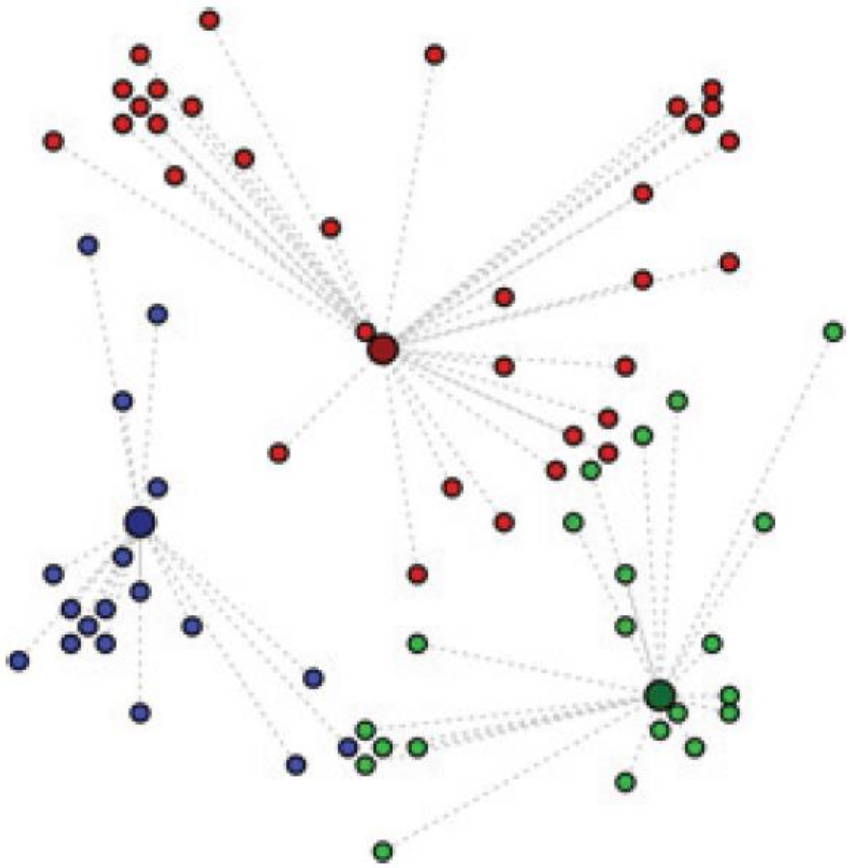
第9章 聚类

聚类（无监督）

- 聚类是指通过无监督技术对相似的数据对象进行分组的过程
  - 这里组也称之为（类）簇
- 聚类是一种经常使用的探索分析数据的方法，是更深入的分析或决策过程的前奏



K=3时的k均值聚类结果

聚类问题的描述

聚类问题：根据给定的数据集，

$$T=\{x_i \mid x_i \in \mathcal{X}, i=1,\cdots,N\}$$

要求寻找 T 上的一个“好”的划分 $C_1,\cdots,C_m$ (划分成 m 个类； m 可以是已知的，也可以是未知的），满足约束条件：

(1) $T=\bigcup_{i=1}^{m} C_i$;

(2) $C_i \neq \varnothing, i=1,\cdots,m$;

(3) $C_i \bigcap C_j = \varnothing, i \neq j, i, j=1,\cdots,m$.

聚类问题的描述

模糊聚类问题：根据给定的数据集，

$$T=\{x_i \mid x_i \in \mathcal{X}, i=1,\cdots,N\}$$

要求寻找 T 上的一个“好”的模糊划分 $u_1,\cdots,u_m$ (划分成m个模糊集），满足约束条件：

(1) $\sum_{j=1}^{m} u_j(x_i)=1, i=1,\cdots,N;$ (每个样本属于m个类的隶属度之和为1）

(2) $0<\sum_{i=1}^{N} u_j(x_i)<N, j=1,\cdots,m;$（每个类不为空集）

这里$u_j:\mathcal{X}\to[0,1]$表示$\mathcal{X}$上的一个模糊集

- 模糊聚类问题可以看成是前面聚类问题（**硬聚类**）的一个推广，**当u_j的值域限制为{0,1}时，模糊聚类就是硬聚类.**

样本间的接近性度量

- 差异性度量（Dissimilarity Measure, DM）
  - 对称性
  - 自己与自己的差异性最小

  例子：距离差异性度量

- 相似性度量（Similarity Measure，SM）
  - 对称性
  - 自己与自己的相似性最大

常用的接近性度量

- 点与点之间
- 点与集合之间
- 集合与集合之间

点与点之间：差异性度量

$$d_p(x,y)=\left(\sum_{i=1}^{l} w_i \mid x_i-y_i \mid^p\right)^{1/p} \quad (0<p<\infty, w_i\geq 0) \quad (DM)$$

$$d_2(x,y)=\left(\sum_{i=1}^{l} w_i \mid x_i-y_i \mid^2\right)^{1/2}$$，进一步推广为$d_2(x,y)=\sqrt{(x-y)^T B(x-y)}$

马氏距离

$$d_1(x,y)=\sum_{i=1}^{l} w_i \mid x_i-y_i \mid$$

$$d_\infty(x,y)=\max_{1\leq i\leq l} w_i \mid x_i-y_i \mid$$

点与点之间：相似性度量

$$s_{inner}(x,y) = x^T y \quad (\textit{The inner product measue, generally } x, y \textit{ are normaized})$$

$$s_T = \frac{x^T y}{x^T x + y^T y - x^T y} = \frac{1}{1 + \frac{(x-y)^T (x-y)}{x^T y}}$$

(*Tanimoto measure*)

