

---


# 第10章 聚类分析：基本概念和方法

丁钰

南京农业大学人工智能学院

# 第10章：聚类分析：基本概念和方法

---

- 聚类分析 
- 划分方法
- 层次方法
- 基于密度的方法
- 基于网格的方法
- 聚类评估
- 小结

# 什么是聚类分析?

- 聚类: 数据对象的集合/簇 (cluster)
  - 同一簇中的对象彼此相似
  - 不同簇中的对象彼此相异
- 聚类分析
  - 将数据对象分组成为多个类或簇
- 聚类是**无监督的**分类: 没有预先定义的类
- 典型应用
  - 作为洞察数据内部分布的独一无二的工具
  - 作为其它算法的预处理步骤

# 聚类的一般应用

---

- 模式识别
- 空间数据分析
  - 聚类产生GIS(地理信息系统)的专题地图thematic maps
  - 在空间数据挖掘中检测空间聚类并解释它们
- 图像处理
- 经济学 (特别是市场研究)
- WWW
  - 文本分类
  - Web日志数据聚类, 发现类似访问模式群

# 聚类应用的例子

- 市场营销:

帮助市场营销者发现他们的基本顾客的不同组群，然后利用这一知识制定有针对性的营销计划

- 国土利用

在地球观测数据库中识别类似的国土使用区域

- 保险

对汽车保险持有者的分组

- 城市规划

根据房子的类型，价值，和地理位置对一个城市中房屋的分组

- 地震研究

应当将观测到的地震震中沿大陆板块断裂进行聚类

# 聚类分析的主要步骤

---

- 特征选择
  - 选择与任务密切相关的信息
  - 尽可能减少信息冗余
- 相似度评价
  - 两个特征向量的相似性
- 聚类的评价准则
  - 通过代价函数或某些规则
- 聚类算法
  - k-均值、极大似然、...
- 结果验证
  - 验证聚类结果的有效性
- 结果解释
  - 根据实际应用解释聚类结果

# 什么是好的聚类方法？

---

- 一个好的聚类方法应当产生高质量的聚类
  - 类内相似性高
  - 类间相似性低
- 聚类结果的质量依赖于方法所使用的相似性度量和它的实现.
- 聚类方法的质量也用它发现某些或全部隐藏的模式的能力来度量

# 数据挖掘对聚类的要求

- 可伸缩性
  - 有的算法当数据对象少于200时处理很好,但对大量数据对象偏差较大
  - 大型数据库包含数百万个对象
- 处理不同属性类型的能力
  - 许多算法专门用于数值类型的数据
  - 实际应用涉及不同数据类型 (数值和分类数据混合)
- 发现任意形状的聚类
  - 基于距离的聚类趋向于发现具有相近尺度和密度的球状簇
  - 一个簇可能是任意形状的



# 数据挖掘对聚类的要求(续)

- 用于决定输入参数的领域知识最小化
  - 许多聚类算法要求用户输入一定的参数，如希望产生的簇的数目。
  - 参数难以确定，增加用户负担，使聚类质量难以控制
- 处理噪声数据和孤立点的能力
  - 一些聚类算法对于噪音数据敏感，可能导致低质量的聚类结果
  - 现实世界中的数据库大都包含了孤立点, 空缺, 或者错误的
- 对于输入记录的顺序不敏感
  - 一些聚类算法对于输入数据的顺序是敏感的, 以不同的次序输入会导致不同的聚类

# 数据挖掘对聚类的要求(续)

- 高维性 (high dimensionality)
  - 许多聚类算法擅长处理低维的数据, 可能只涉及两到三维
  - 数据库或者数据仓库可能包含若干维或者属性, 数据可能非常稀疏, 而且高度偏斜
- 整合用户指定的约束
  - 现实世界的应用可能需要在各种约束条件下进行聚类
  - 要找到既满足特定的约束, 又具有良好聚类特性的数据分组是一项具有挑战性的任务
- 可解释性和可用性
  - 用户希望聚类结果是可解释的, 可理解的, 和可用的
  - 聚类可能需要和特定的语义解释和应用相联系

# 聚类分析的方法


- 划分方法:
  - 构建各种分区, 然后通过某种标准评估它们, e.g., minimizing the sum of square errors
  - Typical methods: k-means, k-medoids, CLARANS
- 层次方法:
  - 使用某些条件创建数据集 (或对象) 的分层或分解
  - Typical methods: Diana, Agnes, BIRCH, CAMELEON
- 基于密度的方法:
  - 基于对象之间的距离和密度函数
  - Typical methods: DBSACN, OPTICS, DenClue
- 基于网格的方法:
  - 把对象空间量化为有限个单元, 形成一个网格结构
  - Typical methods: STING, WaveCluster, CLIQUE

# 聚类分析的方法

- 基于模型的方法：
  - 为每个聚类假设一个模型，并尝试找到该模型彼此的最佳拟合
  - **Typical methods: EM, SOM, COBWEB**
- 基于频繁模式的方法：
  - **Based on the analysis of frequent patterns**
  - **Typical methods: p-Cluster**
- 基于约束的方法：
  - **Clustering by considering user-specified or application-specific constraints**
  - **Typical methods: COD (obstacles), constrained clustering**
- 基于链接的方法：
  - **Objects are often linked together in various ways**
  - **Massive links can be used to cluster objects: SimRank, LinkClus**

# 第10章：聚类分析：基本概念和方法

---

- 聚类分析
- 划分方法 
- 层次方法
- 基于密度的方法
- 基于网格的方法
- 聚类评估
- 小结

# 划分方法

- 划分方法: 构造 $n$ 个对象数据库 $D$ 的划分, 将其划分成 $k$ 个聚类
- 给定  $k$ , 找 $k$  个 *clusters* 对于选定的划分标准它是最优的
  - 全局最优(Global optimal): 枚举所有的划分
  - 启发式方法(Heuristic methods):  $k$ -平均( $k$ -means)和 $k$ -中心点( $k$ -medoids)算法
  - $k$ -平均(MacQueen'67): 每个簇用簇的重心(簇的平均值)表示
  - $k$ -中心点或PAM (Partition around medoids) (Kaufman & Rousseeuw'87): 每个簇用接近聚类中心的一个对象来表示

# $k$ -平均聚类算法

- 算法:  $k$ -平均

- (1) 任意选择 $k$ 个对象作为初始的簇中心;
- (2) repeat
- (3) 根据簇中对象的平均值, 将每个对象(重新)赋给最类似的簇;
- (4) 更新簇的平均值, 即重新计算每个簇中对象的平均值;
- (5) until 不再发生变化

- 通常, 采用平方误差准则作为收敛函数, 其定义如下

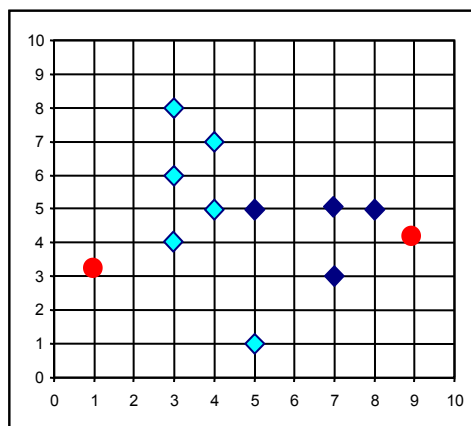
$$E = \sum_{i=1}^k \sum_{p \in C_i} |p - m_i|^2$$

其中,  $m_i$ 是簇 $C_i$ 的平均值

该准则试图使生成的结果簇尽可能紧凑, 独立

# $k$ -平均聚类算法(续)

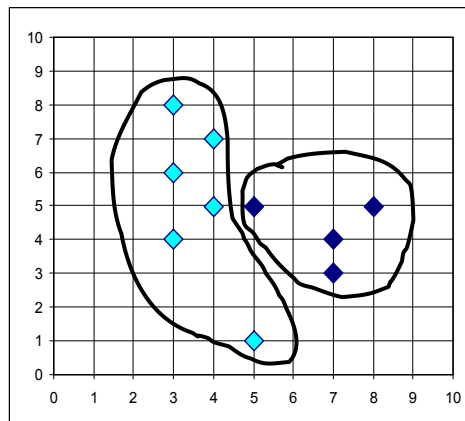
## ■ 例



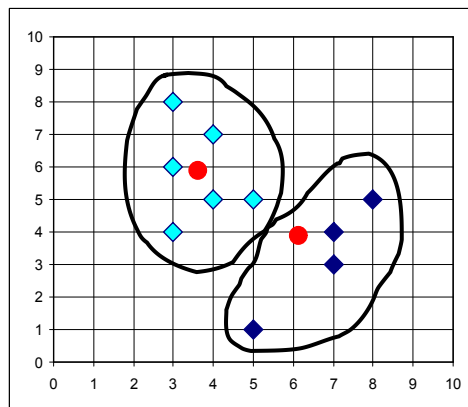
$K=2$

任意选择  $K$  个对象作为初始聚类中心

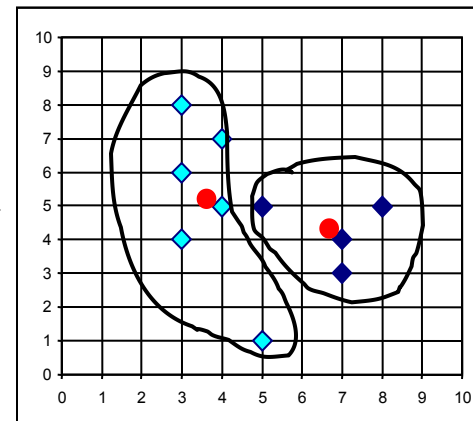
将每个对象赋给最相似中心



重新赋值

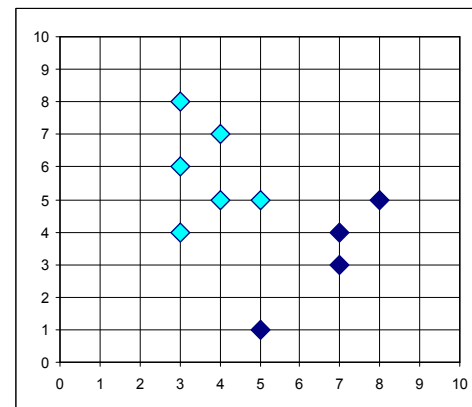


更新簇平均值



重新赋值

更新簇平均值





# $k$ -平均聚类算法(续)

- 优点: 相对有效性:  $O(tkn)$ ,  
其中,  $n$ 是对象数目,  $k$ 是簇数目,  $t$ 是迭代次数;  
通常:  $k, t \ll n$ 
  - 比较: PAM:  $O(k(n-k)^2)$ , CLARA:  $O(ks^2 + k(n-k))$
- 当结果簇是密集的, 簇与簇之间区别明显, 效果较好
- Comment: 常常终止于局部最优
- 全局最优可以使用诸如确定性的退火 (deterministic annealing) 和遗传算法 (genetic algorithms) 等技术得到

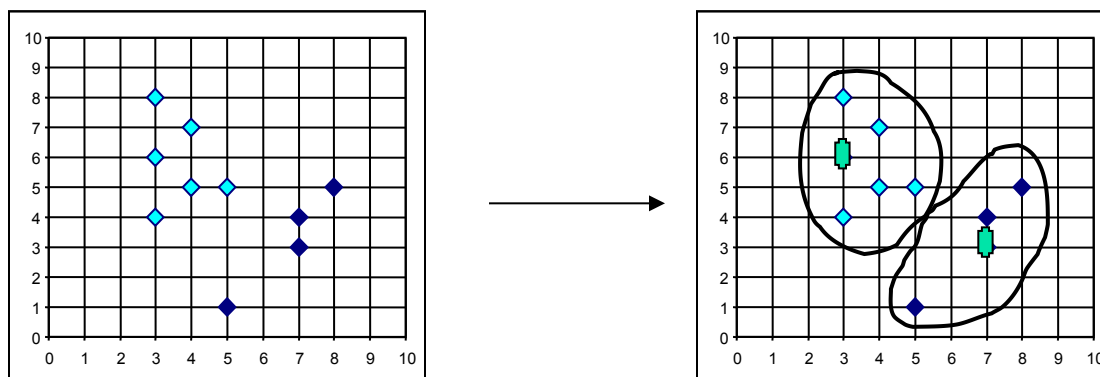
# $k$ -平均聚类算法(续)

## ■ 弱点

- 只有在簇的平均值(*mean*)被定义的情况下才能使用
- 可能不适用于某些应用，例如涉及有分类属性的数据
- 需要预先指定簇的数目  $k$
- 不能处理噪音数据和孤立点(*outliers*)
- 不适合用来发现非凸形状(*non-convex shapes*)的簇

# $k$ -中心点聚类方法

- $k$ -平均值算法对孤立点很敏感
  - 因为具有特别大的值的对象可能显著地影响数据的分布  
 $\{1,2,3,8,9,10,25\} \rightarrow \{1,2,3\} \{8,9,10,25\} \quad \{1,2,3,8\} \{9,10,25\}$
- $k$ -中心点( $k$ -Medoids): 不采用簇中对象的平均值作为参照点, 而是选用簇中位置最中心的对象, 即中心点(medoid)作为参照点



# $k$ -中心点聚类方法(续)

- 找聚类中的代表对象(中心点)
- ***PAM*** (**Partitioning Around Medoids, 1987**)
  - 首先为每个簇随意选择一个代表对象, 剩余的对象根据其  
与代表对象的距离分配给最近的一个簇
  - 反复地用非代表对象替代代表对象, 以改进聚类的质量
  - *PAM* 对于较小的数据集非常有效, 但不能很好地扩展到  
大型数据集
- ***CLARA*** (**Kaufmann & Rousseeuw, 1990**)抽样
- ***CLARANS*** (**Ng & Han, 1994**): 随机选样

# $k$ -中心点聚类方法(续)

## ■ 算法: $k$ -中心点

(1) 随机选择 $k$ 个对象作为初始的代表对象;

(2) repeat

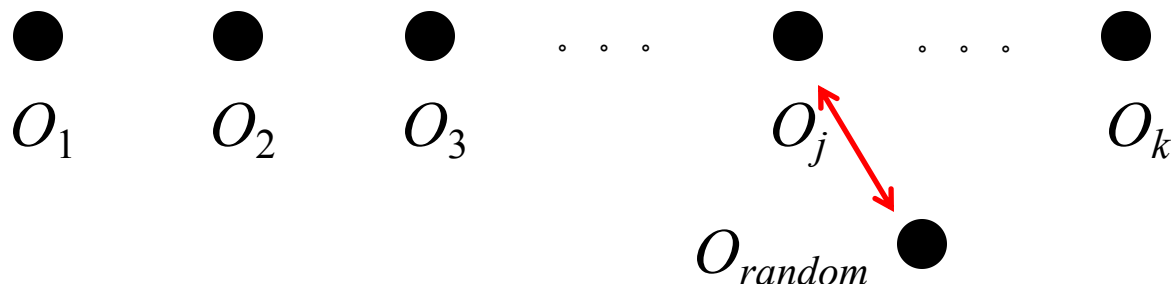
(3) 指派每个剩余的对象给离它最近的代表对象所代表的簇;

(4) 随意地选择一个非代表对象 $O_{\text{random}}$ ;

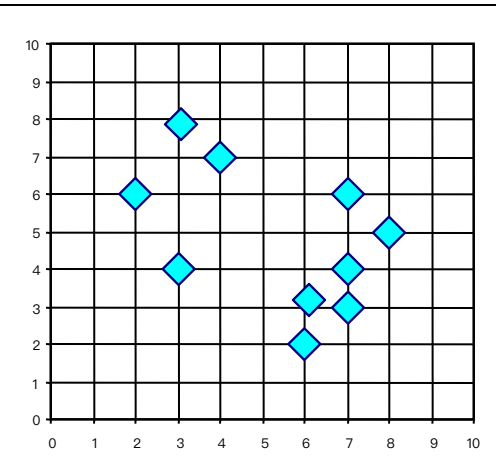
(5) 计算用 $O_{\text{random}}$ 代替 $O_j$ 的总代价 $S$ ;

(6) 若 $S < 0$ , 用 $O_{\text{random}}$ 替换 $O_j$ , 形成新的 $k$ 个代表对象的集合;

(7) until 不发生变化



# k-中心点聚类方法(续)

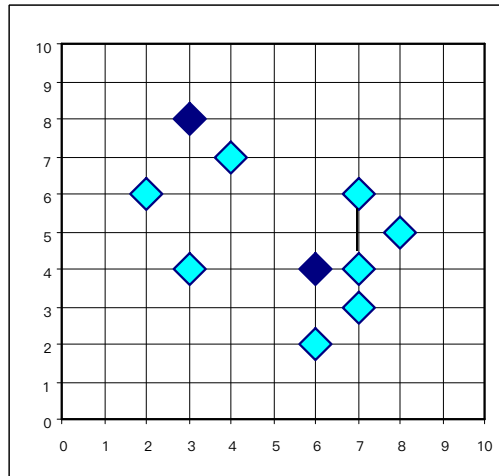


任意选择  $k$  个数据对象作为类中心点

$k=2$

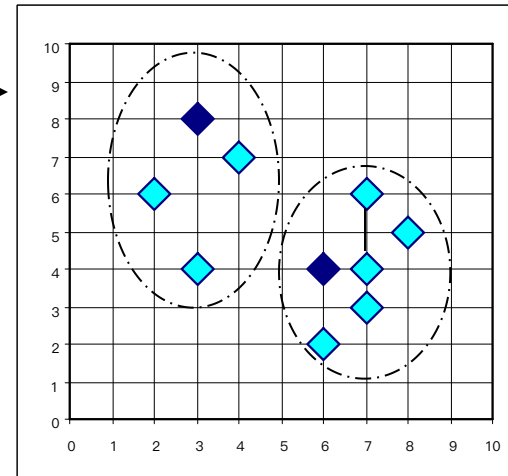
执行循环直至类不再发生变化

若聚类质量上升, 交换  $O$  与  $O_{\text{random}}$ .



