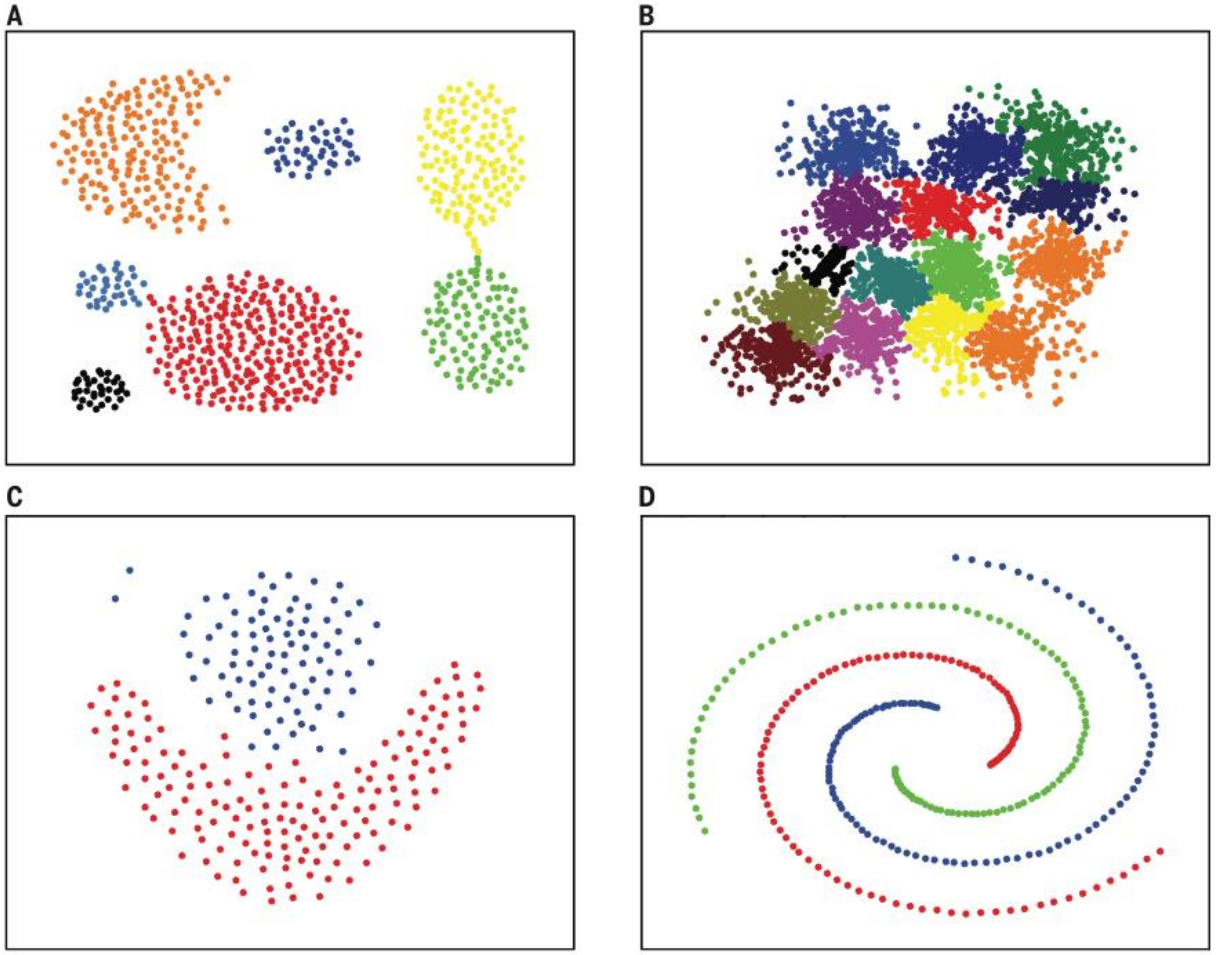
数据挖掘

第四章 聚类基础

2023秋

聚类效果



学习目标

- 描述聚类分析的概念，区分聚类和分类
- 应用基本的聚类方法解决实际应用问题
- 了解聚类评估的任务有哪些

目 录

1. 聚类分析概述

2. 基本聚类方法

3. 聚类评估

1 聚类分析概述

- 什么是聚类?
  - 是把数据对象集合按照相似性划分成多个子集的过程。
  - 每个子集是一个簇（cluster），使得簇中的对象彼此相似，但与其他簇中的对象不相似。
- 聚类是无监督学习，因为给的数据没有类标号信息

分类 vs. 聚类

- 分类
  - 有监督学习
  - 通过有标签样本学习分类器
- 聚类
  - 无监督学习
  - 通过观察学习，将数据分割成多个簇

聚类的应用

- 商业领域
  - 聚类分析被用来发现不同的客户群，并且通过购买模式刻画不同的客户群的特征。
- 电子商务
  - 聚类出具有相似浏览行为的客户，并分析客户的共同特征，可以更好的帮助电子商务的用户了解自己的客户，向客户提供更合适的服务。
- 舆情监控
  - 发现热点主题、话题、事件等
  - 发现未知异常

数据挖掘对聚类的要求

- **可扩展性**
  - 许多聚类算法在小于几百个数据对象的小数据集合上工作得很好
  - 而一个大规模数据库可能包含几百万个对象
- **处理不同数据类型的能力**
  - 许多聚类算法专门用于数值类型的数据
  - 而实际应用涉及不同的数据类型，如二元的、分类的、图像的等
- **发现任意形状的能力**
  - 基于距离的聚类算法往往发现的是球形的聚类
  - 而现实的聚类是任意形状的
- **用于决定输入参数的领域知识最小化**
  - 聚类结果对于输入参数十分敏感
  - 而参数很难决定，聚类的质量也很难控制

数据挖掘对聚类的要求

- **处理噪声数据的能力**
  - 很多数据库都包含了孤立点，缺失或错误的数据
  - 而一些聚类算法对于这样的数据敏感，可能导致低质量的聚类结果
- **对输入数据的顺序不敏感和增量聚类**
  - 同一个数据集合，以不同的次序提交给同一个算法，应该产生相似的结果
  - 能将新加入的数据合并到已有聚类中
- **高维度**
  - 许多聚类算法擅长处理低维数据，可能只涉及两到三维
  - 而数据库或者数据仓库可能包含若干维或属性

目 录

1. 聚类分析概述

2. 基本聚类方法

3. 聚类评估

2 基本聚类方法

2.1 划分方法

- 划分方法：将有n 个对象的数据集D划分成k个簇，并且k≤n，满足如下的要求：
  - 每个簇至少包含一个对象
  - 每个对象属于且仅属于一个簇
- 基本思想
  - 首先创建一个初始k划分(k为要构造的划分数)
  - 然后不断迭代地计算各个簇的聚类中心并依新的聚类中心调整聚类情况，直至收敛
- 目标
  - 同一个簇中的对象之间尽可能“接近”或相关
  - 不同簇中的对象之间尽可能“远离”或不同

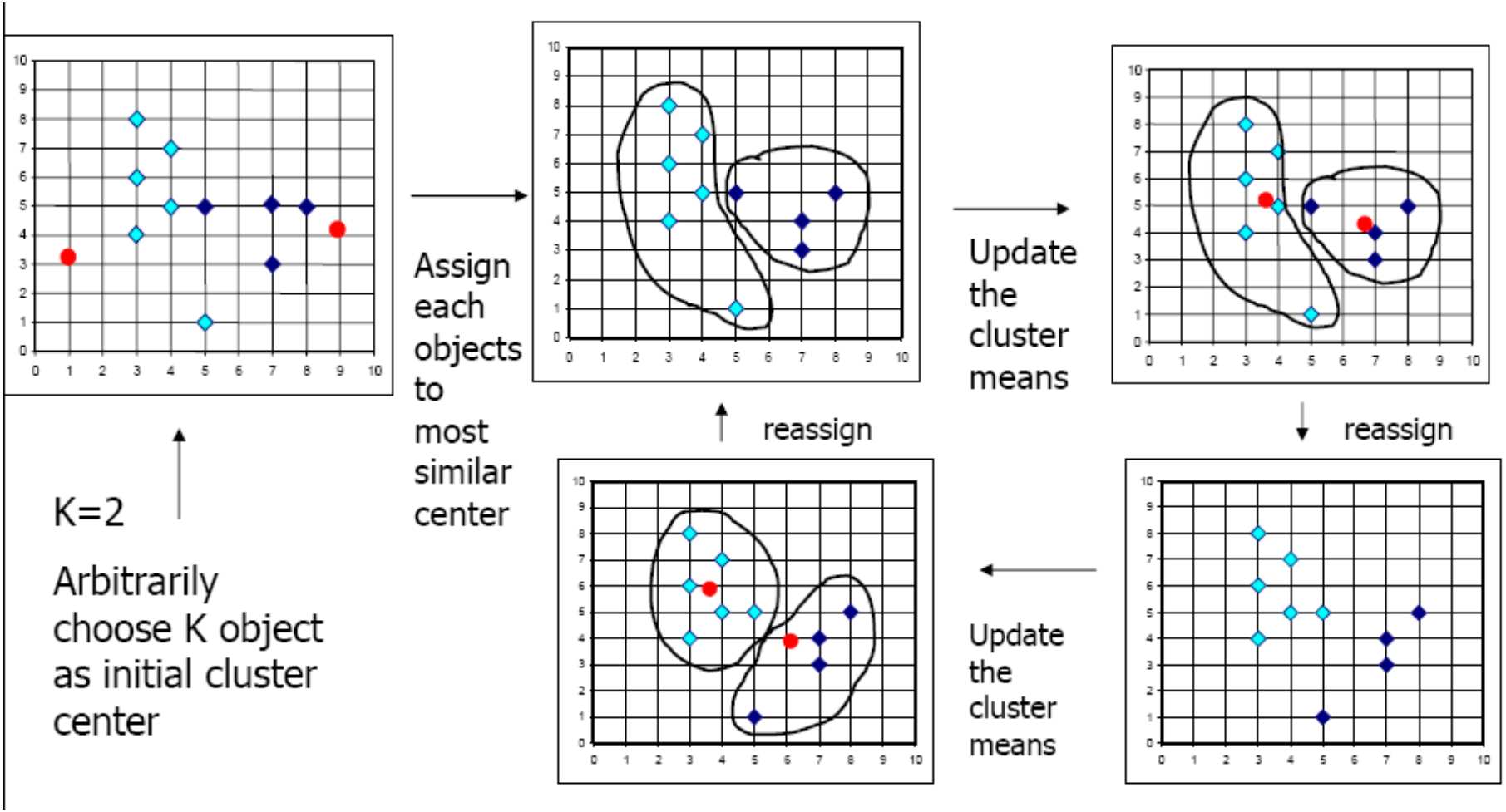
2.1 划分方法

- 启发式方法 $E = \Sigma_{i=1}^{k} \Sigma_{p \in C_i} (d(p, c_i))^2$
  - k-均值(k-means)
    - 每个簇用该簇中对象的均值来表示
    - 基于质心的技术
  - k-中心点(k-medoids)
    - 每个簇用接近簇中心的一个对象来表示
    - 基于代表对象的技术
- 适用性
  - 这些启发式算法适合发现中小规模数据库中的球状聚类
  - 对于大规模数据库和处理任意形状的聚类，这些算法需要进一步扩展

2.1.1 KMeans

1. 从数据集中随机取K个样本（也可以是随机点）作为初始的聚类中心$\mathrm{C} = \{c_1, c_2, \ldots, c_k\}$
2. 针对数据集中每个样本$\mathrm{x_i}$，计算它到K个聚类中心的距离并将其分到距离最小的聚类中心对应的类中
3. 针对每个类别c_i，重新计算其聚类中心$c_\mathrm{i} = \frac{1}{|c_\mathrm{i}|}\sum_{\mathrm{x}\in c_\mathrm{i}} \mathrm{x}$
4. 重复第2和第3步骤，直到$\sum_{\mathrm{i}=0}^{n} \min_{c_j \in C}(\| x_j - c_i \|^2)$小于特定的阈值

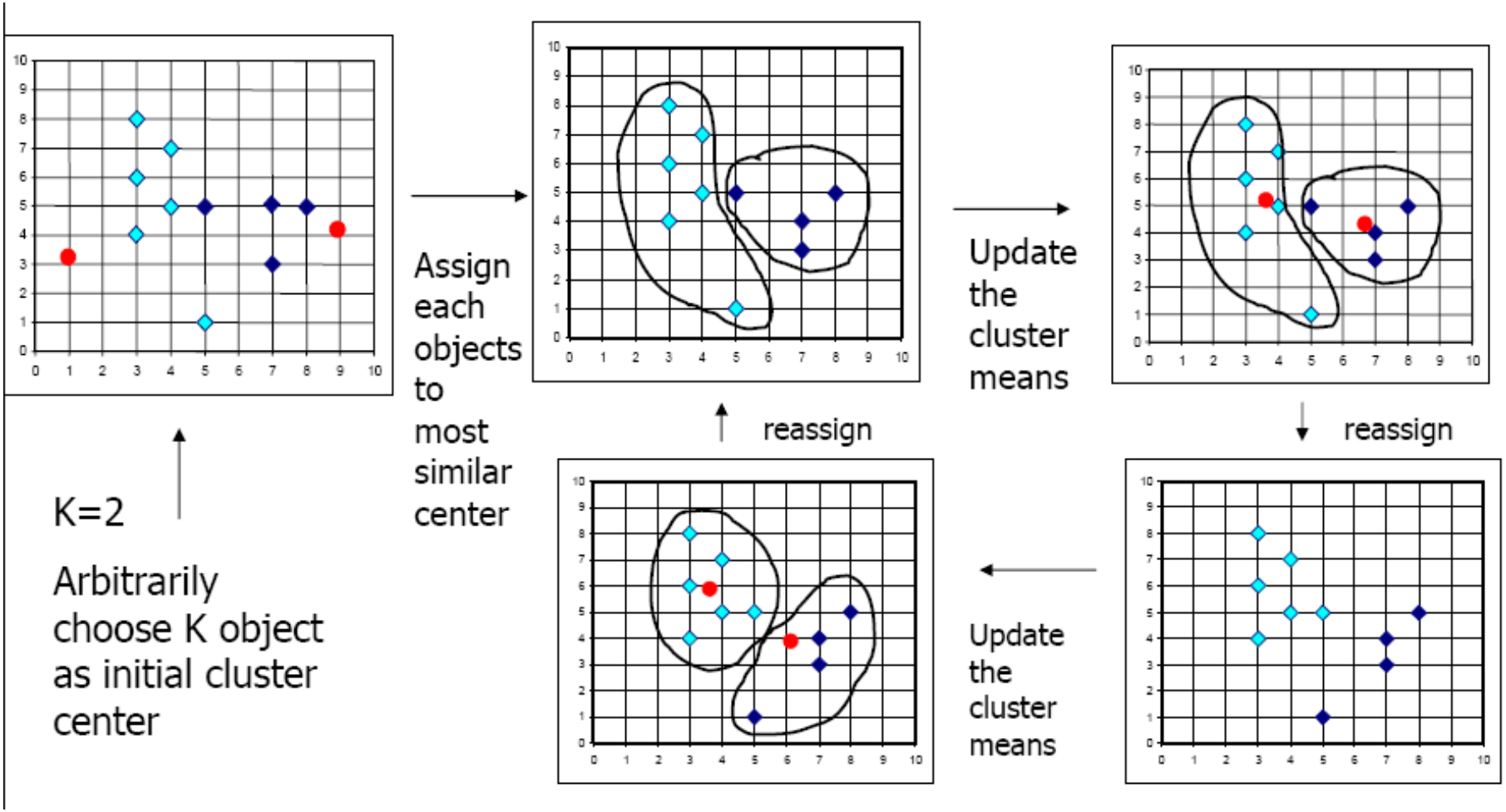
2.1.1 KMeans


K=2
Arbitrarily choose K object as initial cluster center
Assign each objects to most similar center
reassign
Update the cluster means
reassign
Update the cluster means