谷本系数

$$s_c(x,y) = 1 - \frac{d_2(x,y)}{\| x \| + \| y \|}$$

$$s_g(x,y) = \exp\{-\frac{\| x-y \|^2}{\sigma^2}\}$$

点与集合之间

$$The\ \max\ proximity\ function: p(x,C) = \max_{y \in C}\ p(x,y)$$

$$The\ \min\ proximity\ function: p(x,C) = \min_{y \in C}\ p(x,y)$$

$$The\ average\ proximity\ function: p(x,C) = \frac{1}{|C|}\sum_{y \in C} p(x,y)$$

$$d(x,H) = \min_{y \in H} d(x,y),\ where\ hyperplane\ H: a^T x + b = 0$$

$$d(x,Q) = \min_{y \in Q} d(x,y),\ where\ hypersphere\ Q: (x-c)^T (x-c) = r^2$$

集合与集合之间

The max *proximity function* : $p(B,C) = \max_{x \in B, y \in C} p(x,y)$

The min *proximity function* : $p(B,C) = \min_{x \in B, y \in C} p(x,y)$

The average proximity function : $p(B,C) = \frac{1}{|B| \times |C|} \sum_{x \in B, y \in C} p(x,y)$

The mean proximity function : $p(B,C) = p(m_B, m_C)$

$$p(B,C) = \sqrt{\frac{|B| \times |C|}{|B| + |C|}} p(m_B, m_C)$$

离差平方和法：

$$d(B,C) = S(B \cup C) - S(B) - S(C)$$

这里$S(G)$是数据集G的方差

聚类评价准则

- 类簇内部样本间的接近度尽可能地大，而不同类簇样本间的接近度尽可能地小

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x}_j \in S_i} \|\mathbf{x}_j - \boldsymbol{\mu}_i\|^2$$

枚举聚类不可行！

- N个样本聚为m类的可能聚类数S(N,m):

S(N,1)=1;S(N,N)=1;S(N,m)=0,for m>N

S(N,m)=mS(N-1,m)+S(N-1,m-1)

$$\Rightarrow S(N,m)=\frac{1}{m!}\sum_{i=0}^{m}(-1)^{m-i}C_m^i i^N$$

- S(15,3)=2375101;S(20,4)=45232115901
- S(25,8)=690223721118368580;S(100,5) ≈10^{68}

- **枚举聚类是行不通的！**

常见聚类算法简介

1. 顺序聚类算法
2. 划分聚类算法（K-means聚类算法）
3. 层次聚类算法（ AGNES 、 DIANA ）
4. 密度聚类算法（ DBSCAN ）

常见聚类算法简介

1. 顺序聚类算法
2. 划分聚类算法（K-means聚类算法）
3. 层次聚类算法（ AGNES 、 DIANA ）
4. 密度聚类算法（ DBSCAN ）

顺序聚类

增量式聚类

- 最基本的顺序聚类算法

（1）第1个样本归为第1类；

（2）计算下一个样本到已有类的最短距离，若其距离小于给定的域值，则将该样本归为其对应的类，否则增加一个新类，并将该样本归为新类。

（3）重复（2），直到所有样本都被归类。

- 特点
  - 聚类结果与**样本的顺序**和**给定的域值**有关；
  - 聚类速度快，不需要事前给定类簇的个数。

常见聚类算法简介

1. 顺序聚类算法
2. 划分聚类算法（K-means聚类算法）
3. 层次聚类算法（ AGNES 、 DIANA ）
4. 密度聚类算法（ DBSCAN ）

K-means聚类算法

- k-means是划分方法中最经典的聚类算法之一。由于该算法的效率高，所以在对大规模数据进行聚类时被广泛应用。目前，许多算法均围绕着该算法进行扩展和改进。
- k-means算法以k为参数，把n个对象分成k个簇，使簇内具有较高的相似度，而簇间的相似度较低。
- 假设我们提取到原始数据的集合为D($\mathbf{x}_1$, $\mathbf{x}_2$, ..., $\mathbf{x}_n$)，并且每个**x**i为d维的向量，K-means聚类的目的就是，在给定分类组数k（$k \le n$）值的条件下，将原始数据分成k类，**S** = {S_1, S_2, ..., S_k}，在数值模型上，即对以下表达式求最小值：

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x}_j \in S_i} \|\mathbf{x}_j - \boldsymbol{\mu}_i\|^2$$

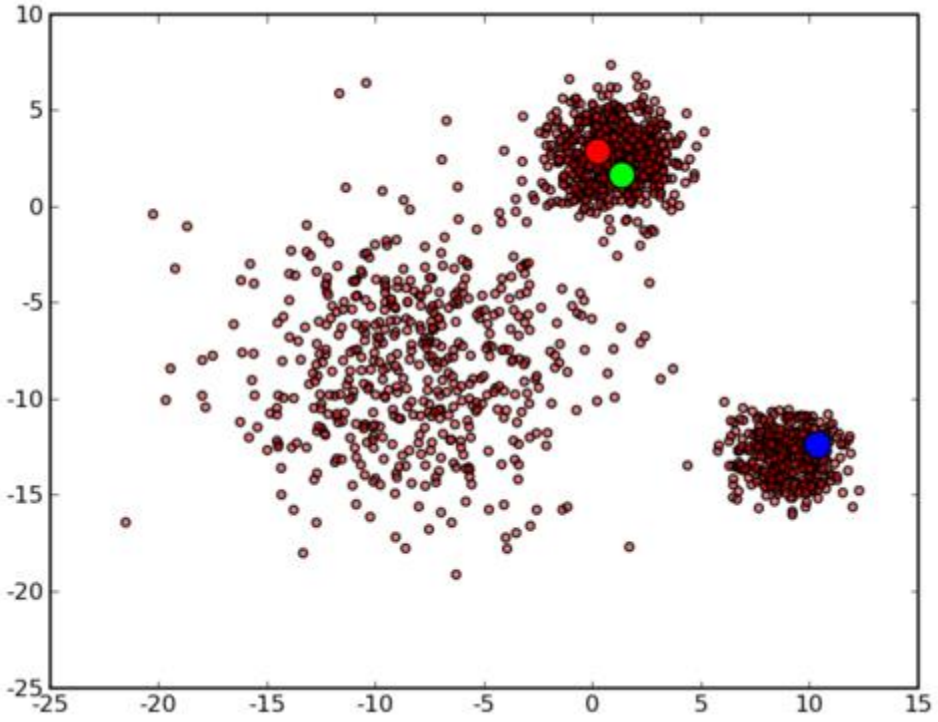
这里$\boldsymbol{\mu}_i$表示分类S_i的平均值。

k-means聚类算法计算机实现步骤

1. 从D中随机取k个元素，作为k个簇的各自的中心。
2. 分别计算剩下的元素到k个簇中心的相异度（即距离），将这些元素分别划归到相异度最低的簇。
3. 根据聚类结果，重新计算k个簇各自的中心，计算方法是取簇中所有元素各自维度的算术平均数。
4. 将D中全部元素按照新的中心重新聚类。
5. 重复第3步和第4步，直到聚类结果不再变化。
6. 输出结果。

