第八讲 聚类简介

2025年4月

概要

- 与监督学习相比，无监督学习基于数据集$D = \{x_i\}_{i=1}^N$.
- 对每个数据特征向量x_i来说，不再有相应的标记信息y_i;
- 无监督学习的任务
  - 不再是学得$h_D : \mathcal{X} \to \mathcal{Y}$,
  - 而是发现数据集D（空间）所蕴含的知识模式.
- 不能通过对比训练样本x_i的标记y_i和学得的模型给出的预测$\hat{y}_i$来对学习算法作出评估.
- 无监督学习面临更多挑战.
- 聚类(Clustering)作为无监督学习的典型学习任务之一.

- 聚类目标在于发现数据内在的分布结构.
- 给定数据集$D = \{x_i\}_{i=1}^N$，对D中数据进行聚类就是将这些样本划分成若干个子集（通常互不相交），使得
  - 属于同一子集中的样本尽可能相互相似,
  - 不同子集中的样本尽可能不同.
- 称划分的每个样本子集为“簇”(Cluster).
- 认为每个簇对应于一些潜在的概念.
- 聚类算法的评价通常带有一定的主观性，而且与学习场景相关.
- 三类比较有代表性的聚类算法
  - 基于原型的聚类方法
  - 层次聚类方法
  - 基于密度的聚类方法

概要

- 基于原型的方法通常假设数据内在的分布结构可以通过一组原型来刻画.
  - 这类方法通常先对原型进行初始化，然后按照相应的策略和准则对原型进行迭代更新.
  - 原型表示和原型更新机制是基于原型的聚类算法的核心.
  - 比如K-means 聚类算法系列
- 另一方面，从数据生成机制来看，也可以认为不同簇的数据来自不同的分布，因而各个簇对应的概率分布模型也可以作为其原型.
  - 比如基于高斯混合模型的聚类方法.

K-means 聚类算法

给定样本集$D = \{x_i\}_{i=1}^N$，其中$x_i \in \mathbf{R}^n$，$i = 1, 2, \ldots, N$.

- 我们设聚类算法将样本集D划分为K个非空簇$C = \{C_l\}_{l=1}^K$使得$\cup_{l=1}^K C_l = D$且对任意$l \neq m$，有$C_l \cap C_m = \emptyset$.
- 和分类学习场景类似，如果$x_i \in C_{\lambda_i}$，我们就以λ_i作为x_i的簇标记，并将聚类的结果表示为簇标记向量

$$\lambda = (\lambda_1, \lambda_2, \cdots, \lambda_N)^T.$$

- K-means聚类算法以每个簇中样本的均值向量作为该簇的原型表示，即对簇C_l来说，以均值向量

$$\mu_l = \frac{1}{|C_l|} \sum_{i=1}^{N} x_i I(x_i \in C_l)$$

作为簇心.

K-means 聚类算法

- K-means聚类算法以平方误差和

$$E_l = \sum_{\mathbf{x} \in C_l} \| x - \mu_l \|_2^2$$

来度量该簇内样本围绕簇心的紧密程度.

- 注意到

$$2E_l = 2 \sum_{\mathbf{x} \in C_l} \| x - \mu_l \|_2^2 = \frac{1}{|C_l|} \sum_{\mathbf{x}_i, \mathbf{x}_j \in C_l} \| x_i - x_j \|_2^2,$$

则E_l也刻画了簇C_l内样本的平均彼此相近程度.

- K-means聚类算法采用贪心策略，通过迭代优化来近似求解最小化问题

$$\min_C \sum_{l=1}^{K} E_l.$$

K-means聚类算法

输入： 样本集$D=\{x_i\}_{i=1}^N$；聚类簇个数K

输出： 样本簇划分$\mathcal{C}=\{C_l\}_{l=1}^K$

(1) 选择K个样本点作为初始簇心$\mu_1,\mu_2,\cdots,\mu_K$；

(2) 令$C_l=\emptyset\ (1\le l\le K)$

(3) 对每个x_i，求x_i的簇标记 $\lambda_i=\underset{1\le j\le K}{\operatorname{argmin}}\parallel x_i-\mu_j\parallel_2$，
将x_i划分到C_{λ_i}中：$C_{\lambda_i}=C_{\lambda_i}\cup\{x_i\}$.

