智能制造过程大数据技术

Big Data Technology in Intelligent Manufacturing Process

第五讲：聚类分析

Lecture 5: Clustering analysis

丁敏 dingmin@cug.edu.cn

中国地质大学(武汉) 自动化学院

School of Automation, China University of Geosciences

目录

©ISTOCKPHOTO.COM/ZAPP2PHOTO
IEEE INDUSTRIAL ELECTRONICS MAGAZINE ■ DECEMBER 2019 PP.38

目录

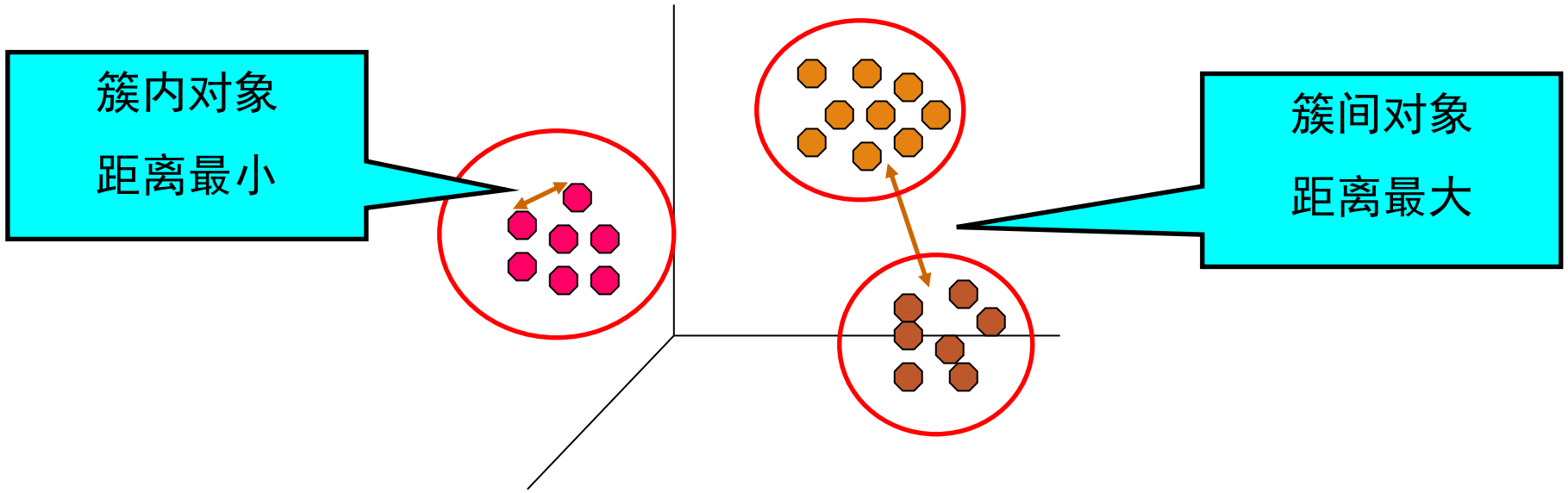
©ISTOCKPHOTO.COM/ZAPP2PHOTO

IEEE INDUSTRIAL ELECTRONICS MAGAZINE ■ DECEMBER 2019 PP.38

1 聚类分析的基本概念

➢ 聚类分析的定义

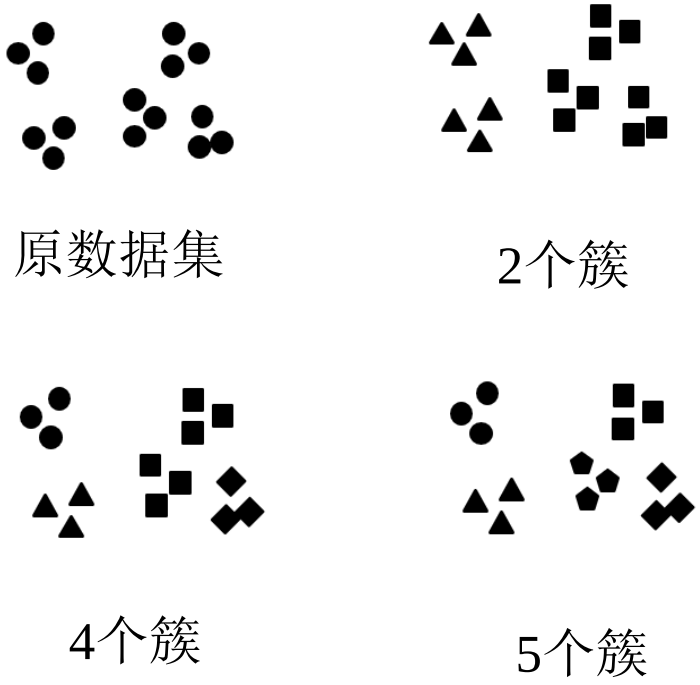
- 聚类：将物理或抽象对象的集合分成由类似对象组成的多个类或者簇的过程
- 度量方法：对象之间的相似度或距离
- 簇：聚类所生的一组对象的集合



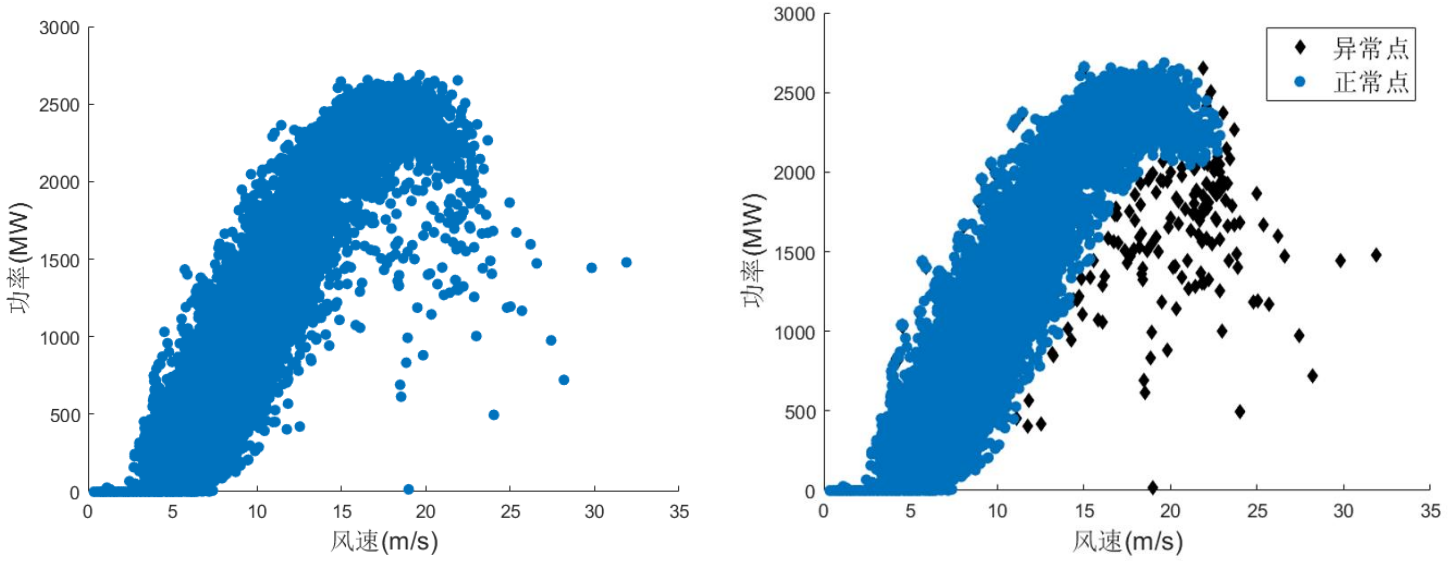
1 聚类分析的基本概念

➢ 聚类分析的定义

◘ 相同数据集的不同聚类方法



◘ 风电功率数据异常点检测



1 聚类分析的基本概念

➢ 聚类分析的定义

◘ 聚类目标

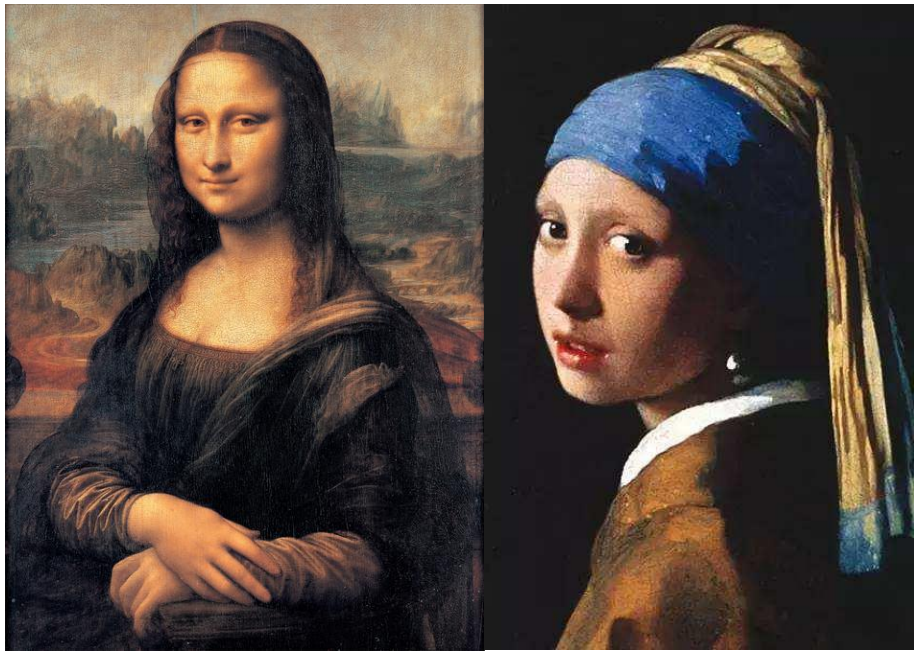
- 同一簇中对象相似度较高或距离较近
- 其他簇中的对象相似度较低或距离较远

◘ 聚类分析应用

- 数据预处理
- 工况识别、异常检测
- 故障诊断、故障预测、负荷预测

1 聚类分析的基本概念

- 在未知样本类别的情况下，通过计算样本彼此间的相似性来估计样本所属类别



1 聚类分析的基本概念

➢ 聚类算法的性能要求

- 伸缩性：聚类结果准确度不会因为数据量的大小而变化
- 处理不同字段类型的能力：数值型、二元的、对称的、序数的、图、序列等
- 发现具有任意形状的聚类的能力：球状簇、凸型簇、其他任意形状簇
- 能够处理异常数据：群点、缺失值、未知或错误的数据
- 增量聚类和对输入次序的不敏感：针对不用次序的数据输入，聚类结果不变

1 聚类分析的基本概念

➢ 聚类算法的性能要求

- 处理高维数据的能力：避免只能处理低维数据
- 初始化参数的需求最小化：减少用户提供的初始化参数
- 可解释性和可用性：聚类结果易解释和理解，并且可应用

单选题 5分

设置

聚类分析是一种无监督的学习方式，但聚类前就已经知道分类对象可以分为哪几种具体的类了，只是需要通过聚类确定样本应该归属于哪些类而已

A √

B ×

提交

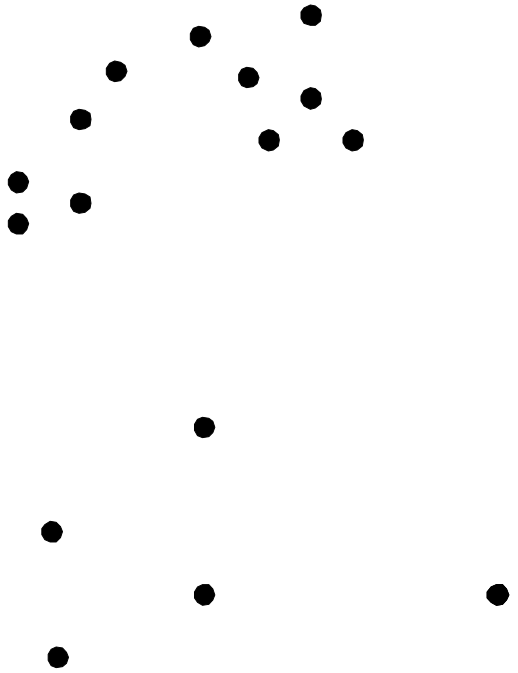
目录

©ISTOCKPHOTO.COM/ZAPP2PHOTO
IEEE INDUSTRIAL ELECTRONICS MAGAZINE ■ DECEMBER 2019 PP.38

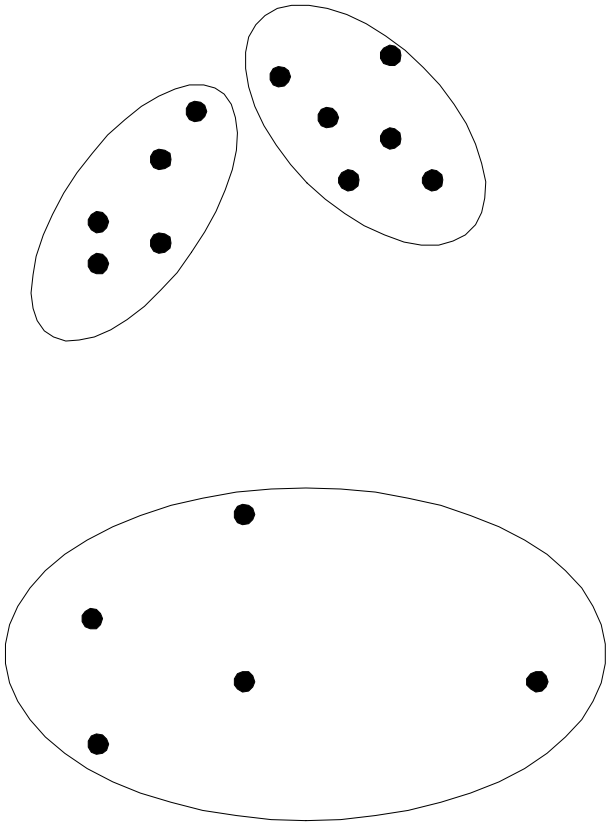
2 划分聚类方法

➢ 划分聚类方法

- 划分聚类：简单地将数据对象集划分成不重叠的子集，使得每个数据对象恰在一个子集

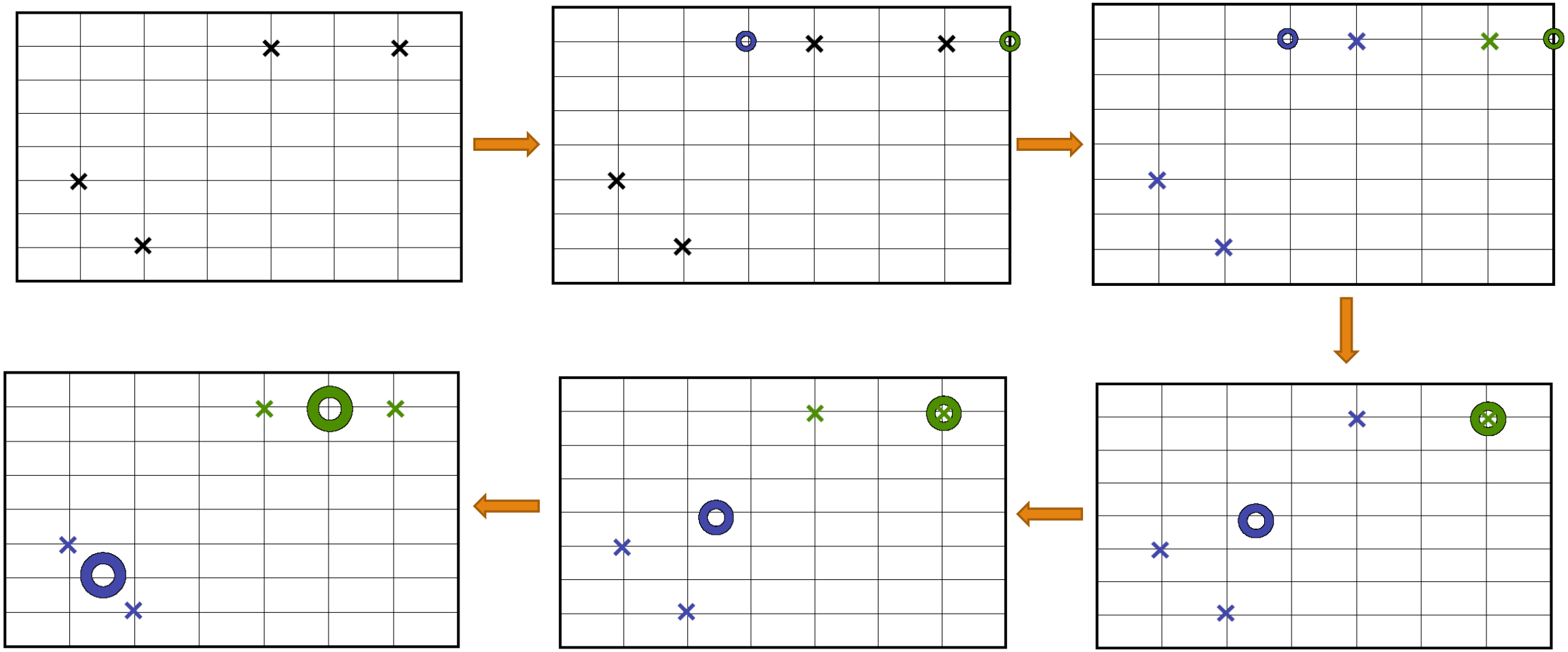


Original Points



A Partitional Clustering

- 划分聚类方法



➢ *K*-means算法

- 最常用也最为经典的一种基于划分的聚类算法
- 基本思想：选定初始聚类中心后，选取距离作为相似度指标，将样本进行划分类。聚类目标是使得误差平方和函数最小，即最小化：

$$SSE=\sum_{i=1}^{k}\sum_{x\in C_i} d(x,c_i)^2$$

  - x是集合D的对象，C_i代表第i个簇，c_i是C_i的中心，$c_i=\frac{1}{m_i}\sum_{x\in C_i}x$，$m_i$是$C_i$中心数据对象的个数

2 划分聚类方法

➢ *K*-means算法

◘ *K*-means算法的过程：

- 从*n*个样本点中任意选择（一般是随机分配）*k*个对象作为初始簇质心；
- 对于剩下的其他样本点，计算它们与各簇质心的距离，分别将它们划分到距离其最近的簇；
- 更新每个新簇的质心；
- 重复上述两个步骤，直到簇中对象不再变化

输入：所期望的簇的数目*k*，包含*n*个对象的数据集D

输出：*k*个簇的集合

1. 从D中任意选择*k*个对象作为初始簇中心；
2. repeat
3. 将每个点指派到最近的中心，形成*k*个簇；
4. 重新计算每个簇的中心；
5. until 质心不再发生变化；

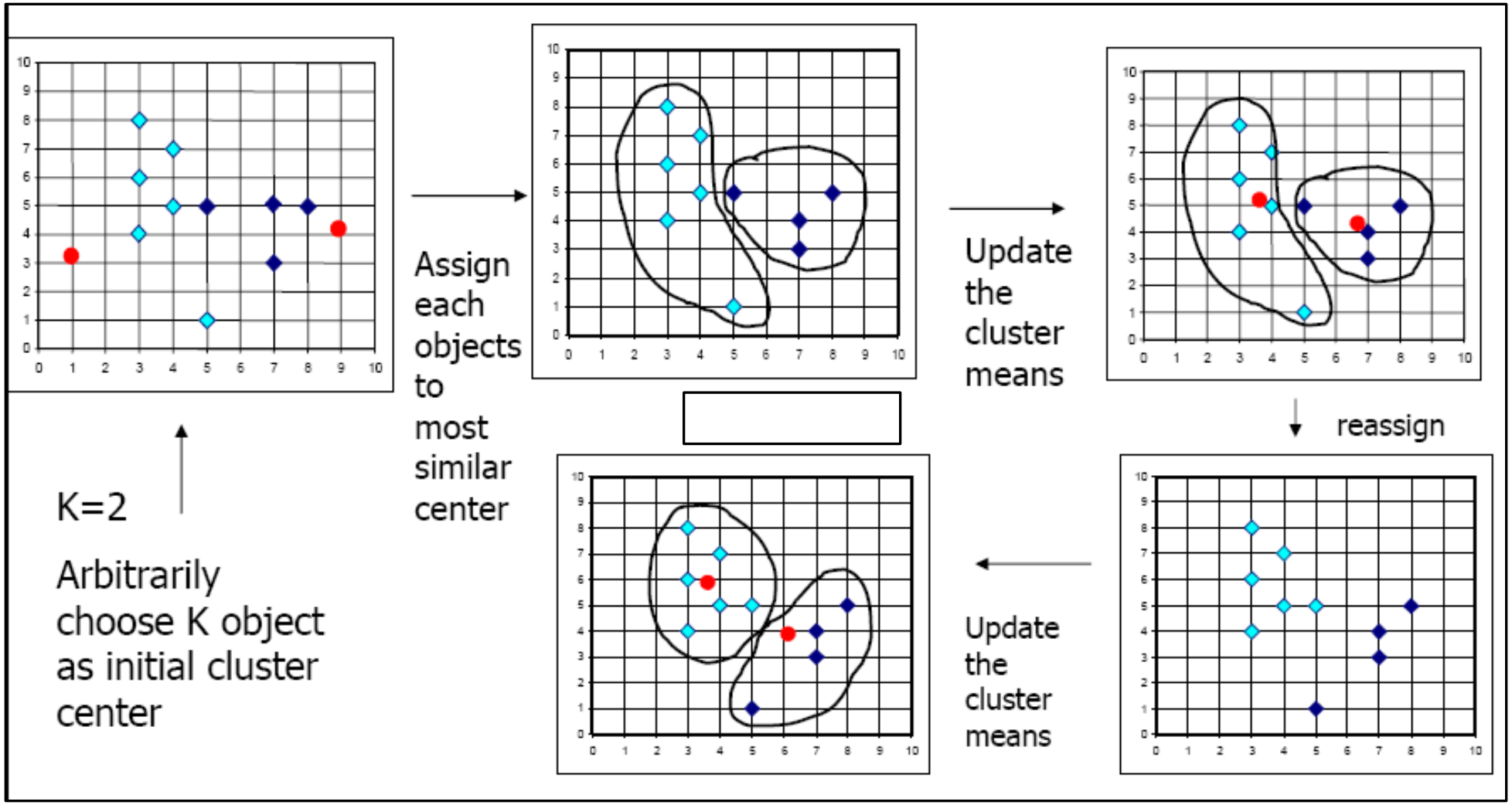
算法分析：

（1）步骤3通过将每个点指派到最近的中心形成簇，最小化关于给定中心的目标函数*E*；

（2）步骤4重新计算每个簇的中心

2 划分聚类方法

- *K*-means算法实例



➢ *K*-means算法实例

- 数据对象集合S见右表，作为一个聚类分析的二维样本，要求的簇的数量k=2。
- 选择 $O_1(0,2)$ ，$O_2(0,0)$ 为初始的簇中心，即 $M_1=O_1(0,2)$ ，$M_2=O_2(0,0)$ 。
- 对剩余的对象，根据其与各个簇中心的距离，将它赋给最近的簇。

O	x	y
1	0	2
2	0	0
3	1.5	0
4	5	0
5	5	2

- 对于 O_3：$d(M_1,O_3)=\sqrt{(0-1.5)^2+(2-0)^2}=2.5$

  $d(M_2,O_3)=\sqrt{(0-1.5)^2+(0-0)^2}=1.5$
- 显然，$d(M_2,O_3)<d(M_1,O_3)$ ，故将 O_3 分配给 C_2
- 对于 O_4：$d(M_1,O_4)=\sqrt{(0-5)^2+(2-0)^2}=\sqrt{29}$

  $d(M_2,O_4)=\sqrt{(0-5)^2+(0-0)^2}=5$
- 因为，$d(M_2,O_3)<d(M_1,O_3)$ ，故将 O_4 分配给 C_2

