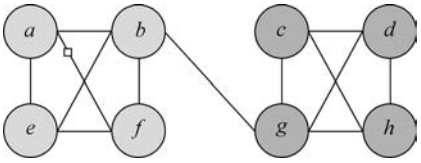

图 5-11　聚类一次后结果

表 5-14　第二次求和项计算结果

—	$a(3)$	$b(4)$	$c(3)$	$d(3)$	$e(3)$	$f(3)$	$g(4)$	$h(3)$
$a(3)$	0	0.54	0	0	0.65	0.65	0	0
$b(4)$	0.54	0	0	0	0.54	0.54	0	0
$c(3)$	0	0	0	0.54	0	0	0.54	0.65
$d(3)$	0	0	0.54	0	0	0	0.54	0.65
$e(3)$	0.65	0.54	0	0	0	0.65	0	0
$f(3)$	0.65	0.54	0	0	0.65	0	0	0
$g(4)$	0	0	0.54	0.54	0	0	0	0.54
$h(3)$	0	0	0.65	0.65	0	0	0.54	0

计算得到模块性值为：

$$M = 12.22/2L = 12.22/26 = 0.47$$

归一化因子 $2L$ 将模块性的上限值设置成了 1。模块性接近或小于 0 表示该网络的当前聚类没有用处。模块性越高，该网络聚类成不同团体的程度就越好。通过使模块性最大化，可以找到聚类该网络的最佳方法。下面来分析如何找到一种聚类方式，使得网络的模块性最大。

首先想到的是遍历所有可能的聚类方式。组合学(Combinatorics)告诉我们，对于一个仅有 8 个顶点的网络，即存在 4140 种不同的聚类方式。16 个顶点的网络的聚类方式将超过 100 亿种。32 个顶点网络的可能聚类方式更是将超过 128 septillion(10^{24})种。如果网络有 80 个顶点，那么其可聚类方式的数量就已经超过了可观测宇宙中的原子数量。

有一种快速贪婪模块性最大化(Fast-Greedy Modularity-Maximization)的算法，这种算法在一定程度上类似于上面描述的层次聚类算法。只是这里并不根据距离来融合聚类，而是根据模块性的改变来对团体进行融合。

算法过程如下：

(1) 每个顶点独自构成一个聚类，然后计算整个网络的模块性 M。

(2) 尝试选择两个聚类融合到一起，计算由此造成的模块性改变 ΔM。

(3) 取 ΔM 出现了最大增长的两个聚类进行融合，然后为这个聚类计算新的模块性 M，并记录下来。

(4) 不断重复步骤(2)和(3)，每次都融合一对聚类，得到 ΔM 的最大增益，然后记录新的聚类模式及其相应的模块性 M。

这样，直到所有的顶点都被分组成了一个聚类时为止。该算法会检查这个聚类过程中的所有记录，然后找到其中返回了最高 M 值的聚类模式，这就是图团体检测算法得到的聚类结构。

5.4 习题

1. 简述聚类算法的实质是什么，常用的几种聚类算法分别适用于哪些场合。

2. 分别取 $k=2$ 和 3，利用 K-均值聚类算法对以下的点聚类：(2,1)、(1,2)、(2,2)、(3,2)、(2,3)、(3,3)、(2,4)、(3,5)、(4,4)、(5,3)，并讨论 k 值以及初始聚类中心对聚类结果的影响。

3. 根据表 5-15 中的数据判断属性的类型，比较 Cat 与其他 3 种动物的相似性。

表 5-15 第 3 题数据

	Small size	Big size	2 legs	4 legs	Hair	Mane	Feather	Hunt	Run	Fly
Owl	1	0	1	0	0	0	1	1	0	1
Cat	1	0	0	1	1	0	0	1	0	0
Tiger	0	1	0	1	1	0	0	1	1	0
Lion	0	1	0	1	1	1	0	1	1	0

4. 表 5-16 是机器学习数据库中鸢尾花的一个子集，其中属性 type 是花的类别，请利用 K-均值聚类算法将其聚为 3 类，并与真实类别进行比较。

表 5-16　第 4 题数据

sep_length	sep_width	pet_length	pet_width	type
5.7	2.9	4.2	1.3	Iris-versicolor
6.2	2.9	4.3	1.3	Iris-versicolor
5.7	2.8	4.1	1.3	Iris-versicolor
6.3	3.3	6.0	2.5	Iris-virginica
5.8	2.7	5.1	1.9	Iris-virginica
7.1	3.0	5.9	2.1	Iris-virginica
5.1	3.8	1.6	0.2	Iris-setosa
4.6	3.2	1.4	0.2	Iris-setosa
5.3	3.7	1.5	0.2	Iris-setosa