数据科学导论

Introduction to Data Science

第四章 数据挖掘基础

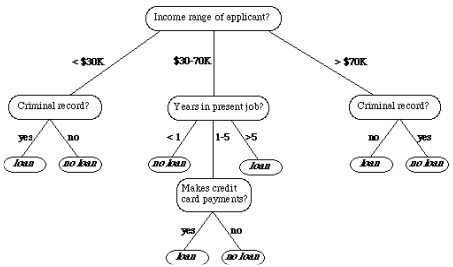
陈恩红，黄振亚
Email: huangzhy@ustc.edu.cn, cheneh@ustc.edu.cn

课程主页：
http://staff.ustc.edu.cn/~huangzhy/Course/DS2024.html

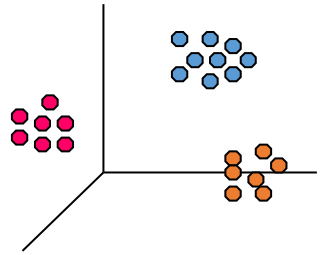
数据挖掘基础

- 数据挖掘——四个任务有哪些常用方法?

分类与预测



聚类



数据

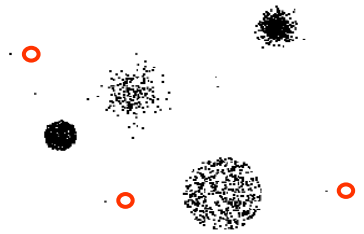
	T		H		P	
	L	H	L	H	L	H
J	-6.0	8.8	60	100	986	1044
F	-2.8	10.9	48	100	973	1025
M	-5.6	17.7	34	100	976	1037
A	-1.2	22.2	27	100	996	1036
M	-0.8	27.8	25	100	1003	1034
J	5.2	29.1	26	100	998	1030
J	9.8	30.6	23	99	997	1027
A	5.6	26.1	31	100	992	1029
S	5.2	24.8	35	100	998	1028
O	-0.4	21.3	42	100	990	1031
N	-7.6	17.3	55	100	963	1023
D	-10.4	9.2	53	100	987	1039

table 17a

2010 monthly weather variation, Cambridge (UK)

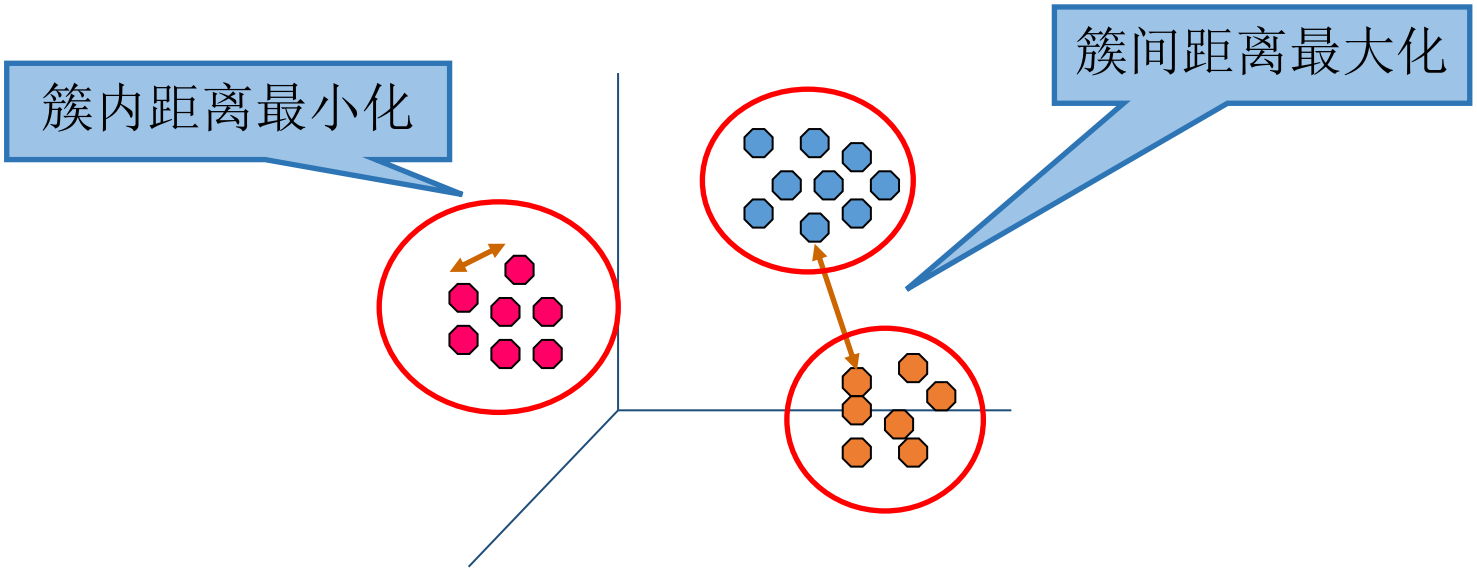
关联分析

异常检测



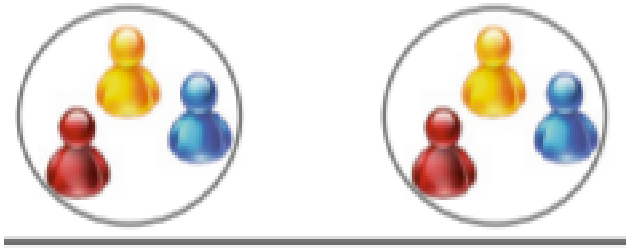
聚类分析

- 数据挖掘任务 —— 聚类(Clustering)
  - 目标：对数据进行**“群体性”分析**，将样本分为若干个**簇**（Clusters）
    - 其中，每个簇都由相似的样本所组成
  - 簇的特点：簇内相似（距离近），簇间相异（距离远）



聚类分析

- 聚类分析要解决三个问题
  - **1. 如何定义簇？：即，思考我们的目标（但具有主观性）**
    - “群体性”的依据：不同的“群体性”立场，可以得到不同的簇
    - 例如，学生分组应该考虑 **技能互补？** 还是 **能力相近**？
  - **2. 如何定义相关性？** 即，度量数据之间的相似性
    - 相似性度量往往存在一定局限性，未必反映聚类的真实意图
    - 例如，常用向量表征数据(人的爱好)， 但是否绝对相似？

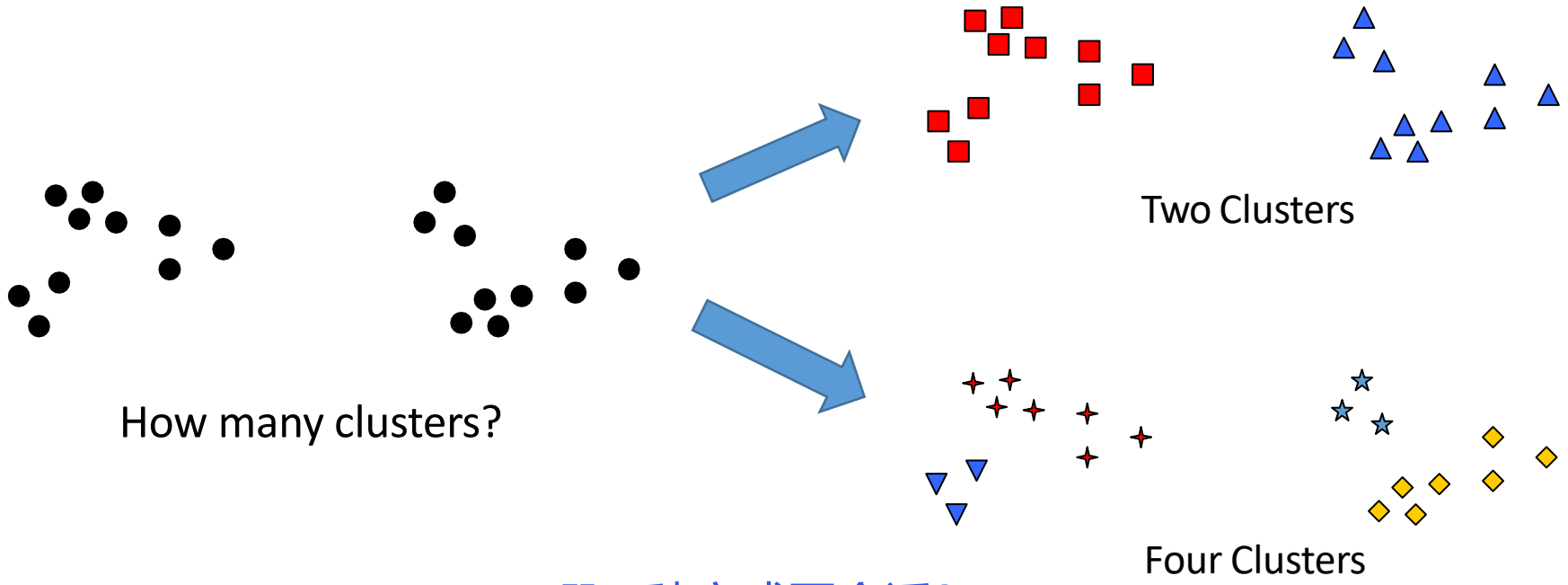


技能互补？ 能力相近？

图片本身相似，但代表的类别完全不同？

聚类分析

- 聚类分析要解决三个问题
  - **3. 如何决定簇的数量？** 即，选择合适数量的簇
    - 数据没有天然标签，簇的数量往往是个开放性问题
    - 避免过大或过小的簇，会导致失去代表性，但这未必可通过簇数调节