(4) 对每个$C_l(1\le l\le K)$，更新其簇心为

$$\mu_l=\frac{1}{|C_l|}\sum_{i=1}^{N}x_i I(x_i\in C_l)$$

(5) 重复(2)-(4)，直到划分不再变化；

(6) 返回簇划分$\mathcal{C}=\{C_l\}_{l=1}^K$.

对*K*-means聚类算法来说

- 最终划分结果可能依赖于随机选择的初始簇心.
- 一般基于不同的初始簇心多次运行*K*-means聚类算法.
- *K*-means聚类算法以簇内样本的均值向量作为簇心，以平方误差函数作为簇紧密度的度量方式，这对非凸形状的簇的发现来说并不一定合适.
- 均值运算对噪声和离群点（远离大多数样本的点）比较敏感.

K-means聚类算法的变体

- K-中心点方法通过挑选簇内相对处于最中心位置的一个实际样本点而非样本均值向量来作为簇心.
- 其他每个样本点每轮都被划分到当前簇心与其最相似的簇中.
- 用o_l表示簇C_l的簇心样本点，用$dist(x_i, o_l)$表示样本点x_i和o_l的相异程度度量，则K-中心点方法相当于通过最小化绝对误差

$$E = \sum_{l=1}^{K} \sum_{x \in C_l} dist(x, o_l)$$

来完成聚类划分.

- 围绕中心点的划分算法(Partitioning Around Medoids，简称PAM)是一种典型的K-中心点聚类算法.
  - 首先对每个簇的中心点进行随机初始化，并将非中心点的样本划分到簇心与其最相似的簇中，形成样本集的初始划分.
  - 然后采用贪心策略，迭代更新划分，直到没有变化为止.
    - 对当前的一个中心点o_l，随机选择一个非中心点样本x_i，评估以x_i 替代o_l作为簇心能否得到更好的划分.
    - 如果这种替代能得到更好的划分，则以x_i作为簇C_l的新中心点，然后对当前的非中心点样本进行重新划分;
    - 尝试这样所有可能的替换，直到簇划分不再发生变化为止.
- PAM算法使用中心点作为簇的原型表示，可以避免均值向量作为原型时易受离群点影响的问题.

- 高斯混合聚类算法假定不同的簇的样本是依据不同的概率分布(高斯分布)生成的.
- 以各个簇所对应的高斯分布作为划分簇的一组原型刻画.
- 高斯混合模型是指具有如下概率分布密度函数的分布模型:

$$p_{GM}(x|\theta) = \sum_{i=1}^{k} \alpha_i p(x|\mu_i, \Sigma_i),$$

这里$\theta = \{(\alpha_i, \mu_i, \Sigma_i)\}_{i=1}^{k}$为模型参数，其中

- $\alpha_i \geq 0$为混合系数，$\sum_{i=1}^{k} \alpha_k = 1$;
- $p(x|\mu_i, \Sigma_i)$为均值向量为μ_i，协方差矩阵为Σ_i的n元高斯分布密度函数，即

$$p(x|\mu_i, \Sigma_i) = \frac{1}{\sqrt{(2\pi)^n |\Sigma_i|}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mu_i)^T \Sigma_i^{-1} (\mathbf{x} - \mu_i)\right),$$

我们称其为第i个分模型.

- 特别地，如果$n = 1$，则分模型为高斯分布密度函数

$$p(x|\mu_i, \sigma_i^2) = \frac{1}{\sqrt{2\pi}\sigma_i} \exp(-\frac{(x-\mu_i)^2}{2\sigma_i^2}),$$

其中μ_i和σ_i^2分别为该分布的期望和方差.