➢ *K*-means算法实例

- 对于 O_5：$d(M_1,O_5)=\sqrt{(0-5)^2+(2-2)^2}=5$

  $d(M_2,O_5)=\sqrt{(0-5)^2+(0-2)^2}=\sqrt{29}$

- 因为 $d(M_1,O_5)<d(M_2,O_5)$，故将 O_5 分配给 C_1

- 更新，得到新簇 $C_1=\{O_1,O_5\}$ 和 $C_2=\{O_2,O_3,O_4\}$

- 计算平方误差准则，单个方差为

  $E_1=\left[(0-0)^2+(2-2)^2\right]+\left[(0-5)^2+(2-2)^2\right]=25 \quad M_1=O_1=(0,2)$

  $E_2=27.25 \quad M_2=O_2=(0,0)$

- 总体平均方差是：$E=E_1+E_2=25+27.25=52.25$

O	x	y
1	0	2
2	0	0
3	1.5	0
4	5	0
5	5	2

➢ *K*-means算法实例

- 计算新的簇的中心 $M_1=\left((0+5)/2,(2+2)/2\right)=(2.5,2)$

$$M_2=\left((0+1.5+5)/3,(0+0+0)/3\right)=(2.17,0)$$

O	x	y
1	0	2
2	0	0
3	1.5	0
4	5	0
5	5	2

- 重复步骤2、3，得到 O_1 分配给 C_1；O_2 分配给 C_2，O_3 分配给 C_2 ，O_4 分配给 C_2 ，O_5 分配给 C_1。更新，得到新簇 $C_1=\{O_1,O_5\}$ 和 $C_2=\{O_2,O_3,O_4\}$ 中心为 $M_1=\{2.5,2\}$，$M_1=\{2.17,0\}$。
- 单个方差分别为 $E_1=\left[(0-2.5)^2+(2-2)^2\right]+\left[(2.5-5)^2+(2-2)^2\right]=12.5$

$$E_2=13.15$$

- 总体平均误差为 $E=E_1+E_2=12.5+13.15=25.65$

由上可以看出，第一次迭代后，总体平均误差值 52.25~25.65，显著减小。由于在两次迭代中，簇中心不变，所以停止迭代过程，算法停止。

单选题 5分

设置

簇的数量k=2,选择 $O_1(0,2)$,$O_5(5,2)$ 为初始的簇中心，$M_1=O_1(0,2)$ 即 $M_2=O_5(5,2)$， 分类结果为

A $C_1=\{O_1,O_2\}$ $C_2=\{O_3,O_4,O_5\}$

B $C_1=\{O_1,O_2,O_3\}$ $C_2=\{O_4,O_5\}$

C $C_1=\{O_1,O_5\}$ $C_2=\{O_2,O_3,O_4\}$

D $C_1=\{O_1,O_2,O_5\}$ $C_2=\{O_3,O_4\}$

O	x	y
1	0	2
2	0	0
3	1	0
4	5	0
5	5	2

提交

➢ *K*-means算法实例

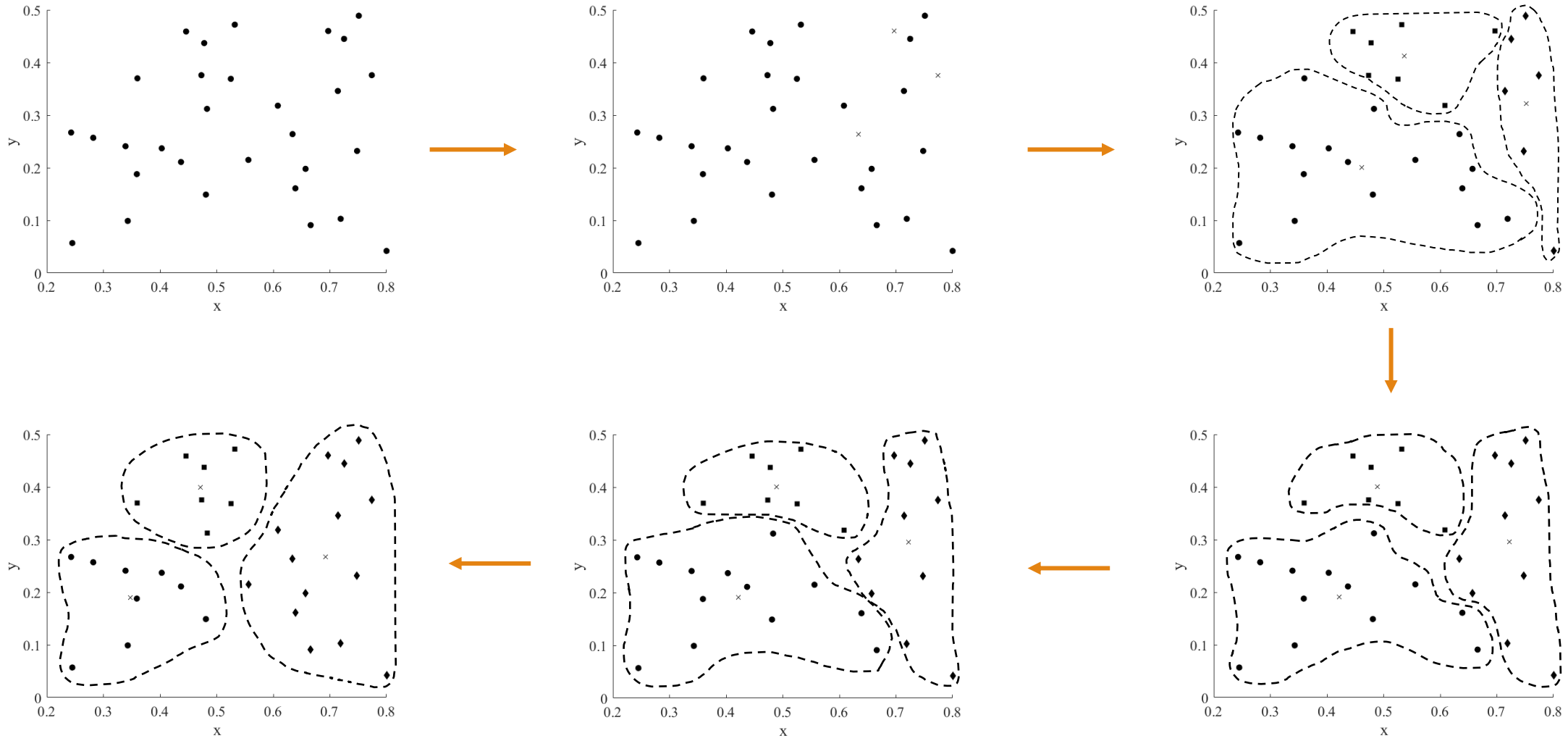
点	属性1	属性2	点	属性1	属性2	点	属性1	属性2
1	0.697	0.460	11	0.245	0.057	21	0.748	0.232
2	0.774	0.376	12	0.343	0.099	22	0.714	0.346
3	0.634	0.264	13	0.639	0.161	23	0.483	0.312
4	0.608	0.318	14	0.657	0.198	24	0.478	0.437
5	0.556	0.215	15	0.360	0.370	25	0.525	0.369
6	0.403	0.237	16	0.800	0.042	26	0.751	0.489
7	0.481	0.149	17	0.719	0.103	27	0.532	0.472
8	0.437	0.211	18	0.359	0.188	28	0.473	0.376
9	0.666	0.091	19	0.339	0.241	29	0.725	0.445
10	0.243	0.267	20	0.282	0.257	30	0.446	0.459

属性1为铣床电流传感器的采集数据

属性2为铣床振动传感器的采集数据

- 步骤1：根据*K*-means算法步骤，首先任意选择3个对象（点1、点2、点3）作为初始的簇中心，其中簇中心用“×”标记。根据与簇中心的距离（假设采用欧式距离），每个对象被分配到最近的一个簇
- 步骤2：更新簇中心。即根据簇中的当前对象，重新计算每个簇的均值。使用这些新的簇中心，把对象重新分布到里簇中心最近的簇中
- 重复这一过程，直到对象的重新分配不再发生，处理结果结束，输出聚类结果

➢ *K*-means算法实例



多选题 5分

设置

可以作为K均值聚类算法迭代终止条件的有

A 最大迭代次数

B 聚类结果不再改变

C 类别个数不再改变

D 类中心不再改变

提交

2 划分聚类方法

➢ *K*-means算法

优点：

（1）计算简单、快速

（2）算法的时间复杂度和空间复杂度都相对较低

（3）当结果簇是密集的，而簇与簇之间的区别明显时，它的效果较好

（4）对处理大数据集，该算法是相对可伸缩和高效率的

（5）复杂度是$O(nkt)$，其中n是样本量，k是簇的数目，d是数据对象的维度。通常$k<<n$且$t<<n$

缺点：

（1）对参数k和初始中心点的选择比较敏感

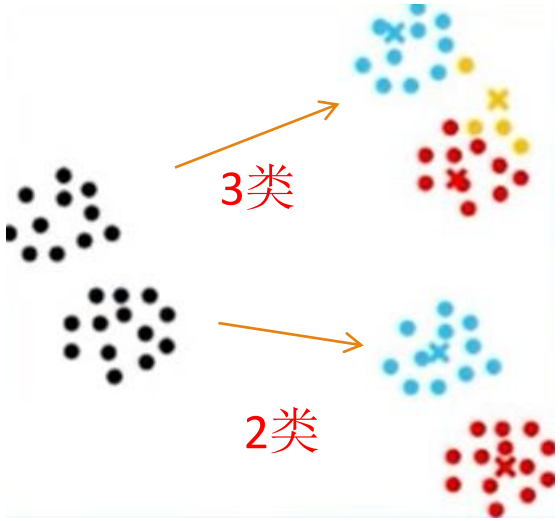
（2）在连续属性的数据集上容易实现，但在具有离散性的数据集上却不能适用

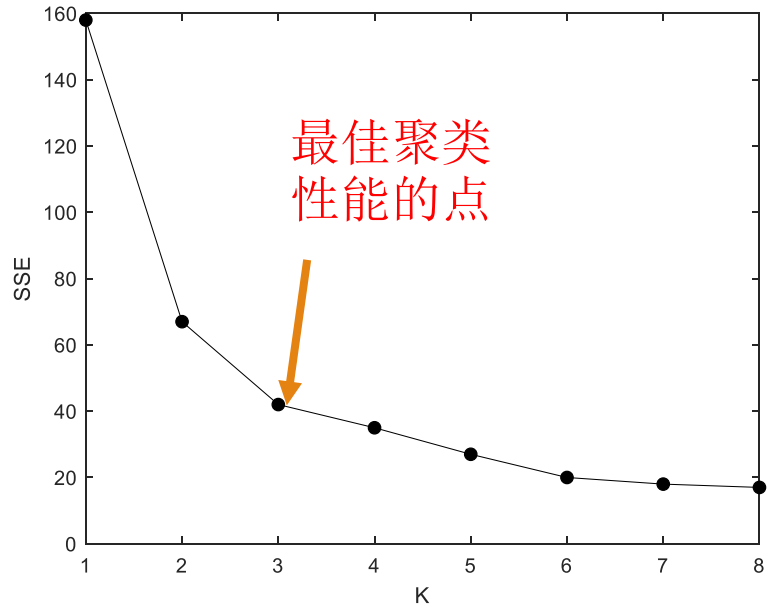
（3）主要发现圆形或者球形簇，对不同形状和密度的簇效果不好

（4）对“噪声”和离群点数据是敏感的，少量的该类数据能够对平均值产生极大的影响

➢ *K*-means算法类别个数*k*的影响

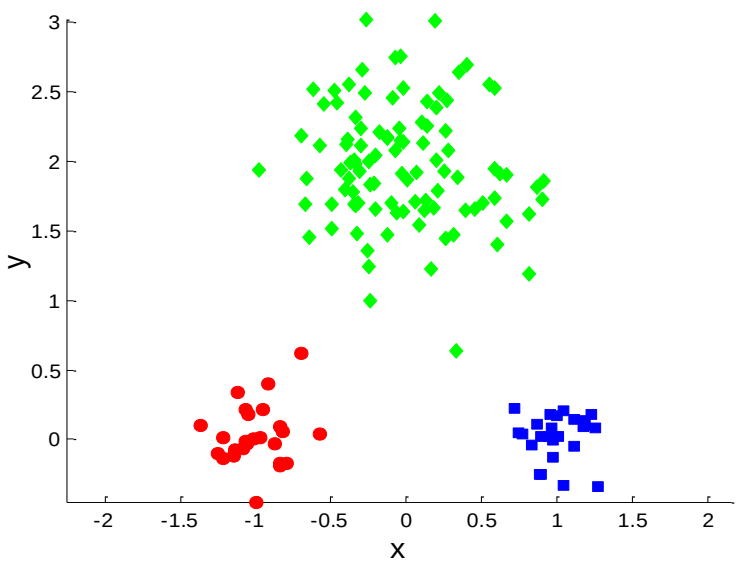
- 最优的类别个数未知，不同的类别个数会导致不一样的结果
- 最常用的最佳聚类簇*k*的选取方法：肘部法
  - 最佳聚类性能的点：SSE随着*K*的增大而减小，并且在达到第一个临界转折点处，SSE减小速度变缓，这个临界转折点就可以考虑为最佳聚类性能的点，该点对应的*K*就是最佳聚类数



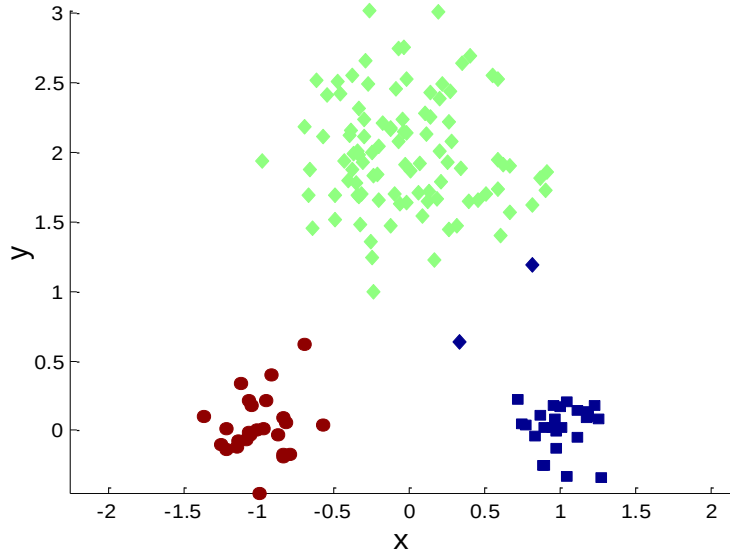


K-means算法不同初始类中心的影响

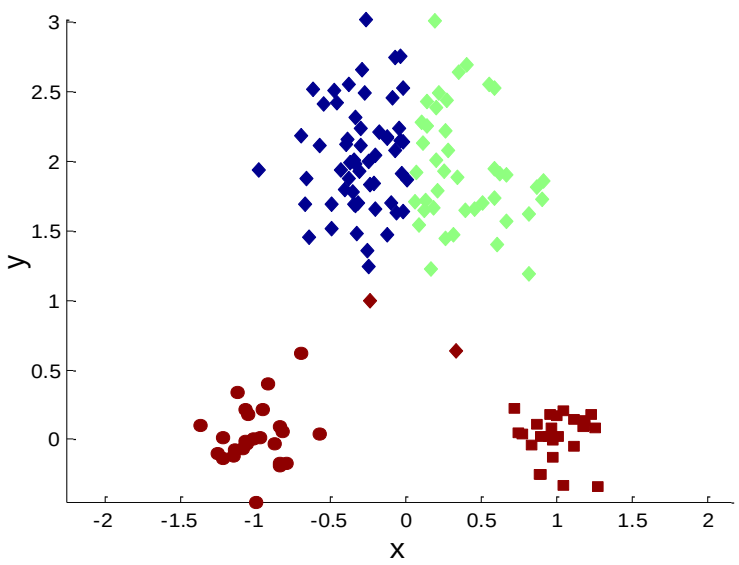
- 不同的初始质心：由于无法保证收敛到全局最优，初始类中心的位置会影响聚类结果



原始点



最优聚类



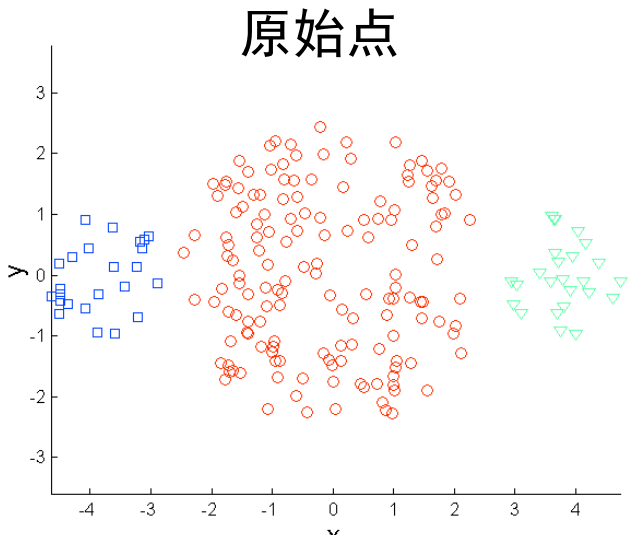
次最优聚类

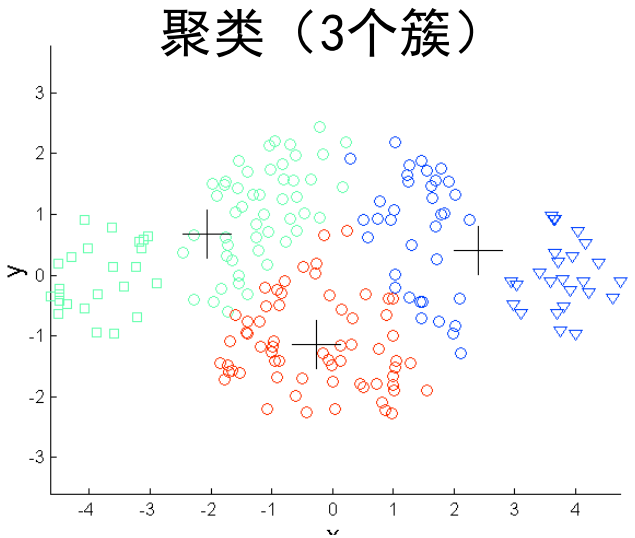
➢ *K*-means算法不同类中心的影响

- 选取初始类中心的方法：
  - ✓ 在相互间隔超过某个指定最小距离的前提下，随机选取k个个体
  - ✓ 选择数据集前k个相互间隔超过某指定最小距离的个体
  - ✓ 选择k个相互距离最远的个体
  - ✓ 选个k个等距离网格点，可能不是数据集的点
- 簇均值被定义的情况下才能使用，无法直接处理离散型数据

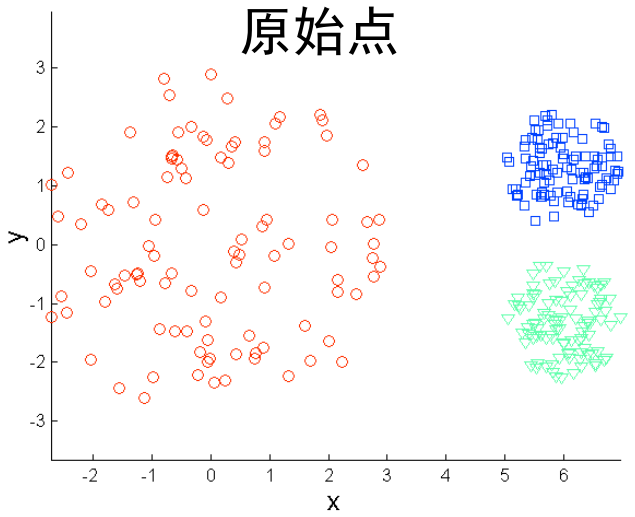
➢ *K*-means算法以误差平方和最小化为优化目标的局限性

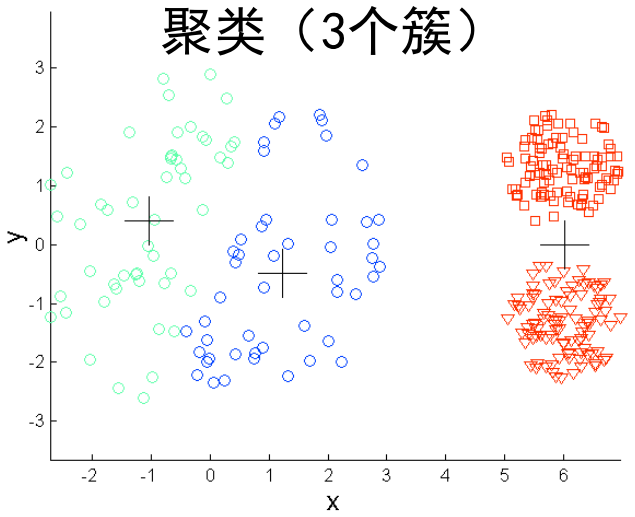
- 不同尺度的簇





- 不同密度的簇

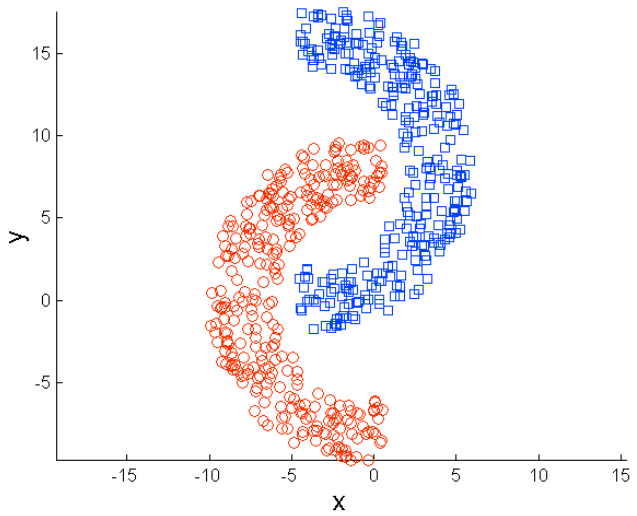




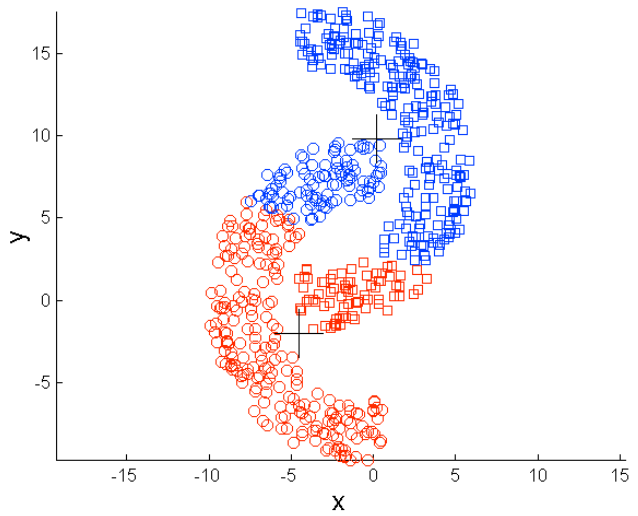
➢ *K*-means算法以误差平方和最小化为优化目标的局限性

- 非球形的簇

原始点



聚类（2个簇）



➢ *K*-means算法以均值作为聚类中心的局限性

- 对噪声和离群点敏感，计算时易受到离群点影响造成中心点偏移，产生聚类分布上的误差
  - 一维空间的7个点：1,2,3,8,9,10,25　令k=2
  - （1）合适的类别为{1,2,3}和{8,9,10,25}

    簇内误差平方和为：$(1-2)^2+(2-2)^2+(3-2)^2+(8-13)^2+(9-13)^2+(10-13)^2+(25-13)^2=196$
  - （2）k-means聚类结果为{1,2,3,8}和{9,10,25}

    簇内误差平方和为：$(1-3.5)^2+(2-3.5)^2+(3-3.5)^2+(8-3.5)^2+(9-14.67)^2+(10-14.67)^2+(25-14.67)^2=189.67$

