聚类

1. 聚类任务

在无监督学习中，训练样本的标注信息是未知的， 需要挖掘数据的内在规律来寻找信息，应用最广的是聚类
聚类算法的思想:
把数据集分成数据簇，每一个簇对应于一个潜在的概念，聚类算法无法确定这样的概念是什么，但是可以依赖于聚类完成对数据集的划分，而标记或者是概念可以由使用者来命名.
数学表示:
假定样本集可以写作$D=\{x_1,x_2,\cdots,x_m\}$，每一个样本$x_i=(x_{i1},x_{i2},\cdots,x_{im})$，聚类算法将样本划分为k个不相交的簇$\{C_l|l=1,2,3\cdots,k\}$, 满足$C_i\cap C_j=\varnothing(i\neq j)$, $D=\cup_{l=1}^{k}C_l$, 用$\lambda_j\in\{1,2,\cdots,k\}$，来表示样本$x_j$的簇标记, 聚类的结果表示为$\lambda=(\lambda_1,\lambda_2,\cdots,\lambda_m)$

2. 性能度量

我们需要性能度量来评价算法的效果, 而且性能评价指标也可以作为优化函数. 我们希望簇内相似度高并且簇间相似度低.

$$性能度量：\begin{cases}内部指标：直接考察聚类结果\\外部指标：依赖于外部的参考模型\end{cases}$$

2.1 外部指标的度量方法

设基于参考模型$\{C_l^*|l=1,2,3\cdots,s\}$(通常$s\neq k$)，令$\lambda,\lambda^*$分别是对应的簇标记向量，考虑两两样本的配对，共$\frac{m(m-1)}{2}$组，定义下面的统计量:

$$a=|SS|,SS=\{(x_i,x_j)|\lambda_i=\lambda_j,\lambda_i^*=\lambda_j^*,i<j\}$$

$$b=|SD|,SD=\{(x_i,x_j)|\lambda_i=\lambda_j,\lambda_i^*\neq\lambda_j^*,i<j\}$$

$$c=|DS|,DS=\{(x_i,x_j)|\lambda_i\neq\lambda_j,\lambda_i^*=\lambda_j^*,i<j\}$$

$$d=|DD|,DD=\{(x_i,x_j)|\lambda_i\neq\lambda_j,\lambda_i^*\neq\lambda_j^*,i<j\}$$

进而得到外部指标:

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

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

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

以上度量指标均定义在$[0,1]$, 值越大越好.

2.2 内部指标的度量方法

对于$\mathcal{C}=\{C_l|l=1,2,3\cdots,k\}$，考虑其中一簇$C$定义簇C内的平均距离和最大距离

$$avg(C) = \frac{2}{|C|(|C|-1)} \sum_{1 \le i < j \le |C|} dist(x_i, x_j)$$
$$diam(C) = \max_{1 \le i < j \le |C|} dist(x_i, x_j)$$

对于不同的两个簇C_i, C_j，簇间的最小距离和簇中心点距离

$$d_{min}(C_i, C_j) = \min_{x_i \in C_i, X_j \in C_j} dist(x_i, x_j)$$
$$d_{cen}(C_i, C_j) = dist(\mu_i, \mu_j)$$

进而得到外部指标:

$$DBI = \frac{1}{k} \sum_{i}^{k} \max_{j \neq i} (\frac{avg(C_i) + avg(C_j)}{d_{cen}(\mu_i, \mu_j)})$$
$$DI = \min_{1 \le i \le k} \{ \min_{j \neq i} (\frac{d_{min}(C_i, C_j)}{\max_{1 \le l \le k} diam(C_l)}) \}$$

DBI的分子表征簇内的平均距离，分母表征簇间的中心距离，显然越小越哈
DI对于每一个i求出与其他簇的最小距离，分母是所有簇的簇内最大距离，越大越好.