- 给定样本集$D = \{x_j\}_{j=1}^N$,基于高斯混合模型的聚类算法假定样本x_j依据高斯混合分布生成,即先以概率α_i选择第i个分模型,然后依据被选择的分模型对应的高斯分布$p(x|\mu_i, \Sigma_i)$生成相应样本x_j.
- 不妨设与第i个分模型对应的簇为C_i，$i = 1, 2, \cdots, k$.
- 对每个x_j，定义一个随机变量z_j来表示生成x_j的分模型，即$z_j = i$表示x_j依据第i个分模型生成，故有

$$P(z_j = i) = \alpha_i, i = 1, 2, \cdots, k.$$

- 基于高斯混合模型的聚类算法采用后验概率最大化原则，将x_j划分到后验概率$P(z_j = i|x_j)$最大的簇中，即

$$\lambda_j = \underset{i\in[1,k]}{\operatorname{argmax}} P(z_j = i|x_j)$$

- 那么后验概率如何计算呢？由贝叶斯公式，

$$P(z_j = i|x_j) = \frac{\alpha_i p(x_j|\mu_i, \Sigma_i)}{\sum_{l=1}^{k} \alpha_l p(x_j|\mu_l, \Sigma_l)}.$$

相应的划分规则也可以写成

$$\lambda_j = \underset{i\in[1,k]}{\operatorname{argmax}} \alpha_i p(x_j|\mu_i, \Sigma_i).$$

给定样本集$D = \{x_j\}_{j=1}^N$，我们考虑对数似然函数

$$LL(\theta|D) = \log(\prod_{j=1}^{N} p_{GM}(x_j|\theta))$$

$$= \sum_{j=1}^{N} \log\left(p_{GM}(x_j|\theta)\right) = \sum_{j=1}^{N} \log\left(\sum_{i=1}^{k} \alpha_i p(x_j|\mu_i, \Sigma_i)\right).$$

对这类问题，直接求解比较困难.

- 回顾关于数据分两步生成的假定:
  - 第一步相当于对分模型进行抽样，但这些数据是不可观测的，
  - 观测到的是第二步依据第一步采样得到的分模型抽样的样本.
- 定义如下随机变量z_{ji}来刻画样本x_j是依据第i个分模型采样，即没有观测到的第一步的采样结果:

$$z_{ji} = \begin{cases} 1, & x_j \text{来自第 } i \text{ 个分模型} \\ 0, & \text{否则} \end{cases}$$

我们称这样的不可观测的随机变量为隐变量.

- 引入隐变量z_{ji}，可以将第二步得到的样本x_j扩展得到完全数据$(x_j, z_{j1}, \cdots, z_{jk})$.
- 以Z表示隐变量的数据，则完全数据的对数似然函数为

$$LL(\theta|D,Z) = \log\left(\prod_{j=1}^{N}\prod_{i=1}^{k}\left[\alpha_i p(x_j|\mu_i,\Sigma_i)\right]^{z_{ji}}\right)$$

$$= \log\left(\prod_{i=1}^{k}\left(\alpha_i^{(\sum_{j=1}^{N} z_{ji})}\right)\prod_{j=1}^{N}\left[p(x_j|\mu_i,\Sigma_i)\right]^{z_{ji}}\right)$$

$$= \sum_{i=1}^{k}\left\{(\sum_{j=1}^{N} z_{ji})\log\alpha_i + \sum_{j=1}^{N} z_{ji}\log p(x_j|\mu_i,\Sigma_i)\right\}.$$

- 常采用EM算法迭代求解这类含隐数据的对数似然函数极大化问题.

EM算法每次迭代包含两步：

- E步，求期望：基于当前参数θ的估计值$\theta^{(t)}$，计算对数似然函数关于隐变量Z的期望：

$$\begin{aligned} Q(\theta|\theta^{(t)}) &= E_Z[LL(\theta|D,Z)|D,\theta^{(t)}] \\ &= \sum_Z LL(\theta|D,Z)P(Z|D,\theta^{(t)}) \end{aligned}$$

- M步，极大化：通过极大化期望确定参数估计的更新$\theta^{(t+1)}$：

$$\theta^{(t+1)} = \underset{\theta}{\operatorname{argmax}}\, Q(\theta|\theta^{(t)}).$$