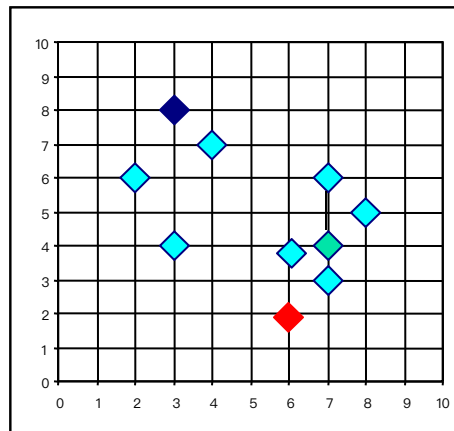
将每个数据对象安排到最近的类

Total Cost = 20

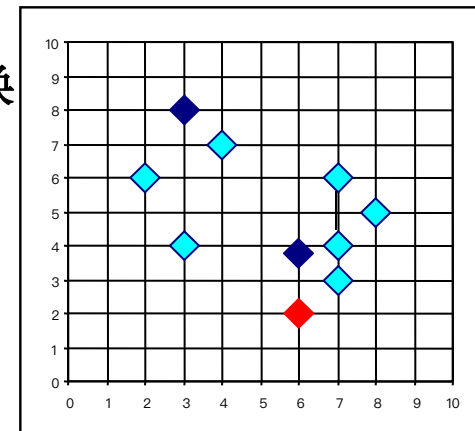


随机选择一个非类中心数据对象  $O_{\text{random}}$

Total Cost = 26



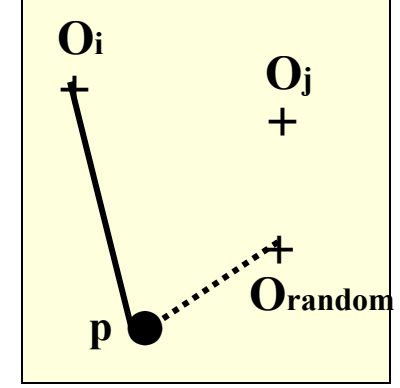
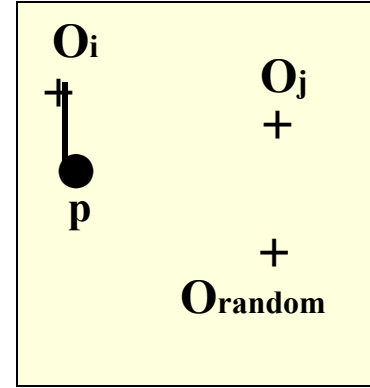
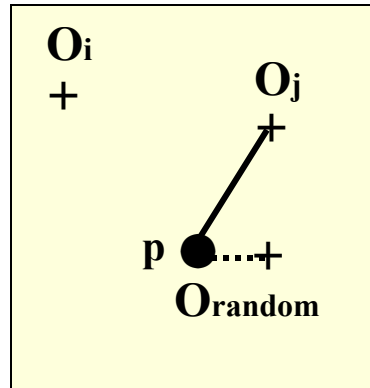
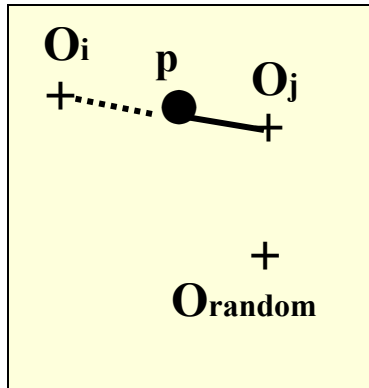
计算交换的代价



## $k$ -中心点聚类方法(续)

- 为了判定一个非代表对象 $O_{\text{random}}$ 是否是当前一个代表对象 $O_j$ 的好的替代, 对于每一个非代表对象, 考虑下面四种情况:
  - 第一种情况:  $p$ 当前隶属于代表对象 $O_j$ . 如果 $O_j$ 被 $O_{\text{random}}$ 所代替, 且 $p$ 离 $O_i$ 最近,  $i \neq j$ , 那么 $p$ 被重新分配给 $O_i$
  - 第二种情况:  $p$ 当前隶属于代表对象 $O_j$ . 如果 $O_j$ 被 $O_{\text{random}}$ 代替, 且 $p$ 离 $O_{\text{random}}$ 最近, 那么 $p$ 被重新分配给 $O_{\text{random}}$
  - 第三种情况:  $p$ 当前隶属于 $O_i$ ,  $i \neq j$ . 如果 $O_j$ 被 $O_{\text{random}}$ 代替, 而 $p$ 仍然离 $O_i$ 最近, 那么对象的隶属不发生变化
  - 第四种情况:  $p$ 当前隶属于 $O_i$ ,  $i \neq j$ . 如果 $O_j$ 被 $O_{\text{random}}$ 代替, 且 $p$ 离 $O_{\text{random}}$ 最近, 那么 $p$ 被重新分配给 $O_{\text{random}}$

# $k$ -中心点聚类方法(续)



1. 重新分配给  $O_i$     2. 重新分配给  $O_{\text{random}}$     3. 不发生变化    4. 重新分配给  $O_{\text{random}}$

● 数据对象    + 簇中心    — 替代前    ..... 替代后

$k$ -中心点聚类代价函数的四种情况



## $k$ -中心点聚类方法(续)

- 当存在噪音和孤立点时，PAM 比  $k$ -平均方法更健壮，其原因是中心点不象平均值那么容易被极端数据影响
- PAM对于小数据集工作得很好，但不能很好地用于大数据集

■ 每次迭代  $O(k(n-k)^2)$        $k$ -平均 :  $O(tkn)$


其中,  $n$  是数据对象数目,  $k$  是聚类数

→ 基于抽样的方法:

**CLARA(Clustering LARge Applications)**

# 第10章： 聚类分析： 基本概念和方法

---

- 聚类分析
- 划分方法
- 层次方法 
- 基于密度的方法
- 基于网格的方法
- 聚类评估
- 小结

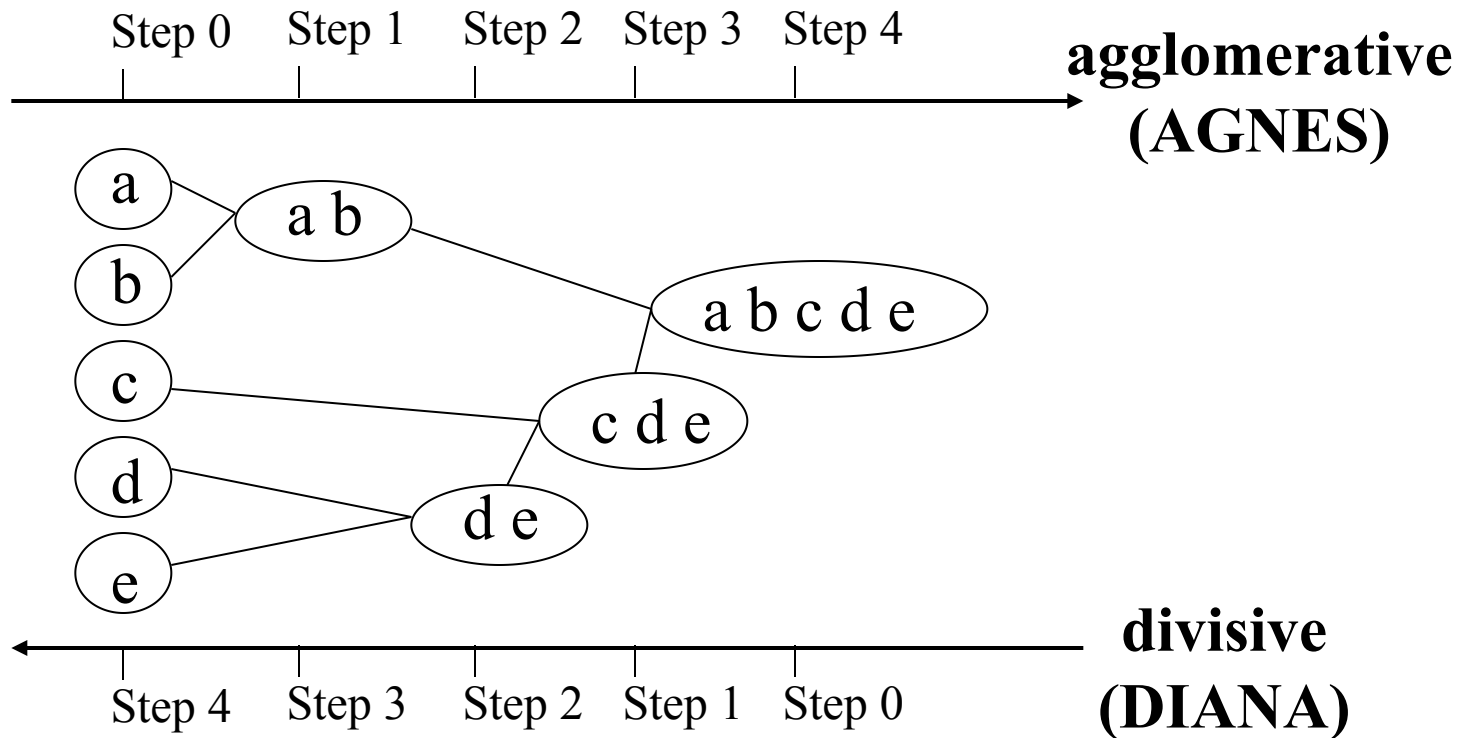
# 层次方法

---

- 层次的聚类方法将数据对象组成一棵聚类的树
- 根据层次分解是自底向上, 还是自顶向下形成, 层次的聚类方法可以进一步分为凝聚的(**agglomerative**)和分裂的(**divisive**)层次聚类
- 使用距离矩阵作为聚类标准. 该方法不需要输入聚类数目  $k$ , 但需要终止条件

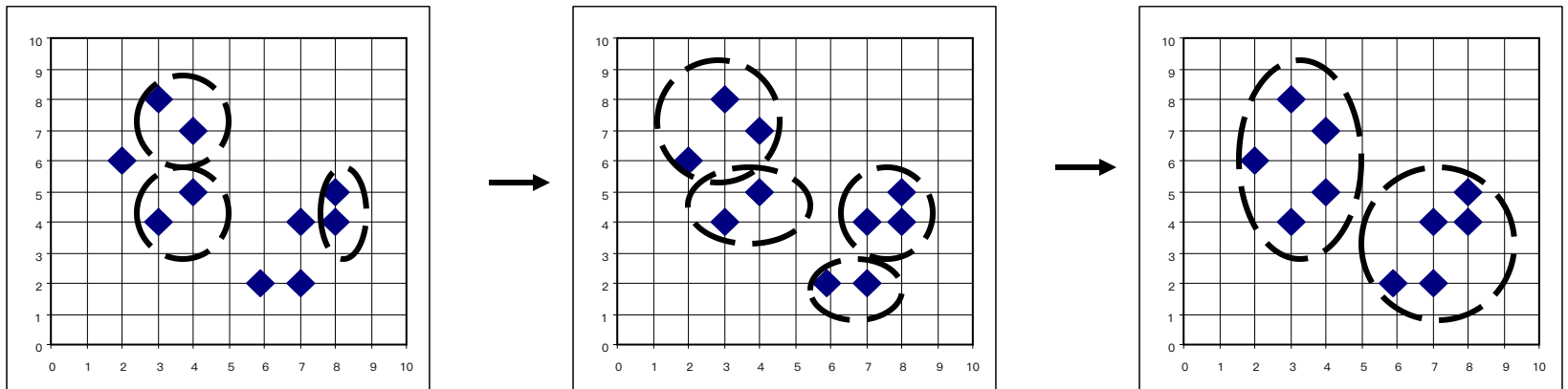
# 层次方法 (续)

- 凝聚的(agglomerative)和分裂的(divisive)层次聚类图示



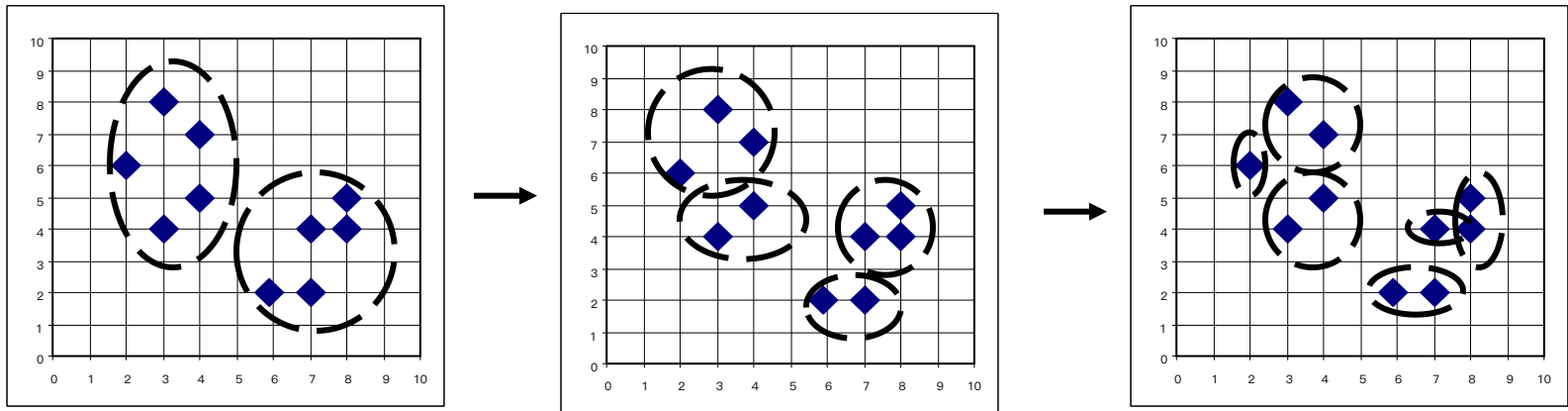
# AGNES (Agglomerative Nesting)

- 由 Kaufmann和Rousseeuw提出(1990)
- 已在一些统计分析软件包中实现 . 如 Splus
- 使用单链接(Single-Link)方法和相异度矩阵
- 合并具有最小相异度的节点
- 以非递减的方式继续
- 最终所有的节点属于同一个簇



# DIANA (Divisive Analysis)

- 由 Kaufmann和Rousseeuw提出 (1990)
- 已在一些统计分析软件包中实现 . 如 Splus
- 是 AGNES的逆
- 最终每个节点自己形成一个簇



# 层次方法(续)

## ■ 四个广泛采用的簇间距离度量方法

- 最小距离:  $d_{min}(C_i, C_j) = \min_{p \in C_i, p' \in C_j} |p-p'|$
- 最大距离:  $d_{max}(C_i, C_j) = \max_{p \in C_i, p' \in C_j} |p-p'|$
- 平均值的距离:  $d_{mean}(C_i, C_j) = |m_i - m_j|$
- 平均距离:  $d_{avg}(C_i, C_j) = \sum_{p \in C_i} \sum_{p' \in C_j} |p-p'| / n_i n_j$

其中:  $|p-p'|$ 是两个对象 $p$ 和 $p'$ 之间的距离;  $m_i$ 是簇 $C_i$ 的平均值,  $n_i$ 是簇 $C_i$ 中对象的数目

# 层次方法(续)

## ■ 层次聚类的主要缺点

- 不具有很好的可伸缩性: 时间复杂性至少是  $O(n^2)$ , 其中  $n$  对象总数
- 合并或分裂的决定需要检查和估算大量的对象或簇
- 不能撤消已做的处理, 聚类之间不能交换对象. 如果某一步没有很好地选择合并或分裂的决定, 可能会导致低质量的聚类结果