哪一种方式更合适？

聚类分析

- 聚类方法：最常见的无监督学习算法
- 常用方法
  - K均值聚类(K-means)
  - 层次聚类(Hierarchical Clustering)
  - 密度聚类(Density-based Clustering)
  - 聚类效果验证
  - 前沿聚类方法

聚类分析：K-means

- K-means的基本概念
  - 数据：视作高维空间中的一个点，表示为向量
  - 中心点：簇的中心，反应簇的共有属性
  - 簇的数量：人为设定
  - 数据的关系度量：用“平均”的方式簇中心与簇中数据

聚类分析：K-means

- K-means的特点
  - 关于中心点
    - 中心点一般采用随机初始化，因此重复K-means得到的结果可能不同
    - 中心点一般设置为簇内数据的平均向量（算术平均）
  - 关于相关性度量
    - 可用欧几里得距离、余弦相似度、相关系数等度量
  - 关于算法运行
    - K均值算法常常几轮就收敛
    - 算法停止条件为：Until relatively few points change clusters
      - 低于一定数量的数据 更新簇的归属，即，每个簇中 只有 很少的数据 发生变化
    - 算法复杂度$O(n \times K \times I \times d)$
      - n = 样本总数, K = 簇的个数数, I = 迭代轮数, d = 特征维度

聚类分析：K-means

- K-means的特点: 如何解决初始中心选择的问题?
  - 最简单的方法：多次运行
    - 但效率较低 （能否得到好的结果 看你的运气 ）
  - 采少数样本，借助其他聚类（如层次聚类）先确定出初始中心
    - 然而层次聚类开支较大，同时此方法仅适用于K较小的情况
  - 初始选择大于K的数量，然后从中挑选聚类分隔较为明显的中心
  - 后处理 “修补” 聚类的结果
  - 二分K均值方案（Bisecting K-means）

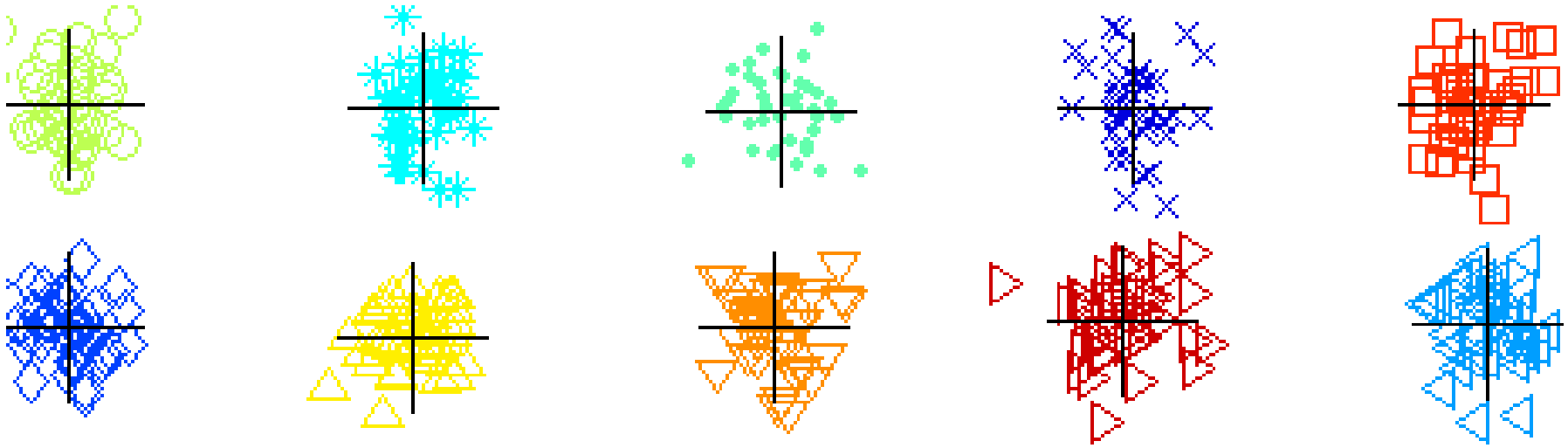
聚类分析：K-means

- K-means的特点: 如何解决初始中心选择的问题?
  - 二分K均值方案（Bisecting K-means）
    - 不容易受到初始化问题的影响
    - K-means的变体，类似于一种层次聚类的思想
    - 基本思想：为了得到K个簇，先分为 2 个簇，然后不断选择其中一个分裂

1: Initialize the list of clusters to contain the cluster containing all points.
2: **repeat**
3: Select a cluster from the list of clusters
4: **for** $i = 1$ to $number\_of\_iterations$ **do**
5: Bisect the selected cluster using basic K-means
6: **end for**
7: Add the two clusters from the bisection with the lowest SSE to the list of clusters.
8: **until** Until the list of clusters contains K clusters

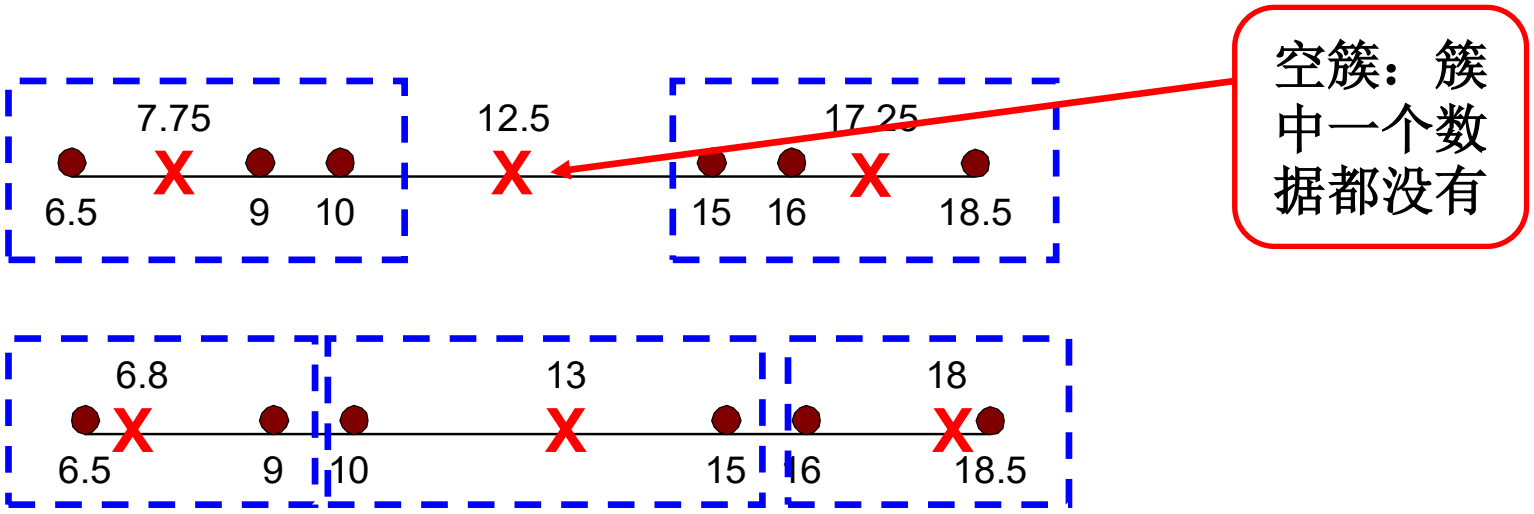
聚类分析：K-means

- K-means的特点: 如何解决初始中心选择的问题?
  - 实例：二分K均值方案（Bisecting K-means）
  - 从这个实例可以看出， 二分K均值受初始中心的影响不大
  - 究其原因，二分K均值 可视作一个“逐步求精” 的过程



数据挖掘基础

- K-means的特点: 可能返回空簇
  - 例如，所有的点在分配时都未被分配到某个簇
    - 解决方法：以样本作为初始中心，则不会出现，即簇内至少一个点
  - 处理空簇：一般而言，新生成一个簇来替代空簇(思路类似后处理)
    - 解决方法1：选择一个最远样本点新生成一个簇
    - 解决方法2：将最大SSE的簇进行拆分