➢ *K*-medoids算法

- *K*-medoids（*K*-中心点）算法选用簇的中心点代替均值点作为聚类中心，根据各数据对象与这些聚类中心之间的距离之和最小化的原则，进行簇的划分
- 优点：对离群点不敏感、可以处理离散型数据
- 算法流程

输入：簇的个数*K*和数据集D

输出：*K*个簇的集合

从D中任意选择*K*个对象作为初始簇中心；

1. repeat
2. 根据与中心点最近的原则，将剩余点分配到当前最佳的中心点代表的簇中重新计算每个簇的中心；
3. 在每一个簇中，计算每个样本点对应的准则函数，选取准则函数最小时对应的点作为新的中心点
4. until中心点不发生变化或达到设定的最大迭代次数

➢ PAM

- PAM（Partitioning Around Medoid，围绕中心点的划分），是聚类分析中基于划分的聚类算法，是最早提出的k-中心点算法之一
- 基本思想：选用簇中位置最中心的对象，试图对n个对象给出k个划分。初随机选择k个对象作为中心点，该算法反复地用非代表对象来代替代表对象，试图找出更好的中心点，以改进聚类的质量
- 代表对象：中心点；非代表对象：除中心点外的其他对象
- 交换的总代价：所有对象的代价之和
  - 总代价为负值，则表明交换后的类内聚合度更好，原来的代表对象将被替代；
  - 总代价为正值，则表明原来的代表对象更好，不应被取代

➢ PAM算法思想

- 在PAM算法中，可以把过程分为两个步骤：

（1）建立：

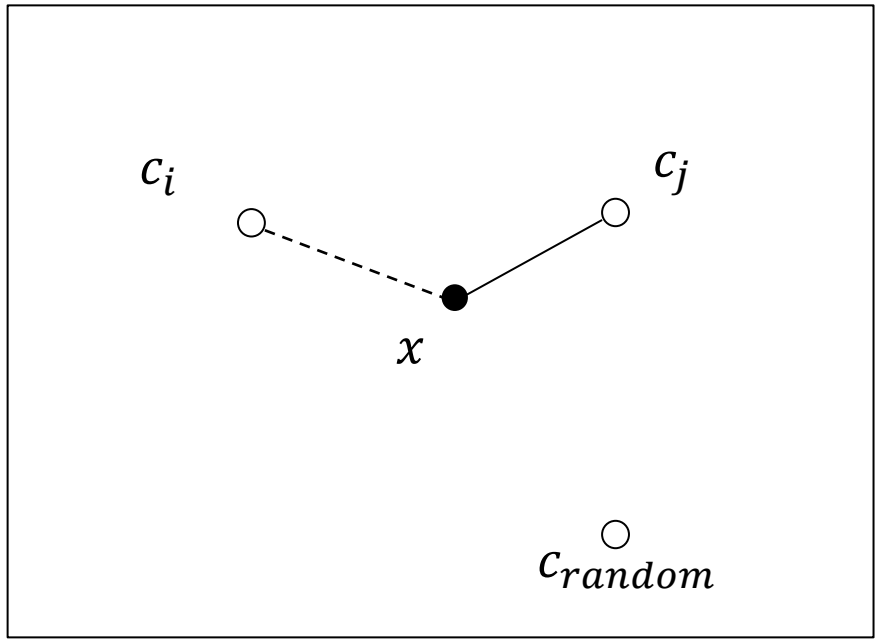
随机寻找 k个中心点作为初始的簇中心点

（2）交换：

对于所有可能的对象对进行分析——判定一个非代表对象c_{random}是否是当前一个代表对象 c_j的好的替代，找到交换后可以使误差平方值 E 减少的对象，代替原中心点

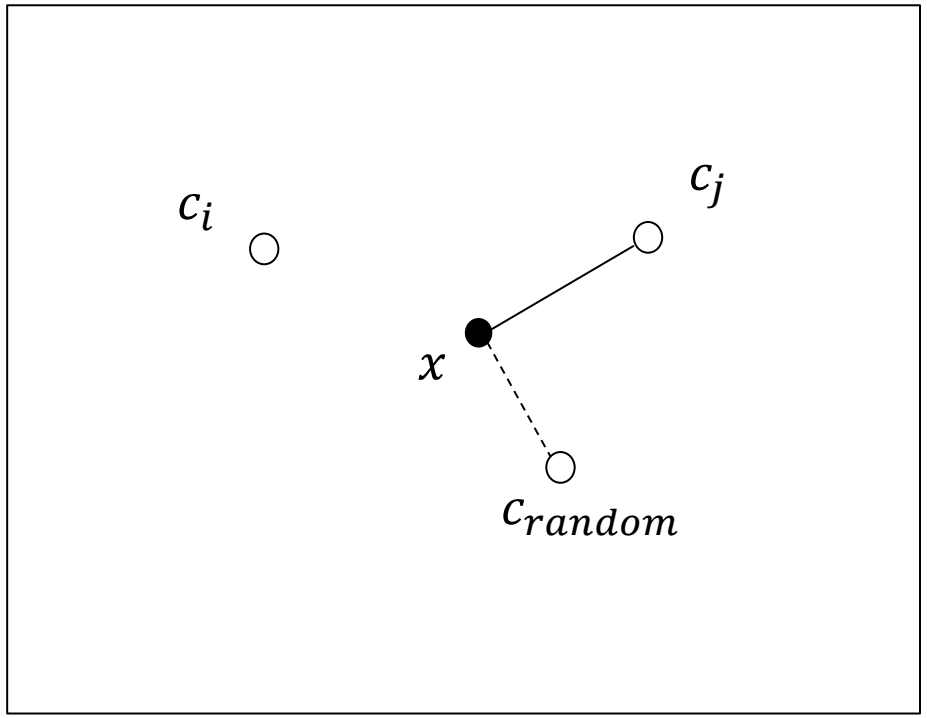
➢ 代价计算的四种情况

- 为了判定一个非代表对象c_{random}是否可以替换当前的某个代表对象c_j，根据x位置的不同，分为4种情况：



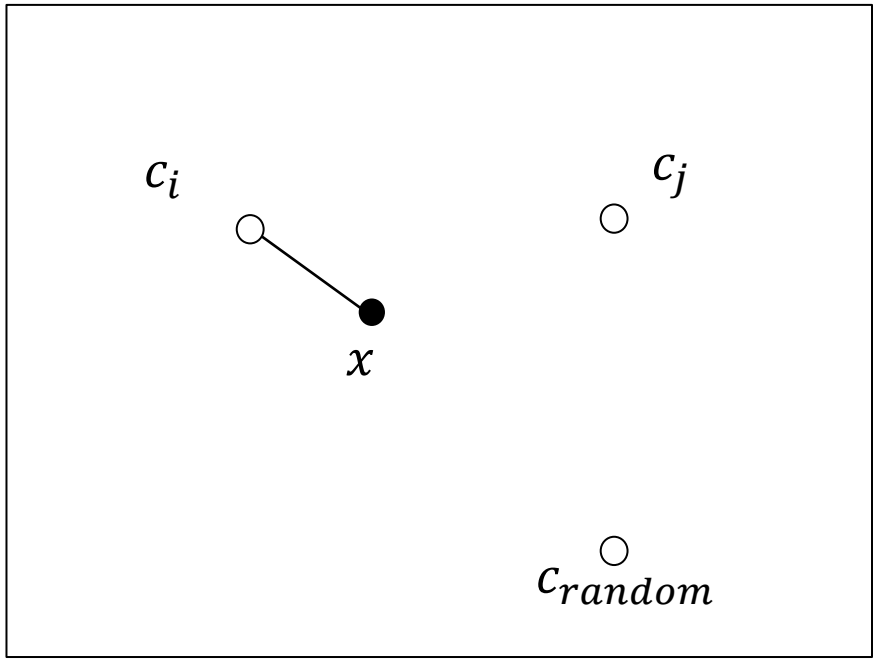
- 情况1：对象x当前属于以c_j为中心点的簇，如果用c_{random}替换c_j代表对象，x将更接近其他簇的中心点c_i，那么就将x归类到以c_i为中心点的簇中。

代价计算的四种情况



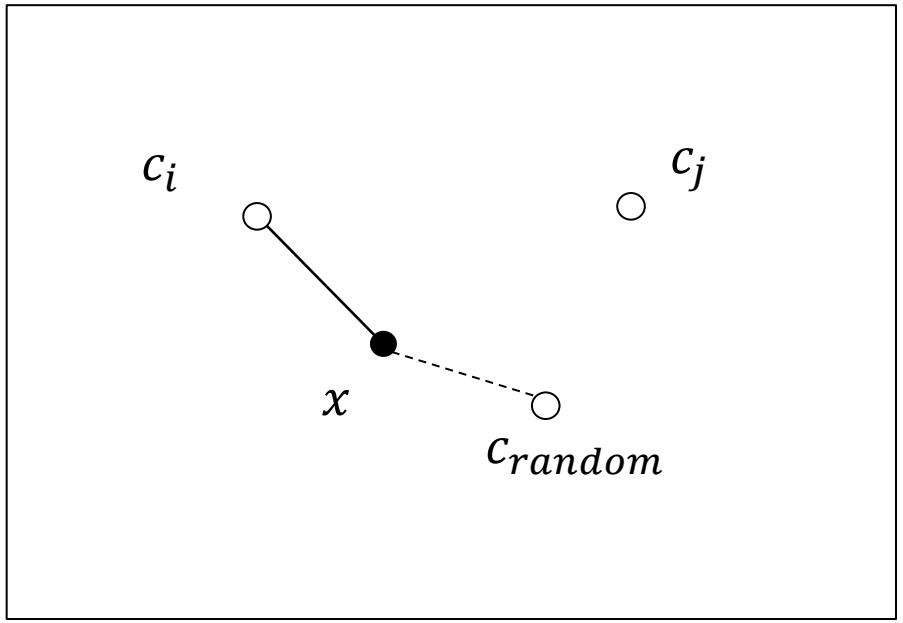
- 情况2：对象x当前属于以c_j为中心点的簇，如果用c_{random}替换c_j作为新的代表对象，x将更接近c_{random}，那么就将 x 归类到以 c_{random} 为中心点的簇中。

代价计算的四种情况



- 情况3：对象x当前属于以c_i为中心点的簇，如果用c_{random}替换c_j作为新的代表对象，x仍然最接近c_i，那么x的归类将保持不变。

代价计算的四种情况

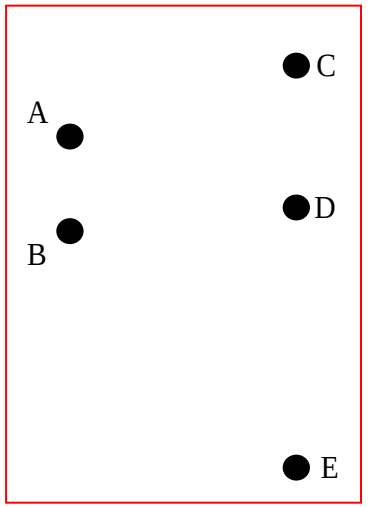


- 情况4：对象x当前属于以c_i为中心点的簇，如果用c_{random}替换c_j作为新的代表对象，x将更接近c_{random}，那么就将 x 归类到以 c_{random} 为中心点的簇中。

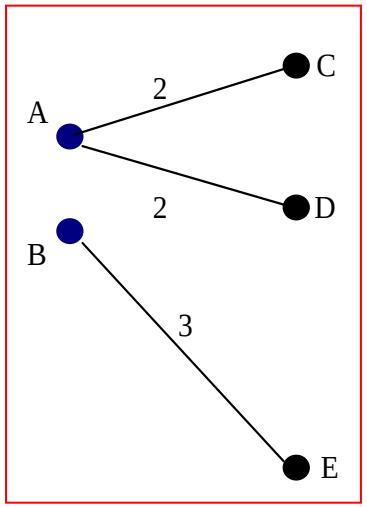
PAM算法实例

- 假如空间中的五个点｛A、B、C、D、E｝如图1所示，各点之间的距离关系如表所示，根据所给的数据对其运行PAM算法实现划分聚类（设k=2）。 样本点间距离如下表所示:

样本点	A	B	C	D	E
A	0	1	2	2	3
B	1	0	2	4	3
C	2	2	0	1	5
D	2	4	1	0	3
E	3	3	5	3	0



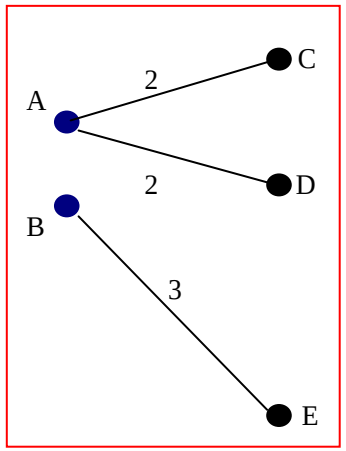
样本点



初始中心点

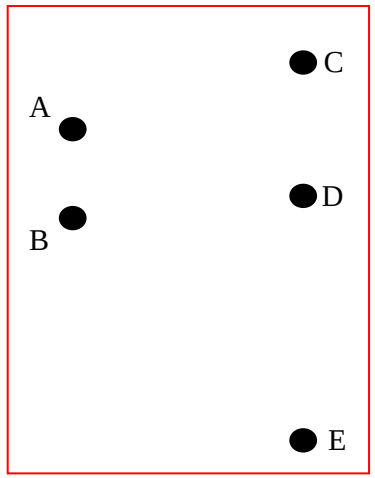
2 划分聚类方法

- 步骤1：建立阶段：假如从5个对象中随机抽取的2个中心点为{A，B},则样本被划分为{A、C、D}和{B、E}，如下图所示。



- 步骤2：交换阶段：假定中心点A、B分别被非中心点C、D、E替换，根据PAM算法需要计算下列代价：

TC_{AC}、TC_{AD}、TC_{AE}、TC_{BC}、TC_{BD}、TC_{BE}



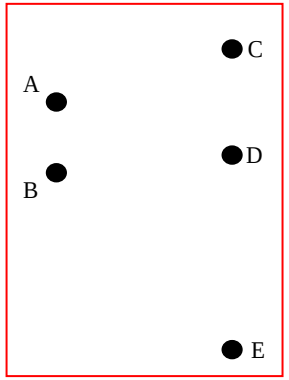
- 以 TC_{AC}为例说明计算过程：

a) 当A被C替换以后，A不再是一个中心点，因为A离B比A离C近，A被分配到B中心点代表的簇， $C_{AAC} = d(A, B) - d(A, A) = 1$

b) B是一个中心点，当A被C替换以后，B不受影响， $C_{BAC} = 0$

c) C原先属于A中心点所在的簇，当A被C替换以后，C是新中心点，符合PAM算法代价函数的第二种情况$C_{CAC} = d(C, C) - d(C, A) = 0 - 2 = -2$。

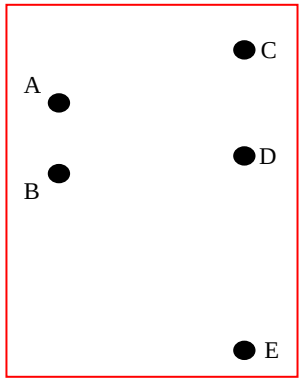
d) D原先属于A中心点所在的簇，当A被C替换以后，离D最近的中心点是C，$C_{DAC} = d(D, C) - d(D, A) = 1 - 2 = -1$。



- 以 TC_{AC}为例说明计算过程：

  e) E原先属于B中心点所在的簇，当A被C替换以后，离E最近的中心仍然是 B，根据PAM算法代价函数的第三种情况$C_{EAC}=0$。因此，$TC_{AC}=CA_{AC}+CB_{AC}+CB_{AC}+CD_{AC}+CE_{AC}=1+0-2-1+0=-2$。

  - 上述代价计算完毕后，选取一个最小的代价，如果有多种替换可以选择，则选择第一个最小代价的替换。



所有情况：

替换中心点A：

C替换A, TC_{AC}= -2

D替换A, TC_{AD}= -2

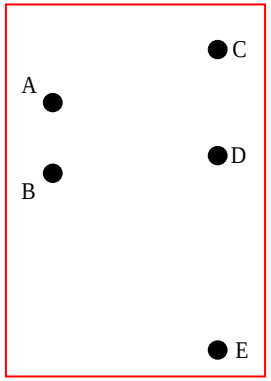
E替换A, TC_{AE}= -1

替换中心点B：

C替换B, TC_{BC}= -2

D替换B, TC_{BD}= -2

E替换B, TC_{BE}= -2



- 通过上述计算，已经完成了K-medoids算法的第一次迭代。在下一迭代中，将用其他的非中心点{A、D、E}替换中心点{B、C}，找出具有最小代价的替换。一直重复上述过程，直到代价不再减小为止

➢ *K*-means++算法

- *K*-means++（K均值++）算法是*K*-means的改进，其主要区别在于初始化确定初始聚类中心
- 区别：
  - *K*-means：随机确定初始聚类中心
  - *K*-means++：根据当前样本点到簇中心的距离计算该样本点成为下一个聚类中心的概率

$$P(x)=\frac{\sum_{c_i\in C} d(x,c_i)^2}{\sum_{c_i\in C}\sum_{i\in X} d(i,c_i)^2}$$

其中，x为当前样本点，c_i代表第i个簇中心，C代表已确定的簇中心集合，X为样本点总数

➢ *K*-means++算法

- 基本思想：距离越大则概率越大，然后根据概率大小抽取下一个聚类中心，不断重复直至抽取K个聚类中心。选出初始聚类中心后，继续使用标准的K-means算法进行聚类。

输入：簇的个数K和数据集D

输出：K个簇的集合

从D中任意选择K个对象作为初始簇中心c_0

1. repeat
2. 根据簇中均值，分配对象到最相似的簇中更新均值，重新计算每个簇的质心
3. 更新簇均值，即重新计算每个簇中均值
4. Until质心不发生变化或达到设定的最大迭代次数

➢ *K*-means++算法实例

点	属性1	属性2	点	属性1	属性2	点	属性1	属性2
1	0.697	0.460	11	0.245	0.057	21	0.748	0.232
2	0.774	0.376	12	0.343	0.099	22	0.714	0.346
3	0.634	0.264	13	0.639	0.161	23	0.483	0.312
4	0.608	0.318	14	0.657	0.198	24	0.478	0.437
5	0.556	0.215	15	0.360	0.370	25	0.525	0.369
6	0.403	0.237	16	0.800	0.042	26	0.751	0.489
7	0.481	0.149	17	0.719	0.103	27	0.532	0.472
8	0.437	0.211	18	0.359	0.188	28	0.473	0.376
9	0.666	0.091	19	0.339	0.241	29	0.725	0.445
10	0.243	0.267	20	0.282	0.257	30	0.446	0.459

- 步骤1：随机选择一个簇中心（点1）。计算出其他样本点成为下一个簇中心的概率

- 根据右表结果显示，概率最大点为点11，故选择点11为第二个簇中心点。同理，通过计算，得出第三个簇中心点为点16。

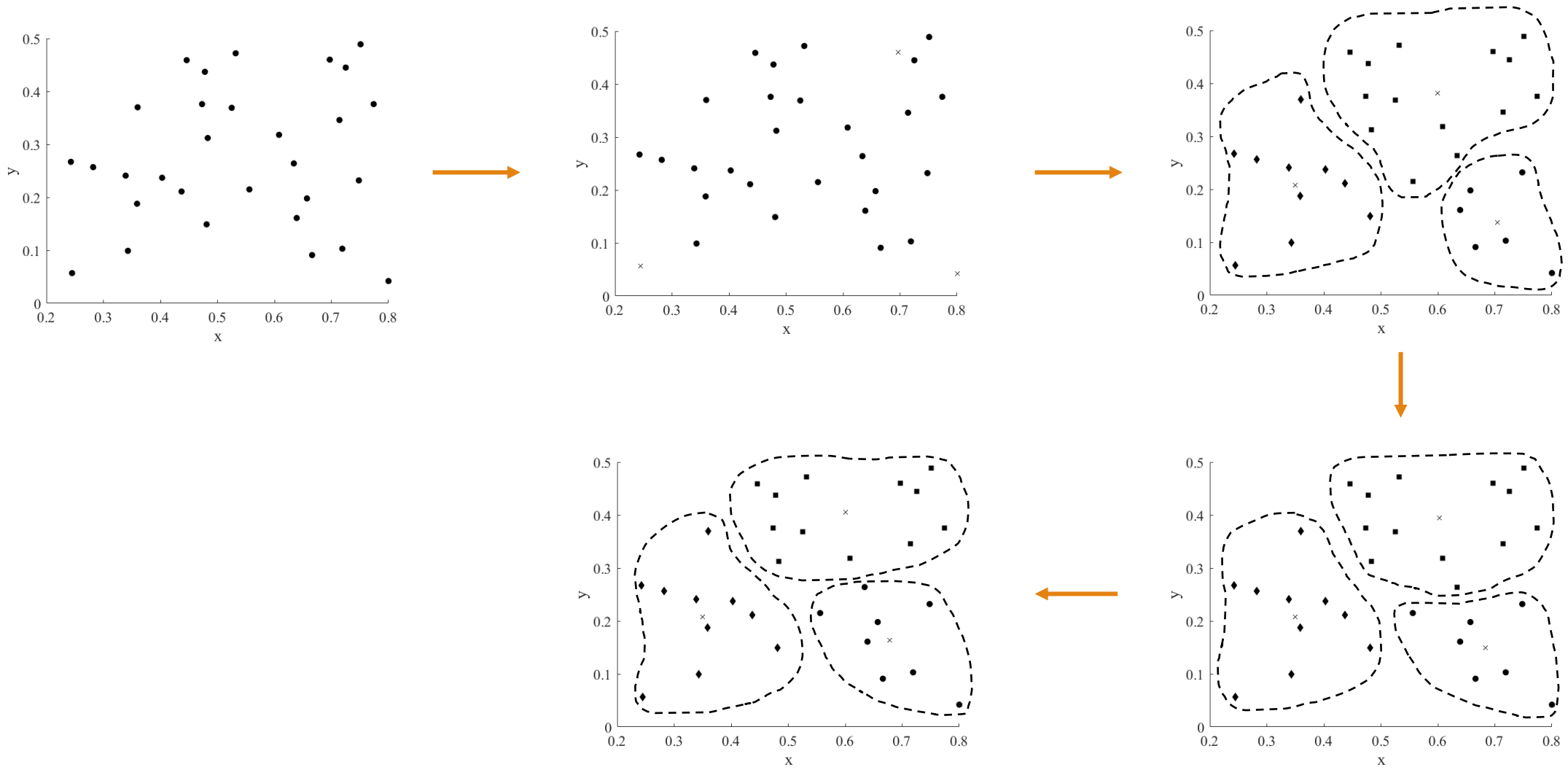
- 初始化簇中心分别为点1、点11、点16，簇中心用“×”表示。根据与簇中心的距离（假设采用欧式距离），其他对象被分配到最近的一个簇。

- 步骤2：更新簇中心。即根据簇中的当前对象，重新计算每个簇的均值。使用这些新的簇中心，把对象重新分布到里簇中心最近的簇中。

- 重复这一过程，直到对象的重新分配不再发生，处理结果结束，输出聚类结果

点	1	2	3	4	5	6	7	8	9	10
概率	0	0.013	0.024	0.019	0.033	0.043	0.044	0.042	0.043	0.057
点	11	12	13	14	15	16	17	18	19	20
概率	0.070	0.058	0.035	0.031	0.040	0.050	0.041	0.050	0.049	0.053
点	21	22	23	24	25	26	27	28	29	30
概率	0.027	0.013	0.030	0.025	0.023	0.007	0.019	0.028	0.004	0.029

➢ K-means++算法实例



目录

©ISTOCKPHOTO.COM/ZAPP2PHOTO
IEEE INDUSTRIAL ELECTRONICS MAGAZINE ■ DECEMBER 2019 PP.38

3 层次聚类方法

➢ 层次聚类方法

- 通过将数据组织为若干组并形成一个组的树来进行聚类

✓ 凝聚的层次聚类：一种自底向上的策略，首先将每个对象作为一个簇，然后通过逐步合并相近的簇从而形成越来越大的簇，直至所有的对象都在一个簇里，或者满足一定的终止条件为止。代表算法是AGNES算法。

