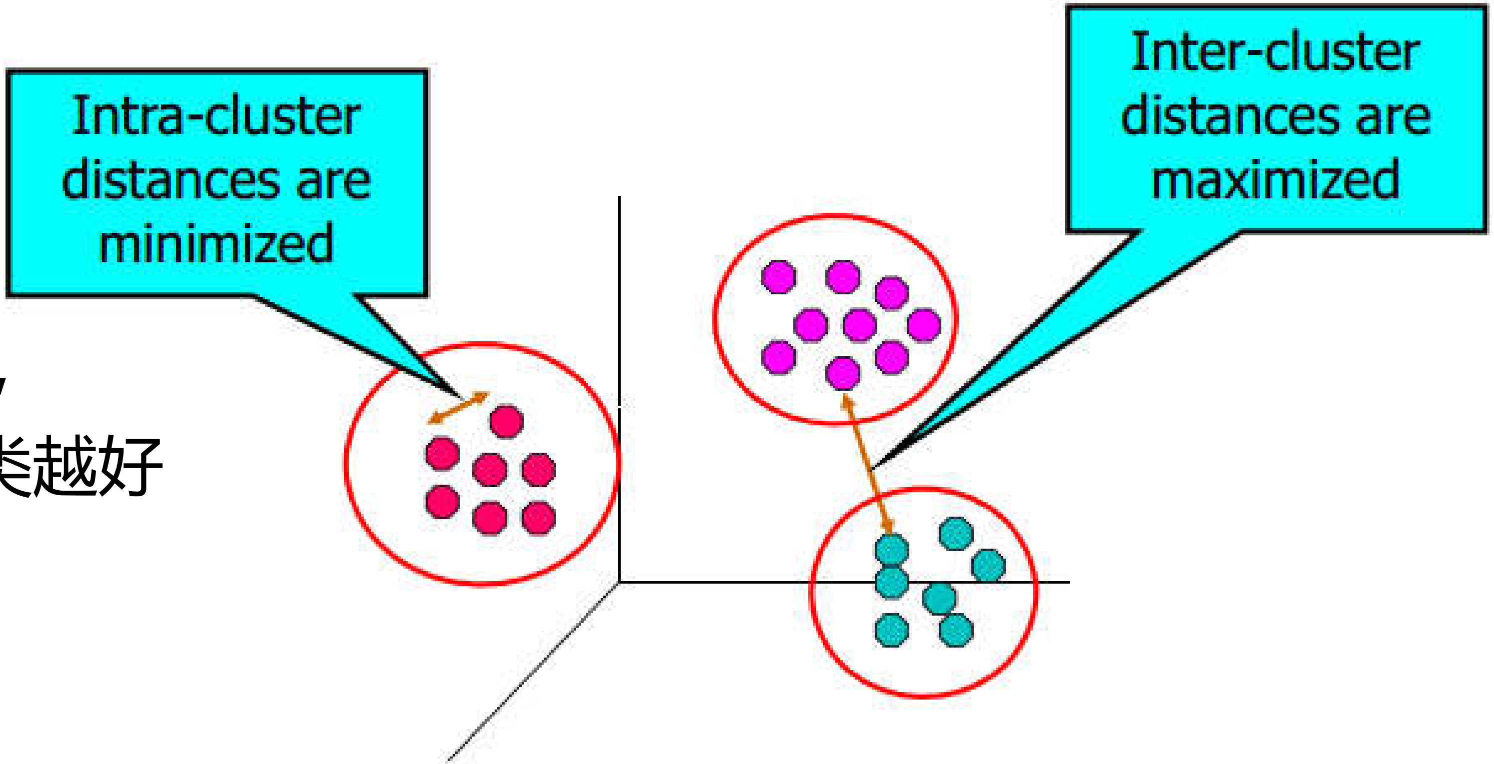
大数据分析

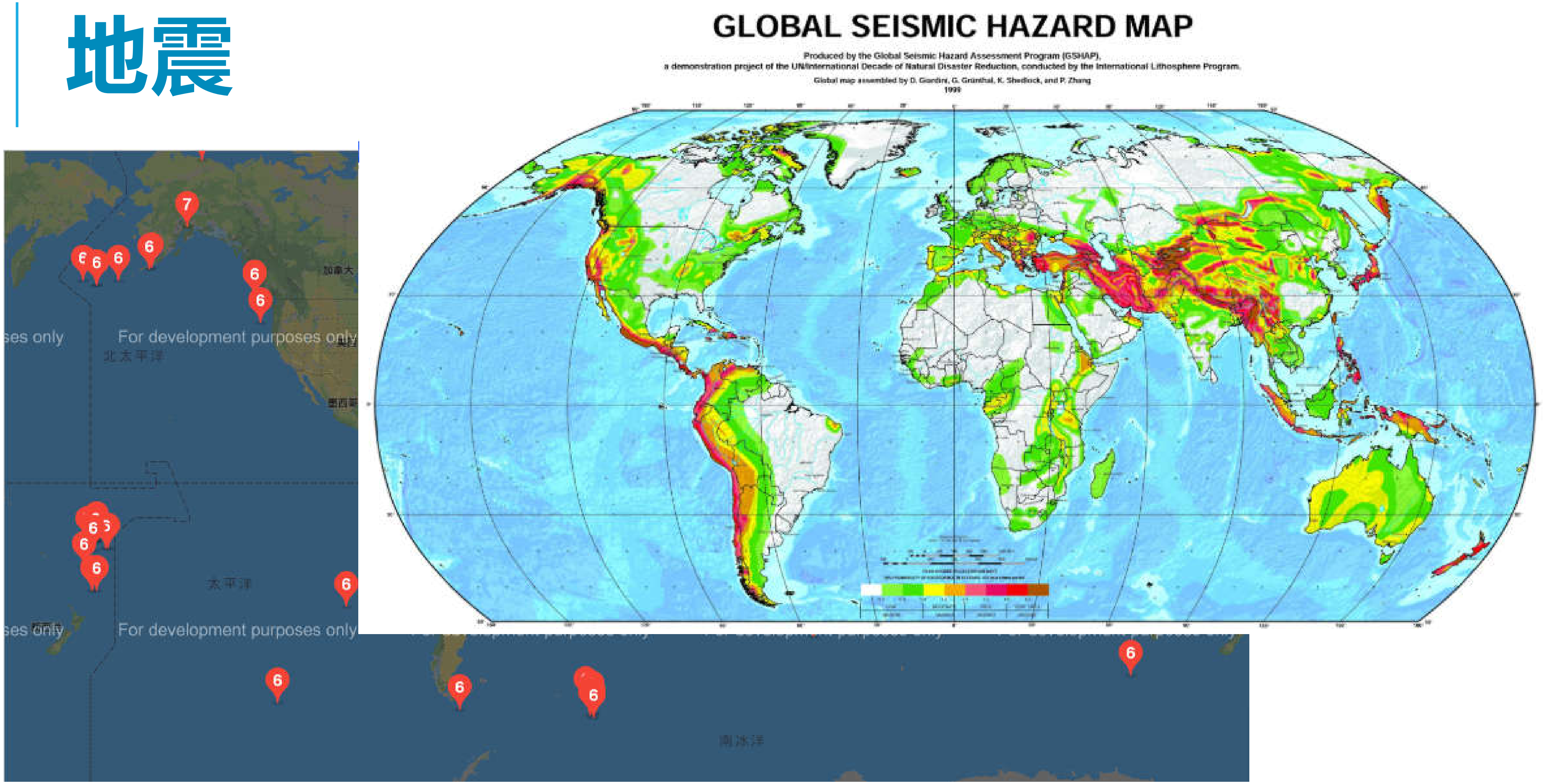
聚类分析

什么是聚类

- 聚类是指对相似的数据对象进行分组，形成簇。
- 组内的对象之间是相似的，不同组中的对象是不同的
- 组内相似度越大，组间差别越大，聚类越好



地震



GLOBAL SEISMIC HAZARD MAP
Produced by the Global Seismic Hazard Assessment Program (GSHAP),
a demonstration project of the UN/International Decade of Natural Disaster Reduction, conducted by the International Lithosphere Program.
Global map assembled by D. Giardini, G. Grünthal, K. Shedlock, and P. Zhang
1999
加拿大
For development purposes only
北太平洋
太平洋
For development purposes only
南冰洋

簇的概念

How many clusters?

Six Clusters

Two Clusters

Four Clusters

K-MEANS

- 算法过程：
  - 1）从N个样本数据中随机选取K个对象作为初始的质心；
  - 2）分别计算每个样本到各个质心的距离，将对象分配到距离最近的质心；
  - 3）所有对象分配完成后，重新计算K个簇的质心；
  - 4）与前一次计算得到的K个质心比较，如果质心发生变化，转2)，否则转5)；
  - 5）当质心不发生变化时停止并输出聚类结果。

相似性度量-连续属性

- 对于连续属性，要先对各属性值进行零-均值规范，再进行距离的计算。
- 用 p 个属性来表示 n 个样本的数据矩阵如下：

$$\begin{bmatrix} x_{11} & \cdots & x_{1p} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{np} \end{bmatrix}$$

相似性度量-连续属性（续）

$$\begin{bmatrix} x_{11} & \cdots & x_{1p} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{np} \end{bmatrix}$$

- 欧几里德距离：

$$d(i,j)=\sqrt{(x_{i1}-x_{j1})^2+(x_{i2}-x_{j2})^2+\cdots+(x_{ip}-x_{jp})^2}$$

- 曼哈顿距离：

$$d(i,j)=|x_{i1}-x_{j1}|+|x_{i2}-x_{j2}|+\cdots+|x_{ip}-x_{jp}|$$

- 闵可夫斯基距离：

$$d(i,j)=\sqrt[q]{|(x_{i1}-x_{j1}|)^q+(|x_{i2}-x_{j2}|)^q+\cdots+(|x_{ip}-x_{jp}|)^q}$$

相似性度量-文档数据

- 对于文档数据使用余弦相似性度量，先将文档数据整理成文档—词矩阵格式：

	lost	win	team	score	music	happy	sad	…	coach
文档一	14	2	8	0	8	7	10	…	6
文档二	1	13	3	4	1	16	4	…	7
文档三	9	6	7	7	3	14	8	…	5

