Clustering and Outlier Detection

👉 聚类定义

- 将数据分为多个簇，使得在同一个簇内对象间具有较高的相似度，而不同簇间差别较大
- 是**无监督学习**
- 聚类目的——寻找数据中潜在的**自然分组结构**

👉 聚类算法分类

- 划分方法(partitioning method)
- 层次的方法(hierarchical method)
- 基于密度的方法(density-based method)
- 基于网格的方法(grid-based method)

划分聚类方法

👉 定义

- 将数据划分为k个簇，这些簇满足
  - 每个簇至少包含一个对象
  - 每一个对象属于且仅属于一个簇

👉 基本思想

- 对于给定的k，先给定一个初始划分方法
- **反复迭代**改变划分，使得改进后的划分方案比前一次更好

👉 常见的算法

- K-Means(K-均值)
- K-Medoids(K-中心点)

K-Means

👉 聚类目标函数：簇对象到簇中心**平方误差**最小

$$E=\sum_{i=1}^{k}\sum_{x\in C_i}|x-\overline{x_i}|^2$$ $\overline{x_i}$是第i个簇的均值 C_i代表第i个簇

K-Means算法步骤

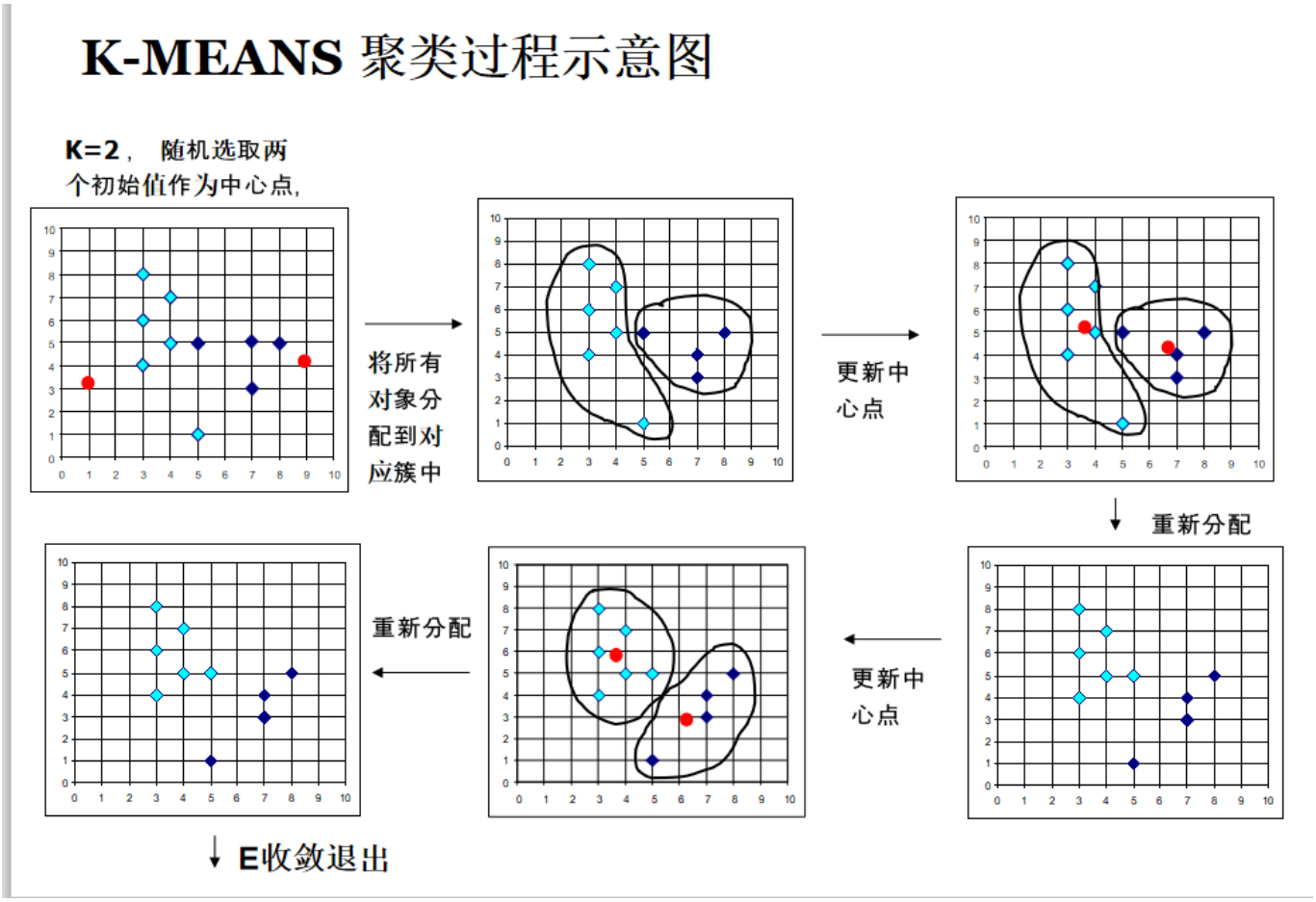
输入：簇的数目**k**和包含**n**个对象的数据集合。
输出：**k**个簇，使平方误差准则最小。
(1)assign initial values for means; /*任意分配到**k**个初始值作为簇的平均值*/
(2) REPEAT
(3) FOR *j*=1 to *n* DO assign each x_j to the closest clusters;
(4) FOR *i*=1 to *k* DO /*更新簇平均值*/

