

第6章

数据聚类

Data Cluster

赫 然

rhe@nlpr.ia.ac.cn

<http://rhe-web.github.io/>

智能感知与计算研究中心 (CRIPAC)

中科院自动化研究所 模式识别国家重点实验室

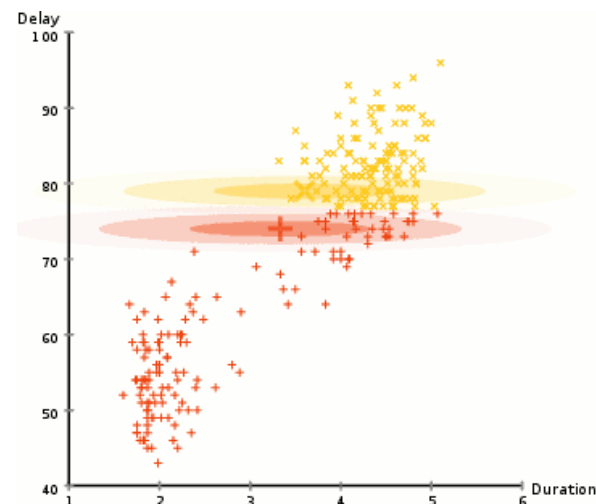
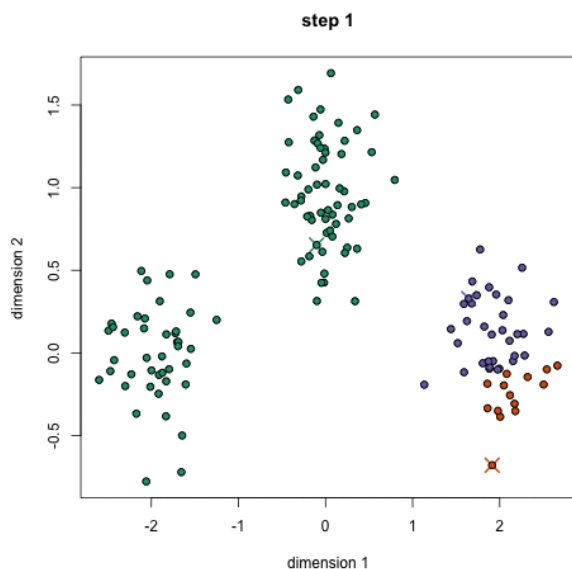
内容提要

- 聚类与无监督学习
- 聚类评价指标
- 常用的聚类算法
 - 原型聚类
 - 密度聚类
 - 层次聚类
- 谱聚类
- 子空间上的聚类
- 集成聚类

聚类

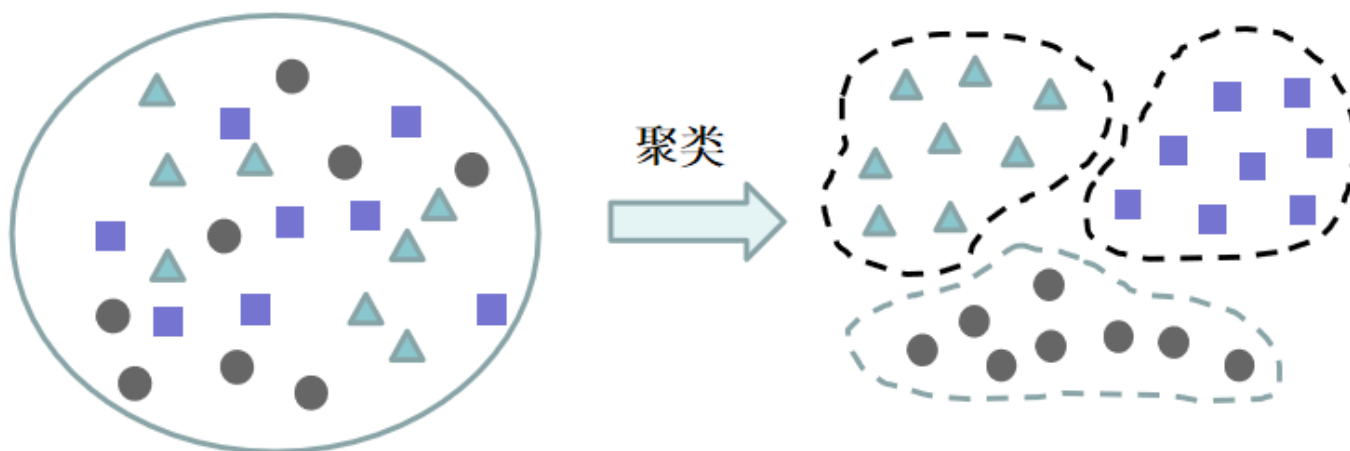
定义

- 对一批没有类别标签的样本集，按照样本之间的相似程度分类，相似的归为一类，不相似的归为其它类，每个类称为**簇 (cluster)**。这种分类称为**聚类分析**，也称为**无监督分类**。
- 聚类的质量(或结果)取决于对**度量标准**的选择。
- 聚类结果**因不同任务而不同**。



聚类

目标：将数据集中的样本划分为若干个通常不相交的子集



簇内相似度 (intra-cluster similarity) , 高
簇间相似度 (inter-cluster similarity) , 低

引言

聚类任务

- **模型法(原型聚类)**：为每一个簇引入一个模型，然后对数据进行划分，使其满足各自分派的模型，如 **K-Means**。
- **层次法**：对给定样本进行层次划分，如**层级聚类**。
- **密度法**：估计数据的密度函数，如**混合高斯模型**、**Mean-Shift**方法。
- **网格法**：将数据空间划分为有限个单元网络结构，然后基于网络结构进行聚类，如**矢量量化**。

内容提要

- 聚类与无监督学习
- **聚类评价指标**
- 常用的聚类算法
 - 原型聚类
 - 密度聚类
 - 层次聚类
- 谱聚类
- 子空间上的聚类
- 集成聚类

性能评估

□ 聚类性能评估：

- 外部指标 (external index)

有聚类结果参考标准。

- 内部指标 (internal index)

无聚类结果参考标准。

性能评估

外部指标

- 假定存在某个参考聚类结果，通过与它比较计算得到外部指标
- 数据集 $X = \{x_1, x_2, \dots, x_m\}$
- 通过聚类给出的簇划分 $D = \{d_1, d_2, \dots, d_k\}$
- 参考聚类结果为 $G = \{g_1, g_2, \dots, g_s\}$



有如下定义

- a : 在 D 中和 G 中都属于相同聚类的样本对个数
- b : 在 D 中属于相同聚类、但在 G 中属于不同聚类的样本对个数
- c : 在 D 中属于不同聚类、但在 G 中属于相同聚类的样本对个数
- d : 在 D 中和 G 中都属于不同聚类的样本对个数

$$a + b + c + d = m(m-1) /$$

性能评估

外部指标

- Jaccard系数 (Jaccard Coefficient)


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

- FM指数 (Fowlkes and Mallows Index)

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

- Rand指数 (Rand Index)

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



区间属于[0,1],
越大越好

性能评估

通过聚类给出的簇划分 $D = \{d_1, d_2, \dots, d_k\}$

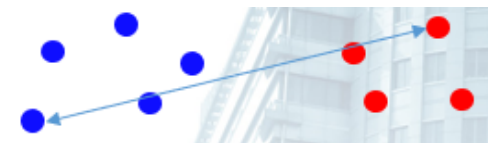
内部指标

在无参考聚类结果下，根据聚类簇样本之间的相互距离定义一些内部评价指标。

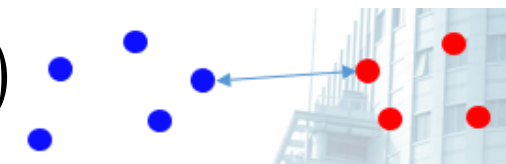
- **簇内平均距离:** $avg(D) = \frac{1}{|D|(|D|-1)} \sum_{x_i, x_j \in D, x_i \neq x_j} dist(x_i, x_j)$



- **簇内最远距离:** $diam(D) = \max_{x_i, x_j \in D, x_i \neq x_j} dist(x_i, x_j)$



- **簇间最近距离:** $d_{\min}(D_i, D_j) = \min_{x_i \in D_i, x_j \in D_j} dist(x_i, x_j)$



- **簇内中心距离:** $d_{cen}(D_i, D_j) = dist(u_i, u_j), u_i = \frac{1}{|D_i|} \sum_{x_j \in D_i} x_j$



性能评估

内部指标

- DB指数 (Davies-Bouldin Index, DBI)

$$DBI = \frac{1}{k} \sum_{i=1}^k \max_{j \neq i} \left(\frac{avg(D_i) + avg(D_j)}{d_{cen}(u_i, u_j)} \right)$$

性能评估

内部指标

- DB指数 (Davies-Bouldin Index, DBI)

$$DBI = \frac{1}{k} \sum_{i=1}^k \max_{j \neq i} \left(\frac{\text{avg}(D_i) + \text{avg}(D_j)}{d_{cen}(u_i, u_j)} \right)$$

簇内平均距离

簇间中心距离

- 如果每个簇样本距离均值越小（即簇内样本距离都很近），则DBI越小；如果簇间中心点的距离越大（即簇间样本距离相互都很远），则DBI越小。

性能评估

内部指标

- DB指数 (Davies-Bouldin Index, DBI)

$$DBI = \frac{1}{k} \sum_{i=1}^k \max_{j \neq i} \left(\frac{\overbrace{avg(D_i) + avg(D_j)}^{\text{簇内平均距离}}}{\underbrace{d_{cen}(u_i, u_j)}_{\text{簇间中心距离}}} \right)$$

- Dunn指数 (Dunn Index, DI)

$$DI = \min_{1 \leq i \leq k} \left\{ \min_{j \neq i} \left(\frac{\overbrace{d_{\min}(D_i, D_j)}^{\text{簇间最近距离}}}{\underbrace{\max_{1 \leq l \leq k} diam(D_l)}_{\text{簇内最远距离}}} \right) \right\}$$

- 如果任意两个簇之间最近的距离的最小值越大（即簇间样本距离相互都很远），则DI越大；如果任意一个簇内距离最远的两个点的距离的最大值越小（即簇内样本距离都很近），则DI越大

性能评估

距离度量

- 距离度量的性质

非负性: $\text{dist}(x_i, x_j) \geq 0$

同一性: $\text{dist}(x_i, x_j) = 0$, 当且仅当 $x_i = x_j$

对称性: $\text{dist}(x_i, x_j) = \text{dist}(x_j, x_i)$

直递性: $\text{dist}(x_i, x_j) \leq \text{dist}(x_i, x_k) + \text{dist}(x_k, x_j)$

性能评估

距离度量

- 常用距离:

闵可夫斯基距离 (Minkowski distance)

$$dist(x_i, x_j) = \left(\sum_{u=1}^n |x_{iu} - x_{ju}|^p \right)^{\frac{1}{p}}$$

$p=2$, 欧式距离 (Euclidean distance)

$p=1$, 曼哈顿距离 (Manhattan distance)

性能评估

距离度量

- 属性介绍

- 连续属性 (continuous attribute)

在定义域上有无穷多个可能的取值

- 离散属性 (categorical attribute)

在定义域上是有限个可能的取值

- 有序属性 (ordinal attribute)

如定义域为 $\{1, 2, 3\}$ 的离散属性, “1” 与 “2” 比较接近、与 “3” 比较远, 称为 “有序属性”

- 无序属性 (non-ordinal attribute)

定义域为 {飞机, 火车, 轮船} 这样的离散属性, 不能直接在属性值上进行计算, 称为 “无序属性”。

性能评估

距离度量

- 属性介绍

- Value Difference Metric, VDM (处理无序属性)

令 $m_{u,a}$ 表示属性 u 上取值为 a 的样本数, $m_{u,a,i}$ 表示在第 i 个样本簇中在属性 u 上取值为 a 的样本数, k 为样本数, 则属性 u 上两个离散值 a 与 b 之间的VDM距离为:

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

性能评估

距离度量

- 属性介绍

- MinkovDMP（处理混合属性）：

$$MinkovDM_p(x_i, x_j) = \left(\sum_{u=1}^{n_c} |x_{iu} - x_{ju}|^p + \sum_{u=n_c+1}^n VDM_p(x_{iu}, x_{ju}) \right)^{\frac{1}{p}}$$

- 加权距离（样本中不同属性的重要性不同时）：

$$dist(x_i, x_j) = \left(w_1 |x_{i1} - x_{j1}|^p + \cdots + w_n |x_{in} - x_{jn}|^p \right)^{\frac{1}{p}}$$

$$w_i \geq 0, \sum_{i=1}^n w_i = 1$$

内容提要

- 聚类与无监督学习
- 聚类评价指标
- 常用的聚类算法
 - 原型聚类
 - 密度聚类
 - 层次聚类
- 谱聚类
- 子空间上的聚类
- 集成聚类

常用聚类算法

- 原型聚类

- 也称为“基于原型的聚类” (prototype-based clustering)，此类算法假设聚类结果能通过一组**原型**（样本空间中具有代表性的点）刻画。

- 算法过程：

- 初始化原型
 - 对原型进行迭代

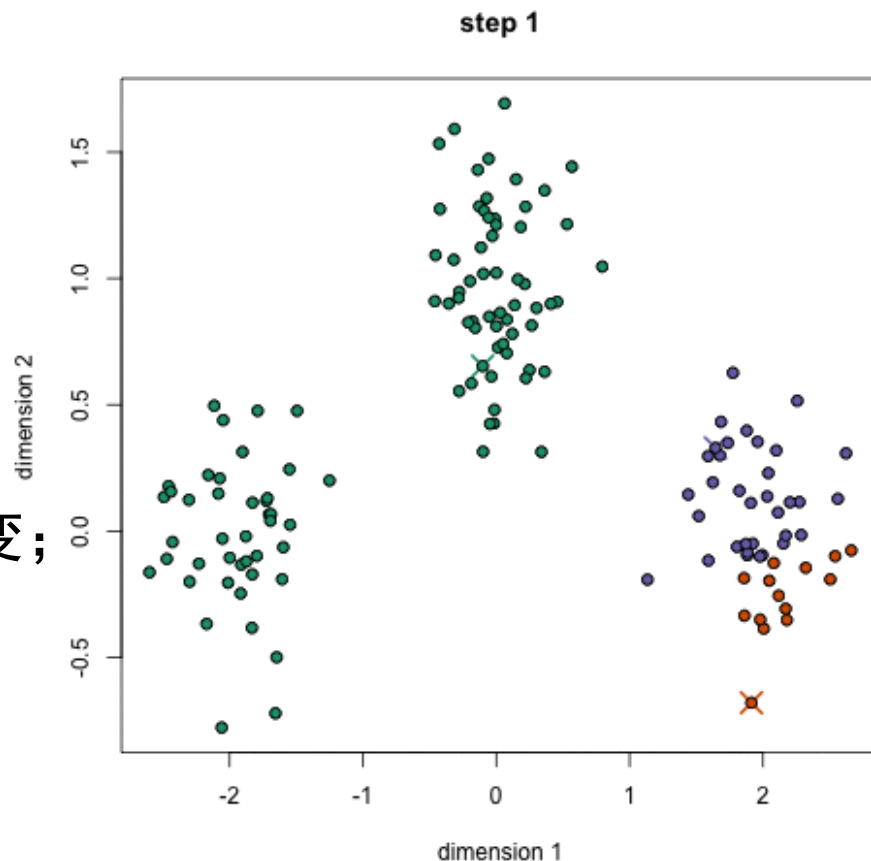
- 典型算法：

- K均值算法
 - 学习向量量化算法
 - 高斯混合聚类算法

常用聚类算法

● K均值算法

- 1 有 n 个样本， c 个聚类中心；
- 2 将 n 个样本按照一定原则分配给离它最近的聚类中心；
- 3 重新计算聚类中心；
- 4 重复2和3，直到聚类中心不改变；
- 5 返回簇 $\mu_1, \mu_2, \dots, \mu_c$ ；



常用聚类算法

- 基本思想

- K-均值聚类是**误差平方和**最小准则下的聚类方法

- 设 n_i 表示属于 D_i 类样本的个数，算法中的 μ_i 是这些样本的均值：

$$\mu_i = \frac{1}{n} \sum_{x \in D_i} x$$

- “误差平方和” 聚类准则：

- $$J_e = \sum_{i=1}^c J_i \quad J_i = \sum_{x \in D_i} \|x - \mu_i\|^2$$

- 对于不同的划分（聚类），会得到不同的 μ_i 。因此 J_e 的值也是不同的。使 J_e **最小** 的聚类就是**误差平方和准则**下的最优结果。因此，称这类聚类方法为**最小方差划分法**。

K-均值聚类

假设样本 \hat{x} 从类 D_i 移动到 D_j ，此时，两个类中心将时进行变化：

$$\mu_j^* = \mu_j + \frac{\hat{x} - \mu_j}{n_j + 1} \qquad \mu_i^* = \mu_i - \frac{\hat{x} - \mu_i}{n_i - 1}$$

此时，属于第 j 类的样本点引起的**误差平方和将增加**为：

$$J_j^* = \sum_{x \in D_j} \|x - \mu_j^*\|^2 + \|\hat{x} - \mu_j\|^2 = J_j + \frac{n_j \|\hat{x} - \mu_j\|^2}{n_j + 1}$$