# 层次方法(续)

- 改进层次方法的聚类质量的方法：将层次聚类和其他的聚类技术进行集成，形成多阶段聚类
  - **BIRCH (1996)**: 使用 CF-tree对对象进行层次划分, 然后采用其他的聚类算法对聚类结果进行求精
  - **ROCK1999**: 基于簇间的互联性进行合并
  - **CHAMELEON (1999)**: 使用动态模型进行层次聚类
  - **CURE (1998)**: 采用固定数目的代表对象表示每个簇, 依据一个指定的收缩因子向着聚类中心对它们进行收缩

# BIRCH (1996)

- **Birch (Balanced Iterative Reducing and Clustering using Hierarchies):** 利用层次方法的平衡迭代归约和聚类
- 两个重要概念
  - 聚类特征(Clustering Feature, CF)
  - 聚类特征树(Clustering Feature Tree, CF树)
- 聚类特征
  - 聚类特征(CF)是一个三元组, 给出对象子类的信息的汇总描述
  - 设某个子类中有 $N$ 个 $d$ 维的点或对象 $\{o_i\}$ , 则该子类的CF定义如下  $CF = (N, LS, SS)$

# 聚类特征

**Clustering Feature:  $CF = (N, LS, SS)$**

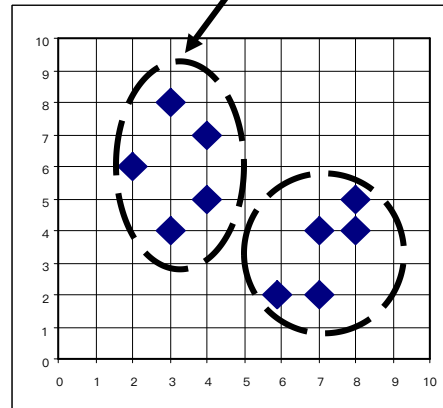
$N$ : 数据点数目

$LS$  线性:

$$\sum_{i=1}^N X_i$$

$SS$  平方和:

$$\sum_{i=1}^N X_i^2$$



$CF = (5, (16,30), (54,190))$

(3,4)

(2,6)

(4,5)

(4,7)

(3,8)

可加性 (类合并) :

$$CF = CF1 + CF2 = (N1 + N2, LS1 + LS2, SS1 + SS2)$$

# 聚类特征

$$\begin{aligned}\text{簇质心} \quad \vec{X}_0 &= \frac{\sum_{i=1}^N \vec{X}_i}{N} = \frac{LS}{N} \\ \text{簇半径} \quad R &= \sqrt{\frac{\sum_{i=1}^N (\vec{X}_i - \vec{X}_0)^2}{N}} = \sqrt{\frac{NSS - LS^2}{N^2}} \\ \text{簇直径} \quad D &= \sqrt{\frac{\sum_{i=1}^N \sum_{j=1}^N (\vec{X}_i - \vec{X}_j)^2}{N(N-1)}} = \sqrt{\frac{2NSS - 2LS^2}{N(N-1)}}\end{aligned}$$

簇半径 **R** 是成员对象到质心的平均距离；  
簇直径 **D** 是簇中两两数据点的平均距离；  
这两个统计量都反映了簇内紧密程度。  
因此我们说，**CF** 结构概括了簇的基本信息，并且是高度压缩的，它存储了小于实际数据点的聚类信息。

# BIRCH的CF树

## ■ 聚类特征

- 从统计学的观点来看，聚类特征是子聚类的0阶, 1阶和2阶矩( moments)
- 记录了计算聚类和有效利用存储的关键度量, 并有效地利用了存储，因为它汇总了关于子类的信息，而**不是**存储所有的对象

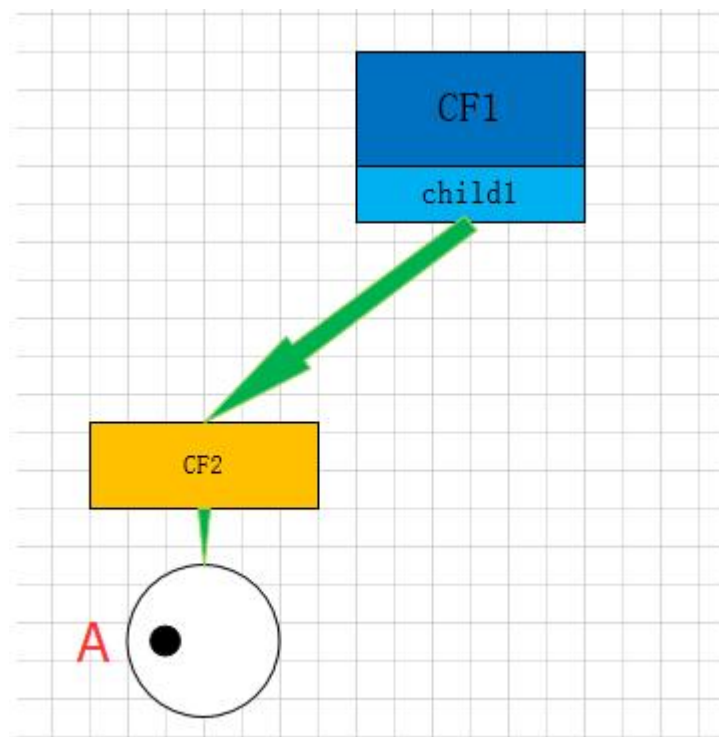
# BIRCH的CF树

- CF 树是高度平衡的树，它存储了层次聚类的**聚类特征**
  - 树中的非叶节点有后代或“孩子”
  - 非叶节点存储了其孩子的CF的总和，即汇总了关于其孩子的聚类信息
- CF树有两个参数——影响CF树的大小
  - 分支因子B: 定义非树叶节点的孩子的最大个数
  - 阈值T: 定义存储在树的叶子节点中的子类的最大直径

# BIRCH的CF树

最开始的 CF 树 是空的，没有任何样本，我们一边添加数据，一边进行学习。

首先，读入第一条数据 A，构造一个叶子节点和一个子簇。

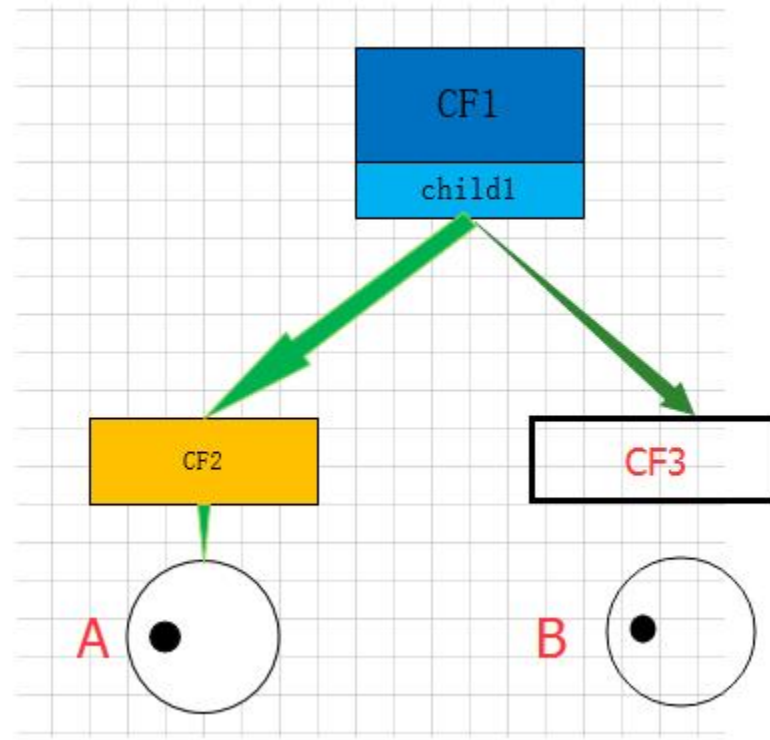
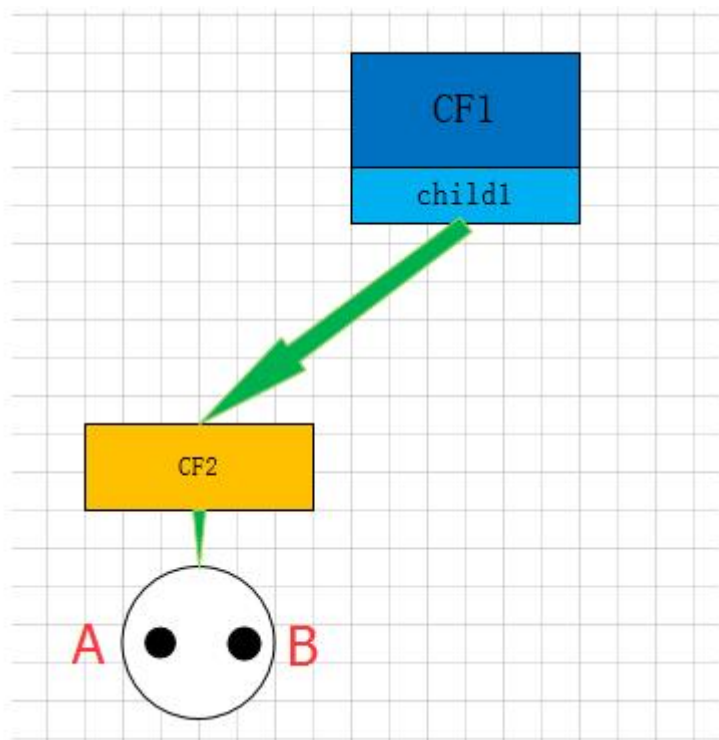


# BIRCH的CF树

当引入第二个样本B时，需要判断，样本A和样本B是否在半径为  $\tau$  的超球体范围内？

如果在，也就是说他们属于同一个CF，根据 CF 可加性，更新CF

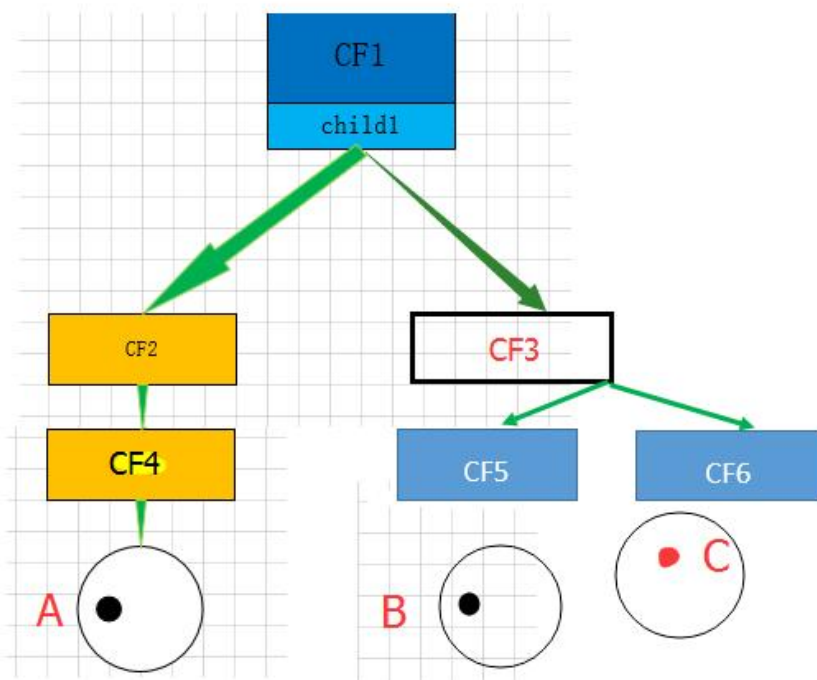
如果不在，我们需要一个新的CF三元组，来容纳这个新的值



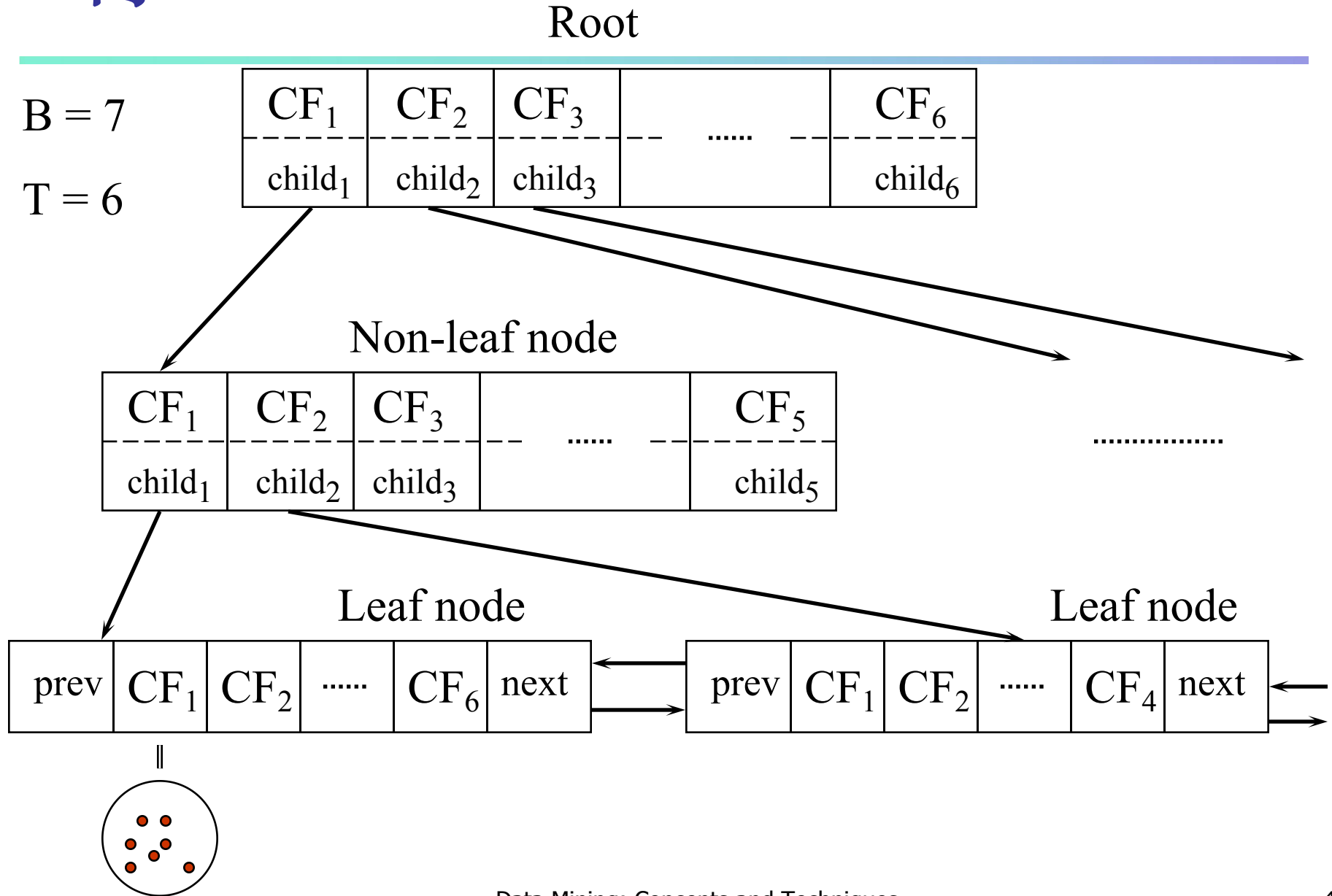


# BIRCH的CF树

假设我们设置枝平衡因子 $\beta=2$ ，即内部节点的分支不得多于2个。这时候来了一个点C，离CF3最近，但不属于A，B两个超球体范围内，本应该将它单独设一个分支，但是CF1的分枝数已经达到最大值，此时，我们只能选择继续向下分裂



# CF树



# BIRCH (续)

---

- BIRCH增量地构造一棵 CF 树(Clustering Feature Tree), CF 树是一个层次数据结构, 用于多阶段聚类
  - 阶段 1: 扫描 DB, 建立一棵初始的存放在内存的 CF树 (数据的多层压缩, 试图保留数据内在的聚类结构)
  - 阶段 2: 使用某种聚类算法对CF树的叶节点进行聚类

# BIRCH (续)

- 在阶段一：随着对象被插入，CF树被动态地构造
  - 一个对象被插入到最近的叶子条目（子聚类）
  - 如果在插入后存储在叶子节点中的子类的直径大于阈值，那么该叶子节点（可能还有其他节点）被分裂
  - 新对象插入后，关于该对象的信息向着树根传递
  - 通过修改阈值，CF树的大小可以改变
  - 树重建：从旧树的叶子节点建造一个新树，重建树的过程不需要重读所有的对象——建树只需读一次数据
- 阶段二：采用任何聚类算法对叶节点进行聚类，例如典型的划分方法