给定参数θ的初始值$\theta^{(0)}$，EM算法重复E步和M步，直到收敛到局部最优解。

用EM算法估计高斯混合模型混合系数和各分模型的参数：

$$LL(\theta|D,Z)=\sum_{i=1}^{k}\left\{(\sum_{j=1}^{N}z_{ji})\log\alpha_i+\sum_{j=1}^{N}z_{ji}\log p(x_j|\mu_i,\sigma_i^2)\right\},$$

令$n_i=\sum_{j=1}^{N}z_{ji}$，则

$$LL(\theta|D,Z)=\sum_{i=1}^{k}\left\{n_i\log\alpha_i+\sum_{j=1}^{N}z_{ji}\log p(x_j|\mu_i,\sigma_i^2)\right\}$$

$$=\sum_{i=1}^{k}\left\{n_i\log\alpha_i+\sum_{j=1}^{N}z_{ji}\left[\log\left(\frac{1}{\sqrt{2\pi}}\right)-\log\sigma_i-\frac{1}{2\sigma_i^2}(x_j-\mu_i)^2\right]\right\}.$$

- 为了计算给定$\theta^{(t)}$的条件下对数似然函数关于隐变量Z的期望，需要计算z_{ji}关于隐变量Z的期望$\gamma_{ji}^{(t)}$:

$$\gamma_{ji}^{(t)} = E_Z[z_{ji}|D,\theta^{(t)}] = P(z_{ji}=1|D,\theta^{(t)})$$

$$= P(z_j=i|D,\theta^{(t)}) = \frac{\alpha_i^{(t)} p(x_j|\mu_i^{(t)},\sigma_i^{2(t)})}{\sum_{l=1}^{k} \alpha_l^{(t)} p(x_j|\mu_l^{(t)},\sigma_l^{2(t)})}.$$

则对E步来说，$Q(\theta|\theta^{(t)}) = E_Z[LL(D,Z|\theta)|D,\theta^{(t)}]$

$$= \sum_{i=1}^{k}\left\{(\sum_{j=1}^{N}\gamma_{ji}^{(t)})\log\alpha_i + \right.$$

$$\left.\sum_{j=1}^{N}\gamma_{ji}^{(t)}\left[\log\left(\frac{1}{\sqrt{2\pi}}\right) - \log\sigma_i - \frac{1}{2\sigma_i^2}(x_j-\mu_i)^2\right]\right\}.$$

对M步来说，

$$\theta^{(t+1)} = \underset{\theta}{\operatorname{argmax}}\, Q(\theta|\theta^{(t)}).$$

- 令$\frac{\partial Q(\theta|\theta^{(t)})}{\partial \mu_i} = 0$得到

$$\sum_{j=1}^{N} \gamma_{ji}^{(t)} \left[\frac{1}{\sigma_i^2}(x_j - \mu_i)\right] = 0.$$

因此

$$\mu_i^{(t+1)} = \frac{\sum_{j=1}^{N} \gamma_{ji}^{(t)} x_j}{\sum_{j=1}^{N} \gamma_{ji}^{(t)}}, \quad i = 1, 2, \cdots, k.$$

- 令

$$\frac{\partial Q(\theta|\theta^{(t)})}{\partial \sigma_i^2} = 0$$

得到

$$(\sigma_i^2)^{(t+1)} = \frac{\sum_{j=1}^{N} \gamma_{ji}^{(t)} (x_j - \mu_i^{(t+1)})^2}{\sum_{j=1}^{N} \gamma_{ji}^{(t)}}, \quad i = 1, 2, \cdots, k.$$

- 对参数α_i来说，还要受到$\sum_{i=1}^{k} \alpha_i = 1, \alpha_i \geq 0$的约束. 我们采用拉格朗日对偶法，构造拉格朗日函数

$$L(\theta, \beta) = Q(\theta|\theta^{(t)}) + \beta(1 - \sum_{i=1}^{k} \alpha_i).$$

- 令

$$\frac{\partial L(\theta,\beta)}{\partial \alpha_i}=0$$

得

$$n_i^{(t)}=\beta\alpha_i.$$

- 两边对所有分模型求和得

$$N=\sum_{i=1}^{k}n_i^{(t)}=\beta\sum_{i=1}^{k}\alpha_i=\beta.$$

故

$$\alpha_i^{(t+1)}=\frac{n_i^{(t)}}{\beta}=\frac{\sum_{j=1}^{N}\gamma_{ji}^{(t)}}{N},\quad i=1,2,\cdots,k.$$