属于第 i 类的样本点引起的**误差平方和将减少**为：

$$J_i^* = J_i - \frac{n_i \|\hat{x} - \mu_i\|^2}{n_i - 1}$$

K-均值聚类 (K-means Clustering)

如果**减少量大于增加量**，因此鼓励这种移动，即将样本 \hat{x} 从类 D_i 移动到 D_j 会减少总体误差：

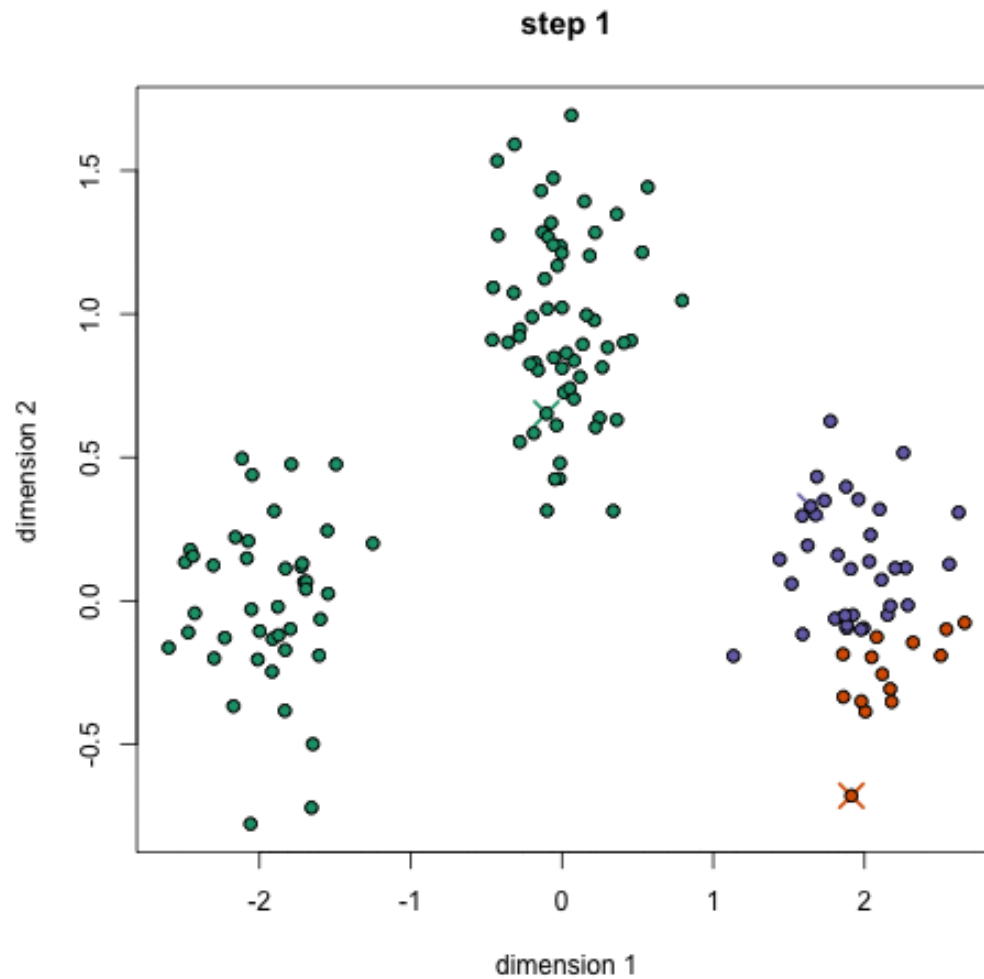
$$\frac{n_j ||\hat{x} - \mu_j||^2}{n_j + 1} < \frac{n_i ||\hat{x} - \mu_i||^2}{n_i - 1}$$



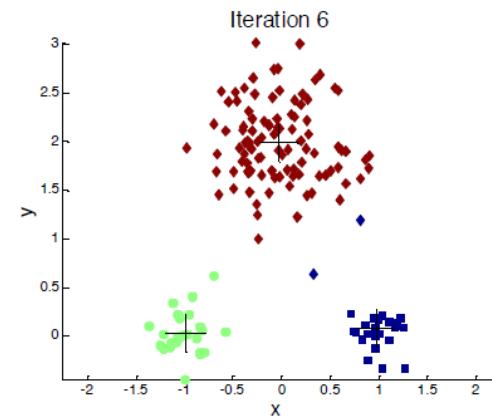
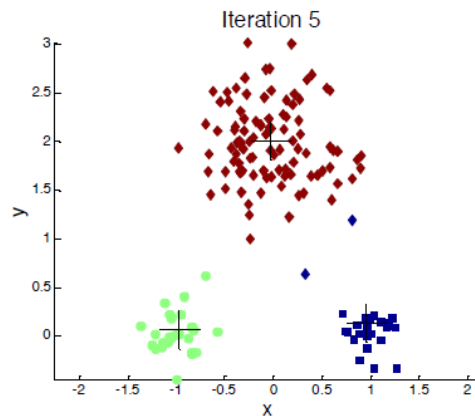
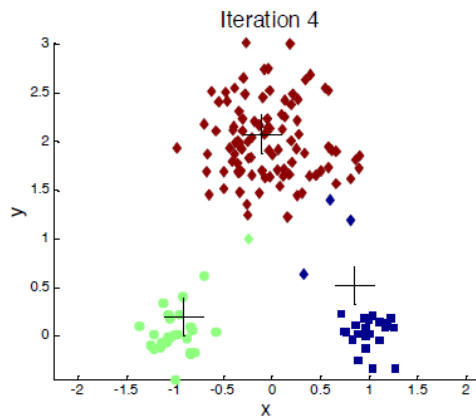
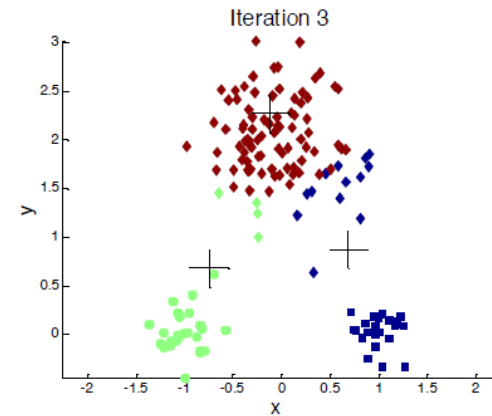
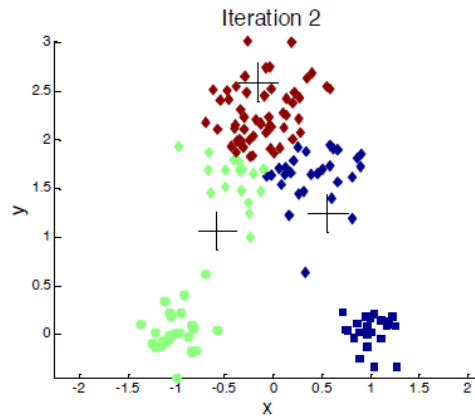
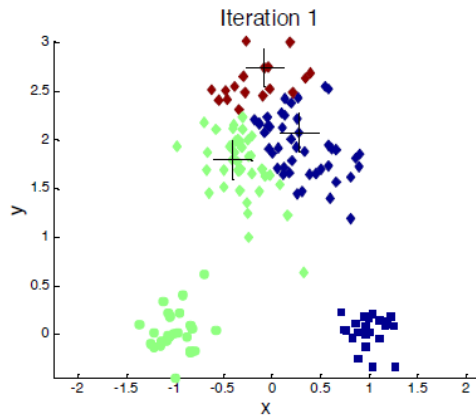
结论

从一个类引出样本会减少该类均方误差；但移入样本至一个类会增加该类均方误差。如果**减少量大于增加量**，对这样的样本进行移动是**有利于总体误差减少的**，**算法中3的准则**。

K-均值聚类 (K-means Clustering)



K-均值聚类 (K-means Clustering)



K-均值聚类 (K-means Clustering)

综上所述，基于**最小方差划分法**的k-means算法步骤如下：

基于**最小方差划分法**的k-means算法步骤

- 1 有 n 个样本, c 个聚类中心以及聚类中心 $\mu_1, \mu_2, \dots, \mu_c$;
- 2 随机挑选一个样本 \hat{x} , 找出它最佳的聚类中心下标, 即

$$i \leftarrow \arg \min_i \|\mu_i - \hat{x}\|$$

- 3 如果 n_i 不等于0, 则计算:

$$\rho_j = \begin{cases} \frac{n_j}{n_j + 1} \|\hat{x} - \mu_j\|, j \neq i \\ \frac{n_j}{n_j - 1} \|\hat{x} - \mu_j\|, j = i \end{cases}$$

- 4 在所有的 ρ_j 中找到最小的值记为 ρ_k ;
- 5 对于所有的 j 而言, 有 $\rho_k < \rho_i$, 那么将 \hat{x} 移动到 D_k 中;
- 6 重新计算 J_e, μ_i, μ_k ;
- 7 重复2-6, 直到对于 n 个样本, J_e 的值不再改变;
- 8 返回聚类中心 $\mu_1, \mu_2, \dots, \mu_c$ 。

常用聚类算法

● K-均值聚类优缺点

优点

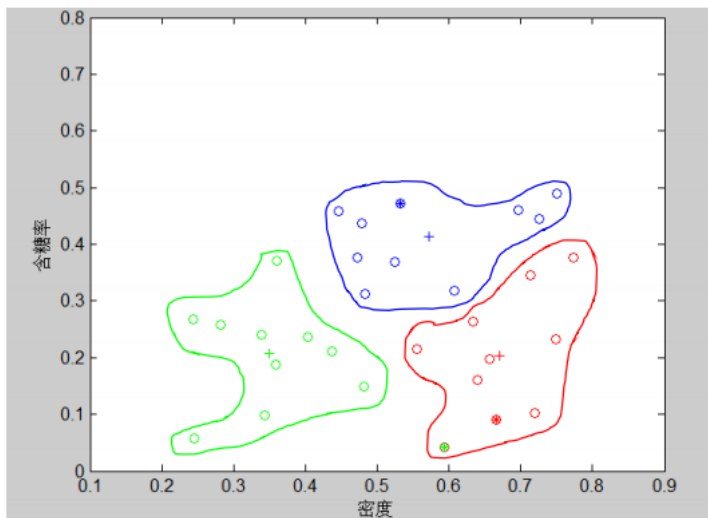
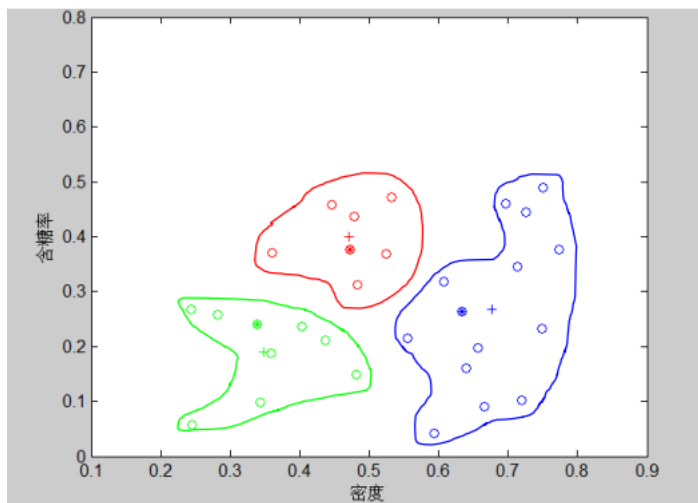
- 是解决聚类问题的一种经典算法，简单、快速
- 对处理大数据集，该算法仍可保持其高效率
- 对于密集簇，聚类效果很好

缺点

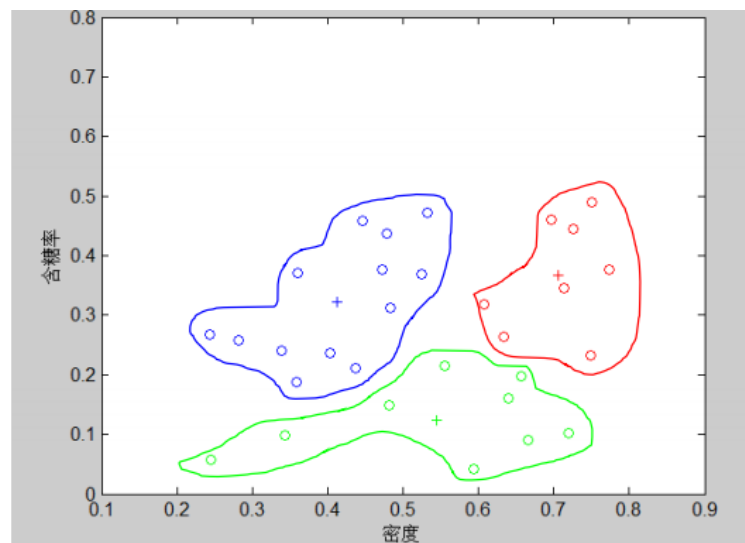
- 必须事先给定簇的个数，且对初始值敏感
- 不适合于发现非凸曲面的簇以及大小相差很大的簇
- 对噪声、孤立数据点、野点很敏感

常用聚类算法

- K-均值聚类优缺点



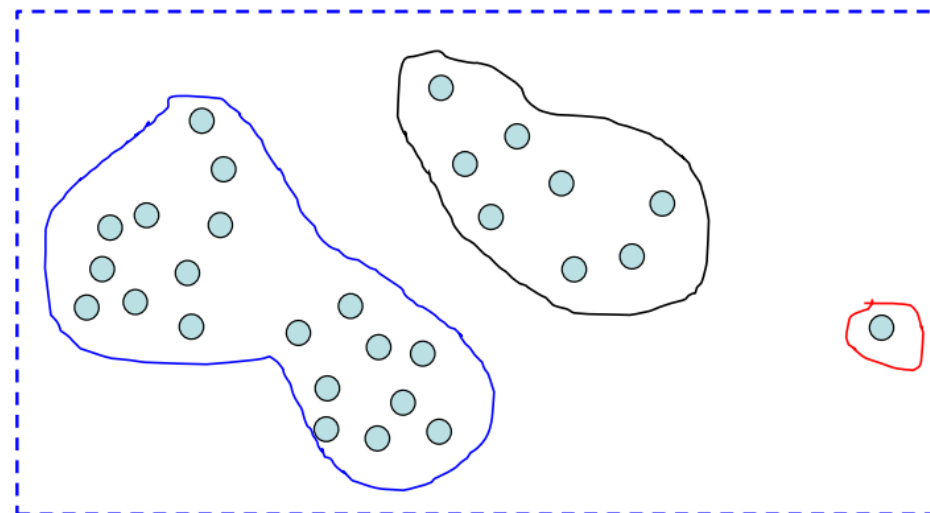
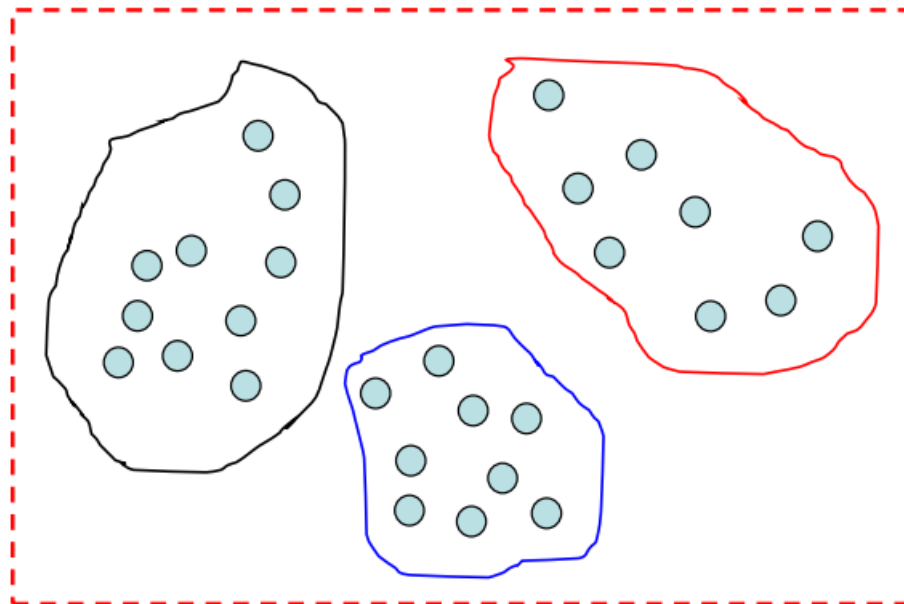
不同初始值可能会得到不同的聚类结果