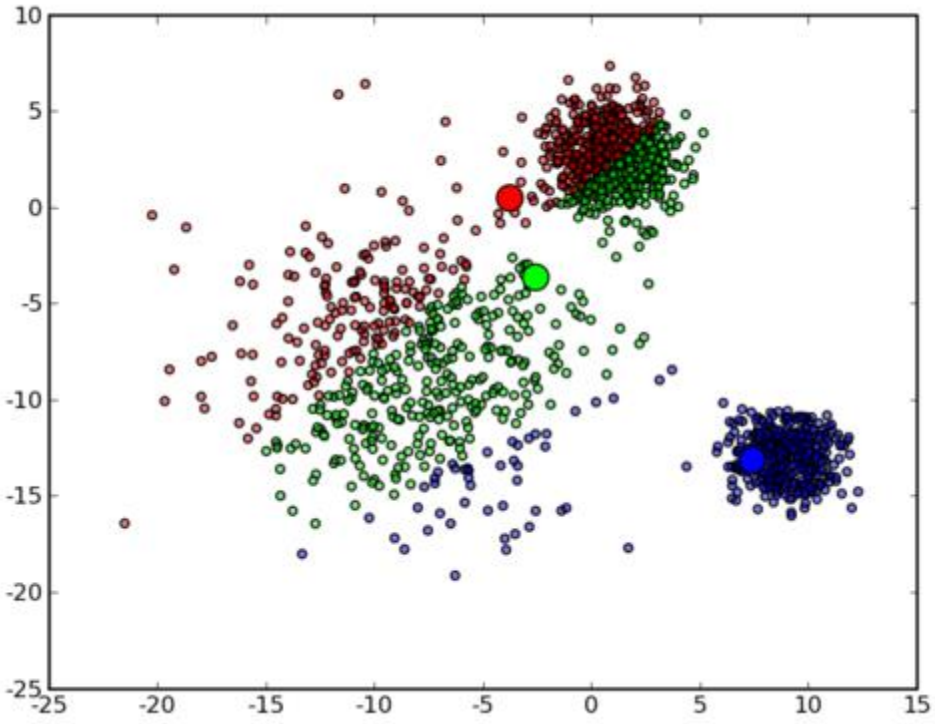
k-means聚类算法示例

对于一个数据集合D，假设K=3，首先 3 个中心点被随机初始化，所有的数据点都还没有进行聚类，默认全部都标记为红色，如下图所示：



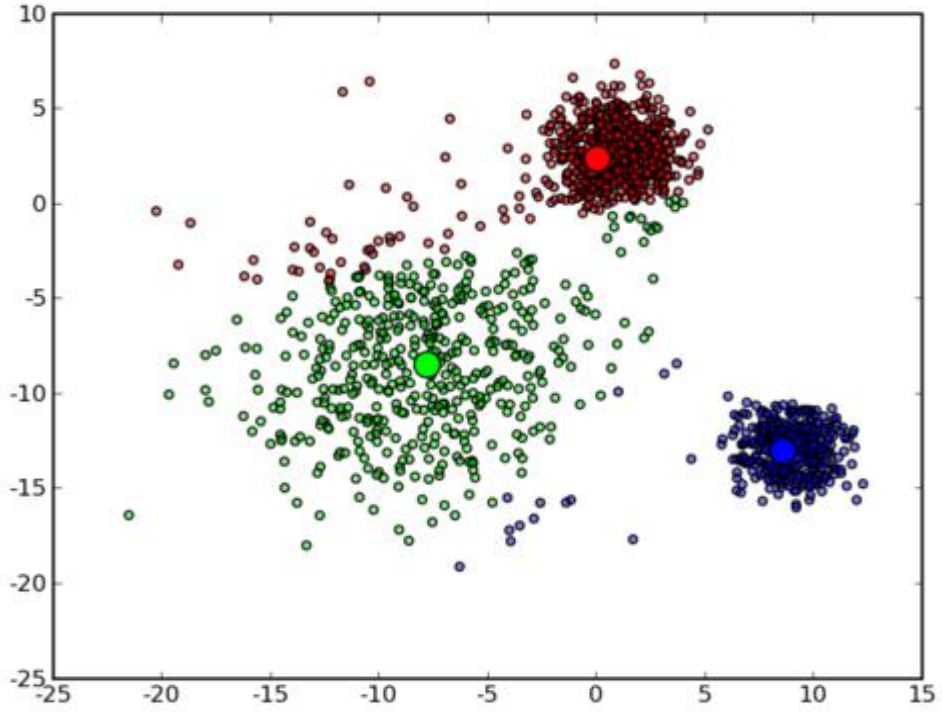
k-means聚类算法示例

然后进入第一次迭代：按照初始的中心点位置为每个数据点着上颜色，重新计算 3 个中心点，结果如下图所示：



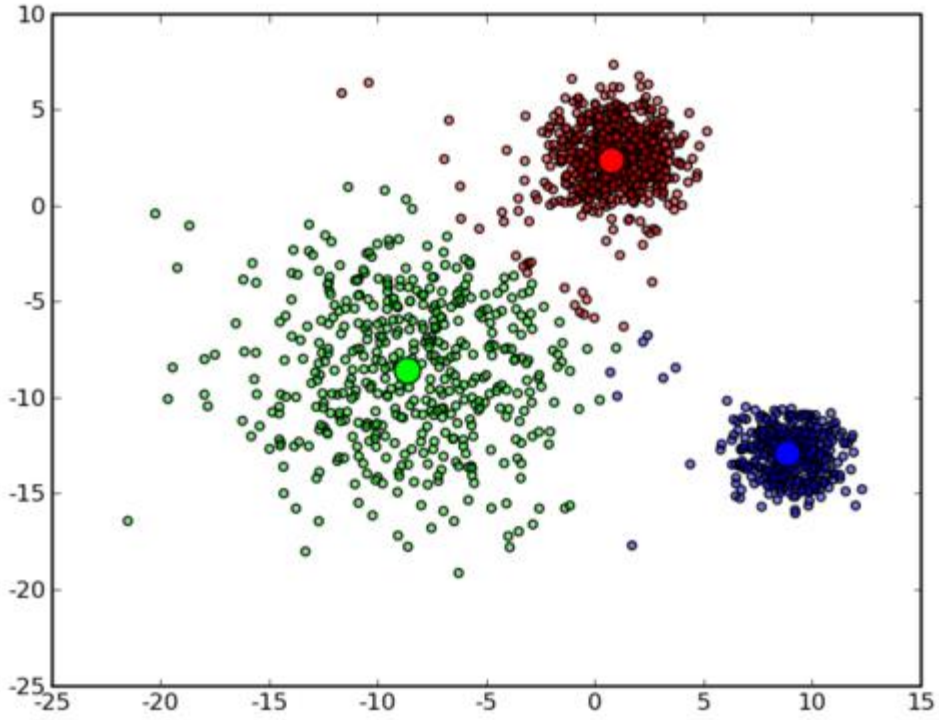
k-means聚类算法示例

可以看到，由于初始的中心点是随机选的，这样得出来的结果并不是很好，接下来是下一次迭代的结果：



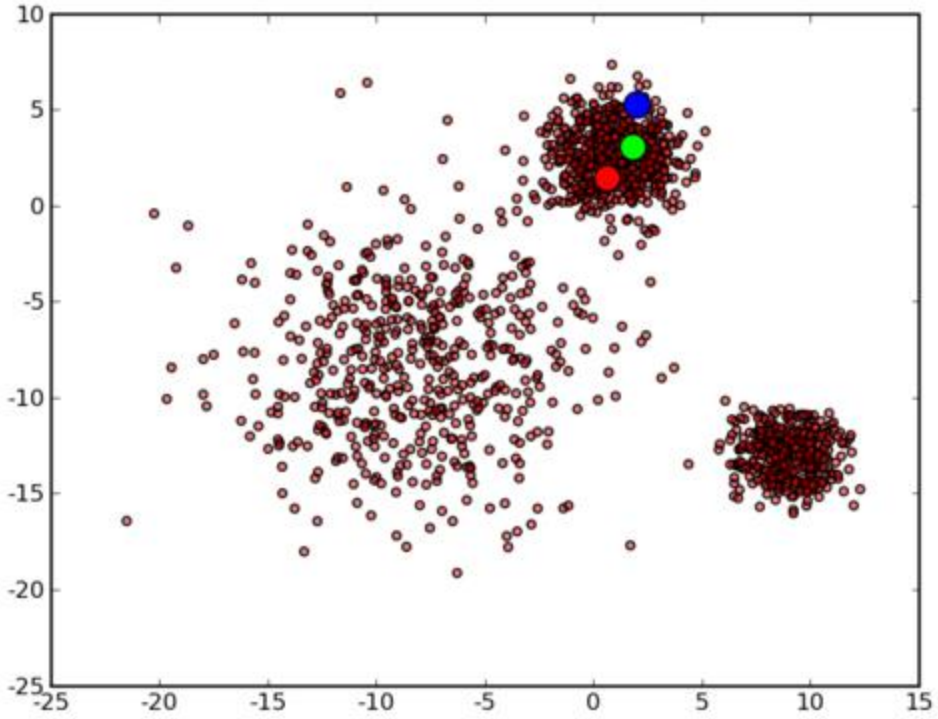
k-means聚类算法示例

可以看到大致形状已经出来了。再经过两次迭代之后，基本上就收敛了，最终结果如下：



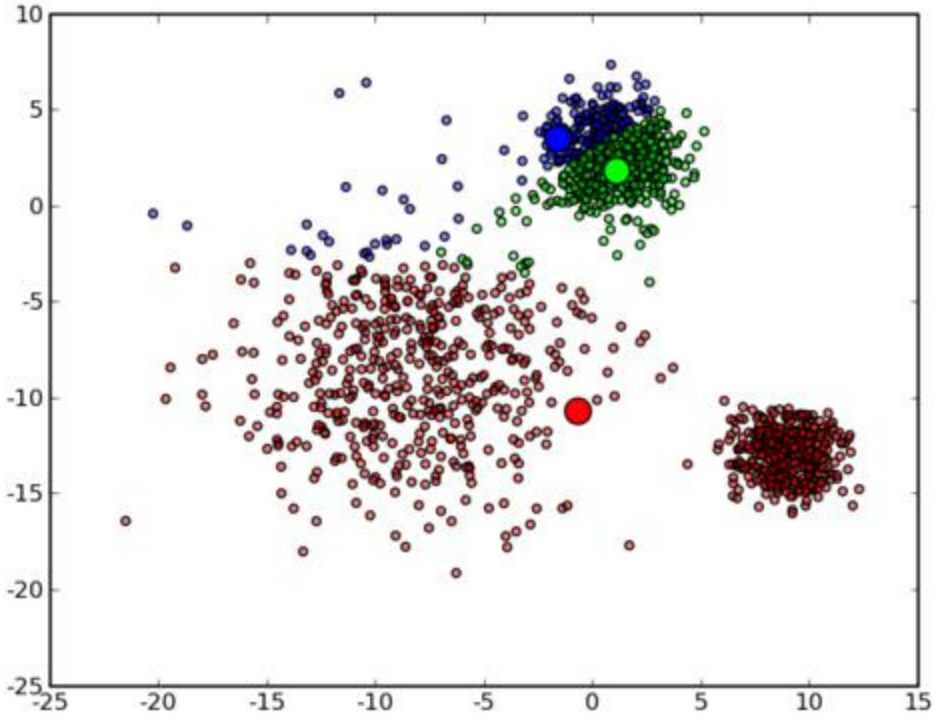
k-means聚类算法示例

但k-means 并不是万能的，虽然许多时候都能收敛到一个比较好的结果，但是也有运气不好的时候会收敛到一个让人不满意的局部最优解，例如选用下面这几个初始中心点：



k-means聚类算法示例

最终会收敛到这样的结果：



k-means聚类算法优缺点

优点：

1. 算法简洁、易于执行
2. 时间复杂度近于线性，对大数据集有较高的效率

缺点：