3. 距离的计算

在空间中定义距离，需要满足非负性，自身距离为0，对称性，三角不等式.
最常用的:
`Minkowski`距离

$$dist_{mk}(x_i, x_j) = \big(\sum_{i}^{n} |x_{iu} - x_{ju}|^p \big)^{\frac{1}{p}}$$

取$p = 2$即得到欧式距离
取$p = 1$即得到曼哈顿距离

$$dist_{man}(x_i, x_j) = ||x_i - x_j||_1 = \sum_{u=1}^{n} |x_{iu} - x_{ju}|$$

`Minkowski`距离适合于计算有序属性的距离(连续属性可以定义距离，离散属性要有序，比如(高，中，低))

对于无序属性可以采用VDM距离

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

这个值越小证明a, b一个属性的两个取值越接近.
考虑同时具有有序属性和无序属性的情况，可以将Minkowski与VDM结合(n_c个有序，$n - n_c$个无序)

$$\mathtt{MinkovoskiDM}_p(x_i, x_j) = \big(\sum_{u}^{n_c} |x_{iu} - x_{ju}|^p + \sum_{u=n_c+1}^{n} \mathrm{VDM}_p(x_{iu}, x_{ju}) \big)^{\frac{1}{p}}$$

当属性之间的重要性不同时, 可以再求和号里面加上权值.
通常我们基于相似度来定义距离比如VDM，叫做相似度度量, 这种度量方法不一定要满足所有的距离性质，如果不满足三角不等式，我们称之为非度性度量
如果算式不确定则可以通过"距离度量学习"的算法来确定算式

4. 原型聚类

假设聚类结构能通过一组原型刻画，我们先对聚类用原型结构初始化，然后对原型进行更新迭代，采用不同的原型表示，和不同的求解方式，可以产生不同的算法.

4.1 k-均值算法

优化目标:
将数据集分成k个簇$\mathcal{C} = \{C_l | l = 1, 2, 3 \cdots, k\}$

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

其中$\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$是簇$C_i$的均值向量, 反映了簇内样本的相似度.
算法描述:

> 输入样本数据集D, 和待划分的聚类组数k
>
> 过程:
>
> 1. 从D中随机选取k个样本记为$\mu_1, \cdots, \mu_k$
> 2. 对于每一个数据集中的样本找到最近的μ，归入对应的簇
> 3. 求出每一个簇的中心点，如果与对应的μ不相等则更新μ, 如果都相等则结束
> 4. 重复2，3直至μ的值无需更新，或者达到一定的轮数.

算法原理:
是一种贪心的思想每一次将样本划入簇中和更新均值都是使得优化函数最小化的过程.

4.2 学习向量量化

Learning Vector Quantization简称LVQ，假设样本带有标记$D = \{(x_1, y_1), (x_2, y_2), \cdots, (x_m, y_m)\}$, 目标是学习得到一组n维原型向量(n是每一个数据样本的属性个数)$\{p_1, p_2, \cdots, p_q\}$, 每一个原型向量代表一个聚类簇，每个簇的标记记为t_i
算法描述:

> 输入样本集$D = \{(x_1, y_1), (x_2, y_2), \cdots, (x_m, y_m)\}$
> 原型向量个数q，各原型向量的预设类标记$\{t_1, t_2, \cdots, t_q\}$
> 学习率$\eta \in (0, 1)$
> 过程:
>
> 1. 初始化一组原型向量$\{p_1, p_2, \cdots, p_q\}$
> 2. 从样本中随机选取样本(x_j, y_j), 计算选取的随机样本与每一个原型向量的距离，找出与x_j最相近的原型向量p_i^*
> 3. 如果$y_j = t_i^*$(x_j的标记与原型向量的类标记相同)
>   更新$p_i^* \leftarrow p_i^* + \eta(x_j - p_i^*)$使得原型向量靠近样本$x_j$

> 如果$y_j \neq t_i^*$(x_j的标记与原型向量的类标记不同)
> 更新$p_i^* \leftarrow p_i^* - \eta(x_j - p_i^*)$使得原型向量原离样本$x_j$
> 4. 如果更新很小或者达到最大迭代轮数就停止

为什么会靠近或者远离?
$p_i^* \leftarrow p^{'} = p_i^* + \eta(x_j - p_i^*)$计算如下

$$||x_j - p^{'}||_2 = ||p_i^* + \eta(x_j - p_i^*) - x_j||_2 = (1-\eta)||x_j - p_i^*||_2$$

$p_i^* \leftarrow p^{'} = p_i^* - \eta(x_j - p_i^*)$计算如下

$$||x_j - p^{'}||_2 = ||p_i^* - \eta(x_j - p_i^*) - x_j||_2 = (1+\eta)||x_j - p_i^*||_2$$

算法的结果就是每一个样本都距离自己的簇的原型向量最近，根据簇间原型向量间的距9离两原型向量距离相等的平面可以将空间剖分，划给每一个簇.