常用聚类算法

- K-均值聚类优缺点

孤立数据点对聚类结果影响很大



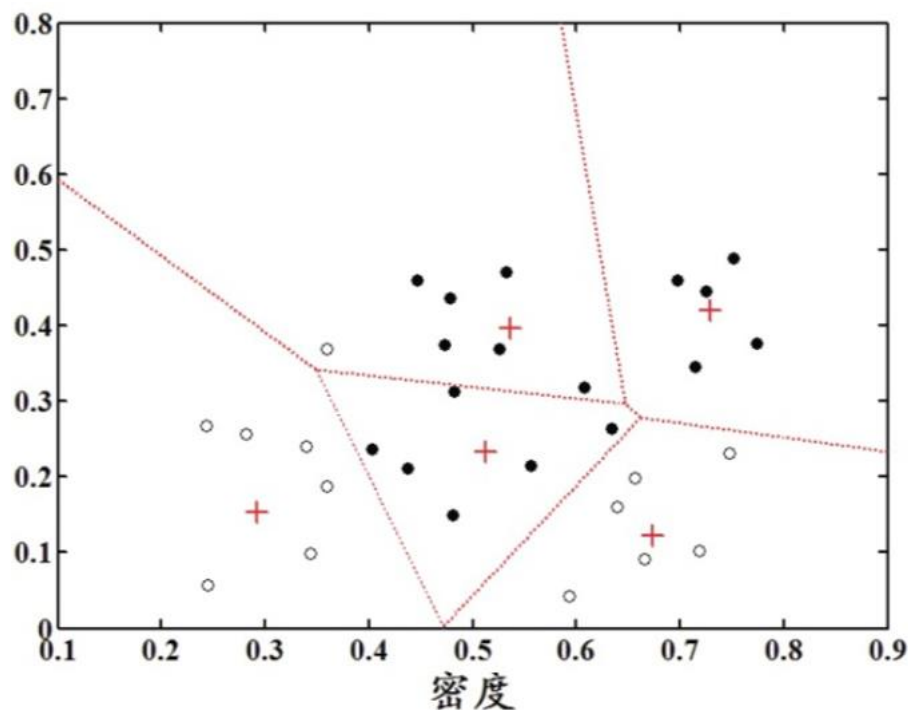
学习向量量化 (Learning Vector Quantization)

- 基本思想

- 与一般聚类算法不同的是，LVQ假设数据样本带有类别标记，学习过程中利用样本的这些**监督信息**来辅助聚类。

- 给定样本集合 $D = \{(x_1, z_1), (x_2, z_2), \dots, (x_m, z_m)\}$

- **目标：**学得一组 n 维原型向量 $\{y_1, y_2, \dots, y_q\}$ ，每个原型向量代表一个**聚类簇**。



学习向量量化 (Learning Vector Quantization)

- 基本思想

- 与一般聚类算法不同的是，LVQ假设数据样本带有类别标记，学习过程中利用样本的这些**监督信息**来辅助聚类。

- 给定样本集合 $D = \{(x_1, z_1), (x_2, z_2), \dots, (x_m, z_m)\}$

- **目标：**学得一组 n 维原型向量 $\{y_1, y_2, \dots, y_q\}$ ，每个原型向量代表一个**聚类簇**。

- 关键步骤：对于每个样本点 x_i 找到最为接近的代表点 y_k ，

- 若两个点类别相同，则将 y_k 拉向 x_i

$$y'_k = y_k + \alpha(x_i - y_k)$$

- 若两个点类别不同，则将 y_k 与 x_i 拉远

$$0 < \alpha < 1$$

$$y'_k = y_k - \alpha(x_i - y_k)$$

学习向量量化 (Learning Vector Quantization)

- 基本思想

- 若两个点类别相同, 则将 y_k 拉向 x_i

$$y'_k = y_k + \alpha(x_i - y_k)$$

- 若两个点类别不同, 则将 y_k 与 x_i 拉远 $0 < \alpha < 1$

$$y'_k = y_k - \alpha(x_i - y_k)$$

分析:

$$\|y'_k - x_i\|_2 = \|y_k + \alpha(x_i - y_k) - x_i\|_2 = (1 - \alpha)\|y_k - x_i\|_2$$

$$(1 - \alpha)\|y_k - x_i\|_2 < \|y_k - x_i\|_2$$

所以拉近

学习向量量化

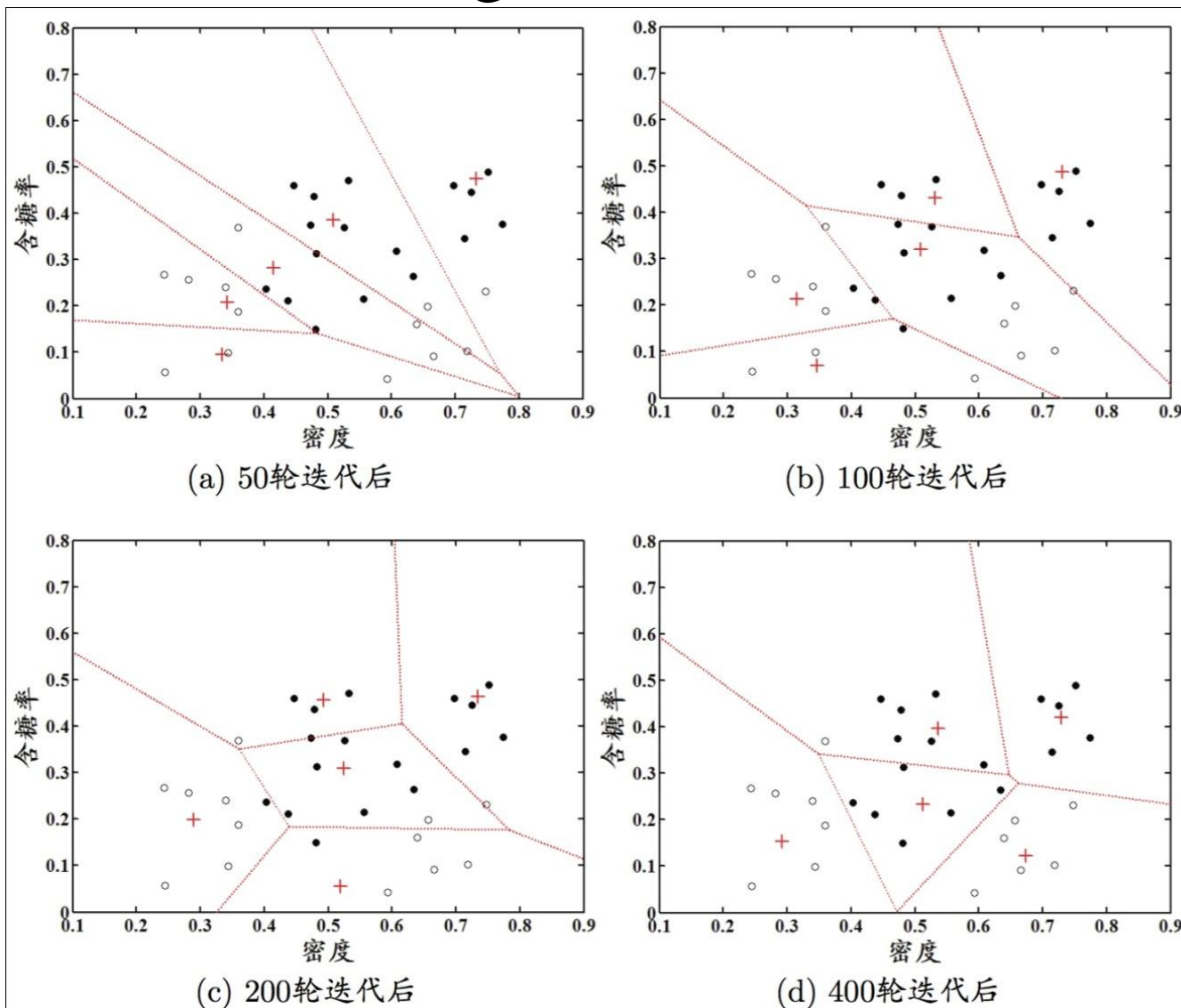
学习向量量化算法步骤

- 1 **Input:** 样本集合 $D = \{(x_1, z_1), (x_2, z_2), \dots, (x_m, z_m)\}$ 原型向量个数 q , 各原型向量预设的类别标记 $\{t_1, t_2, \dots, t_q\}$ 学习率 $\alpha \in \{0 - 1\}$
- 2 初始化一原型向量 $\{y_1, y_2, \dots, y_q\}$
- 3 repeat
- 4 从样本集 D 中随机选取样本 (x_i, y_i)
- 5 计算样本 x_i 到 y_i 的距离 $d_{ii} = \|x_i - y_i\|_2$
- 6 找出与 x_i 最近的原型向量 $i^* = \operatorname{argmin} d_{ii}$
- 7 if $z_i = t_{i^*}$
- 8 $y_k = y_{i^*} + \alpha(x_i - y_{i^*})$
- 9 else
- 10 $y_k = y_{i^*} - \alpha(x_i - y_{i^*})$
- 11 end if
- 12 将原型向量 y_{i^*} 更新为 y_k
- 13 until 满足停止条件, 返回原型向量 $\{y_1, y_2, \dots, y_q\}$

学习向量量化 (Learning Vector Quantization)

● 基本思想

— 实验结果



EM算法 (Expectation Maximization)

- 基本思想

- 首先从另一个角度看K-means算法

K-means 算法的步骤:

- (1) 先随机选择初始节点, 然后计算每个样本所属类别
- (2) 然后通过类别再更新初始化节点

EM算法 (Expectation Maximization)

- 基本思想

- 首先从另一个角度看K-means算法

K-means 算法的步骤:

- (1) 先随机选择初始节点, 然后计算每个样本所属类别
- (2) 然后通过类别再更新初始化节点

- 收敛证明

损失函数为:

$$J_e = \sum_{i=1}^c \sum_{j=1}^n r_{ji} \|x_j - \mu_i\|^2 \quad r_{ji} = \begin{cases} 1, x_j \in k \\ 0, else \end{cases}$$

EM算法 (Expectation Maximization)

- 基本思想

- 收敛证明

损失函数为:
$$J_e = \sum_{i=1}^c \sum_{j=1}^n r_{ji} \|x_j - \mu_i\|^2$$

$$\frac{\partial J_e}{\partial \mu_k} = 2 \sum_{j=1}^n r_{jk} (x_j - \mu_k) = 0$$

$$\mu_k = \frac{\sum_{j=1}^n r_{jk} x_j}{\sum_{j=1}^n r_{jk}}$$

μ_k 是指第 k 个中心, 可以看出, 新的中心点就是所有该类的质心。

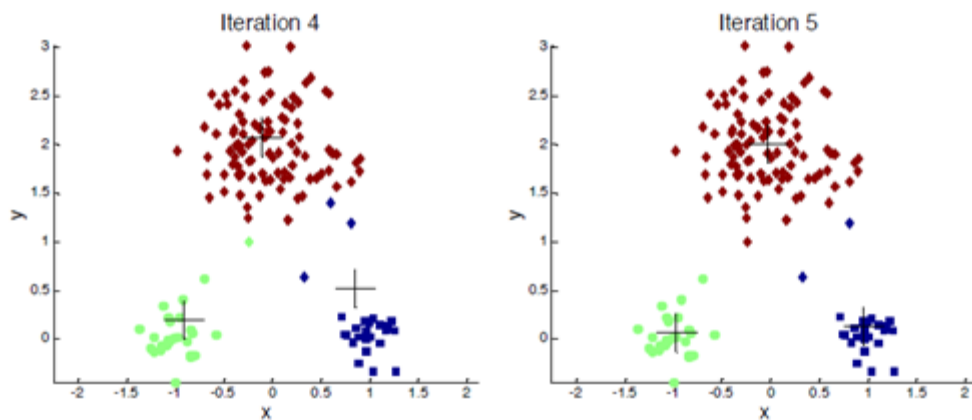
参考自: https://zhuanlan.zhihu.com/p/78798251?utm_source=qq

EM算法 (Expectation Maximization)

- 基本思想

- EM算法角度看K-means

在K-means中的**隐变量是每个类别所属类别**，K-means算法迭代步骤中的“**每次确认中心点以后重新进行标记**”对应 EM 算法中的 **E步** 即求当前参数条件下的 Expectation，根据“**标记重新求中心点**”对应 EM 算法中的 **M步** 即求似然函数最大化时（损失函数最小时）对应的参数）



EM算法 (Expectation Maximization)

- 基本思想

- 首先根据已经给出的观测数据，估计出模型参数的值（**EM中的E**）；
- 然后再依据上一步估计出的参数值**估计缺失数据的值**，再根据估计出的缺失数据加上之前已经观测到的数据**重新对参数值进行估计**（**EM中的M**），然后反复迭代，直至最后收敛，迭代结束。

高斯混合模型 (Mixture-of-Gaussian)

- 基本思想

- 与k均值、LVQ用原型向量来刻画聚类结构不同，高斯混合聚类采用**概率模型**来表达聚类原型：

- 给定对 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

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

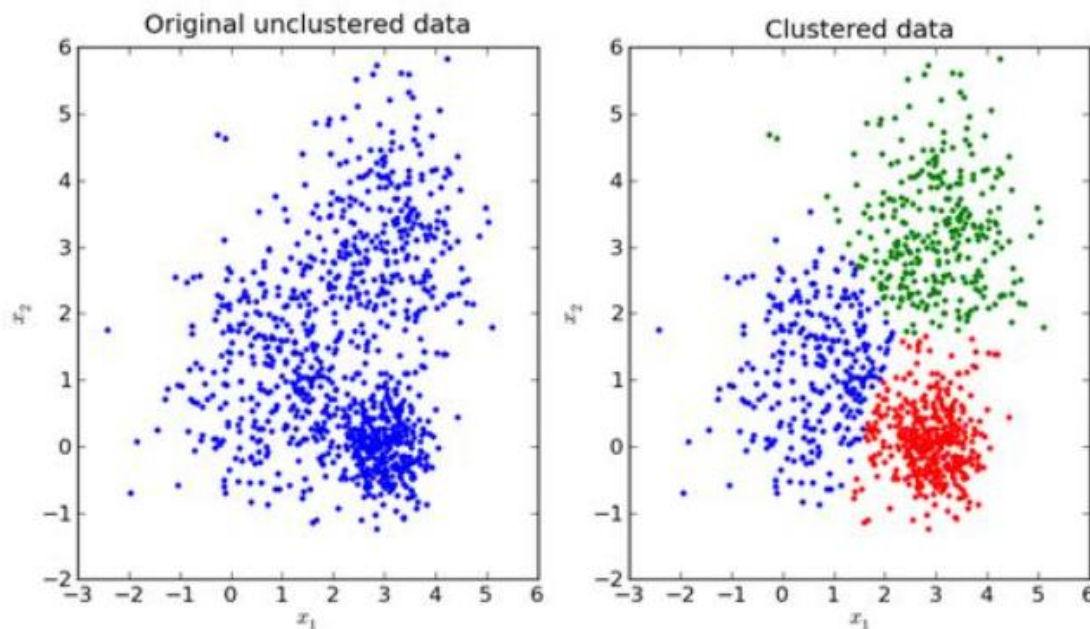
其中 μ 是 n 维均值向量， Σ 是 $n \times n$ 维的协方差矩阵。也可将概率密度函数记作 $p(x | \mu, \Sigma)$

高斯混合模型 (Mixture-of-Gaussian)

- 基本思想

— 给定对 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

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



图来自: <https://zhuanlan.zhihu.com/p/30483076>

高斯混合模型 (Mixture-of-Gaussian)

- 基本思想

- 高斯混合分布的定义

$$p_M(x) = \sum_{i=1}^k \alpha_i p(x_i | \mu_i, \Sigma_i)$$

该分布由 k 个混合分布组成，每个分布对应一个高斯分布。其中 μ_i 与 Σ_i 是第 i 个高斯混合成分的参数。而 $\alpha_i > 0$ 为相应的“混合系数”

$$\sum_{i=1}^k \alpha_i = 1$$

- 假设样本的生成过程由高斯混合分布给出

- 首先， $\alpha_1, \alpha_2, \dots, \alpha_k$ 定义的先验分布选择高斯混合成分，其中 α_i 表示选择第 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采样，从而生成相应的样本。

高斯混合模型 (Mixture-of-Gaussian)

- 基本思想

— 模型求解：最大化对数似然

无法直接求导

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

引入辅助变量 r_{ji} ：

$$\begin{aligned} LL(D) &= \sum_{j=1}^m \ln\left(\sum_{i=1}^k r_{ji} \frac{\alpha_i p(x_j | \mu_i, \Sigma_i)}{r_{ji}}\right) \\ &\geq \sum_{j=1}^m \sum_{i=1}^k r_{ji} \ln \frac{\alpha_i p(x_j | \mu_i, \Sigma_i)}{r_{ji}} \end{aligned}$$

$$r_{ji} = p_M(z_j = i | x_j) = \frac{\alpha_i \cdot p(x_j | \mu_i, \Sigma_i)}{\sum_{l=1}^k \alpha_l \cdot p(x_j | \mu_l, \Sigma_l)}, \quad z_j \text{ 为隐变量}$$

高斯混合模型 (Mixture-of-Gaussian)

- 基本思想

- 模型求解：最大化对数似然

$$\begin{aligned}l(\theta, \theta^t) &= \sum_{j=1}^m \sum_{i=1}^k r_{ji}^t \ln \frac{\alpha_i p(x_j | \mu_i, \Sigma_i)}{r_{ji}^t} \\&= \sum_{j=1}^m \sum_{i=1}^k r_{ji} (\ln \alpha_i - \ln r_{ji}^t - \ln \sqrt{2\pi \Sigma_i^2} - \frac{(x_j - \mu_i)^2}{2\Sigma_i^2}) \\l(\alpha_i, \lambda) &= \sum_{j=1}^m r_{ji}^t \ln \alpha_i + \lambda (\sum_{i=1}^k \alpha_i - 1)\end{aligned}$$