- 两个文档之间的相似度的计算公式为：

$$d(i,j) = \cos(i,j) = \frac{\vec{i} \cdot \vec{j}}{|\vec{i}||\vec{j}|}$$

目标函数

- 使用误差平方和SSE作为度量聚类质量的目标函数，对于两种不同的聚类结果，选择误差平方和较小的分类结果。

1）连续属性的SSE计算公式为：

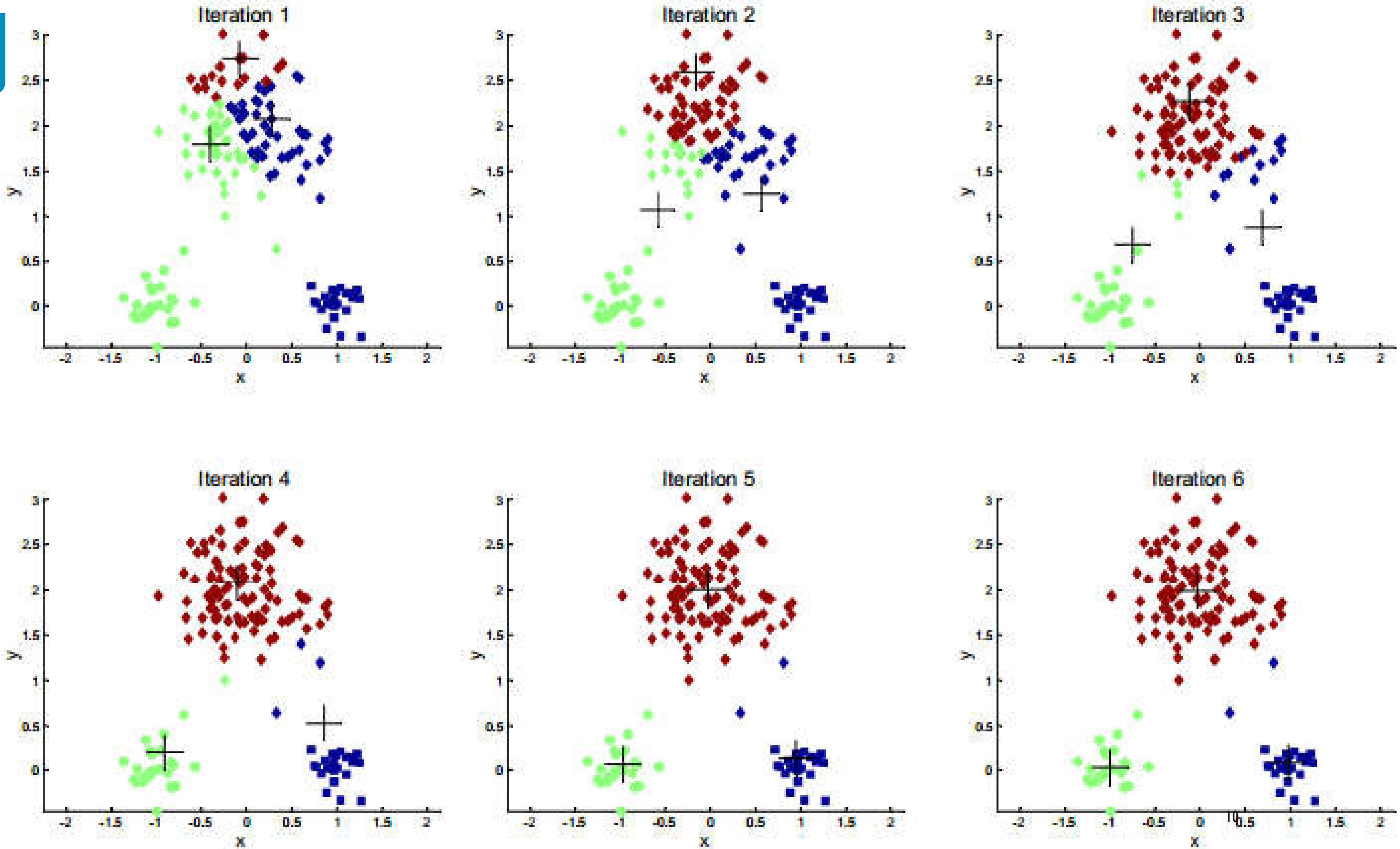
$$SSE = \sum_{i=1}^{K} \sum_{x \in E_i} dist(e_i, x)^2$$

2）文档数据的SSE计算公式为：

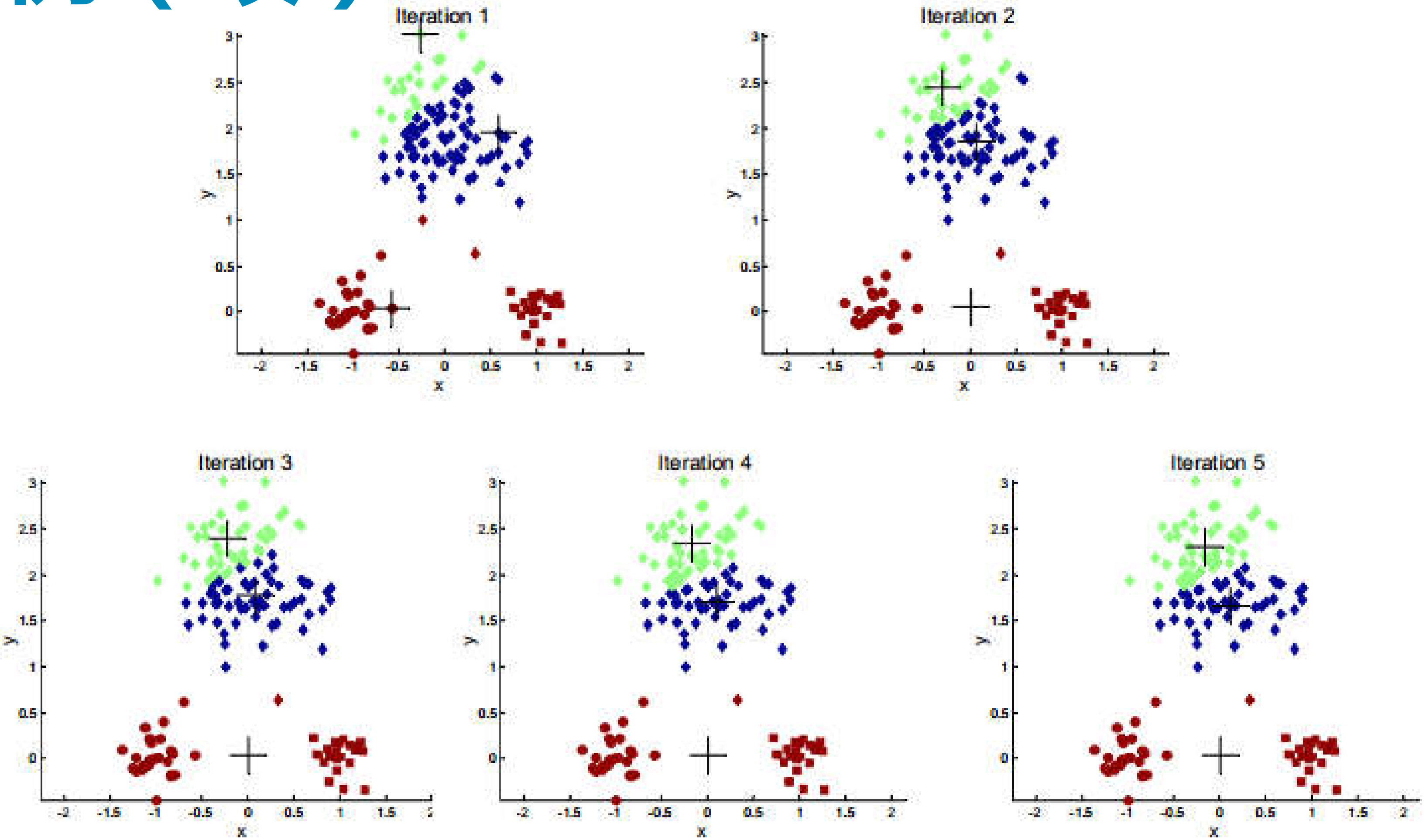
$$SSE = \sum_{i=1}^{K} \sum_{x \in E_i} \cos(e_i, x)^2$$

符号	含义
K	簇的个数
E_i	第i个簇
x	对象（样本）
e_i	簇 E_i的质心

举例


Iteration 1
Iteration 2
Iteration 3
Iteration 4
Iteration 5
Iteration 6
x
y

举例（续）


Iteration 1
Iteration 2
Iteration 3
Iteration 4
Iteration 5
x
y

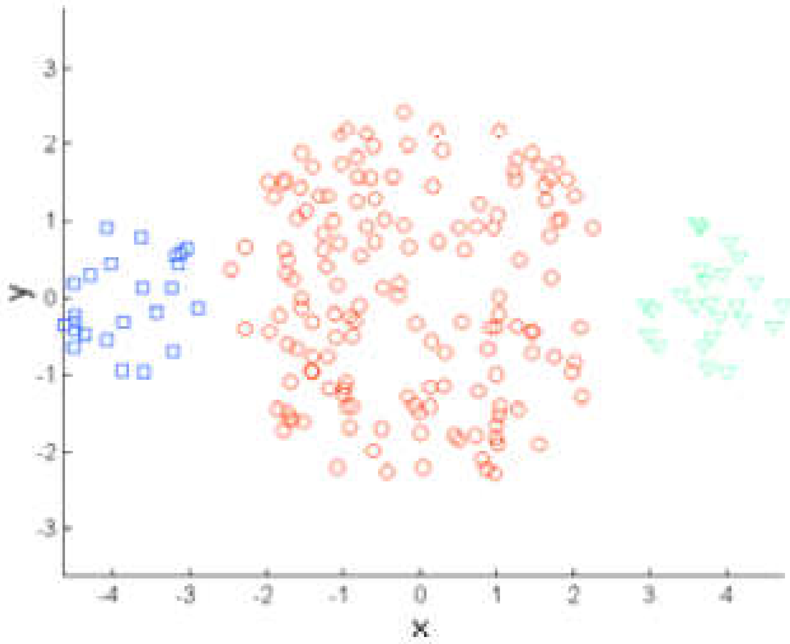
选择初始质心

- 多次运行，每次随机地选取一组初始质心
- 使用层次聚类技术进行聚类，从层次聚类中提取k个簇，并用这些簇的质心作为初始质心
- 随机地选择第一个点，或取所有点的质心作为第一个点，然后对于每个后继初始质心，选择离已经选取过的质心最远的点。
- 二分k均值，后处理
- 。。。