$$\overline{x_i}=\frac{1}{|C_i|}\sum_{x\in C_i}x$$

(5) Compute /*计算准则函数**E**/

$$E=\sum_{i=1}^{k}\sum_{x\in C_i}\left|x-\overline{x_i}\right|^2$$

(6) UNTIL **E**不发生变化。



K-Means评价

👉 优点

- 简单、快速
- 处理大数据集，相对可伸缩、高效

👉 缺点

- K-Means对于**球状分布数据**能取得较好结果
- **初值敏感**，初始点选取不同，结果会不同
- **噪声点(Outlier)敏感**，噪声点会严重影响K-Means的性能
  - K-中心点可以解决这个问题
  - 但K-中心点时间复杂度太高，用的少
- 必须事先给出k(parameter-free clustering algorithms)

👉 解决办法

- 先设置k值较大——大簇变小簇
  - 可以近似球状
  - 每个小簇的纯度很高

层次聚类方法

👉 对数据集进行层次的分解

👉 使用**距离**作为合并或分裂的标准，常见的距离度量

距离度量

$$\text{Minimum distance:} \quad dist_{min}(C_i, C_j) = \min_{p \in C_i, p' \in C_j} \{|p - p'|\}$$

$$\text{Maximum distance:} \quad dist_{max}(C_i, C_j) = \max_{p \in C_i, p' \in C_j} \{|p - p'|\}$$

$$\text{Mean distance:} \quad dist_{mean}(C_i, C_j) = |m_i - m_j|$$

$$\text{Average distance:} \quad dist_{avg}(C_i, C_j) = \frac{1}{n_i n_j} \sum_{p \in C_i, p' \in C_j} |p - p'|$$

- 用的最广的
  - Minimum distance——称为"Single Link"
  - Average distance——称为"Complete Link"

层次凝聚——AGNES算法

👉 **自底向上**的策略(像构造**哈夫曼树**的思想)，首先将每个对象作为一个簇，然后合并这些簇为更大的簇，相似度计算使用Minimum distance(Single Link)

AGNES算法步骤

算法 **AGNES**（自底向上凝聚算法）
输入：包含**n**个对象的数据库，终止条件簇的数目**k**。
输出：**k**个簇，达到终止条件规定簇数目。
(1) 将每个对象当成一个初始簇；
(2) REPEAT
(3) 根据两个簇中最近的数据点找到最近的两个簇；
(4) 合并两个簇，生成新的簇的集合；
(5) UNTIL 达到定义的簇的数目；

层次分裂——DIANA算法

👉 **自顶向下**的策略(像DT)，首先将所有对象置于一个簇中，逐渐细分为更小的簇

👉 使用两种测度方法

- 簇的直径
  - 在一个簇中的任意两个数据点的距离中的最大值
- Average distance

DIANA算法步骤

算法 DIANA（自顶向下分裂算法）
输入：包含n个对象的数据库，终止条件簇的数目k。
输出：k个簇，达到终止条件规定簇数目。
（1）将所有对象整个当成一个初始簇；
（2）FOR （i=1; i≠k; i++) DO BEGIN
（3）在所有簇中挑出具有最大直径的簇C；
（4）找出C中与其它点平均相异度最大的一个点p并把p放入splinter group，剩余的放在old party中；
（5）. REPEAT
（6）在old party里找出到最近的splinter group中的点的距离不大于到old party中最近点的距离的点，并将该点加入splinter group。
（7）UNTIL 没有新的old party的点被分配给splinter group；
（8）splinter group和old party为被选中的簇分裂成的两个簇，与其它簇一起组成新的簇集合。
（9）END.