拉格朗日乘子法

高斯混合模型 (Mixture-of-Gaussian)

- 基本思想

- 模型求解：m样本数目、k混合模型个数

$$l(\alpha_i, \lambda) = \sum_{j=1}^m r_{ji}^t \ln \alpha_i + \lambda \left(\sum_{i=1}^k \alpha_i - 1 \right)$$

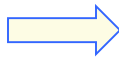
$$\sum_{i=1}^k \alpha_i = \sum_{i=1}^k \left(-\frac{\sum_{j=1}^m r_{ji}^t}{\lambda} \right) = 1$$



$$-\sum_{i=1}^k \left(\sum_{j=1}^m r_{ji}^t \right) = \lambda$$



$$\lambda = -m$$



$$\frac{\partial l(\alpha_i, \lambda)}{\partial \alpha_i} = \sum_{j=1}^m r_{ji}^t \frac{1}{\alpha_i} + \lambda = 0$$

$$\alpha_i = -\frac{\sum_{j=1}^m r_{ji}^t}{\lambda}$$

$$\therefore \lambda = -m \quad \alpha_i^{t+1} = \frac{\sum_{j=1}^m r_{ji}^t}{m}$$

高斯混合模型 (Mixture-of-Gaussian)

- 基本思想

- 模型求解：最大化对数似然

$$\frac{\partial l(\theta, \theta^t)}{\partial \mu_i} = 0$$

$$\Rightarrow \sum_{j=1}^m \frac{r_{ji}^t (x_j - \mu_i)}{\Sigma_i^2} = 0$$

$$\Rightarrow \sum_{j=1}^m r_{ji}^t \mu_i = \sum_{j=1}^m r_{ji}^t x_j \Rightarrow \mu_i \sum_{j=1}^m r_{ji}^t = \sum_{j=1}^m r_{ji}^t x_j$$

$$\Rightarrow \mu_i^{t+1} = \frac{\sum_{j=1}^m r_{ji}^t x_j}{\sum_{j=1}^m r_{ji}^t}$$

高斯混合模型 (Mixture-of-Gaussian)

- 基本思想

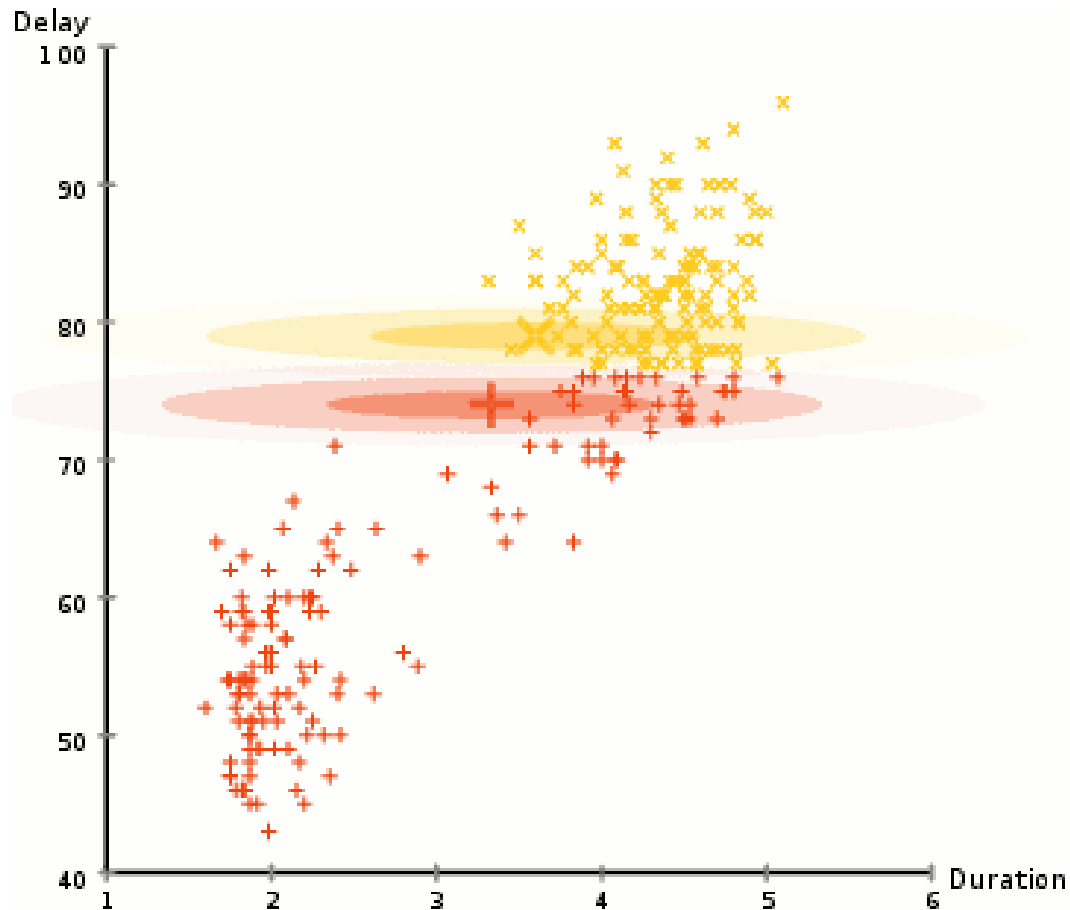
- 模型求解：最大化对数似然

$$\begin{aligned} & \frac{\partial l(\theta, \theta^t)}{\partial \Sigma_i} = 0 \\ \Rightarrow & \sum_{j=1}^m r_{ji}^t \left(-\frac{1}{\Sigma_i} + \frac{(x_j - \mu_i)^2}{\Sigma_i^3} \right) = 0 \\ \Rightarrow & \sum_{j=1}^m r_{ji}^t \Sigma_i^2 = \sum_{j=1}^m r_{ji}^t (x_j - \mu_i)^2 \\ \Rightarrow & \Sigma_i^{2^{t+1}} = \frac{\sum_{j=1}^m r_{ji}^t (x_j - \mu_i^{t+1})^2}{\sum_{j=1}^m r_{ji}^t} \end{aligned}$$

- 1 **Input**: 样本集合 $D = \{x_1, x_2, \dots, x_m\}$, 高斯混合成分个数 k
- 2 初始化高斯混合模型参数 $\{(\alpha_i, \mu_i, \Sigma_i) | 1 \leq i \leq k\}$
- 3 repeat
- 4 for $j = 1, 2, \dots, m$ do
- 5 计算 x_j 由各混合成分生成的**后验概率**, 即 $r_{ji} = P_M(z_j = i | x_j)$
- 6 end for
- 7 for $i = 1, 2, \dots, k$ do
- 8 计算新均值向量 $\mu_i = \frac{\sum_{j=1}^m r_{ji} x_j}{\sum_{j=1}^m r_{ji}}$
- 9 计算新协方差矩阵 $\Sigma_i = \frac{\sum_{j=1}^m r_{ji} (x_j - \mu_i)(x_j - \mu_i)^T}{\sum_{j=1}^m r_{ji}}$
- 10 计算新混合系数 $\alpha_i = \frac{\sum_{j=1}^m r_{ji}}{m}$
- 11 end for
- 12 将模型的参数由 $\{(\alpha_i, \mu_i, \Sigma_i) | 1 \leq i \leq k\}$ 更新到 $\{(\alpha'_i, \mu'_i, \Sigma'_i) | 1 \leq i \leq k\}$
- 13 until 满足条件
- 14 $C_i = \emptyset (1 \leq i \leq k)$
- 15 for $j = 1, 2, \dots, m$ do
- 16 确定 x_j 的簇标记 λ_j
- 17 将 x_j 划入相应的簇 $C_{\lambda_j} = C_{\lambda_j} \cup \{x_j\}$
- 18 end for
- 19 输出簇划分 $C = \{C_1, C_2, \dots, C_k\}$

高斯混合模型 (Mixture-of-Gaussian)

- 结果



内容提要

- 聚类与无监督学习
- 聚类评价指标
- 常用的聚类算法
 - 原型聚类
 - 密度聚类
 - 层次聚类
- 谱聚类
- 集成聚类
- 子空间上的聚类

常用聚类算法

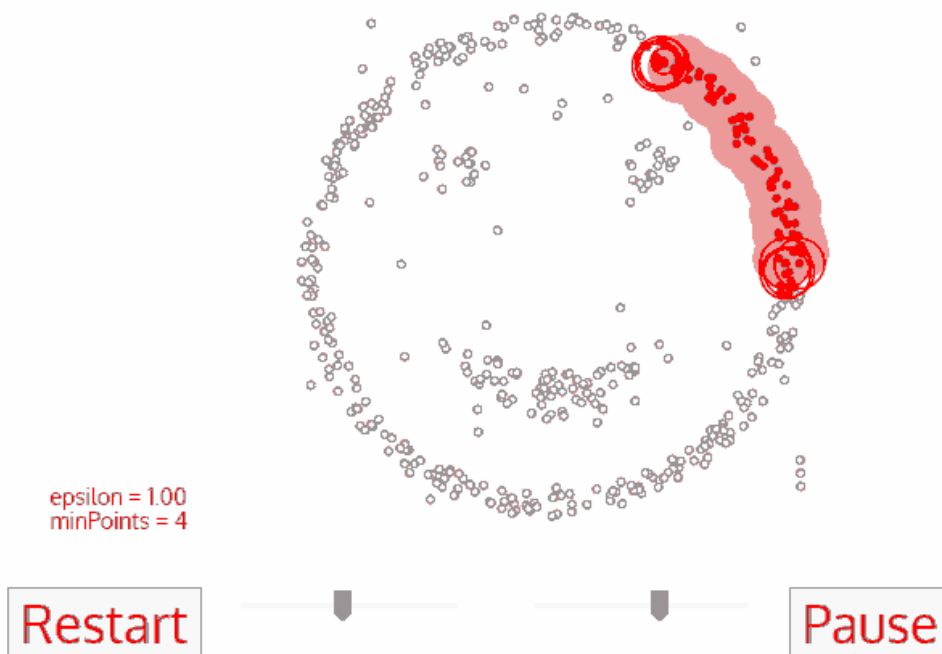
- 密度聚类

- 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。

- 典型算法

- **DBSCAN算法**



DBSCAN算法

- 基本思想

- DBSCAN算法：基于一组“邻域”参数来刻画样本分布的紧密程度。

- 基本概念

- ϵ 邻域：对样本 $x_j \in D$ ，其 ϵ 邻域包含样本集 D 中与 x_j 的距离不大于 ϵ 的样本；
 - 核心对象：若样本 x_j 的 ϵ 邻域至少包含 $MinPts$ 个样本，则该样本点为一个核心对象；

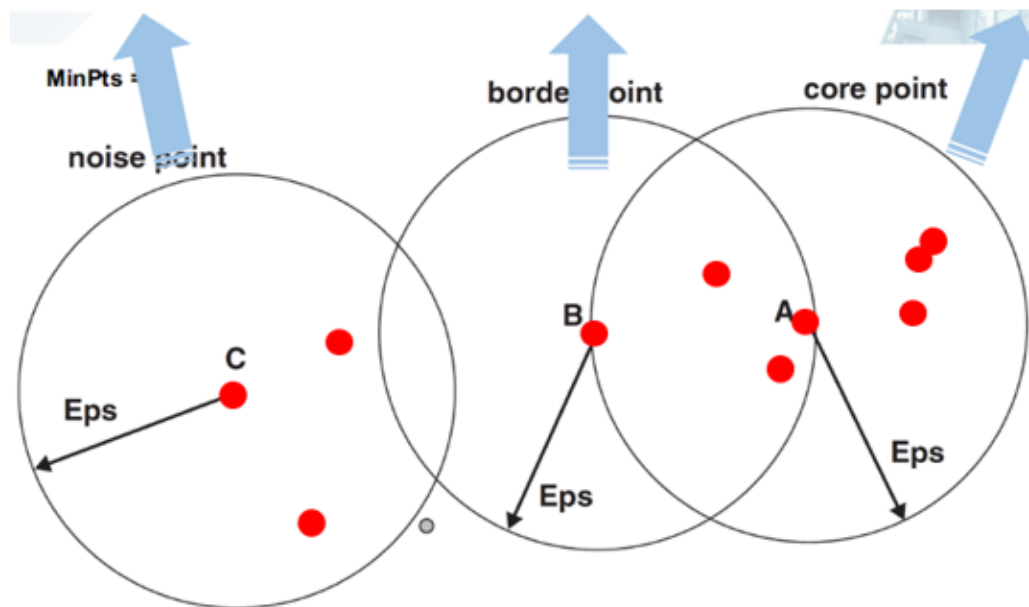
常用聚类算法

● 密度聚类

噪声点为非边缘点和非核心点

边缘点在核心对象的领域内

核心点的邻域内最少包含
MinPts 个样本

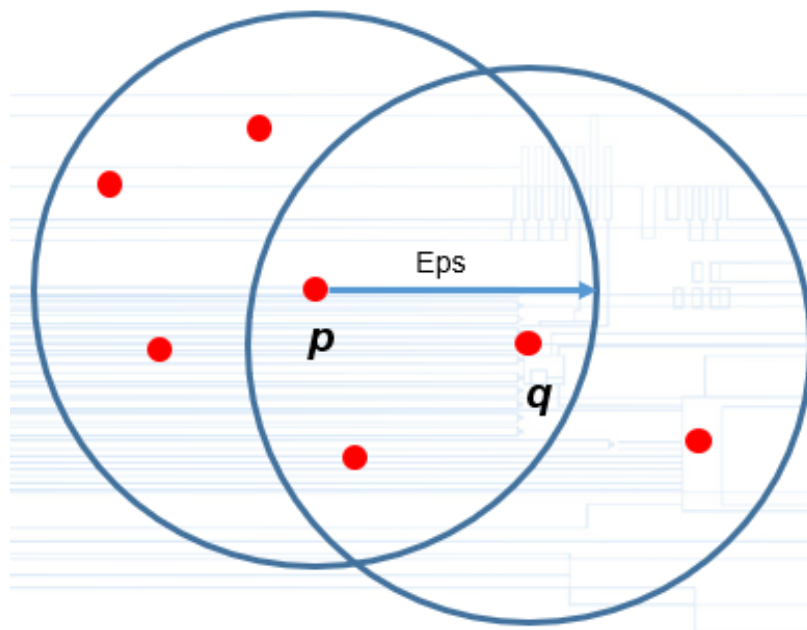


MinPts = 6

常用聚类算法

- 密度聚类

- 例子



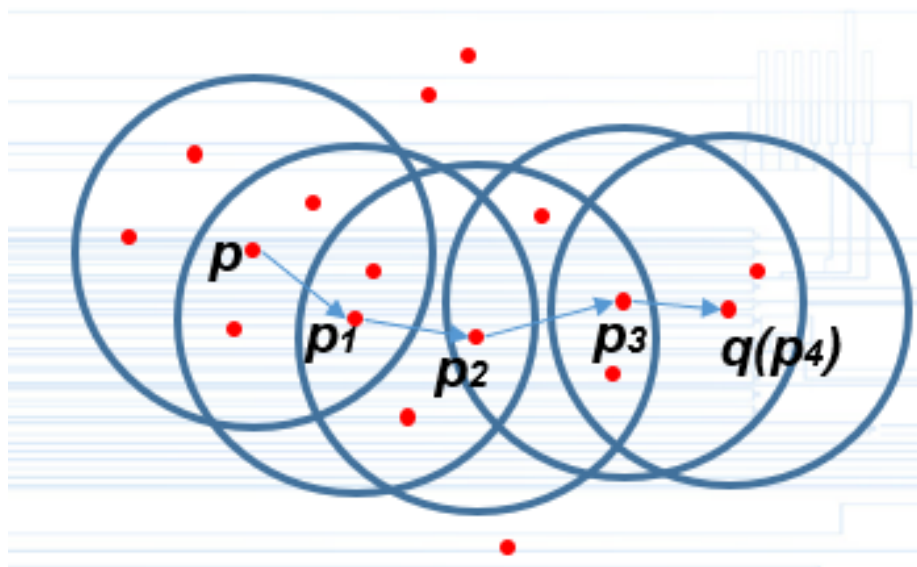
$$\textit{MinPts} = 6$$

- 密度直达：若样本 q 位于样本 p 的 ϵ 邻域中，且 p 是一个核心对象，则称样本 q 由 p 密度直达；
- p 不是由 q 密度直达；
- 密度直达不对称；