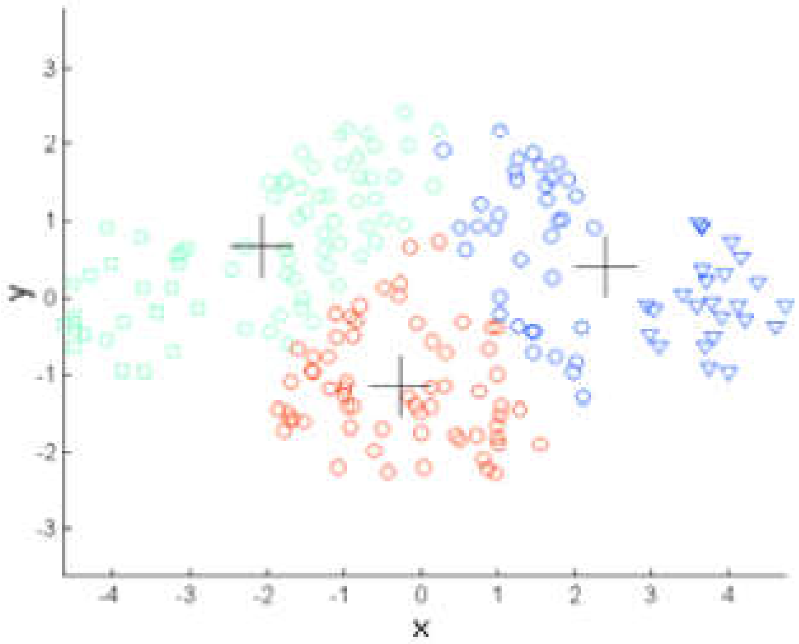
K-MEANS的局限

- K-means has problems when clusters are of differing
  - Sizes
  - Densities
  - Non-globular shapes
- K-means has problems when the data contains outliers.

不同尺寸

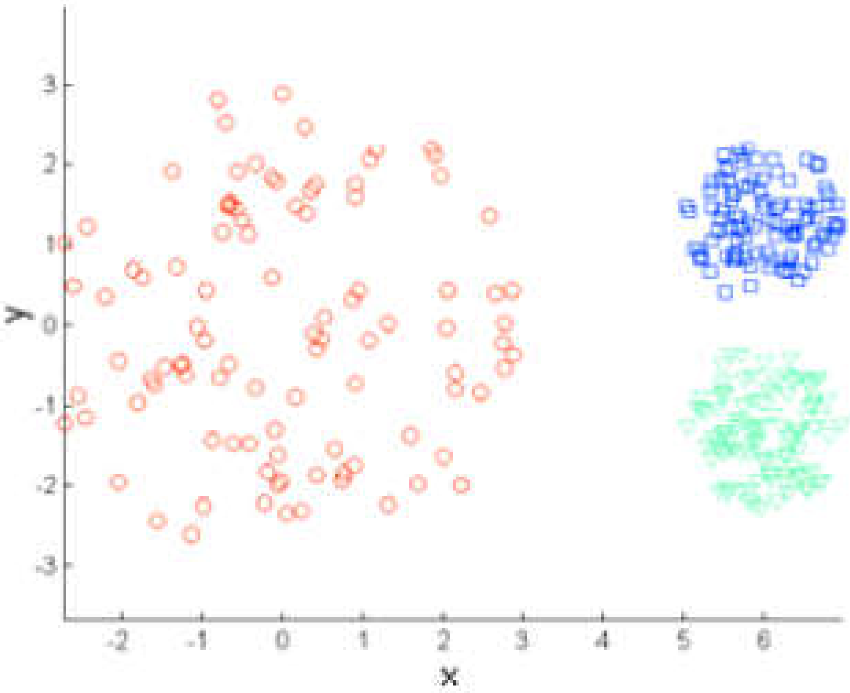


Original Points

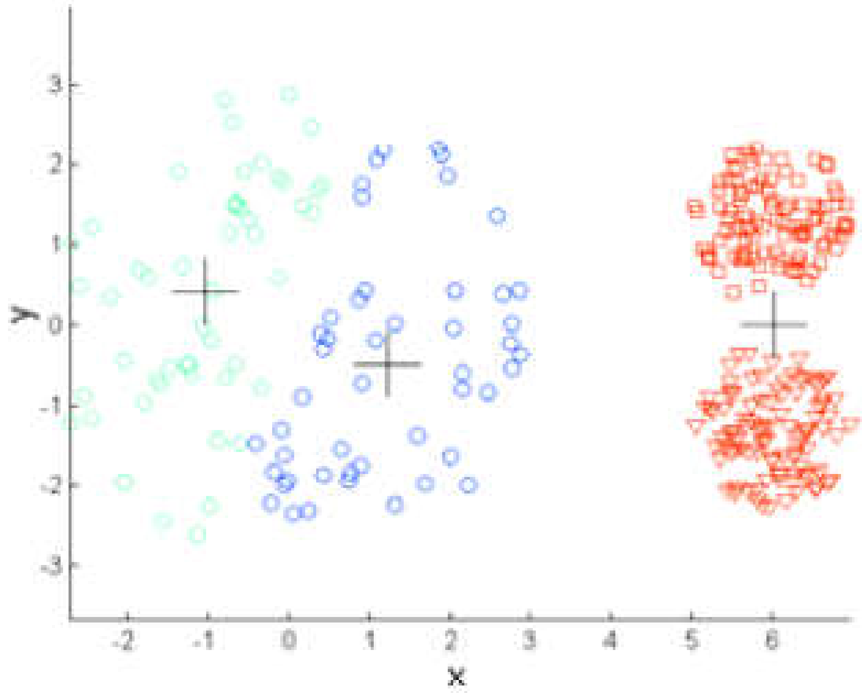


K-means (3 Clusters)

不同密度

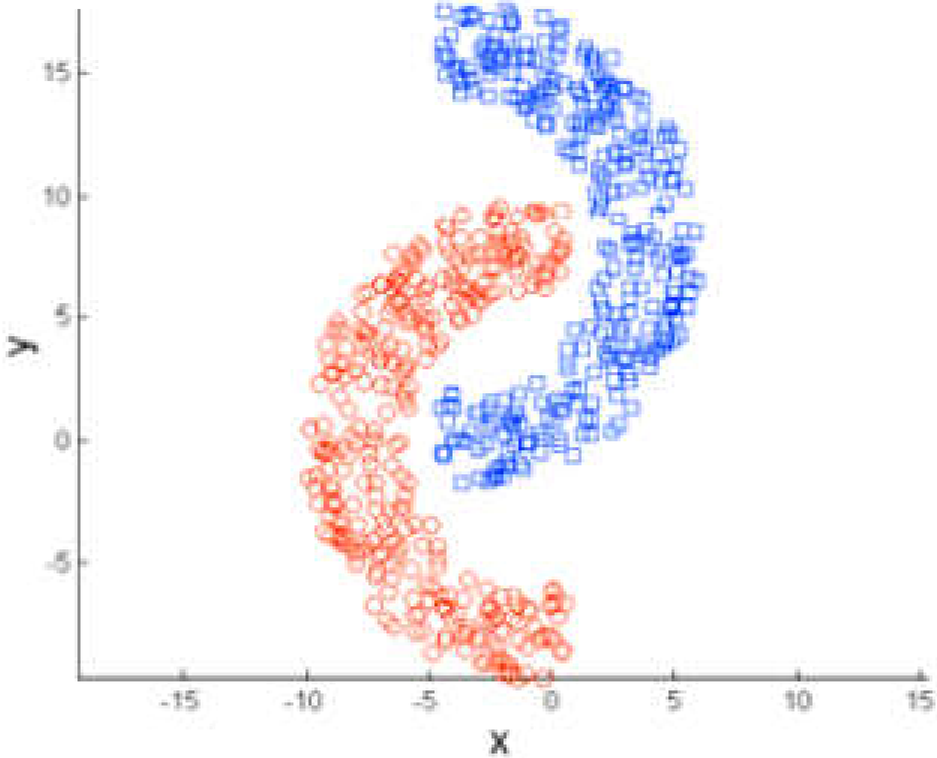


Original Points

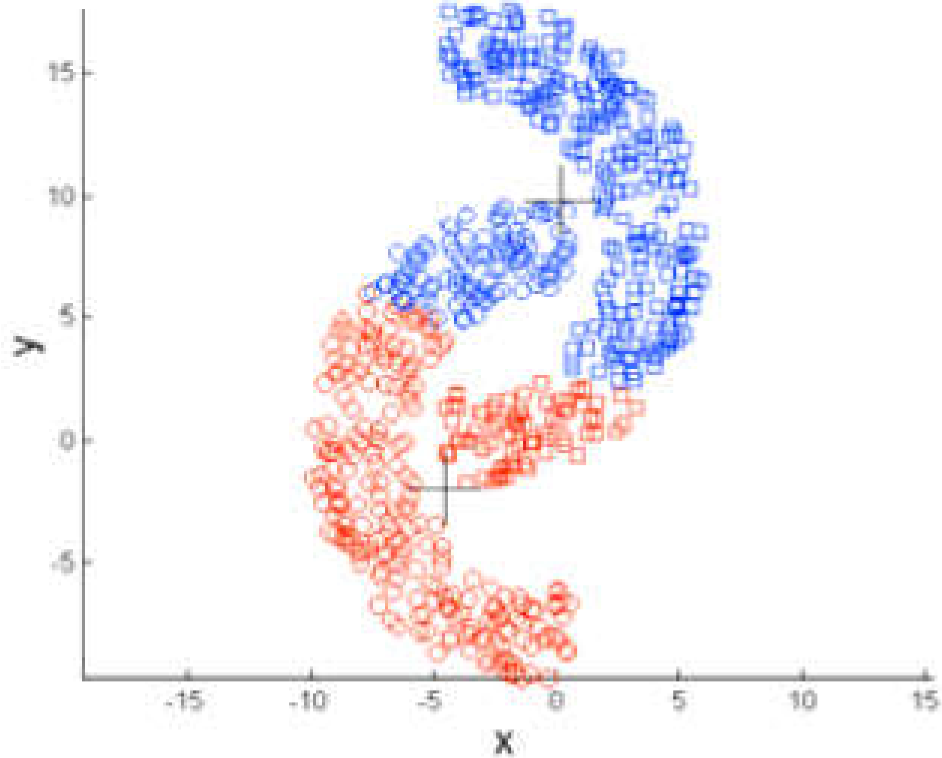


K-means (3 Clusters)