# BIRCH (续)

## ■ BIRCH的性能

- 支持增量聚类
- 线性可伸缩性: 计算复杂性 $O(n)$ , 单遍扫描, 附加的扫描可以改善聚类质量
- 较好的聚类质量

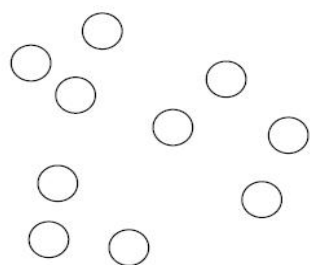
## ■ 缺点

- 只能处理数值数据
- 对数据的输入次序敏感
- Cf树结点不总是对应于[用户考虑的]自然簇(参数B和T)
- 簇非球形时效果不好(使用半径/直径控制簇边界)

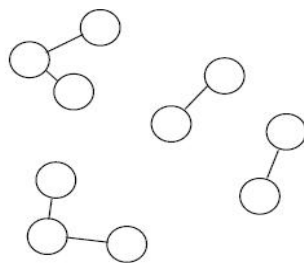
# CHAMELEON

- **CHAMELEON** :一个利用动态模型的层次聚类算法 (Hierarchical clustering using dynamic modeling) 由G. Karypis, E.H. Han, and V. Kumar'99 提出
- **CHAMELEON**基于动态模型度量相似性
  - 如果两个簇间的互连性和近似度与簇内部对象间的互连性和近似度高度相关, 则合并这两个簇
- **两阶段算法**
  - 使用图划分算法: 将数据对象聚为大量相对较小的子类
  - 使用凝聚的层次聚类算法: 通过反复地合并子类来找到真正的结果簇

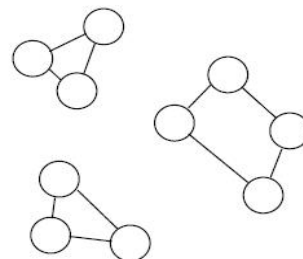
# CHAMELEON (续)



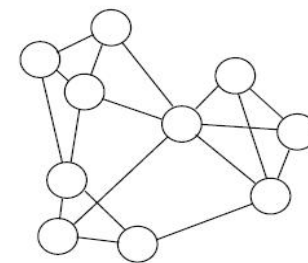
(a) Original Data in 2D



(b) 1-nearest neighbor graph



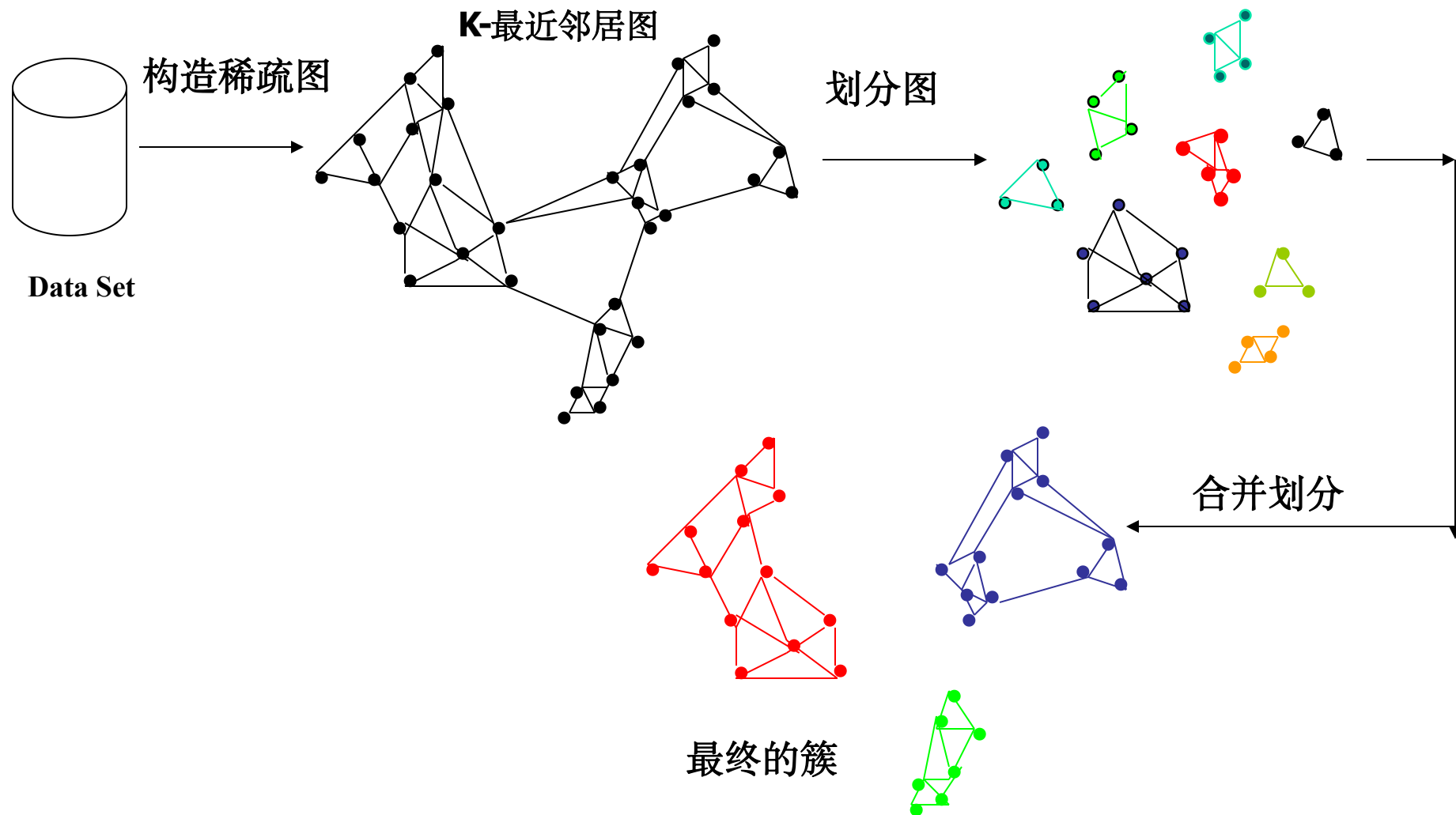
(c) 2-nearest neighbor graph



(d) 3-nearest neighbor graph

- 既考虑互连性, 又考虑簇间的近似度, 特别是簇内部的特征, 来确定最相似的子类.
- 这样, 它不依赖于静态的用户提供的模型, 能够自动地适应被合并的簇的内部特征
  - **割边最小化**——簇 $C$ 划分为两个子簇 $C_i$ 和 $C_j$ 时需要割断的边的加权和最小。
  - 割边用 $EC_{\{C_i, C_j\}}$ 表示, 评估 $C_i$ 和 $C_j$ 的簇间的**绝对互联性**。

# CHAMELEON (续)





# CHAMELEON (续)

- **k-最近邻图  $G_k$** : 图中的每个点代表一个数据对象, 如果一个对象是另一个对象的k个最类似的对象之一, 在这两个点之间存在一条边
- **k-最近邻图  $G_k$  动态地捕捉邻域的概念**: 一个对象的邻域半径由对象所在区域的密度所决定
  - 在一个密集区域, 邻域的定义范围相对狭窄; 在一个稀疏区域, 它的定义范围相对较宽
- **区域的密度作为边的权重被记录下来**
  - 一个密集区域的边趋向于比稀疏区域的边有更大的权重

# CHAMELEON (续)

- Chameleon通过两个簇的相对互连性 $RI(C_i, C_j)$ 和相对接近度 $RC(C_i, C_j)$ 来决定簇间的相似度
  - $RI(C_i, C_j)$ 定义为 $C_i$ 和 $C_j$ 之间的绝对互联性关于两个簇的内部互连性的规范化

$$RI(C_i, C_j) = \frac{|EC_{\{C_i, C_j\}}|}{\frac{1}{2}(|EC_{C_i}| + |EC_{C_j}|)}$$

其中,  $EC_{\{C_i, C_j\}}$ 是包含 $C_i$ 和 $C_j$ 的簇分裂为 $C_i$ 和 $C_j$ 的割边,  $EC_{C_i}$  (或 $EC_{C_j}$ ) 是它的最小截断等分线的大小 (即将图划分为两个大致相等的部分需要切断的边的加权和)

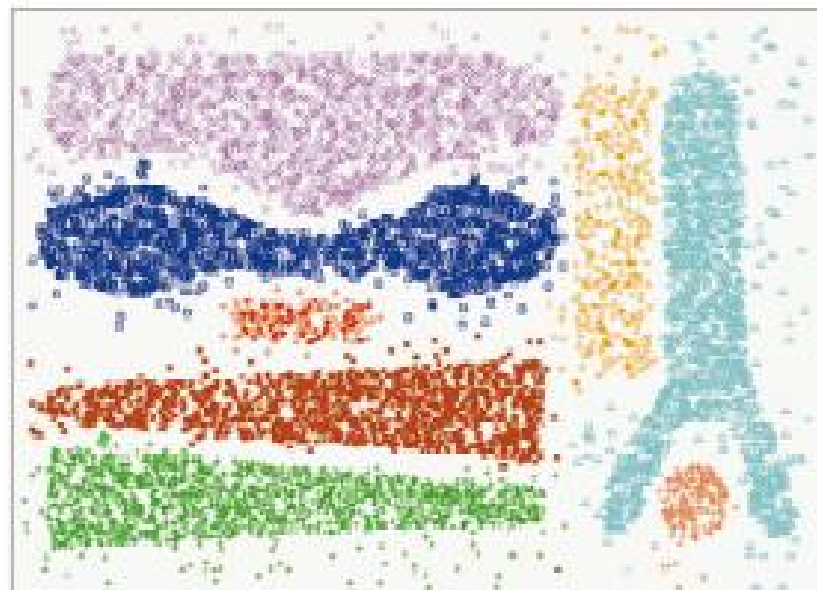
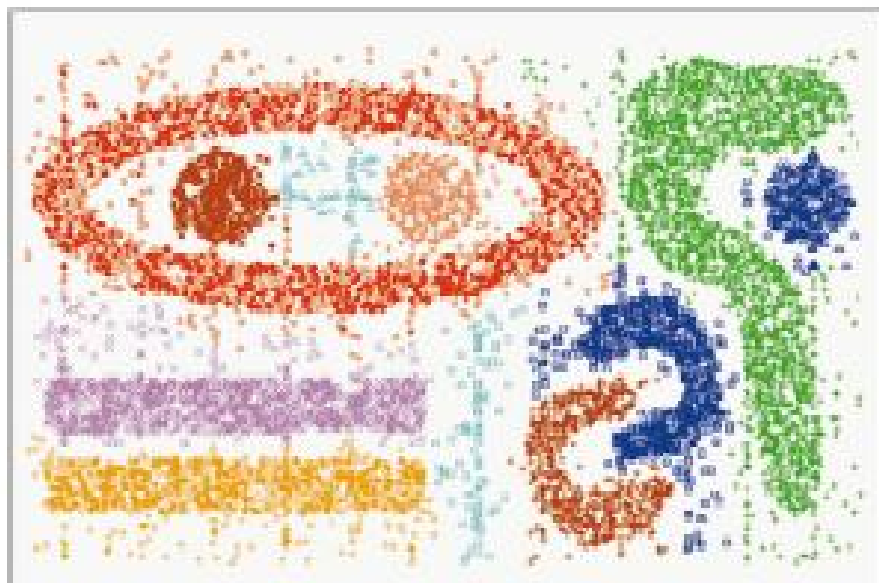
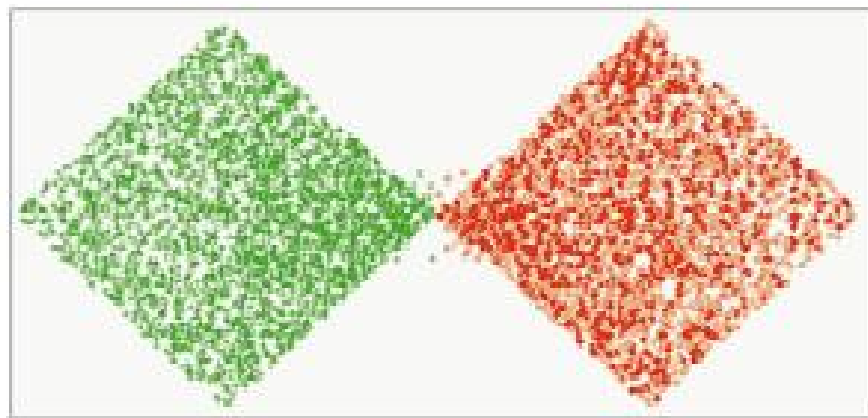
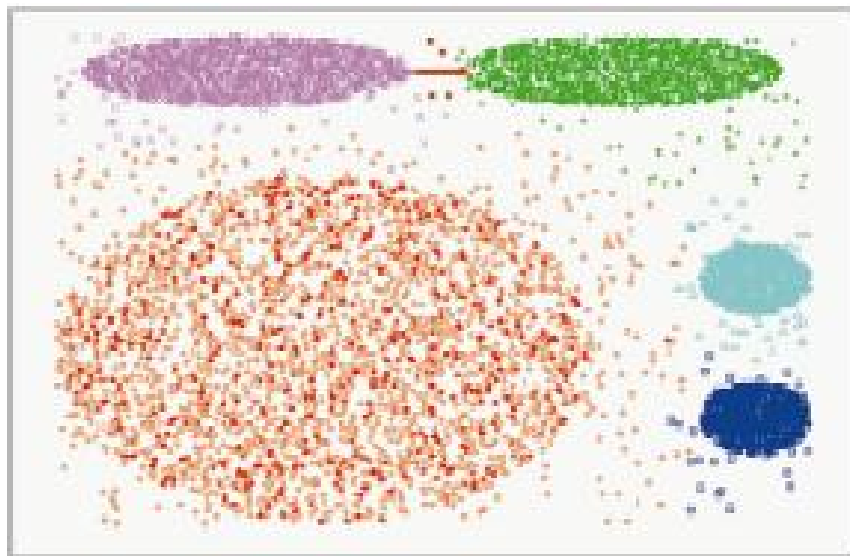
# CHAMELEON (续)

- $RC(C_i, C_j)$  定义为  $C_i$  和  $C_j$  之间的绝对接近度关于两个簇的内部接近度的规范化

$$RC(C_i, C_j) = \frac{\overline{S}_{EC\{C_i, C_j\}}}{\frac{|C_i|}{|C_i| + |C_j|} \overline{S}_{EC_{C_i}} + \frac{|C_j|}{|C_i| + |C_j|} \overline{S}_{EC_{C_j}}}$$

其中,  $\overline{S}_{EC(C_i, C_j)}$  是连接  $C_i$  和  $C_j$  顶点的边的平均权重  
 $\overline{S}_{EC_{C_i}}$  是  $C_i$  的最小二等分的边的平均权重

# CHAMELEON (续)



# 概率层次聚类

- 算法的层次聚类方法缺点
  - 为层次聚类选择一种好的距离量度常常是困难的
  - 为了使用算法的方法，数据对象不能有缺失的属性值
  - 大部分算法的层次聚类方法都是启发式的，因此结果聚类层次结构的优化目标不清晰
- 概率层次聚类
  - 利用概率模型表征聚类之间的距离

# 生成模型

- 给定1维数据点集 $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ , 假设其服从高斯分布

$$\mathcal{N}(\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- 那么 $\mathbf{x}_i \in \mathbf{X}$ 被该模型生成的概率是:

$$P(x_i|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}$$

- $\mathbf{X}$ 被该模型生成的似然为:

$$L(\mathcal{N}(\mu, \sigma^2) : \mathbf{X}) = P(\mathbf{X}|\mu, \sigma^2) = \prod_{i=1}^n \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}$$

- 学习生成模型的任务是: 找出使得似然最大的 $\mu$ 和 $\sigma^2$

$$\mathcal{N}(\mu_0, \sigma_0^2) = \arg \max \{L(\mathcal{N}(\mu, \sigma^2) : \mathbf{X})\}$$