密度聚类方法

密度聚类思想

👉 只要一个区域中的点的密度大于某个阈值，就将其加到与之相近的聚类中

- 可以克服基于距离算法**只能发现类圆形**的聚类的特点，可以发现任何形状的聚类，且对噪声不敏感

DBSCAN

DBSCAN基本定义

- 对象的,ε,-邻域
  - 给定对象在半径,ε,内的区域
- 核心对象
  - 若一个对象的,ε,-邻域至少包含最小数目$MinPts$个对象，称该对象为核心对象

- 直接密度可达
  - 若p在q的$,\varepsilon,$-邻域内，而q是一个核心对象，则称p从q出发时直接密度可达的
- 密度可达(间接密度可达)
  - 存在一个对象链$p_1,p_2,\cdots,p_n,p_1=q,p_n=p$,对$p_i\in D$， ,p_{i+1},是从,p_i,直接密度可达的，则称对象p是从对象q密度可达的

例如，在图中，**ε=1cm，MinPts=5，q**是一个核心对象，**p1**是从**q**关于**ε**和**MitPts**直接密度可达，**p**是从p_1关于**ε**和**MitPts**直接密度可达，则对象**p**从对象**q**关于**ε**和**MinPts**密度可达的 （间接密度可达）。

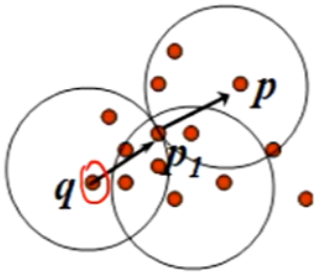


图 密度可达

- 密度相连
  - 存在一个对象o，使得对象p和q时关于o密度可达的，则称对象p和q密度相连

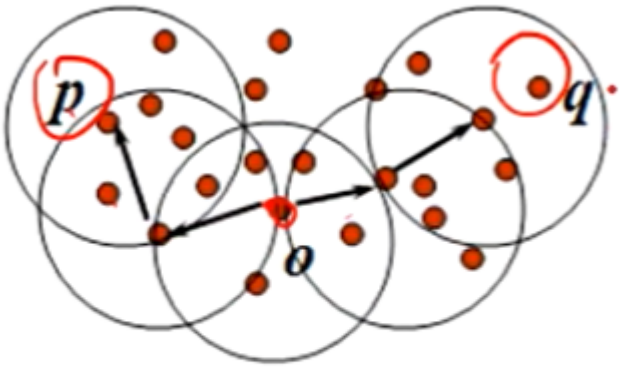


图 密度相连

  -
- 噪声
  - 一个基于密度的簇是基于**密度可达性**的**最大的密度相连对象的集合**

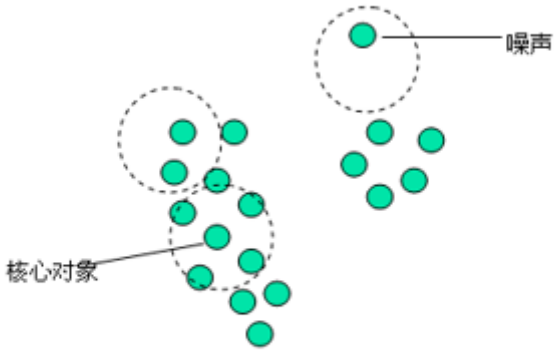


图 噪声

- 不包含任何簇中的对象被认为是"噪声"

DBSCAN基本思想

- 对于一个类中的每个对象，在其给定半径的邻域中包含的对象不能少于某一给定的最小数目
- 一个类能够被其中的任意一个核心对象所确定

DBSCAN算法步骤

DBSCAN算法描述
算法5-5 DBSCAN
输入：包含n个对象的数据库，半径 ε，最少数目MinPts。
输出：所有生成的簇，达到密度要求。
1. REPEAT
2. 从数据库中抽取一个未处理过的点；
3. IF 抽出的点是核心点 THEN找出所有从该点密度可达的对象，形成一个簇
4. ELSE 抽出的点是边缘点(非核心对象)，跳出本次循环，寻找下一点；
5. UNTIL 所有点都被处理；