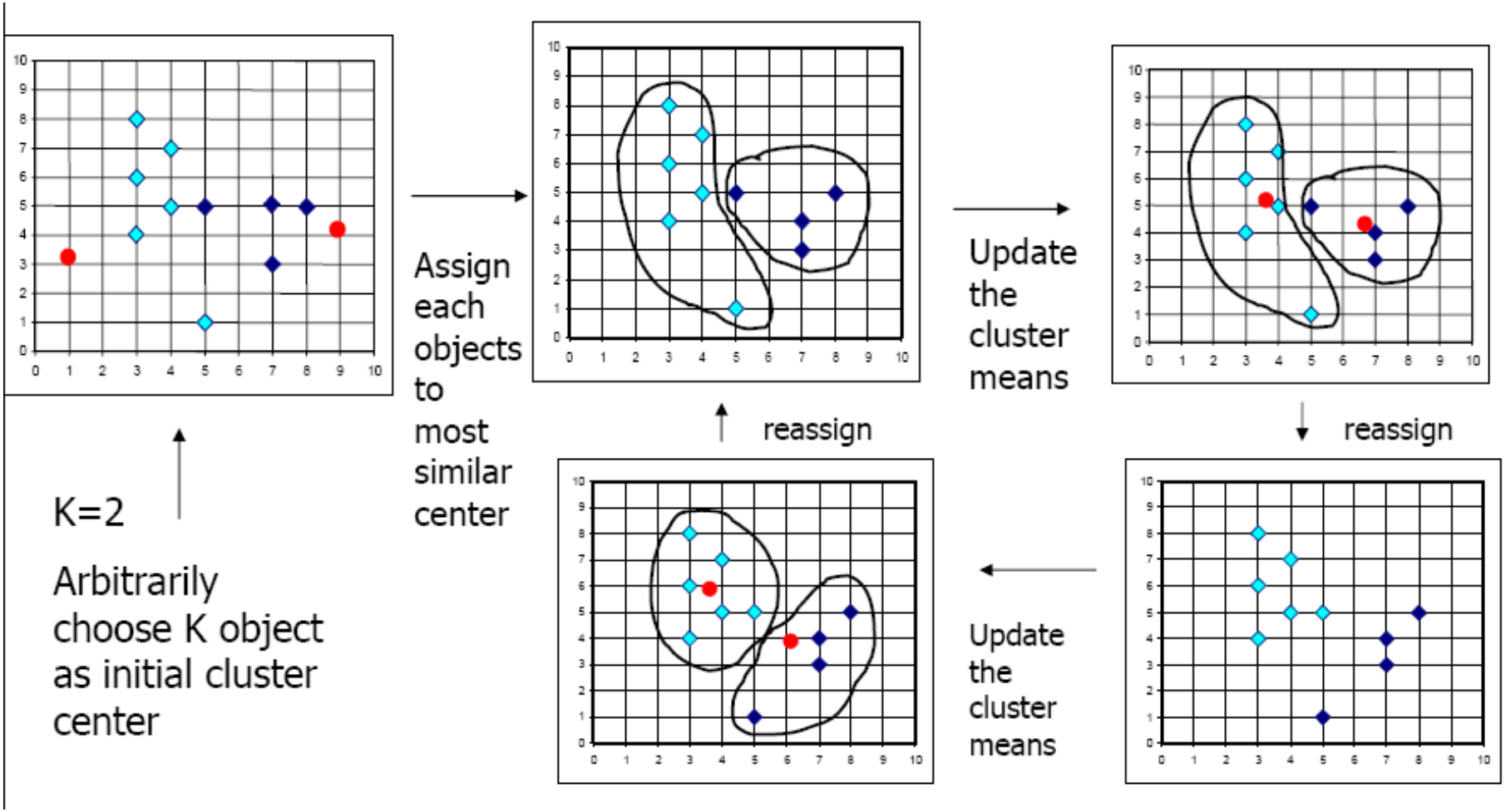
2.1.1 KMeans

1. 从数据集中随机取K个样本（也可以是随机点）作为初始的聚类中心$\mathrm{C} = \{c_1, c_2, \ldots, c_k\}$
2. 针对数据集中每个样本$\mathrm{x_i}$，计算它到K个聚类中心的距离并将其分到距离最小的聚类中心对应的类中
3. 针对每个类别c_i，重新计算其聚类中心$c_\mathrm{i} = \frac{1}{|c_\mathrm{i}|}\sum_{\mathrm{x}\in c_\mathrm{i}} \mathrm{x}$
4. 重复第2和第3步骤，直到$\sum_{\mathrm{i}=0}^{n} \min_{c_j \in C}(\| x_j - c_i \|^2)$小于特定的阈值

2.1.1 KMeans


K=2
Arbitrarily choose K object as initial cluster center
Assign each objects to most similar center
Update the cluster means
reassign
Update the cluster means
reassign

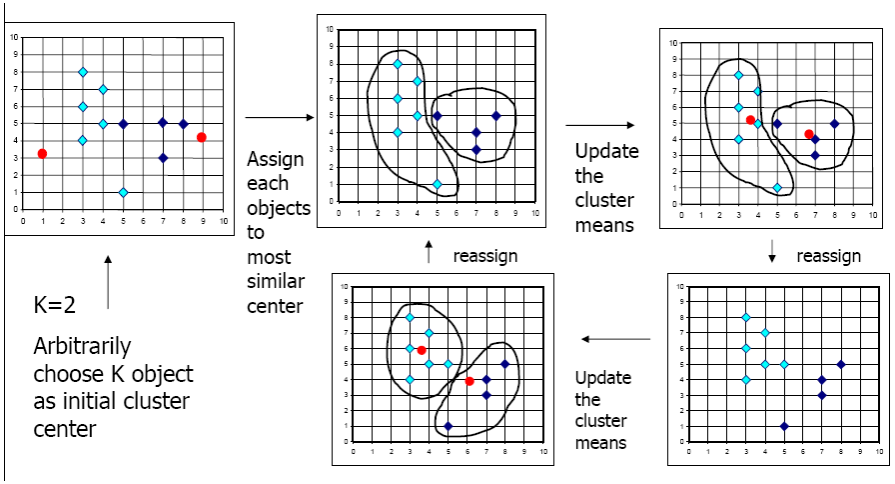
2.1.1 KMeans

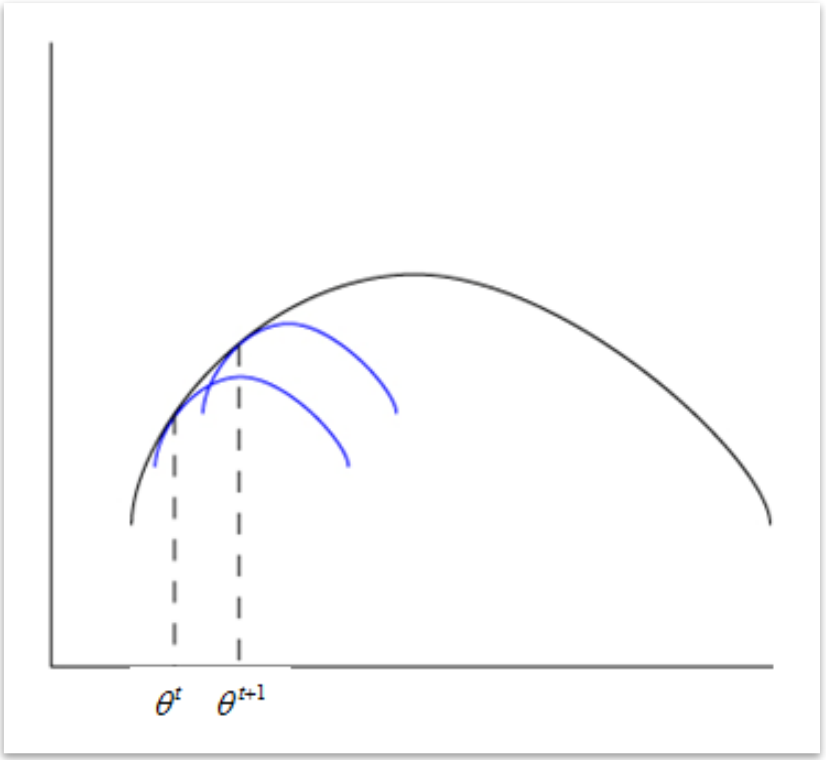


Kmeans算法为启发式算法，遵循的寻优原则：

每次聚类保证局部最优，随后调整聚类，利用局部最优聚类的上限来不断逼近全局最优

2.1.1 KMeans





Kmeans算法为启发式算法，遵循的寻优原则：

每次聚类保证局部最优，随后调整聚类，利用局部最优聚类的上限来不断逼近全局最优

2.1.1 k-means计算示例

- 假设：给定如下要进行聚类的对象：

  {2，4，10，12，3，20，30，11，25}，k = 2，请使用k均值划分聚类

- 步骤如下：

m1	m2	K1	K2
2	4	{2,3}	{4,10,12,20,30,11,25}
2.5	16	{2,3,4}	{10,12,20,30,11,25}
3	18	{2,3,4,10}	{12,20,30,11,25}
4.75	19.6	{2,3,4,10,11,12}	{20,30,25}
7	25	{2,3,4,10,11,12}	{20,30,25}

2.1.1 An Example of K-Means Clustering

【例】假定我们对A、B、C、D四个样品分别测量两个变量和得到结果见表。

样品	变量	
	X_1	X_2
A	5	3
B	-1	1
C	1	-2
D	-3	-2

样品测量结果

试将以上的样品聚成两类。

单选题 1分

设置

最后形成的2类为

A (A,B)(C,D)

B (A,B,C)(D)

C (A)(B,C,D)

样品	变量	
	X_1	X_2
A	5	3
B	-1	1
C	1	-2
D	-3	-2

第一步：按要求取K=2，为了实施均值法聚类，我们将这些样品随意分成两类，比如（A、B）和（C、D），然后 计算这两个聚类的中心坐标，见下表所示。

中心坐标是通过原始数据计算得来的。

聚类	中心坐标	
	$-X_1$	$-X_2$
（A、B）	2	2
（C、D）	-1	-2

提交

样品	变量	
	X_1	X_2
A	5	3
B	-1	1
C	1	-2
D	-3	-2

第一步：按要求取K=2，为了实施均值法聚类，我们将这些样品随意分成两类，比如（A、B）和（C、D），然后　计算这两个聚类的中心坐标，见下表所示。

中心坐标是通过原始数据计算得来的。

聚类	中心坐标	
	$-X_1$	$-X_2$
(A、B)	2	2
(C、D)	-1	-2

2.1.1 An Example of K-Means Clustering

第二步：计算某个样品到各类中心的欧氏平方距离，然后将该样品分配给最近的一类。对于样品有变动的类，重新计算它们的中心坐标，为下一步聚类做准备。先计算A到两个类的平方距离：

$$d^2(A,(AB)) = (5-2)^2 + (3-2)^2 = 10$$

$$d^2(A,(CD)) = (5+1)^2 + (3+2)^2 = 61$$

由于A到（A、B）的距离小于到（C、D）的距离，因此A不用重新分配。计算B到两类的平方距离：

$$d^2(B,(AB)) = (-1-2)^2 + (1-2)^2 = 10$$

$$d^2(B,(CD)) = (-1+1)^2 + (1+2)^2 = 9$$

2.1.1 An Example of K-Means Clustering

由于B到（A、B）的距离大于到（C、D）的距离，因此B要分配给（C、D）类，得到新的聚类是（A）和（B、C、D）。更新中心坐标如下表所示。

聚类	中心坐标	
	$-X_1$	$-X_2$
(A)	5	3
(B、C、D)	-1	-1

更新后的中心坐标

2.1.1 An Example of K-Means Clustering

第三步：再次检查每个样品，以决定是否需要重新分类。计算各样品到各中心的距离平方，结果见下表。

聚类	样品到中心的距离平方			
	A	B	C	D
(A)	0	40	41	89
(B、C、D)	52	4	5	5

到现在为止，每个样品都已经分配给距离中心最近的类，因此聚类过程到此结束。最终得到K=2的聚类结果是A独自成 一类，B、C、D聚成一类。

2.1.1 k-means算法效率分析

- 算法的计算复杂度为O(nkt)
- 其中

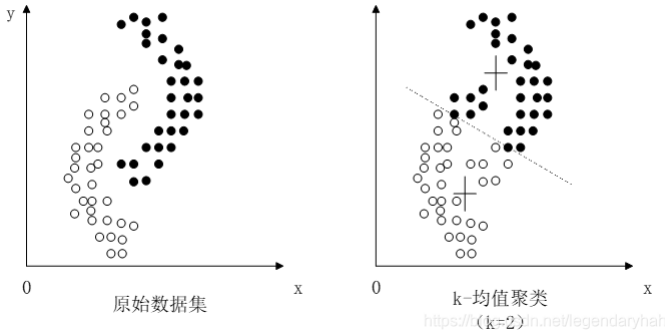
  n为数据集中对象的数目

  k为期望得到的簇的数目

  t为迭代的次数
- 在处理大数据库时也是相对有效的(可扩展性)

2.1.1 k-means优缺点

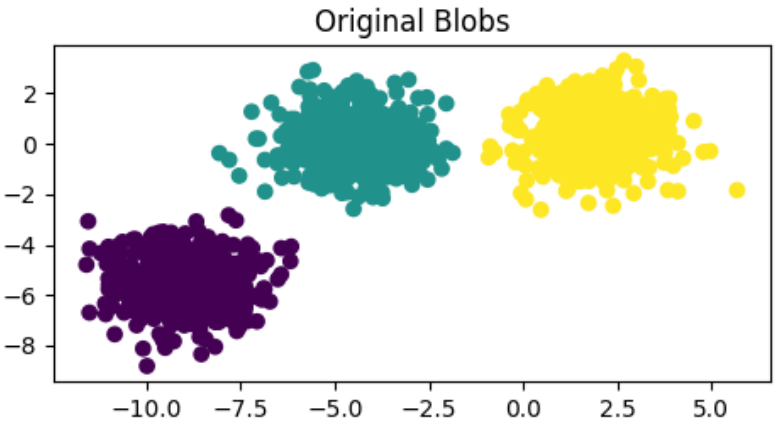
- 优点
  - 聚类时间快
  - 当结果簇是密集的，而簇与簇之间区别明显时，效果较好
  - 相对可扩展和有效，能对大数据集进行高效划分
- 缺点
  - 用户必须事先指定聚类簇的个数
  - 常常终止于局部最优
  - 只适用于数值属性聚类(计算均值有意义)
  - 对噪声和异常数据也很敏感
  - 不同的初始值，结果可能不同
  - 不适合发现非凸面形状的簇