# 概率层次聚类算法

- 若一个对象集和被分成 $m$ 个聚类 $C_1, \dots, C_m$ , 那么其聚类质量可用下式度量

$$Q(\{C_1, \dots, C_m\}) = \prod_{i=1}^m P(C_i)$$

其中 $P()$ 是针对某一类的最大似然


- 若合并其中的两类 $C_{j_1}$ 和 $C_{j_2}$ 成为 $C_{j_1} \cup C_{j_2}$ , 那么其聚类质量变化为

$$\begin{aligned} & Q((\{C_1, \dots, C_m\} - \{C_{j_1}, C_{j_2}\}) \cup \{C_{j_1} \cup C_{j_2}\}) - Q(\{C_1, \dots, C_m\}) \\ = & \frac{\prod_{i=1}^m P(C_i) \cdot P(C_{j_1} \cup C_{j_2})}{P(C_{j_1})P(C_{j_2})} - \prod_{i=1}^m P(C_i) \\ = & \prod_{i=1}^m P(C_i) \left( \frac{P(C_{j_1} \cup C_{j_2})}{P(C_{j_1})P(C_{j_2})} - 1 \right) \end{aligned}$$

- 定义 $C_1$ 和 $C_2$ 两类之间的距离  $dist(C_i, C_j) = -\log \frac{P(C_1 \cup C_2)}{P(C_1)P(C_2)}$

# 第10章：聚类分析：基本概念和方法

---

- 聚类分析
- 划分方法
- 层次方法
- 基于密度的方法 
- 基于网格的方法
- 聚类评估
- 小结



# 基于密度的方法

- 基于密度聚类 (**Density-Based Clustering**)
- 主要特点:
  - 发现任意形状的聚类
  - 处理噪音
  - 一遍扫描
  - 需要密度参数作为终止条件
- 一些有趣的研究:
  - **DBSCAN**: Ester, et al. (KDD'96)
  - **OPTICS**: Ankerst, et al (SIGMOD'99).
  - **DENCLUE**: Hinneburg & D. Keim (KDD'98)
  - **CLIQUE**: Agrawal, et al. (SIGMOD'98)

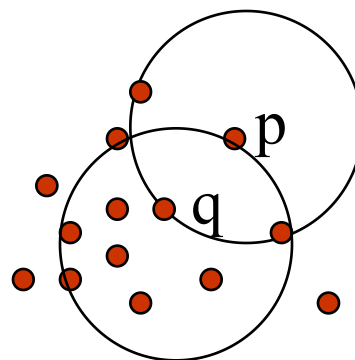
# 密度概念

- $\epsilon$ -邻域: 给定对象半径 $\epsilon$ 内的领域
- 核心对象 (core object) : 一个对象的 $\epsilon$ -邻域至少包含最小数目个对象 ( MinPts )
- **直接**密度可达的 (Directly density reachable, DDR) : 给定对象集合D, 如果p是在q的 $\epsilon$ -邻域内, 而q是核心对象, 我们说对象p是从对象q直接密度可达的
- 密度可达的(density reachable): 存在一个从p到q的DDR对象链
- 密度相连的(density-connected): 如果对象集合D中存在一个对象o, 使得对象p和q是从o关于 $\epsilon$ 和MinPts密度可达的, 那么对象p和q是关于 $\epsilon$ 和MinPts密度相连的

# 基于密度的聚类: 背景I

- 两个参数:
  - $\varepsilon$ : 邻域的最大半径
  - **MinPts**: 在  $\varepsilon$ -邻域中的最少点数
  - $N_\varepsilon(p)$ :  $\{q \text{ belongs to } D \mid \text{dist}(p,q) \leq \varepsilon\}$
- 直接密度可达的: 点  $p$  关于  $\varepsilon$ , **MinPts** 是从点  $q$  直接密度可达的, 如果
  - 1)  $p$  属于  $N_\varepsilon(p)$ :
  - 2) 核心点条件:

$$|N_{\varepsilon}(q)| \geq \text{MinPts}$$



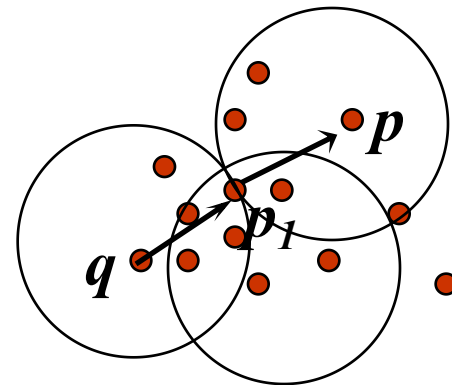
$$\text{MinPts} = 5$$

$$\varepsilon = 1 \text{ cm}$$

# 基于密度的聚类: 背景II

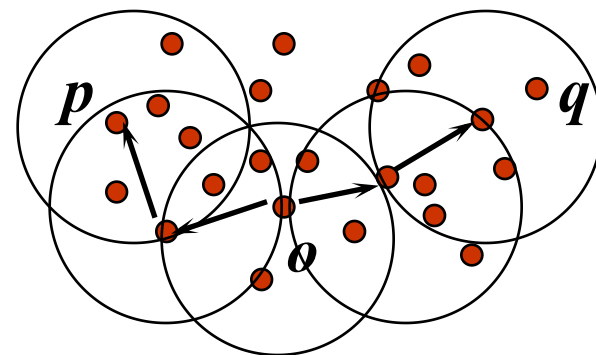
## ■ 密度可达:

- 点  $p$  关于  $\epsilon$ , MinPts 是从  $q$  密度可达的, 如果存在一个节点链  $p_1, \dots, p_n$ ,  $p_1 = q$ ,  $p_n = p$  使得  $p_{i+1}$  是从  $p_i$  直接密度可达的



## ■ 密度相连的:

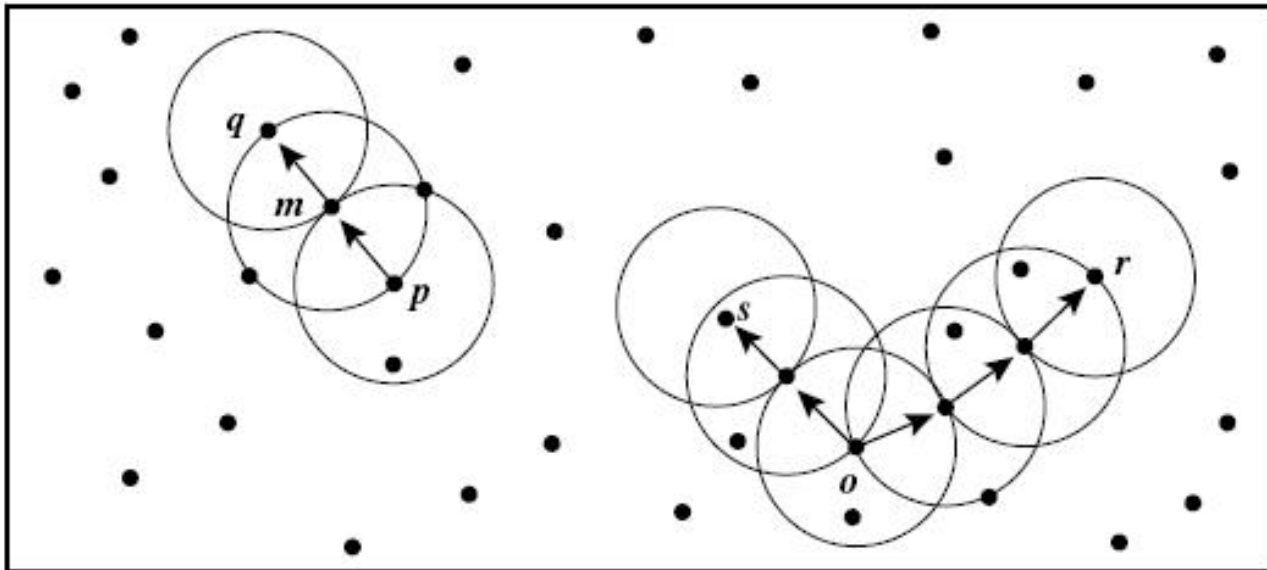
- 点  $p$  关于  $\epsilon$ , MinPts 与点  $q$  是密度相连的, 如果存在点  $o$  使得,  $p$  和  $q$  都是关于  $\epsilon$ , MinPts 是从  $o$  密度可达的



# 例子

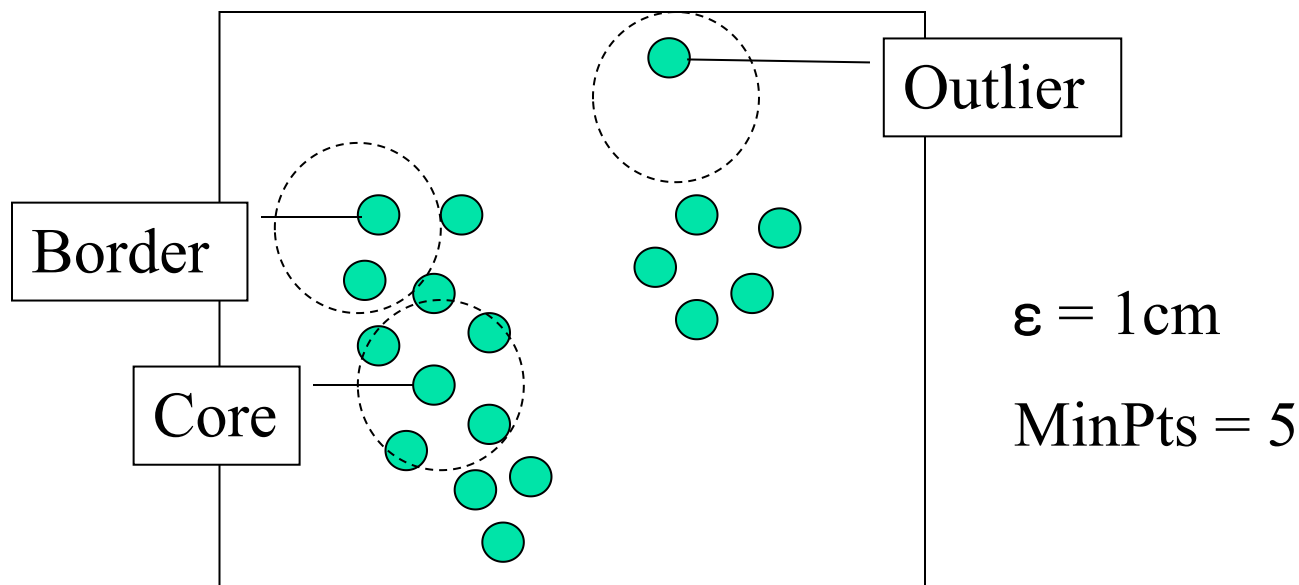
## ■ MinPts=3

- q是从p密度可达； p不是从q密度可达（q非核心）
- s和r从o密度可达； o从r密度可达；
- r, s, o密度相连



# DBSCAN(1996)

- **DBSCAN(Density Based Spatial Clustering of Applications with Noise)** 一个基于密度的聚类算法
- 可以在带有“噪音”的空间数据库中发现任意形状的聚类



# DBSCAN(续)

- 算法 [Visualizing DBSCAN Clustering \(naftaliharris.com\)](http://naftaliharris.com)
  - 任意选取一个点  $p$
  - 得到所有从  $p$  关于  $\epsilon$  和  $\text{MinPts}$  密度可达的点.
  - 如果  $p$  是一个核心点, 则找到一个聚类.
  - 如果  $p$  是一个边界点, 没有从  $p$  密度可达的点, DBSCAN 将访问数据库中的下一个点.
  - 继续这一过程, 直到数据库中的所有点都被处理.
- DBSCAN的复杂度
  - 采用空间索引, 复杂度为  $O(n \log n)$ , 否则为  $O(n^2)$
- DBSCAN的缺点:
  - 对用户定义的参数是敏感的, 参数难以确定(特别是对于高维数据), 设置的细微不同可能导致差别很大的聚类.  
(数据倾斜分布) 全局密度参数不能刻画内在的聚类结构

序号	属性1	属性2
A	4	6
B	8	2
C	3	5
D	2	4
E	6	3
F	4	4
G	6	5
H	7	3
I	7	1
J	8	4

给出一个样本事务数据库，对它实施  
DBSCAN算法

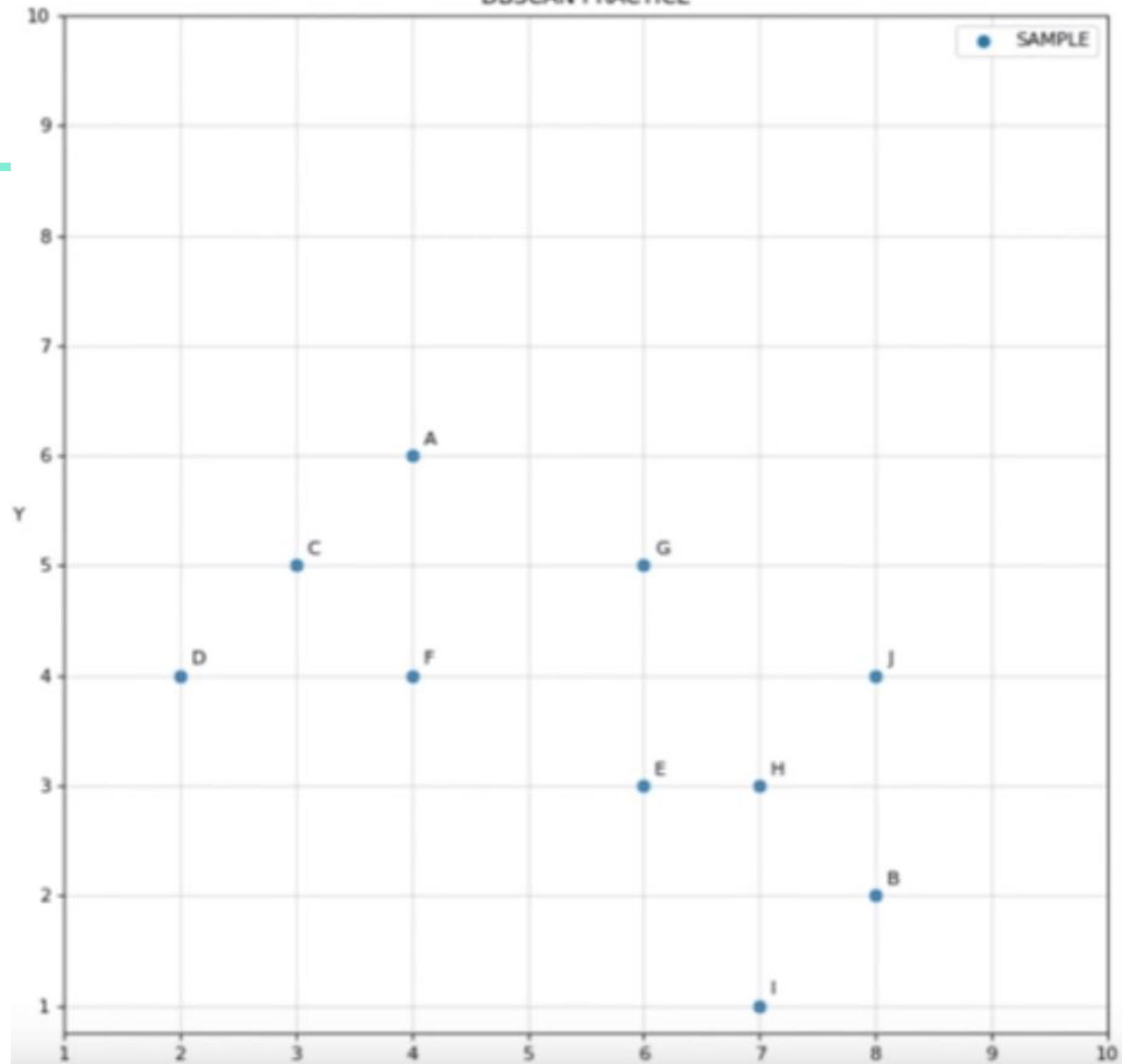
$n=10$  MinPts=4  $\epsilon=2$

- 1、给出所有的核心对象
- 2、对每个核心对象，写出从该点出发直接密度可达的点
- 3、对每个核心对象，写出从该点出发间接密度可达的点
- 4、写出最后的聚类结果

- 1、BCFH
- 2、B: BHIJ C:ACDF F:ACDF H:BEHIJ
- 3、B: E
- 4、{acdf}, {BEHIJ}



# DBSCAN PRACTICE



# OPTICS (1999)

---

- **OPTICS(Ordering Points To Identify the Clustering Structure)**
  - Ankerst, Breunig, Kriegel, 和 Sander 提出(SIGMOD'99)
  - 考虑DBSCAN, 根据给定的 $\epsilon$ 和MinPts聚类对象, 通常参数选择较为困难
  - 现实的高维数据集常常具有非常倾斜的分布, 全局密度参数不能很好的刻画其内在聚类结构
  - 为了同时构建不同的聚类, 应当以特定的顺序来处理对象, 优先选择最小的 $\epsilon$ 值密度可达的对象, 以便高密度的聚类能被首先完成

# OPTICS(续)

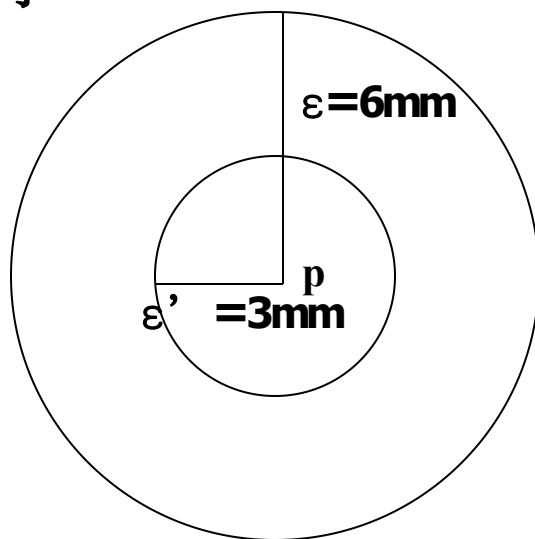
---

- 每个对象需要存储两个值
  - 对象p的**核心距离 (core-distance)** 是使得p成为核心对象的最小 $\epsilon$ 。如果p不是核心对象, p的核心距离没有定义
  - 对象q关于另一个对象p的**可达距离 (reachability-distance)** 是p的核心距离和p与q的欧几里得距离之间的较大值. 如果p不是一个核心对象, p和q之间的可达距离没有定义

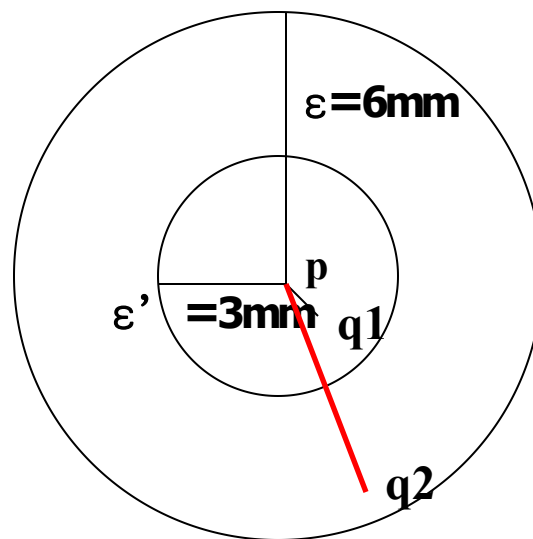
# OPTICS(续)

■ 例: 设 $\epsilon=6(\text{mm})$ ,  $\text{MinPts}=5$ .