非球形

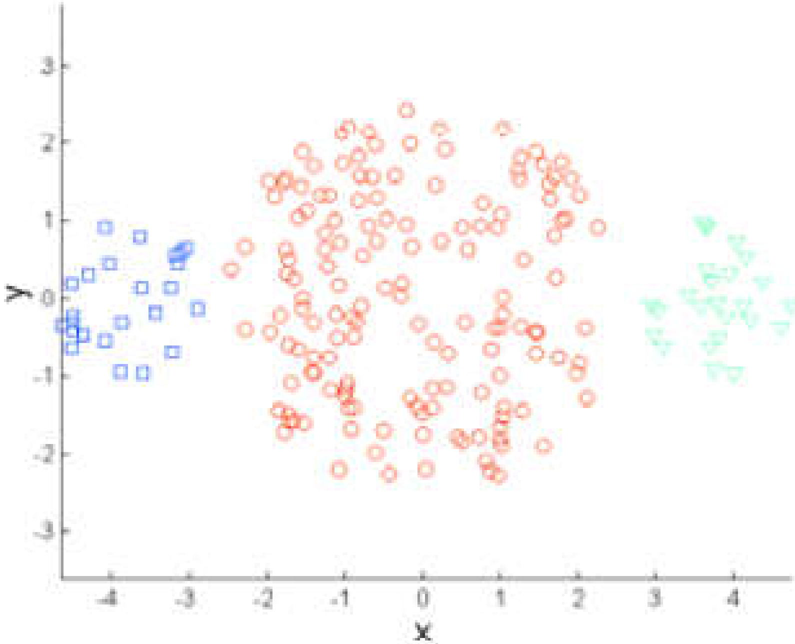


Original Points

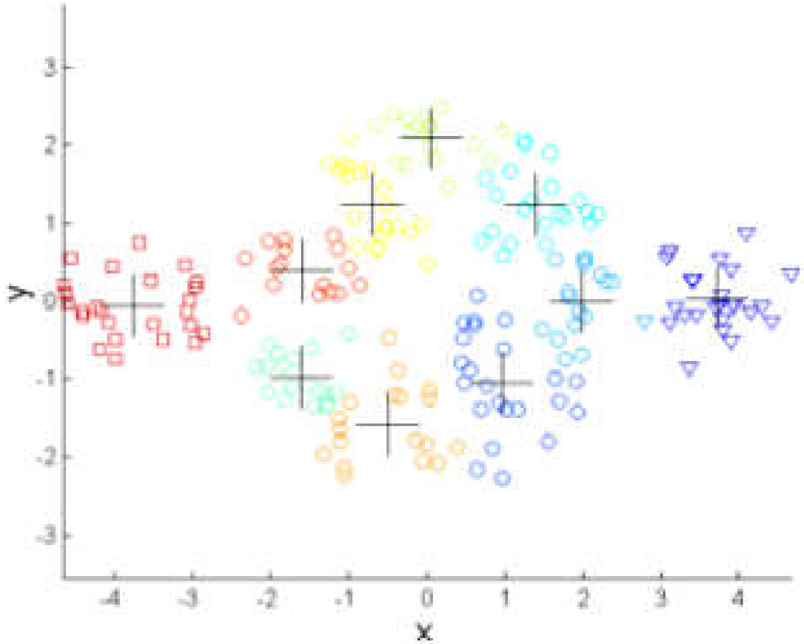


K-means (2 Clusters)