对于多元高斯混合模型来说，EM算法的参数更新为：

$$\mu_i^{(t+1)} = \frac{\sum_{j=1}^{N} \gamma_{ji}^{(t)} x_j}{\sum_{j=1}^{N} \gamma_{ji}^{(t)}},$$

$$\Sigma_i^{(t+1)} = \frac{\sum_{j=1}^{N} \gamma_{ji}^{(t)} (x_j - \mu_i^{(t+1)})(x_j - \mu_i^{(t+1)})^T}{\sum_{j=1}^{N} \gamma_{ji}^{(t)}},$$

$$\alpha_i^{(t+1)} = \frac{\sum_{j=1}^{N} \gamma_{ji}^{(t)}}{N}, \quad i = 1, 2, \cdots, k,$$

其中 $\gamma_{ji}^{(t)} = P(z_j = i | D, \theta^{(t)}) = \frac{\alpha_i^{(t)} p(x_j | \mu_i^{(t)}, \Sigma_i^{(t)})}{k}$.

基于高斯混合模型的聚类算法

输入： 样本集$D = \{x_i\}_{i=1}^N$；聚类簇个数k

输出： 样本簇划分$\mathcal{C} = \{C_l\}_{l=1}^k$

(1) 令$C_l = \emptyset\ (1 \leq l \leq k)$

(2) 初始化高斯混合模型的参数$\theta = \{(\alpha_i, \mu_i, \Sigma_i)\}_{i=1}^k$

(3) E步： 基于当前参数估计，计算后验概率

$$\gamma_{ji} = P(z_j = i|D, \theta)$$

$$i = 1, 2, \cdots, k; j = 1, 2, \cdots, N.$$

(4) M步： 基于当前参数估计，更新参数

$$\mu_i = \frac{\sum_{j=1}^{N} \gamma_{ji} x_j}{\sum_{j=1}^{N} \gamma_{ji}},$$

$$\Sigma_i = \frac{\sum_{j=1}^{N} \gamma_{ji} (x_j - \mu_i)(x_j - \mu_i)^T}{\sum_{j=1}^{N} \gamma_{ji}},$$

$$\alpha_i = \frac{\sum_{j=1}^{N} \gamma_{ji}}{N}, \quad i = 1, 2, \cdots, k.$$

(5) 重复(3)-(4)，直到收敛；

(6) 对每个x_j，求x_j的簇标记

$$\lambda_j = \underset{i\in[1,k]}{\operatorname{argmax}} \alpha_i p(x_j|\mu_i, \Sigma_i).$$

将x_j划分到C_{λ_j}中：

$$C_{\lambda_j} = C_{\lambda_j} \cup \{x_j\}.$$

(7) 返回簇划分$\mathcal{C} = \{C_l\}_{l=1}^k$.

概要

层次聚类算法

- 允许在聚类过程中对已有的簇进行合并或者分裂.
- 通过对样本集不同层次的划分形成树形的层次聚类结构.
- 可以将所有样本看作是一个初始簇,采用自顶向下的拆分策略.
- 也可以将每个样本看成一个初始微簇，采用自底向上的聚合策略.
- 通常层次聚类算法不一定需要提前设定簇的个数.
- 可以设置其他的终止条件来结束簇的分裂或者合并过程.

AGNES(AGglomerative NESting)算法

- 一种典型层次聚类算法.
- 采用自底向上的聚合策略进行簇聚合.
  - 每个样本构成的簇作为初始簇结构,
  - 重复执行将当前簇中距离最近的两个簇进行合并的操作,
  - 直到满足终止条件（比如当前簇的个数达到预定的数目）.

核心问题: 如何度量两个簇之间的距离呢?

- 簇C_i和C_j之间的距离$d(C_i, C_j)$可以通过分别属于两个簇的样本之间的距离来定义.