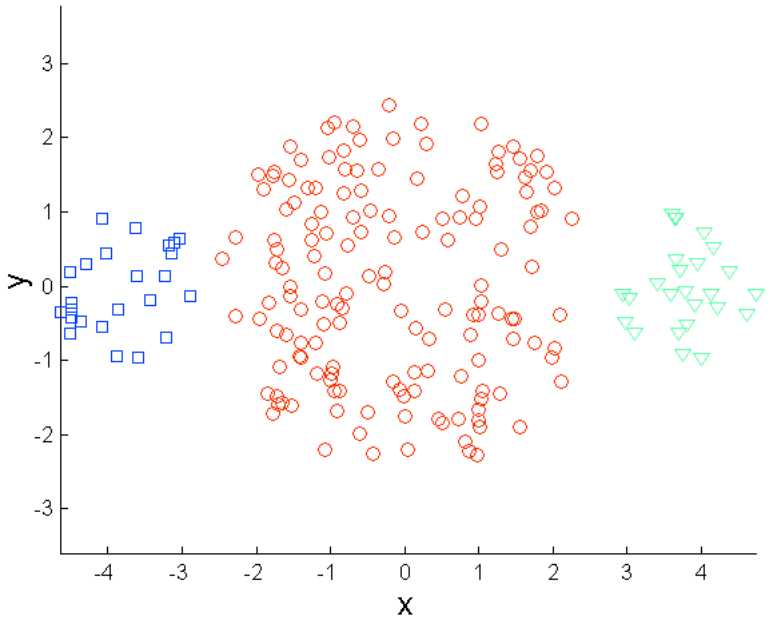


聚类分析：K-means

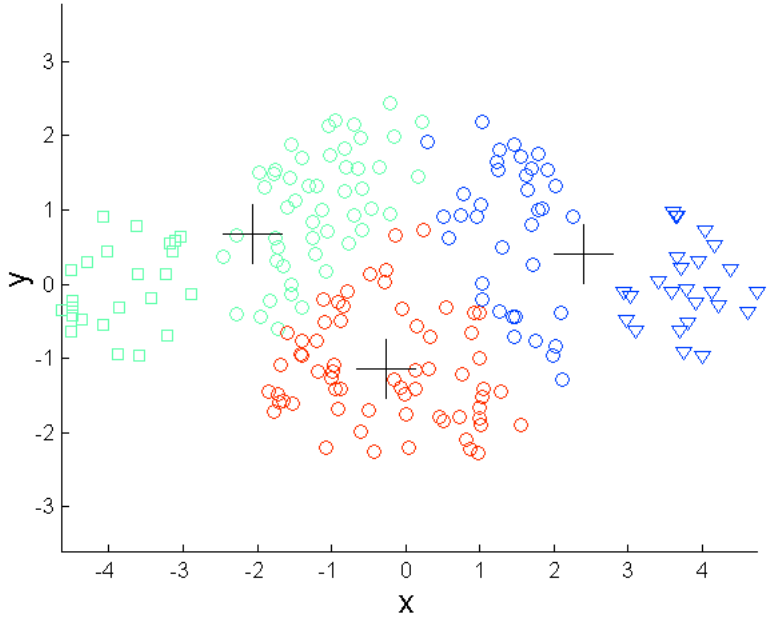
- K-means的局限性
  - 簇的特点会影响K-means聚类的结果
    - 1. 簇的规模
    - 2. 簇的（数据）密度
    - 3. 簇的（不规则）形状

聚类分析：K-means

- K-means的局限性
  - 1. 簇的规模：当出现规模不同的簇时，往往结果会受到一定干扰



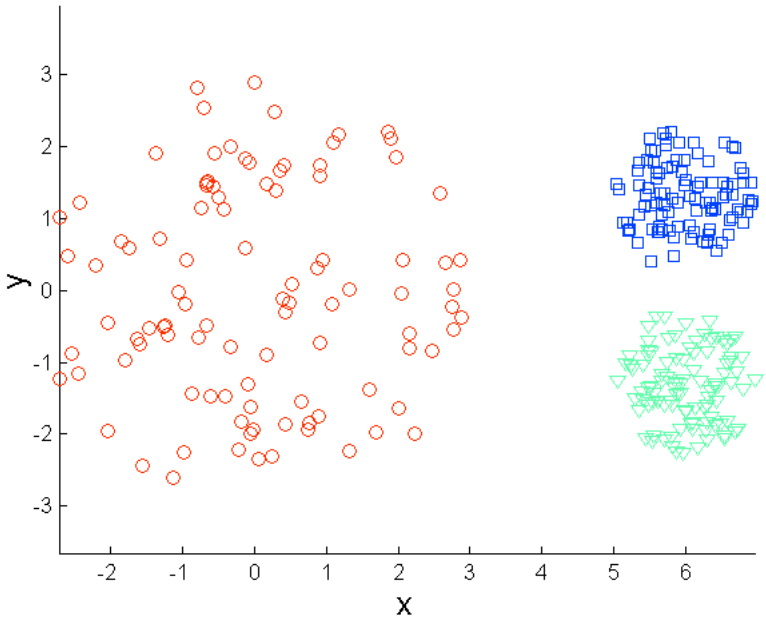
Original Points



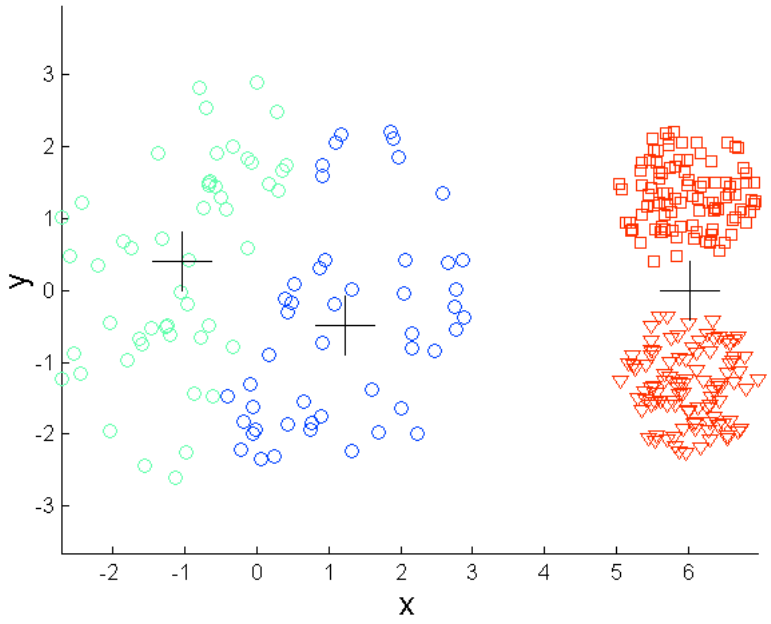
K-means (3 Clusters)