✓ 分裂的层次聚类：采用自顶向下的策略，初始时是将所有的对象置于一个簇中，然后逐步将其细分为较小的簇，直到每个对象自成一个簇，或者满足一定的终止条件为止。代表算法是DIANA算法。

3 层次聚类方法

➢ 层次聚类方法

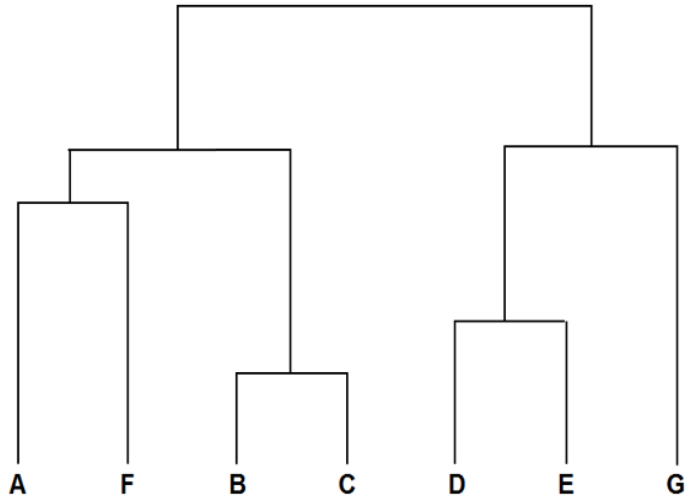
◘ 层次聚类步骤总结：

（1）将每个对象归为一类，共得到N类，每类仅包含一个对象。类与类之间的距离就是它们包含的对象之间的距离

（2）找到最接近的两个类合并成一类，于是总的类数减少一个

（3）重新计算新类与所有旧类之间的距离

（4）重复（2）、（3），直到最后合并成一个类为止（此类包含了N个对象）



- 最顶部的根节点：整个数据集
- 中间节点：若干个对象构成的子簇
- 最底部的叶子节点：数据集中的对象

3 层次聚类方法

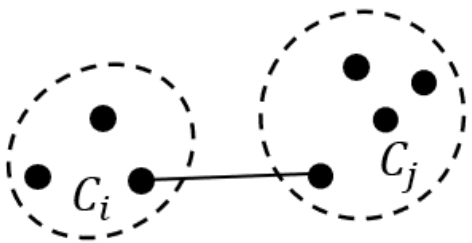
➢ 算法的距离度量方法

- 度量簇之间的距离常用的方法有：最小距离、最大距离、平均距离、质心距离等

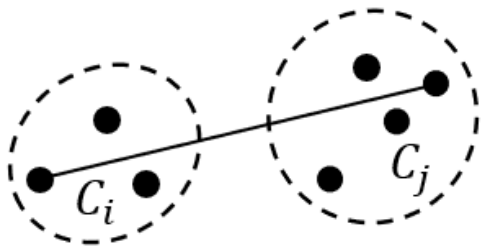
✓ 最小距离：
$$d_{\min}=(C_i,C_j)=\min_{x\in C_i,z\in C_j} dist(x,z)$$

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

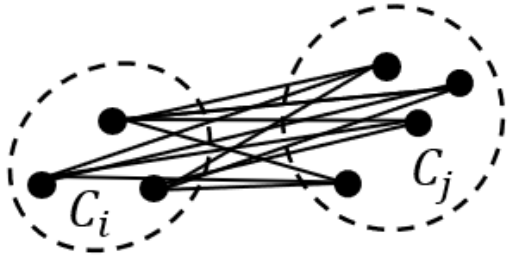
✓ 平均距离：
$$d_{avg}(C_i,C_j)=\frac{1}{|C_i||C_j|}\sum_{C_i}\sum_{C_j} dist(x,z)$$



最小距离



最大距离



平均距离

3 层次聚类方法

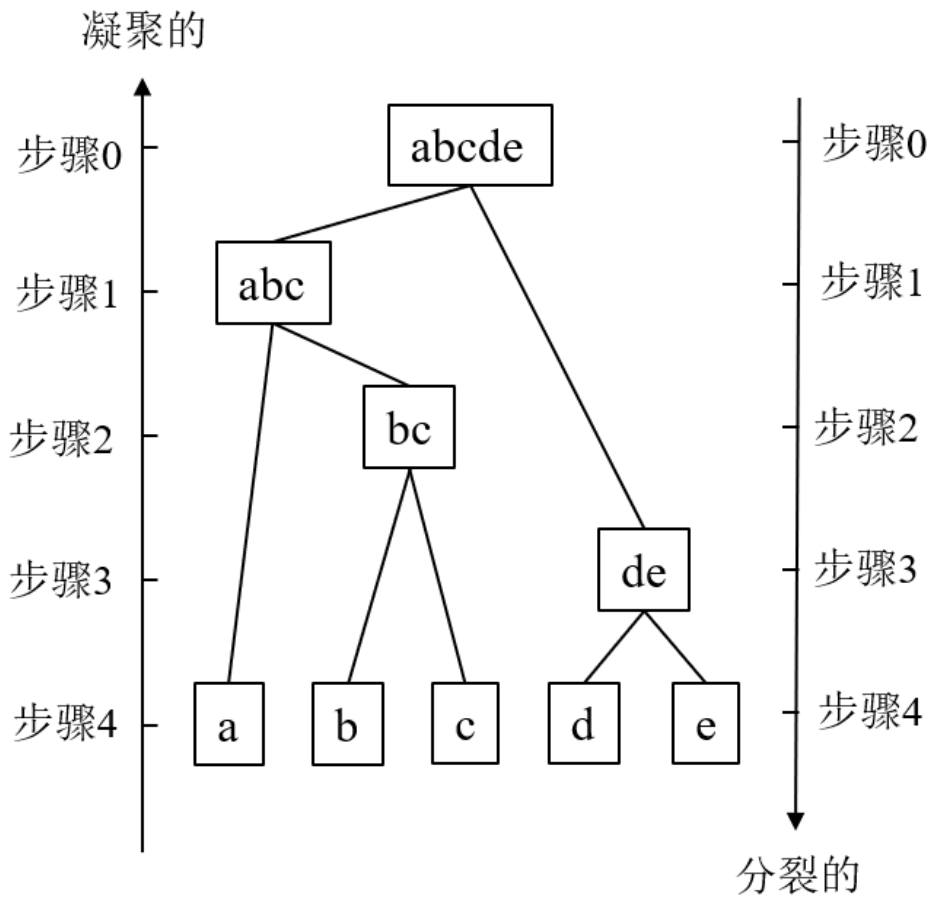
➢ 凝聚的与分裂的层次聚类

- 对给定的数据集进行层次的分解，直到满足某种条件或者达到最大迭代次数

(AGNES)

凝聚的层次聚类方法

使用自底向上的策略



(DIANA)

分裂的层次聚类方法

使用自顶向下的策略

3 层次聚类方法

AGNES算法

- AGNES (AGglomerative NESting)算法是凝聚的层次聚类的典型代表
- 处理步骤：

（1）将数据集中的每个对象作为一个簇

（2）每次找到距离最近的两个簇进行合并

（3）合并过程反复进行，直至不能再合并或者达到结束条件为止

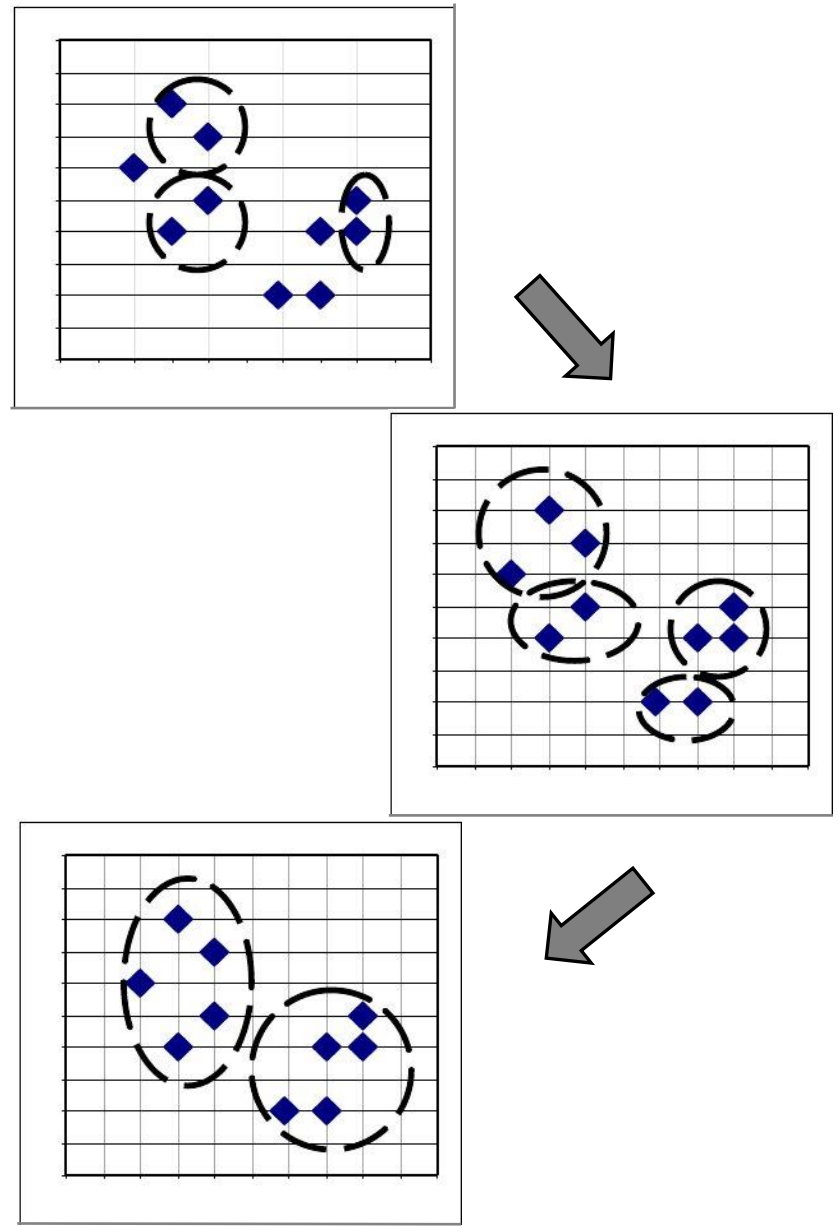
3 层次聚类方法

➢ AGNES算法

输入：数据集$X = (x_1, x_2, \dots, x_m)^T$

输出：k个簇，达到终止条件规定簇数目

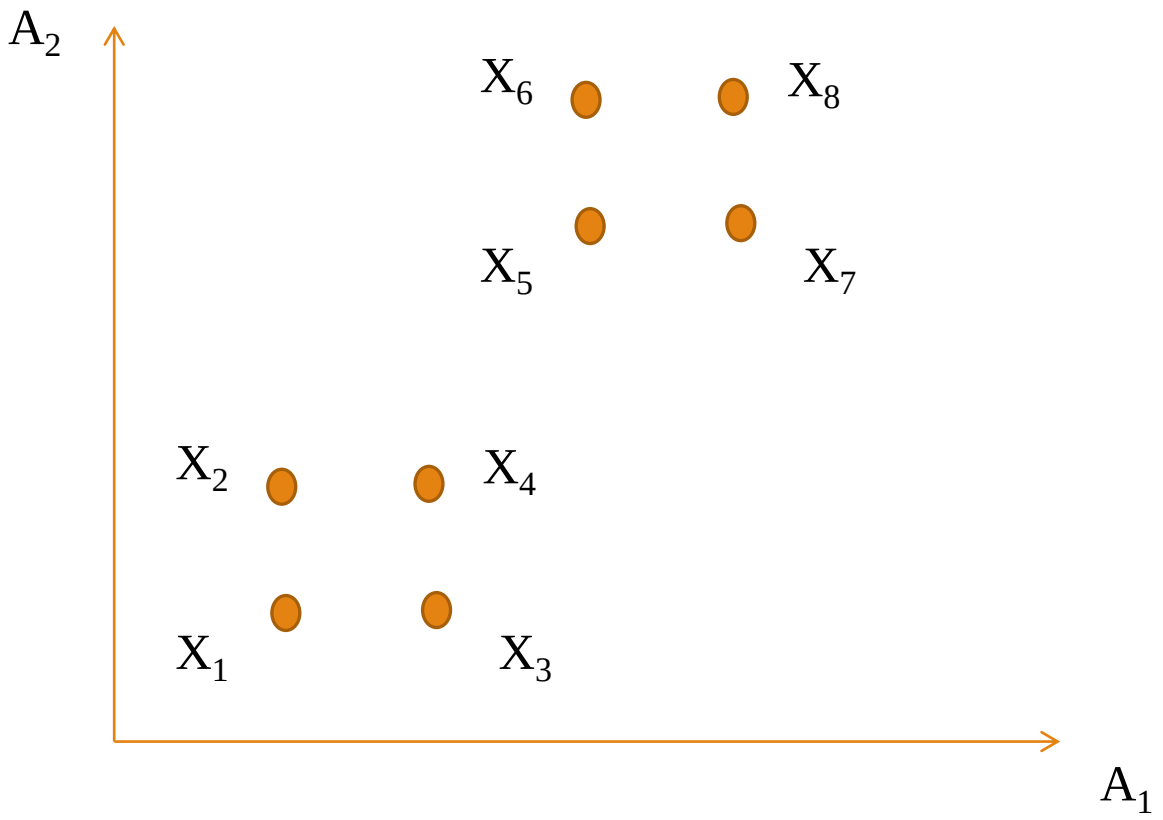
1. 初始化：将每个数据对象当成一个初始簇，分别计算每对簇之间的距离，结果存入距离矩阵$D = (d_{ij})$
2. repeat
3. 寻找距离最近的一对簇C_i和C_j
4. 构造一个新簇$C_k = C_i \cup C_j$，这一过程相当于在树状图中新增一个节点，并将该节点分别连接簇C_i和簇C_j的节点
5. 更新距离矩阵，即计算簇C_k与其他簇（除了C_i和C_j）之间的距离
6. 从矩阵D中删除对应C_i和C_j的行和列，增加对应C_k的行和列
7. until所有数据对象都在同一个簇中



算法实例

- S为有8个数据对象的数据集S，簇数k=2。试用AGNES算法聚类
- 本题采用簇间的中心距离为目标函数

id	A_1	A_2
X_1	1	1
X_2	1	2
X_3	2	1
X_4	2	2
X_5	3	4
X_6	3	5
X_7	4	4
X_8	4	5



算法实例

- 初始步，每个对象为一个簇：

$\{X_1\},\{X_2\},\{X_3\},\{X_4\},\{X_5\},\{X_6\},\{X_7\},\{X_8\}$

第1 步、为方便使用簇间中心距离平方。此例开始时有8个簇，共需计算28个簇间距离平方。但仅有

$$d(X_1,X_2)^2 = d(X_1,X_3)^2 = d(X_2,X_4)^2 = d(X_3,X_4)^2 = 1$$

$$d(X_5,X_6)^2 = d(X_5,X_7)^2 = d(X_6,X_8)^2 = d(X_7,X_8)^2 = 1$$

而其它对象之间的中心距离平方都大于1。按照数据对象坐标编号小者顺序优先合并，即选择$\{X_1\},\{X_2\}$合并为$\{X_1,X_2\}$。即得 7个簇$\{X_1,X_2\}$，$\{X_3\},\{X_4\},\{X_5\},\{X_6\},\{X_7\},\{X_8\}$

算法实例

- 第2步、重新计算簇$\{X_1,X_2\}$的中心点为$C^{(1)}=(1, 1.5)$，并增加计算$C^{(1)}$与$\{X_3\},\{X_4\},\{X_5\},\{X_6\},\{X_7\},\{X_8\}$的距离平方。

  $d(C^{(1)}, X_3)^2=1.25$，$d(C^{(1)}, X_4)^2=1.25$

- 其它$d(C^{(1)}, X_5)^2$，…，$d(C^{(1)}, X_8)^2$等都大于1.25。因

  $d(X_3, X_4)^2=1$，故簇$\{X_3\}$和$\{X_4\}$合并为$\{X_3, X_4\}$，见下表计算步骤2所在的行

3 层次聚类方法

➢ 算法实例

计算步骤	最近的两个簇	合并后的新簇	簇个数k
0	-	$\{X_1\},\{X_2\},\{X_3\},\{X_4\},\{X_5\},\{X_6\},\{X_7\},\{X_8\}$	8
1	$\{X_1\},\{X_2\}$	$\{X_1,X_2\},\{X_3\},\{X_4\},\{X_5\},\{X_6\},\{X_7\},\{X_8\}$	7
2	$\{X_3\},\{X_4\}$	$\{X_1,X_2\},\{X_3,X_4\},\{X_5\},\{X_6\},\{X_7\},\{X_8\}$	6
3	$\{X_1,X_2\}, \{X_3,X_4\}$	$\{X_1,X_2,X_3,X_4\},\{X_5\},\{X_6\},\{X_7\},\{X_8\}$	5
4	$\{X_5\},\{X_6\}$	$\{X_1,X_2,X_3,X_4\},\{X_5,X_6\},\{X_7\},\{X_8\}$	4
5	$\{X_7\},\{X_8\}$	$\{X_1,X_2,X_3,X_4\},\{X_5,X_6\},\{X_7,X_8\}$	3
6	$\{X_5,X_6\},\{X_7,X_8\}$	$\{X_1,X_2,X_3,X_4\},\{X_5,X_6,X_7,X_8\}$	2

AGNES算法实例

设n = 8，用户输入的终止条件为两个簇。初始簇为{1}，{2}，{3}，{4}，{5}，{6}，{7}，{8}。（采用最小距离计算）

序号	属性1	属性2	序号	属性1	属性2
1	4.05	0.25	5	5.65	0.24
2	4.56	0.21	6	5.12	0.37
3	4.87	0.16	7	4.97	0.48
4	4.89	0.31	8	5.94	0.34

属性1：风速值
属性2：风向余弦值

AGNES算法实例

- 步骤1：先根据最小距离计算公式，将两两样本点的距离计算出来（距离结果保留小数点后两位）。找到距离最小的两个簇合并，故而将3、4合并。

两两样本点之间的距离

样本点	1	2	3	4	5	6	7	8
1	0	0.512	0.825	0.842	1.600	1.077	0.948	1.892
2	0.512	0	0.314	0.345	1.090	0.582	0.491	1.386
3	0.825	0.314	0	0.151	0.784	0.326	0.335	1.085
4	0.842	0.345	0.151	0	0.763	0.238	0.188	1.050
5	1.600	1.090	0.784	0.763	0	0.546	0.721	0.307
6	1.077	0.582	0.326	0.238	0.546	0	0.186	0.821
7	0.948	0.491	0.335	0.188	0.721	0.186	0	0.980
8	1.892	1.386	1.085	1.050	0.307	0.821	0.980	0

AGNES算法实例

- 步骤2：对上一次合并后的簇进行簇间计算，找出距离最近的两个簇进行合并，合并后6、7合并成为一簇
- 步骤3：重复步骤2的工作，5、8成为一簇
- 步骤4：重复步骤2的工作，1、2成为一簇
- 步骤5：合并{3,4}，{6,7}成为一簇
- 步骤6：合并{3,4,6,7}，{1,2}成为一簇，合并后的簇的数目达到终止条件，计算完毕

步骤	最近的簇距离	最近的两个簇	合并后的新簇
1	0.151	{3},{4}	{1},{2},{3,4},{5},{6},{7},{8}
2	0.186	{6},{7}	{1},{2},{3,4},{5},{6,7},{8}
3	0.307	{5},{8}	{1},{2},{3,4},{5,8},{6,7}
4	0.512	{1},{2}	{1,2},{3,4},{5,8},{6,7}
5	0.188	{3,4},{6,7}	{1,2},{3,4,6,7} {5,8}
6	0.491	{3,4,6,7},{1,2}	{1,2,3,4,6,7},{5,8}结束

DIANA算法

- DIANA (Divisive Analysis)算法是分裂的层次聚类的典型代表
- 处理步骤：

（1）将数据集中的所有集合作为一个簇

（2）根据某种准则，每次将当前最大的簇分裂成两个子簇

（3）分裂过程反复进行，直至每个簇只包含一个对象或者达到结束条件为止

3 层次聚类方法

➢ DIANA算法

- 簇的直径：簇中任意两个数据对象之间距离的最大值
- 平均相异度$d_{avg}(x, C)$：x与C中所有对象（不包括x）之间距离的平均值

$$d_{avg}(x,C)=\begin{cases}\dfrac{1}{|C|-1}\sum\limits_{y\in C, y\neq x} d(x,y), x\in C\\ \dfrac{1}{|C|}\sum\limits_{y\in C} d(x,y), x\notin C\end{cases}$$

|C|表示簇C中对象的个数，d(x, y)是对象x与对象y之间的距离

  - $d_{avg}(x, C)$越大，表明x与C中其他对象的差异性越大，进行簇的分裂时，应当优先选择将x分离出去

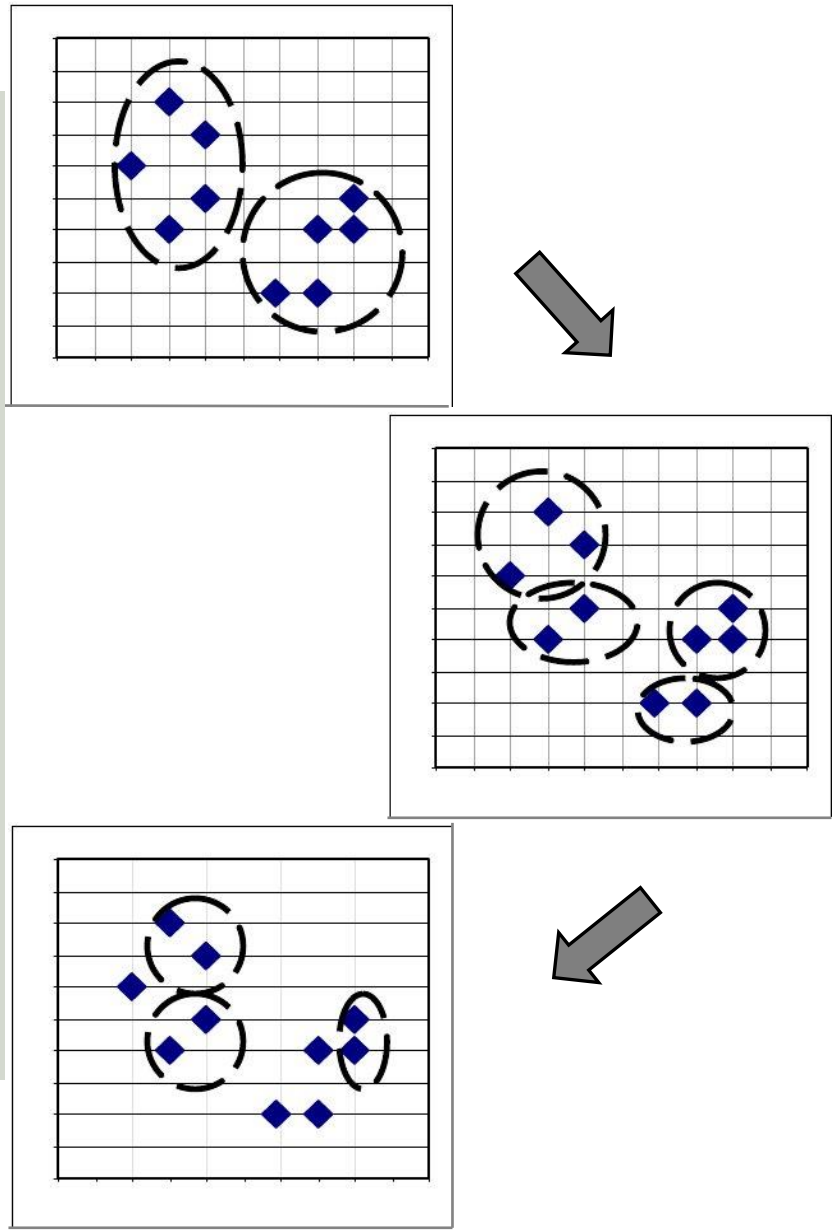
3 层次聚类方法

➢ DIANA算法流程

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放入分裂簇，剩余的放在原始簇中
5. repeat
6. 在原始簇里找出一个点q，使得q到分裂簇的最短距离不大于q到原始簇中其余各点距离的最小值，将该点加入分裂簇
7. UNTIL 没有新的原始簇的点被分配给分裂簇
8. 分裂簇和原始簇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