常见的簇之间的距离定义有:

- 最小距离: $d_{\min}(C_i, C_j) = \min_{x \in C_i, x' \in C_j} dist(x, x')$;
- 最大距离: $d_{\max}(C_i, C_j) = \max_{x \in C_i, x' \in C_j} dist(x, x')$;
- 平均距离: $d_{avg}(C_i, C_j) = \frac{1}{|C_i| \times |C_j|} \sum_{x \in C_i, x' \in C_j} dist(x, x')$;
- 质心距离: $d_{Centroid}(C_i, C_j) = dist(c_i, c_j)$, 其中 $c_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$, $c_j = \frac{1}{|C_j|} \sum_{x' \in C_j} x'$;
- 中心点距离: $d_{Medoid}(C_i, C_j) = dist(o_i, o_j)$, 其中 o_i 和 o_j 分别为簇 C_i 和 C_j 的中心点.

下图中给出了两个簇之间的最大距离、最小距离和质心距离的示意:

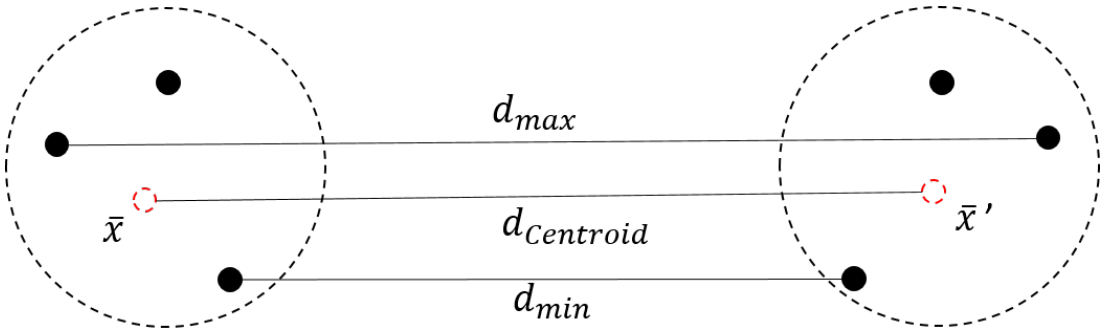

Figure: 簇的最小距离、最大距离和质心距离

- 如果一个聚类算法分别选用最小距离、最大距离、平均距离作为两个簇的距离，则相应的算法分别被称为单连接算法、全连接算法、均连接算法。

- AGNES算法通常采用单连接方式.
- AGNES算法采用距离(相异性)矩阵来保存当前簇之间的距离，初始距离矩阵为$N \times N$的对称矩阵M，且

$$M(i,j) = d(C_i, C_j), \quad i,j = 1,2,\cdots,N.$$

- 随着每次距离最近的两个簇的合并，对距离矩阵也作相应的修正:
  - 不妨设当前距离最近的两个聚类簇为C_{i^*}和C_{j^*}且$i^* < j^*$，则
    (1) 将C_{j^*}并入C_{i^*}，将合并以后的新簇仍然记作C_{i^*}，并将所有$j > j^*$簇C_j的下标减1，重新标记为C_{j-1};
    (2) 删除当前距离矩阵M的第j^*行与第j^*列;
    (3) 将$M(i^*,j)$和$M(j,i^*)$更新为$d(C_{i^*}, C_j)$.
- DIANA(Divisive Analysis)算法正好与AGNES算法的思想正好相反，采用从顶向下的分裂方法进行层次聚类.

- 用树状图来描述AGNES算法不同层次的簇聚合过程.