- $p$ 的核心距离是 $p$ 与第四个最近数据对象之间的距离 $\epsilon'$
- $q_1$ 关于 $p$ 的可达距离是 $p$ 的核心距离(即 $\epsilon'=3\text{mm}$ ), 因为它比从 $p$ 到 $q_1$ 的欧几里得距离要大
- $q_2$ 关于 $p$ 的可达距离是从 $p$ 到 $q_2$ 的欧几里得距离, 它大于 $p$ 的核心距离



$p$ 的核心距离



可达距离  $(p, q_1) = \epsilon' = 3\text{mm}$

可达距离  $(p, q_2) = d(p, q_2)$

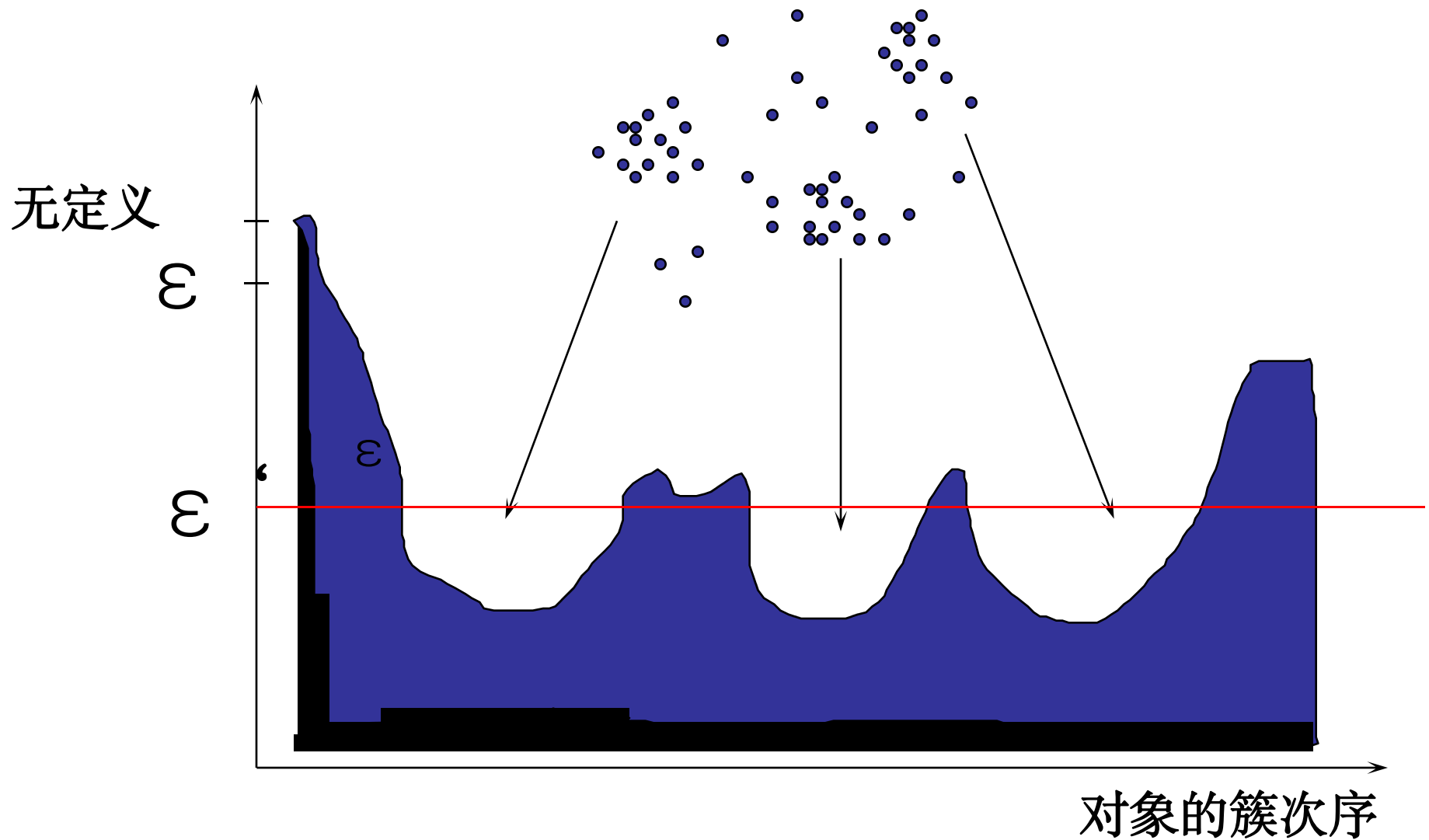
# OPTICS(续)

- 存储每个对象的核心距离和相应可达距离的表（OrderSeeds），存储顺序按照可达距离从小到大排序
- 迭代过程按照表中的顺序输出（遍历）对象，在迭代过程中表不断被更新
- 基本思路：
  - 开始以任意对象 $p$ 作为当前对象，检索当前对象 $p$ 的 $\epsilon$ 领域，并设置可达距离为未定义
  - 若 $p$ 是核心对象，则对于 $p$ 的 $\epsilon$ 领域内每个对象 $q$ ，更新从 $p$ 到 $q$ 的核心距离（即更新OrderSeeds）；转移到OrderSeeds中的下一个对象重复上述步骤
  - 若 $p$ 不是核心对象，转移到OrderSeeds下一个对象（开始阶段OrderSeeds为空，则转移到数据库中任意其它对象）
  - 迭代继续，直到输入耗尽且OrderSeeds为空

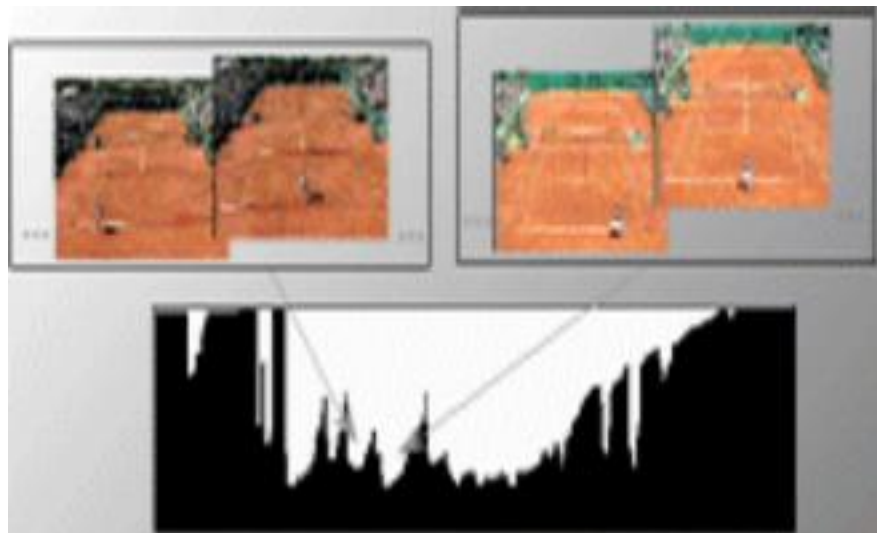
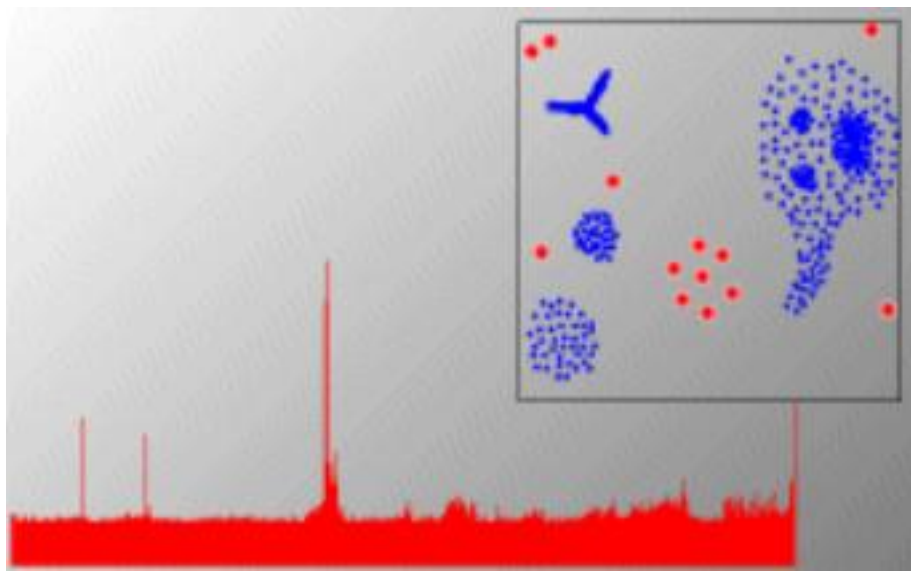
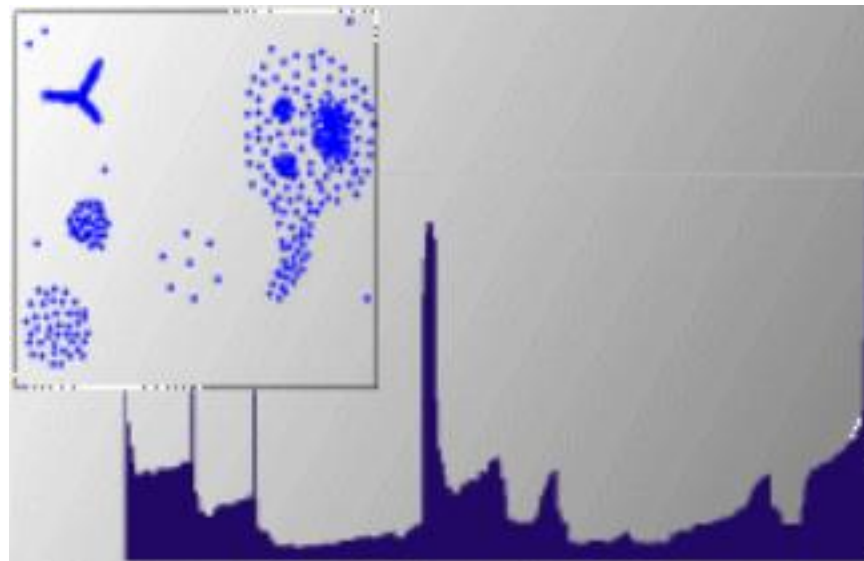
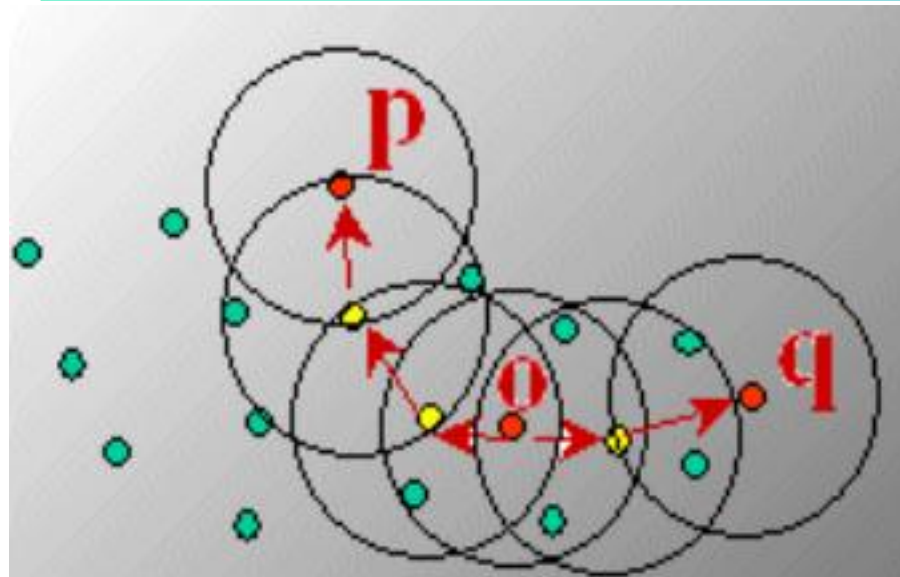
# OPTICS(续)

- OPTICS算法创建了数据库中对象的一个次序, 额外存储了每个对象的核心距离和一个适当的可达距离
- 已经提出了一种算法, 基于OPTICS产生的次序信息来抽取聚类. 对于小于在生成该次序中采用的距离 $\epsilon$ 的任何距离 $\epsilon'$ , 为提取所有基于密度的聚类, 这些信息是足够的
- 一个数据集合的聚类次序可被图形化地描述, 以助于理解
- 由于OPTICS算法与DBSCAN在结构上的等价性, 它具有和DBSCAN相同的时间复杂度, 即当使用空间索引时, 复杂度为 $O(n \log n)$

# 可达距离



# OPTICS及其应用





# DENCLUE(1998)

- **DENCLUE(DENsity-based CLUstEring)** 由Hinneburg 和 Keim (KDD'98)提出, 是基于密度分布函数的聚类方法
- **主要特点**
  - 坚实的数学基础, 概括了其他的聚类方法, 包括基于划分的, 层次的, 及基于位置的方法
  - 适用于具有大量噪音的数据集
  - 可用于高维数据集任意形状的聚类, 它给出了简洁的数学描述
  - 明显快于现有算法 (比 DBSCAN 快 45倍)
  - 但是, 需要大量参数, 要求对密度参数 $\sigma$ 和噪音阈值 $\xi$ 进行仔细的选择

# Denclude: 技术要点

---

- 使用栅格单元, 但只保存实际存放数据点的栅格单元信息, 并且在一个基于树的存取结构中管理这些单元.
- 影响函数(Influence function): 描述数据点在其邻域的影响.
- 数据空间的整体密度可以被模拟为所有数据点的影响函数的总和
- 聚类可以通过确定密度吸引点 (density attractor) 来得到.
- 密度吸引点是全局密度函数的局部最大值.