1. K-means 算法中 K 是事先给定的，这个 K 值的选定是非常难以估计，很多时候，事先并不知道数据集应该分成多少个类别才最合适。
2. K-means 算法中，需要根据初始聚类中心来确定一个初始划分，然后对初始划分进行优化。这个初始聚类中心的选择对聚类结果有较大的影响，一旦初始值选择的不好，可能无法得到有效的聚类结果。
3. 不适合于发现非凸面形状的簇或者大小差别很大的簇。而且，它对于“躁声”和孤立点数据是敏感的。

常见聚类算法简介

1. 顺序聚类算法
2. 划分聚类算法（K-means聚类算法）
3. 层次聚类算法（ AGNES 、 DIANA ）
4. 密度聚类算法（ DBSCAN ）

层次聚类

当采用划分聚类方法（如k-means）K值选取十分困难时，我们不妨考虑可以考虑层次聚类。层次聚类是另一种主要的聚类方法，它具有一些十分必要的特性使得它成为广泛应用的聚类方法。它生成一系列嵌套的聚类树来完成聚类。单点聚类处在树的最底层，在树的顶层有一个根节点聚类。根节点聚类覆盖了全部的所有数据点。

可根据其聚类方式划分为：**凝聚（自下而上）聚类**和**分裂（自上而下）聚类**。层次凝聚的代表是AGNES算法，而层次分裂的代表是DIANA算法。

接下来，我们以AGNES算法为例进行介绍。

AGNES算法

AGNES (AGglomerative NESting)算法最初将每个对象作为一个簇，然后这些簇根据某些准则被一步步地合并。**两个簇间的相似度由这两个不同簇中距离最近的数据点对的相似度来确定**。聚类的合并过程反复进行直到所有的对象最终满足类簇的要求数目。