DBSCAN缺点

- 调参难
  - 对,ε和$MinPts$,敏感，但这两个参数的选取主要依靠主观判断。$MinPts$取$4 \sim 20$
- 数据库较大时要进行较大的IO开销
- 不同密度簇问题
  - 若不同簇的密度不同，很可能把密度稀疏的簇识别为噪声

网格聚类方法

👉 本质：密度聚类

👉 基本思想

- 将特征空间划分为网格结构

👉 优缺

- 优点
  - 处理速度很快
- 缺点
  - 难以处理高维数据

STING算法

离群点检测

👉 离群点类型

- 全局离群点
- 局部离群点
- 集体离群点

👉 离群点分析方法

- 统计学方法
- 基于距离的方法
- 基于偏差的方法
- 基于密度的方法

基于统计学的孤立点检测

👉 基本思想：对给定的数据集合假设一个分布(或概率模型)，根据模型采用**不一致性检验**(假设检验)来确定孤立点

👉 优点

- 直观
- 可解释

👉 缺点

- 大多数假设检验是针对单个属性的，而数据挖掘时要求在多维空间发现孤立点
- 数据分布未知

基于距离的孤立点检测

👉 定义 ,$DB(p,d)$,——孤立点使得数据集中的对象至少有p部分与o的距离大于d

基于索引的算法

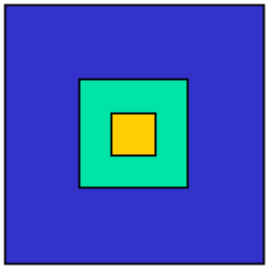
👉 采用多维索引的结构(一般是树)，查找每个对象在半径d范围内的邻居

👉 设M是一个孤立点的d邻域内的最大对象数目，一旦某个对象的$M+1$个邻居被发现，则该对象不是孤立点

嵌套循环算法

👉 两层循环计算距离，若邻域中的计数超过阈值，则不是离群点

基于单元(cell-based)的算法



👉 数据空间被划分为单元。对于一个给定的单元，累计三个计数——单元中对象的数目，单元和第一层中对象的数目，单元和两个层次中对象的数目

👉 设M是一个孤立点的d邻域中可能存在的孤立点的最大数目

- 若单元和第一层对象的数目和大于M，那么该单元中所有的对象不是孤立点
- 若单元和两个层中的对象数目和小于等于M，则单元中所有的对象被认为是孤立点
- 否则，对单元中的每个对象o，逐一判断

基于偏离的离群点检测

👉 基本思想：检查一组对象的主要特征来确定离群点，与特征偏离大的对象被认为离群点

👉 **平滑因子**：一个为序列中的每个子集计算的函数

- 估算从原始数据集中移走子集合可以带来的**相异度**的降低程度
- 平滑因子值最大的子集是异常集

基于密度的离群点检测

👉 **注意 下面所说的k是一个参数**

👉 对象p的$k-distance$,为p与某个对象o的距离

至少存在k个对象$o^{'}\in D/p$使得$d(p,o^{'})\leq d(p,o)$ 至多存在$k-1$个对象$o^{'}\in D/p$使得$d(p,o^{'})<d(p,o)$

👉 对象p的k-距离邻域：给定p的k-距离，邻域包含所有与p距离不超过k-距离的对象

$$N_{k-distance}(p)=q|d(p,q)\leq k-distance(p)$$

👉 对象p相对于对象o的可达距离 即在p的k-距离邻域中的点的可达距离都视为k-距离

$$reach-dist_k(p,o)=maxk-distance(p),d(p,o)$$

👉 对象p的局部可达密度(Local Reachable Density,LRD)：对象p的$MinPts$邻域中**所有对象的平均可达距离**的倒数

$$lrd_{MinPts}(p) = 1 \Big/ \left(\frac{\sum_{o \in N_{MinPts}(p)} reach\text{-}dist_{MinPts}(p,o)}{|N_{MinPts}(p)|} \right)$$

👉 对象p的局部异常因子(Local Outlier Factor):对象p的$MinPts$邻域中所有对象的LRD与对象p的LRD比值求和平均

$$LOF_{MinPts}(p) = \frac{\sum_{o \in N_{MinPts}(p)} \frac{lrd_{MinPts}(o)}{lrd_{MinPts}(p)}}{|N_{MinPts}(p)|}$$

- 局部异常因子越大，则该对象越异常
- 需要注意的是簇内靠近核心点的对象的LOF接近于1，不能认为是局部异常。异常点的LOF值**相对较大**

👉 LOF计算

基于同步的方法