聚类分析：K-means

- K-means的局限性
  - 2. 簇的(数据)密度：当出现密度不同的簇时，往往结果会受到一定干扰



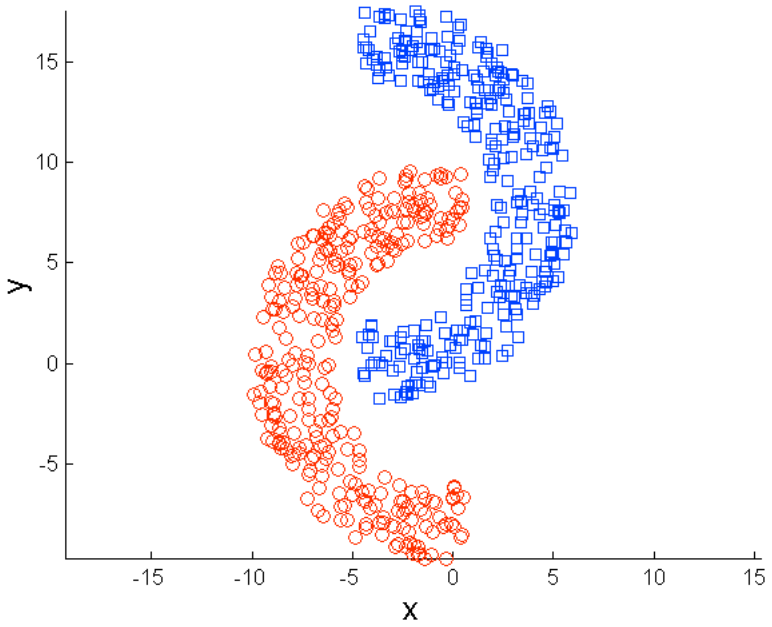
Original Points



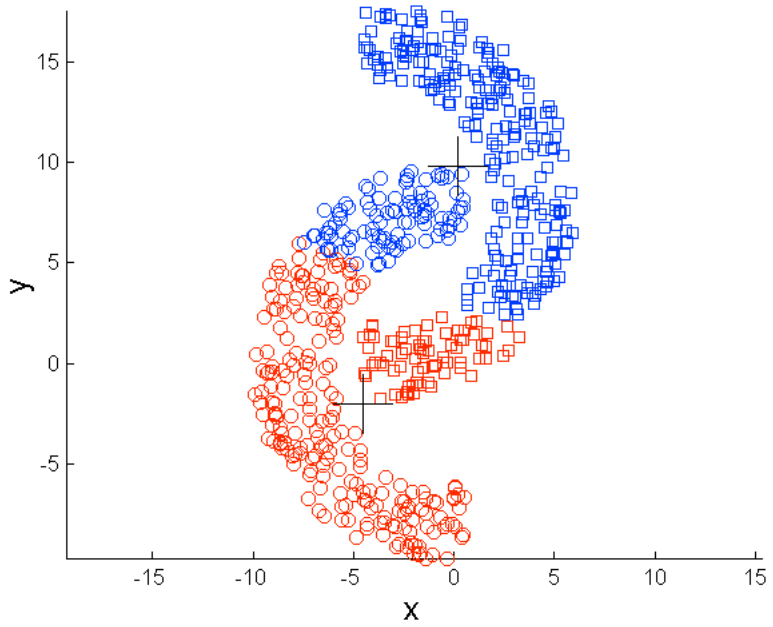
K-means (3 Clusters)

聚类分析：K-means

- K-means的局限性
  - 3. 簇的形状：当出现不规则形状的簇时（非球状），往往很难有效聚类



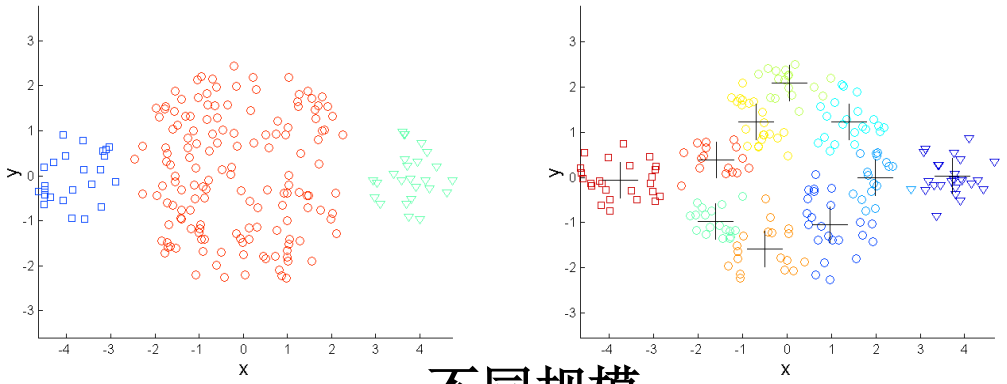
Original Points



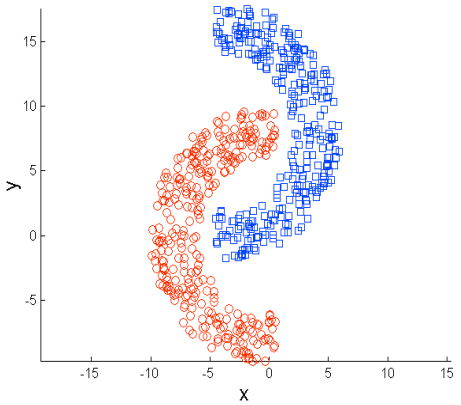
K-means (2 Clusters)

聚类分析：K-means

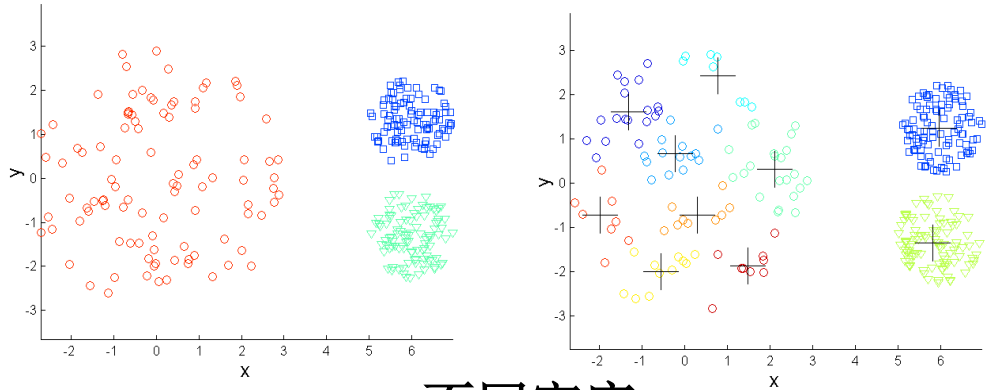
- 如何解决K-means的局限性
  - 一种解决方法：初始时增加簇的个数，然后将多个小簇合并为大簇



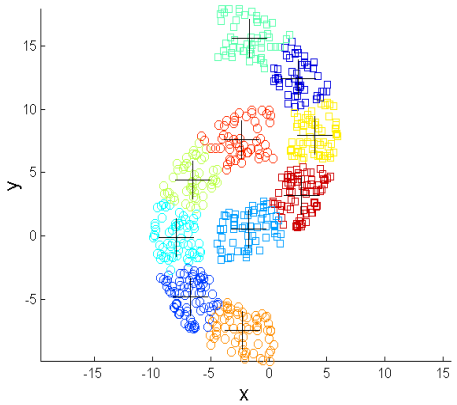
不同规模



不同形状



不同密度

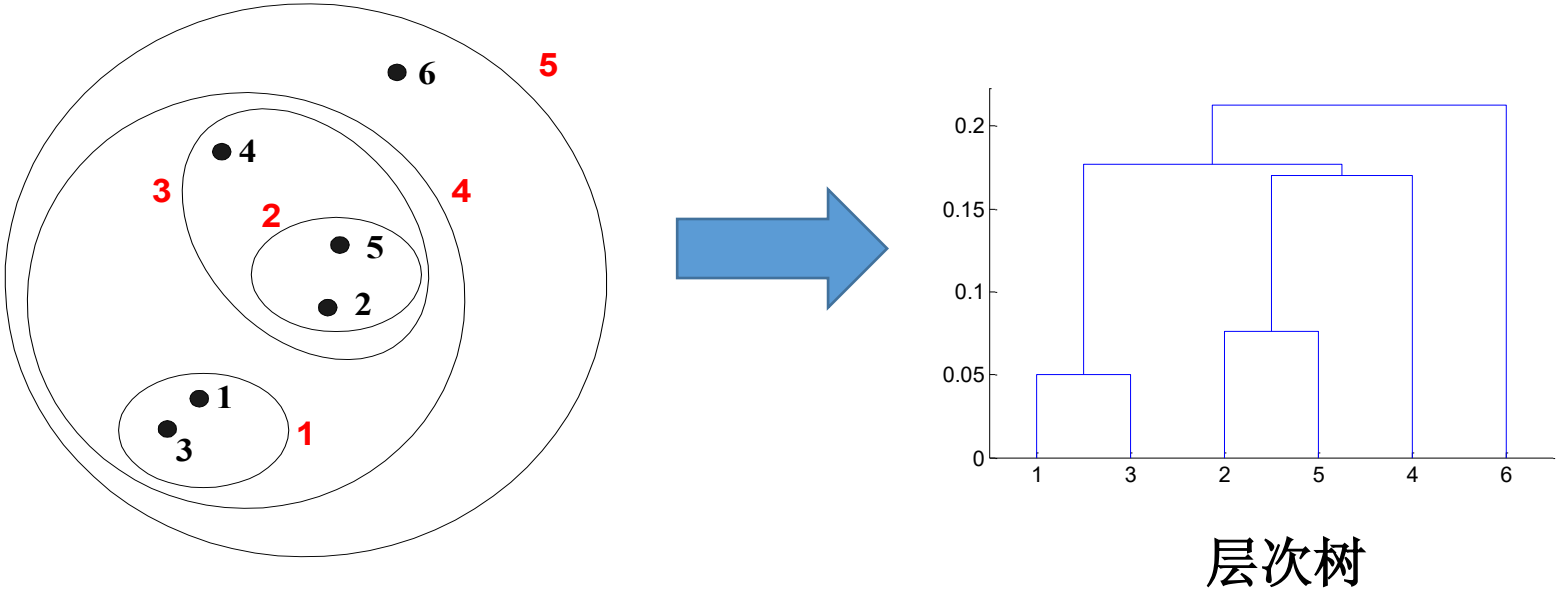


聚类分析

- 聚类方法：最常见的无监督学习算法
- 常用方法
  - K均值聚类(K-means)
  - 层次聚类(Hierarchical Clustering)
  - 密度聚类(Density-based Clustering)
  - 聚类效果验证
  - 前沿聚类方法