AGNES（自底向上凝聚算法）计算机编程实现：

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初始簇；

(2) REPEAT

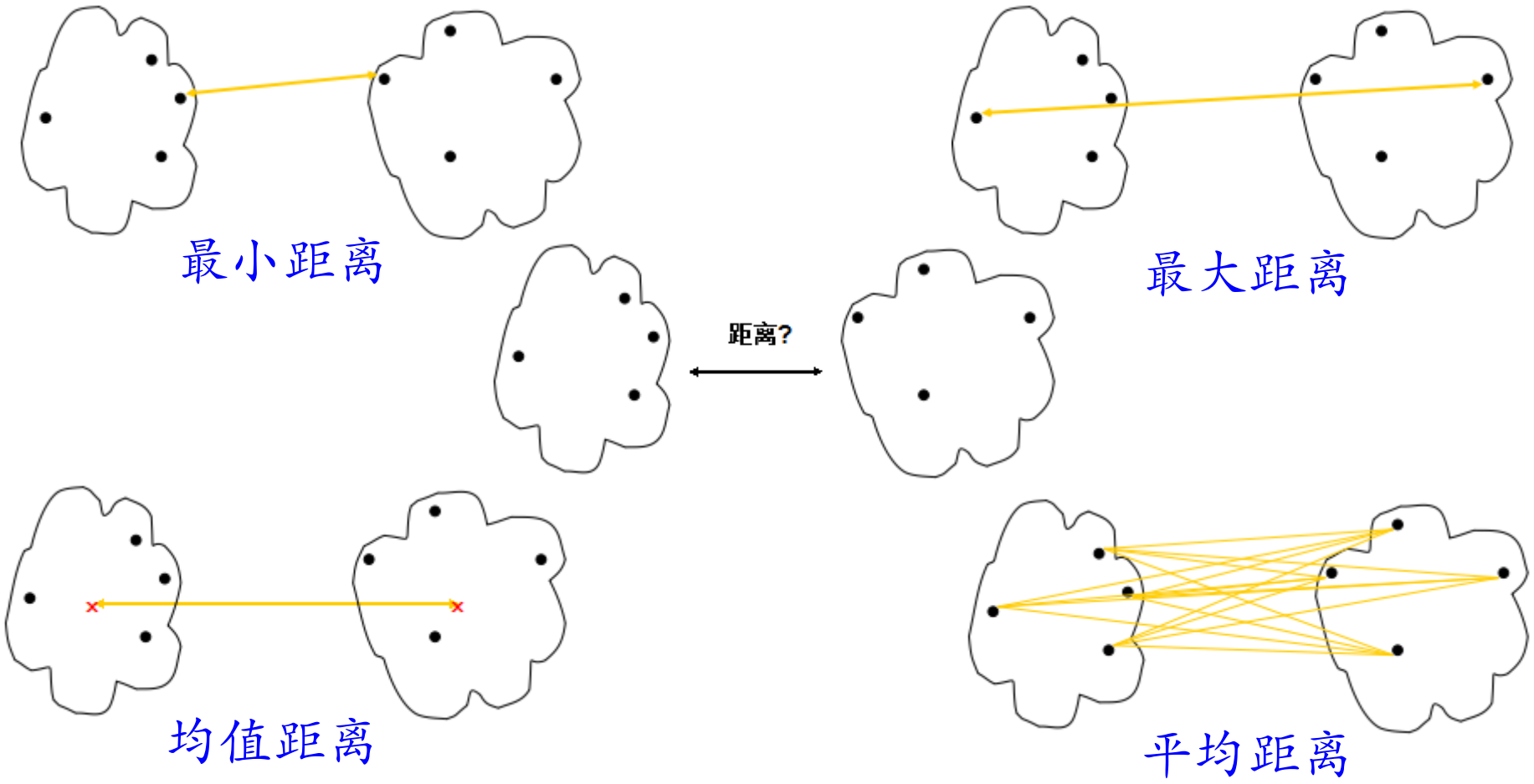
(3) 根据两个簇中最近的数据点找到最近的两个簇；

(4) 合并两个簇，生成新的簇的集合；

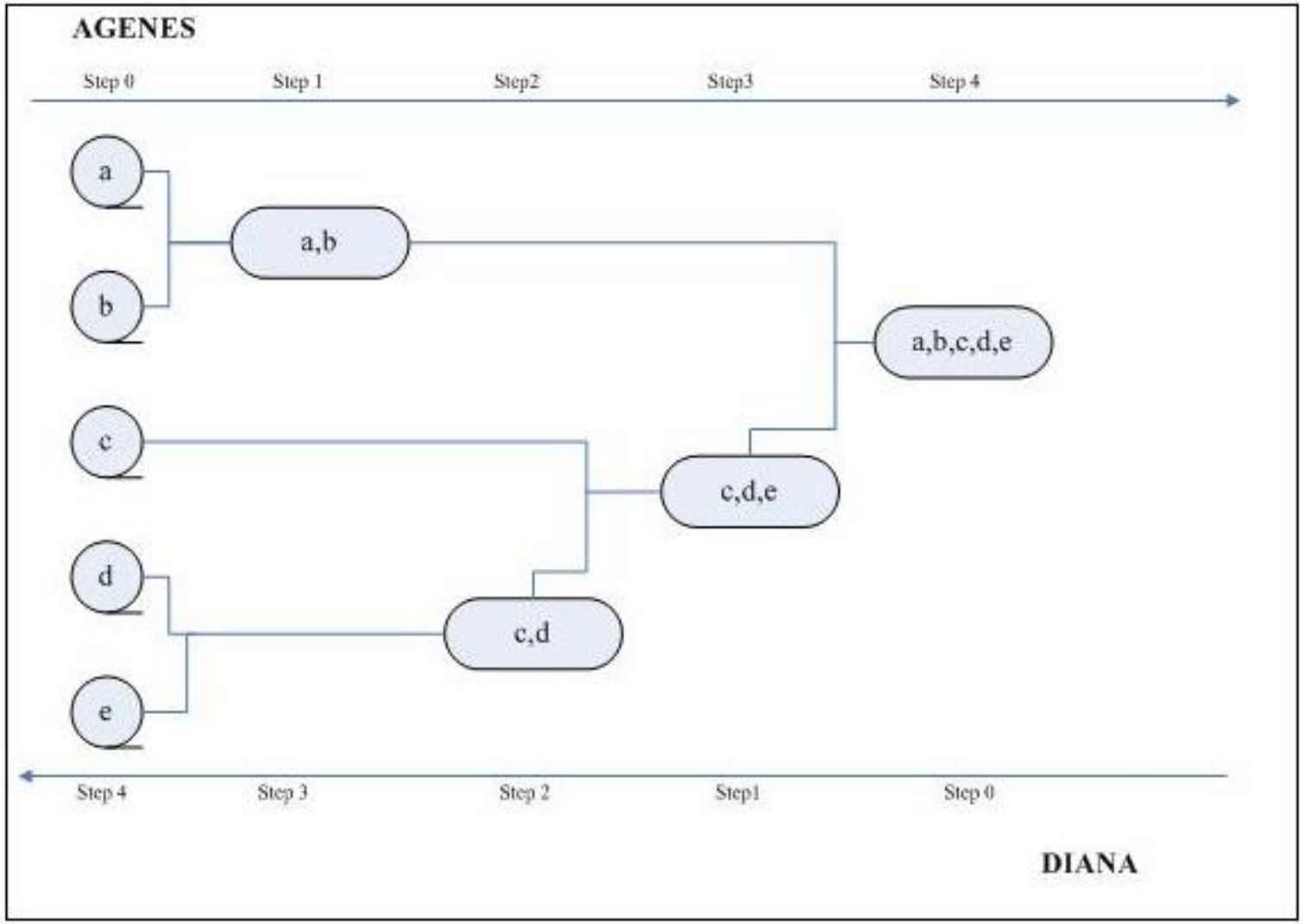
(5) UNTIL 达到定义的簇的数目。

判断两个类之间相似度的方法

1. *SingleLinkage*：又叫做 nearest-neighbor，就是取两个类中距离最近的两个样本的距离作为这两个集合的距离，也就是说，最近两个样本之间的距离越小，这两个类之间的相似度就越大。容易造成一种叫做 Chaining 的效果，两个 cluster 明明从“大局”上离得比较远，但是由于其中个别的点距离比较近就被合并了，并且这样合并之后Chaining效应会进一步扩大，最后会得到比较松散的cluster 。
2. *CompleteLinkage*：这个则完全是Single Linkage的反面极端，取两个集合中距离最远的两个点的距离作为两个集合的距离。其效果也是刚好相反的，限制非常大，两个cluster即使已经很接近了，但是只要有不配合的点存在，就顽固到底，老死不相合并，也是不太好的办法。这两种相似度的定义方法的共同问题就是指考虑了某个有特点的 数据，而没有考虑类内数据的整体特点
3. *Average-linkage*：这种方法就是把两个集合中的点两两的距离全部放在一起求一个平均值，相对也能得到合适一点的结果。*average-linkage* 的一个变种就是取两两距离的中值，与取均值相比更加能够解除个别偏离样本对结果的干扰。


最小距离
最大距离
距离?
均值距离
平均距离

层次聚类：凝聚


AGENES
Step 0
Step 1
Step2
Step3
Step 4
a
b
a,b
a,b,c,d,e
c
c,d,e
d
c,d
e
Step 4
Step 3
Step 2
Step1
Step 0
DIANA