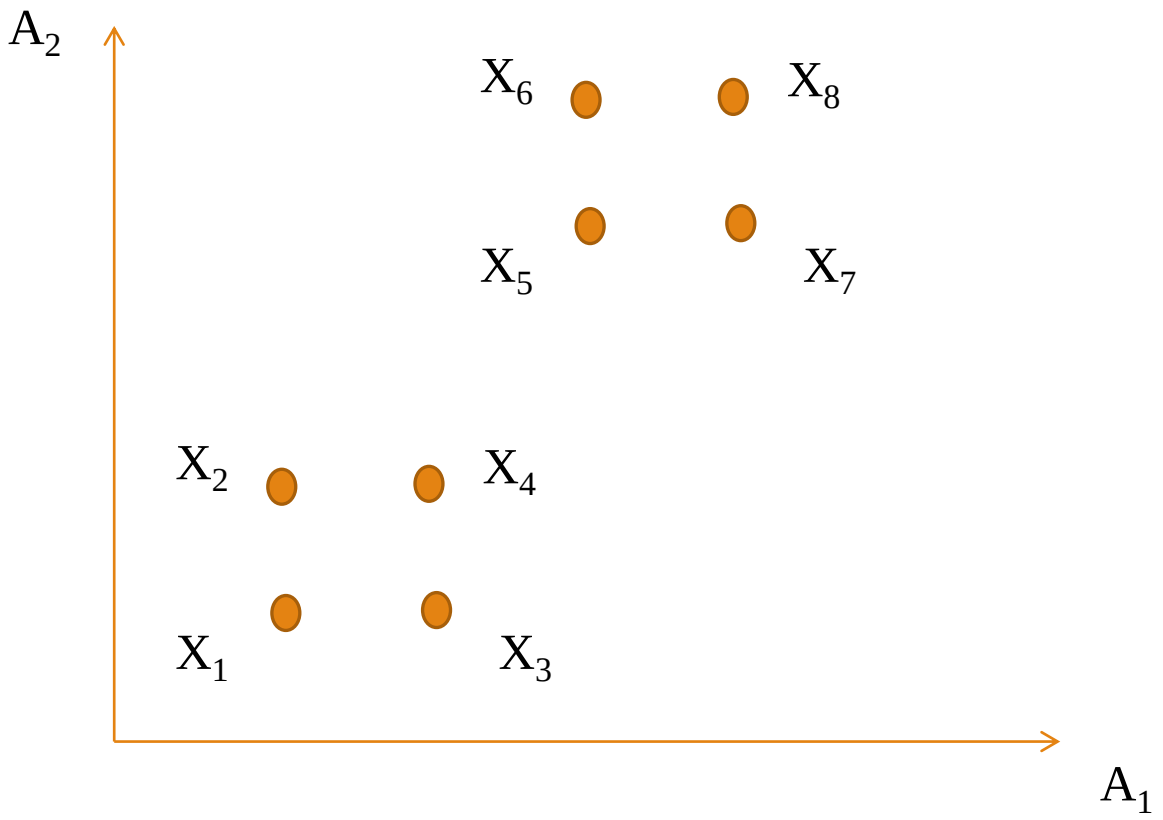


3 层次聚类方法

➢ 算法实例

- S为有8个数据对象的数据集S，簇数k=2。试用DIANA算法聚类

id	A_1	A_2
X_1	1	1
X_2	1	2
X_3	2	1
X_4	2	2
X_5	3	4
X_6	3	5
X_7	4	4
X_8	4	5



算法实例

- 解：因为S有8个数据对象，因此，刚开始所有对象作

为一个簇$C_o=\{X_1,X_2,X_3,X_4,X_5,X_6,X_7,X_8\}$

为将C_o中分裂成两个簇，要计算每个对象X_i到其它对象X_j $(j=1, 2, \ldots, 8; i\neq j\}$的平均相异度

算法实例

第一轮循环FOR h=1时

（1）$d_a(\{X_1\},\{\cdot\})=(1+1+1.41+3.6+4.47+4.24+5)/7=2.96$

(2) $d_a(\{X_2\},\{\cdot\})=(1+1.41+1+2.83+3.6+3.6+4.24)/7=2.53$

(3) $d_a(\{X_3\},\{\cdot\})=2.68$

(4) $d_a(\{X_4\},\{\cdot\})=2.18$

(5) $d_a(\{X_5\},\{\cdot\})=2.18$

(6) $d_a(\{X_6\},\{\cdot\})=2.68$

(7) $d_a(\{X_7\},\{\cdot\})=2.53$

(8) $d_a(\{X_8\},\{\cdot\})=2.96$

算法实例

- 第一轮循环FOR h=1时
- 由于$d_a(\{X_1\},\{\cdot\})= d_a(\{X_8\},\{\cdot\})=2.96$取得最大平均距离，按照对象下标编号小者优先的原则，将X_1从C_o中分裂出去，并分配给C_s，即得$C_s=\{X_1\}$,

  $C_o=\{X_2,X_3,X_4,X_5,X_6,X_7,X_8\}$

3 层次聚类方法

➢ 算法实例

- 内 循环 REPEAT
- (1) 从$C_o=\{X_2,X_3,X_4,X_5,X_6,X_7,X_8\}$中取一点$X_2$，计算它到$C_s$最小距离，经计算可得$d(X_2,C_s)=1$;
- (2) 计算点X_2到C_o中其它点的最小距离，即有$d(X_2,X_4)=1$;
- (3) 因$d(X_2,C_s)\leq d(X_2,X_4)$，所以将$X_2$从$C_o$中分裂出去，并分配给$C_s$，即得$C_o=\{X_3,X_4,X_5,X_6,X_7,X_8\}$，$C_s=\{X_1,X_2\}$。
- 以此类推，把X_3，X_4从C_o中分裂出去，并分配给C_s，即得
- $C_o=\{X_5,X_6,X_7,X_8\}$， $C_s=\{X_1,X_2,X_3,X_4\}$

3 层次聚类方法

➢ 算法实例

- 按照上面方法，分别计算X_5,X_6,X_7,X_8到C_s的最小距离$d_s(X_i,C_s)$ (i=5,6,7,8)，但有

  $d_s(X_i,C_s)> d(X_i,X_j)$ (i=5,6,7,8;i≠j)，即C_o中没有点需要分裂出去分配给C_s；

- 至此，得到聚类$C=\{\{X_1,X_2,X_3,X_4\},\{X_5,X_6,X_7,X_8\}\}=\{C_1,C_2\}$

算法实例

- 但是，如果指定k=4，则算法需要进入第二轮和第三轮循环：

第二轮循环 FOR h=2，得聚类

$C=\{\{X_1,X_2,X_3\},\{X_4\},\{X_5,X_6,X_7,X_8\}\}=\{C_1,C_2,C_3\}$

第三轮循环 FOR h=3，得聚类

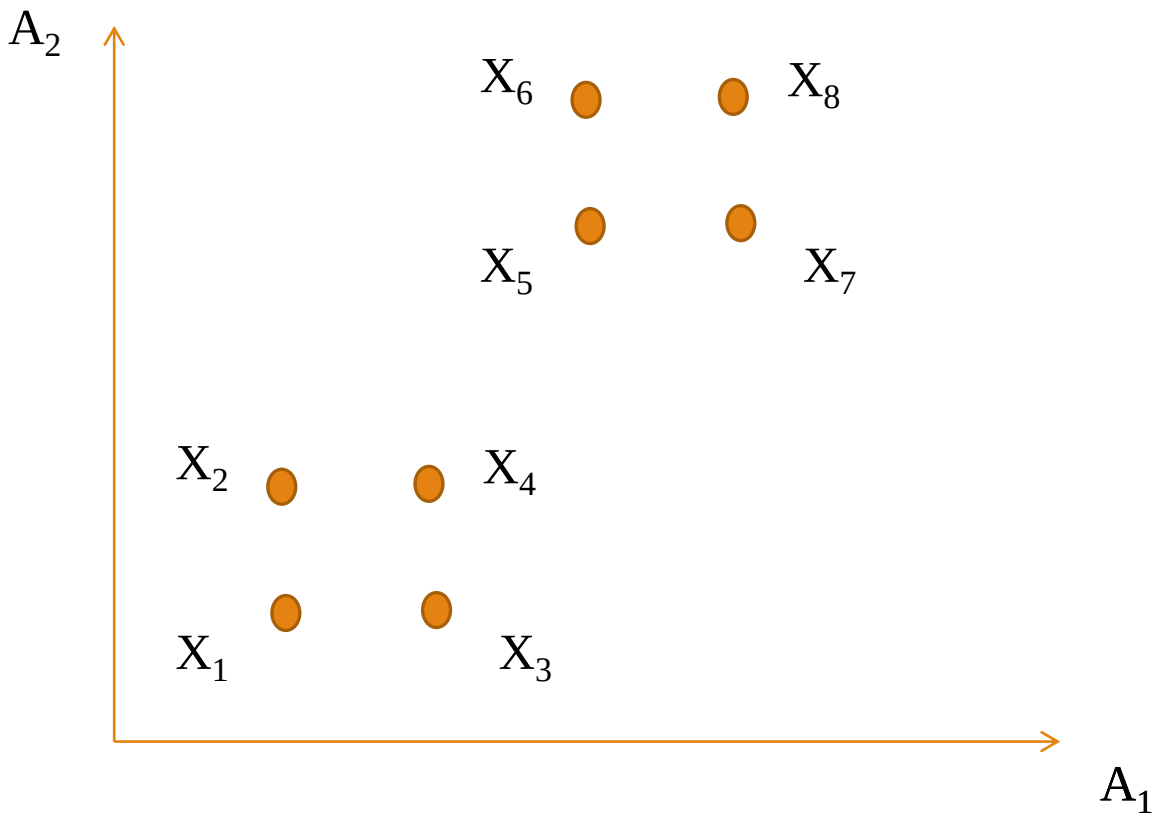
$C=\{\{X_1,X_2\},\{X_3\},\{X_4\},\{X_5,X_6,X_7,X_8\}\}=\{C_1,C_2,C_3,C_4\}$。

- 即DIANA算法得到了包括4个簇的聚类，算法结束

3 层次聚类方法

➢ 算法实例

- $C=\{\{X_1,X_2\},\{X_3\},\{X_4\},\{X_5,X_6,X_7,X_8\}\}=\{C_1,C_2,C_3,C_4\}$
- k=4，从图中看显得不太靠谱，DIANA算法对*k*值选取敏感



DIANA算法实例

设n = 8，用户输入的终止条件为两个簇。初始簇为{1}，{2}，{3}，{4}，{5}，{6}，{7}，{8}。（采用最小距离计算）

序号	属性1	属性2	序号	属性1	属性2
1	4.05	0.25	5	5.65	0.24
2	4.56	0.21	6	5.12	0.37
3	4.87	0.16	7	4.97	0.48
4	4.89	0.31	8	5.94	0.34

属性1：风速值
属性2：风向余弦值

➢ DIANA算法实例

- 步骤1：找到具有最大直径的簇，对簇中的每个点计算平均相异度。可求出各点的平均相异度。挑出平均相异度最大的点放到分裂的簇中，即点1，剩余点在原始簇

各样本点平均相异度

序号	1	2	3	4	5	6	7	8
平均相异度	1.099	0.674	0.546	0.511	0.830	0.539	0.550	1.074

- 步骤2：在原始簇里找出到分裂的簇中的最近的点的距离不大于到原始簇中最近的点的距离的点，将该点放入分裂的簇中。根据表5.7可知，该点是2。
- 步骤3：重复步骤2的工作，在分裂的簇中放入点3
- 步骤4：重复步骤2的工作，在分裂的簇中放入点4

➢ DIANA算法实例

- 步骤5：没有新的原始簇中的点分配给分裂的簇，此时分裂的簇数为2，达到终止条件。如果没有到终止条件，下一阶段还会从分裂好的簇中选一个直径最大的簇按照上边所说的分裂方法继续分裂

步骤	具有最大直径的簇	分裂的簇	原始簇
1	{1,2,3,4,5,6,7,8}	{1}	{2,3,4,5,6,7,8}
2	{1,2,3,4,5,6,7,8}	{1,2}	{3,4,5,6,7,8}
3	{1,2,3,4,5,6,7,8}	{1,2,3}	{4,5,6,7,8}
4	{1,2,3,4,5,6,7,8}	{1,2,3,4}	{5,6,7,8}
5	{1,2,3,4,5,6,7,8}	{1,2,3,4}	{5,6,7,8}终止

3 层次聚类方法

➢ 凝聚的与分裂的层次聚类

AGNES算法特点：

（1）简单，但遇到合并点选择困难的情况；

（2）一旦一组对象被合并，不能撤销

（3）算法的复杂度为$O(n^2)$，不适合大数据集。

DIANA算法特点：

（1）缺点是已做的分裂操作不能撤销，类之间不能交换对象；

（2）如果在某步没有选择好分裂点，可能会导致低质量的聚类结果；

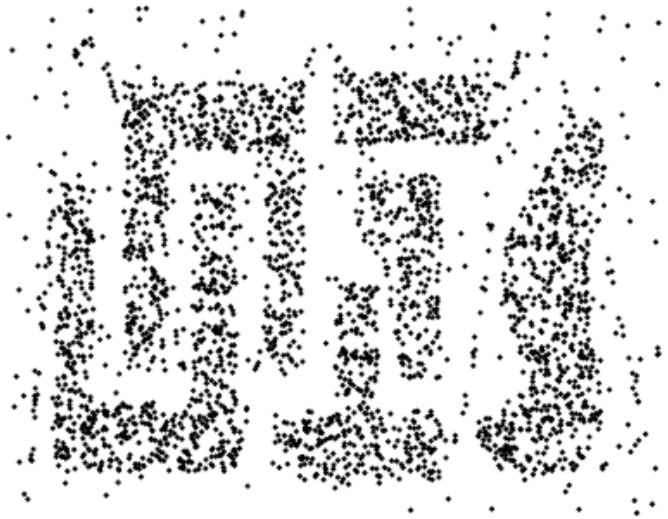
（3）算法的复杂度为$O(n^2)$，不适合大数据集。

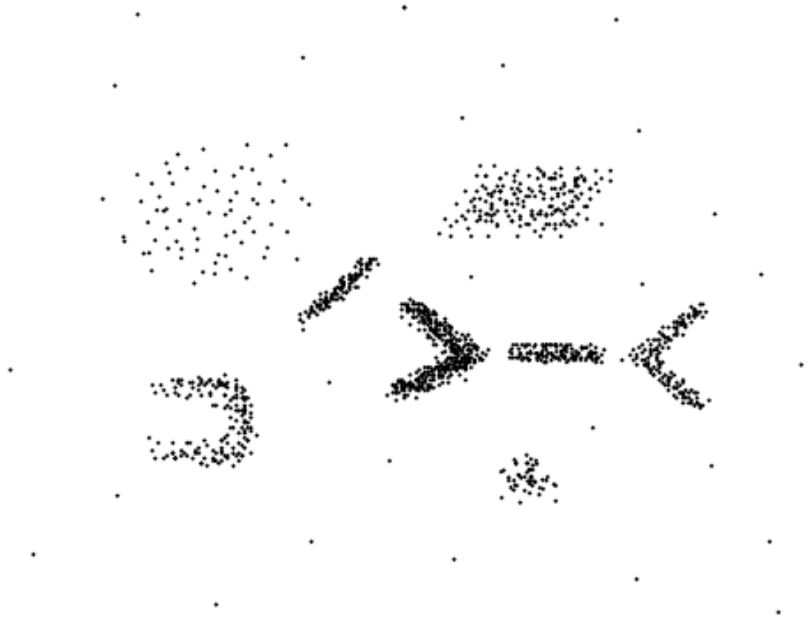
- ✓ 层次聚类方法尽管简单，但经常会遇到合并或分裂点的选择的困难
- ✓ 改进层次方法的聚类质量的一个有希望的方向是将层次聚类和其他聚类技术进行集成，形成多阶段聚类

目录

➢ 基于密度的聚类方法

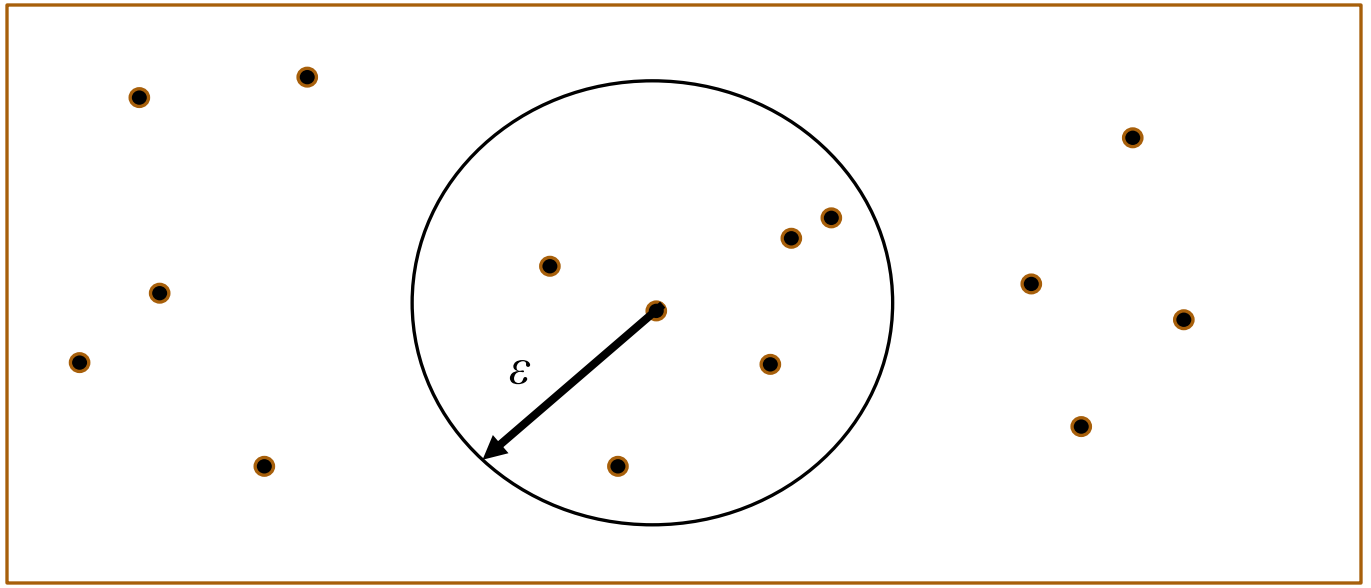
- 与其他方法的根本区别：不是基于距离的，而是基于密度的
- 克服基于距离的算法只能发现球状聚类，对发现任意形状的聚类则显得不足的缺点
- 基于密度的聚类算法的代表：DBSCAN、OPTICS等





基本概念

- 定义1：对象的 ε- 邻域：给定对象在半径 ε 内的区域

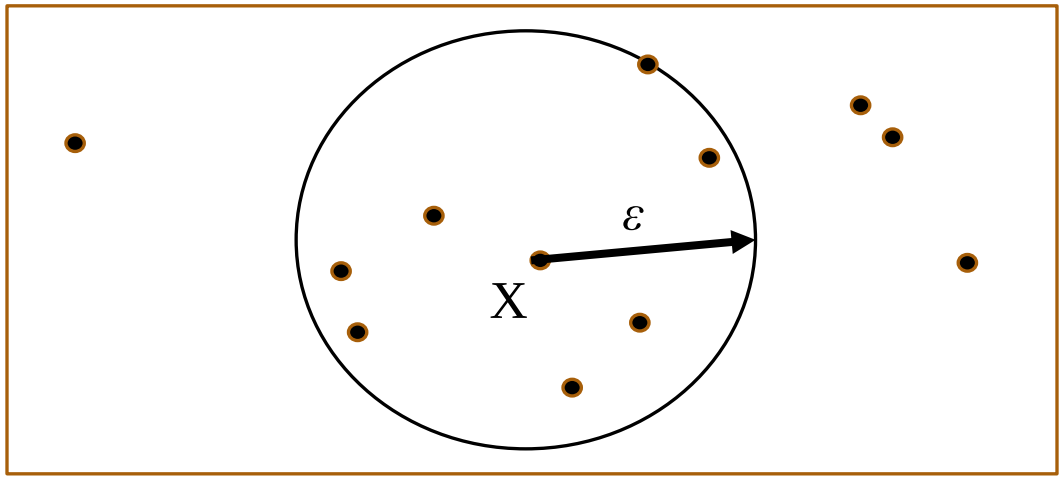


- 定义2：对于给定的实数 $\varepsilon>0$ 和正整数MinPts，称(ε, MinPts)是为数据集S指定的一个密度，简称(ε, MinPts)为密度。

4 基于密度的聚类方法

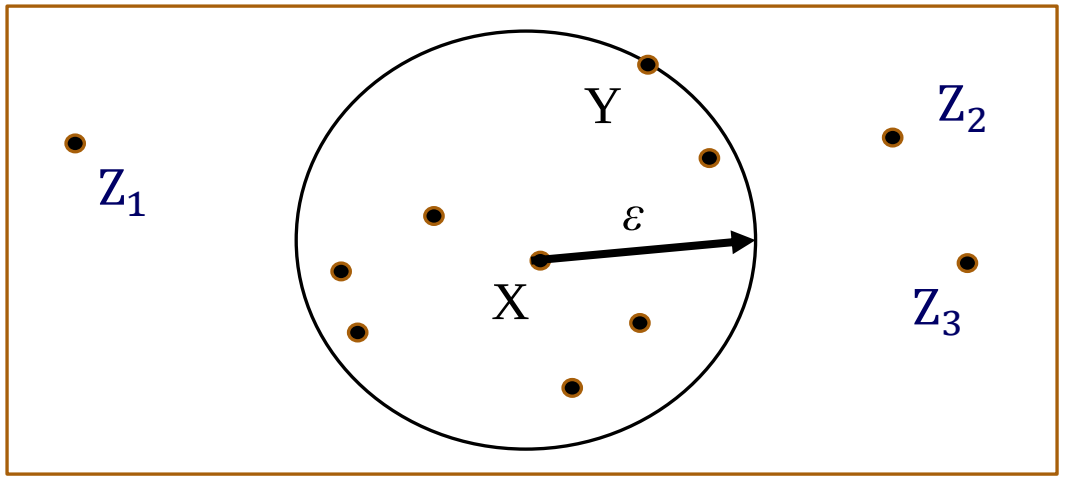
➢ 基本概念

- 在DBSCAN算法中，密度(ε, MinPts)专门用于刻画一个簇中对象的密集程度，为此引入核心对象的概念
- 定义3：如果一个对象 ε-邻域至少包含最小数目MinPts个对象，则称该对象为核心对象
  - 例：假设给定 ε =2.5，MinPts=8，则图中的点 X 就是 S 的一个核心对象。但如果指定MinPts=10，即使 ε 取值不变，X也不再是S的核心对象了