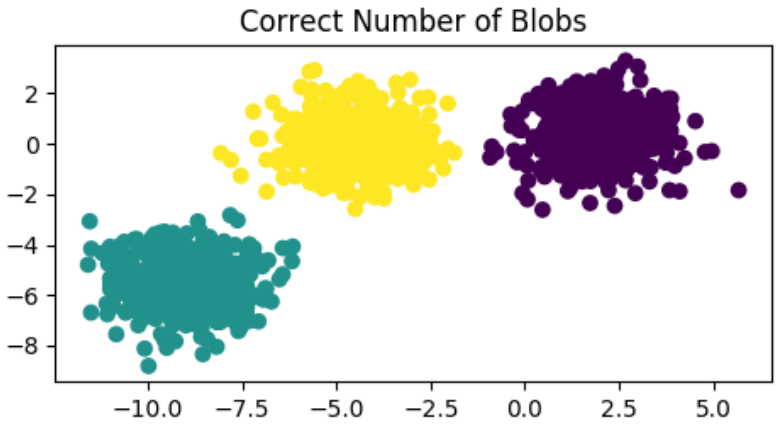


2.1.1 KMeans的问题

1. KMeans中初始簇规模估计正确

```
import numpy as np
import matplotlib.pyplot as plt

from sklearn.cluster import KMeans
from sklearn.datasets import make_blobs

plt.figure(figsize=(12, 12))

n_samples = 1500
random_state = 170
X, y = make_blobs(n_samples=n_samples, random_state=random_state)

plt.subplot(321)
plt.scatter(X[:, 0], X[:, 1], c=y)
plt.title("Original Blobs")


y_pred = KMeans(n_clusters=3, random_state=random_state).fit_predict(X)

plt.subplot(322)
plt.scatter(X[:, 0], X[:, 1], c=y_pred)
plt.title("Correct Number of Blobs")
```

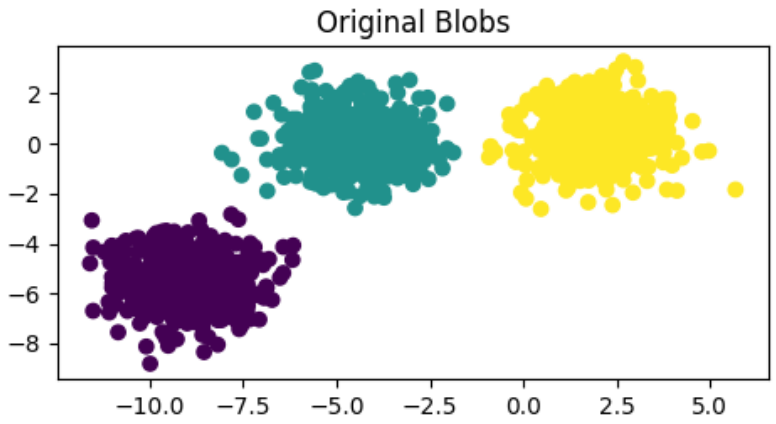


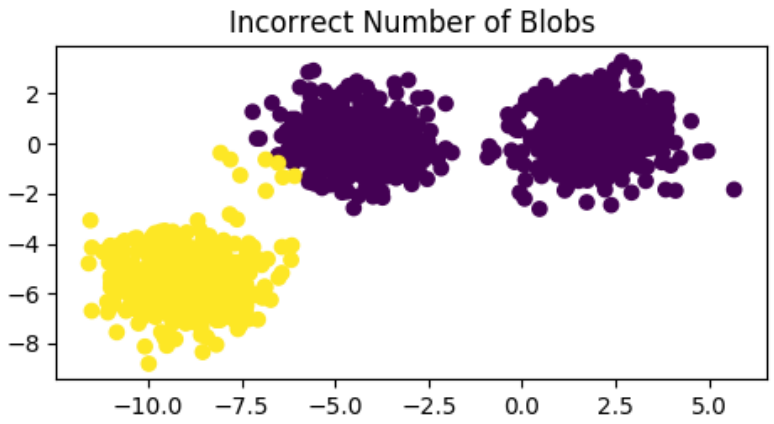


2.1.1 KMeans的问题

2. KMeans中初始簇规模估计错误

```
n_samples = 1500
random_state = 170
X, y = make_blobs(n_samples=n_samples, random_state=random_state)

plt.subplot(321)
plt.scatter(X[:, 0], X[:, 1], c=y)
plt.title("Original Blobs")

# Incorrect number of clusters
y_pred = KMeans(n_clusters=2, random_state=random_state).fit_predict(X)

plt.subplot(322)
plt.scatter(X[:, 0], X[:, 1], c=y_pred)
plt.title("Incorrect Number of Blobs")
```

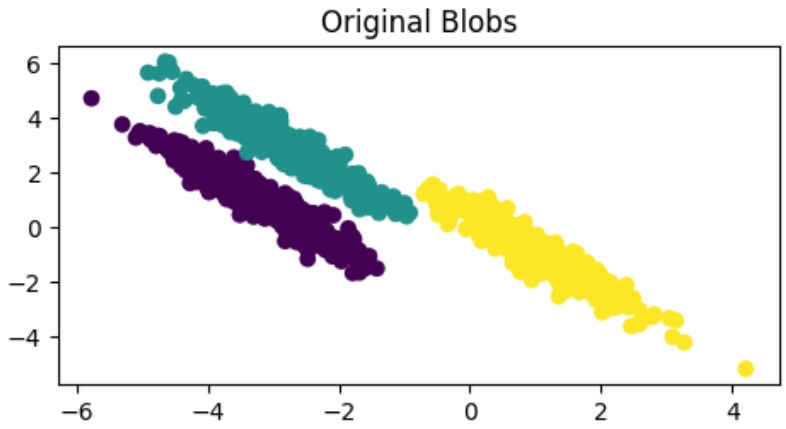


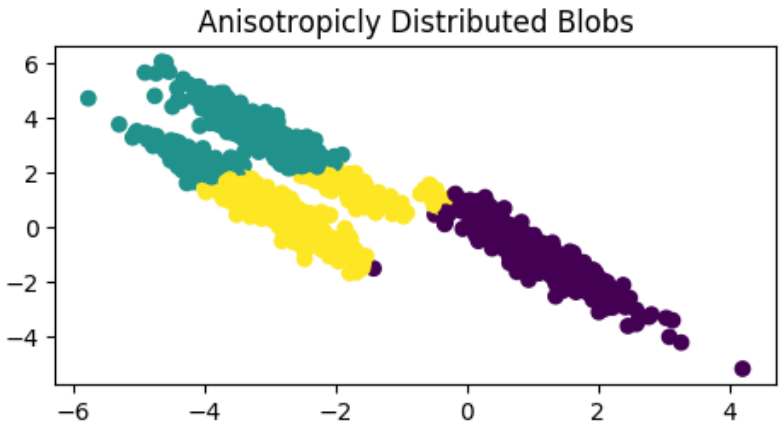


2.1.1 KMeans的问题

3. 数据分布形状对分簇结果的影响

```
X, y = make_blobs(n_samples=n_samples, random_state=random_state)
# Anisotropicly distributed data
transformation = [[0.60834549, -0.63667341], [-0.40887718, 0.85253229]]
X_aniso = np.dot(X, transformation)

plt.subplot(321)
plt.scatter(X_aniso[:, 0], X_aniso[:, 1], c=y)
plt.title("Original Blobs")

y_pred = KMeans(n_clusters=3, random_state=random_state).fit_predict(X_aniso)

plt.subplot(322)
plt.scatter(X_aniso[:, 0], X_aniso[:, 1], c=y_pred)
plt.title("Anisotropicly Distributed Blobs")
```

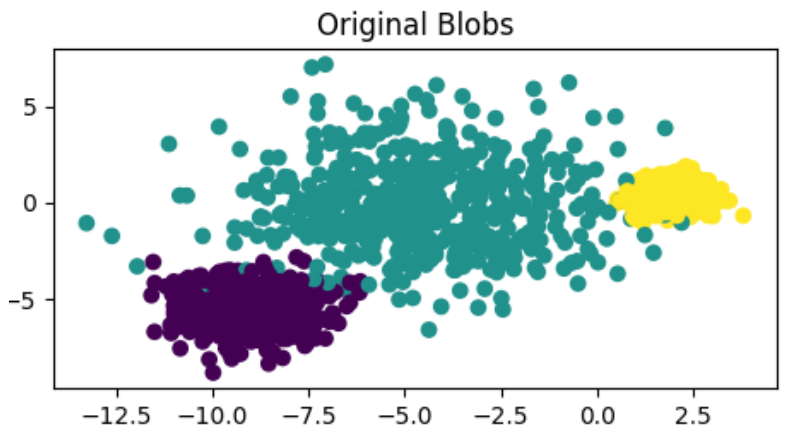


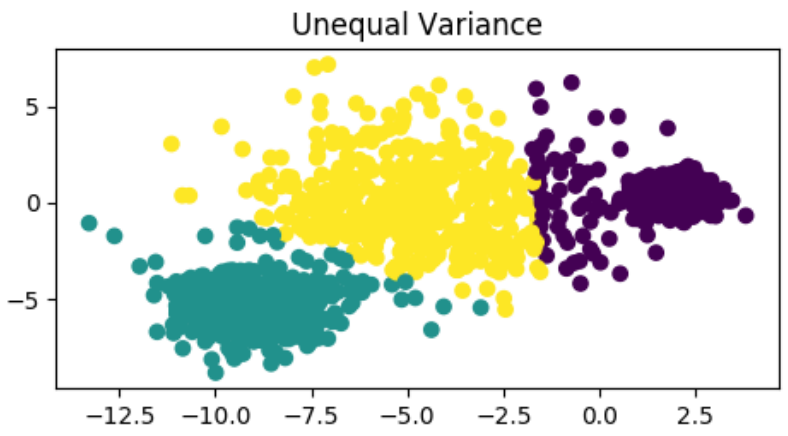


2.1.1 KMeans的问题

4. 数据分散程度对分簇结果的影响

```
# Different variance
X_varied, y_varied = make_blobs(n_samples=n_samples,
                                cluster_std=[1.0, 2.5, 0.5],
                                random_state=random_state)
plt.subplot(321)
plt.scatter(X_varied[:, 0], X_varied[:, 1], c=y_varied)
plt.title("Original Blobs")

y_pred = KMeans(n_clusters=3, random_state=random_state).fit_predict(X_varied)

plt.subplot(322)
plt.scatter(X_varied[:, 0], X_varied[:, 1], c=y_pred)
plt.title("Unequal Variance")
```





2.1.1 KMeans的问题

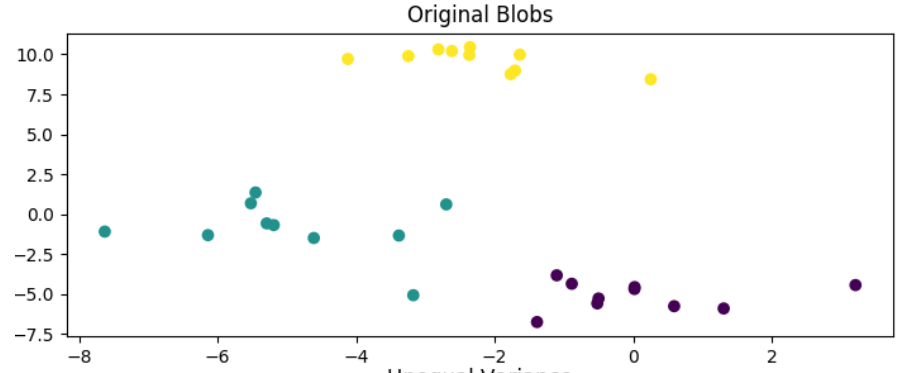
5. 随机初始种子的影响

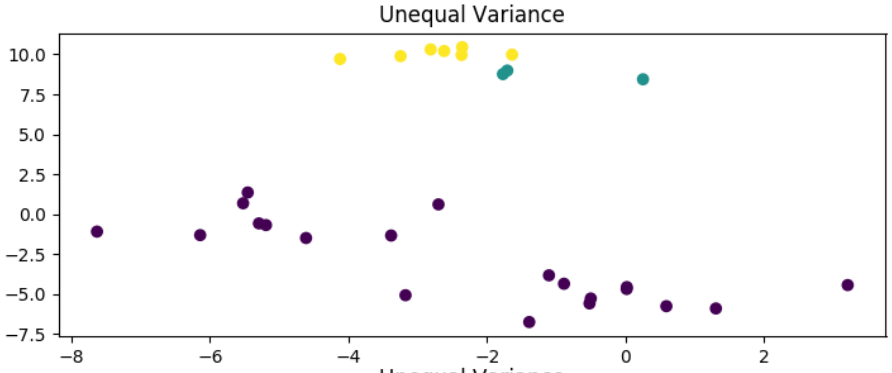
```
n_samples = 30
random_state = 50
X, y = make_blobs(n_samples=n_samples, random_state=random_state)

# Different variance
X_varied, y_varied = make_blobs(n_samples=n_samples,
                                cluster_std=[1, 2, 1],
                                random_state=random_state)
plt.subplot(321)
plt.scatter(X_varied[:, 0], X_varied[:, 1], c=y)
plt.title("Original Blobs")

y_pred = KMeans(init='random',n_clusters=3, n_init=1).fit_predict(X_varied)
plt.subplot(322)
plt.scatter(X_varied[:, 0], X_varied[:, 1], c=y_pred)
plt.title("Unequal Variance")


y_pred = KMeans(init='random',n_clusters=3, n_init=1).fit_predict(X_varied)
plt.subplot(323)
plt.scatter(X_varied[:, 0], X_varied[:, 1], c=y_pred)
plt.title("Unequal Variance")

y_pred = KMeans(init='random',n_clusters=3, n_init=1).fit_predict(X_varied)
plt.subplot(324)
plt.scatter(X_varied[:, 0], X_varied[:, 1], c=y_pred)
plt.title("Unequal Variance")
```

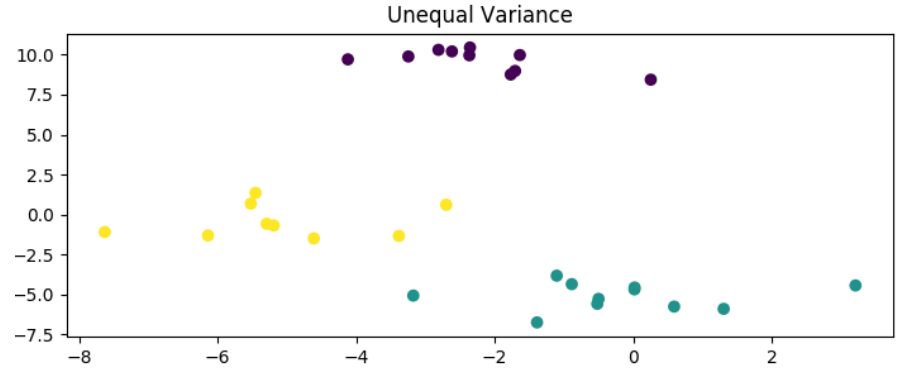
2.1.1 KMeans的问题

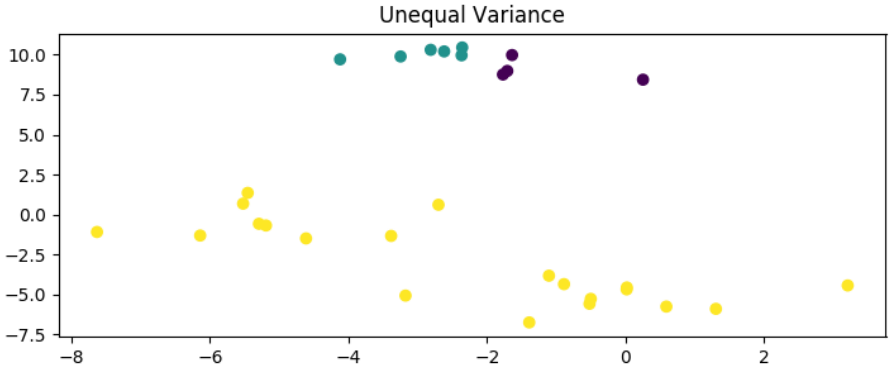
5. 随机初始种子的影响

```
n_samples = 30
random_state = 50
X, y = make_blobs(n_samples=n_samples, random_state=random_state)
```