AGNES算法示例（$k = 2$）

序号	属性 1	属性 2
1	1	1
2	1	2
3	2	1
4	2	2
5	3	4
6	3	5
7	4	4
8	4	5

第1步：根据初始簇计算每个簇之间的距离，（随机）找出距离最小的两个簇，进行合并，最小距离为1，合并后1，2点合并为一个簇。

第2步：对上一次合并后的簇计算簇间距离，找出距离最近的两个簇进行合并，合并后3，4点成为一簇。

第3步：重复第2步的工作，5，6点成为一簇。

第4步：重复第2步的工作，7，8点成为一簇。

第5步：合并{1，2}，{3，4}成为一个包含四个点的簇。

第6步：合并{5，6}，{7，8}，由于合并后的簇的数目已经达到了用户输入的终止条件程序结束。

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

层次聚类算法优缺点及改进算法

- 优点：适用于任意形状和任意属性的数据集，灵活控制不同层次的聚类粒度，强聚类能力。
- 缺点：大大延长了算法的执行时间。

层次聚类方法尽管简单，但经常会遇到合并或分裂点选择的困难。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，比方层次聚类方法BIRTH 和CURE。

常见聚类算法简介

1. 顺序聚类算法
2. 划分聚类算法（K-means聚类算法）
3. 层次聚类算法（ AGNES 、 DIANA ）
4. 密度聚类算法（ DBSCAN ）

密度聚类算法

- 密度聚类方法的指导思想是：**只要一个区域中的点的密度大于某个域值，就把它加到与之相近的聚类中去**。这类算法能克服基于距离的算法只能发现“类圆形”的聚类的缺点，可发现任意形状的聚类，且对噪声数据不敏感。但**计算密度单元的计算复杂度大**，需要建立空间索引来降低计算量，且对数据维数的伸缩性较差。这类方法需要扫描整个数据库，每个数据对象都可能引起一次查询，因此当数据量大时会造成频繁的I/O操作。代表算法有：**DBSCAN**、OPTICS、DENCLUE算法等。

- DBSCAN（Density-Based Spatial Clustering of Applications with Noise）一个比较有代表性的**基于密度的聚类算法**。与划分和层次聚类方法不同，它将簇定义为密度相连的点的最大集合，能够把具有足够高密度的区域划分为类簇，并可在有“噪声”的空间数据库中发现任意形状的聚类。

密度聚类算法

- 对象的ε-临域：给定对象在半径ε内的区域。
- 核心对象：如果一个对象的ε-临域至少包含最小数目MinPts个对象，则称该对象为核心对象。
- 例如，在下图中，ε=1cm，MinPts=5，q是一个核心对象。
- 直接密度可达：给定一个对象集合D，如果p是在q的ε-邻域内，而q是一个核心对象，我们说对象p从对象q出发是直接密度可达的。
- 例如，在图1中，ε=1cm，MinPts=5，q是一个核心对象，对象p从对象q出发是直接密度可达的。

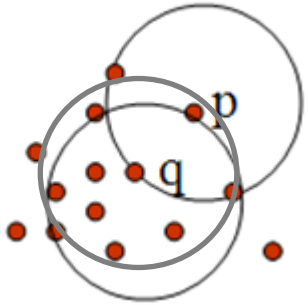


图1 核心对象

密度聚类算法

- 密度可达：如果存在一个对象链$p_1, p_2, \ldots p_n, p_1 = q, p_n = p$，对$p_i \in$D，（1<=i<=n），pi+1是从pi关于ε和MitPts直接密度可达的，则对象p是从对象q关于ε和MinPts密度可达的。
- 例如，在图2中，ε=1cm，MinPts=5，q是一个核心对象，p1是从q关于ε和MitPts直接密度可达，p是从p1关于ε和MitPts直接密度可达，则对象p从对象q关于ε和MinPts密度可达。
- 密度相连：如果对象集合D中存在一个对象o，使得对象p和q是从o关于ε和MinPts密度可达的，那么对象p和q是关于ε和MinPts密度相连的。
- **一个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**。不包含在任何簇中的对象被认为是“噪声”。

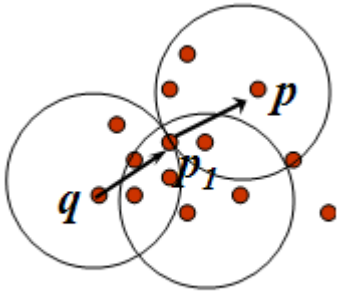


图2 直接密度可达

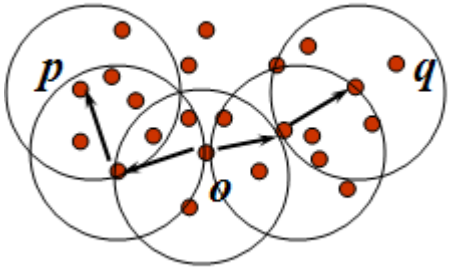


图3 密度相连

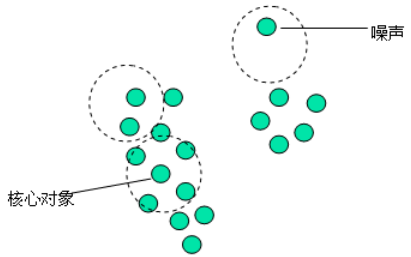


图4 噪声

DBSCAN 聚类算法

DBSCAN通过检查数据集中每个对象的ε-邻域来寻找聚类。如果一个点p的ε-邻域包含多于MinPts个对象，则创建一个p作为核心对象的新簇。然后，DBSCAN反复地寻找从这些核心对象直接密度可达的对象，这个过程可能涉及一些密度可达簇的合并。当没有新的点可以被添加到任何簇时，该过程结束。具体如下：

- 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。

1. REPEAT
2. 从数据库中抽取一个未处理过的点；
3. 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
4. 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；
5. UNTIL 所有点都被处理；

DBSCAN 聚类算法示例

下面给出一个样本事务数据库（见左表），对它实施DBSCAN算法。 根据所给的数据通过对其进行DBSCAN算法，以下为算法的步骤（设n=12，用户输入 ε=1，MinPts=4）