解决方案

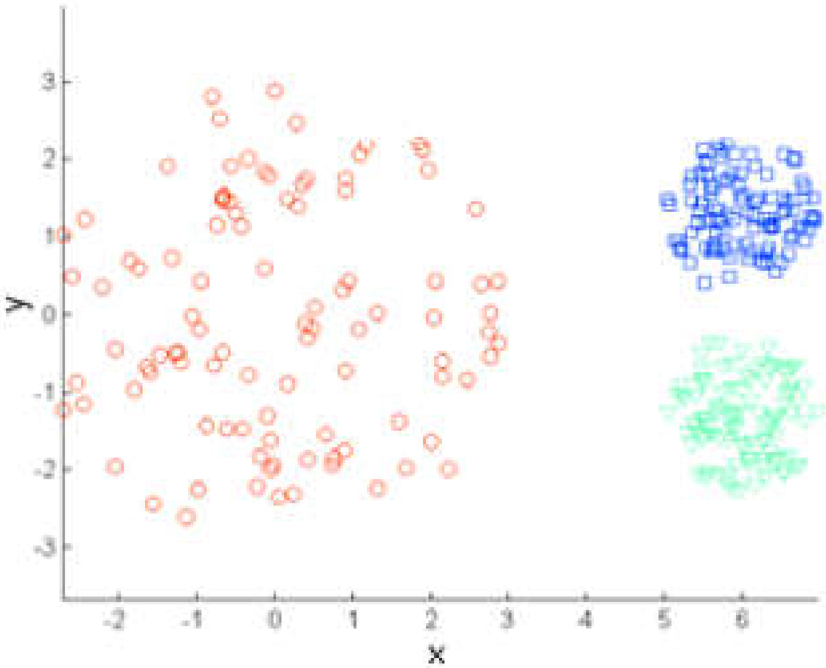


Original Points

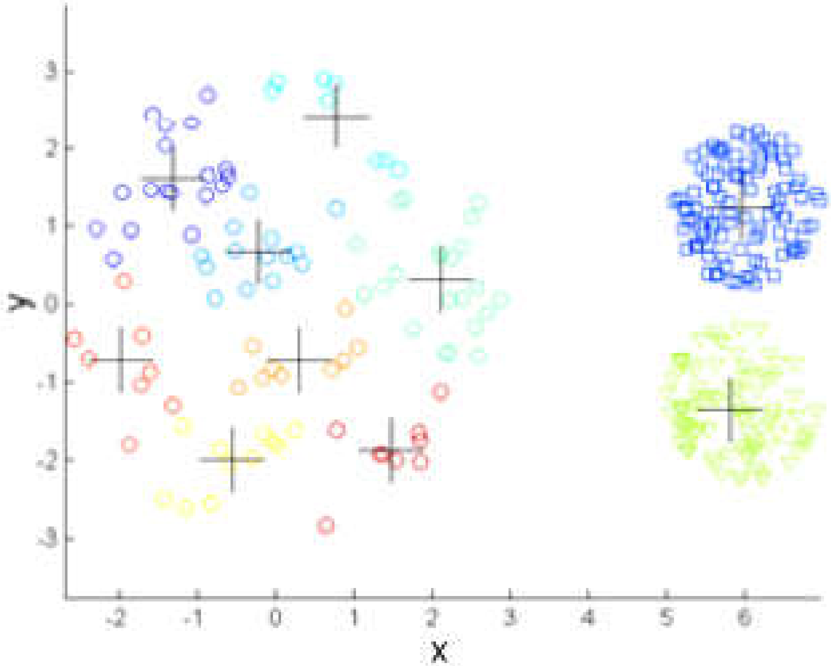


K-means Clusters

解决方案（续）

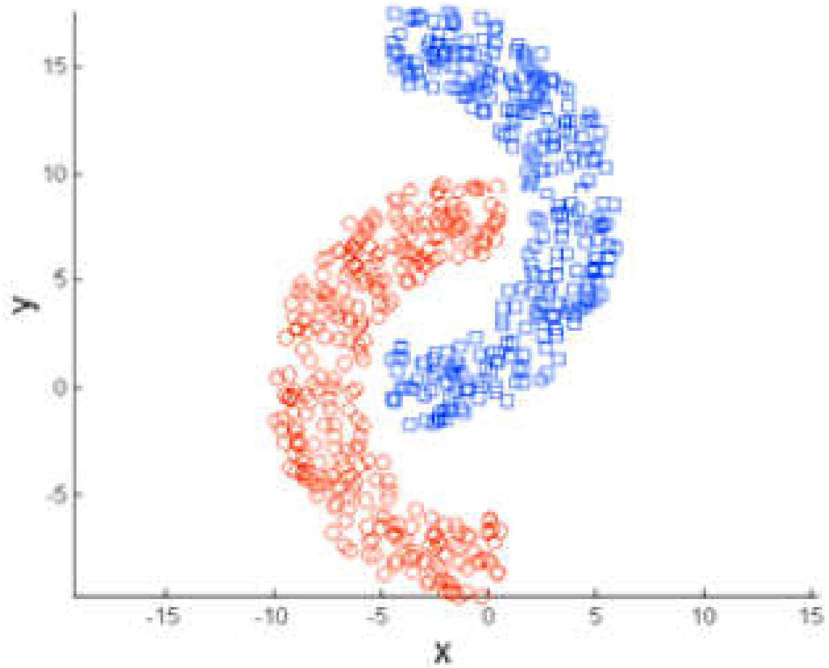


Original Points

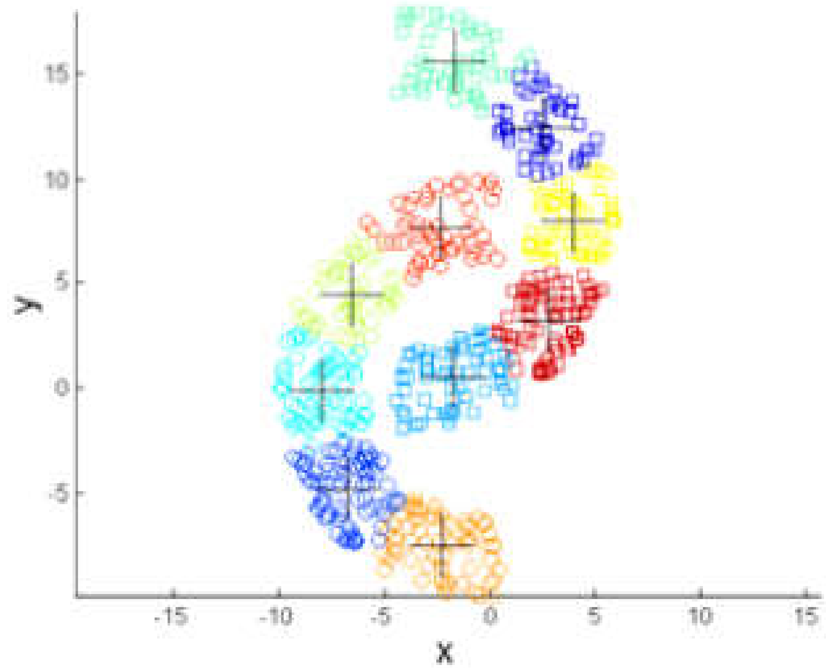


K-means Clusters

解决方案（续）



Original Points



K-means Clusters

课后调研

- 如何优化K-means算法？

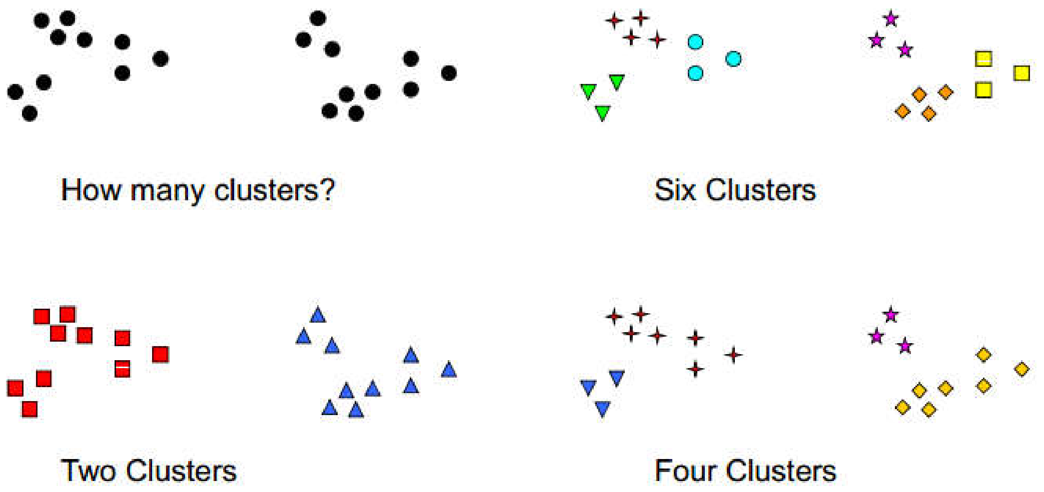
举例：餐饮客户的消费行为

- RFM模型
  - R（Recency）：最近一次消费时间与截止时间的间隔。
  - F（Frequency）：客户在某段时间内所消费的次数。
  - M（Monetary）：客户在某段时间内所消费的金额。

ID	R	F	M
1	37	4	579
2	35	3	616
3	25	10	394
4	52	2	111
5	36	7	521
6	41	5	225
7	56	3	118
8	37	5	793
9	54	2	111
10	5	18	1086
…	…	…	…

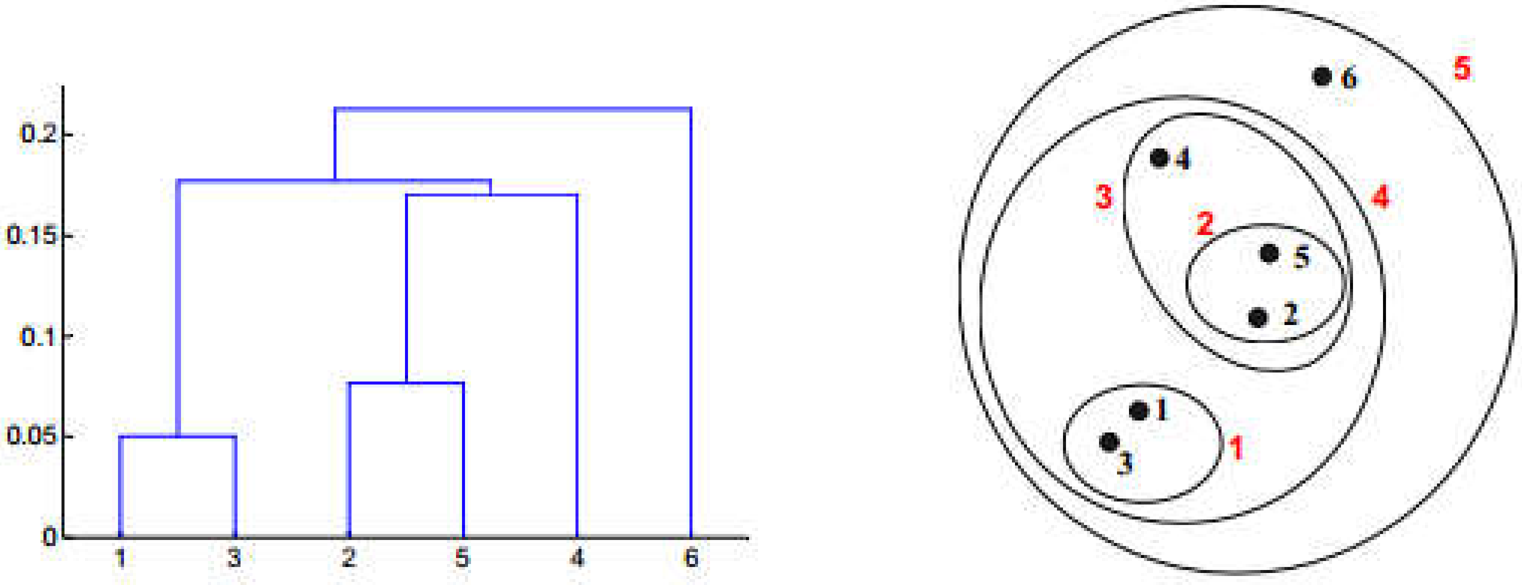
层次聚类

- 划分聚类简单地将数据对象集划分成不重叠的子集（簇），使得每个数据对象恰在一个子集中。
- 层次聚类允许簇具有子簇。层次聚类是嵌套簇的集族，组织成一棵树，除叶结点外，树的每一个结点都是其子女结点的并，而树根是包含所有对象的簇。



凝聚层次聚类

- 从点作为个体簇开始，每一步合并两个最接近的簇
- 常常使用树状图和嵌套簇图来表示簇-子簇联系和簇合并次序



基本凝聚层次聚类算法

■从个体点作为簇开始，相继合并两个最接近的簇，直到只剩下一个簇

Basic algorithm is straightforward

1. Compute the proximity matrix
2. **Repeat**
3. Merge the two closest clusters
4. Update the proximity matrix
5. **Until** only a single cluster remains

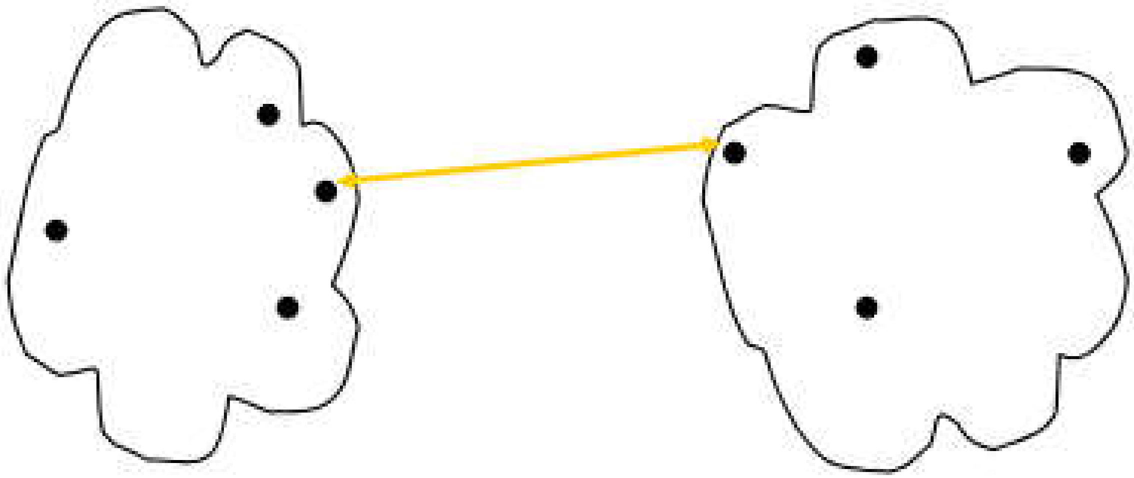
关键操作：计算两个簇之间的邻近度

邻近度定义

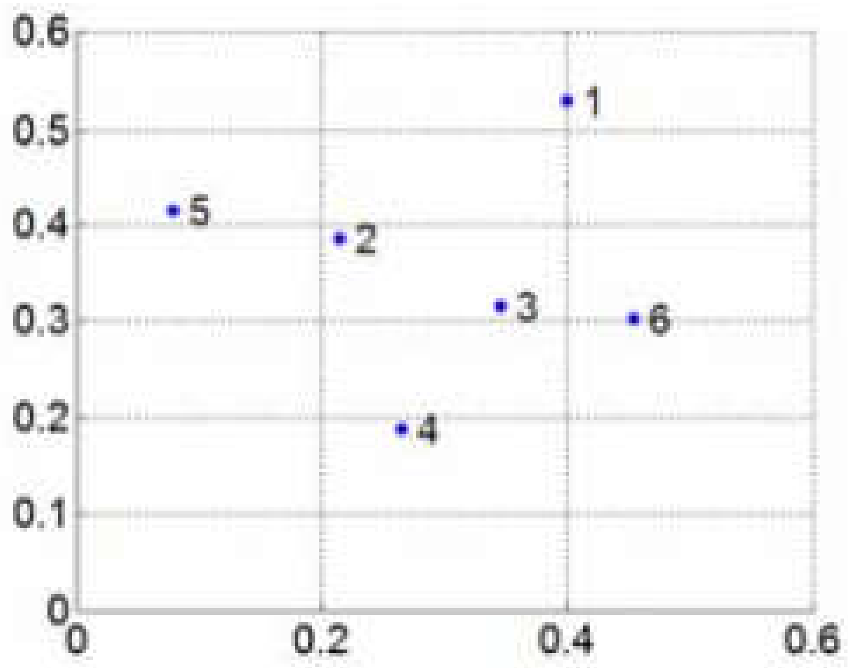
- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  – Ward's Method uses squared error

MIN（单链）

- 取不同簇的两个最近的点之间的邻近度作为簇的邻近度



举例



	P1	P2	P3	P4	P5	P6
P1	0	0.2357	0.2218	0.3688	0.3421	0.2347
P2	0.2357	0	0.1483	0.2042	0.1388	0.2540
P3	0.2218	0.1483	0	0.1513	0.2843	0.1100
P4	0.3688	0.2042	0.1513	0	0.2932	0.2216
P5	0.3421	0.1388	0.2843	0.2932	0	0.3921
P6	0.2347	0.2540	0.1100	0.2216	0.3921	0