# DENCLUE(续)

- 设 $x$ 和 $y$ 是 $d$ 维特征空间 $F^d$ 中的对象. 数据对象 $y$ 对 $x$ 的**影响函数**是一个函数 $f^y_B: F^d \rightarrow R^+_0$ , 它是根据一个基本的影响函数  $f_B$ 来定义的

$$f^y_B(x) = f_B(x, y)$$

- 原则上, 影响函数可以是一个任意的函数, 它由某个邻域内的两个对象之间的距离来决定
- 例如欧几里得距离函数, 用来计算一个方波影响函数(square wave influence function):

$$f_{Square}(x, y) = \begin{cases} 0 & \text{如果 } d(x, y) > \sigma \\ 1 & \text{其它} \end{cases}$$

# DENCLUE(续)

- 高斯影响函数

$$f_{Gauss}(x, y) = e^{-\frac{d(x, y)^2}{2\sigma^2}}$$

- 一个对象  $x \in F^d$  的密度函数被定义为所有数据点的影响函数的和. 给定  $n$  个对象,  $D = \{x_1, \dots, x_n\} \subset F^d$ , 在  $x$  上的密度函数定义如下

$$f_B^D(x) = \sum_{i=1}^n f_B^{x_i}(x)$$

# DENCLUE(续)

- 例如, 根据高斯影响函数得出的密度函数是

$$f_{Gauss}^D(x) = \sum_{i=1}^N e^{-\frac{d(x, x_i)^2}{2\sigma^2}}$$

- 根据密度函数, 我们能够定义该函数的梯度和密度吸引点 (全局密度函数的局部最大)
- 一个点 $x$ 是被一个**密度吸引点**  $x^*$ 密度吸引的, 如果存在一组点 $x_0, x_1, \dots, x_k, x_0=x, x_k=x^*$ , 对 $0 < i < k$ ,  $x_{i-1}$ 的梯度是在 $x_i$ 的方向上
- 对一个连续的, 可微的影响函数, 用梯度指导的爬山算法能用来计算一组数据点的密度吸引点

# 密度吸引点

## Def. 2 (Gradient)

The gradient of a function  $f_B^D(x)$  is defined as

$$\nabla f_B^D(x) = \sum_{i=1}^N (x_i - x) \cdot f_B^{x_i}(x).$$

In case of the Gaussian influence function, the gradient is defined as:

$$\nabla f_{Gauss}^D(x) = \sum_{i=1}^N (x_i - x) \cdot e^{-\frac{d(x, x_i)^2}{2\sigma^2}}.$$

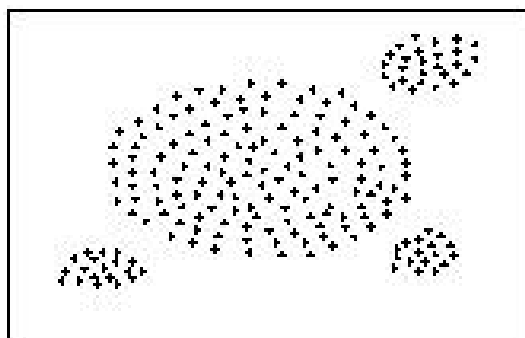
## Def. 3 (Density-Attractor)

A point  $x^* \in F^d$  is called a *density-attractor* for a given influence function, iff  $x^*$  is a local maximum of the density-function  $f_B^D$ .

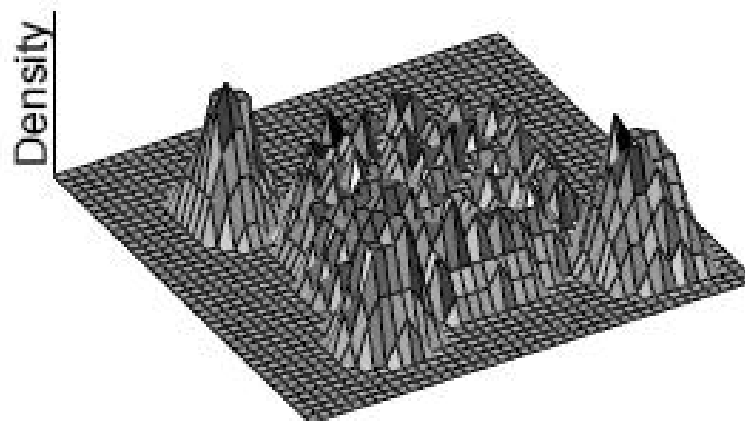
A point  $x \in F^d$  is *density-attracted* to a density-attractor  $x^*$ , iff  $\exists k \in N : d(x^k, x^*) \leq \epsilon$  with

$$x^0 = x, \quad x^i = x^{i-1} + \delta \cdot \frac{\nabla f_B^D(x^{i-1})}{\|\nabla f_B^D(x^{i-1})\|}.$$

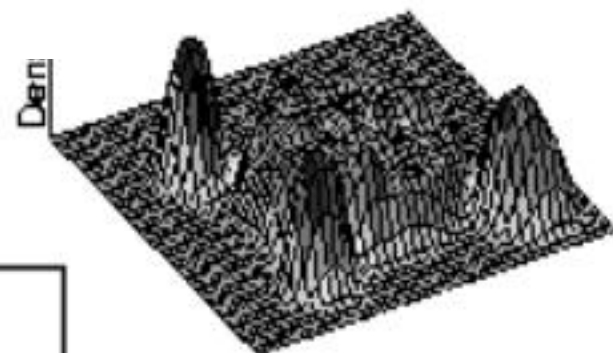
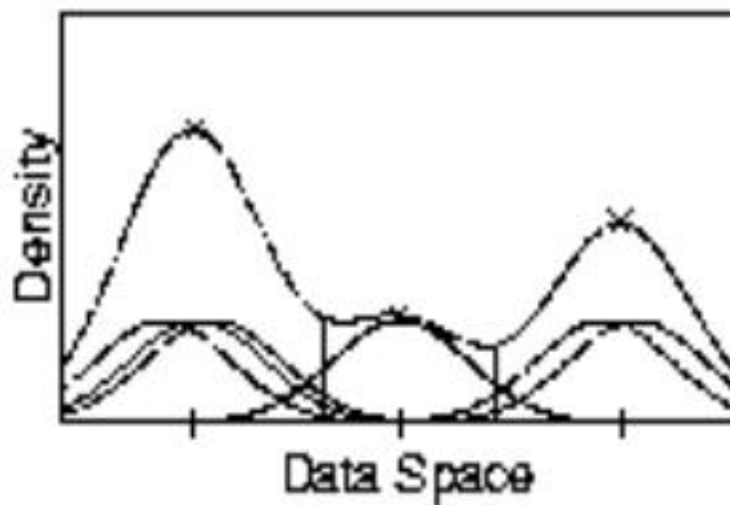
# 密度吸引点



(a) Data Set



(b) Square Wave



(c) Gaussian

# 中心定义的簇和任意形状的簇

- 密度吸引点  $x^*$  的**中心定义的簇**(*center-defined cluster*)是一个被 $x^*$ 密度吸引的子集 $C$ , 在 $x^*$ 的密度函数不小于一个阈值 $\xi$ ; 否则(即如果它的密度函数值小于 $\xi$ ), 它被认为是孤立点
- 一个**任意形状的簇**(*arbitrary-shape cluster*)是子集 $C$ 的集合, 每一个是各自密度吸引子密度吸引的, 有不小于阈值 $\xi$ 的密度函数值, 从每个区域到另一个都存在一条路径 $P$ , 该路径上每个点的密度函数值都不小于 $\xi$



# 中心定义的簇和任意形状的簇

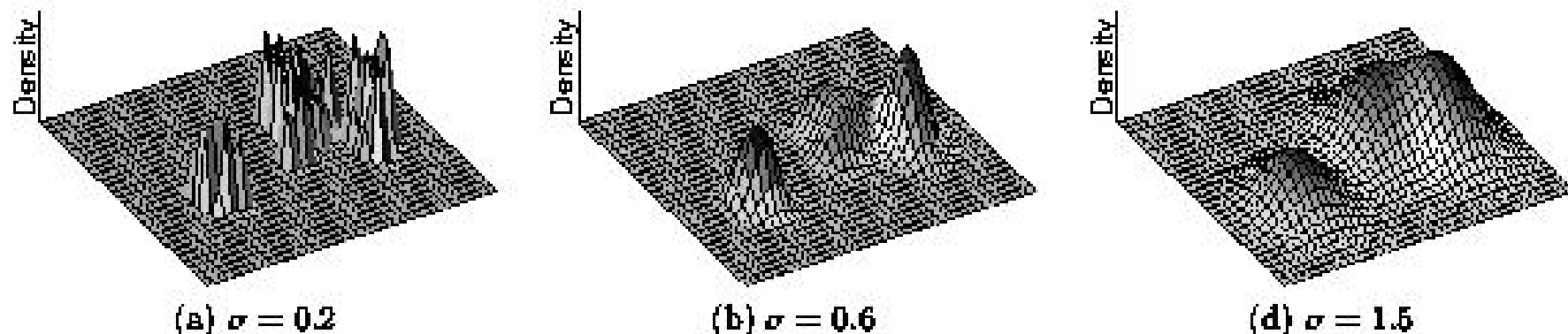


Figure 3: Example of Center-Defined Clusters for different  $\sigma$

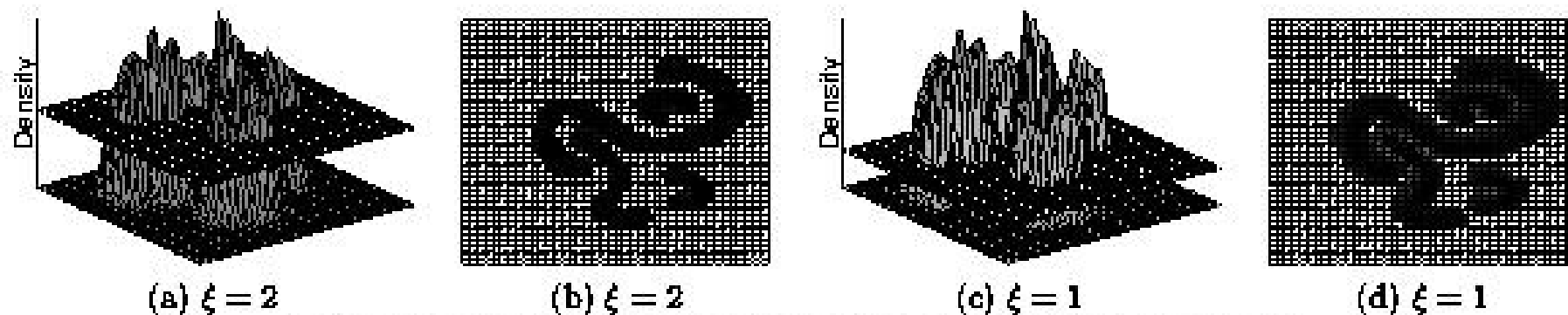



Figure 4: Example of Arbitrary-Shape Clusters for different  $\xi$

# 第10章： 聚类分析： 基本概念和方法

---

- 聚类分析
- 划分方法
- 层次方法
- 基于密度的方法
- 基于网格的方法 
- 聚类评估
- 小结

# 基于网格的聚类方法

---

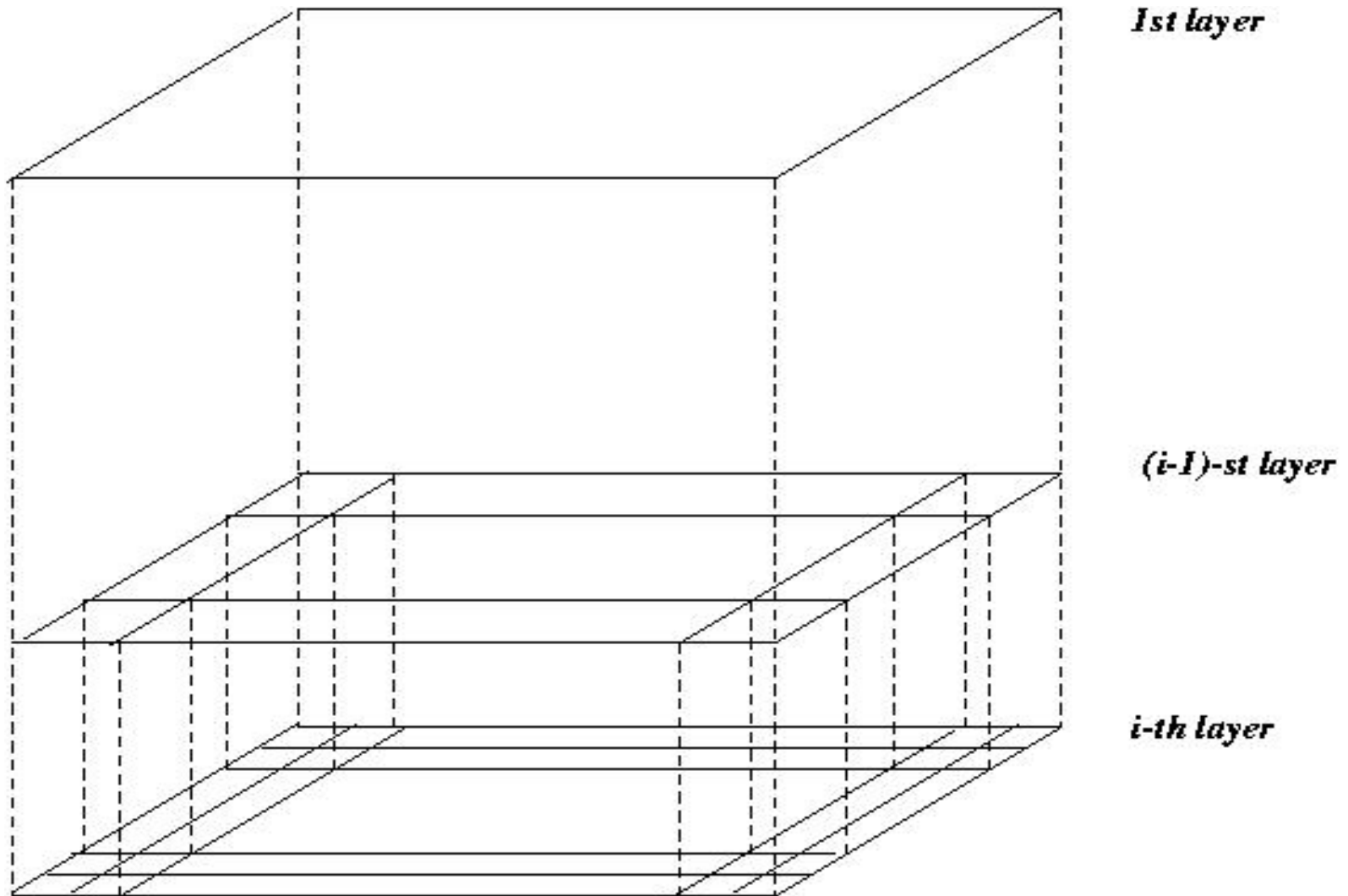
- 使用多分辨率的网格数据结构
- 一些有趣的方法
  - **STING** (统计信息网格: a Statistical Information Grid approach) 由 Wang, Yang 和 Muntz 提出(1997)
  - WaveCluster 由 Sheikholeslami, Chatterjee, 和 Zhang 提出(VLDB'98)
    - 采用小波方法的多分辨率的聚类方法
  - **CLIQUE**: Agrawal, et al. 提出(SIGMOD'98)

# STING: 统计信息网格

---

- **STING( S**Tatistical **I**Nformation **G**rid)是一个基于网格的多分辨率聚类技术, 由 **Wang, Yang** 和 **Muntz** 提出(VLDB'97)
- 空间区域划分为矩形单元
- 多个级别的矩形单元, 对应不同级别的分辨率. 这些单元形成了一个层次结构: 每个高层单元被划分为多个低一层的单元
- 预先计算和存储关于每个网格单元属性的统计信息 (如平均值, 最大值, 和最小值), 用于回答查询

# STING(续)



# STING(续)

- 统计参数包括:
  - 属性无关的参数count; 属性相关的参数m(平均值), s(标准偏差), min(最小值), max(最大值)
  - 该单元中属性值遵循的分布类型: 正态的, 一致的, 指数的, 无(分布未知)
- 分布的值可以由用户指定, 也可以通过假设检验(如 $\chi^2$ 检验)来获得
- 高层单元的统计参数可以很容易地从底层单元的计算得到; 最底层单元的参数count, m, s, min, 和max直接进行计算

# STING(续)

- 使用自顶向下的方法回答空间数据查询
  - 在层次结构选定一层作为查询处理起始点（通常该层包含少量的单元）
  - 对当前层次的每个单元，计算置信度区间（或估算其概率），用以反映该单元与给定查询关联程度
  - 删除不相关的单元，进一步处理不考虑它们
  - 结束当前层的考查后，就处理下一层
  - 重复这一过程，直到最低层

# STING(续)

## ■ 优点:

- 基于网格的计算是独立于查询的: 存储在每个单元中的统计信息是不依赖于查询的汇总信息
- 网格结构有利于并行处理和增量更新
- 效率很高: STING扫描数据库一次来计算单元的统计信息, 因此产生聚类的时间复杂度是 $O(n)$ ,  $n$ 是对象的数目
- 层次结构建立后, 查询处理时间是  $O(K)$ ,  $K$ 是最底层网格单元的数目, 通常远远小于 $n$