聚类分析：层次聚类

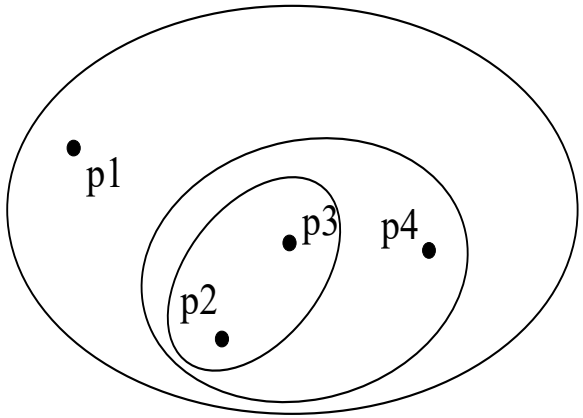
- 层次聚类(Hierarchical Clustering)
  - **特点：**生成一组嵌套的簇，表示为**层次树**
  - 整体呈树状结构，除叶节点外，每个节点是子节点的并集
  - 叶节点：一般为单个样本组成的单元簇
  - 一种树状的图表，记录合并或拆分的顺序



层次树

聚类分析：层次聚类

- 层次聚类有以下两种基本形式
  - 凝聚式聚类 (Agglomerative，自下而上聚类)
    - 1. 初始时，每个数据均视为一个簇
    - 2. 每一轮将最近的两个簇合并
    - 3. 直到只剩一个簇
  - 分裂式聚类（Divisive，自上而下聚类）
    - 1. 初始时，所有数据属于一个簇
    - 2. 每一轮将簇切分
    - 3. 直到每个簇仅包含一个样本
  - 一般而言，凝聚式聚类更为常见



思考：为什么?

聚类分析：层次聚类

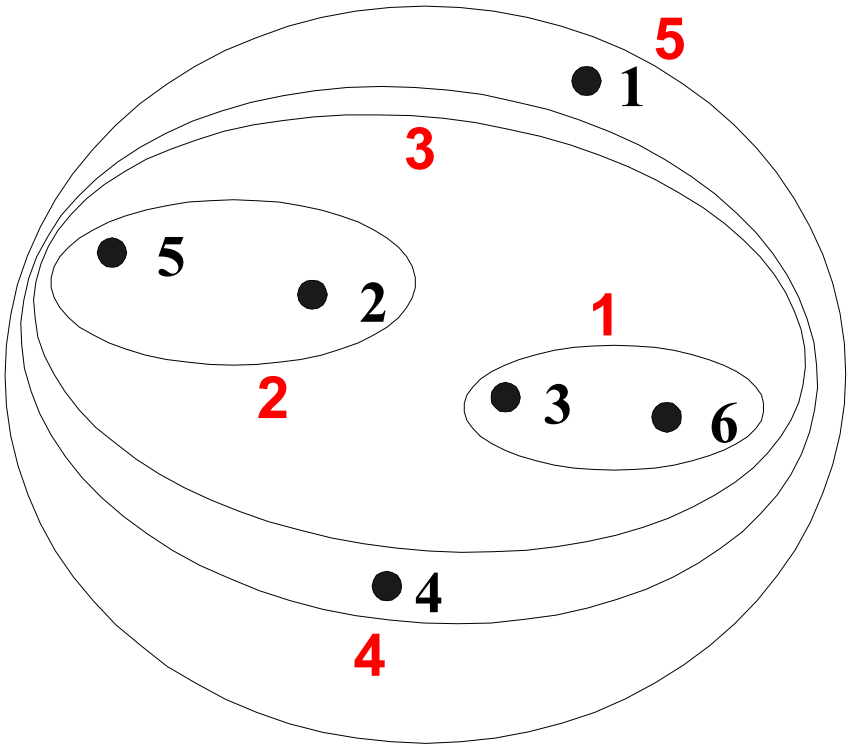
- 凝聚式层次聚类
  - 引入 **邻近度矩阵** 的概念
    - 用于存储两两簇之间的邻近度
  - 基本流程非常直观，主要迭代以下两步，直到仅剩一个簇
    - 1. 合并邻近度最高的两个簇
    - 2. 基于更新的簇重新计算距离，更新距离矩阵
  - 关键点在于计算两个簇的距离

聚类分析：层次聚类

- 凝聚式层次聚类的距离定义
  - 核心问题在于簇距离的计算，不同聚类方法计算方式不同
  - 常见的距离定义与计算方式
  - **1. 单链（Single Link）**，表示为MIN
    - 指不同簇最近的点之间的距离
    - 优势：擅长处理非椭圆形状的簇
    - 缺点：对噪声比较敏感

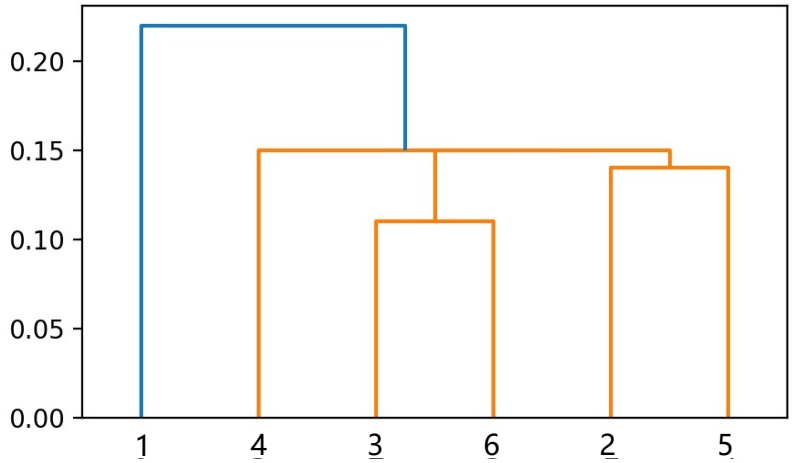
聚类分析：层次聚类

- 例子：凝聚式层次聚类——单链方式MIN



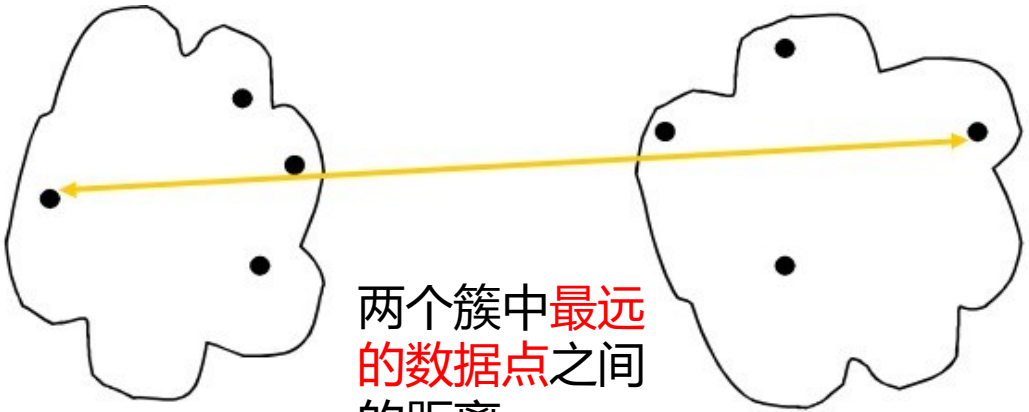
距离邻近度矩阵

	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00



聚类分析：层次聚类

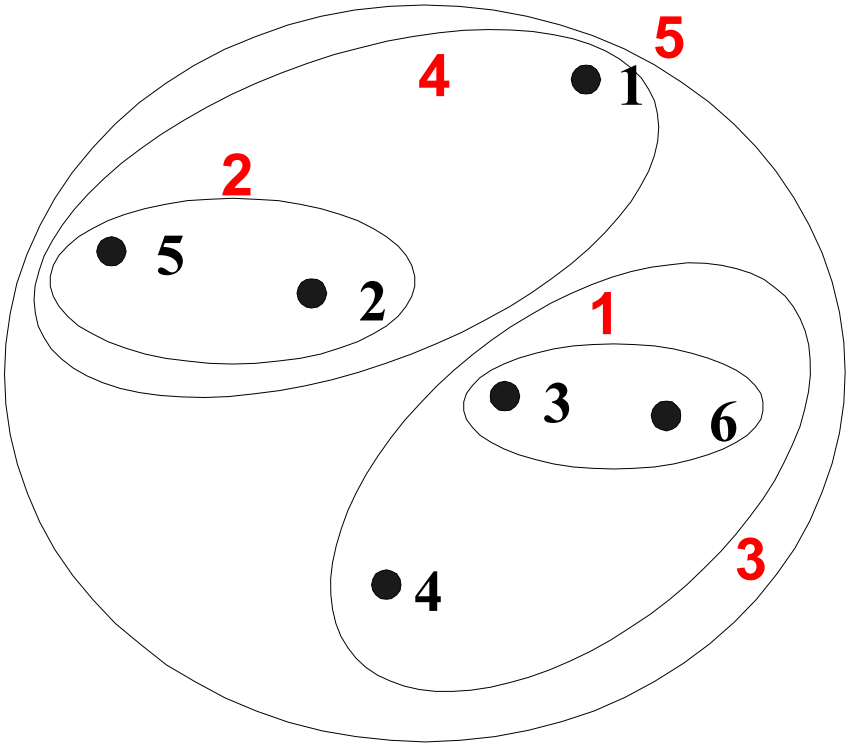
- 凝聚式层次聚类的距离定义
  - 核心问题在于距离的计算，不同聚类方法计算方式不同
  - 常见的距离定义与计算方式
  - **2. 全链（Complete Link），**表示为MAX
    - 指不同簇最远的点之间的距离
    - 优势：对噪声不太敏感
    - 缺点：可能使得较大的簇变得支离破碎