	P1	P2	P3、6	P4	P5
P1	0	0.2357	0.2218	0.3688	0.3421
P2	0.2357	0	0.1483	0.2042	0.1388
P3、6	0.2218	0.1483	0	0.1513	0.2843
P4	0.3688	0.2042	0.1513	0	0.2932
P5	0.3421	0.1388	0.2843	0.2932	0

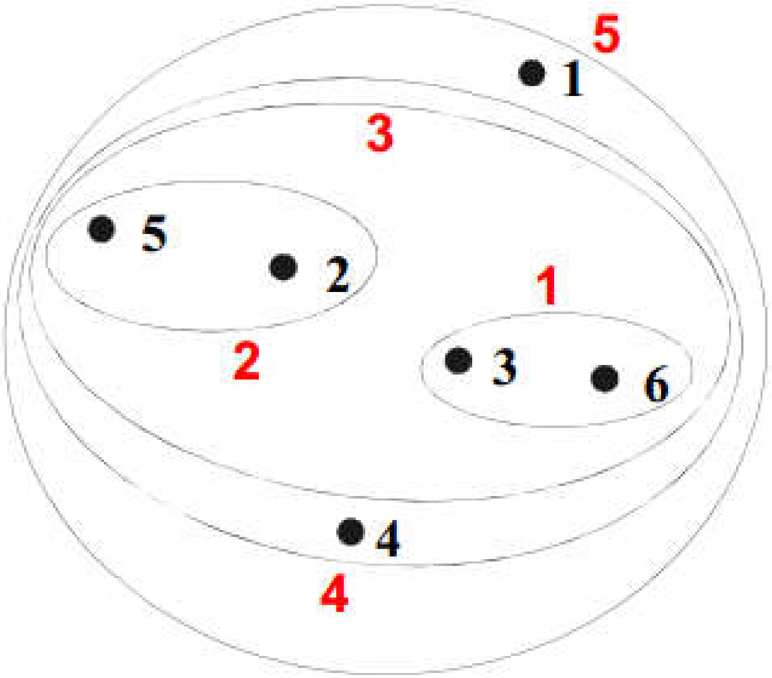
	P1	P2、5	P3、6	P4
P1	0	0.2357	0.2218	0.3688
P2、5	0.2357	0	0.1483	0.2042
P3、6	0.2218	0.1483	0	0.1513
P4	0.3688	0.2042	0.1513	0

dist（{1},{3,6})=min(dist(1,3),dist(1,6))
=min(0.2218,0.2347)=0.2218

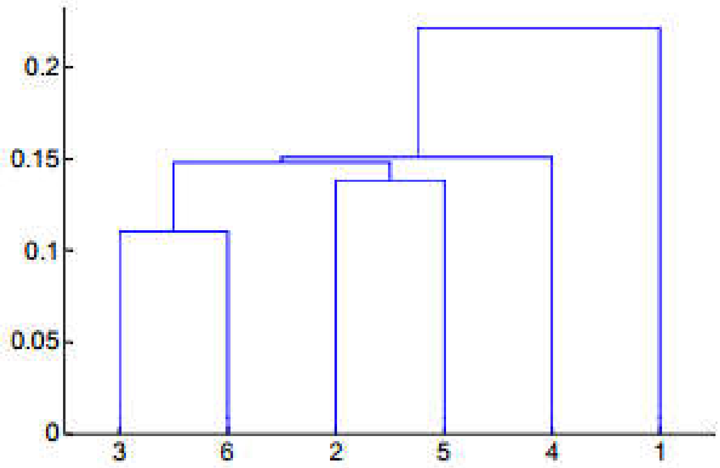
	P1	P2、5、3、6	P4
P1	0	0.2218	0.3688
P2、5、3、6	0.2218	0	0.1513
P4	0.3688	0.1513	0

	P1	P2、5、3、6、4
P1	0	0.2218
P2、5、3、6、4	0.2218	0

举例（续）



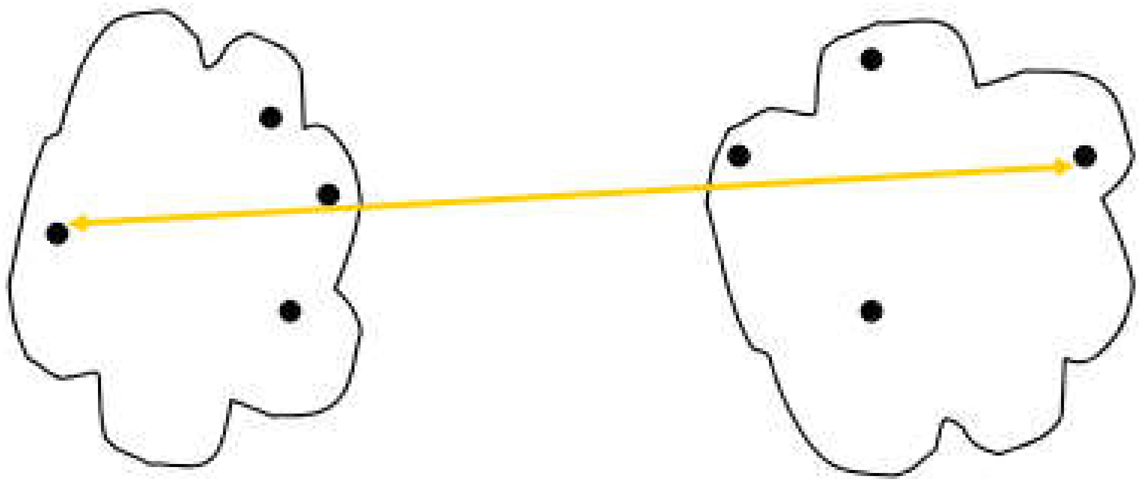
Nested Clusters



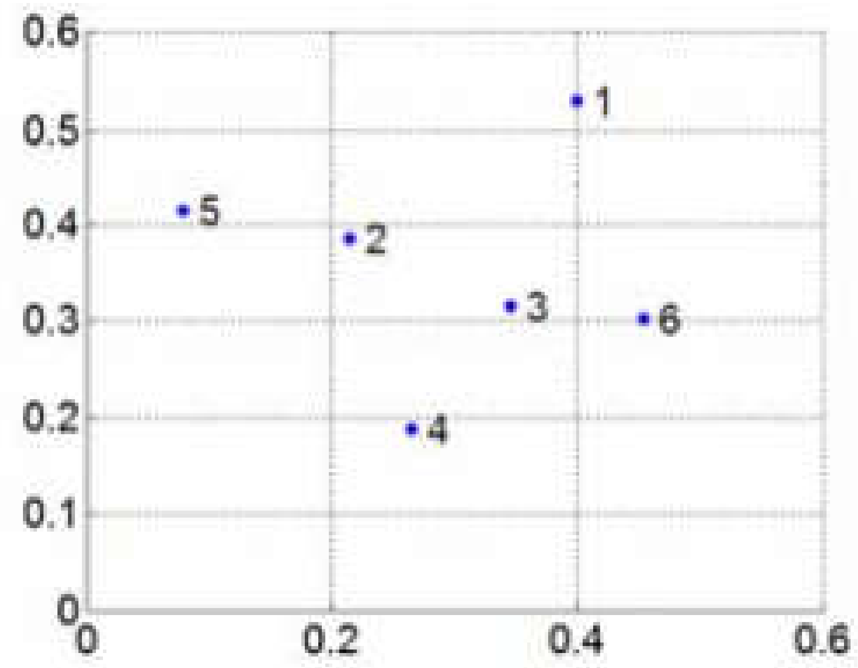
Dendrogram

MAX（全链）

- 取不同簇中两个最远的点之间的邻近度作为簇的邻近度



举例



	P1	P2	P3	P4	P5	P6
P1	0	0.2357	0.2218	0.3688	0.3421	0.2347
P2	0.2357	0	0.1483	0.2042	0.1388	0.2540
P3	0.2218	0.1483	0	0.1513	0.2843	0.1100
P4	0.3688	0.2042	0.1513	0	0.2932	0.2216
P5	0.3421	0.1388	0.2843	0.2932	0	0.3921
P6	0.2347	0.2540	0.1100	0.2216	0.3921	0

	P1	P2	P3、6	P4	P5
P1	0	0.2357	0.2347	0.3688	0.3421
P2	0.2357	0	0.2540	0.2042	0.1388
P3、6	0.2347	0.2540	0	0.2216	0.3921
P4	0.3688	0.2042	0.2216	0	0.2932
P5	0.3421	0.1388	0.3921	0.2932	0

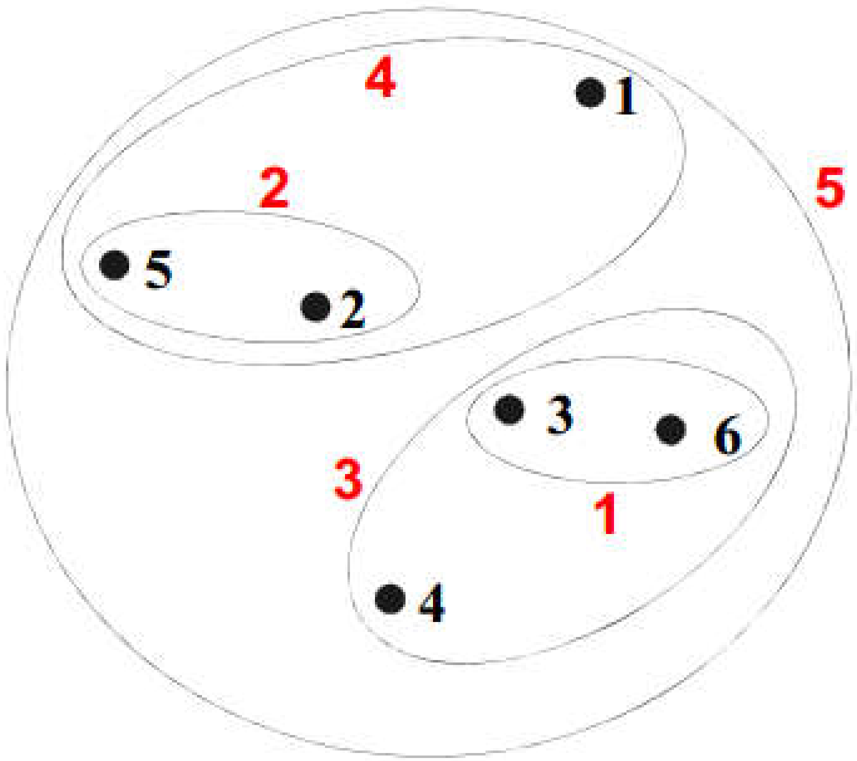
	P1	P2、5	P3、6	P4
P1	0	0.3421	0.2347	0.3688
P2、5	0.3421	0	0.3921	0.2932
P3、6	0.2347	0.3921	0	0.2216
P4	0.3688	0.2932	0.2216	0