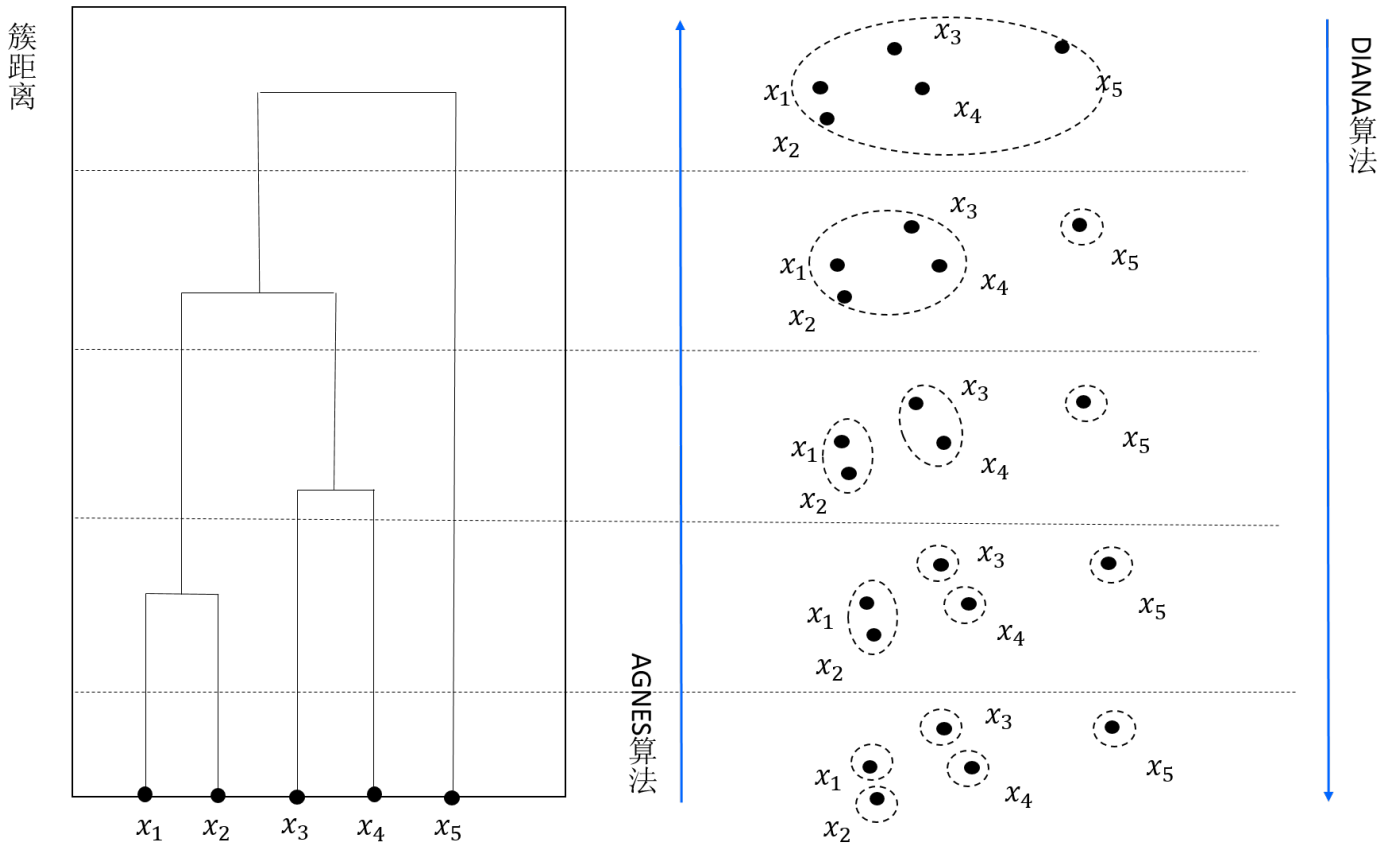

Figure: 簇聚合过程的树状图

概要

基于密度的聚类方法

- 将簇看作是数据空间中被稀疏区域分开的稠密区域.
- 聚类就是发现并不断扩展稠密区域的过程.
- 直观上我们可以从一个样本点的邻域入手，按照某种标准检查这个小区域是否是稠密的.
- 如果是稠密的，则我们看能否将其邻域中的点的邻域包含进来对已经找到的稠密区域进行扩展.
- DBSCAN（Density-based Spatial Clustering of Applications with Noise)算法是一个典型的基于密度的聚类算法.