➢ 基本概念

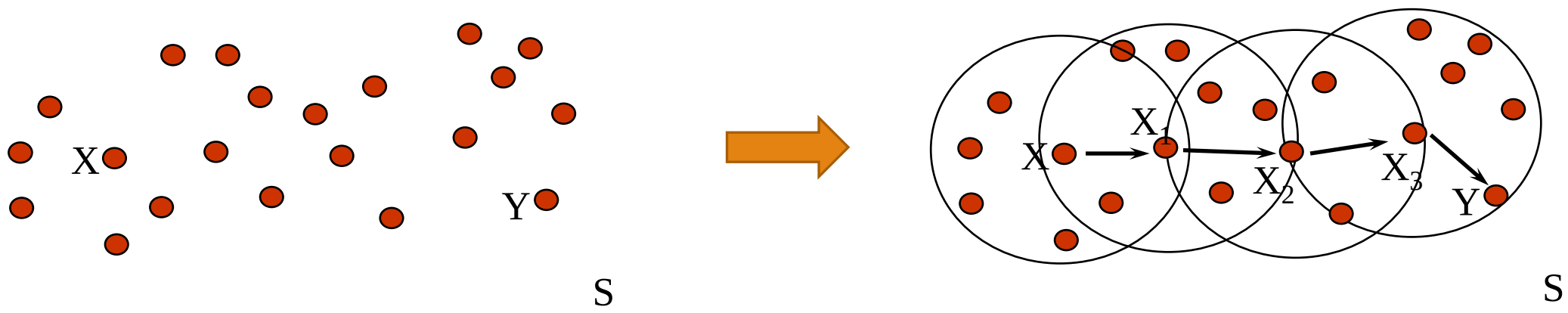
- 定义4：∀Y，X∈S，若X是一个核心对象且Y∈ ε(X)，则称点Y从X出发关于(ε, MinPts)是直接密度可达的，简称从X到Y是直接密度可达的。
  - 例：根据定义4，下图中的对象从X到Y是直接密度可达的。然而从X到Z_i (i=1,2,3)就不是直接密度可达的。



➢ 基本概念

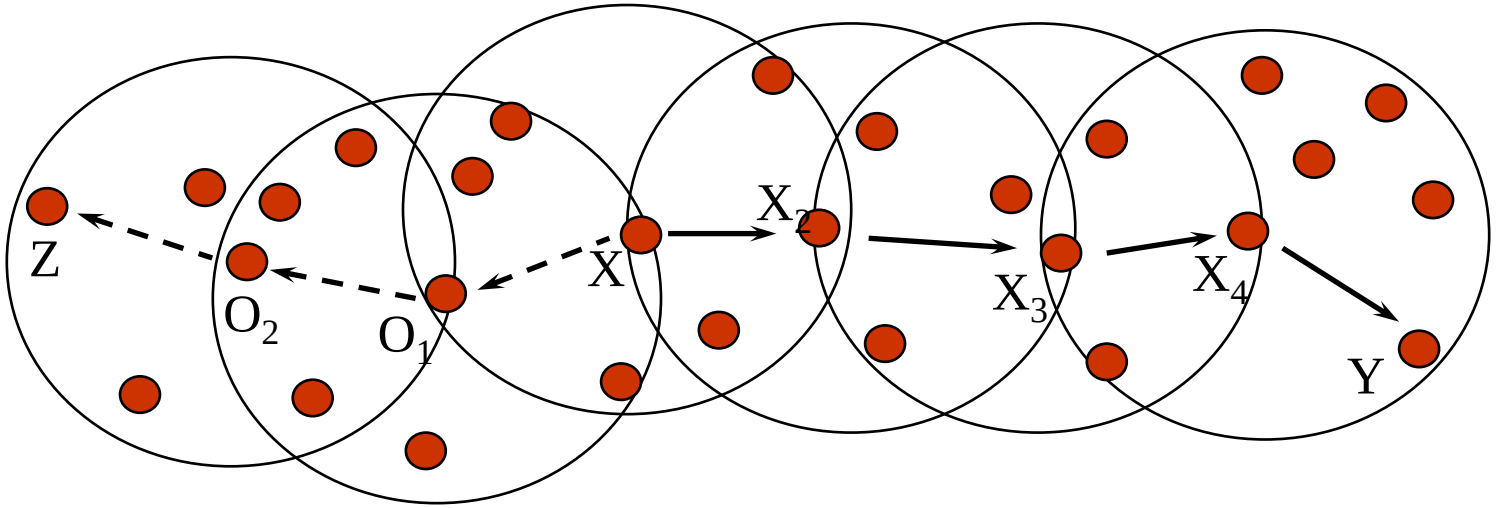
- 定义5：如果S存在一个对象链X_1，X_2，…，X_n，$X_1=X$，$X_n=Y$，且从$X_i(1\le i\le n-1)$到X_{i+1}是直接密度可达的，则称从X到Y是密度可达的。

- 例：对包含21个点的平面数据集S（如图所示），如果密度参数ε=2.5且MinPts=8。请说明从X到Y关于(ε, MinPts)是密度可达的。



基本概念

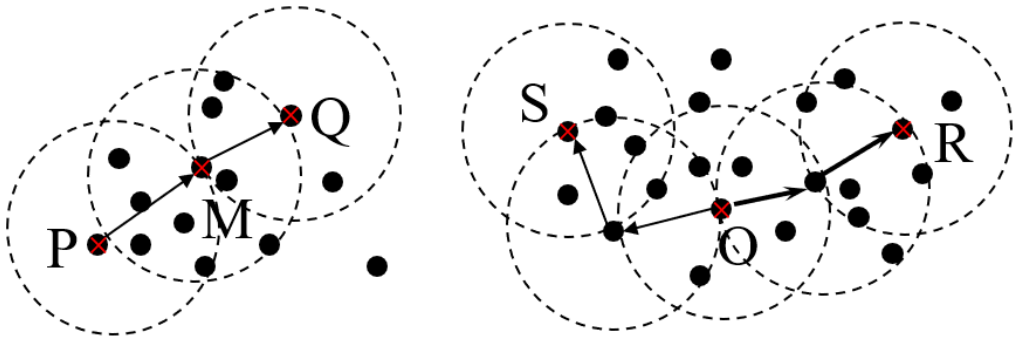
- 定义6：对于任意的Y，Z∈S，若存在X∈S，使从X到Y，从X到Z都是关于(ε, MinPts)密度可达的，则称点Y和Z关于(ε, MinPts)是密度相连的。



➢ 基本概念

- 例：设MinPts=3，下面分析中Q、M、P、S、O、R这6个样本点之间的关系

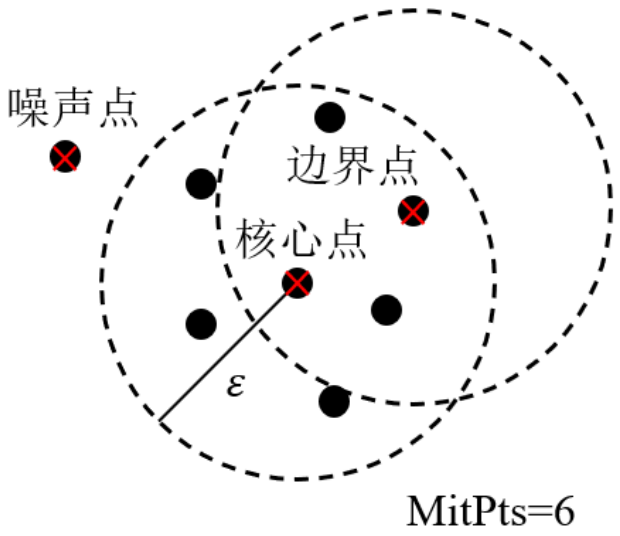
  - M、P、O这3个对象的ε近邻内包含3个以上的点，故它们都是核心对象；
  - M是从P“直接密度可达”的，而Q则是从M“直接密度可达”的；
  - 基于以上结果，Q是从P“密度可达”的，但是P从Q无法“密度可达”（非对称性）
  - 同理，S和R是从O“密度可达”的；因此，O、R、S均是“密度相连”的



由一个核心对象和其密度可达的所有对象构成一个聚类。

➢ 基于密度的聚类方法

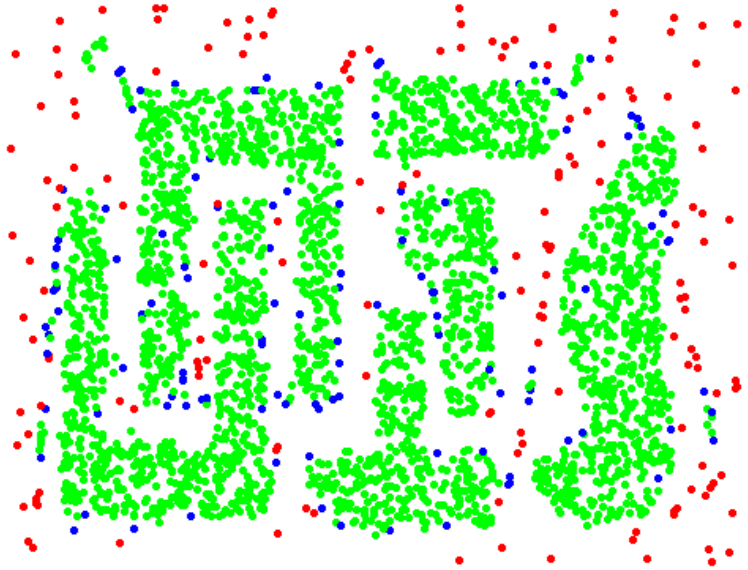
- 基本思想：每个簇的内部点的密度比簇的外部点的密度要高很多，簇是密度相连的点的最大集合，聚类是在数据空间中不断寻找最大集合的过程
- 参数
  - 领域半径ε：邻域的最大半径
  - 最小包含点数MinPts：一个核心对象以ε为半径的邻域内的最小对象数
  - 核心对象（核心点）：近邻点的数目不小于最小包含数
  - 边界点：近邻点数目小于最小包含点且该点位于某个核心点的领域半径范围内，否则为噪声点



➢ DBSCAN算法

- 核心对象、边界点、噪声点

Original Points



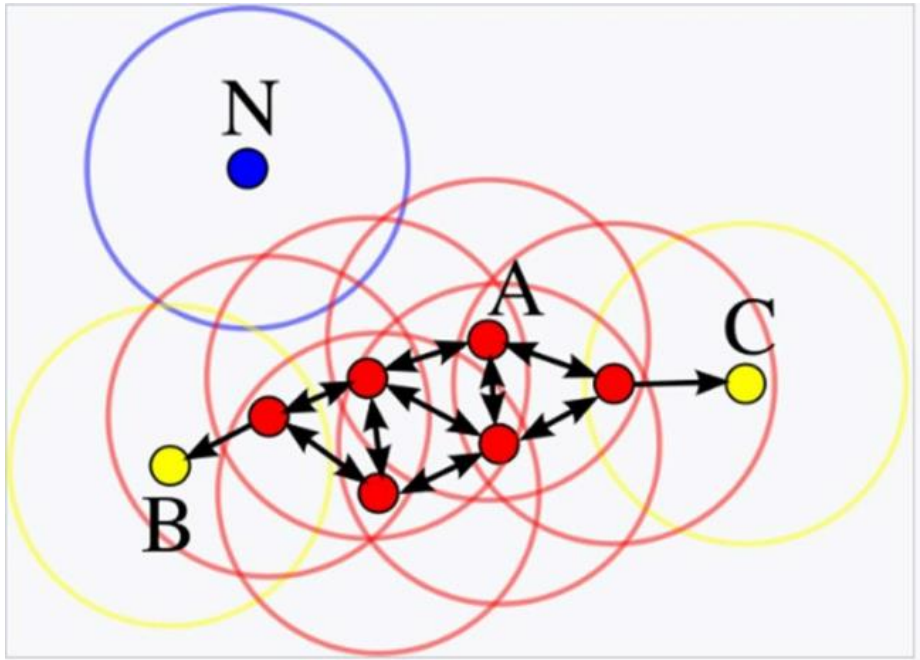
Point types: core, border and noise

Eps = 10, MinPts = 4

填空题 9分

设置

下图中，红色、黄色、蓝色的点分别是：[填空1]，[填空2]，[填空3]。



正常使用填空题需3.0以上版本雨课堂

作答

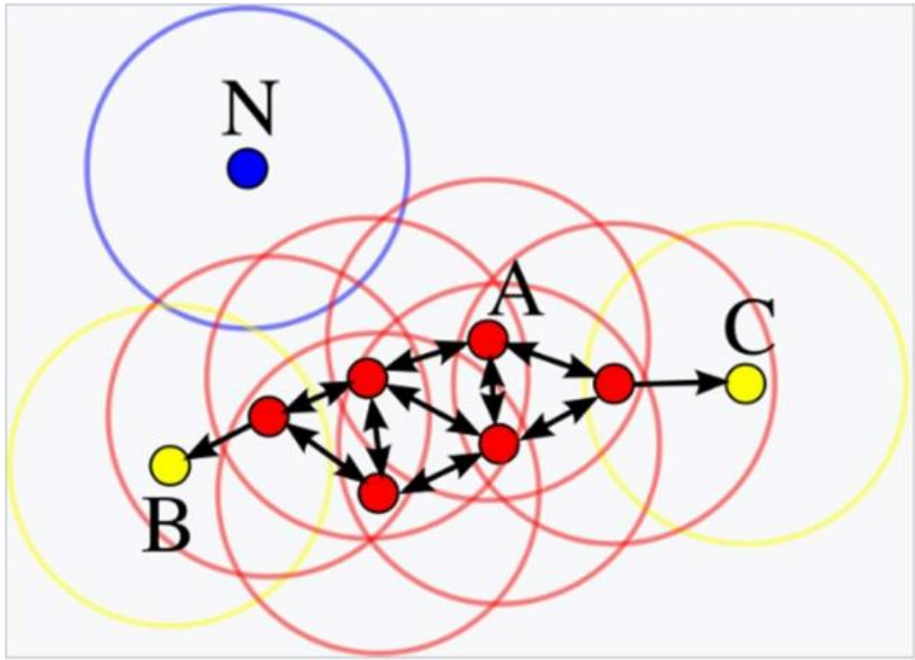
多选题 3分

设置

若N关于(ε, MinPts)为噪声点，怎样操作可以使N不再是噪声点。

A 适当增大ε

B 适当减小MinPts

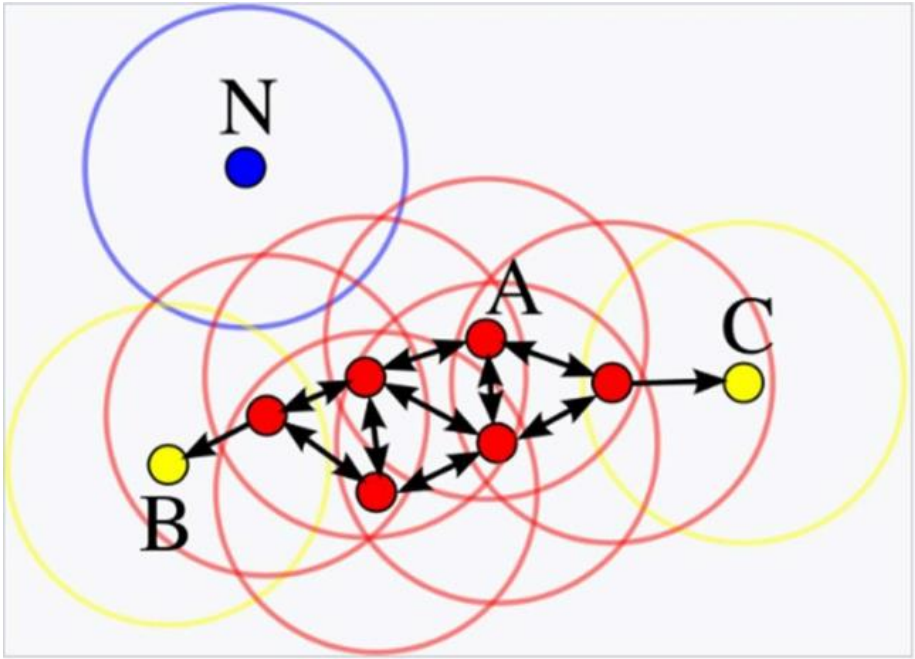


提交

➢ 基本概念

- 对于任意Y∈S，若Y既不是关于密度(ε, MinPts)的核心对象，也不是边界点，则称Y为S关于密度(ε, MinPts)“噪声点”，简称Y为S的噪声点。

S一个对象Y是否为噪声点，完全与给定的密度(ε, MinPts)相关。若Y关于(ε, MinPts)为噪声点，则只要让ε足够大，或适当减小MinPts，就可以使Y不再是噪声点。



➢ DBSCAN算法

- DBSCAN算法两大步骤：

（1）寻找核心点形成临时聚类簇

扫描全部样本点，如果某个样本点ε-领域内点的数目≥MinPts，则将其纳入核心点列表，并将其密度可达的点形成对应的临时聚类簇。

（2）合并临时聚类簇得到聚类簇

对于每一个临时聚类簇，检查其中的点是否为核心点，如果是，将该点对应的临时聚类簇和当前临时聚类簇合并，得到新的临时聚类簇。

➢ DBSCAN算法

◘ DBSCAN算法描述：

输入：数据集D；领域半径ε；给定点在ε邻域内成为核心对象的最小邻域点数MinPts

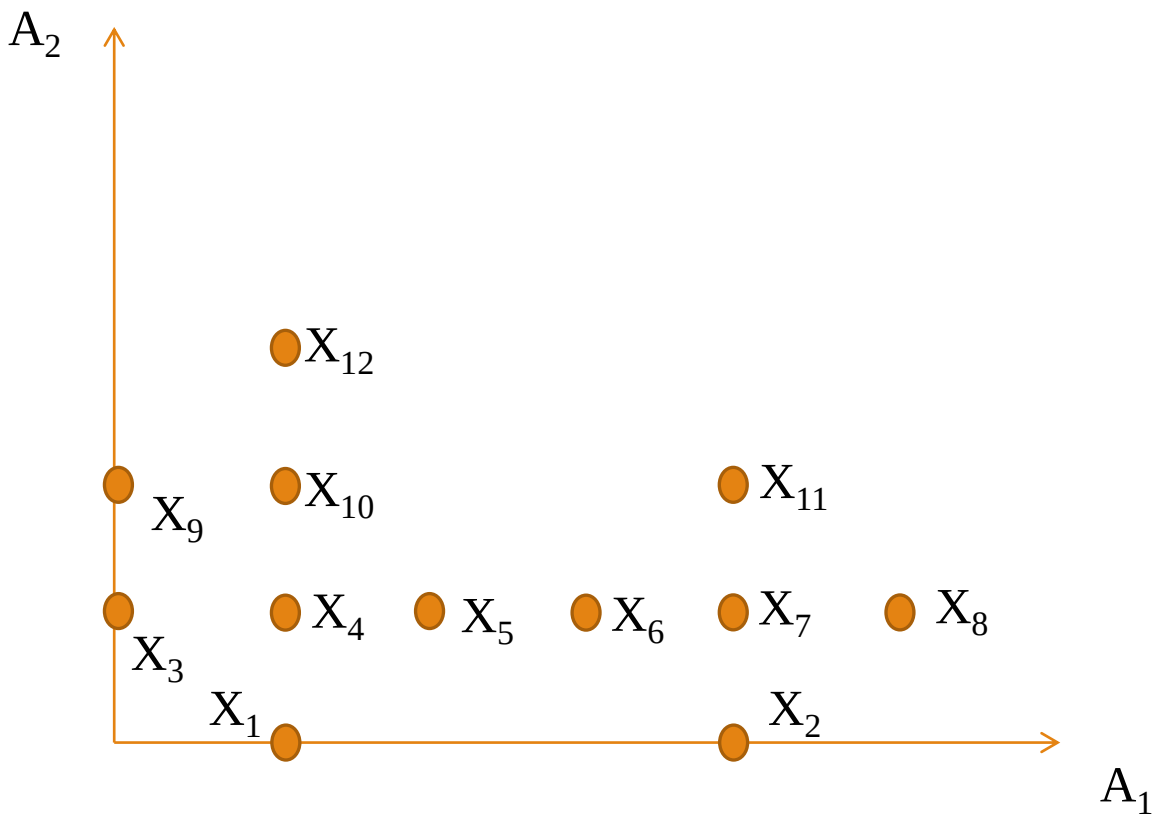
输出：簇集合C

1. repeat
2. 判断输入点是否为核心对象
3. 找出核心对相关的ε邻域中的所有直接密度可达点
4. until所有输入点都判断完毕
5. repeat
6. 针对所有核心对象的ε邻域中的所有直接密度可达点
7. 找到最大密度相连对象集合，中间涉及到一些密度可达对象的合并
8. until所有核心对象的ε邻域都遍历完毕

算法实例

设数据集S共有12个对象，密度(ε=1, MinPts=4)。试用DBSCAN算法对其进行聚类。

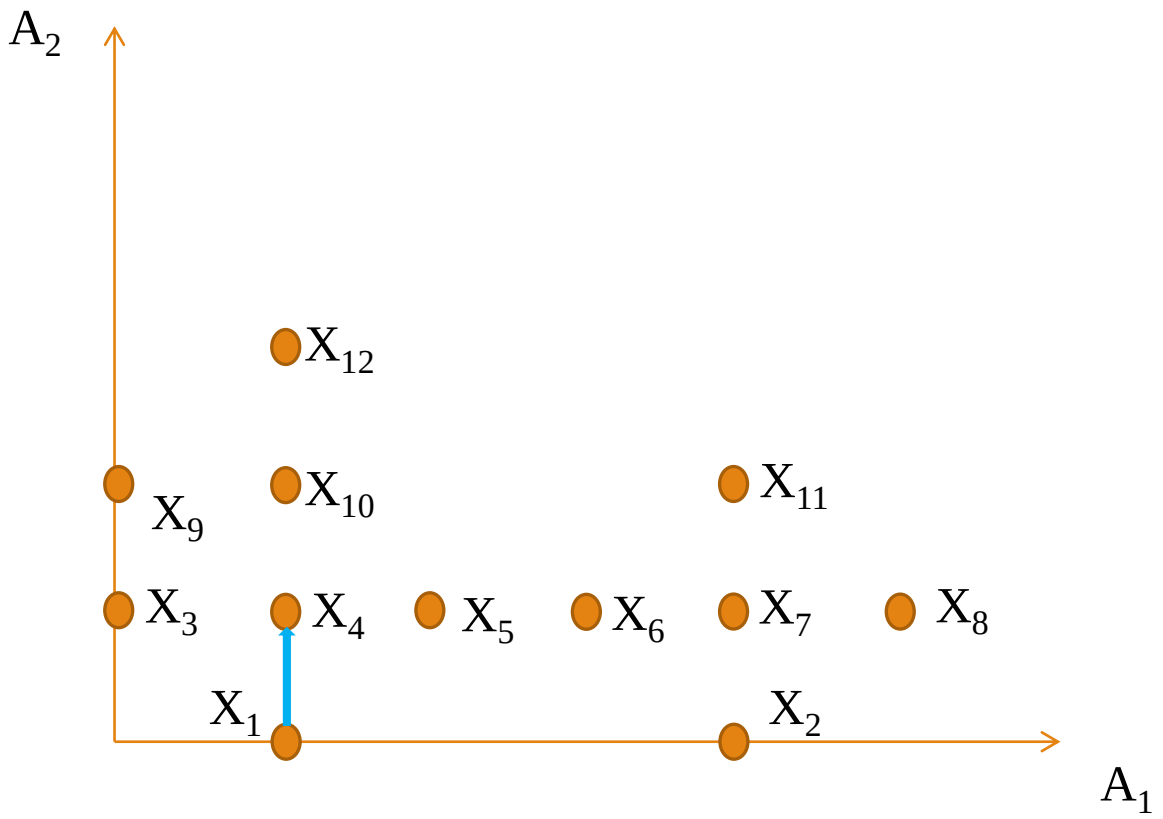
id	A_1	A_2	id	A_1	A_2
X_1	1	0	X_7	4	1
X_2	4	0	X_8	5	1
X_3	0	1	X_9	0	2
X_4	1	1	X_{10}	1	2
X_5	2	1	X_{11}	4	2
X_6	3	1	X_{12}	1	3