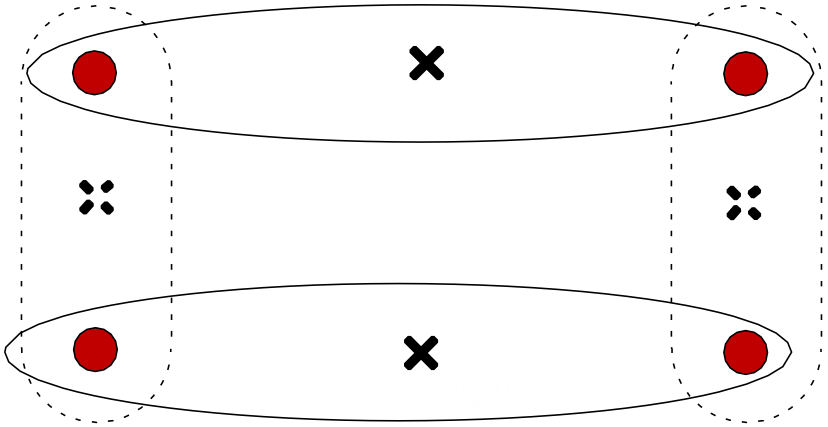

```
y_pred = KMeans(init='random',n_clusters=3, n_init=1).fit_predict(X_varied)
plt.subplot(324)
plt.scatter(X_varied[:, 0], X_varied[:, 1], c=y_pred)
plt.title("Unequal Variance")
```

2.1.1 KMeans优缺点

- 优点
  - 聚类时间快
  - 当结果簇是密集的，而簇与簇之间区别明显时，效果较好
  - 相对可扩展和有效，能对大数据集进行高效划分
- 缺点
  - 用户必须事先指定聚类簇的个数
  - 常常终止于局部最优
  - 只适用于数值属性聚类(计算均值有意义)
  - 对噪声和异常数据也很敏感
  - 不同的初始值，结果可能不同
  - 不适合发现非凸面形状的簇

2.1.2 k-modes算法 —— 解决数据敏感的问题

- Most of the variants of the k-means which differ in
  - Selection of the initial k means（初始k均值选择）
  - Dissimilarity calculations（相异度计算）
  - Strategies to calculate cluster means（计算均值方法）
- Handling categorical data: k-modes
  - Replacing means of clusters with modes
  - Using new dissimilarity measures to deal with categorical objects



2.1.3 KMeans++算法 —— 解决初始点选择问题

- 基本原理

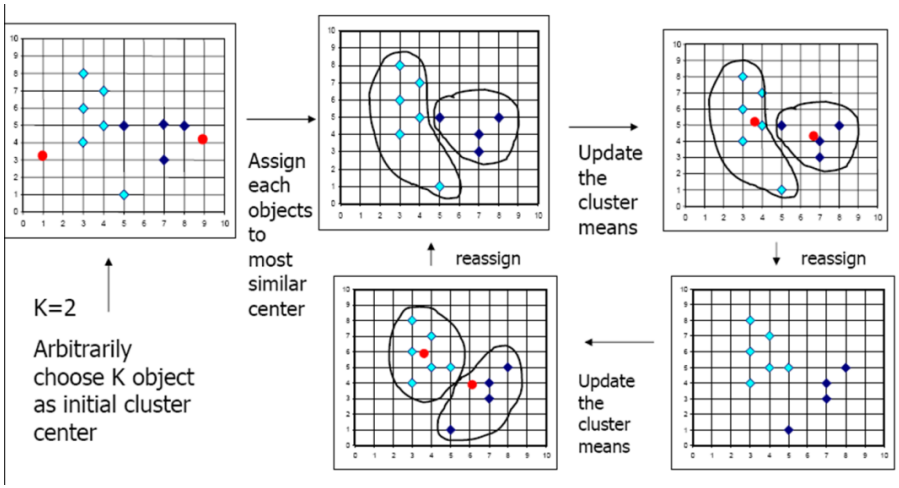
① 从输入的数据点集合中随机选择一个点作为第一个聚类中心

② 对于数据集中的每一个点X，计算其与聚类中心的距离D（X）

③ 选择一个D（X）最大的点作为新的聚类中心

④ 重复2和3步直到K个聚类中心被选出

⑤ 利用K个初始聚类中心运行K-Means



2.1.3 KMeans++算法 —— 解决初始点选择问题

- 基本原理

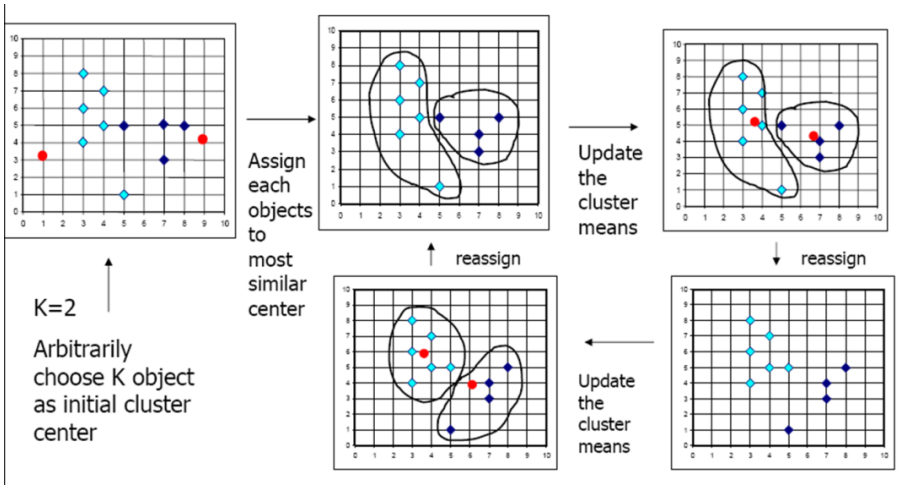
① 从输入的数据点集合中随机选择一个点作为第一个聚类中心

② 对于数据集中的每一个点X，计算其与聚类中心的距离D（X）

③ 选择一个D（X）最大的点作为新的聚类中心

④ 重复2和3步直到K个聚类中心被选出

⑤ 利用K个初始聚类中心运行K-Means



尽可能选择距离远的点来作为初始种子节点

2.1.4 k-中心点方法解决对离群点敏感问题

一维空间的7个点 1 2 3 8 9 10 25 离群点

2.1.4 k-中心点方法解决对离群点敏感问题

一维空间的7个点 1 2 3 8 9 10 25 离群点

设置k=2，直觉应该划分为 (1,2,3) (8,9,10,25) 两个类别

2.1.4 k-中心点方法解决对离群点敏感问题

一维空间的7个点 1 2 3 8 9 10 25 **离群点**

设置k=2，直觉应该划分为 (1,2,3) (8,9,10,25) 两个类别

划分方法聚类质量评价准则：最小化E值

$$E = \Sigma_{i=1}^{k} \Sigma_{p \in C_i} (d(p, c_i))^2$$

填空题 2分

设置

一维空间的7个点1 2 3 8 9 10 25，根据划分方法聚类质量评价准则，设置k为2时，(1,2,3)(8,9,10,25)聚类的E值为：[填空1]，(1,2,3,8)(9,10,25)的E值为：[填空2]

划分方法聚类质量评价准则：最小化E值

$$E = \Sigma_{i=1}^{k} \Sigma_{p \in C_i} (d(p, c_i))^2$$

提交

正常使用填空题需3.0以上版本雨课堂

单选题 1分

设置

一维空间的7个点1 2 3 8 9 10 25，根据划分方法聚类质量评价准则，设置k为2时，将划分为如下哪2个类别

A (1,2,3)(8,9,10,25)

B (1,2,3,8)(9,10,25)

划分方法聚类质量评价准则：最小化E值

$$E = \Sigma_{i=1}^{k} \Sigma_{p \in C_i} (d(p, c_i))^2$$

提交

2.1.4 k-中心点方法解决对离群点敏感问题

一维空间的7个点 1 2 3 8 9 10 25 **离群点**

设置k=2，直觉应该划分为 (1,2,3) (8,9,10,25) 两个类别

划分方法聚类质量评价准则：最小化E值

$$E = \Sigma_{i=1}^{k} \Sigma_{p \in C_i} (d(p, c_i))^2$$

(1,2,3) (8,9,10,25) 聚类的E值为：

$(1-2)^2+ (2-2)^2 + (3-2)^2 + (8-13)^2 + (9-13)^2 + (10-13)^2 + (25-13)^2 =196$

(1,2,3,8) (9,10,25) 的E值为：

$(1-3.5)^2+ (2-3.5)^2 + (3-3.5)^2 + (8-3.5)^2 + (9-14.67)^2 + (10-14.67)^2 + (25-14.67)^2 =189.67$

2.1.4 k-中心点方法解决对离群点敏感问题

一维空间的7个点 1 2 3 8 9 10 25 **离群点**

设置k=2，直觉应该划分为 (1,2,3) (8,9,10,25) 两个类别

划分方法聚类质量评价准则：最小化E值

$$E = \Sigma_{i=1}^{k} \Sigma_{p \in C_i} (d(p, c_i))^2$$

(1,2,3) (8,9,10,25) 聚类的E值为：

$(1-2)^2+ (2-2)^2 + (3-2)^2 + (8-13)^2 + (9-13)^2 + (10-13)^2 + (25-13)^2 =196$

(1,2,3,8) (9,10,25) 的E值为：

$(1-3.5)^2+ (2-3.5)^2 + (3-3.5)^2 + (8-3.5)^2 + (9-14.67)^2 + (10-14.67)^2 + (25-14.67)^2 =189.67$

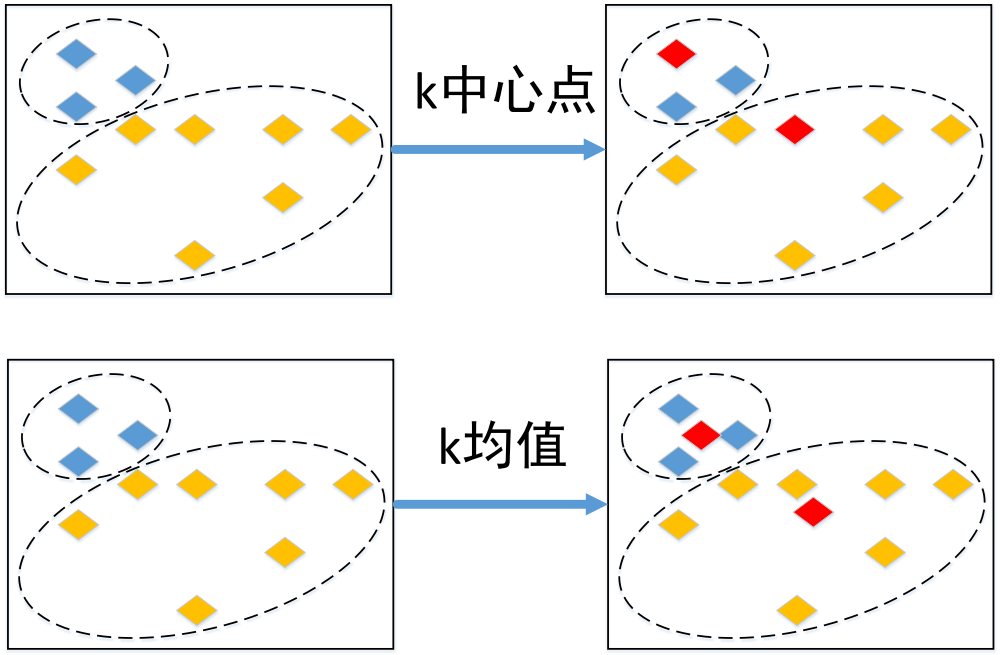
2.1.4 k-中心点方法

- 基本思想
  - 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
  - 然后迭代地用非代表对象来替代代表对象，以改进聚类的质量（**找更好的代表对象**）
  - 聚类结果的质量用一个**代价函数**来估算，该函数评估了对象与其参照对象之间的平均相异度

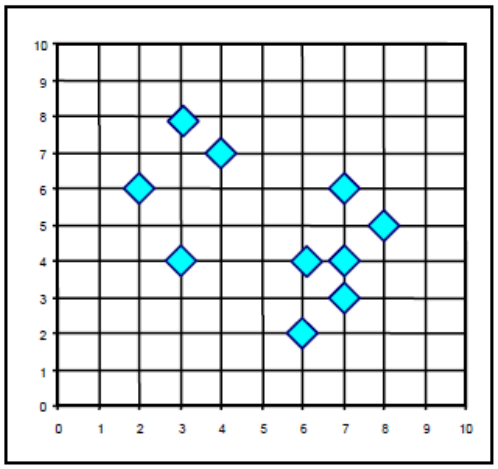
2.1.4 k-中心点

- k-中心点
  - 选用簇中位置**最中心的实际对象**即中心点作为参照点
  - 基于最小化所有对象与其参照点之间的相异度之和的原则来划分(使用绝对误差标准)

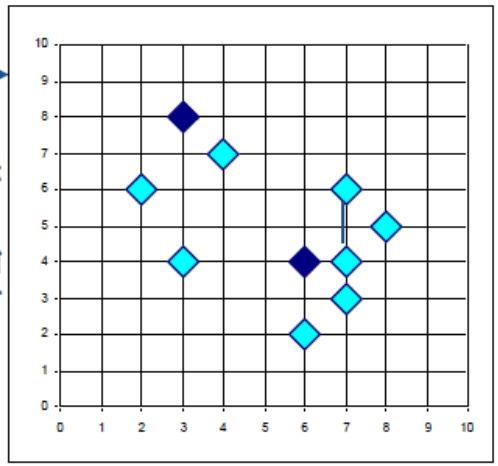
$$E = \sum_{i=1}^{k} \sum_{p \in C_i} | p - o_i |$$



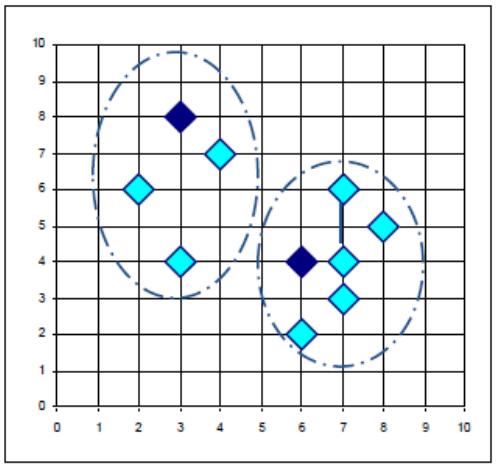
2.1.4 PAM算法示意



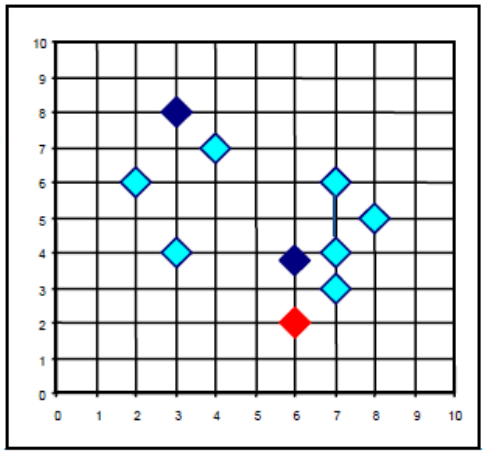
随机选择k个对象作为初始的代表对象



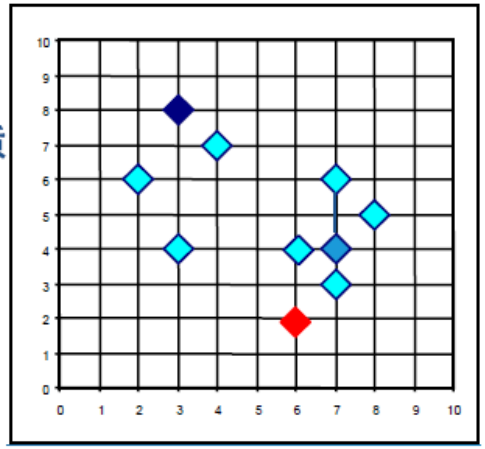
指派每个剩余的对象给离它最近的中心点



随机地选择一个非代表对象O_{random}



计算交换的总代价 S



如果能提高质量，则交换 O_{random} 和 O_j

K=2

循环

直到不发生变化

K中心点例子

对下列表中（左图）的10个数据聚类， k=2。可以看到这里每个数据的维度都为2。

X_1	2	6
X_2	3	4
X_3	3	8
X_4	4	7
X_5	6	2
X_6	6	4
X_7	7	3
X_8	7	4
X_9	8	5
X_{10}	7	6

Data object		Distance to	
i	X_i	$c_1 = (3,4)$	$c_2 = (7,4)$
1	(2, 6)	**3**	7
2	(3, 4)	**0**	4
3	(3, 8)	**4**	8
4	(4, 7)	**4**	6
5	(6, 2)	5	**3**
6	(6, 4)	3	**1**
7	(7, 3)	5	**1**
8	(7, 4)	4	**0**
9	(8, 5)	6	**2**
10	(7, 6)	6	**2**
Cost		**11**	**9**