DBSCAN算法

- 通过找出邻域稠密的样本点，并建立这类样本点之间的连接性来扩展稠密区域进而获得最终的聚类簇.
- DBSCAN算法引进邻域半径$\epsilon > 0$和密度阈值$MinPts > 0$两个参数来刻画邻域稠密的样本点:
  - 给定样本集$D = \{x_i\}_{i=1}^N$，对于样本点$x_i \in D$，定义其ϵ- 邻域

$$N_\epsilon(x_i) = \{x_j \in D | dist(x_i, x_j) \leq \epsilon\}.$$

  - 如果x_i的ϵ- 邻域$N_\epsilon(x_i)$至少包含$MinPts$个样本点，则我们称x_i为核心对象.

如何由这些核心对象和它们的邻域形成稠密的区域？引进样本之间相连接的概念.

- 直观上，一个核心对象和它的ϵ- 邻域中的样本点形成一个小的稠密区域
  - 如果x_j 位于核心对象x_i 的ϵ- 邻域中，则我们称x_j 由x_i 直接密度可达(directly density-reachable).
- 进一步，对x_j和x_i 来说，如果存在一个样本序列 $p_1, p_2, \cdots, p_n$ 使得 $p_1 = x_i, p_n = x_j$， 并且对任何 $1 \leq i \leq n-1$， p_{i+1} 由p_i 直接密度可达，则称x_j由x_i密度可达(density-reachable).
- 密度可达刻画的是对核心对象的ϵ- 邻域中的核心样本可重复使用直接密度可达来带入更多的样本点到稠密区域.
- x_j和x_i是密度相连的(density-connected)，如果存在一个样本$p \in D$使得样本x_j和x_i都是由p密度可达的.

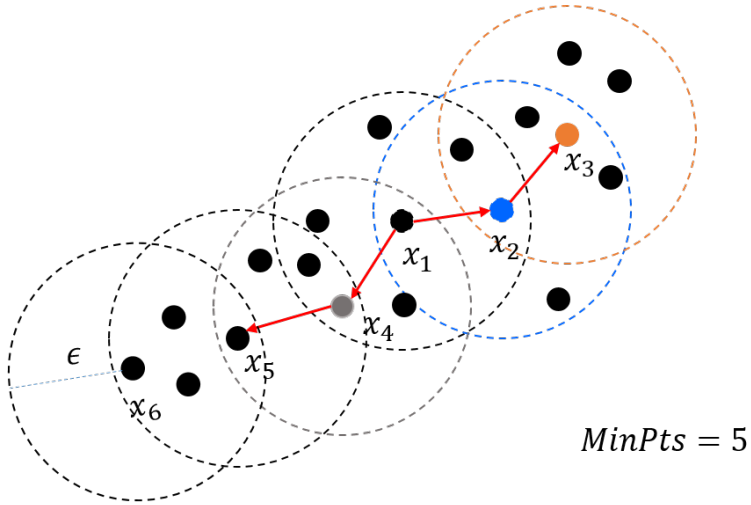

Figure: 密度可达与相连

- 除了x_6，其它各点都是核心对象.
- x_2和x_4都在x_1的ϵ- 邻域中，因此x_2和x_4由x_1 直接密度可达.
- x_3和x_5 由x_1密度可达.
- x_3和x_5是密度相连的.

簇

簇为满足如下条件的样本子集$C \subseteq D$:

(1) 如果$x_i, x_j \in C$， 则x_i与x_j是密度相连的;

(2) 对任一$x_i \in C$，如果 x_j由x_i是密度可达，则$x_j \in C$.

- 簇就是密度相连的关于密度可达的最大样本子集.
- 由一个核心对象出发，由其密度可达的所有样本点的集合正好构成一个簇.

DBSCAN算法

- 先将所有的样本点标记为未访问.
- 然后每次随机选择一个未访问的样本进行访问:
  - 如果该样本是核心对象，则找出由该样本点密度可达的所有样本点生成簇;
  - 如果不是核心对象，则将其标记为噪声.
- 重复这样的过程，直到没有未被访问的样本点.

如果参数ϵ和*MinPts*设置比较恰当，DBSCAN算法可以有效地发现任意形状的簇.

小结

- 聚类分析
- 基于原型的聚类方法
  - *K*-means 聚类算法系列
  - 高斯混合模型
- 层次聚类方法
  - AGNES算法
- 基于密度的聚类方法
  - DBSCAN算法