两个簇中最远的数据点之间的距离

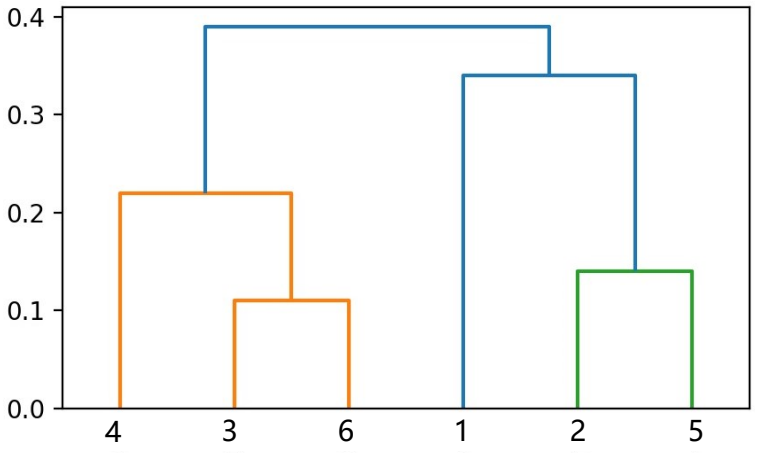
聚类分析：层次聚类

- 例子：凝聚式层次聚类——MAX(全链)



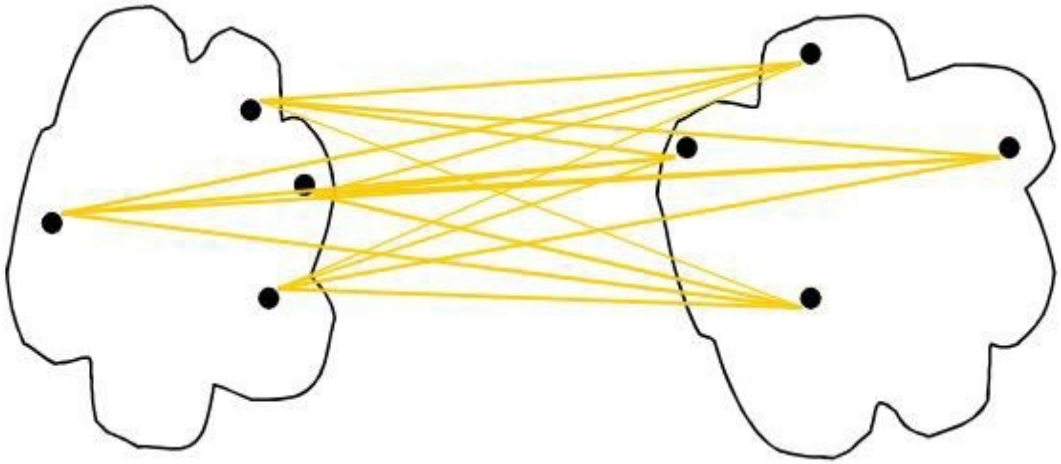
距离邻近度矩阵

	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00



聚类分析：层次聚类

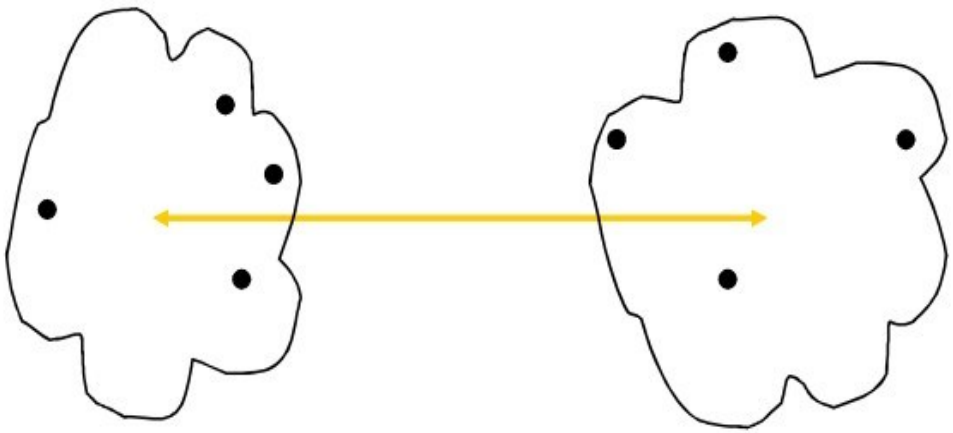
- 凝聚式层次聚类的距离定义
  - 核心问题在于距离的计算，不同聚类方法计算方式不同
  - 常见的距离定义与计算方式
  - **3. 组平均（Group Average）**
    - 所有来自不同簇的两点之间的平均距离
    - 前面两种方法的折中产物



两个簇中的数据点两两之间的平均距离

聚类分析：层次聚类

- 凝聚式层次聚类的距离定义
  - 核心问题在于距离的计算，不同聚类方法计算方式不同
  - 常见的距离定义与计算方式
  - **4. 中心距离（Group Average）**
    - 所有来自不同簇的两个簇中心之间的距离
    - 使用合并两个簇导致的SSE增加值等度量方式来衡量



两个簇中的簇中心的距离

聚类分析：层次聚类

- 层次聚类的局限性
  - 每一步的合并决策都是最终的
    - 一旦做出合并两个簇的决策，就无法撤销
  - 没有全局的优化目标函数
    - 每一步都是一个局部最优的过程
  - 不同的聚类方法（邻近度定义），或多或少都具有一些问题
    - 例如，对于噪声的敏感性，或者难以保留较大的簇等

聚类分析

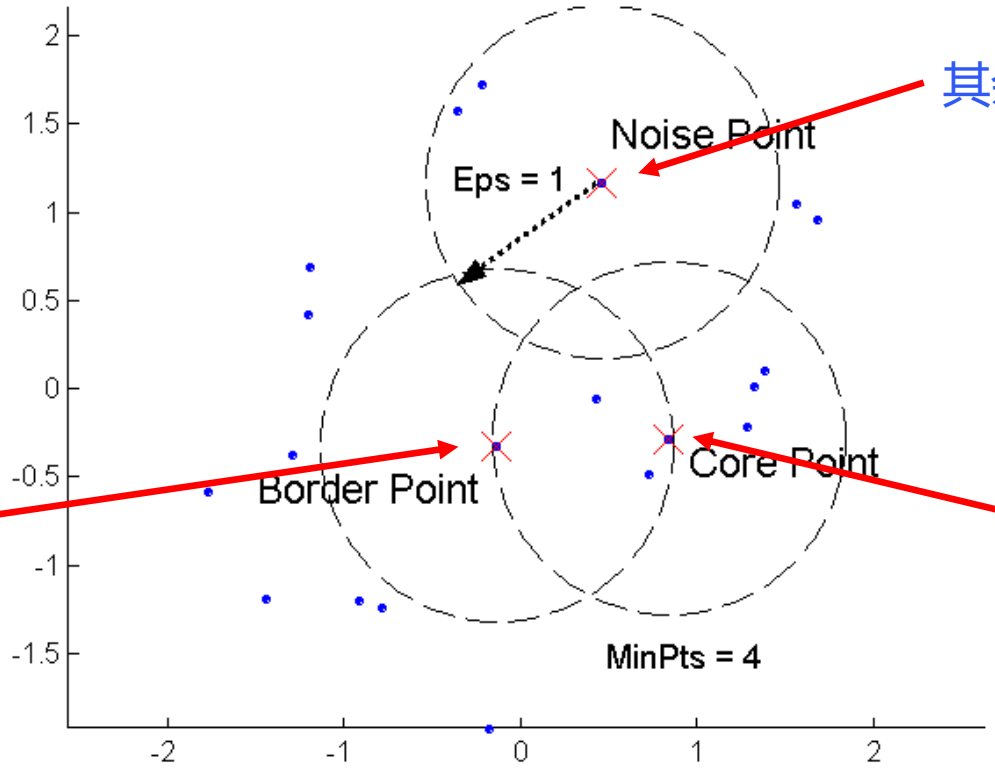
- 聚类方法：最常见的无监督学习算法
- 常用方法
  - K均值聚类(K-means)
  - 层次聚类(Hierarchical Clustering)
  - 密度聚类(Density-based Clustering)
  - 聚类效果验证
  - 前沿聚类方法