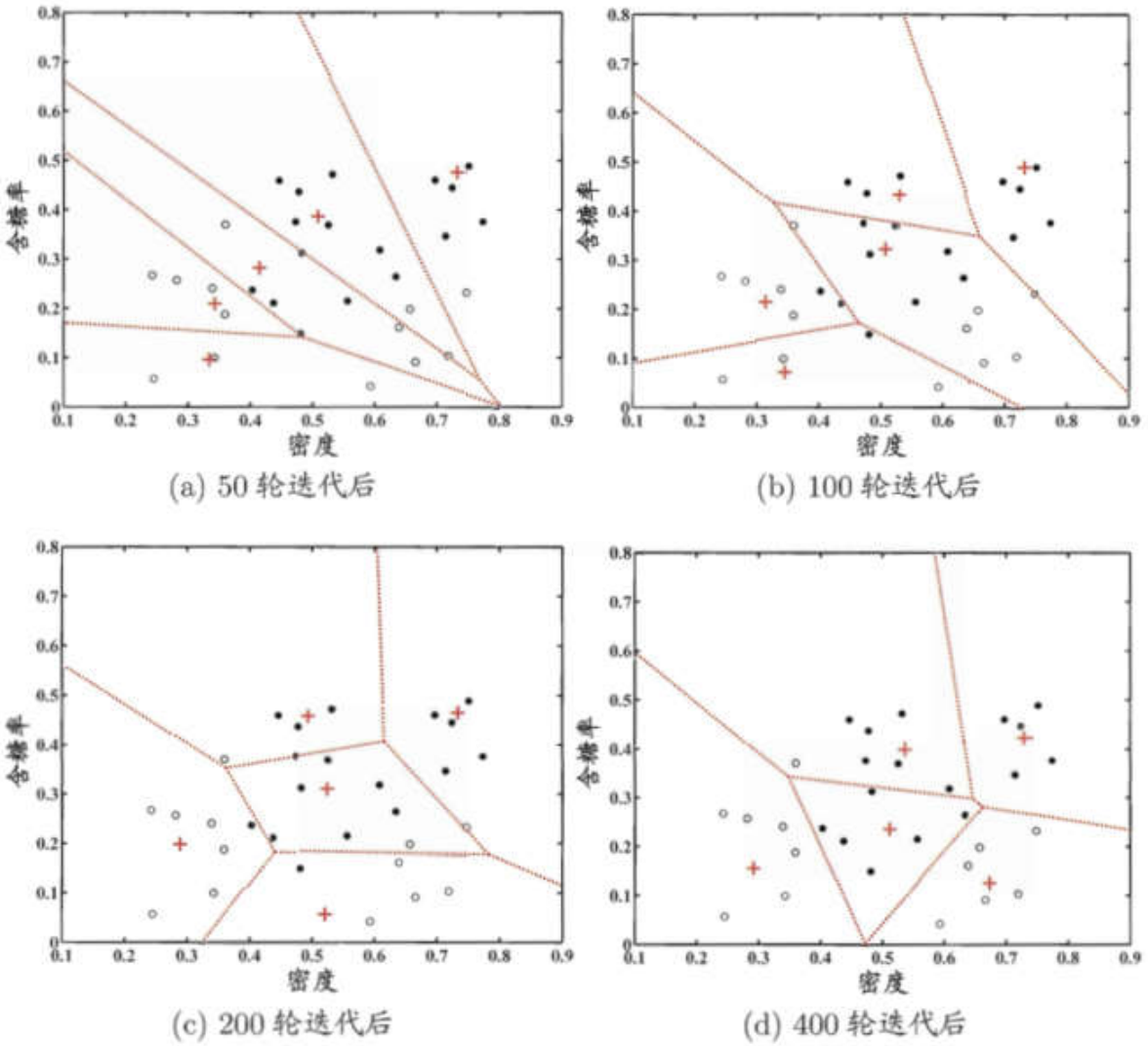


图 9.5 西瓜数据集 4.0 上 LVQ 算法($q = 5$)在不同轮数迭代后的聚类结果. c_1, c_2 类样本点与原型向量分别用"•", "○"与"+"表示, 红色虚线显示出聚类形成的 Voronoi 剖分.