常用聚类算法

- 密度聚类

- 例子



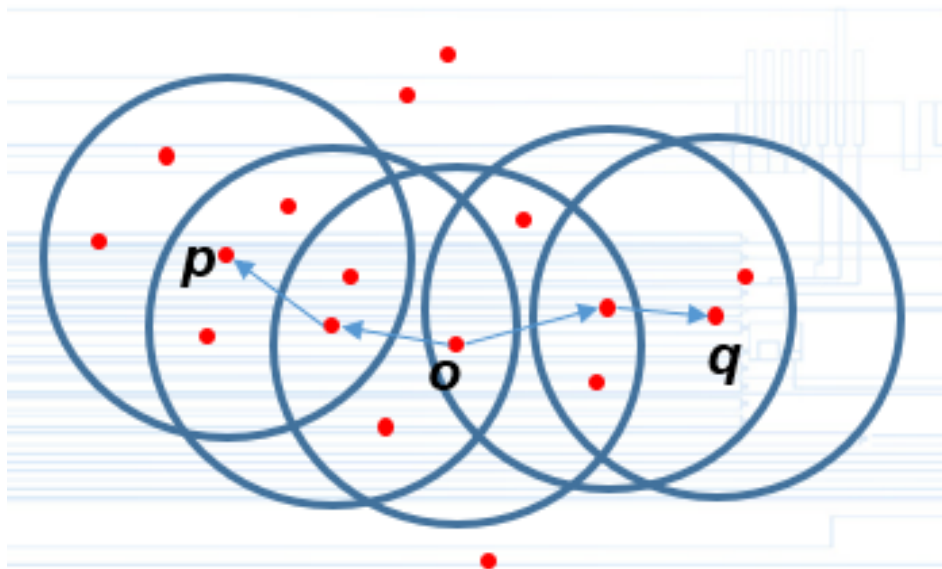
MinPts = 6

- **密度可达**: 对样本 p 与 q ，若存在样本序列 p_1, p_2, \dots, p_n ，其中 $p_1 = p, p_n = q$ 且 p_{i+1} 由 p_i 密度直达，则 q 由 p 密度可达；
- p 不由 q 密度可达；
- 密度可达不对称

常用聚类算法

- 密度聚类

- 例子



MinPts = 6

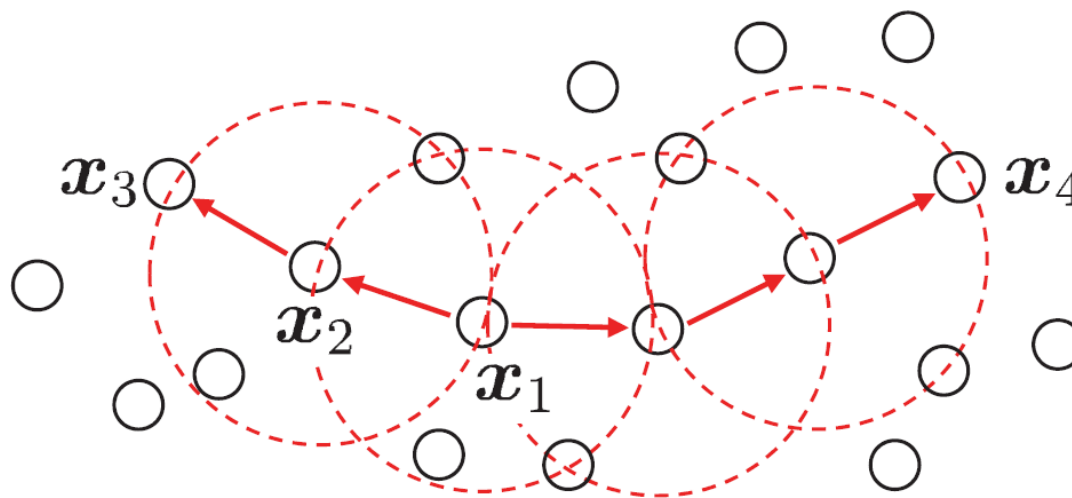
- 若 p 和 q 都可以有同一个点 o 密度可达, 则成 p, q 密度相连;
- q, p 密度相连;
- 密度相连对称;

常用聚类算法

- 密度聚类

- 例子

令 $MinPts=3$ ，则虚线显示出 ϵ 领域， 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。



常用聚类算法

- 密度聚类

- 簇

- **定义**：由**密度可达**关系导出的最大密度相连样本集合
 - **形式化描述**：给定领域参数，簇是满足以下条件的非空样本子集：

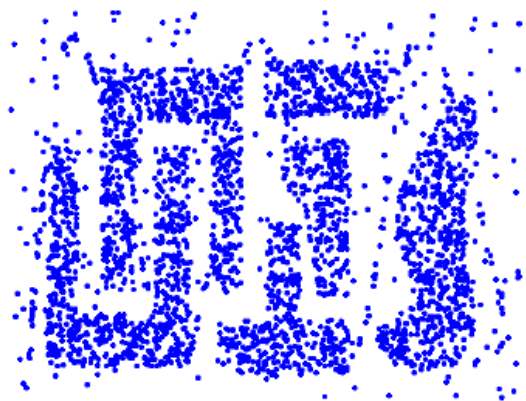
- 连接性： $x_i \in C, x_j \in C$, 可推出, x_i 与 x_j 密度相连

- 最大性： $x_i \in C, x_i$ 与 x_j 密度可达, 可推出 $x_j \in C$

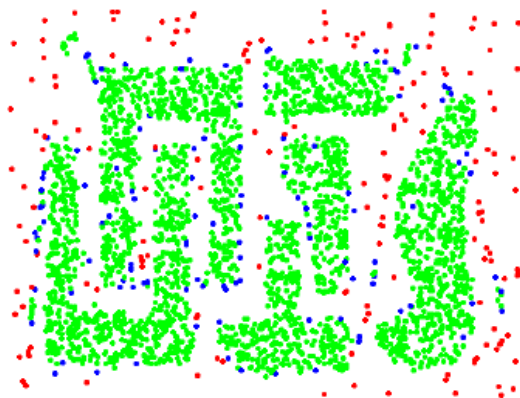
实际上，若 x 为核心对象，由 x 密度可达的所有样本组成的集合记为 $X = \{x' \in D | x' \text{由} x \text{密度可达}\}$ ，则 X 为**满足连接性与最大性的簇**。

常用聚类算法

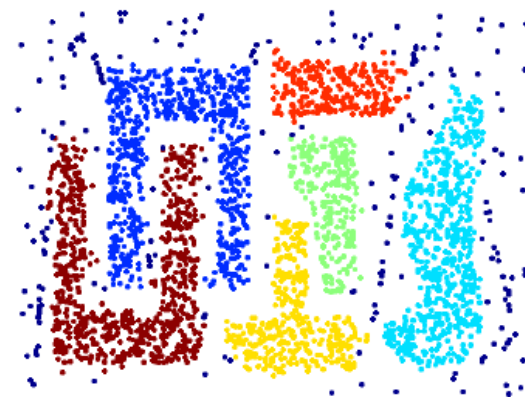
- 密度聚类
 - DBSCAN一个例子



Original Points



Point types: **core**,
border and **noise**



Clusters

https://cse.buffalo.edu/~jing/cse601/fa12/materials/clustering_density.pdf

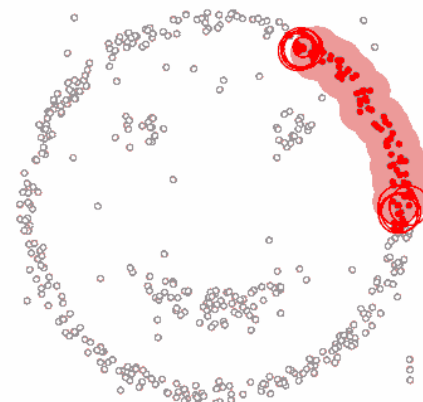
DBSCAN算法步骤

输入: 样本集 $D = \{x_1, x_2, \dots, x_m\}$;
邻域参数 $(\epsilon, MinPts)$.

过程:

```
1: 初始化核心对象集合:  $\Omega = \emptyset$ 
2: for  $j = 1, \dots, m$  do
3:   确定样本  $x_j$  的  $\epsilon$ -邻域  $N_\epsilon(x_j)$ ;
4:   if  $|N_\epsilon(x_j)| \geq MinPts$  then
5:     将样本  $x_j$  加入核心对象集合:  $\Omega = \Omega \cup \{x_j\}$ 
6:   end if
7: end for
8: 初始化聚类簇数:  $k = 0$ 
9: 初始化未访问样本集合:  $\Gamma = D$ 
10: while  $\Omega \neq \emptyset$  do
11:   记录当前未访问样本集合:  $\Gamma_{old} = \Gamma$ ;
12:   随机选取一个核心对象  $o \in \Omega$ , 初始化队列  $Q = \langle o \rangle$ ;
13:    $\Gamma = \Gamma \setminus \{o\}$ ;
14:   while  $Q \neq \emptyset$  do
15:     取出队列  $Q$  中的首个样本  $q$ ;
16:     if  $|N_\epsilon(q)| \geq MinPts$  then
17:       令  $\Delta = N_\epsilon(q) \cap \Gamma$ ;
18:       将  $\Delta$  中的样本加入队列  $Q$ ;
19:        $\Gamma = \Gamma \setminus \Delta$ ;
20:     end if
21:   end while
22:    $k = k + 1$ , 生成聚类簇  $C_k = \Gamma_{old} \setminus \Gamma$ ;
23:    $\Omega = \Omega \setminus C_k$ 
24: end while
25: return 簇划分结果
```

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



epsilon = 1.00
minPoints = 4

Restart

Pause

内容提要

- 聚类与无监督学习
- 聚类评价指标
- 常用的聚类算法
 - 原型聚类
 - 密度聚类
 - 层次聚类
- 谱聚类
- 集成聚类
- 子空间上的聚类

常用聚类算法

- 层次聚类

- 层次聚类试图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”的分拆策略来确定。

- 自底向上

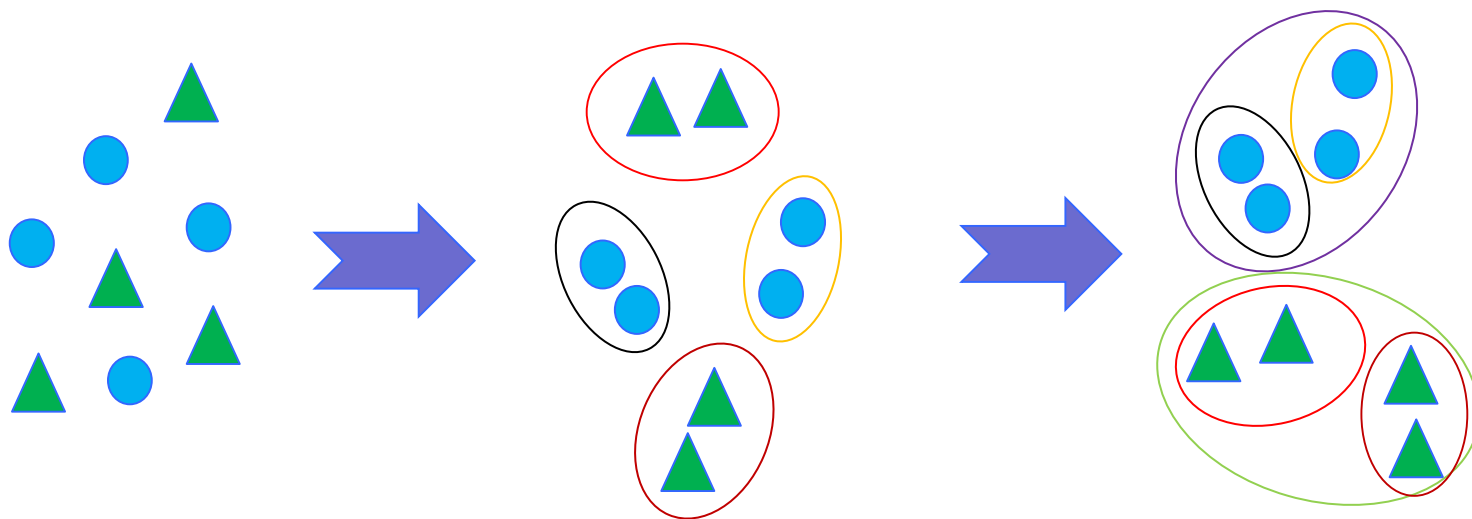
- 将每个样本作为一个簇，然后根据给定的规则逐渐合并一些样本，形成更大的簇，直到所有的样本都被分到一个簇中。

- (1) 初始化：每个样本形成一个类
- (2) 合并：计算任意两个类之间的距离（或相似性），将距离最小（或相似性最大）的两个类合并为一个类，记录下这两个类之间的距离（或相似性），其余类不变
- (3) 重复步骤（2），直到所有样本被合并到一个类之中

常用聚类算法

- 层次聚类

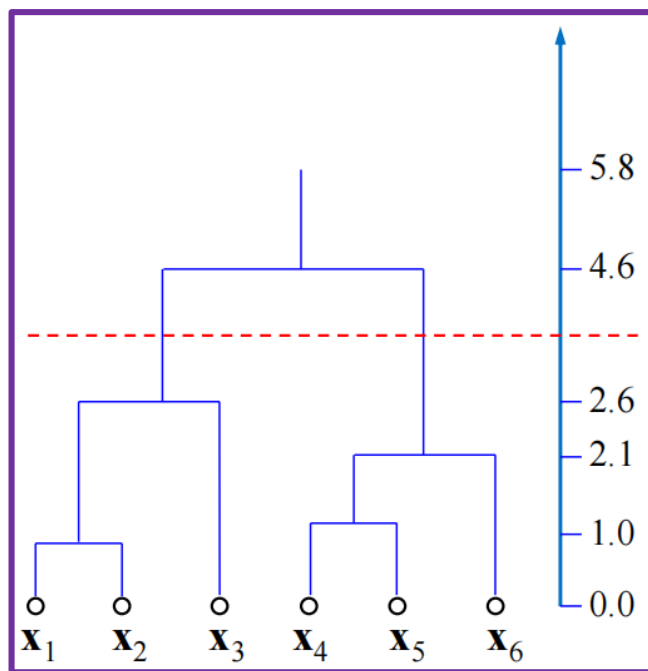
— 层次聚类试图在不同层次对数据集进行划分，从而形成树形的聚类结构。数据集划分既可采用“自底向上”的聚合策略，也可采用“自顶向下”的分拆策略来确定。



常用聚类算法

- 层次聚类

— 聚类树：层次聚类结果用一棵树来表示，称为**聚类树** (dendrogram) 或**系统树图**。



聚类树（采用距离）

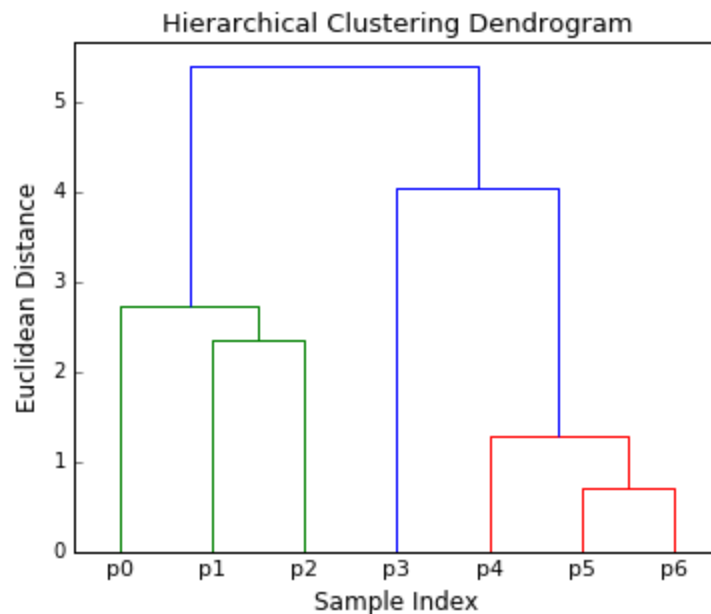
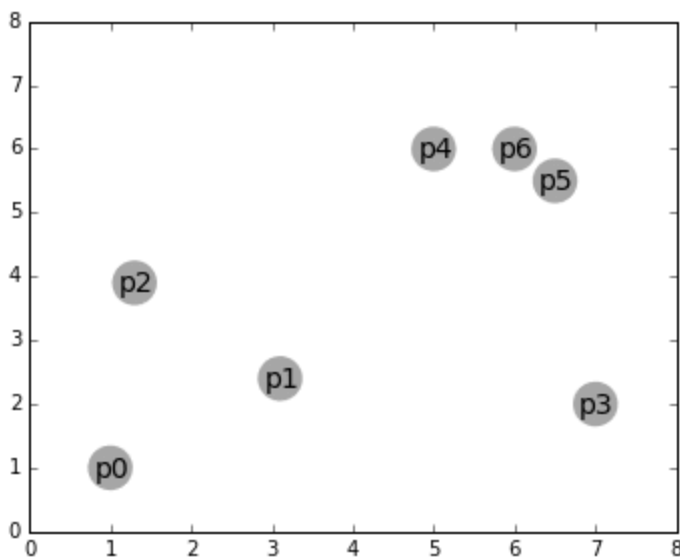
- 最底层的每个节点表示一个样本，采用树枝连接两个合并的样本，树枝的长度反映两个节点之间的距离（或相似性）