聚类分析：密度聚类

- 密度聚类
  - 基本假设：只有达到一定密度，才足以成为一个簇
  - 密度：指定样本一定半径的样本数量
    - 半径，记为Eps
    - 半径内样本数阈值，记为MinPts
- 典型算法：DBSCAN
  - 核心要素：三类不同的数据点
  - 1. 核心点(Core point)：稠密部分内部的点
    - 其Eps的范围内的样本个数不少于MinPts，这些核心点位于簇的中心
  - 2. 边界点(Border point)：非核心点，但是处于稠密区域边界内/上的点
    - 其Eps的范围内的样本个数少于MinPts，但它是某个核心点的邻居
  - 3. 噪音点(Noise point)：处于稀疏区域的点
    - 除核心点和边界点之外的样本

聚类分析：密度聚类

- DBSCAN
  - 三类点：核心点、边界点和噪音点 示意图



其余点

半径内3个点，
小于阈值4
是某个核心点
的邻居

半径内6个点，
大于阈值4

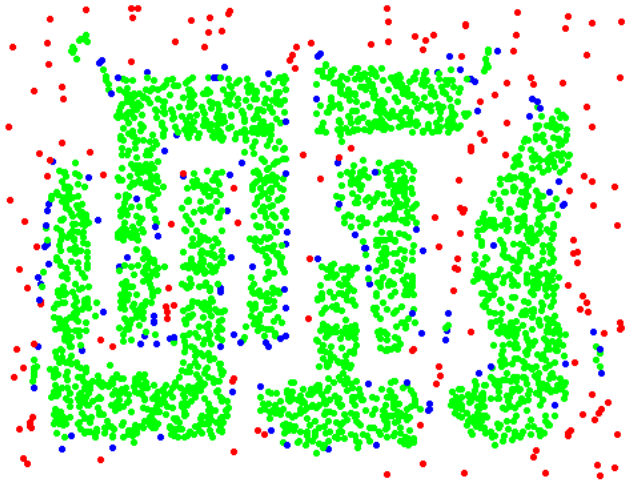
聚类分析：密度聚类

- DBSCAN的基本流程可归纳如下
  - 1. 将所有节点区分为核心点、边界点或噪声点
  - 2. 删除噪声点
  - 3. 将所有距离在预定半径内的核心点之间连一条边
  - 4. 连通的核心点形成一个簇
  - 5. 将所有的边界点指派到一个与之关联的核心点所在的簇中

聚类分析：密度聚类

- DBSCAN实例
  - 半径Eps = 10, 阈值MinPts = 4

Original Points



Point types:

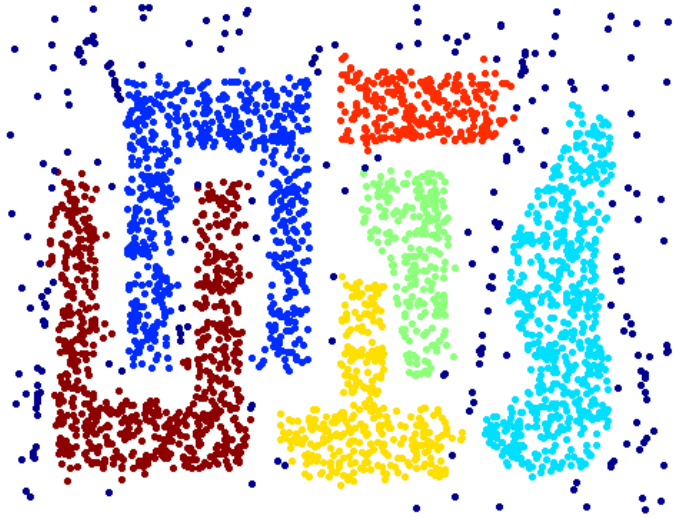
绿色core, 蓝色border, 红色noise

聚类分析：密度聚类

- DBSCAN的优势
  - 对噪声鲁棒
  - 能够处理不同形状和大小的簇

周边的噪声除去，内部的数据很好的聚类

Original Points

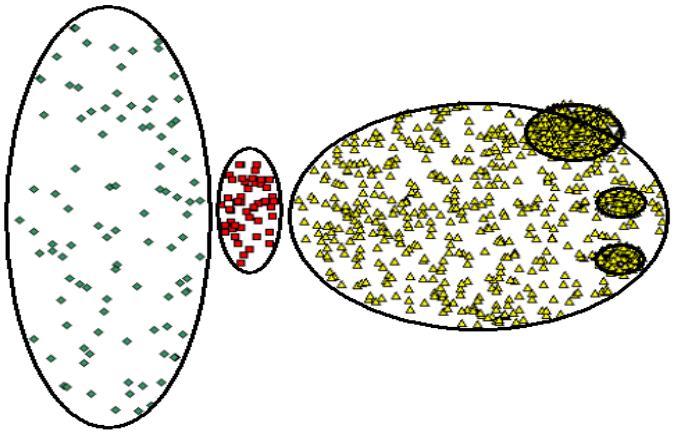


Clusters

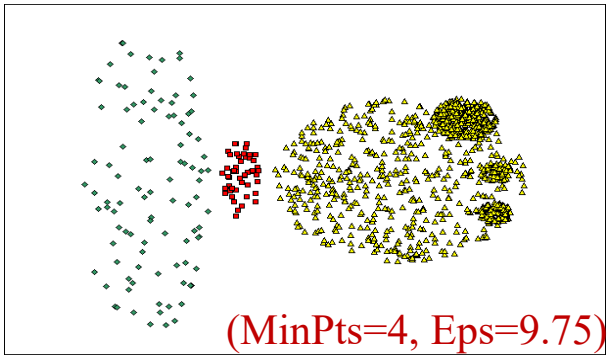
聚类分析：密度聚类

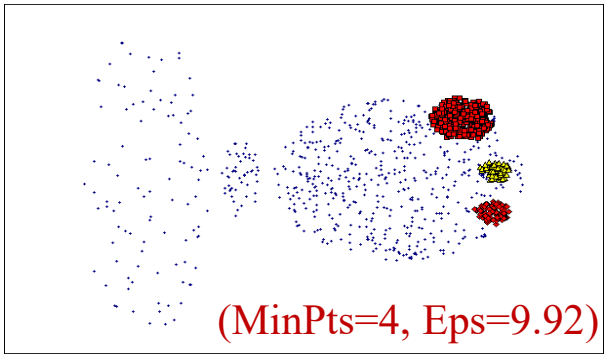
- DBSCAN的局限性
  - 簇的密度变化使得DBSCAN的效果可能会受到影响
  - 参数难以设置：半径Eps、阈值MinPts的选取需与数据维度匹配

例子：两种方式参数相近，但簇的密度完全不同，**DBSCAN**的结果差距很大



Original Points





聚类分析：密度聚类

- DBSCAN算法作者获得ICDM2013 Research Contributions Award

TITEL	ZITIERT VON	JAHR
A density-based algorithm for discovering clusters in large spatial databases with noise. M Ester, HP Kriegel, J Sander, X Xu kdd 96 (34), 226-231	22965	1996

聚类分析

- 聚类方法：最常见的无监督学习算法
- 常用方法
  - K均值聚类(K-means)
  - 层次聚类(Hierarchical Clustering)
  - 密度聚类(Density-based Clustering)
  - 聚类效果验证
  - 前沿聚类方法

聚类分析

- 聚类效果验证
  - 作为无监督学习，聚类问题并没有天然标签，如何评估聚类结果？
  - 首先，我们需要了解，为什么需要评估聚类结果的“好”与“坏”
    - 确定数据集的聚类趋势，确定是否真的有群体性
    - 确定合理的簇的个数
    - 比较两个簇，或者比较两种方法的聚类，看哪种结果更合适
    - 将聚类的簇与已知的客观信息进行比较
      - 例如，外部提供的标签、Query等