4.3 高斯混合聚类

多元高斯函数的定义:
若x服从高斯分布，其概率密度函数为

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

μ,Σ分别是向量均值和协方差矩阵
高斯混合分布

$$p_M(x)=\sum_i^k\alpha_i\cdot p(x|\mu_i,\Sigma_i)$$

其中μ_i Σ_i是第i个高斯分布的参数，α_i是混合参数，$\sum_i^k\alpha_i=1$
假设样本的生成过程由高斯混合分布给出：首先根据k个α定义的先验概率选择高斯混合分布，其中α_i为选择第i个混合成分的概率；然后根据被选择的混合成分的概率密度函数进行采样生成对应的样本.
若训练集$D=\{x_1,x_2,\cdots,x_m\}$由上述过程生成，令随机变量$z_j\in\{1,2,\cdots,k\}$表示生成样本x_j的高斯混合成分，所以随机变量z_j的先验概率$P(z_j=i)$对应于α_i根据贝叶斯定理有下式成立:

$$\begin{aligned}P_M(z_j=i|x_j)&=\frac{P(z_j=i)\cdot P_M(x_j|z_j=i)}{P_M(x_j)}\\&=\frac{\alpha_i\cdot P(x_j|\mu,\Sigma_i)}{\sum_l^k\alpha_l\cdot P(x_j|\mu_l,\Sigma_l)}\end{aligned}$$

上式给出了样本x_j的是由第i个高斯分布给出的后验概率，可以将其简记为γ_{ji}
高斯聚类将样本集D划分为k个簇$\{C_1,C_2,\cdots,C_k\}$, 每个样本的簇标记为λ_j, 如下确定:

$$\lambda_j=\underset{i\in\{1,2,\cdots,k\}}{\arg\max}\gamma_{ji}\tag{*}$$

即要找到能使后验概率最大值最大的簇标记
如何求解模型参数$(\alpha_i,\mu_i,\Sigma_i)$?
最大化对数似然估计:

$$LL(D)=\ln\Big(\prod_j^m P_M(x_j)\Big)=\sum_j^m\ln\Big(\sum_i^k\alpha_i\cdot P(x_j|\mu_i,\Sigma_i)\Big)$$

求得最容易使得样本出现的参数

由$\frac{\partial LL(D)}{\partial\mu_i}=0$有

$$\sum_j^m\gamma_{ji}(x_j-\mu_i)=0$$

$$\mu_i=\frac{\sum_j^m\gamma_{ji}x_j}{\sum_j^m\gamma_{ji}}\tag{1}$$

直观地讲，混合成分的均值可以通过样本的加权评估来决定，样本权重就是后验概率
类似地，$\frac{\partial LL(D)}{\partial\Sigma_i}=0$
可以得到

$$\Sigma_i = \frac{\sum_j^m \gamma_{ji}(x_j - \mu_i)(x_j - \mu_i)^T}{\sum_j^m \gamma_{ji}} \tag{2}$$

对于α_i, 采用Lagrange乘数法，考虑约束条件$\sum_{i=1}^k \alpha_i = 1$

$$LL(D) + \lambda(\sum \alpha_i - 1)$$

对α_i求导

$$\sum_j^m \gamma_{ij} + \alpha_i \lambda = 0$$

对i从1到m累加, 得到$\lambda = -m$,

$$\alpha_i = \frac{1}{m} \sum_j^m \gamma_{ji} \tag{3}$$

由以上推导得到高斯混合模型的EM算法:

> E步: 计算每一个成分的后验概率γ_{ji}
> M步: 更新模型参数$\{(\alpha_i, \Sigma_i, \mu_i)\}$

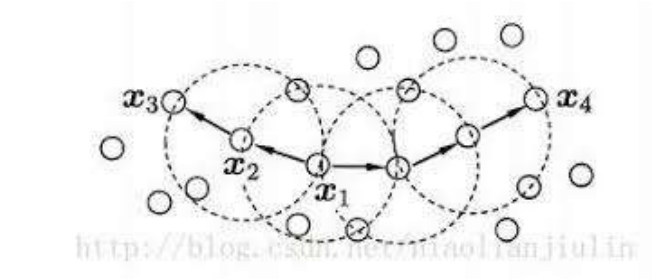
算法描述:

> 1. 初始化高斯混合分布的模型参数$\{(\alpha_i, \mu_i, \Sigma_i)\}$
> 2. 对每一个样本x_j计算其后验概率$\gamma_{ji}(1 \leq i \leq k)$
> 3. 按照上面得到的(1)(2)(3)三个式子来更新$\{(\alpha_i, \mu_i, \Sigma_i)\}$
> 4. 判断是否满足停止条件(达到最大迭代轮数或者似然函数不再增大),若是转5，若否转2继续
> 5. 根据*式，得到每一个样本的簇标记

5. 密度聚类

该算法基于数据的紧密程度来分割数据
DBSCAN算法是一种典型的密度聚类算法, 他基于以下概念:

1. ϵ邻域
   x_j的ϵ邻域：$N(x_j) = \{x_i \in D | dist(x_i, x_j) \leq \epsilon\}$
2. 核心对象
   若x_i的ϵ邻域内有MinPts以上的样本，即$|N_\epsilon(x_i)| \geq MinPts$, 则称$x_i$为一个核心对象
3. 密度直达
   若x_j位于x_i的ϵ邻域内，则称x_j由x_i密度直达
4. 密度可达
   假如存在样本序列"$p_1, p_2, \cdots, p_n$", p_{i+1}由p_i密度直达，则称p_n由p_1密度可达
5. 密度相连
   如果x_i, x_j均由x_k密度可达，则称前两者密度相连



算法目标:
给定邻域参数$(\epsilon, MinPts)$, 簇是数据集的子集，满足下面条件

1. 连接性: $x_i \in C, x_j \in C \Rightarrow x_i, x_j$密度相连
2. 最大性: $x_i \in C, x_j$由x_i密度可达 $\Rightarrow x_j \in C$

综合上述两条，可以定义簇为: 若x为一核心对象，则簇$X = \{x^{'} \in D | x^{'}$由$x$密度可达$\}$易于验证满足符合上面两个条件

算法描述:

> 输入: 样本集D，参数$(\epsilon, MisPts)$
>
> 1. 首先遍历数据集找出所有核心对象
> 2. 初始化聚类簇序号$k = 0$，初始化未访问对象集$\Gamma = D$
> 3. 随机选取一个核心对象，标记访问，然后按照层序寻找邻域点，如果该点是核心，加入队列继续访问，直至队列为空，在Γ中去掉本轮访问过的对象, 这些访问过的对象构成聚类簇
> 4. $k \leftarrow k + 1$, 在核心对象集中去掉访问过的，如果不为空转3继续，否则结束

PS：会存在一些数据点不会被归为任何一个簇，而且也不是核心对象，这样的数据我们认为是噪声.

6. 层次聚类

层次聚类希望在不同层次上对数据集进行划分，形成树形的聚类结构.
AGNES是一种有底向上的聚合策略层次聚类算法，他的思想是首先将每个样本看成一个初始簇，在算法运行的每一步对距离最近的两个簇进行合并. 直到达到预设的簇的个数结束.
簇间距离的定义:

1. 最小距离: $d_{min}(C_i, C_j) = \min\limits_{x \in C_i, z \in C_j} dist(x, z)$
2. 最大距离: $d_{max}(C_i, C_j) = \max\limits_{x \in C_i, z \in C_j} dist(x, z)$
3. 平均距离: $d_{avg}(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{x \in C_i} \sum_{z \in C_j} dist(x, z)$

算法描述:

> 输入: 样本集D，距离度量函数，聚类预期簇数
> 算法步骤:
>
> 1. 将每一个样本视为一簇，生成簇间距离矩阵，初始化簇数为m(样本数)
> 2. 找出最小的一个距离，合并两个簇，改变其后面的序号，更新距离矩阵
> 3. 判断是否达到预期的簇数，达到就停止结束，否则转2继续进行
> 4. 记录每一次合并，生成簇划分集(可以试试并查集来表示?)