样本事务数据库

序号	属性1	属性2
1	1	0
2	4	0
3	0	1
4	1	1
5	2	1
6	3	1
7	4	1
8	5	1
9	0	2
10	1	2
11	4	2
12	1	3

DBSCAN算法执行过程示意

步骤	选择的点	在ε中点的个数	通过计算可达点而找到的新簇
1	1	2	无
2	2	2	无
3	3	3	无
4	4	5	簇C_1：{1，3，4，5，9，10，12}
5	5	3	已在一个簇C_1中
6	6	3	无
7	7	4	簇C_2：{2，6，7，8，11}
8	8	2	已在一个簇C_2中
9	9	3	已在一个簇C_1中
10	10	4	已在一个簇C_1中，
11	11	2	已在一个簇C_2中
12	12	2	已在一个簇C_1中

聚出的类为{1，3，4，5，9，11，12}，{2，6，7，8，10}。

DBSCAN 聚类算法示例

步骤	选择的点	在ε中点的个数	通过计算可达点而找到的新簇
1	1	2	无
2	2	2	无
3	3	3	无
4	4	5	簇C_1: {1, 3, 4, 5, 9, 10, 12}
5	5	3	已在一个簇C_1中
6	6	3	无
7	7	4	簇C_2: {2, 6, 7, 8, 11}
8	8	2	已在一个簇C_2中
9	9	3	已在一个簇C_1中
10	10	4	已在一个簇C_1中,
11	11	2	已在一个簇C_2中
12	12	2	已在一个簇C_1中

第1步，在数据库中选择一点1，由于在以它为圆心的，以1为半径的圆内包含2个点（小于4），因此它不是核心点，选择下一个点。
第2步，在数据库中选择一点2，由于在以它为圆心的，以1为半径的圆内包含2个点，因此它不是核心点，选择下一个点。
第3步，在数据库中选择一点3，由于在以它为圆心的，以1为半径的圆内包含3个点，因此它不是核心点，选择下一个点。
第4步，在数据库中选择一点4，由于在以它为圆心的，以1为半径的圆内包含5个点，因此它是核心点，寻找从它出发可达的点（直接可达4个，间接可达3个），聚出的新类{1，3，4，5，9，10，12}，选择下一个点。
第5步，在数据库中选择一点5，已经在簇1中，选择下一个点。
第6步，在数据库中选择一点6，由于在以它为圆心的，以1为半径的圆内包含3个点，因此它不是核心点，选择下一个点。
第7步，在数据库中选择一点7，由于在以它为圆心的，以1为半径的圆内包含4个点，因此它是核心点，寻找从它出发可达的点，聚出的新类{2，6，7，8，11}，选择下一个点。
第8步，在数据库中选择一点8，已经在簇2中，选择下一个点。
第9步，在数据库中选择一点9，已经在簇1中，选择下一个点。
第10步，在数据库中选择一点10，已经在簇1中，选择下一个点。
第11步，在数据库中选择一点11，已经在簇2中，选择下一个点。
第12步，选择12点，已经在簇1中，由于这已经是最后一点所有点都以处理，程序终止。

密度聚类算法优缺点

优点：

（1）与K-MEANS比较起来，不需要输入要划分的聚类个数

（2）聚类簇的形状没有偏倚；

（3）可以在需要时输入过滤噪声的参数；

缺点：

（1）当数据量增大时，要求较大的内存支持I/O消耗也很大

（2）当空间聚类的密度不均匀、聚类间距差相差很大时，聚类质量较差。

聚类算法小结

对聚类进行研究是数据挖掘中的一个热门方向，由于以上所介绍的聚类方法都存在着某些缺点，因此近些年对于聚类分析的研究很多都专注于改进现有的聚类方法或者是提出一种新的聚类方法。以下将对传统聚类方法中存在的问题以及人们在这些问题上所做的努力做一个简单的总结：

1. 从以上对传统的聚类分析方法所做的总结来看，不管是k-means方法，还是AGNES方法，在进行聚类之前都需要用户事先确定要得到的聚类的数目。然而在现实数据中，聚类的数目是未知的，通常要经过不断的实验来获得合适的聚类数目，得到较好的聚类结果。

聚类算法小结

2. 传统的聚类方法一般都是适合于某种情况的聚类，没有一种方法能够满足各种情况下的聚类，比如BIRCH方法对于球状簇有很好的聚类性能，但是对于不规则的聚类，则不能很好的工作；K-medoids方法不太受孤立点的影响，但是其计算代价又很大。因此如何解决这个问题成为当前的一个研究热点，有学者提出将不同的聚类思想进行融合以形成新的聚类算法，从而综合利用不同聚类算法的优点，在一次聚类过程中综合利用多种聚类方法，能够有效的缓解这个问题。
3. 随着信息时代的到来，对大量的数据进行分析处理是一个很庞大的工作，这就关系到一个计算效率的问题。有文献提出了一种基于最小生成树的聚类算法，该算法通 过逐渐丢弃最长的边来实现聚类结果，当某条边的长度超过了某个阈值，那么更长边就不需要计算而直接丢弃，这样就极大地提高了计算效率，降低了计算成本。

聚类算法小结

4. 处理大规模数据和高维数据的能力有待于提高。目前许多聚类方法处理小规模数据和低维数据时性能比较好，但是当数据规模增大，维度升高时，性能就会急剧下 降，比如k-medoids方法处理小规模数据时性能很好，但是随着数据量增多，效率就逐渐下降，而现实生活中的数据大部分又都属于规模比较大、维度比较 高的数据集。有文献提出了一种在高维空间挖掘映射聚类的方法PCKA（Projected Clustering based on the K-Means Algorithm），它从多个维度中选择属性相关的维度，去除不相关的维度，沿着相关维度进行聚类，以此对高维数据进行聚类。
5. 目前的许多算法都只是理论上的，经常处于某种假设之下，比如聚类能很好的被分离，没有突出的孤立点等，但是现实数据通常是很复杂的，噪声很大，因此如何有效的消除噪声的影响，提高处理现实数据的能力还有待进一步的提高。