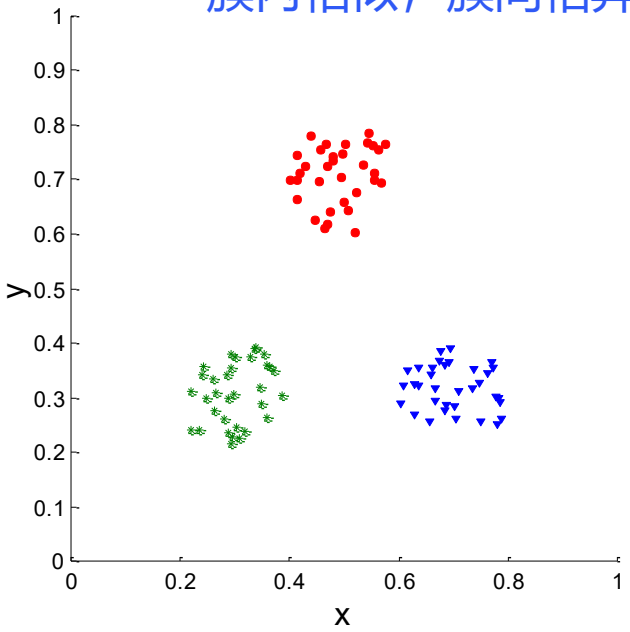
聚类分析

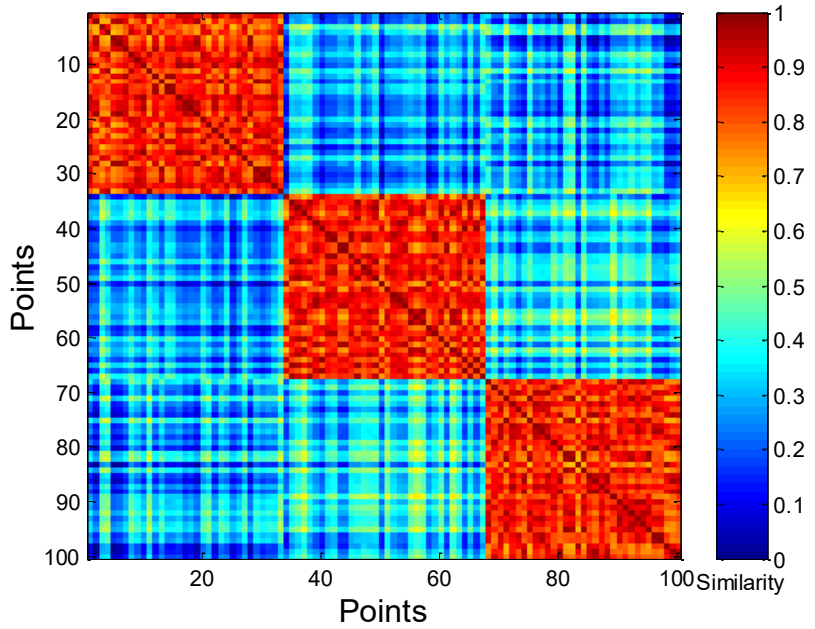
- 聚类效果验证
  - 一般而言，聚类问题的评估标准可以分为以下三类
    - 非监督评估（或内部评估）:仅使用数据本身的特性，而不考虑任何外部标签信息
      - 例如：距离矩阵，SSB(分离度：簇质心m_i到数据点均值m的距离平方和
        $SSB = \sum_{i=1}^{K} |C_i|(m - m_i)^2$，$|C_i|$是簇$i$的大小, m是所有数据点的总均值
    - 有监督评估（或外部评估）:引入外部信息，衡量聚类结构与外部结果的匹配程度
      - 例如：Entropy，Jaccard系数，准确（Precision）、召回（Recall）、F值等
    - 相对评估：主要用于比较两个簇或者两个聚类结果
      - 常常需要外部或内部指标结合, e.g., SSE or entropy

聚类分析

- 方式1：非监督评估:—基于邻近度矩阵
  - 理想的聚类结果是：簇内的点邻近度全为1，簇之间的邻近度全为0
  - 通过邻近度矩阵，可以可视化地评估聚类结果的好坏
    - 通过观察相似度矩阵是否体现出对角模式，可以大致判断结果好坏

簇内相似，簇间相异





聚类分析

- 方式2：有监督评估—基于Jaccard系数
  - 理想的聚类结果是：在邻近度矩阵中
    - 同一个类中的样本，对应的矩阵元素为1
    - 不同类中的样本，对应的矩阵元素为0
  - 通过比较两个“理想”矩阵之间的相关性，可以近似估计聚类结果

f_{00} =具有不同的类和不同的簇的对象对的个数

f_{01} =具有不同的类和相同的簇的对象对的个数

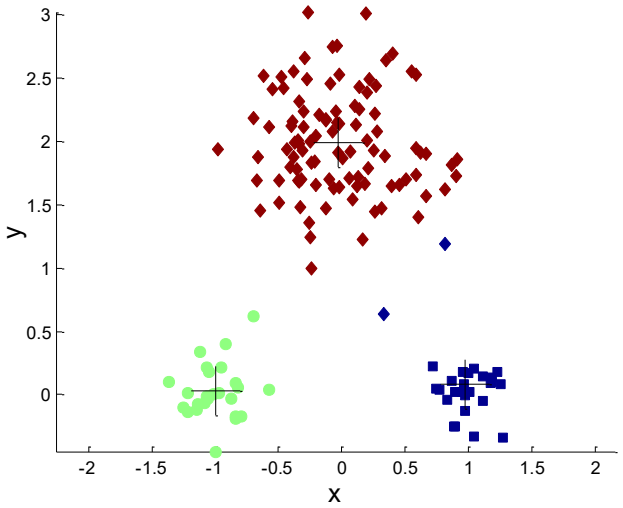
f_{10} =具有相同的类和不同的簇的对象对的个数

f_{11} =具有相同的类和相同的簇的对象对的个数

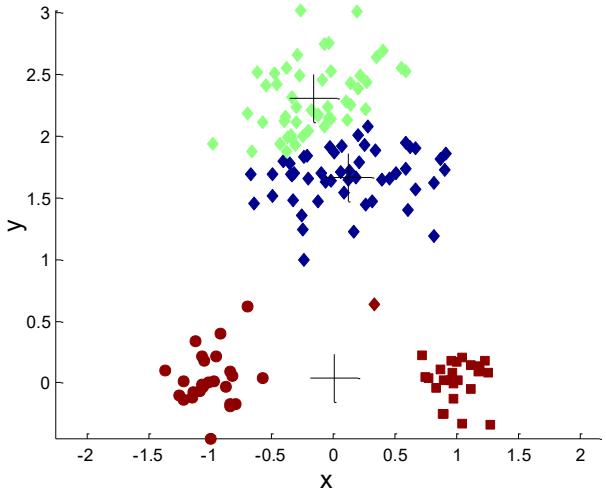
(回顾第2章：数据集成) $$\text{Jaccard 系数} = \frac{f_{11}}{f_{01} + f_{10} + f_{11}}$$

聚类分析

- 方式3：相对评估—基于SSE
  - 对同一样本集合，SSE较小的聚类结果更好



优于



SSE小

SSE大

簇数K小

聚类分析

- Y Liu, Z Li, H Xiong, X Gao, J Wu, “**Understanding of internal clustering validation measures**”. **ICDM 2010.**
- Yanchi Liu, Zhongmou Li, Hui Xiong, Xuedong Gao, Junjie Wu, Sen Wu, ” **Understanding and Enhancement of Internal Clustering Validation Measures**”, **IEEE Transactions on Cybernetics (TC)**, Vol. 43, No. 3, pp. 982-994, 2013.
- J Wu, H Xiong, J Chen, “**Adapting the right measures for k-means clustering**”. **KDD 2009.**

“The validation of clustering structures is the most difficult and frustrating part of cluster analysis. Without a strong effort in this direction, cluster analysis will remain a black art accessible only to those true believers who have experience and great courage.”

----------*Algorithms for Clustering Data*, Jain and Dubes

聚类分析

- 聚类方法：最常见的无监督学习算法
- 常用方法
  - K均值聚类(K-means)
  - 层次聚类(Hierarchical Clustering)
  - 密度聚类(Density-based Clustering)
  - 聚类效果验证
  - 前沿聚类方法

聚类分析

- **前沿聚类方法 — 课外学习**
  - Prototype-based(基于原型的聚类)
    - Fuzzy K-means
    - Mixture Model Clustering
    - Self-Organizing Maps
  - Density-based(基于密度的聚类)
    - Grid-based clustering
    - Subspace clustering
  - Graph-based (基于图的聚类)
    - Chameleon
    - Jarvis-Patrick
    - Shared Nearest Neighbor (SNN)

总结：聚类分析

- 聚类方法：最常见的无监督学习算法
- 常用方法
  - K均值聚类(K-means)
  - 层次聚类(Hierarchical Clustering)
  - 密度聚类(Density-based Clustering)
  - 聚类效果验证
  - 前沿聚类方法