随机挑选k=2个中心点：c1=（3，4），c2=（7，4）.那么将所有点到这两点的距离计算出来（右图），可以看到黑体为到两个中心点距离较小的距离值。那么根据右图，我们可以对所有数据点进行归类： Cluster1 = {(3,4)(2,6)(3,8)(4,7)}
Cluster2 = {(7,4)(6,2)(6,4)(7,3)(8,5)(7,6)} 很容易算出此时的总代价cost为：20

$$\underbrace{3+0+4+4}_{\text{objects in cluster 1}} + \underbrace{3+1+1+0+2+2}_{\text{objects in cluster 2}} = 20$$

挑选一个非中心点O’，让我们假定挑选的为X_7 ，即O‘=（7，3）。那么此时这两个中心点暂时变成了c1(3,4) and O’(7,3)，那么我们要计算一下这一替换措施所带来的损失cost：

i	c_1		Data objects (X_i)		Cost (distance)
1	3	4	2	6	**3**
3	3	4	3	8	**4**
4	3	4	4	7	**4**
5	3	4	6	2	5
6	3	4	6	4	3
8	3	4	7	4	4
9	3	4	8	5	6
10	3	4	7	6	6

i	O′		Data objects (X_i)		Cost (distance)
1	7	3	2	6	8
3	7	3	3	8	9
4	7	3	4	7	7
5	7	3	6	2	**2**
6	7	3	6	4	**2**
8	7	3	7	4	**1**
9	7	3	8	5	**3**
10	7	3	7	6	**3**

如上图所见，此时的cost（很好计算，黑体数值的和）变成了：

total cost = 3+4+4+2+2+1+3+3 = 22

此时的cost为22，比之前的cost=20要大，所以这次替换的代价变大，最终不进行这次替换。

多选题 1分

设置

下列哪些属于基于划分的聚类算法

A K-means

B K-modes

C K-means++

D K中心点

提交

多选题 1分

设置

下列说法正确的是

A K-means算法能够解决有离群点的聚类问题

B K-modes能够解决离散数据的聚类问题

C K-means++能够解决初始点影响聚类效果的问题

D K中心点能够解决有离群点的聚类问题

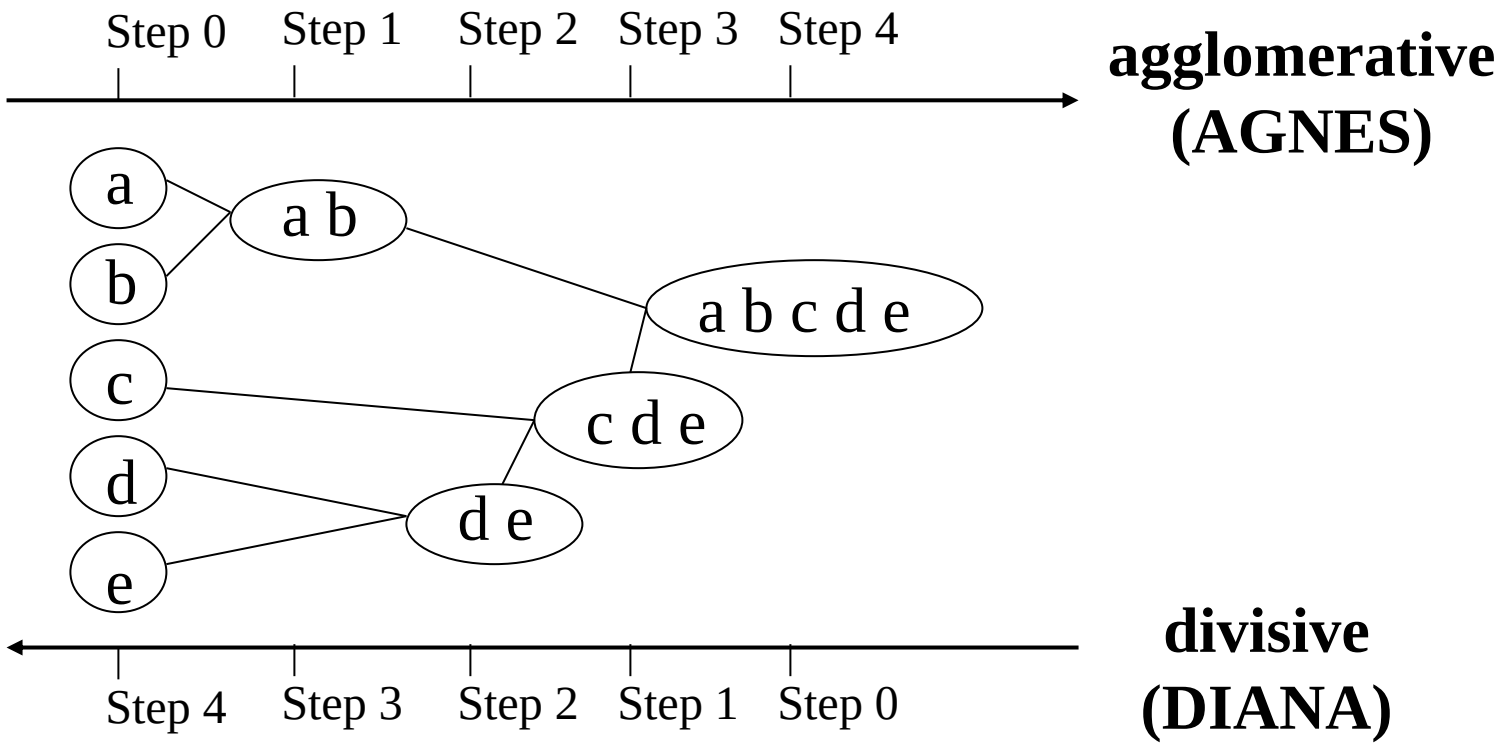
提交

2.2 层次方法

- 层次方法
  - 对给定数据对象集进行层次的分解
  - 使用距离矩阵作为聚类标准
  - 不需要输入聚类数目k，但需要终止条件
- 两种层次方法
  - **自底向上方法（凝聚）**
    - 初始将每个对象作为单独的一个簇，然后相继的合并相近的对象或簇，直到所有的簇合并为一个，或者达到一个终止条件
    - 代表算法：**AGNES算法**
  - **自顶向下方法（分裂）**
    - 初始将所有的对象置于一个簇中，在迭代的每一步，一个簇被分裂为多个更小的簇，直到最终每个对象在一个单独的簇中，或达到一个终止条件
    - 代表算法：**DIANA算法**

2.2 层次聚类过程图示

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



2.2.1 AGNES算法

- AGNES (Agglomerative Nesting)算法
  - 首先，将数据集中的每个样本作为一个簇；
  - 然后，根据某些准则将这些簇逐步合并；
  - 合并的过程反复进行，直至不能再合并或者达到结束条件为止。
- 合并准则
  - 每次找到距离最近的两个簇进行合并。
  - 两个簇之间的距离由这两个簇中距离最近的样本点之间的距离来表示。

2.2.1 AGNES算法

AGNES算法（自底向上合并算法）

输入：包含n个样本的数据集，终止条件簇的数目k。

输出：k个簇，达到终止条件规定的簇的数目。

(1) 初始时，将每个样本当成一个簇；

(2) REPEAT

根据不同簇中最近样本间的距离找到最近的两个簇；

合并这两个簇，生成新的簇的集合；

(3) UNTIL 达到定义的簇的数目。**算法终止条件是什么?**

2.2.1 AGNES算法

AGNES算法（自底向上合并算法）

输入：包含n个样本的数据集，终止条件簇的数目k。

输出：k个簇，达到终止条件规定的簇的数目。

(1) 初始时，将每个样本当成一个簇；

(2) REPEAT

根据不同簇中最近样本间的距离找到最近的两个簇；

合并这两个簇，生成新的簇的集合；

(3) UNTIL 达到定义的簇的数目。 **算法终止条件是什么？**

（1）指定簇的数目k

2.2.1 AGNES算法

AGNES算法（自底向上合并算法）

输入：包含n个样本的数据集，终止条件簇的数目k。

输出：k个簇，达到终止条件规定的簇的数目。

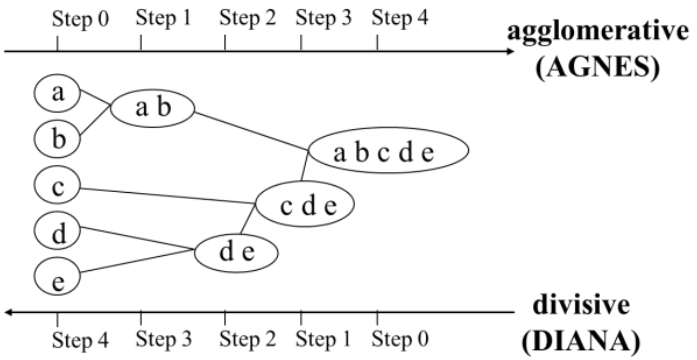
(1) 初始时，将每个样本当成一个簇；

(2) REPEAT

根据不同簇中最近样本间的距离找到最近的两个簇；

合并这两个簇，生成新的簇的集合；

(3) UNTIL 达到定义的簇的数目。 **算法终止条件是什么？**



（1）指定簇的数目k

（2）簇之间的距离超过一定阈值

2.2.1 AGNES算法

AGNES算法（自底向上合并算法）

输入：包含n个样本的数据集，终止条件簇的数目k。

输出：k个簇，达到终止条件规定的簇的数目。

(1) 初始时，将每个样本当成一个簇；

(2) REPEAT

根据不同簇中最近样本间的距离找到最近的两个簇；

合并这两个簇，生成新的簇的集合；

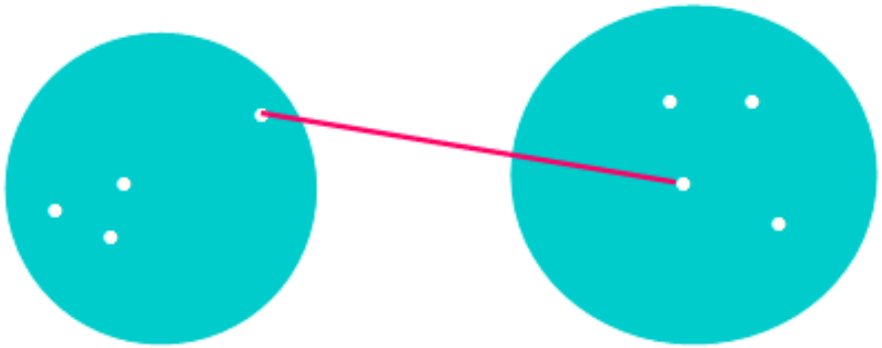
(3) UNTIL 达到定义的簇的数目。

簇之间的距离如何计算?

2.2.1 AGNES算法 —— 最小距离

- **单连接（single-link）方法**
  - 其每个簇可以用簇中所有对象代表，簇间的相似度用属于不同簇中**最近的**数据点对之间的相似度来度量
  - 也称为**最短距离法**，定义簇的邻近度为取自不同簇的所有点对的俩个最近的点之间的邻近度
  - 设d_{ij}表示样本$X_{(i)}$和$X_{(j)}$之间的距离，D_{ij}表示类G_i和G_j之间距离

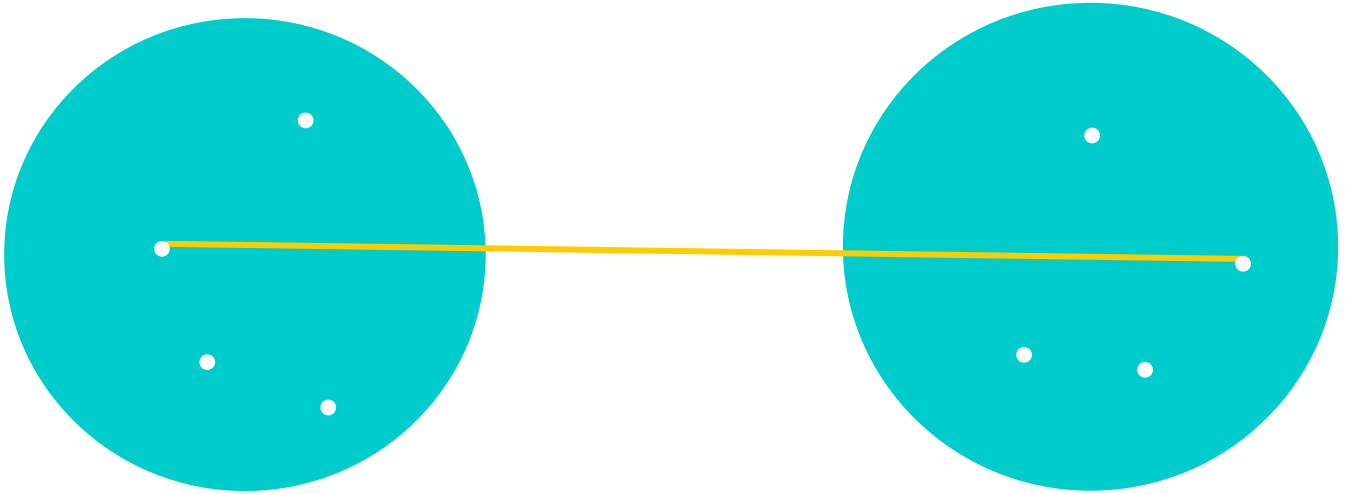
$$D_{ij} = \min_{X_{(i)} \in D_i, X_{(j)} \in D_j} d_{ij}$$