➢ 算法实例

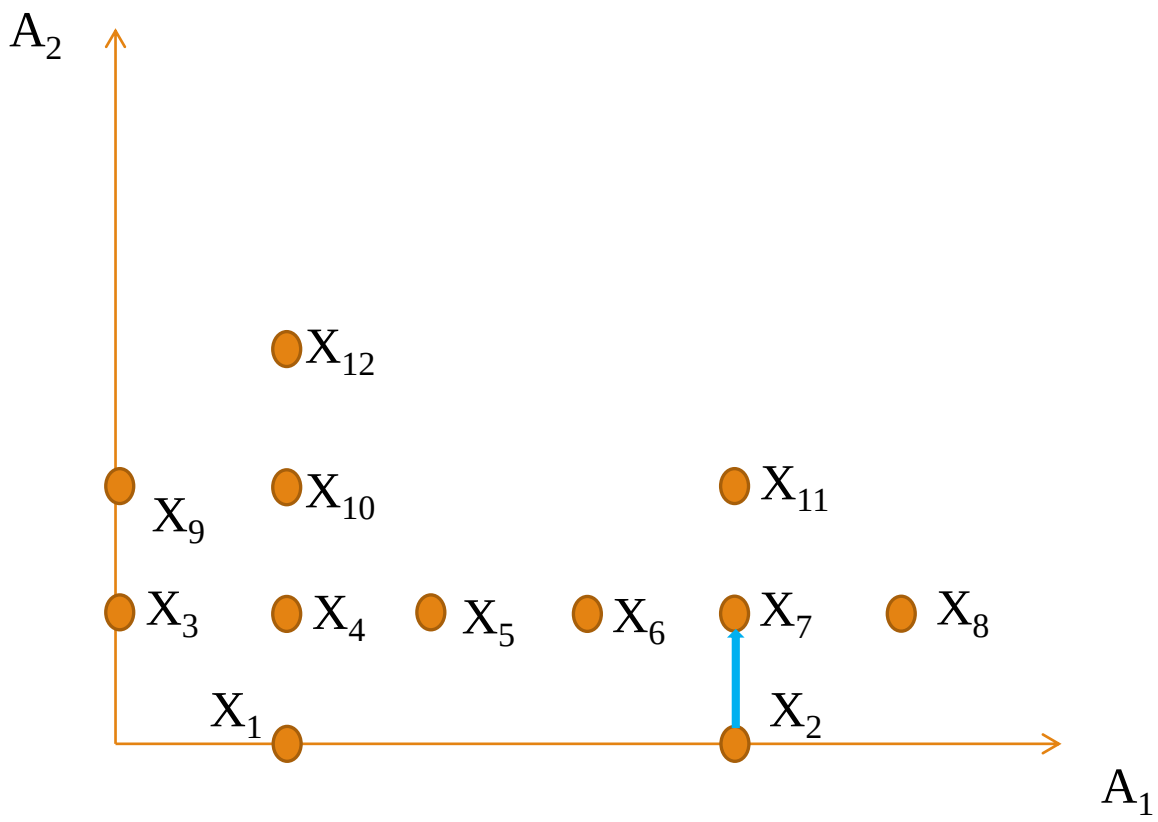
◘ 步骤

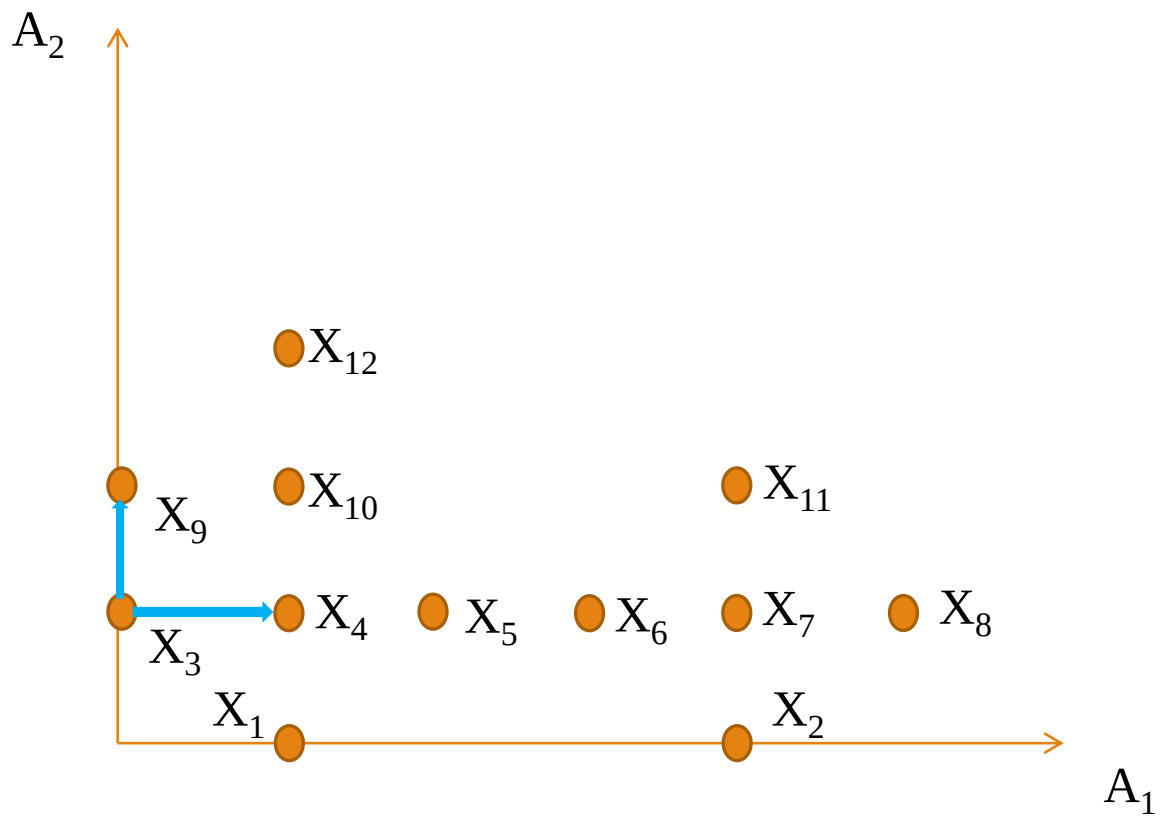
- 在S中选择一点X_1
- 由于以X_1为中心，ε=1为半径的圆内仅包含两个点$\{X_1, X_4\}$
- 即$N(X_1)$=2<MinPts=4
- 因此它不是核心点，继续选择下一个。



➢ 算法实例

- 以此类推，X_2、X_3均不是核心点



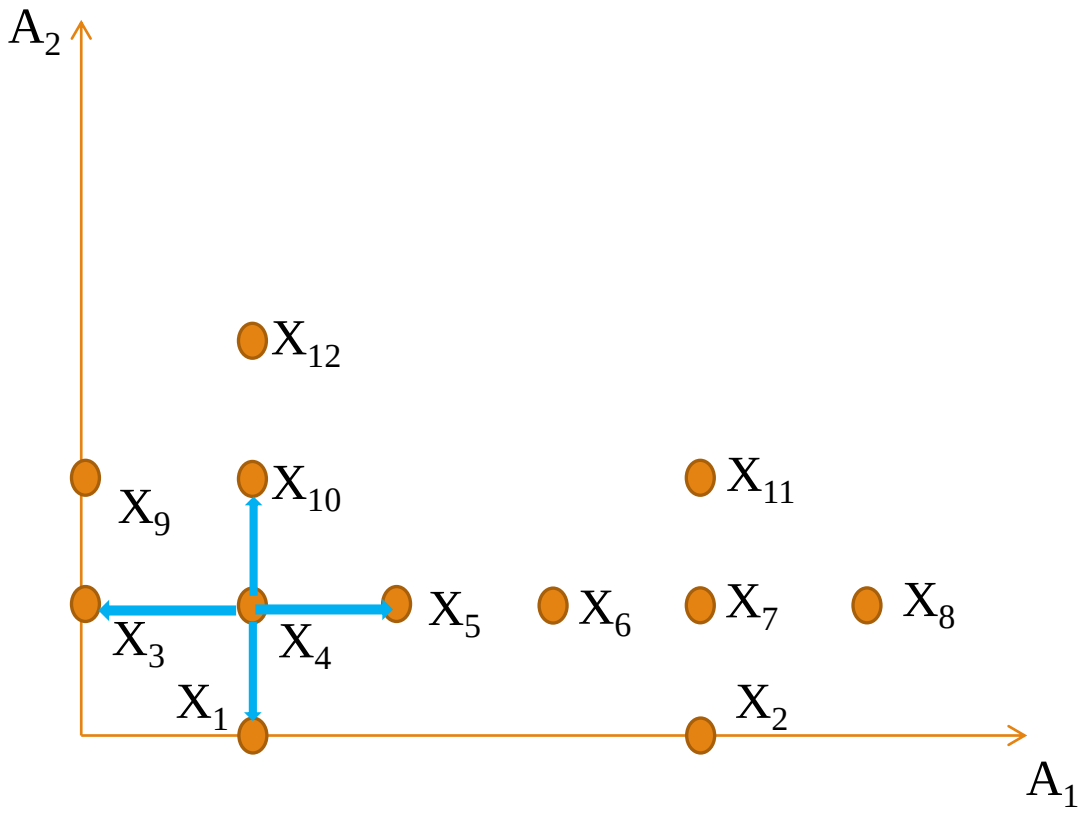


➢ 算法实例

□ 步骤

- 在S中选择一点X_4
- 由于以X_4为中心，ε=1为半径的圆内包含

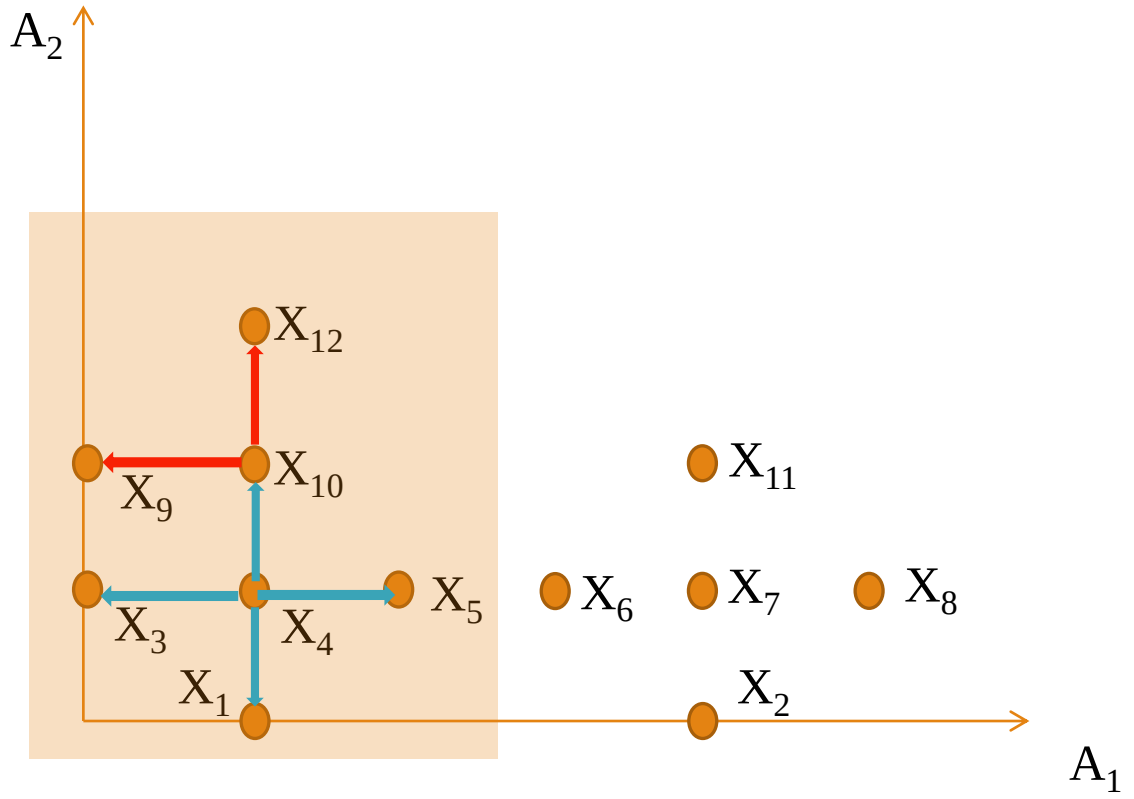
  五个点$\{X_1, X_3, X_4, X_5, X_{10}\}$
- 即$N(X_4)$=5>MinPts=4
- 因此它是关于密度(ε=1, MinPts=4)的一个核心点。



➢ 算法实例

◘ 步骤

- 从X_4出发直接密度可达X_{10}
- 由于以X_{10}为中心，ε=1为半径的圆内包含四个点$\{X_{10}，X_{12}，X_9，X_4\}$
- 即$N(X_{10})$=4=MinPts=4
- X_{10}也是关于密度(ε=1, MinPts=4)的一个核心点。

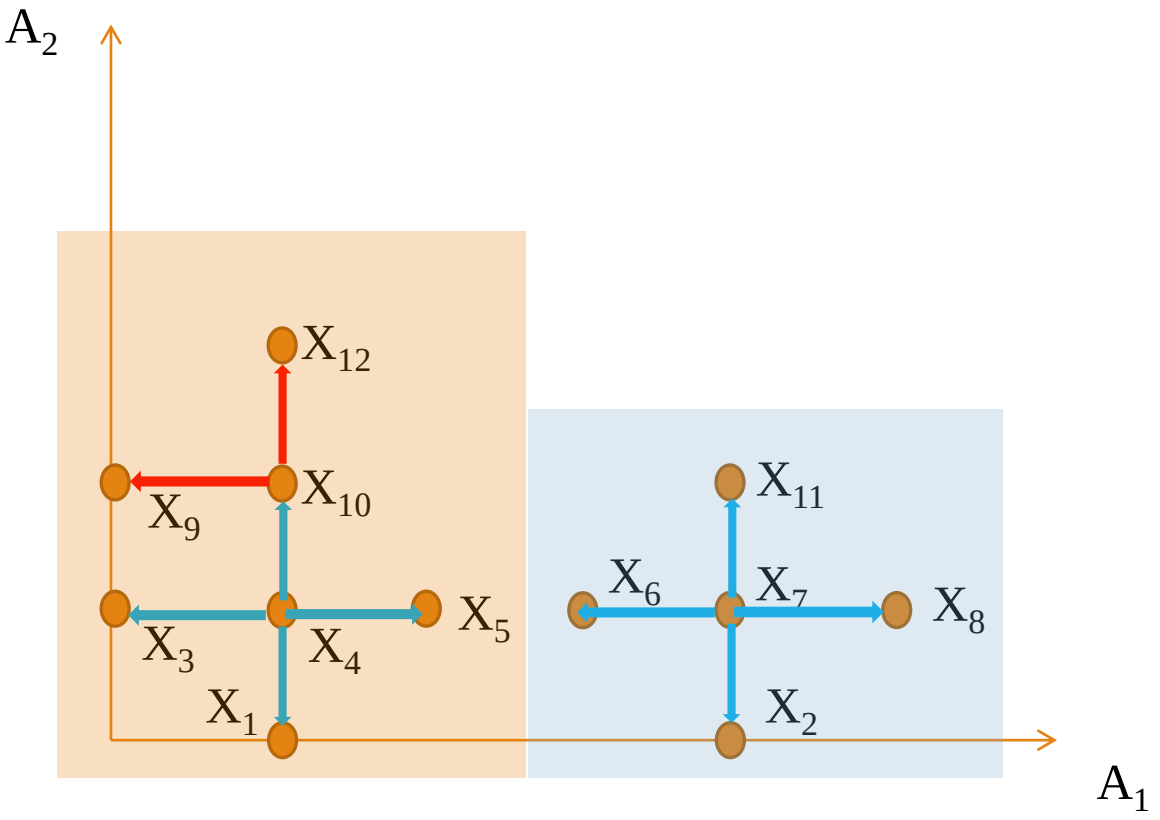


- 得到以X_4出发，密度可达的所有对象形成的簇
- C_1=$\{X_1，X_3，X_4，X_5，X_9，X_{10}，X_{12}\}$

➢ 算法实例

◘ 步骤

- X_5已在C_1中，无需判断。
- X_6不是核心点。
- 选择X_7，它是核心点
- 得到以X_7出发，密度可达的所有对象形成的簇
- C_2={X_2，X_6，X_7，X_8，X_{11}}



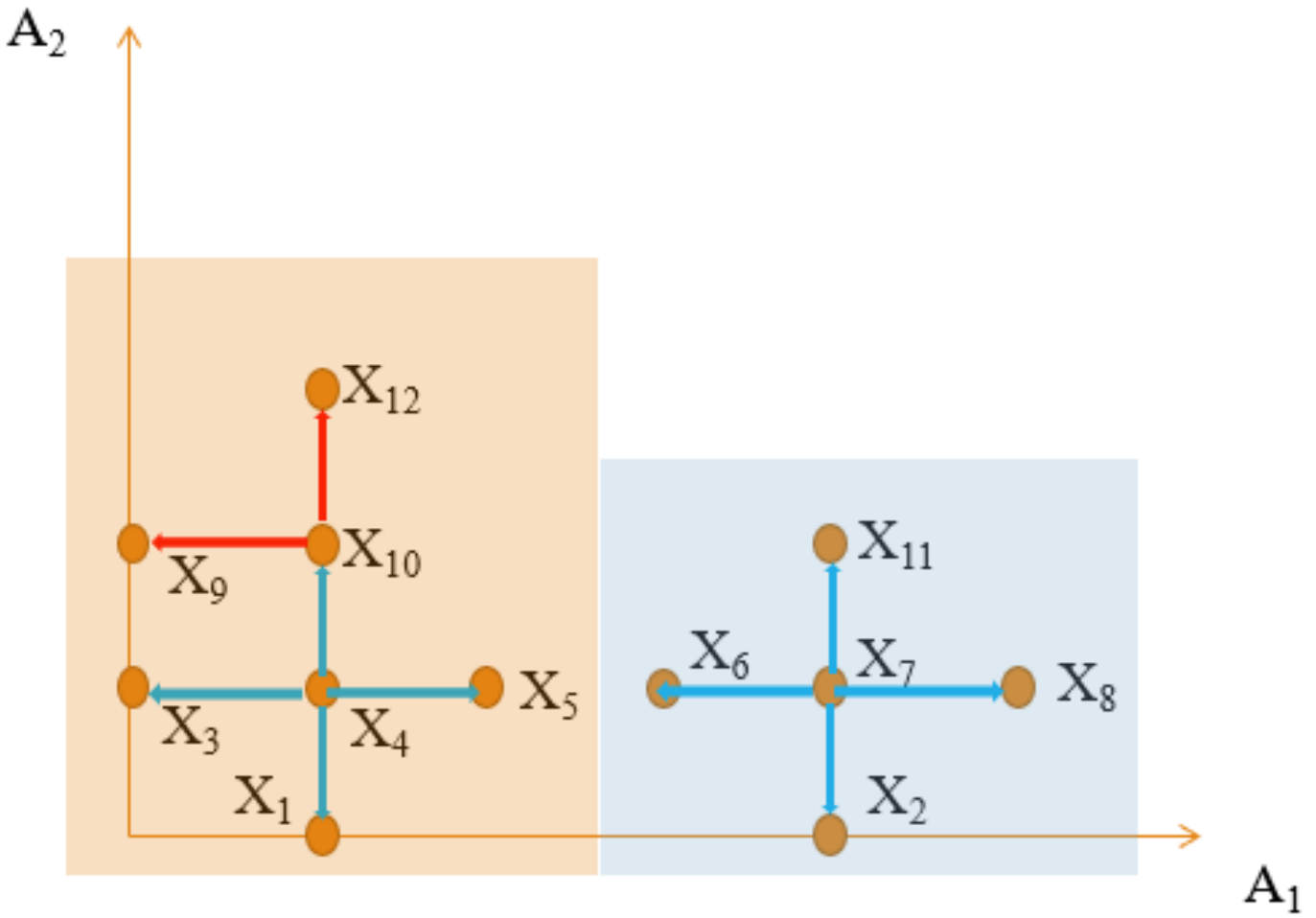
➢ 算法实例

- 类似地分析剩下的点(X_8~X_{12})，结果汇总如表所示。

对象	ε中对象数	生成的簇	对象	ε中对象数	生成的簇
X_1	2	无	X_7	5	C2
X_2	2	无	X_8	2	已在C2中
X_3	3	无	X_9	3	已在C1中
X_4	5	C1	X_{10}	4	已在C1中
X_5	3	已在C1中	X_{11}	2	已在C2中
X_6	3	无	X_{12}	2	已在C1中

➢ 算法实例

- 聚类结果



➢ DBSCAN算法优缺点

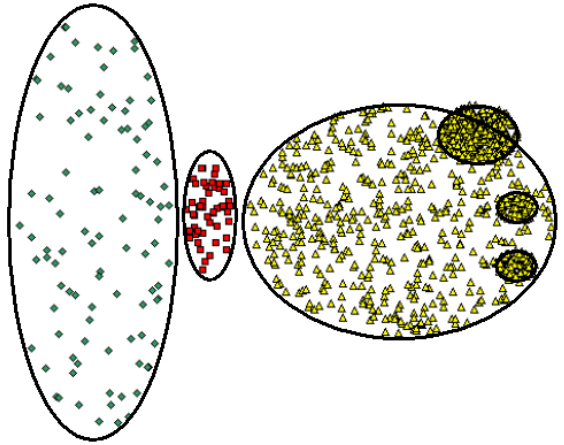
◘ 优点

- ✓ DBSCAN不需要事先知道要形成的簇的数量
- ✓ 聚类簇的形状没有偏倚
- ✓ 对噪声点不敏感，并可以在需要时输入过滤噪音的参数

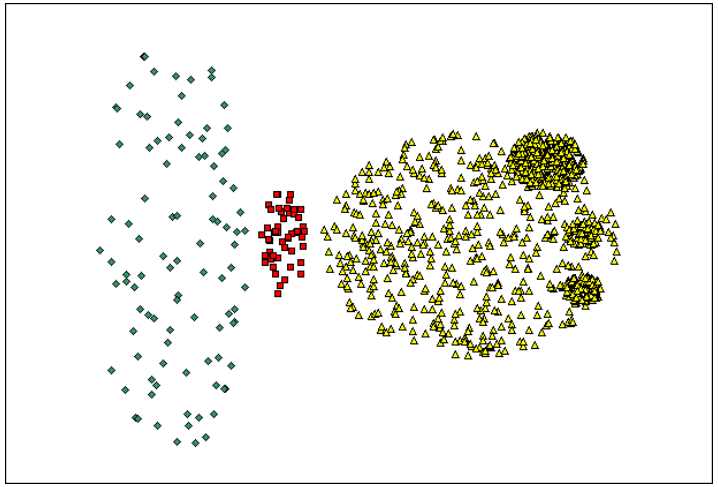
◘ 缺点

- 当数据量增大时，要求较大的内存支持，I/O消耗也很大
- 当遇到密度变化的数据集（即密度分布不均匀）时，聚类间距相差很大时，聚类质量较差
- 不能很好反映高维数据
- 初始参数需手动设置，并且聚类质量十分依赖这两个参数的取值

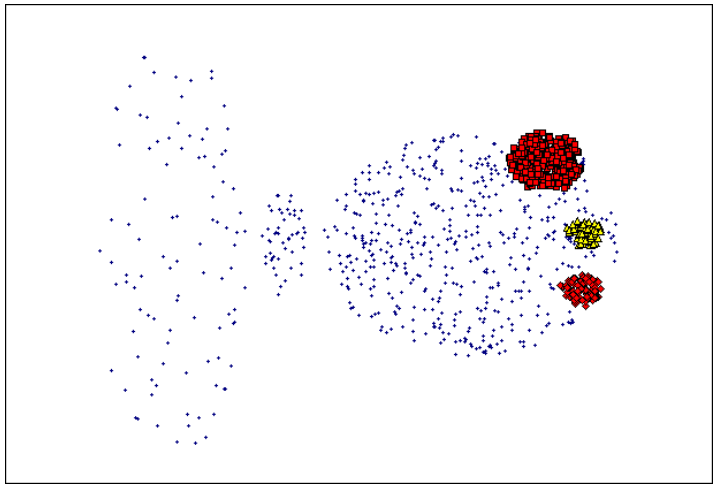
➢ DBSCAN算法优缺点



Original Points



(MinPts=4, Eps=9.75)