常用聚类算法

- 层次聚类

- 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**,

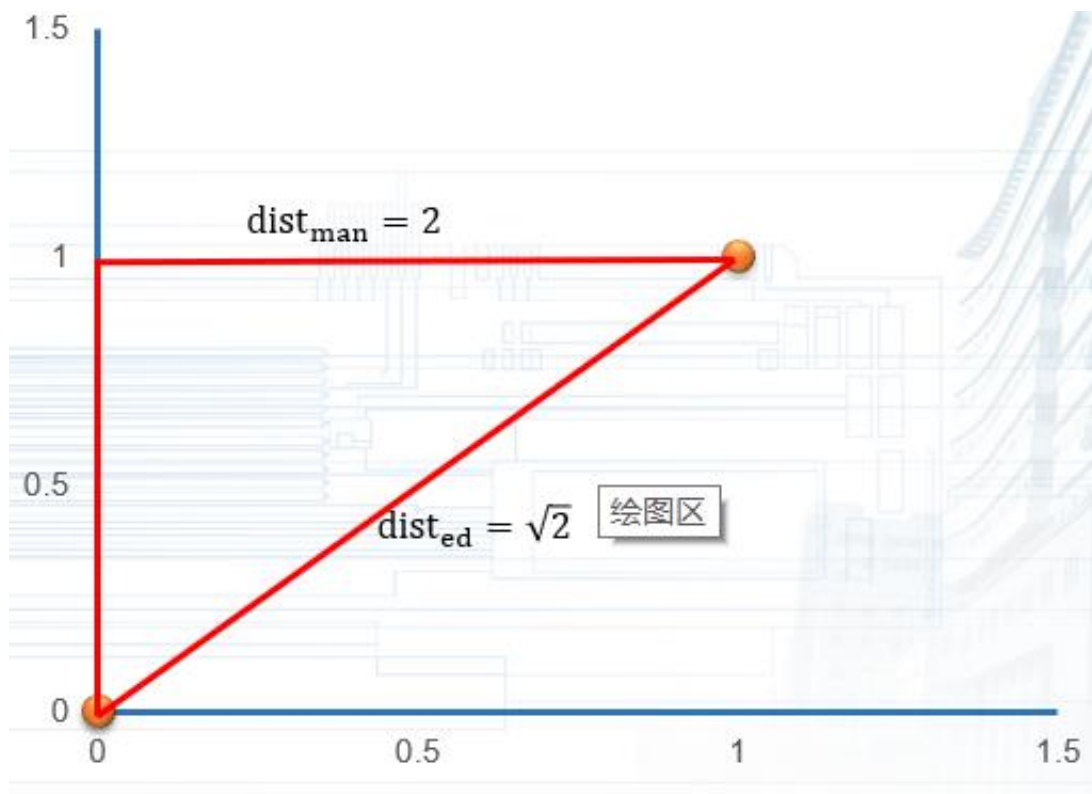


常用聚类算法

- 层次聚类

- AGNES算法（**自底向上**的层次聚类算法）

怎样计算距离？



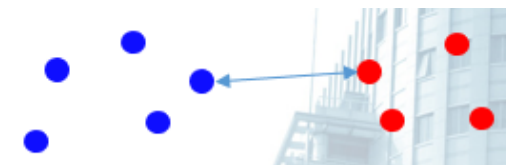
常用聚类算法

- 层次聚类

- 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 $d_{\min}(C_i, C_j) = \min_{x \in C_i, z \in C_j} \text{dist}(x, z) \Rightarrow$



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



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



AGNES算法步骤

输入: 样本集 $D = \{x_1, x_2, \dots, x_m\}$;
聚类簇距离度量函数 $d \in \{d_{\min}, d_{\max}, d_{\text{avg}}\}$;
聚类簇数 k .

过程:

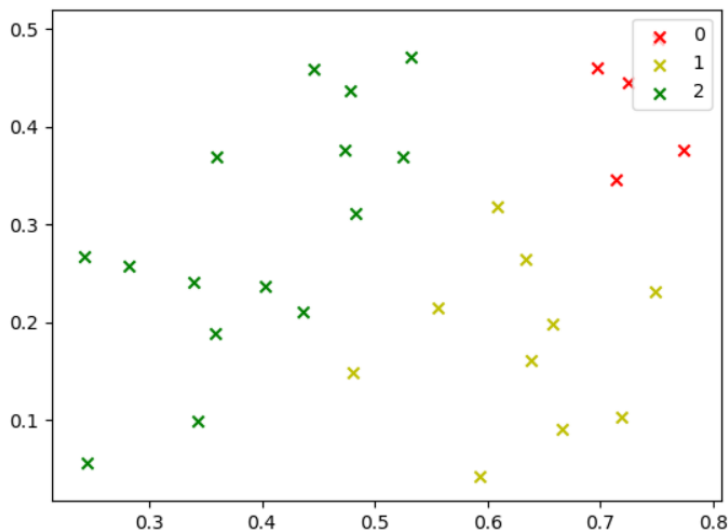
```
1: for  $j = 1, \dots, m$  do
2:    $C_j = \{x_j\}$ 
3: end for
4: for  $i = 1, \dots, m$  do
5:   for  $j = i, \dots, m$  do
6:      $M(i, j) = d(C_i, C_j)$ ;
7:      $M(j, i) = M(i, j)$ 
8:   end for
9: end for
10: 设置当前聚类簇个数:  $q = m$ 
11: while  $q > k$  do
12:   找出距离最近的两个聚类簇( $C_{i^*}, C_{j^*}$ );
13:   合并( $C_{i^*}, C_{j^*}$ ):  $C_{i^*} = C_{i^*} \cup C_{j^*}$ ;
14:   for  $j = j^* + 1, \dots, q$  do
15:     将聚类簇  $C_j$  重编号为  $C_{j-1}$ 
16:   end for
17:   删除距离矩阵  $M$  的第  $j^*$  行与第  $j^*$  列;
18:   for  $j = 1, \dots, q - 1$  do
19:      $M(i^*, j) = d(C_{i^*}, C_j)$ ;
20:      $M(j, i^*) = M(i^*, j)$ 
21:   end for
22:    $q = q - 1$ 
23: end while
24: return 簇划分结果
```

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

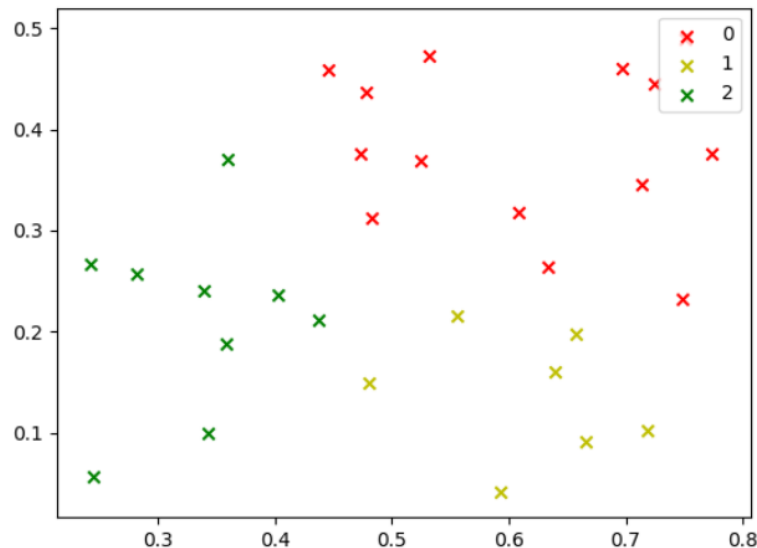
常用聚类算法

- 层次聚类
- 结果

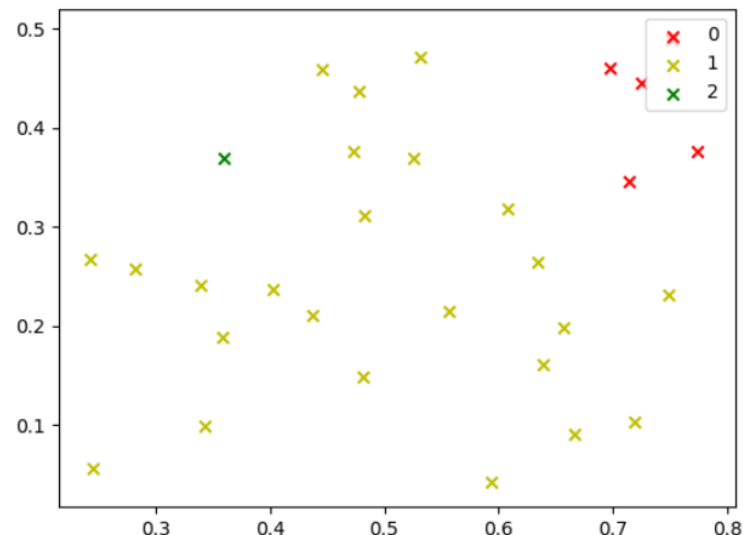
采用 d_{avg}



采用 d_{max}



采用 d_{min}



内容提要

- 聚类与无监督学习
- 聚类评价指标
- 常用的聚类算法
 - 原型聚类
 - 密度聚类
 - 层次聚类
- 谱聚类
- 子空间上的聚类
- 集成聚类

谱聚类 (Spectral Clustering)

— 定义

以**图论**为基础，为空间中的数据点建立关联图，即**距离越近**（**亲和度高**），**边权越高**。接着利用切图的方式分割该图为若干子图，**要求子图之间权重之和尽量低，而子图内部权重尽可能高**，从而达到聚类的目的，是一个图上的关于**顶点划分的最优问题**。

— 亲和矩阵 (相似矩阵)

邻接矩阵的另外一种形式，元素是**边的权重**（称为亲和度）。

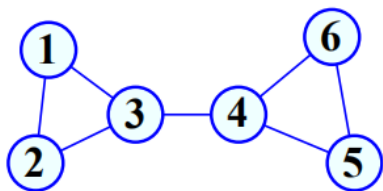
谱聚类 (Spectral Clustering)

— 定义

以**图论**为基础，为空间中的数据点建立关联图，即**距离越近**（**亲和度高**），**边权越高**。接着利用切图的方式分割该图为若干子图，**要求子图之间权重之和尽量低，而子图内部权重尽可能高**，从而达成聚类的目的，是一个图上的关于**顶点划分的最优问题**。

— 亲和矩阵 (相似矩阵)

邻接矩阵的另外一种形式，元素是**边的权重**（称为亲和度）。



$$W = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$



邻接矩阵的元素也可以是**边上的权值**，此时表示两点之间的**相似度（亲和性）**。

谱聚类 (Spectral Clustering)

— 定义

— 亲和矩阵 (相似矩阵)

邻接矩阵的另外一种形式，元素是边的权重（称为亲和度）。

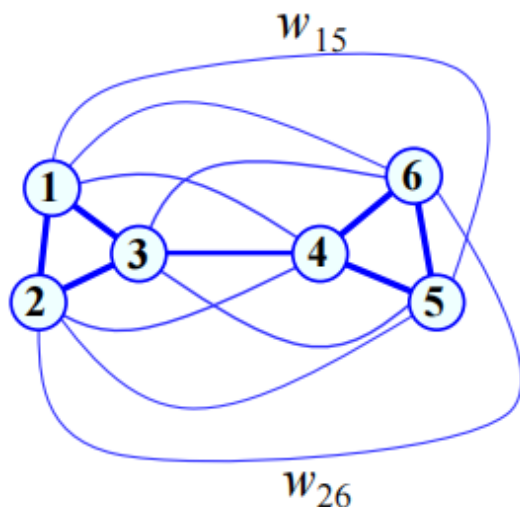
— 拉普拉斯矩阵：度矩阵减去邻接矩阵。

- 度矩阵 D ：邻接矩阵各行元素累加至对应的主对角元素形成的一个对角矩阵。

$$\mathbf{W} = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix} \quad \mathbf{D} = \begin{pmatrix} 2 & & & & & \\ & 2 & & & & \\ & & 3 & & & \\ & & & 3 & & \\ & & & & 2 & \\ & & & & & 2 \end{pmatrix} \quad \mathbf{L} = \mathbf{D} - \mathbf{W} = \begin{pmatrix} 2 & -1 & -1 & 0 & 0 & 0 \\ -1 & 2 & 1 & 0 & 0 & 0 \\ -1 & -1 & 3 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 \\ 0 & 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & -1 & 2 \end{pmatrix}$$

谱聚类 (Spectral Clustering)

— 构图——根据某种测度**构建**点对相似度矩阵



$$\begin{pmatrix} 0 & w_{12} & w_{13} & w_{14} & w_{15} & w_{16} \\ & 0 & w_{23} & w_{24} & w_{25} & w_{26} \\ & & 0 & w_{34} & w_{35} & w_{36} \\ \text{对} & & & 0 & w_{45} & w_{46} \\ & & & & 0 & w_{56} \\ & & & & & 0 \end{pmatrix}$$

点对相似度矩阵

谱聚类 (Spectral Clustering)

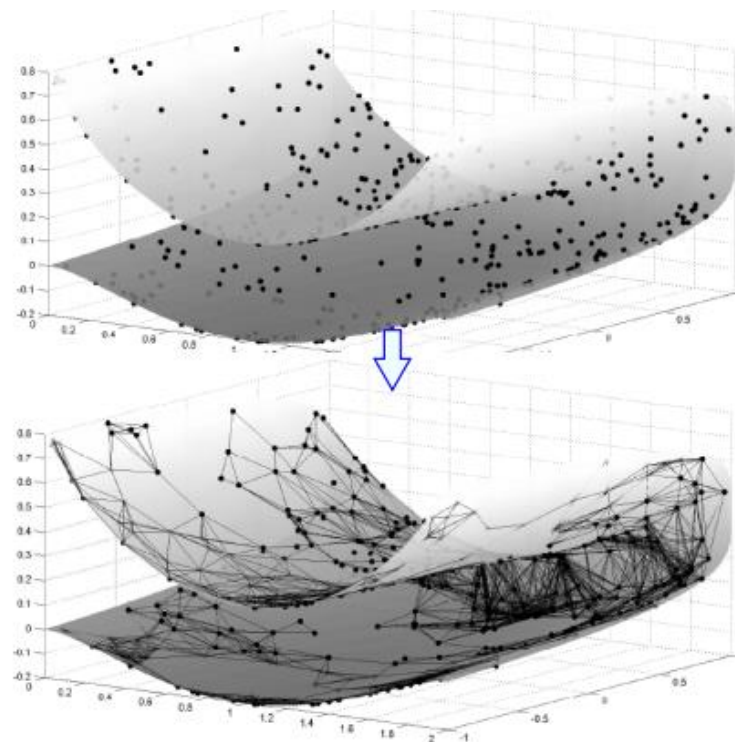
— 构图——根据某种测度**构建**点对相似度矩阵

— 全连接

— 局部连接

- k - 近邻
- ε - 邻近

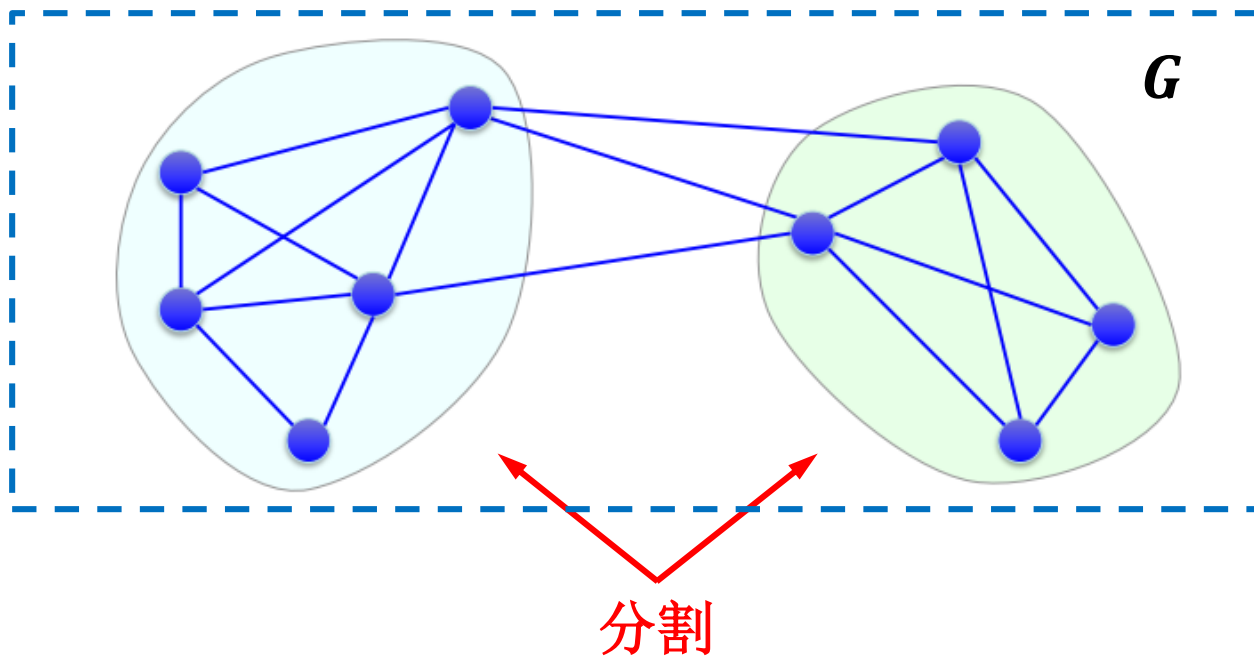
k -近邻: 对每个数据点 x_i , 首先在所有样本中找出不包含 x_i 的 k 个最邻近的样本点, 然后 x_i 与每个邻近样本点均有一条边相连, 从而完成图构造。