2.2.1 AGNES算法 —— 最大距离

- **全连接**
  - 取自不同簇中的俩个最远的点之间邻近度作为簇的邻近度，或者使用图的术语，不同的结点子集中俩个结点之间的最长边

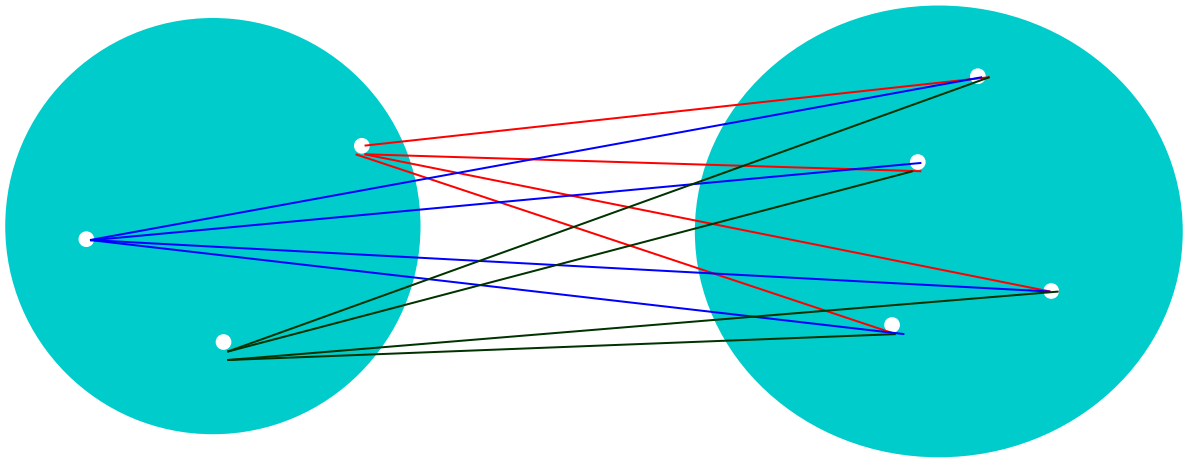
$$D_{ij} = \max_{X_{(i)} \in G_i, X_{(j)} \in G_j} d_{ij}$$



2.2.1 AGNES算法 —— 平均距离

- **组平均（average linkage method）**
  - 类间所有样本点的平均距离
  - 该法利用了所有样本的信息，被认为是较好的系统聚类法

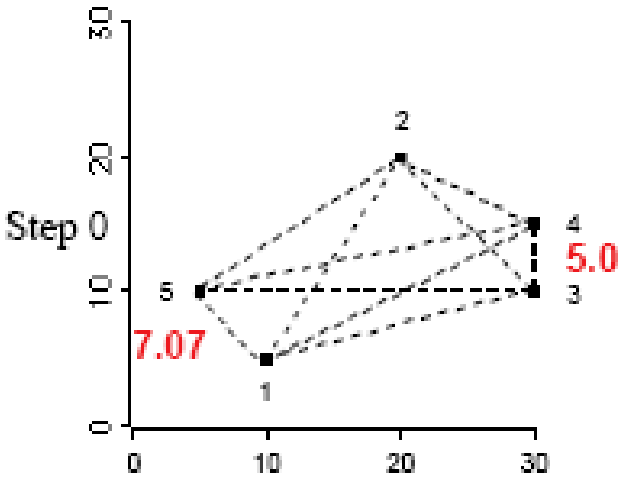
$$d_{avg}(C_i, C_j) = \frac{1}{n_i n_j} \sum_{p \in C_i} \sum_{p' \in C_j} \|p - p'\|$$



2.2.1 单连接算法

	x1	x2
1	10	5
2	20	20
3	30	10
4	30	15
5	5	10

	1	2	3	4	5
1	0.00				
2	18.0	0.00			
3	20.6	14.1	0.00		
4	22.4	11.2	6.00	0.00	
5	7.07	18.0	25.0	25.5	0.00

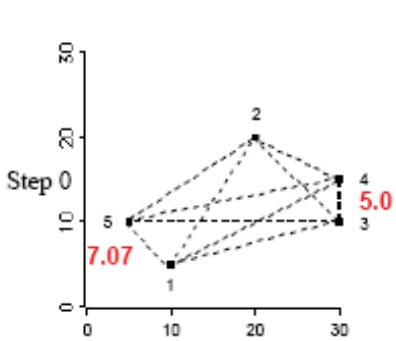


2.2.1 单连接算法

- 先将五个样本都分别看成是一个簇，最靠近的两个簇是3和4，因为他们具有最小的簇间距离 D（3，4）=5.0

	x1	x2
1	10	5
2	20	20
3	30	10
4	30	15
5	5	10

	1	2	3	4	5
1	0.00				
2	18.0	0.00			
3	20.6	14.1	0.00		
4	22.4	11.2	5.00	0.00	
5	7.07	18.0	25.0	25.5	0.00

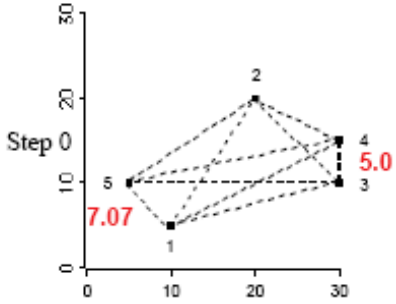


2.2.1 单连接算法

- 先将五个样本都分别看成是一个簇，最靠近的两个簇是3和4，因为他们具有最小的簇间距离 D（3，4）=5.0
- 合并簇3和4，得到新簇集合1,2,（34）,5更新距离矩阵
  - D(1,(34)) = min(D(1,3),D(1,4)) = min(20.6, 22.4) = 20.6
  - D(2,(34)) = min(D(2,3),D(2,4)) = min(14.1, 11.2) = 11.2
  - D(5,(34)) = min(D(3,5),D(4,5)) = min(25.0, 25.5) = 25.0

	x1	x2
1	10	5
2	20	20
3	30	10
4	30	15
5	5	10

	1	2	3	4	5
1	0.00				
2	18.0	0.00			
3	20.6	14.1	0.00		
4	22.4	11.2	0.00	0.00	
5	7.07	18.0	25.0	25.5	0.00



	1	2	5	(34)
1	0.00			
2	18.0	0.00		
5	7.07	18.0	0.00	
(34)	20.6	11.2	25.0	0.00

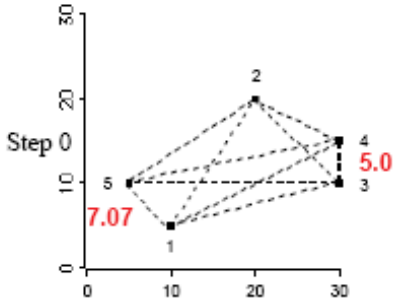
	2	(34)	(15)
2	0.00		
(34)	11.2	0.00	
(15)	18.0	20.6	0.00

2.2.1 单连接算法

- 先将五个样本都分别看成是一个簇，最靠近的两个簇是3和4，因为他们具有最小的簇间距离 D（3，4）=5.0
- 合并簇3和4，得到新簇集合1,2,（34）,5更新距离矩阵
  - D(1,(34)) = min(D(1,3),D(1,4)) = min(20.6, 22.4) = 20.6
  - D(2,(34)) = min(D(2,3),D(2,4)) = min(14.1, 11.2) = 11.2
  - D(5,(34)) = min(D(3,5),D(4,5)) = min(25.0, 25.5) = 25.0
  - 原有簇1,2,5间的距离不变，故在四个簇1,2,(34),5中，最靠近的两个簇是1和5，它们具有最小簇间距离D（1,5） = 7.07

	x1	x2
1	10	5
2	20	20
3	30	10
4	30	15
5	5	10

	1	2	3	4	5
1	0.00				
2	18.0	0.00			
3	20.6	14.1	0.00		
4	22.4	11.2	6.00	0.00	
5	7.07	18.0	25.0	25.5	0.00



	1	2	5	(34)
1	0.00			
2	18.0	0.00		
5	7.07	18.0	0.00	
(34)	20.6	11.2	25.0	0.00

	2	(34)	(15)
2	0.00		
(34)	11.2	0.00	
(15)	18.0	20.6	0.00

2.2.1 单连接算法

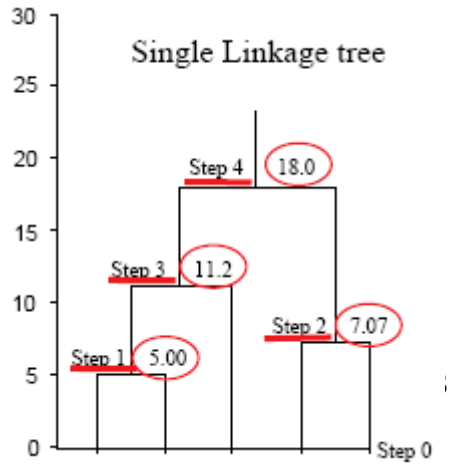
- 先将五个样本都分别看成是一个簇，最靠近的两个簇是3和4，因为他们具有最小的簇间距离 D（3，4）=5.0
- 合并簇3和4，得到新簇集合1,2,（34）,5更新距离矩阵
  - D(1,(34)) = min(D(1,3),D(1,4)) = min(20.6, 22.4) = 20.6
  - D(2,(34)) = min(D(2,3),D(2,4)) = min(14.1, 11.2) = 11.2
  - D(5,(34)) = min(D(3,5),D(4,5)) = min(25.0, 25.5) = 25.0
  - 原有簇1,2,5间的距离不变，故在四个簇1,2,(34),5中，最靠近的两个簇是1和5，它们具有最小簇间距离D（1,5） = 7.07

	x1	x2
1	10	5
2	20	20
3	30	10
4	30	15
5	5	10

	1	2	3	4	5
1	0.00				
2	18.0	0.00			
3	20.6	14.1	0.00		
4	22.4	11.2	0.00	0.00	
5	7.07	18.0	25.0	25.5	0.00

	1	2	5	(34)
1	0.00			
2	18.0	0.00		
5	7.07	18.0	0.00	
(34)	20.6	11.2	25.0	0.00

	2	(34)	(15)
2	0.00		
(34)	11.2	0.00	
(15)	18.0	20.6	0.00



2.2.2 层次方法的问题及改进

- 层次聚类的优缺点
  - 合并或分裂的决定需要检查和估算大量的对象或簇（计算复杂度高）
  - 一个步骤一旦完成便不能被撤消（无法扩展）
    - 避免考虑选择不同的组合，减少计算代价
    - 不能更正错误的决定
- 改进方法：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：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
  - CURE：采用固定数目的代表对象来表示每个簇，然后依据一个指定的收缩因子向着聚类中心对它们进行收缩
  - CHAMELEON：使用动态模型进行层次聚类

多选题 1分

设置

层次聚类法计算2个簇之间的距离有哪些方法

A 最小值

B 最大值

C 平均值

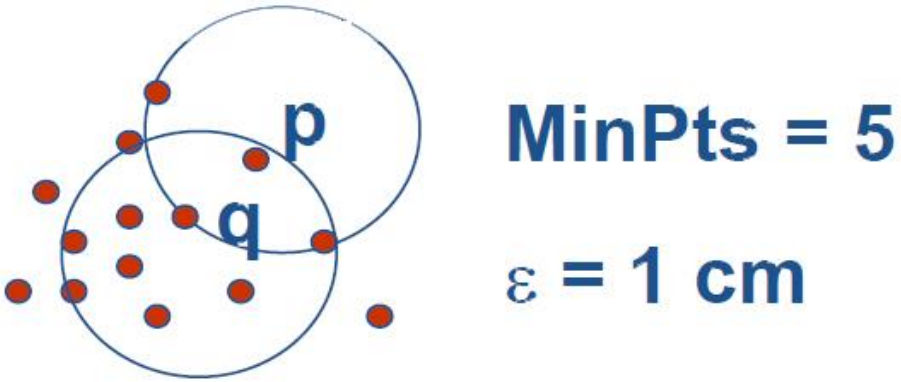
提交

2.3 基于密度的方法

- 基于密度聚类
  - 根据密度条件对邻近对象分组形成簇，簇的增长或者根据邻域密度，或者根据特定的密度函数(只要临近区域的密度超过某个阈值，就继续聚类)
- 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

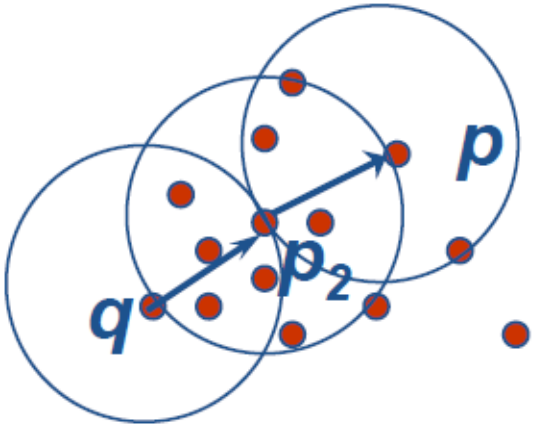
2.3.1 密度概念

- ε-邻域
  - 给定对象半径ε内的邻域称为该对象的ε-邻域
- 核心对象
  - 如果对象的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
- 直接密度可达
  - 给定对象集合D，如果p 是在q 的ε-邻域内，而q 是核心对象，则称对象p 是从对象q 关于ε和MinPts直接密度可达的

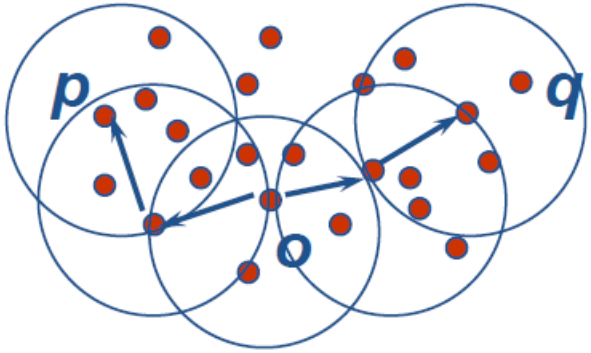


2.3.1 密度概念

- 密度可达
  - 如果存在一个对象链$p_1, \ldots, p_n$，$p_1 = q$，$p_n = p$，使得p_{i+1}是从p_i直接密度可达的，则称对象p是从对象q关于ε和MinPts(间接)密度可达的

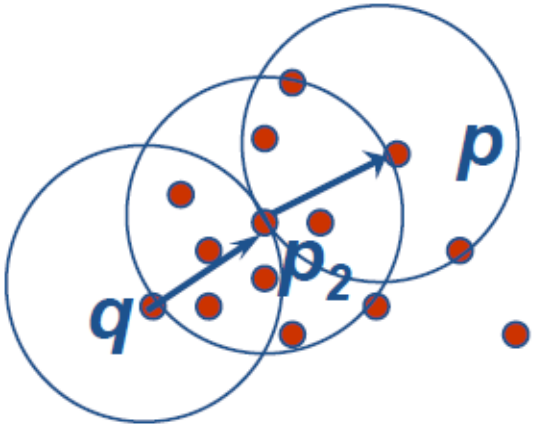


- 密度相连的
  - 如果存在对象o 使得，p 和q 都是从o关于ε和MinPts密度可达的，则称对象p与q关于ε和MinPts是密度相连的

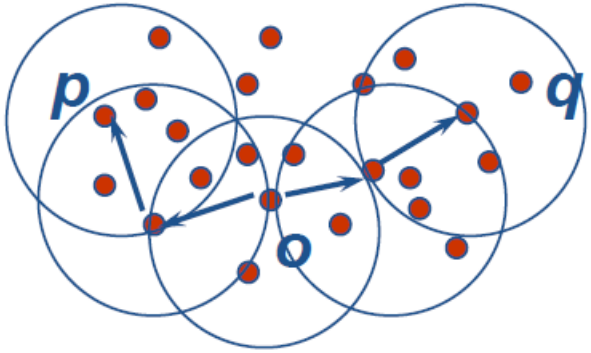


2.3.1 密度概念

- 密度可达
  - 如果存在一个对象链$p_1, \ldots, p_n$，$p_1 = q$，$p_n = p$，使得p_{i+1}是从p_i直接密度可达的，则称对象p是从对象q关于ε和MinPts(间接)密度可达的