dist({1},{3,6})=max(dist(1,3),dist(1,6))
=max(0.2218,0.2347)=0.2347

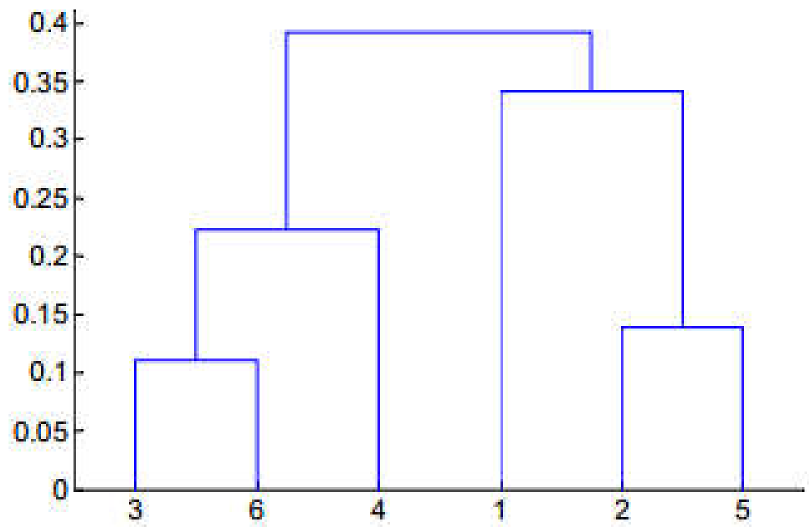
	P1	P2、5	P3、4、6
P1	0	0.3421	0.3688
P2、5	0.3421	0	0.3921
P3、4、6	0.3688	0.3921	0