## ■ 缺点:

- 所有的聚类边界或是水平, 或是竖直, 没有斜的分界线
- 尽管该技术有快速的处理速度, 但可能降低簇的质量和精确性



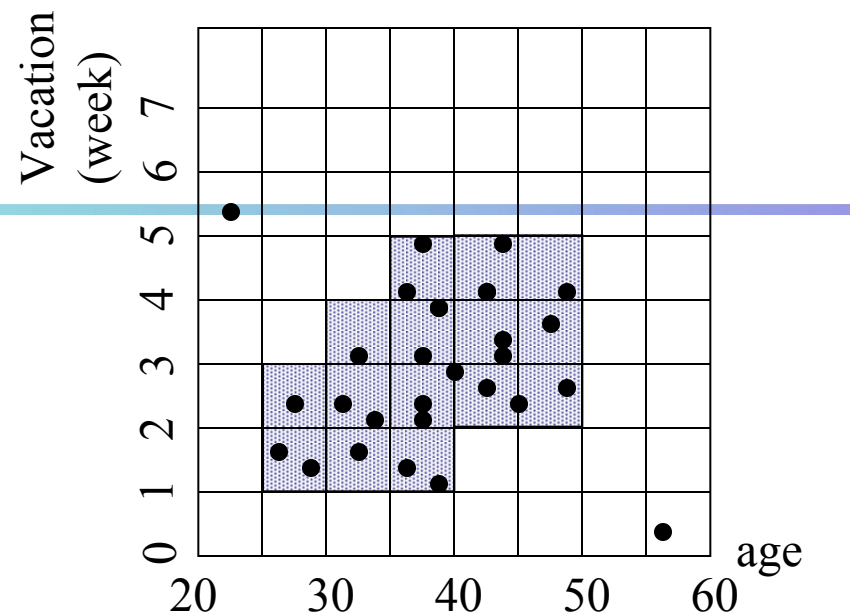
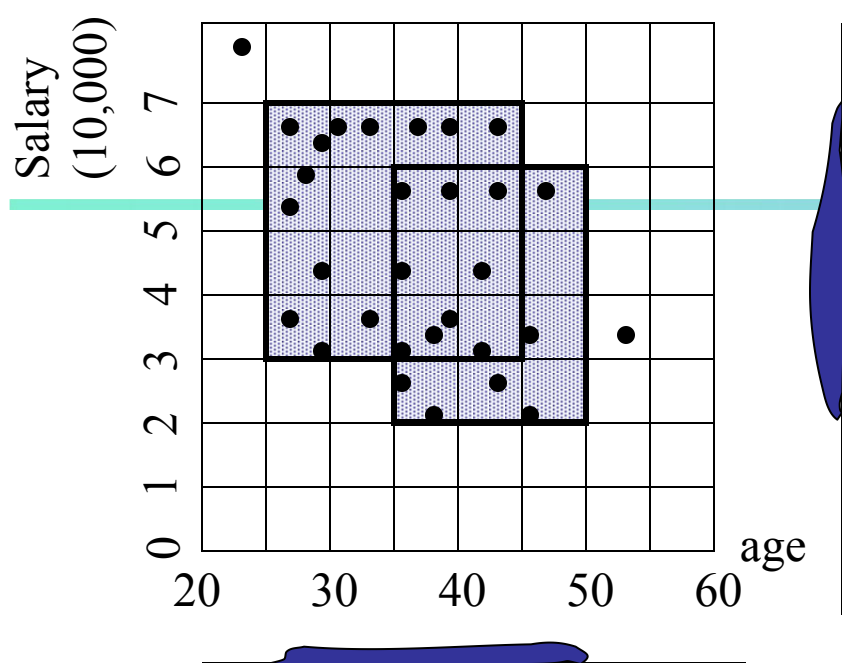
# CLIQUE

- CLIQUE(Clustering In QUEst)综合了基于密度和基于网格的聚类方法,对于大型数据库中的高维数据的聚类非常有效
- 自动识别高维数据空间的子空间,在子空间上聚类比在原空间上更好
- 基本思想: CLIQUE是基于密度和基于网格的聚类方法
  - 它将每个维划分成相同个数的等长区间
  - 它将m-维数据空间划分成不重叠的长方形单元
  - 一个单元是稠密的,如果包含在该单元中的数据点占全部数据点的比例超过输入的模式参数
  - 簇是相连的密集单元的最大集合

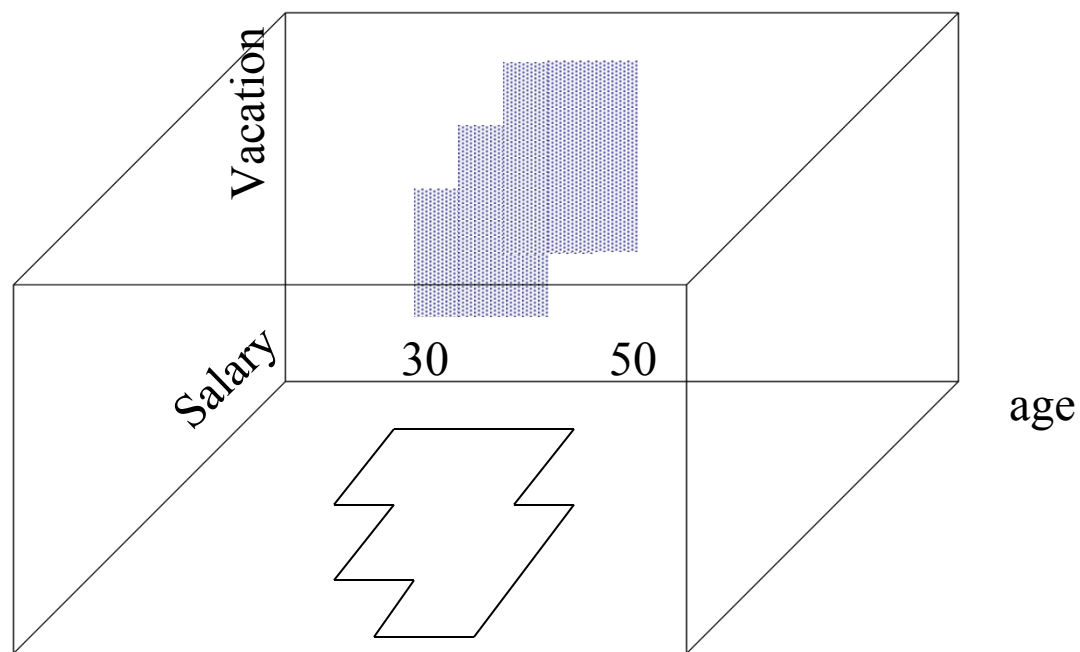
# CLIQUE

---

- CLIQUE所采用的先验性质:
  - 如果一个 $k$ 维单元是密集的, 那么它在 $k-1$ 维空间上的投影也是密集的
  - 也就是说, 给定一个 $k$ 维的候选密集单元, 如果我们检查它的 $k-1$ 维投影单元, 发现任何一个不是密集的, 那么我们知道第 $k$ 维的单元也不可能是密集的
  - 可以从 $k-1$ 维空间中发现的密集单元来推断 $k$ 维空间中潜在的或候选的密集单元 (由 $k-1$ 维密集单元构建 $k$ 维密集单元)



$\tau = 3$



# CLIQUE : 主要步骤

---

- 第一步，对n维数据空间进行划分：划分为互不相交的矩形单元，同时识别其中的密集单元
- 第二步，识别聚类：采用深度优先算法获得一系列相互连通的密集单元的集和
- 第三步，为每个簇生成最小化描述：对于每个连通的密集单元的集和生成最大覆盖，从已有的最大覆盖中移走样本数目最少的多余空间，直到无法移除

# CLIQUE的优缺点

---

## ■ 优点


- 它自动地找出高维的子空间, 高密度的聚类存在于在这些子空间中
- 对元组的输入顺序不敏感, 无需假设任何规范的数据分布
- 它随输入数据的大小线性地扩展, 当数据的维数增加时具有良好的可扩展性

## ■ 缺点

- 由于方法大大简化, 聚类结果精确性可能会降低

# 第10章：聚类分析：基本概念和方法

---

- 聚类分析
- 划分方法
- 层次方法
- 基于密度的方法
- 基于网格的方法
- 聚类评估 
- 小结

# 估计聚类的趋势

- 评估给定数据集是否具有可以导致有意义聚类的非随机结构
- 霍普金斯统计量:
  - 假设数据集D是随机变量o的一个样本, 确定o在多大程度上不同于数据空间中的均匀分布
  - 均匀的从D的空间抽取n个点 $p_1, \dots, p_n$ , 对于每个 $p_i$ ,找出其在D的最近邻:  $x_i = \min\{\text{dist}(p_i, v)\}$  where  $v$  in D
  - 均匀的从D中抽取n个点 $q_1, \dots, q_n$ , 对于每个 $q_i$ ,找出其在D- $\{q_i\}$ 的最近邻:  $y_i = \min\{\text{dist}(q_i, v)\}$  where  $v$  in D and  $v \neq q_i$
  - 计算霍普金斯统计量:

$$H = \frac{\sum_{i=1}^n y_i}{\sum_{i=1}^n x_i + \sum_{i=1}^n y_i}$$

- 若D是均匀分布,  $\sum x_i$ 与 $\sum y_i$ 相近且H接近0.5. 若D聚类, H接近0

# 确定聚类的数目

## ■ 经验方法

- $k \approx \sqrt{n/2}$ ,  $n$ 为点数

## ■ 肘时法

- 不同的 $k$ 可以获得不同的类内方差和（随类数据增加下降），取类内方差和随 $k$ 变化曲线的拐点

## ■ 交叉验证法

- 将数据划分为 $m$ 部分
- 用 $m - 1$ 部分获得聚类模型，余下部分评估聚类质量（测试样本与类中心的距离和）
- 对于 $k > 0$ 重复 $m$ 次，比较总体的质量，选择能获得最好聚类质量的 $k$



# 度量聚类的质量

---

- 两类方法：外在方法和内在方法
- 外在方法：有监督的方法，需要基准数据
  - 用一定的度量评判聚类结果与基准数据的符合程度
- 内在方法：无监督的方法，无需基准数据
  - 类内聚集程度和类间离散程度

# 外在方法

- 聚类质量评估: 用  $Q(C, C_g)$  表示聚类  $C$  在给定基准数据  $C_g$  条件下的质量度量.
- $Q$  的好坏取决于四个条件:
  - 簇的同质性: 簇内越纯越好
  - 簇的完整性: 能够将基准数据中属于相同类的样本聚类为相同的类
  - 碎布袋: 把一个异种数据加入纯类应该比放入碎布袋受到更大的“处罚”
  - 小簇的保持性: 把小簇划分成更小簇比把大簇划分为小簇的危害更大

# 内在方法

## ■ 轮廓系数:

- 对于D中的每个对象o, 计算o与o所属的簇内其它对象之间的平均距离a(o):

$$a(o) = \frac{\sum_{o' \in C_i, o' \neq o} dist(o, o')}{|C_i| - 1}$$

- 类似的, b(o)是o到不包含o的所有簇的最小平均距离:

$$b(o) = \min_{C_j: 1 \leq j \leq k, j \neq i} \left\{ \frac{\sum_{o' \in C_j} dist(o, o')}{|C_j|} \right\}$$


- 轮廓系数定义为:

$$s(o) = \frac{b(o) - a(o)}{\max\{a(o), b(o)\}}$$

- 轮廓系数在-1到1之间1表明 (类内紧凑, 类间离散) 聚类质量高, 反之则不高。

# 第10章：聚类分析：基本概念和方法

---

- 聚类分析
- 划分方法
- 层次方法
- 基于密度的方法
- 基于网格的方法
- 聚类评估
- 小结 

# Summary

---

- **Cluster analysis** groups objects based on their **similarity** and has wide applications
- Measure of similarity can be computed for **various types of data**
- Clustering algorithms can be **categorized** into partitioning methods, hierarchical methods, density-based methods, grid-based methods, and model-based methods
- **K-means** and **K-medoids** algorithms are popular partitioning-based clustering algorithms
- **Birch** and **Chameleon** are interesting hierarchical clustering algorithms, and there are also probabilistic hierarchical clustering algorithms
- **DBSCAN**, **OPTICS**, and **DENCLU** are interesting density-based algorithms
- **STING** and **CLIQUE** are grid-based methods, where CLIQUE is also a subspace clustering algorithm
- Quality of clustering results can be evaluated in various ways

# References (1)

---

- R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. SIGMOD'98
- M. R. Anderberg. Cluster Analysis for Applications. Academic Press, 1973.
- M. Ankerst, M. Breunig, H.-P. Kriegel, and J. Sander. Optics: Ordering points to identify the clustering structure, SIGMOD'99.
- Beil F., Ester M., Xu X.: "Frequent Term-Based Text Clustering", KDD'02
- M. M. Breunig, H.-P. Kriegel, R. Ng, J. Sander. LOF: Identifying Density-Based Local Outliers. SIGMOD 2000.
- M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases. KDD'96.
- M. Ester, H.-P. Kriegel, and X. Xu. Knowledge discovery in large spatial databases: Focusing techniques for efficient class identification. SSD'95.
- D. Fisher. Knowledge acquisition via incremental conceptual clustering. Machine Learning, 2:139-172, 1987.
- D. Gibson, J. Kleinberg, and P. Raghavan. Clustering categorical data: An approach based on dynamic systems. VLDB'98.
- V. Ganti, J. Gehrke, R. Ramakrishan. CACTUS Clustering Categorical Data Using Summaries. KDD'99.

# References (2)

---

- D. Gibson, J. Kleinberg, and P. Raghavan. Clustering categorical data: An approach based on dynamic systems. In Proc. VLDB'98.
- S. Guha, R. Rastogi, and K. Shim. Cure: An efficient clustering algorithm for large databases. SIGMOD'98.
- S. Guha, R. Rastogi, and K. Shim. ROCK: A robust clustering algorithm for categorical attributes. In *ICDE'99*, pp. 512-521, Sydney, Australia, March 1999.
- A. Hinneburg, D.I A. Keim: An Efficient Approach to Clustering in Large Multimedia Databases with Noise. KDD'98.
- A. K. Jain and R. C. Dubes. Algorithms for Clustering Data. Printice Hall, 1988.
- G. Karypis, E.-H. Han, and V. Kumar. CHAMELEON: A Hierarchical Clustering Algorithm Using Dynamic Modeling. *COMPUTER*, 32(8): 68-75, 1999.
- L. Kaufman and P. J. Rousseeuw. Finding Groups in Data: an Introduction to Cluster Analysis. John Wiley & Sons, 1990.
- E. Knorr and R. Ng. Algorithms for mining distance-based outliers in large datasets. VLDB'98.

# References (3)

- G. J. McLachlan and K.E. Bkasford. Mixture Models: Inference and Applications to Clustering. John Wiley and Sons, 1988.
- R. Ng and J. Han. Efficient and effective clustering method for spatial data mining. VLDB'94.
- L. Parsons, E. Haque and H. Liu, Subspace Clustering for High Dimensional Data: A Review, SIGKDD Explorations, 6(1), June 2004
- E. Schikuta. Grid clustering: An efficient hierarchical clustering method for very large data sets. Proc. 1996 Int. Conf. on Pattern Recognition,.
- G. Sheikholeslami, S. Chatterjee, and A. Zhang. WaveCluster: A multi-resolution clustering approach for very large spatial databases. VLDB'98.
- A. K. H. Tung, J. Han, L. V. S. Lakshmanan, and R. T. Ng. Constraint-Based Clustering in Large Databases, ICDT'01.
- A. K. H. Tung, J. Hou, and J. Han. Spatial Clustering in the Presence of Obstacles, ICDE'01
- H. Wang, W. Wang, J. Yang, and P.S. Yu. Clustering by pattern similarity in large data sets, SIGMOD' 02.
- W. Wang, Yang, R. Muntz, STING: A Statistical Information grid Approach to Spatial Data Mining, VLDB'97.
- T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH : An efficient data clustering method for very large databases. SIGMOD'96.
- Xiaoxin Yin, Jiawei Han, and Philip Yu, "[LinkClus: Efficient Clustering via Heterogeneous Semantic Links](#)", in Proc. 2006 Int. Conf. on Very Large Data Bases (VLDB'06), Seoul, Korea, Sept. 2006.



# 提问与解答环节





# 谢谢聆听

· 学习就是为了达到一定目的而努力去干, 是为一个目标去战胜各种困难的过程,这个过程会充满压力、痛苦和挫折

Learning Is To Achieve A Certain Goal And Work Hard, Is A Process To Overcome Various Difficulties For A Goal