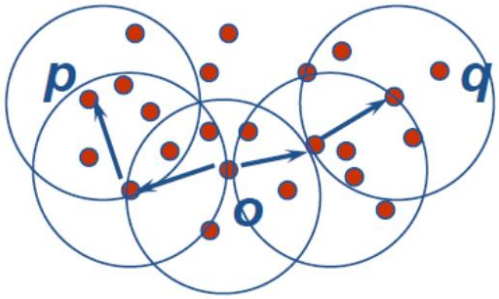


- 密度相连的
  - 如果存在对象o 使得，p 和q 都是从o关于ε和MinPts密度可达的，则称对象p与q关于ε和MinPts是密度相连的

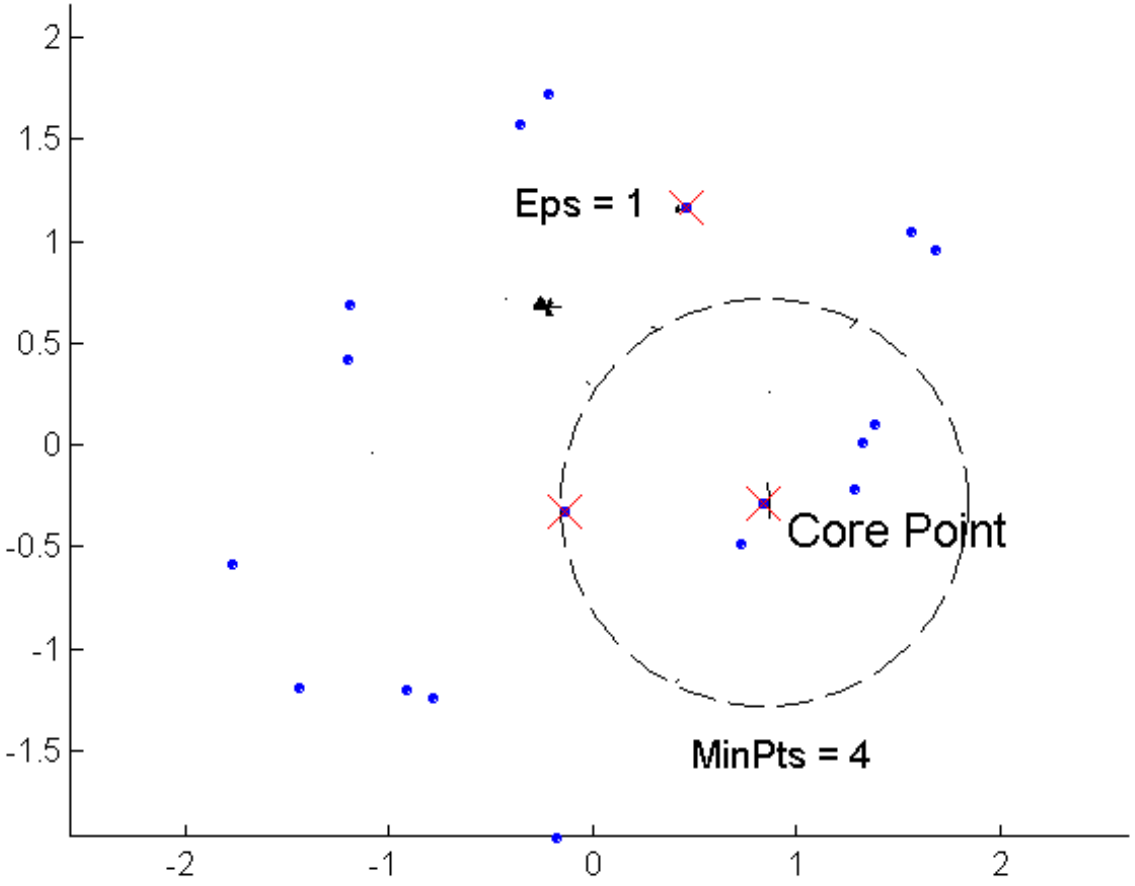


2.3.2 基于密度的簇

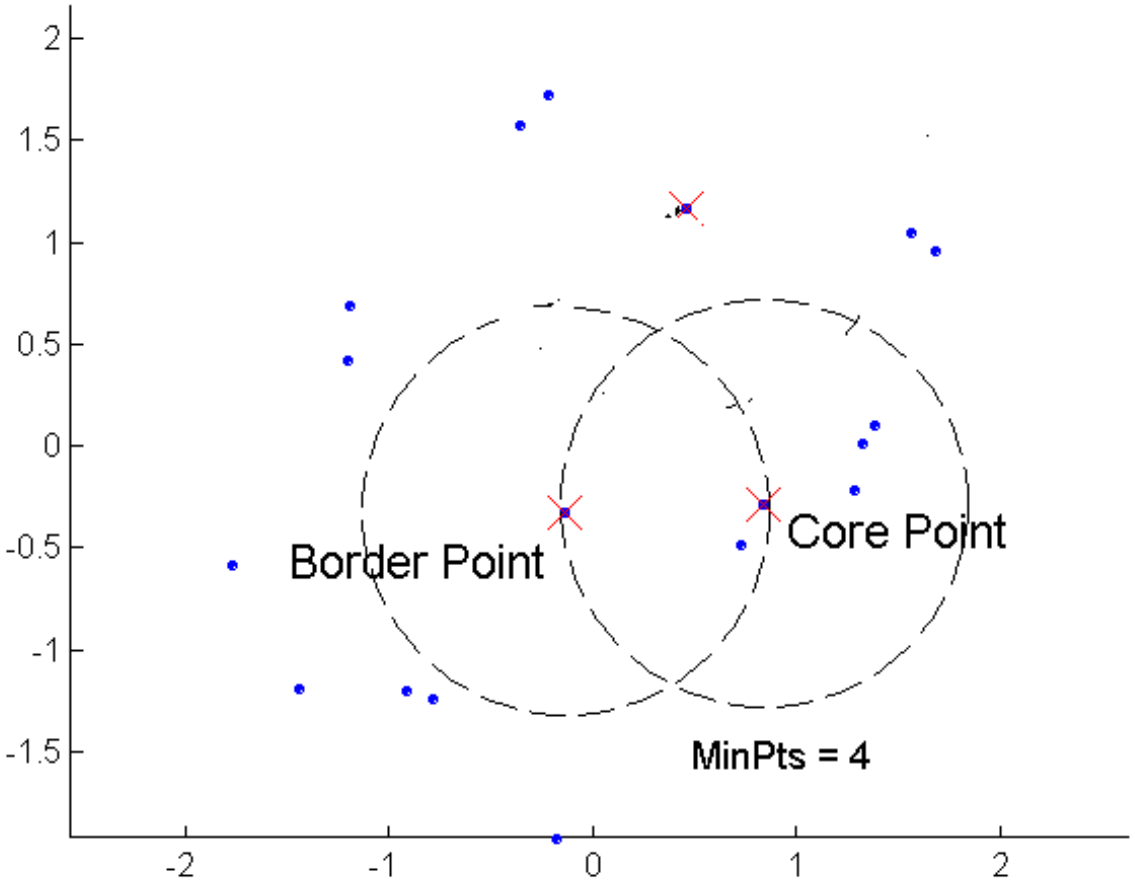
- 基于密度的簇是基于密度可达性的最大的密度相连对象的集合，即簇C 满足
  - 连通性：对于C 中任意的p 和q，p 与q 是关于ε和MinPts密度相连的
  - 极大性：对于任意的p 和q，如果p 属于C 簇，并且q 是从p 出发关于ε和MinPts密度可达的，则q 也属于C 簇
- 边界点（ Border point ）的ε邻域有少于MinPts 个对象，但它的邻域中有核心对象
- 噪声点（ Noise point ）是除核心对象和边界点之外的点
- 典型算法
  - DBSCAN、 DENCLUE、 OPTICS



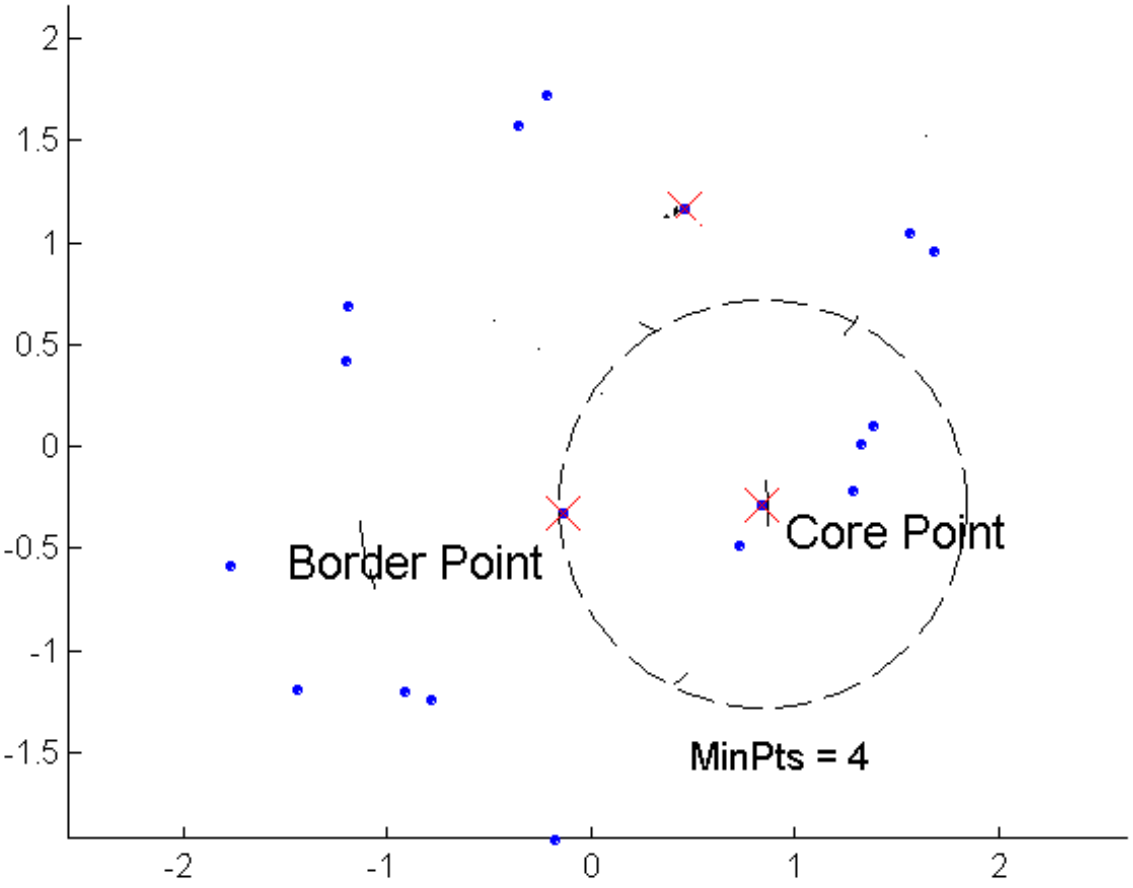
2.3.3 基本概念理解



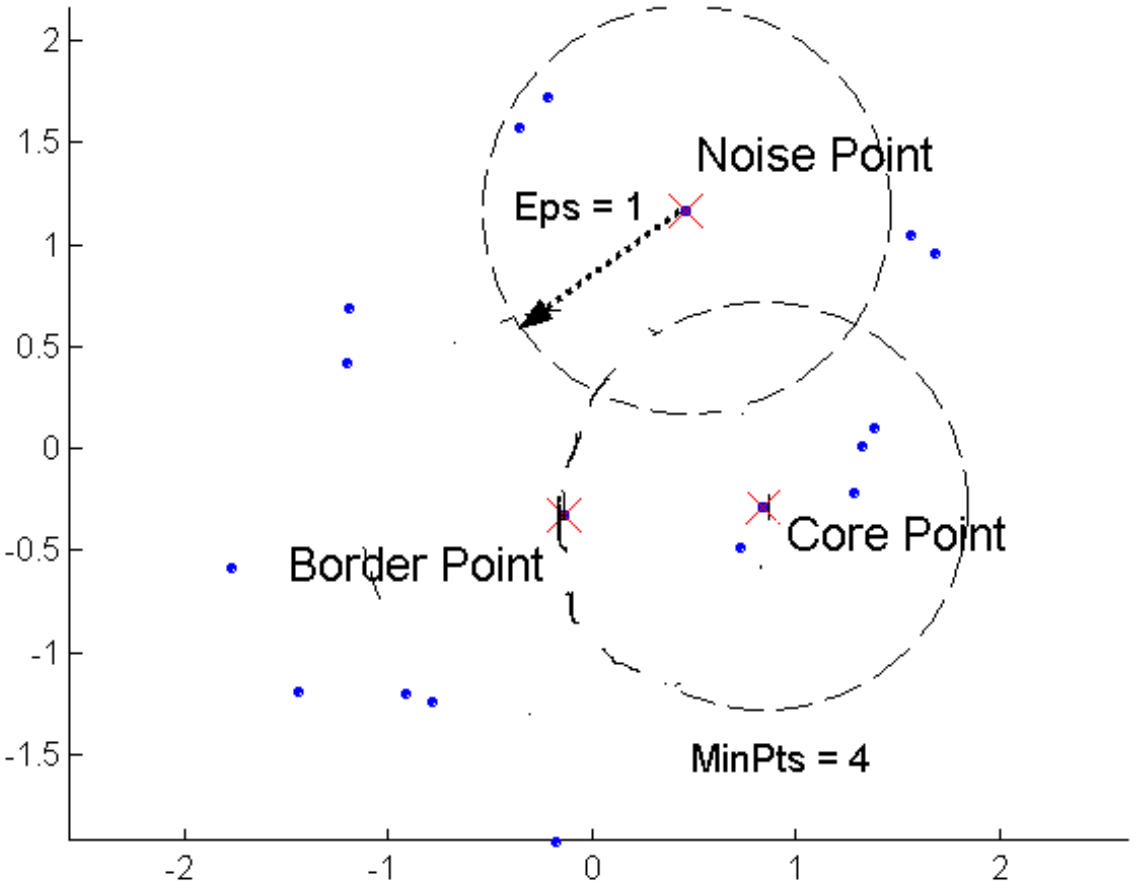
2.3.3 基本概念理解



2.3.3 基本概念理解



2.3.3 基本概念理解

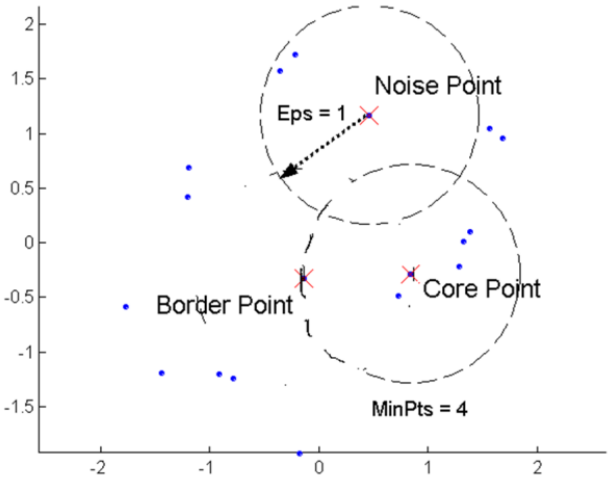

Noise Point
Eps = 1
Border Point
Core Point
MinPts = 4
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-1
0
1
2