	P1、2、5	P3、4、6
P1、2、5	0	0.3921
P3、4、6	0.3921	0

举例



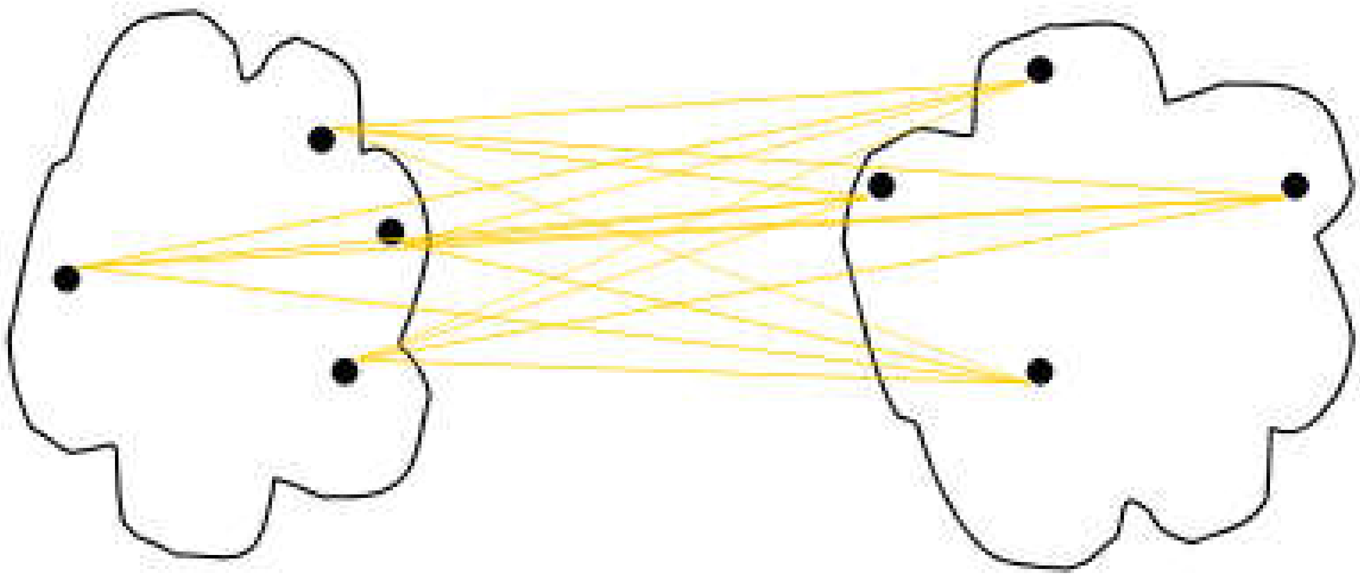
Nested Clusters



Dendrogram

GROUP AVERAGE

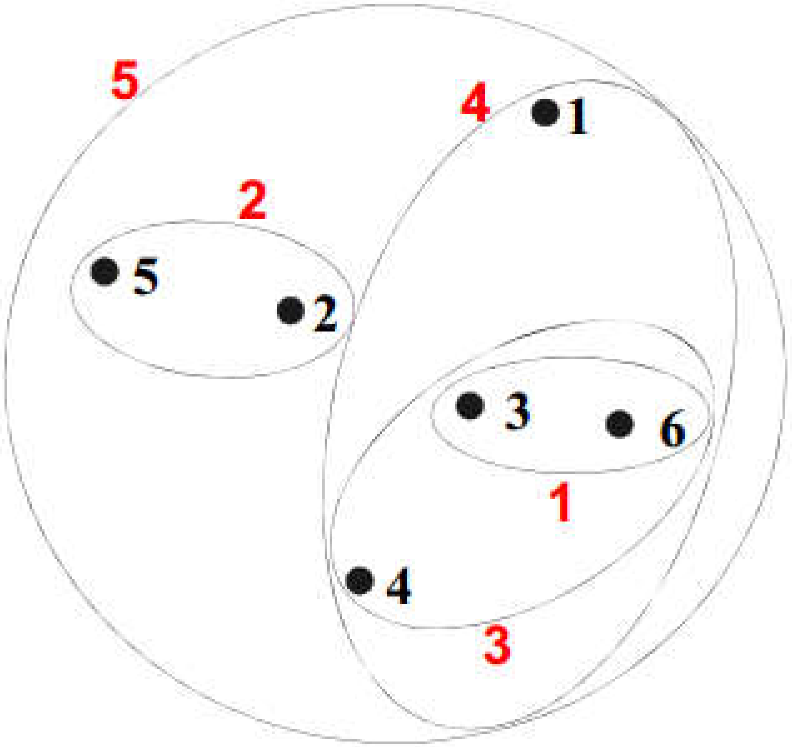
- 取不同簇的所有点对邻近度的平均值作为簇的邻近度



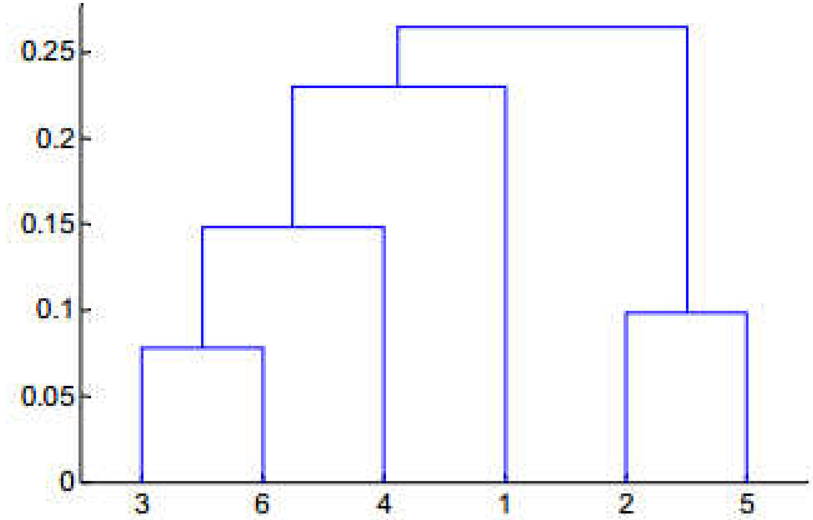
举例

Dist({3,6,4},{1})=(0.2218+0.2347+0.3688)/(3*1)=0.28

Dist({3,6,4},{2,5})=(0.1483+0.2843+0.2541+0.3921+0.2042+0.2932)/(3*2)=0.26



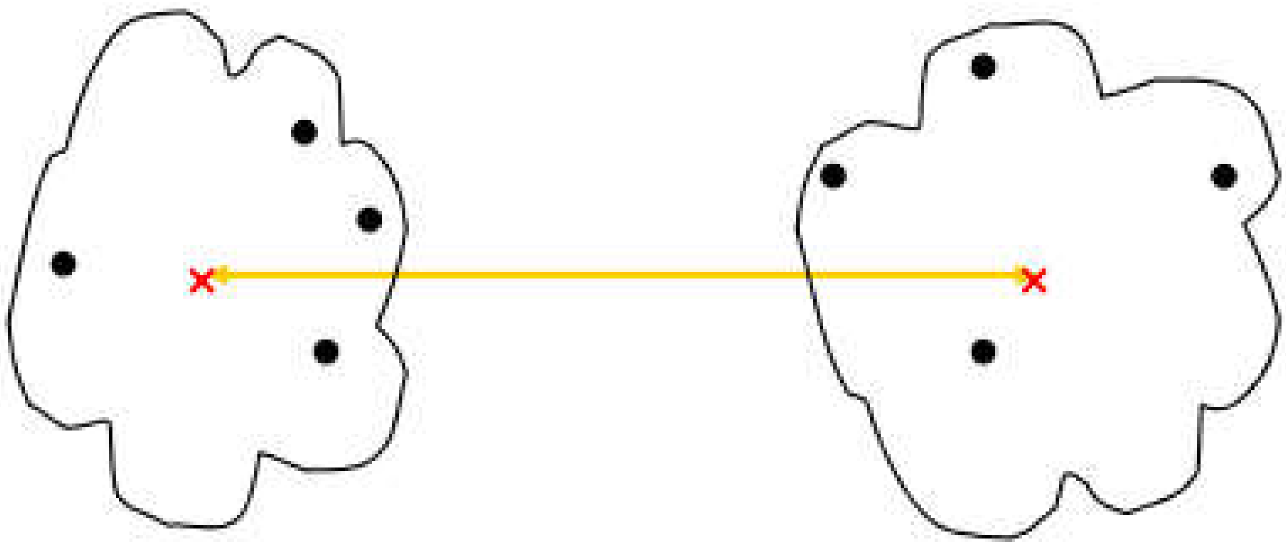
Nested Clusters



Dendrogram

DISTANCE BETWEEN CENTROIDS

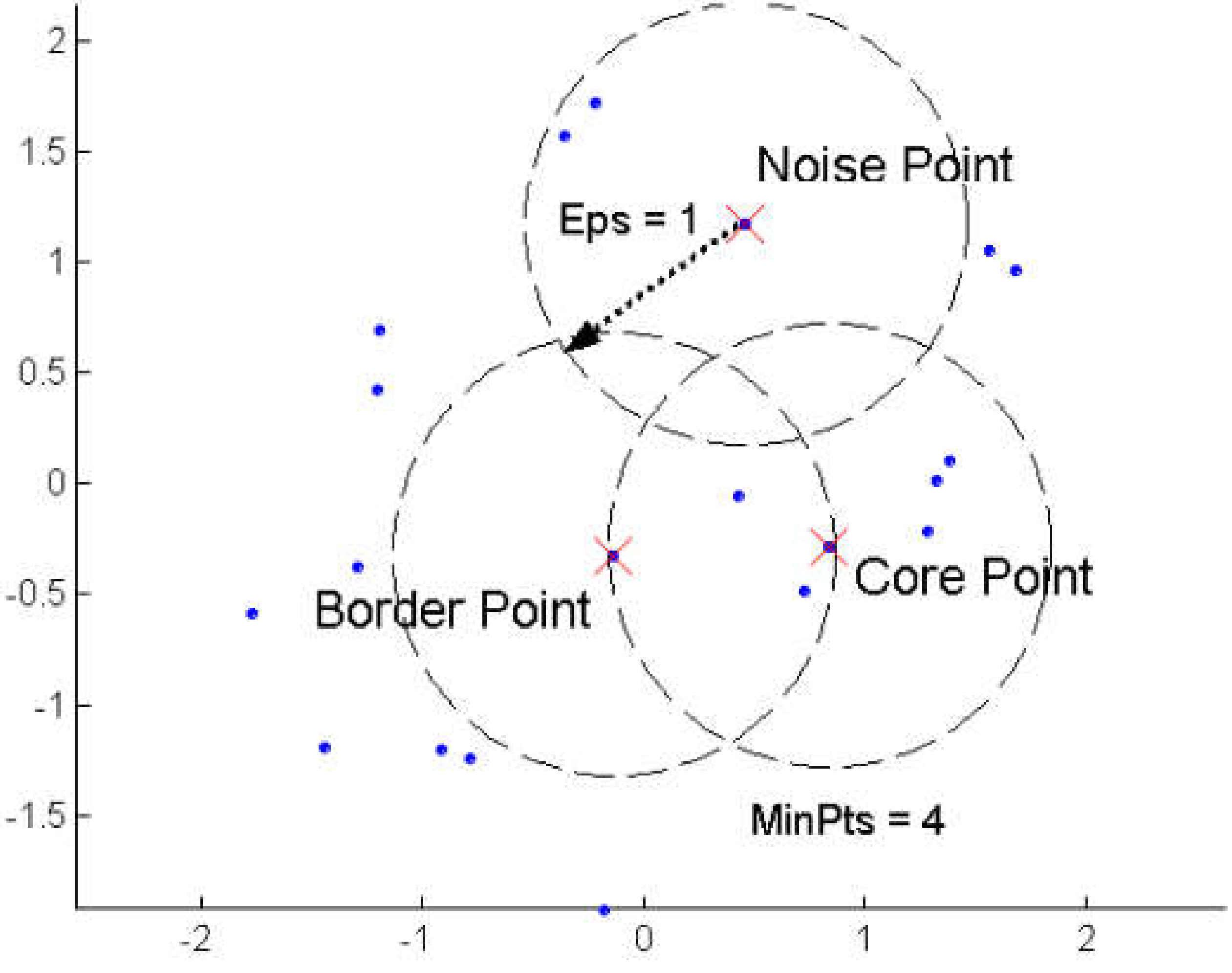
- 取不同簇的质心之间的距离作为簇的邻近度



DBSCAN

- DBSCAN is a density-based algorithm.
  - Density = number of points within a specified radius (Eps)
  - A point is a core point if it has more than a specified number of points (MinPts) within Eps
    - These are points that are at the interior of a cluster
  - A border point has fewer than MinPts within Eps, but is in the neighborhood of a core point
  - A noise point is any point that is not a core point or a border point.

DBSCAN


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

DBSCAN算法

1、将所有点标记为核心点、边界点或噪声点

2、删除噪声点

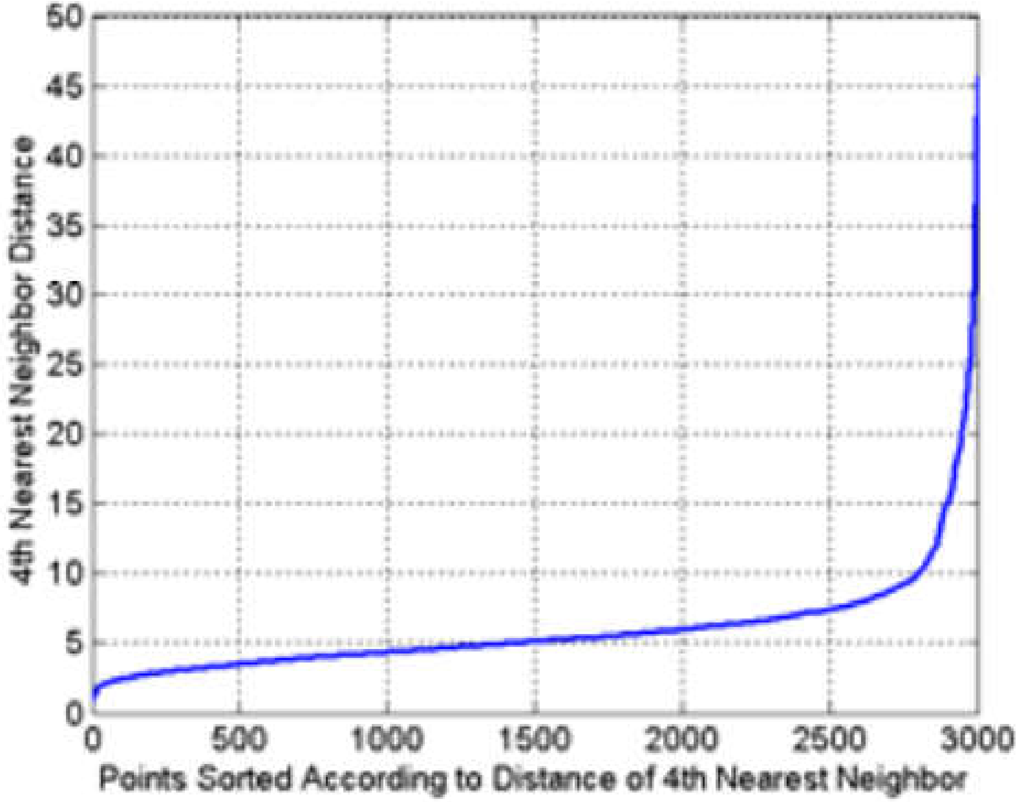
3、为距离在Eps之内的所有核心点之间赋予一条边

4、每组连通的核心点形成一个簇

5、将每个边界点指派到一个与之关联的核心点的簇中

举例

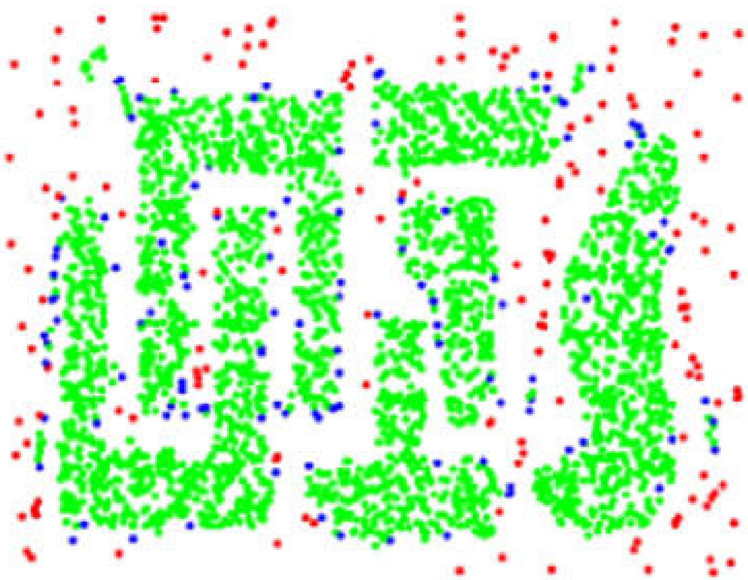
Original Points



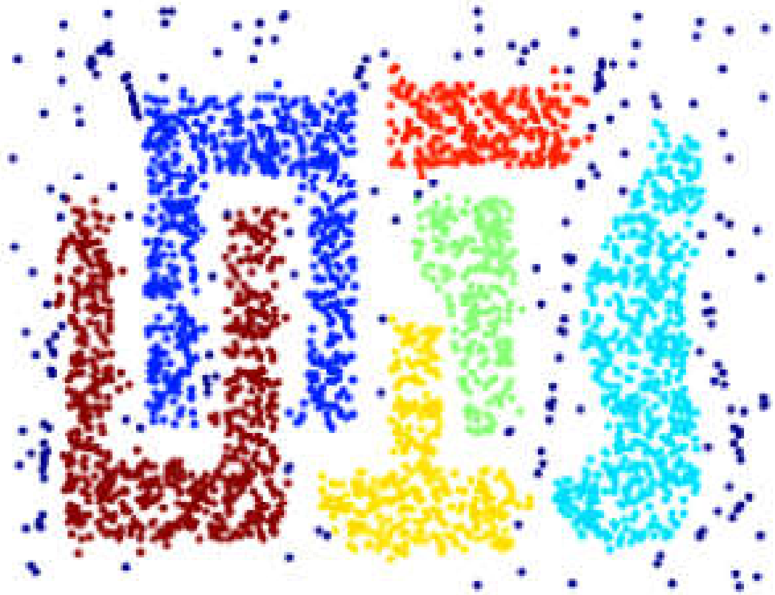
Eps = 10, MinPts = 4

举例（续）

- Resistant to Noise
- Can handle clusters of different shapes and sizes



Point types: core, border and noise



Clusters