(MinPts=4, Eps=9.92)

➢ Optics算法

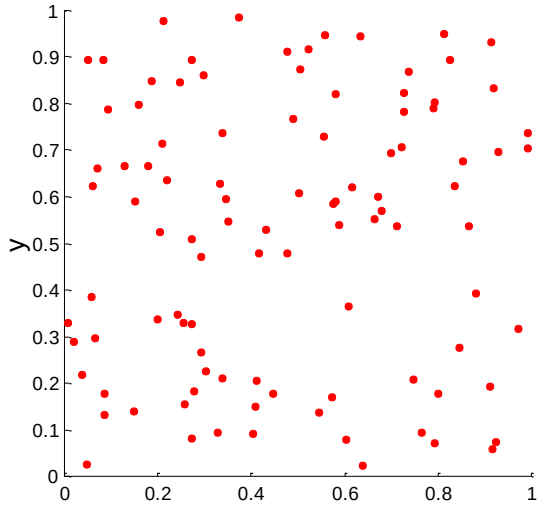
- 在密度高低的衡量上，增加了两个新的概念“核心距离”和“可达距离”以降低聚类结果对初始参数的敏感度

目录

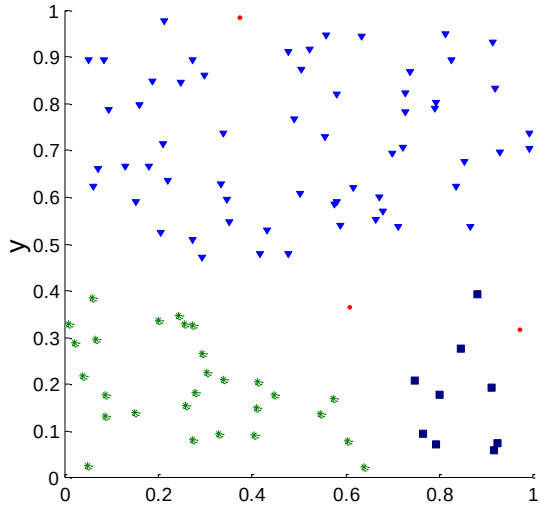
5 聚类方法性能评估

➢ 聚类方法比较

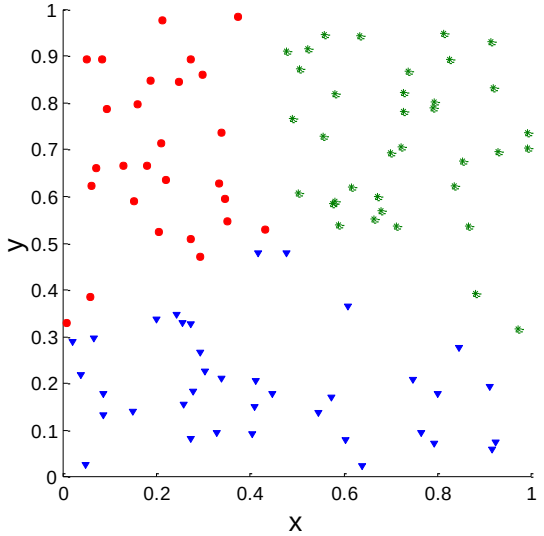
Random Points



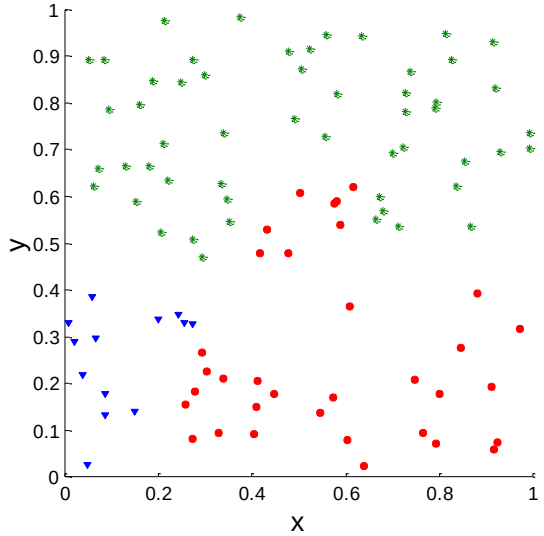
DBSCAN



K-means



Complete Link



5 聚类方法性能评估

➢ 聚类方法评估的原因

- 一个好的聚类方法要能产生高质量的聚类结果——簇，这些簇要具备以下两个特点：
  - 高的簇内相似性
  - 低的簇间相似性
- 聚类质量的度量方法
  - ✓ 外部准则法
  - ✓ 内部准则法
  - ✓ 相对准则法

5 聚类方法性能评估

➢ 聚类方法评估

- 对于聚类结果的评价方法一般可以分为内部评估法（internal evaluation）与外部评估方法（external evaluation）
  - **外部评估方法**是指假定知道真实标签（ground truth ）的情况下来评估聚类结果的好坏

    例如：调整兰德指数（Adjusted Rand Index）、互信息评分（Mutual Information）等
  - **内部评估法**是不借助于外部信息，依赖数据自身特征和量值对聚类结果进行评价

    例如：轮廓系数（Silhouette Coefficient）、CH指标（Calinski-Harabasz Index）等
  - **相对评估法**是针对同一个聚类算法的不同参数设置进行算法测试(如不同的分簇个数)，最终选择最优的参数设置和聚类模式

➢ 内部评估法

- 非监督的度量方法，没有可参考的外部信息，只能依赖数据集自身的特征和量值对聚类结果进行评价
  - 轮廓系数

    对于数据集D中的任意对象x_i，假设聚类算法将x_i划分到簇C

$$s_i = \frac{b_i - a_i}{\max(a_i, b_i)}$$

    - a_i：x_i与本簇中其他对象之间的平均距离，反映了x_i所属簇的紧密程度；越小，簇的紧密程度越好
    - b_i：x_i与最近簇的对象之间的平均距离，表示x_i与其他簇的分离程度，该值越大，分离度越高
    - s_i：在[-1,1]之间，可用于评价对象x_i是否适合所属的簇；值越大，分配越合理

内部评估法

- CH指标：通过类内协方差矩阵描述紧密度，类间协方差描述分离度

$$CH(k)=\frac{trB_k/(k-1)}{trW_k/(n-k)}$$

$$W_k=\sum_{q=1}^{k}\sum_{x\in C_q}(x-C_q)(x-C_q)^T \qquad B_k=\sum_q n_q(C_q-C)(C_q-C)^T$$

- B_k：簇之间的协方差矩阵；W_k：簇内部数据的协方差矩阵；n：训练集样本数；k：簇个数；C_q：簇q的中心样本；C：训练集的中心样本；n_q：簇q的样本数目；tr：矩阵的迹

CH指标值越大表示聚类结果的性能越好

目录

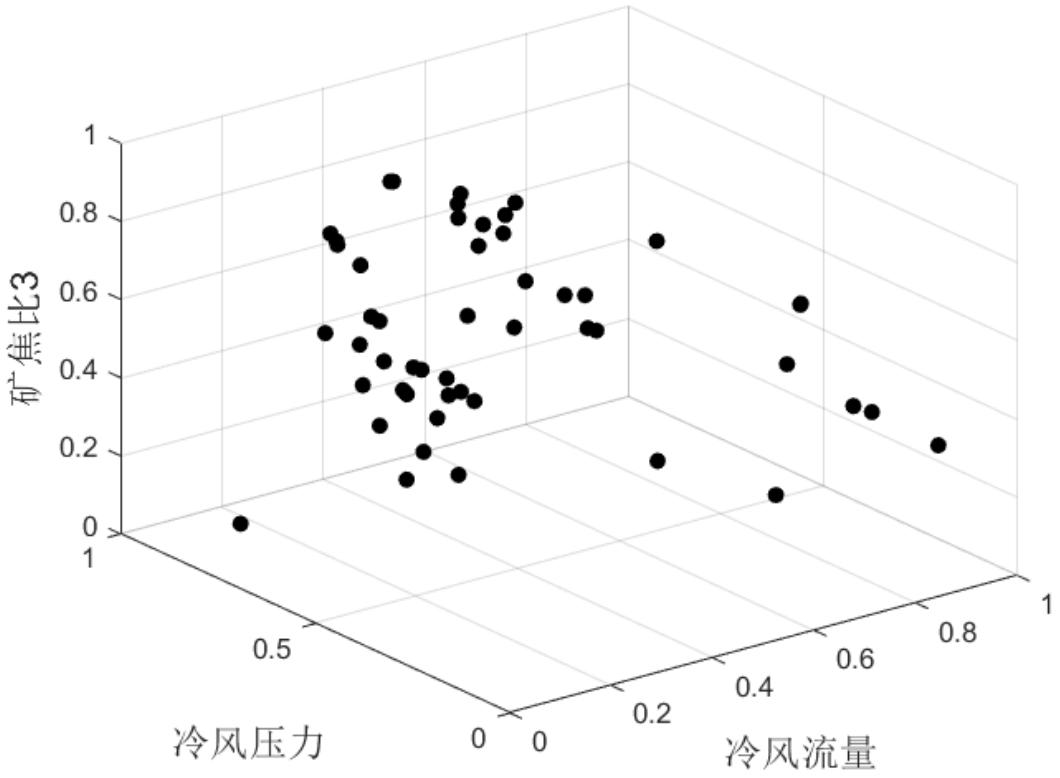
©ISTOCKPHOTO.COM/ZAPP2PHOTO

IEEE INDUSTRIAL ELECTRONICS MAGAZINE ■ DECEMBER 2019 PP.38

6 实例：

➢ 高炉煤气操作参数聚类过程

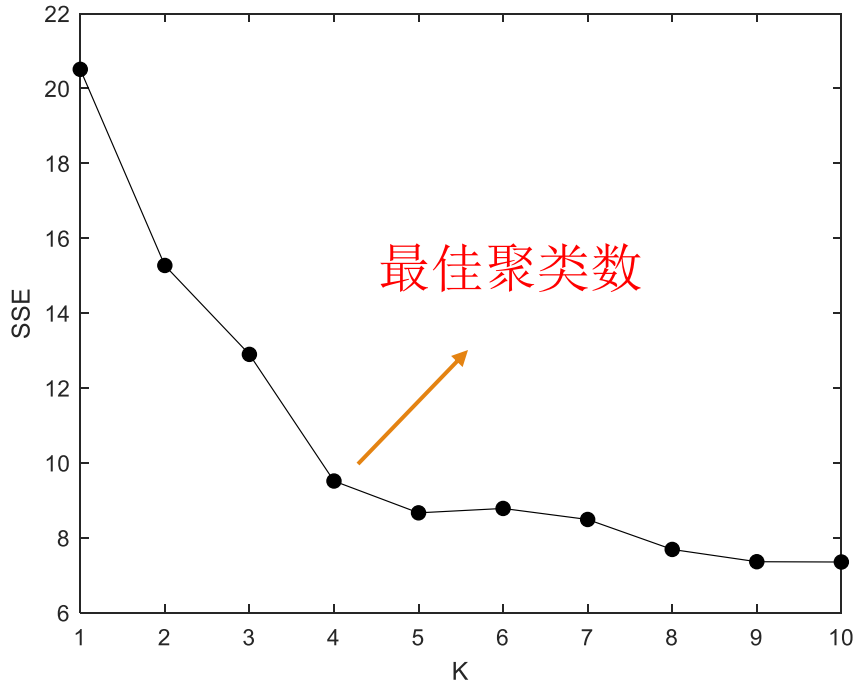
- 高炉煤气利用率的大小被确立为用来反映高炉冶炼效果的指标之一
- 高炉煤气利用率的大小受各种操作参数的影响
- 通过工况的分类构建不同的预测模型往往能获得更好的预测效果
- 利用聚类方法对各操作参数做分类分析，为后续高炉煤气利用率预测建模做准备



6 实例：

➢ K-means聚类方法聚类

◘ 利用肘部法确定最佳聚类数



✓ 根据K-means聚类算法进行聚类，初始聚类中心选择点1、点2、点3，K取4

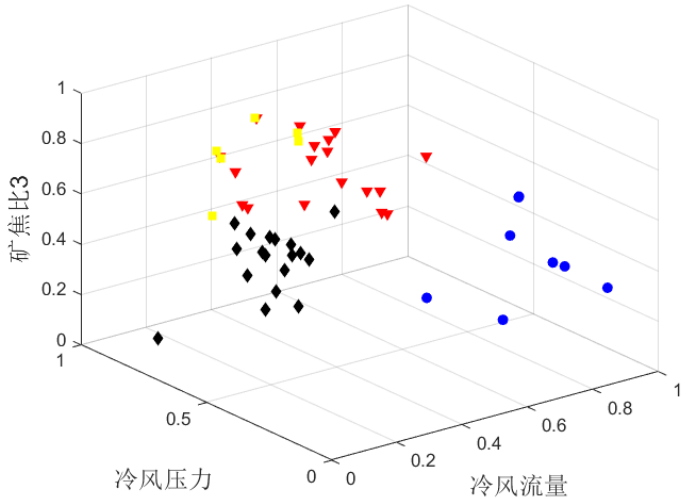
6 实例：

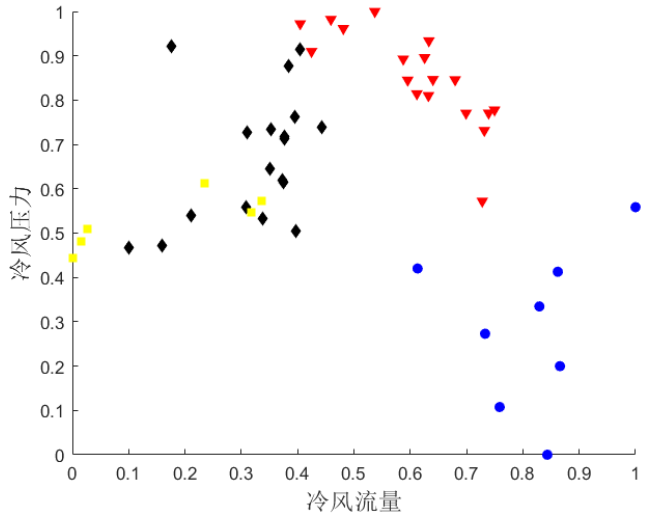
➢ K-means聚类方法聚类

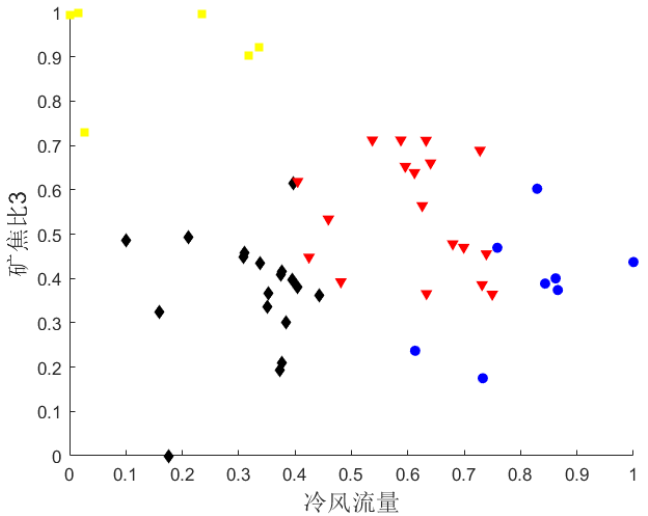
◘ 聚类结果

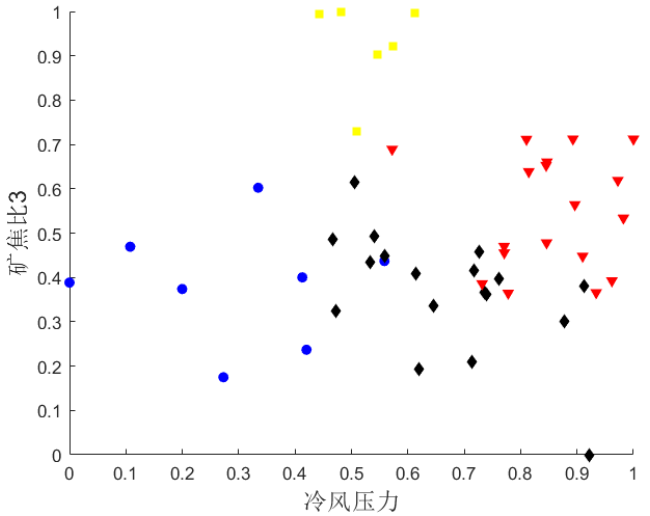
根据聚类结果，将两两操作参数的二维映射图进行展示

- ✓ 不同颜色的散点代表不同簇
- ✓ 同颜色的散点代表相同簇
- ✓ 聚类性能评估：利用轮廓系数对聚类结果进行评估，计算得s=0.7729。









6 实例：

➢ DBSCAN聚类方法聚类

- 设置不同参数，利用轮廓系数评估聚类质量

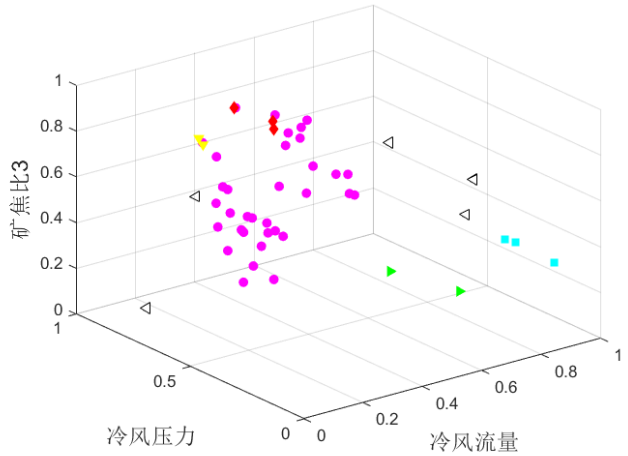
MinPts / ε	1	2	3	4	5
0.1	0.4634	0.4758	0.2729	0.2815	0.0081
0.2	0.5432	0.6539	0.5430	0.4270	0.4270
0.3	0.3857	0.3857	0.3857	0.3857	0.5523
0.4	-1	-1	-1	-1	-1

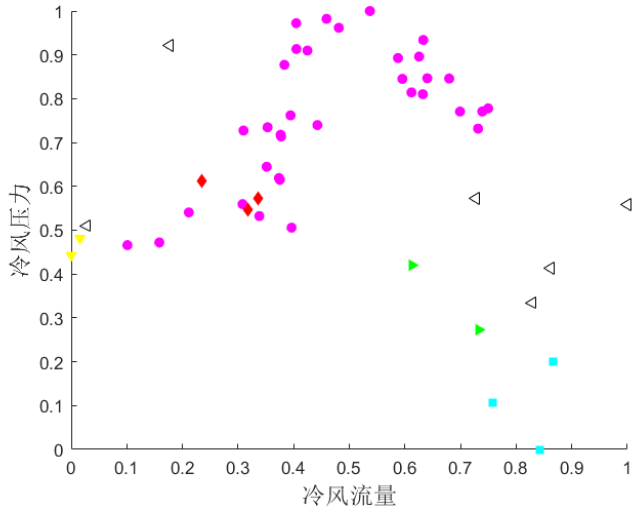
✓ 根据轮廓系数评估结果选择最佳的ε、MinPts

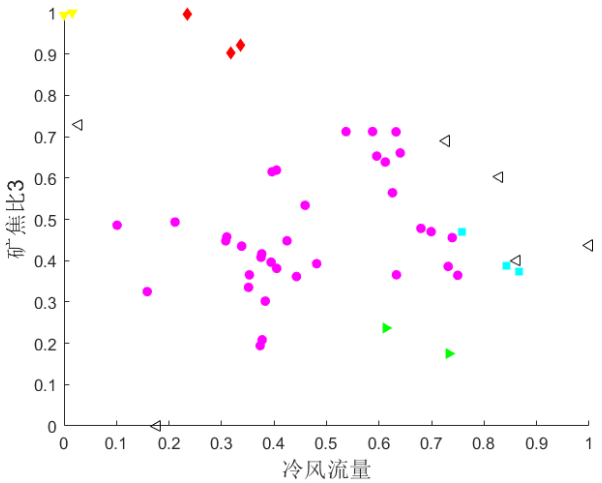
6 实例：

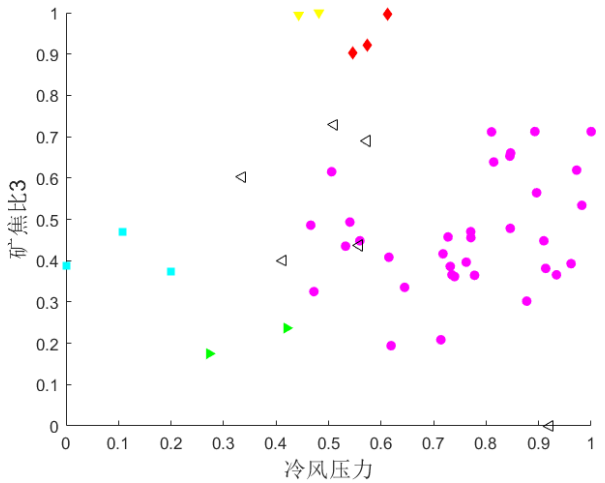
➢ DBSCAN聚类方法聚类

◘ 聚类结果









- ✓ 不同颜色的散点代表不同簇
- ✓ 同颜色的散点代表相同簇
- ✓ △ 代表噪声点

6 实例：

➢ 聚类结果对比

- K-means：轮廓系数为0.7729
- DBSCAN：轮廓系数为0.6539

→ 轮廓系数越大，聚类效果越好

- 结论：从轮廓系数的比较可知，K-means聚类相对DBSCAN聚类更合适该数据集的聚类

对于密度分布不均、维数较高的数据集，使用K-means聚类相对更合适

7 本章小结

- 聚类分析的基本概念
  - 聚类分析的定义
  - 聚类算法的性能要求
- 划分聚类方法
  - K-means算法
  - K-medoids算法
  - K-means++算法
- 层次聚类方法
  - 距离度量
  - 凝聚的与分裂的层次聚类算法
- 基于密度的聚类方法
  - DBSCAN算法
  - OPTICS算法
- 聚类分析性能评估
  - 外部准则法、内部准则法和相对准则法

参考文献

- 章永来, 周耀鉴. 聚类算法综述[J]. 计算机应用, 2019, 39(07): 1869-1882.
- G. Sun, C. Jiang, P. Cheng, Y. Liu, X. Wang, Y. Fu, Y. He, Short-term wind power forecasts by a synthetical similar time series data mining method, *Renewable Energy*, 115(2018): 575–584.
- M. Ding, H. Zhou, H. Xie, M. Wu, K.-Z. Liu, Y. Nakanishi, R. Yokoyama, A time series model based on hybrid-kernel least-squares support vector machine for short-term wind power forecasting, *ISA Transactions*, 108 (2021): 58–68.
- Salma Ben Jemaa, Mohamed Hammami, Hanene Ben-Abdallah. Data-mining process: application for hand detection in contact free settings[J]. *IET Image Processing*, 2013, 7(8).
- Karataş F, Korkmaz S A. Big Data: controlling fraud by using machine learning libraries on Spark[J]. *International Journal of Applied Mathematics Electronics and Computers*, 2018, 6(1): 1-5.
- Pinheiro D N, Aloise D, Blanchard S J. Convex fuzzy k-medoids clustering[J]. *Fuzzy Sets and Systems*, 2020, 389: 66-92.
- Veselina Bureva, Stanislav Popov, Velichka Traneva, Stoyan Tranev. Generalized Net Model of Cluster Analysis Using CLIQUE: Clustering in Quest[J]. *International Journal Bioautomation*, 2019, 23(2).