2.3.4 DBSCAN

- 算法描述

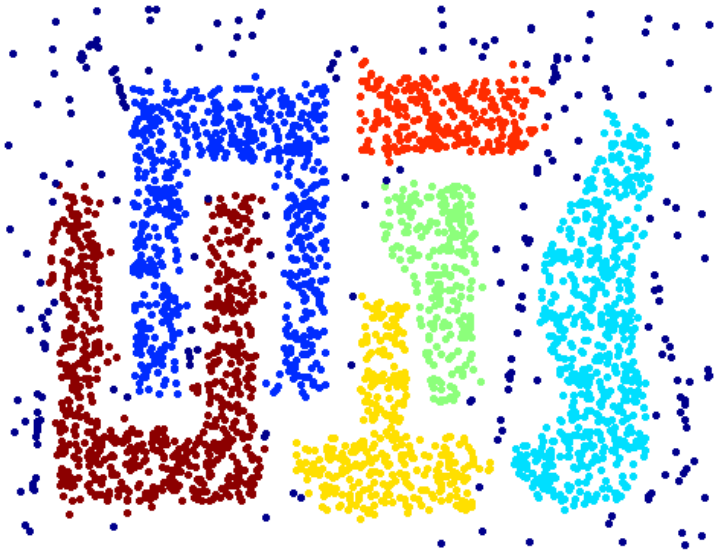
输入：包含n个对象的数据库，半径ε，最少数目MinPts。
输出：所有生成的簇，达到密度要求。

REPEAT
  从数据库中抽取一个未处理过的点；
  IF 抽出的点是核心点 THEN
        找出所有从该点密度可达的对象，形成一个簇
    ELSE
        抽出的点是边缘点(非核心对象)，跳出本次循环，寻找下一点；
UNTIL 所有点都被处理；



2.3.5 DBSCAN优点

Original Points

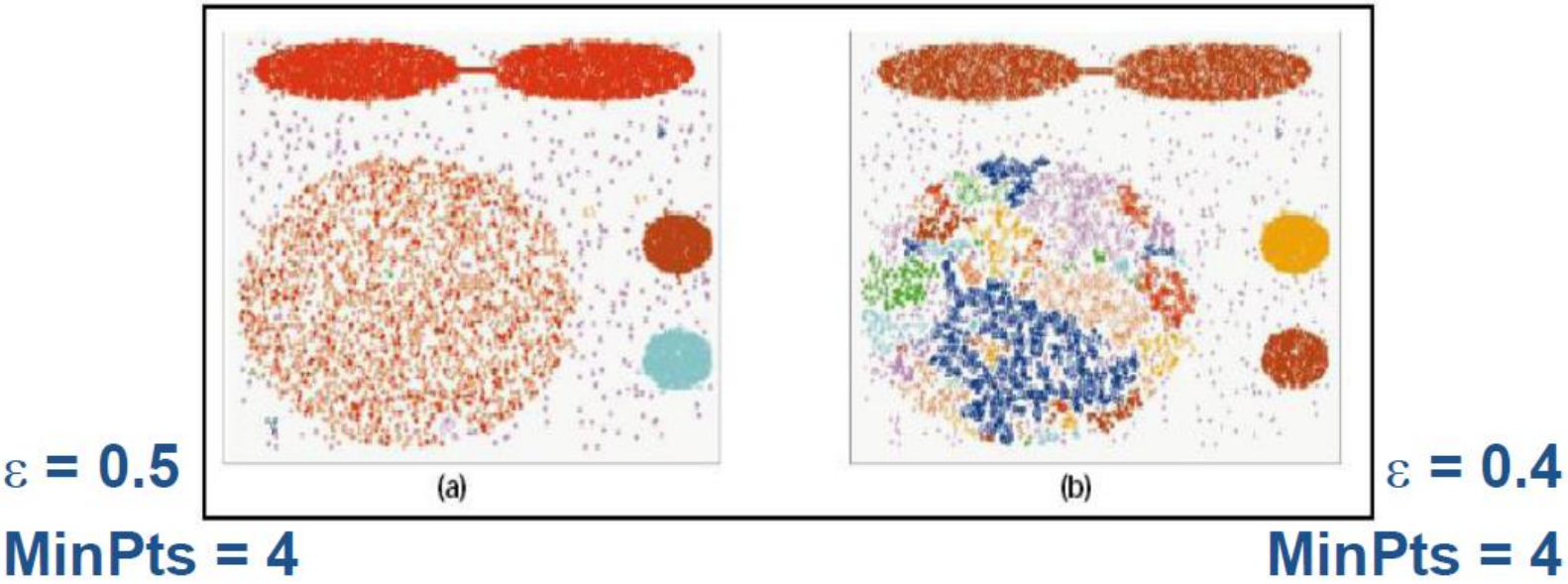


Clusters

- 抗噪声
- 能处理各种形状和大小集群

2.3.6 DBSCAN缺点

- 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。全局密度参数不能刻画内在的聚类结构



2.3.6 DBSCAN缺点

- 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。全局密度参数不能刻画内在的聚类结构

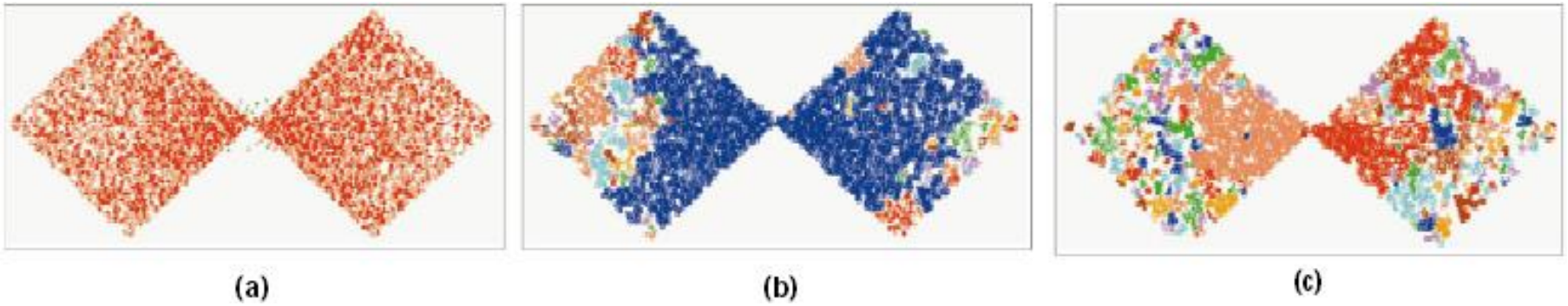


Figure 9. DBScan results for DS2 with MinPts at 4 and Eps at (a) 5.0, (b) 3.5, and (c) 3.0.

单选题 1分

设置

下列哪种聚类算法对聚簇的形状不敏感

A K-means

B K-中心点

C AGNES

D DBSCAN

提交

目 录

1. 聚类分析概述

2. 基本聚类方法

3. 聚类评估

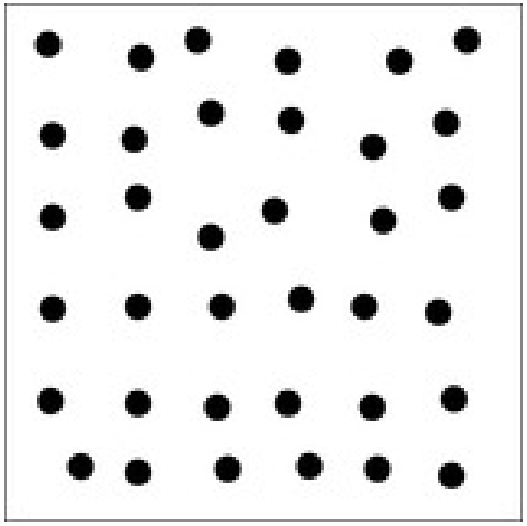
3 聚类评估

- 估计在数据集上**进行聚类的可行性和被聚类方法产生的结果的质量**
- 聚类评估的任务
  - 估计聚类趋势：评估数据集是否存在非随机结构
  - 确定数据集中的簇数：在聚类之前，估计簇数
  - 测定聚类质量：聚类之后，评估结果簇的质量

2.3.1 估计聚类趋势

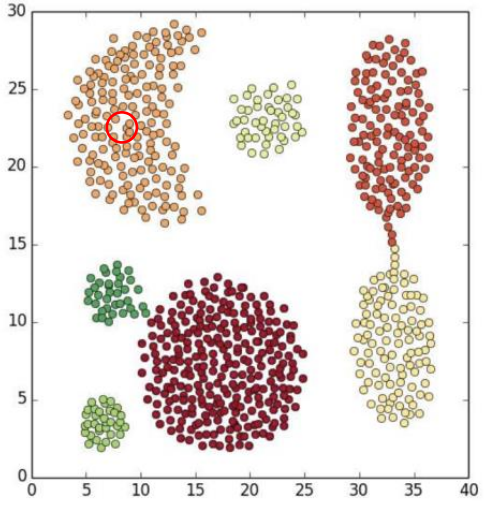
- 从统计学的角度，就是检测数据是否是随机或线性分布的

如果数据服从均匀分布，显然对其进行的聚类操作都是没有意义的



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

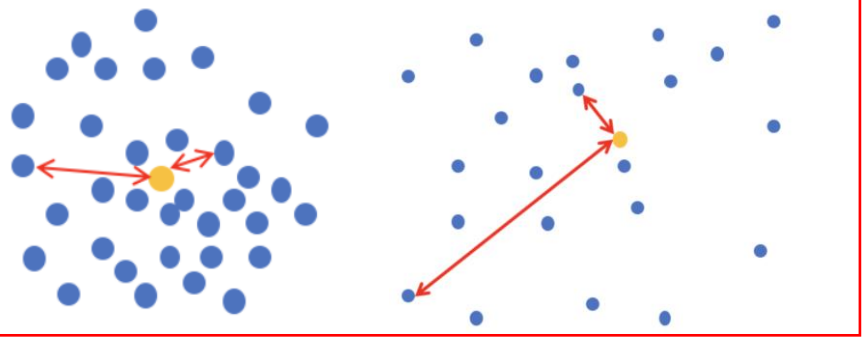
霍普金斯统计量

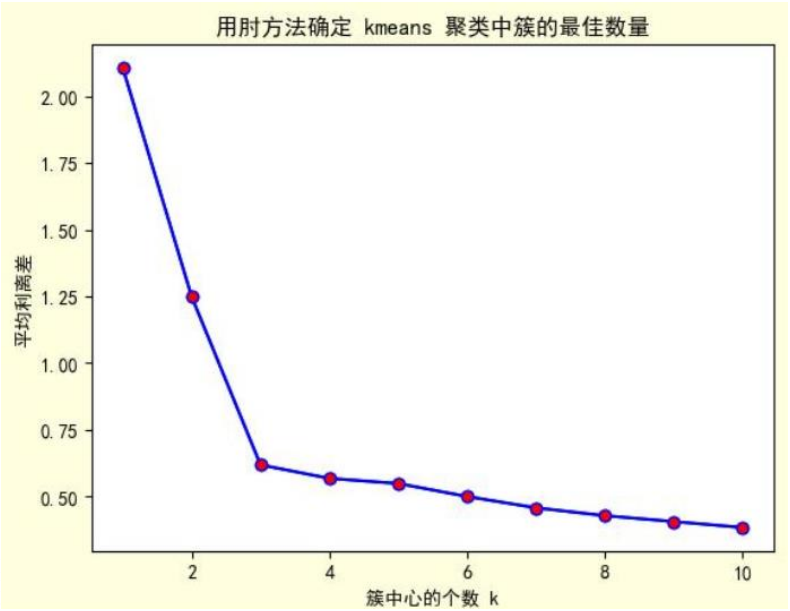


2.3.2 确定数据集中的簇数

- 实验方法
  - 对于n个点的数据集，簇数$\sqrt{n/2}$，每个簇约有$\sqrt{2n}$个点
- 肘方法（Elbow method）
  - 增加簇数可以降低**簇内方差**之和，但是如果形成太多的簇，降低**簇内方差之和的边缘效应**可能下降。

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





2.3.3 测定聚类质量

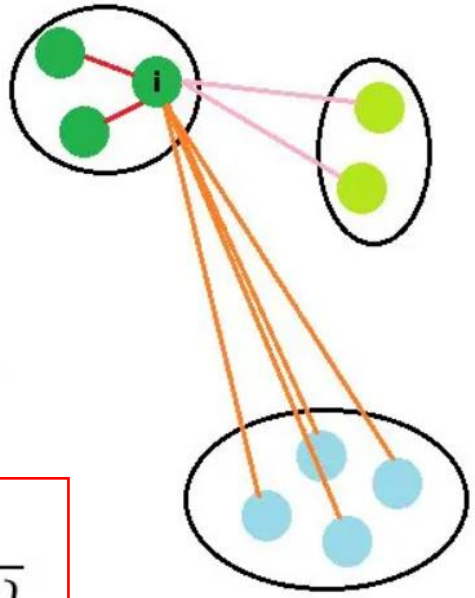
- 外在方法：有监督的
  - 用某种聚类质量度量对聚类结果和**基准**进行比较
  - 基准：一种理想的聚类，由专家构建
- 内在方法：无监督的
  - 通过考察簇的分离情况和簇的紧凑情况来评估聚类
  - 例：轮廓系数

✓ 值介于 [-1,1]

✓ 越趋近于1代表内聚度和分离度都相对较优

a(i) =avg

b(i)=min avg avg

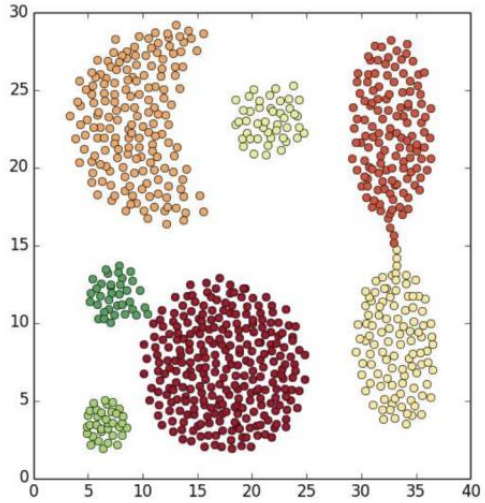


$$S(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

2.3.4 测定聚类质量

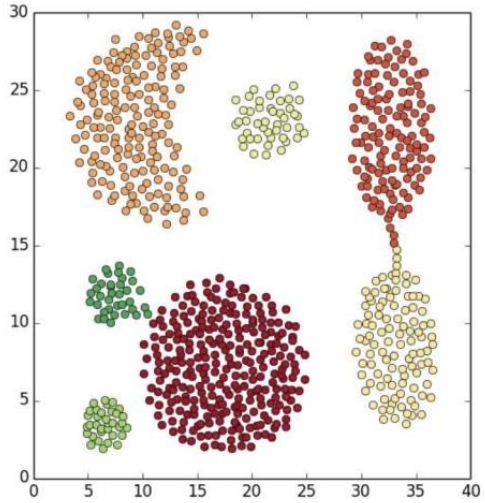
- 簇的同质性:簇内越纯越好
- 簇的完全性：相同类别被分配到同类别
- 碎布袋性：噪声不被聚到已存在的簇中
- 小簇保持性：小类别不被划分

2.3.4 测定聚类质量



- 簇的同质性
- 簇的完全性
- **碎布袋性**
  - 把一个异构对象放到一个纯聚簇中要比把它放到一个碎布袋里（例如，“杂项”或“其他”类别）更糟糕
  - 也就是说更希望噪音不被聚到已存在的聚簇中
  - putting a heterogeneous object into a pure cluster should be penalized more than putting it into a rag bag (i.e., “miscellaneous” or “other” category)
- 小簇保持性

2.3.4 测定聚类质量



- 簇的同质性
- 簇的完全性
- 碎布袋性
- **小簇保持性**
  - 把一个小类别聚簇分成更小的聚簇比把一个大类别分成小聚簇更有害
  - 也就是我们希望小类别聚簇不再被划分
  - splitting a small category into pieces is more harmful than splitting a large category into pieces