谱聚类 (Spectral Clustering)

— 图切割

- **分割**: 设 A_1, A_2, \dots, A_k 为顶点集合 V 的非空连通子集, 如果 $A_i \cap A_j = \emptyset, i \neq j$ 且 $A_1 \cup A_2 \cup \dots \cup A_k = V$, 则称 A_1, A_2, \dots, A_k 为图 G 的一个**分割**。

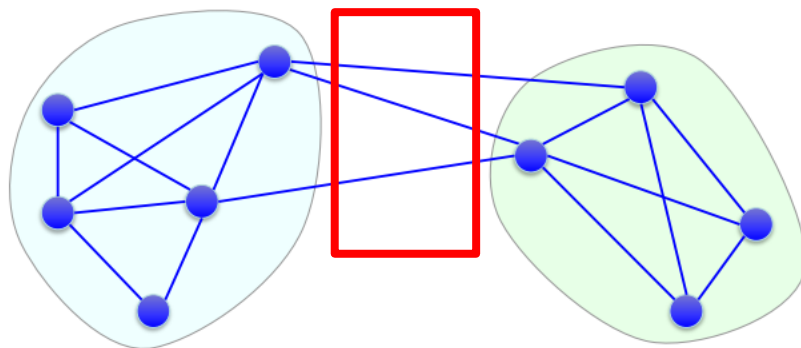


谱聚类 (Spectral Clustering)

— 图切割

- **分割**: 设 A_1, A_2, \dots, A_k 为顶点集合 A 的非空连通子集, 如果 $A_i \cap A_j = \emptyset, i \neq j$ 且 $A_1 \cup A_2 \cup \dots \cup A_k = V$, 则称 A_1, A_2, \dots, A_k 为图 G 的一个**分割**。
- **子图相似度**: 子图 A 与子图 B 的**相似度**定义为连接两个子图所有边的权重之和:

$$W(A, B) = \sum_{i \in A, j \in B} w_{ij}$$



谱聚类 (Spectral Clustering)

— 图切割

- **分割**: 设 A_1, A_2, \dots, A_k 为顶点集合 A 的非空连通子集, 如果 $A_i \cap A_j = \emptyset, i \neq j$ 且 $A_1 \cup A_2 \cup \dots \cup A_k = V$, 则称 A_1, A_2, \dots, A_k 为图 G 的一个**分割**。
- **子图相似度**: 子图 A 与子图 B 的**相似度**定义为连接两个子图所有边的权重之和:

$$W(A, B) = \sum_{i \in A, j \in B} w_{ij}$$

- **子图之间的切割**:

$$cut(A, B) = W(A, B) = \sum_{i \in A, j \in B} w_{ij}$$

不相连, 权重为0

谱聚类 (Spectral Clustering)

— 图切割

- 最小二分切割 (Minimum bipartitional cut)

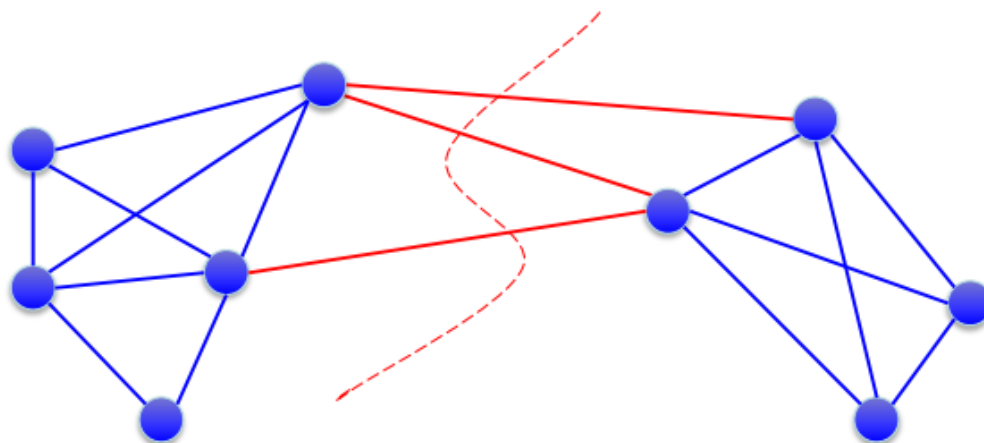
- 在所有的图切割中，找一个**最小代价的切割**，将图分为两个**不连通**的子图。即切开之后，两个子图之间的**相似性**要最小。

谱聚类 (Spectral Clustering)

— 图切割

- 最小二分切割 (Minimum bipartitional cut)

— 在所有的图切割中，找一个**最小代价的切割**，将图分为两个**不连通**的子图。即切开之后，两个子图之间的**相似性**要最小。



谱聚类 (Spectral Clustering)

— 图切割

● 最小二分切割 (Minimum bipartitional cut)

— 在所有的图切割中，找一个**最小代价的切割**，将图分为两个**不连通**的子图。即切开之后，两个子图之间的**相似性**要最小。

— 最优问题

$$\mathit{min}_A \mathit{cut}(A, \bar{A}) := W(A, \bar{A}) = \sum_{i \in A, j \in \bar{A}} w_{ij}$$

$$A \neq \emptyset \quad A \cap \bar{A} \neq \emptyset \quad A \cup \bar{A} \neq V$$

谱聚类 (Spectral Clustering)

— 图切割

- 最小二分切割 (Minimum bipartitional cut)

- 在所有的图切割中，找一个**最小代价的切割**，将图分为两个**不连通**的子图。即切开之后，两个子图之间的**相似性**要最小。

- 最优问题

- 在实践中，上述目标函数通常**将一个点(比如野点)**从其余各点中分离出来。从聚类的角度看，这并不是我们所期望的。

谱聚类 (Spectral Clustering)

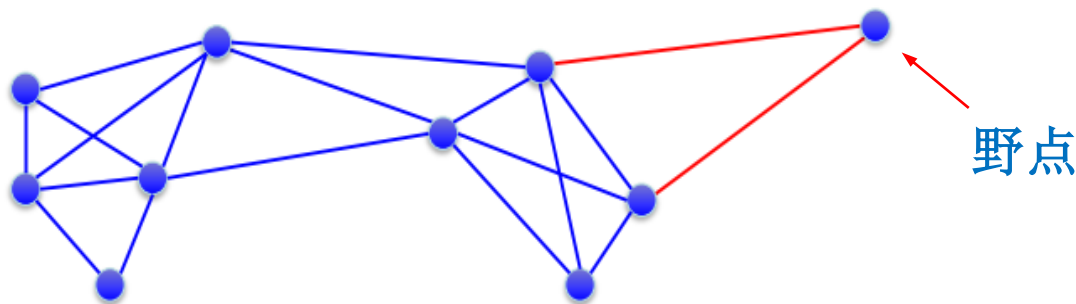
— 图切割

- 最小二分切割 (Minimum bipartitional cut)

- 在所有的图切割中，找一个**最小代价的切割**，将图分为两个**不连通**的子图。即切开之后，两个子图之间的**相似性**要最小。

- 最优问题

- 在实践中，上述目标函数通常**将一个点 (比如野点)**从其余各点中分离出来。从聚类的角度看，这并不是我们所期望的。



谱聚类 (Spectral Clustering)

— 图切割

- 最小二分切割 (Minimum bipartitional cut)

- 在所有的图切割中，找一个**最小代价的切割**，将图分为两个**不连通**的子图。即切开之后，两个子图之间的**相似性**要最小。

- 最优问题

- 在实践中，上述目标函数通常**将一个点(比如野点)**从其余各点中分离出来。从聚类的角度看，这并不是我们所期望的。

- 产生上述问题的原因对子图的规模没有加以限制

谱聚类 (Spectral Clustering)

— 图切割

- 最小二分切割 (Minimum bipartitional cut)

- 归一化最小二分切割

— **基本假设**: 子图之间的规模相差不大。

— **基本做法**: 采用子图的势或者体积来对切割进行归一化。

- 采用子图的**势**:

$$\text{Radiocut}(A, \bar{A}) := \frac{1}{2} \left(\frac{\text{cut}(A, \bar{A})}{|A|} + \frac{\text{cut}(A, \bar{A})}{|\bar{A}|} \right) \quad \bar{A} = V - A$$

- 采用子图的**体积**:

$$\text{Ncut}(A, \bar{A}) := \frac{1}{2} \left(\frac{\text{cut}(A, \bar{A})}{\text{vol}(A)} + \frac{\text{cut}(A, \bar{A})}{\text{vol}(\bar{A})} \right)$$

顶点的**个数**的称为图的**势**，图中所有顶点的**度数之和**称为**体积**

谱聚类 (Spectral Clustering)

— 图切割

- 最小二分切割 (Minimum bipartitional cut)
- 归一化最小二分切割
- K切割 ($K > 2$)

— 考虑将图分成 k 个子图： A_1, A_2, \dots, A_k 。一种直观的方法是将图切割问题理解为**多个二分切割问题的综合**。

谱聚类 (Spectral Clustering)

— 图切割

- 最小二分切割 (Minimum bipartitional cut)

- 归一化最小二分切割

- K切割 ($K > 2$)

— 考虑将图分成 k 个子图: A_1, A_2, \dots, A_k 。一种直观的方法是将图切割问题理解为多个二分切割问题的综合。

- 未归一化切割目标函数

$$\text{cut}(A_1, A_2, \dots, A_k) := \frac{1}{2} \sum_{i=1}^k W(A_i, \bar{A}_i)$$

谱聚类 (Spectral Clustering)

— 图切割

- 最小二分切割 (Minimum bipartitional cut)

- 归一化最小二分切割

- K切割 ($K > 2$)

— 考虑将图分成 k 个子图: A_1, A_2, \dots, A_k 。一种直观的方法是将图切割问题理解为**多个二分切割问题的综合**。

- 未归一化切割目标函数

- 比例切割目标函数

$$Radiocut(A_1, A_2, \dots, A_k) := \frac{1}{2} \sum_{i=1}^k \frac{W(A_i, \bar{A}_i)}{|A_i|} = \sum_{i=1}^k \frac{cut(A_i, \bar{A}_i)}{|A_i|}$$

谱聚类 (Spectral Clustering)

— 图切割

- 最小二分切割 (Minimum bipartitional cut)

- 归一化最小二分切割

- K切割 ($K > 2$)

— 考虑将图分成 k 个子图: A_1, A_2, \dots, A_k 。一种直观的方法是将图切割问题理解为**多个二分切割问题的综合**。

- 未归一化切割目标函数

- 比例切割目标函数

- 归一化切割目标函数

$$Ncut(A_1, A_2, \dots, A_k) := \frac{1}{2} \sum_{i=1}^k \frac{W(A_i, \bar{A}_i)}{vol(A_i)} = \sum_{i=1}^k \frac{cut(A_i, \bar{A}_i)}{vol(A_i)}$$

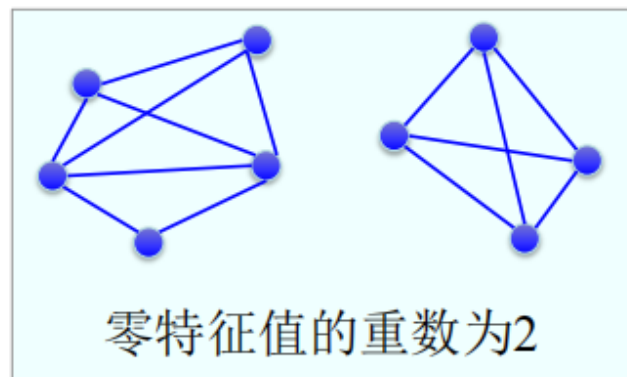
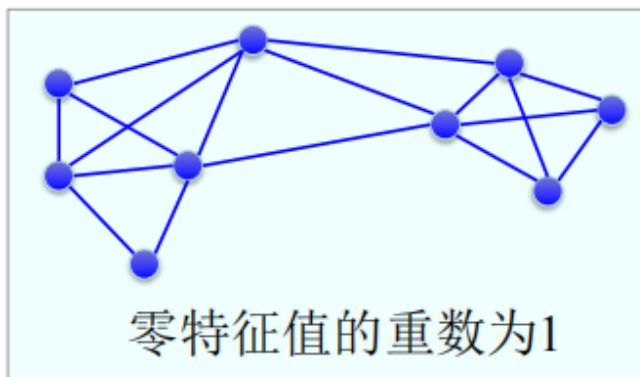
图聚类

谱聚类 (Spectral Clustering)

— 性质

- 拉普拉斯矩阵: $L = D - W$
- 图的**连通子图**与**拉普拉斯矩阵** L 的特征值的关系

★ — 设 G 为一个具有非负连接权重的无向图，由图 G 导出的**拉普拉斯矩阵**的**零特征值**的重数等于图 G 的**连通子图**的个数 k 。



谱聚类 (Spectral Clustering)

— 性质

- 拉普拉斯矩阵: $L = D - W$

- 图的连通子图与拉普拉斯矩阵 L 的特征值的关系

★ — 设 G 为一个具有非负连接权重的无向图, 由图 G 导出的拉普拉斯矩阵的零特征值的重数等于图 G 的连通子图的个数 k 。

- 归一化图拉普拉斯 (Graph Laplacian)

— 有两种构造归一化图拉普拉斯矩阵的方法

- 对称型:

$$L_{sym} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}} = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$$

- 随机游走型 (random walk):

$$L_{rw} = D^{-1} L = I - D^{-1} W$$

谱聚类 (Spectral Clustering)

- 谱聚类算法:

- 根据不同的图拉普拉斯构造方法, 可以得到不同的谱聚类算法形式

- 但是, 这些算法的核心步骤都是**相同**的:

- 利用点对之间的相似性, 构建**亲和度矩阵**;
- 构建**拉普拉斯矩阵**;
- 求解拉普拉斯矩阵**最小的特征值对应的特征向量** (通常舍弃零特征所对应的特征向量) ;
- 由这些特征向量构成样本点的新特征, **采用K-means等聚类方法完成最后的聚类**。

谱聚类 (Spectral Clustering)

算法1: 经典的谱聚类算法步骤 (Un-normalized (classical) Spectral Clustering)

- 1 输入: 亲和度矩阵 W , 聚类个数 k
- 2 计算非标准化的拉普拉斯矩阵 $L = D - W$
- 3 计算 L 的前 k 个特征向量 u_1, u_2, \dots, u_k
- 4, 矩阵 $U \in R^{n \times k}$ 是由上述的特征向量构成的矩阵, 即
$$U = [u_1, u_2, \dots, u_k] \in R^{n \times k}$$
- 5 有循环 $i = 1, 2, \dots, n$, 此时 $y_i \in R^k$ 是 U 矩阵的第 i 行
- 6 分别将 $\{y_i\}_{i=1,2,\dots,n}$ 利用 k-means 算法将它们归类到指定的簇中 A_1, A_2, \dots, A_k
- 7 输出 A_1, A_2, \dots, A_k

谱聚类 (Spectral Clustering)

算法2: 标准谱聚类算法步骤 (Normalized Spectral Clustering, Shi算法)

- 1 输入: 相似度矩阵 W , 聚类个数 k
- 2 compute the un-normalized Laplacian matrix $L = D - W$
- 3 根据广义特征问题 $Lu = \lambda Du$, 计算前 k 个广义特征向量 u_1, u_2, \dots, u_k
- 4 矩阵 $U \in R^{n \times k}$ 是由上述的特征向量构成的矩阵, 即
$$U = [u_1, u_2, \dots, u_k] \in R^{n \times k}$$
- 5 有循环 $i = 1, 2, \dots, n$, 此时 $y_i \in R^k$ 是 U 矩阵的第 i 行
- 6 分别将 $\{y_i\}_{i=1,2,\dots,n}$ 利用k-means算法将它们归类到指定的簇中 A_1, A_2, \dots, A_k
- 7 输出 A_1, A_2, \dots, A_k

谱聚类 (Spectral Clustering)

算法3: 标准谱聚类算法步骤 (Normalized Spectral Clustering, **Ng算法**)

1 输入: 亲和度矩阵 W , 聚类个数 k

2 计算 $L_{sym} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}}$

3 计算 L_{sym} 的前 k 个特征向量 u_1, u_2, \dots, u_k

4 矩阵 $U \in R^{n \times k}$ 是由上述的特征向量构成的矩阵, 即

$$U = [u_1, u_2, \dots, u_k] \in R^{n \times k}$$

5 将 U 矩阵的每一行标准化为1-范式, 形成矩阵 $T \in R^{n \times k}$

$$t_{ij} = \frac{u_{ij}}{\sqrt{\sum_{m=1}^k u_{im}^2}}$$

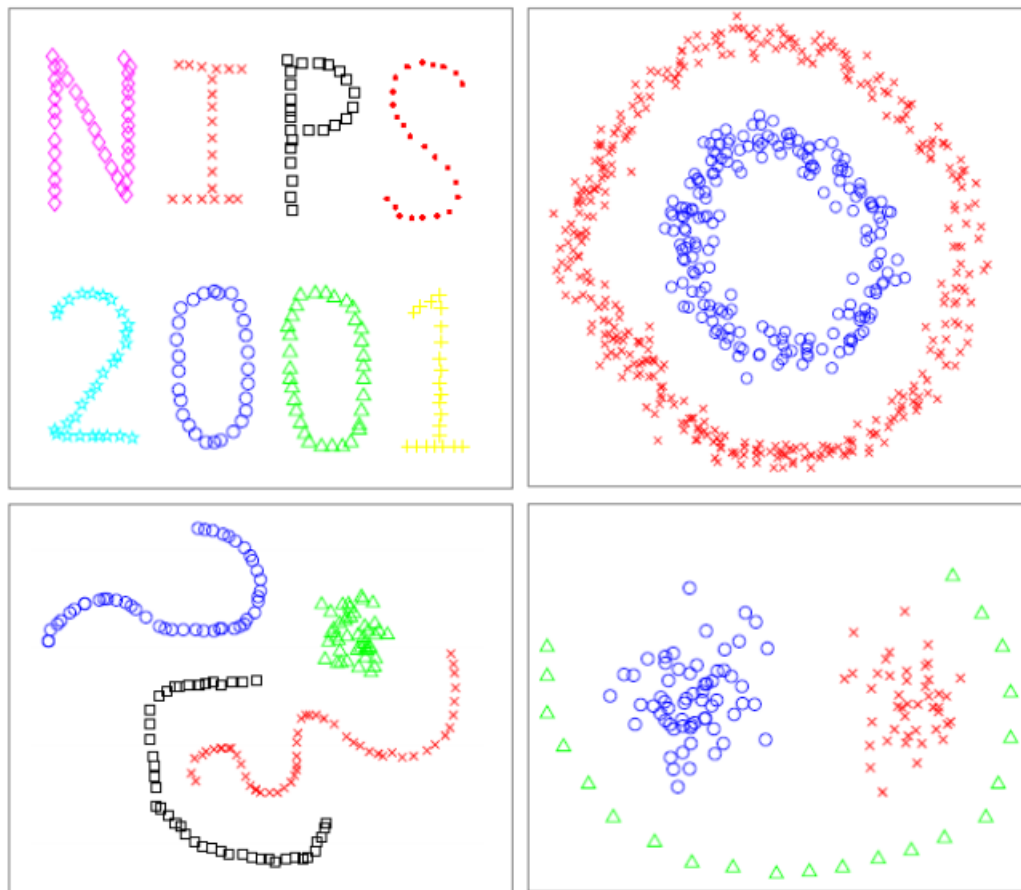
6 有循环 $i = 1, 2, \dots, n$, 此时 $y_i \in R^k$ 是 U 矩阵的第 i 行

7 分别将 $\{y_i\}_{i=1,2,\dots,n}$ 利用**k-means算法**将它们归类到指定的簇中 A_1, A_2, \dots, A_k

8 输出 A_1, A_2, \dots, A_k

谱聚类 (Spectral Clustering)

- 一些例子



A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: analysis and an algorithm. NIPS, pp. 849-856, 2002.

内容提要

- 聚类与无监督学习
- 聚类评价指标
- 常用的聚类算法
 - 原型聚类
 - 密度聚类
 - 层次聚类
- 谱聚类
- 子空间上的聚类
- 集成聚类

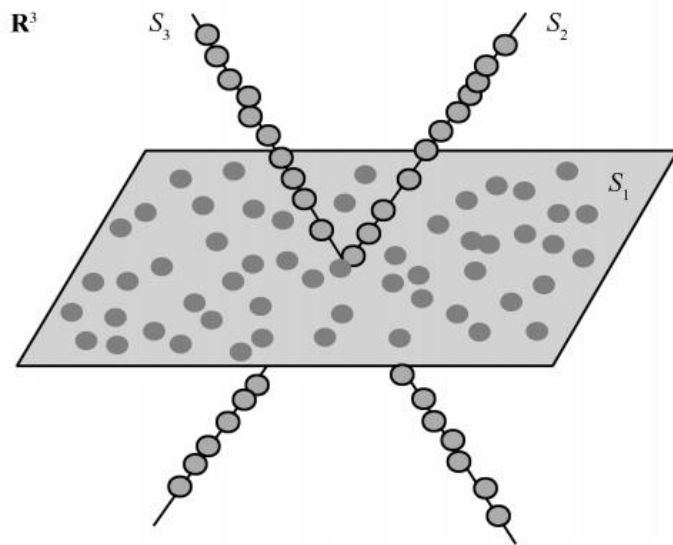
子空间聚类 (Clustering ensemble)

— 定义

给定一个 n 个样本构成的矩阵 $X = [x_1, x_2, \dots, x_n] \in R^{m \times n}$, $x_i \in R^m$, 并且已知这 n 个样本分别来自 k 个子空间 $S_i, i = 1, 2, \dots, k$, 令 $d_i < m, i = 1, 2, \dots, k$ 表示 k 个子空间的维度, d_i 未知。

— 目标

将上述的 n 个样本正确地规划到各自所属的子空间中去, 即将 n 个样本聚成 k 类, 每一类就是一个子空间。



该空间由一个二维平面 S_1 与两条直线 S_2, S_3 分为三个子空间

子空间聚类 (Clustering ensemble)

— 主流算法

- **基于统计的方法**：混合数据假设是从**服从某一概率分布**（如**混合高斯分布**）中**抽取出的独立样本集**，于是数据的分割问题就转化为一模型估计问题。代表性的工作有凝聚有损压缩和随机抽样一致。
- **基于矩阵分解的方法**：将**数据矩阵分解为一正交基矩阵和一低秩矩阵的乘积**，从分解结果的结构来揭示聚类的**特性**。当子空间含有噪声和奇异值，或者独立子空间的假设不成立时，此类方法的效果不尽人意。代表性的工作有**K子空间分割**。

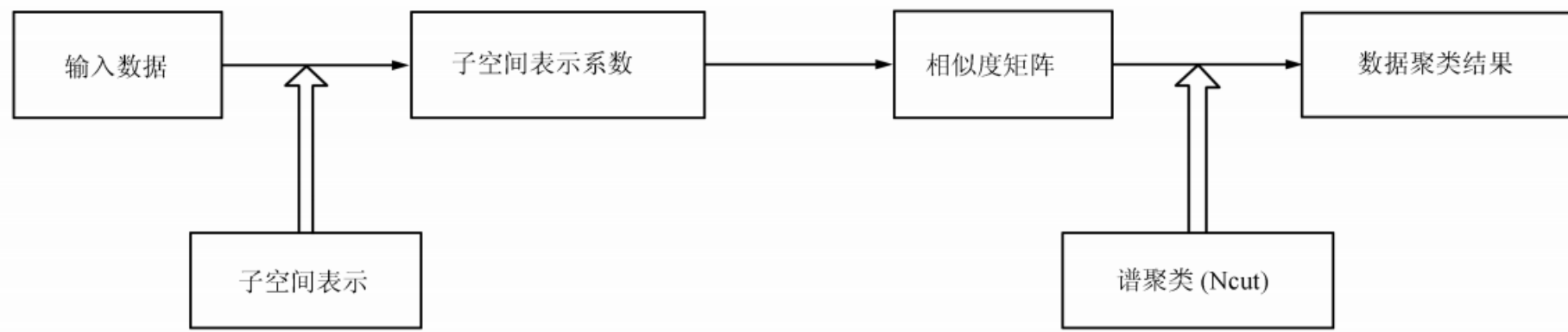
子空间聚类 (Clustering ensemble)

— 主流算法

- **基于代数的方法**：以处理子空间不是相互独立的情况，但计算量大，且**对噪声和奇异值敏感**。代表性的工作有 **Generalized PCA (GPCA)**
- **基于谱聚类的方法**：基于谱聚类的子空间分割算法先根据观测样本求得一个**相似矩阵**，然后对这个**相似矩阵进行谱聚类**获得最终的聚类结果。代表性的工作有**稀疏子空间聚类**和**低秩表示子空间聚类**

子空间聚类 (Clustering ensemble)

— 稀疏子空间聚类法



稀疏子空间聚类法大致流程

稀疏子空间聚类

稀疏子空间聚类算法步骤

- 1 给定数据 $X = [x_1, x_2, \dots, x_n] \in R^{D \times N}$, 空间 $S = [S_1, S_2, \dots, S_n]$
- 2 将数据 $x_i \in S_a$ 表示为其他数据的线性组合, 即:
- 3
$$x_i = \sum_{j \neq i} Z_{ij} x_j$$
- 4 对表示系数 Z_{ij} 施加约束使得对所有的 $x_j \notin S_a$:
- 5
$$Z_{ij} = 0$$
- 6 将所有数据及其表示系数按照一定规律排列为矩阵, 则2中的式子等价于 $X = XZ$
- 7 则系数矩阵 $Z \in R^{N \times N}$ 满足: x_i 与 x_j 属于不同子空间, $Z_{ij} = 0$
- 8 若已知数据子空间结构, 并将数据按类别逐列排放, 则在一定条件下可使系数矩阵 Z 具有块对角结构, 即
$$\begin{pmatrix} Z_1 & & & \\ & Z_2 & & \\ & & \ddots & \\ & & & Z_k \end{pmatrix}$$
 这里 $Z_a (a = 1, 2, \dots, k)$ 表示子空间 S_a 表示系数矩阵。
- 9 对系数矩阵 Z 采用不同的**稀疏约束**, 使其尽可能 具有理想结构, 从而实现子空间聚类
即 $\min \|Z\|_1 \text{ s.t. } X = XZ$, 加上噪声为 $\min \|Z\|_1 + \lambda \|E\|_F \text{ s.t. } X = XZ + E$
- 10 利用谱聚类得到聚类结果

参考自: 王卫卫, 稀疏子空间聚类综述, 自动化学报, 2015.

参考自: <https://xijunlee.github.io/2016/12/22/2016-12-22-man-tan-gao-wei-shu-ju-ju-lei-2-zi-kong-jian-ju-lei/>

内容提要

- 聚类与无监督学习
- 聚类评价指标
- 常用的聚类算法
 - 原型聚类
 - 密度聚类
 - 层次聚类
- 谱聚类
- 子空间上的聚类
- 集成聚类

集成聚类 (Clustering ensemble)

– 定义

集成是为了提高聚类结果的准确性、稳定性和鲁棒性的一种算法，通过集成多个基聚类结果可以产生一个**较优**的结果。

– 步骤

生成：采用不同的聚类算法和不同的初始化参数，可以得到多个聚类成员。

选择：从生成的聚类成员中选择要集成的聚类成员。

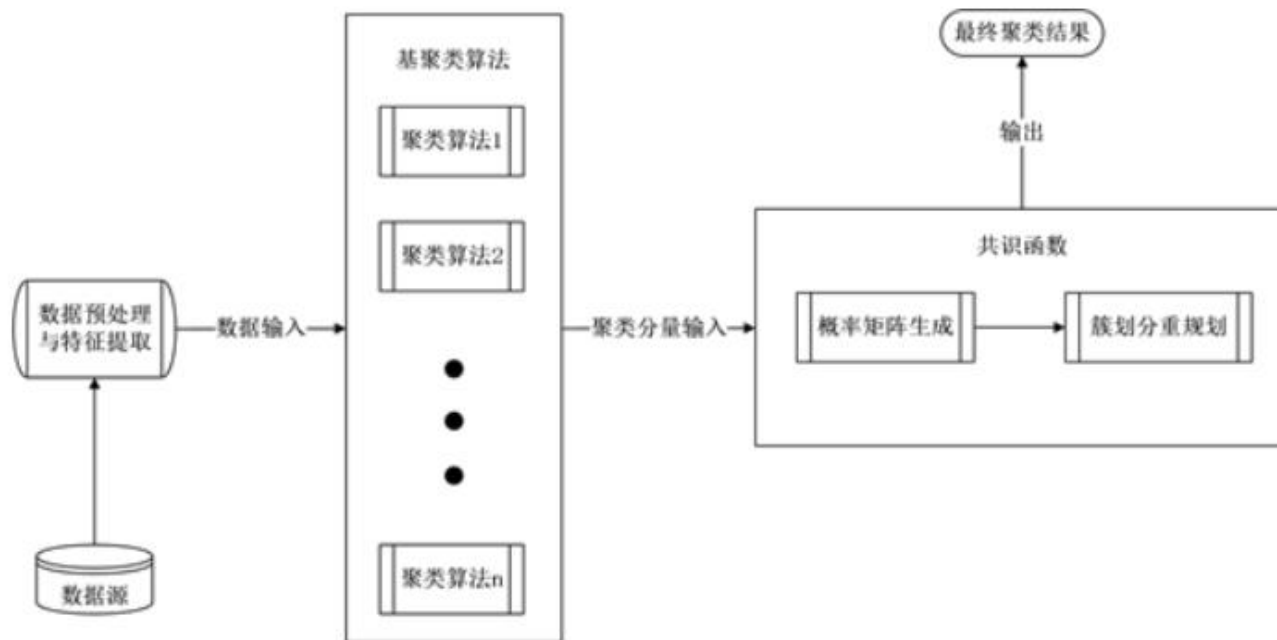
集成：基于共识函数（一致性函数）对聚类成员进行集成。

集成聚类 (Clustering ensemble)

— 基本思想

用多个独立的**基聚类器**分别对原始数据集进行聚类，然后使用某种**集成方法**进行处理，并获得一个最终的集成结果。

— 目标



- 第一阶段，应尽可能地使用**多种方式**来获取基聚类结果。
- 第二阶段，应选择一个**最合适的集成解决方案**来处理这些结果。

集成聚类 (Clustering ensemble)

- 聚类集成方法

- 基于相似度的方法：以相似度矩阵表达基聚类器的信息，然后使用矩阵平均的方式结合多个聚类结果。
- 基于图的方法：以无向图表达基聚类器的信息，然后用图分割的方式结合结果。
- 基于重标记的方法：以标记向量表达基聚类器的信息，然后用标记指派的方式结合结果。
- 基于变换的方法：以特征重表示来表示基聚类器信息，然后用聚类的方式结合结果。

集成聚类 (Clustering ensemble)

- 基于相似度的方法

- 以相似度矩阵表达基聚类器的信息，然后使用矩阵平均的方式结合多个聚类结果。

$$c^{(1)} = \{1, 1, 2, 2, 3\}$$

$$c^{(2)} = \{1, 2, 2, 2, 3\}$$

$$c^{(3)} = \{2, 2, 3, 1, 3\}$$



1				
1	1			
0	0	1		
0	0	1	1	
0	0	0	0	1



平均

1				
0	1			
0	1	1		
0	1	1	1	
0	0	0	0	1

集成聚类 (Clustering ensemble)

- 基于图的方法

- 以无向图表达基聚类器的信息，然后用图分割的方式结合结果。

$$c^{(1)} = \{1, 1, 2, 2, 3\}$$

$$c^{(2)} = \{1, 2, 2, 2, 3\}$$

$$c^{(3)} = \{2, 2, 3, 1, 3\}$$

构建超图



$$\left\{ \begin{array}{l} c_1^{(1)}, c_2^{(1)}, c_3^{(1)}, c_1^{(2)}, c_2^{(2)}, \\ c_3^{(2)}, c_1^{(3)}, c_2^{(3)}, c_3^{(3)} \end{array} \right\}$$

$$c_1^{(1)} = \{x_1, x_2\}$$

$$c_1^{(1)} = \{x_3, x_4\}$$

$$c_1^{(1)} = \{x_5\} \quad c_1^{(2)} = \{x_1\}$$

$$c_2^{(2)} = \{x_2, x_3, x_4\}$$

集成聚类 (Clustering ensemble)

- 基于重标记的方法


- 标记向量表达基聚类器的信息，然后用标记指派的方式结合结果
- 简单投票法、加权投票法、选择性投票法、选择性加权投票法

$$\begin{array}{l} x_1, x_2, x_3, x_4, x_5 \\ c^{(1)} = \{1, 1, 2, 2, 3\} \\ c^{(2)} = \{1, 2, 2, 2, 3\} \\ c^{(3)} = \{2, 2, 3, 1, 3\} \end{array} \xrightarrow{\text{简单投票法}} c = \{1, 2, 2, 2, 3\}$$

集成聚类 (Clustering ensemble)

- 基于变换的方法

- 将每个示例重新表示为一个 r 元组，其中 r 是基聚类器的数量； r 元组中的第 q 个元素表示第 q 个基聚类器对该示例的簇分配，最终可以通过在 r 元组上进行簇类分析得到聚类集成结果。

$c^{(1)} = \{1, 1, 2, 2, 3\}$	转化为 $r=3$ 元组 	$\hat{x}_1 = \{1, 1, 2\}^T$
$c^{(2)} = \{1, 2, 2, 2, 3\}$		$\hat{x}_2 = \{1, 2, 2\}^T$
$c^{(3)} = \{2, 2, 3, 1, 3\}$		$\hat{x}_3 = \{2, 2, 3\}^T$
		$\hat{x}_4 = \{2, 2, 1\}^T$
		$\hat{x}_5 = \{3, 3, 3\}^T$

参考文献

- 周志华. 《机器学习》
- 李航. 《统计学习方法》

致 谢

- 感谢**向世明**老师的20版PPT作为原始材料
- 感谢**王锐**与**段俊贤**对本PPT的制作与修改

Thank All of You!
(Questions?)

赫然

rhe@nlpr.ia.ac.cn

<http://rhe-web.github.io/>

智能感知与计算研究中心 (CRIPAC)

中科院自动化研究所 模式识别国家重点实验室