机器学习-第十章 聚类

黄海广 副教授

2021年12月

本章目录

1.无监督学习概述

1.无监督学习方法概述

监督学习和无监督学习的区别

监督学习

在一个典型的监督学习中，训练集有标签y，我们的目标是找到能够区分正样本和负样本的决策边界，需要据此拟合一个假设函数。

无监督学习

与此不同的是，在无监督学习中，我们的数据没有附带任何标签y，无监督学习主要分为聚类、降维、关联规则、推荐系统等方面。

1.无监督学习方法概述

主要的无监督学习方法

- ✓ 聚类（Clustering）
  - ✓ 如何将教室里的学生按爱好、身高划分为5类?
- ✓ 降维（ Dimensionality Reduction ）
  - ✓ 如何将将原高维空间中的数据点映射到低维度的空间中?
- ✓ 关联规则（ Association Rules）
  - ✓ 很多买尿布的男顾客，同时买了啤酒，可以从中找出什么规律来提高超市销售额?
- ✓ 推荐系统（ Recommender systems）
  - ✓ 很多客户经常上网购物，根据他们的浏览商品的习惯，给他们推荐什么商品呢?

1.无监督学习方法概述

聚类

主要算法
K-means、密度聚类、层次聚类

主要应用

市场细分、文档聚类、图像分割、图像压缩、聚类分析、特征学习或者词典学习、确定犯罪易发地区、保险欺诈检测、公共交通数据分析、IT资产集群、客户细分、识别癌症数据、搜索引擎应用、医疗应用、药物活性预测……

1.无监督学习方法概述

聚类案例

1.医疗

医生可以使用聚类算法来发现疾病。以甲状腺疾病为例。当我们对包含甲状腺疾病和非甲状腺疾病的数据集应用无监督学习时，可以使用聚类算法来识别甲状腺疾病数据集。

1.无监督学习方法概述

聚类案例

2.市场细分

为了吸引更多的客户，每家公司都在开发易于使用的功能和技术。为了了解客户，公司可以使用聚类。聚类将帮助公司了解用户群，然后对每个客户进行归类。这样，公司就可以了解客户，发现客户之间的相似之处，并对他们进行分组。

1.无监督学习方法概述

聚类案例

3.金融业

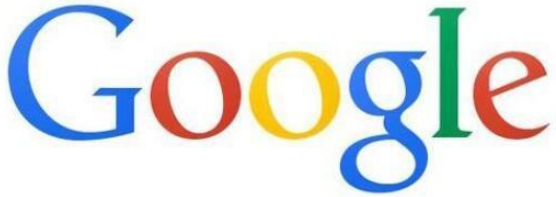
银行可以观察到可能的金融欺诈行为，就此向客户发出警告。在聚类算法的帮助下，保险公司可以发现某些客户的欺诈行为，并调查类似客户的保单是否有欺诈行为。

1.无监督学习方法概述

聚类案例

4.搜索引擎

百度是人们使用的搜索引擎之一。举个例子，当我们搜索一些信息，如在某地的超市，百度将为我们提供不同的超市的选择。这是聚类的结果，提供给你的结果就是聚类的相似结果。



1.无监督学习方法概述

聚类案例

5.社交网络

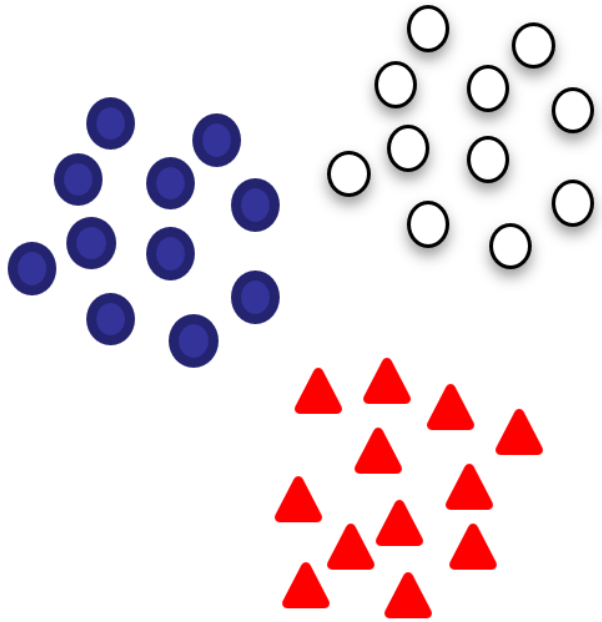
比如在社交网络的分析上。已知你朋友的信息，比如经常发email的联系人，或是你的微博好友、微信的朋友圈，我们可运用聚类方法自动地给朋友进行分组，做到让每组里的人们彼此都熟识。

2.K-means聚类

2.K-means聚类

聚类的背景知识--基本思想

图中的数据可以分成三个分开的**点集**(称为**簇**)，一个能够分出这些点集的算法，就被称为**聚类算法**。



聚类算法示例

2.K-means聚类

K-均值算法(K-means)算法概述

K-means算法是一种**无监督学习**方法，是最普及的聚类算法，算法使用一个**没有标签**的数据集，然后将数据聚类成不同的组。

K-means算法具有一个迭代过程，在这个过程中，数据集被分组成若干个预定义的不重叠的聚类或子组，使簇的内部点尽可能相似，同时试图保持簇在不同的空间，它将数据点分配给簇，以便**簇的质心和数据点之间的平方距离之和最小**，在这个位置，簇的质心是簇中数据点的算术平均值。

距离度量

闵可夫斯基距离(Minkowski distance)

p取1或2时的闵氏距离是最为常用的

$p = 2$即为欧氏距离

$p = 1$时则为曼哈顿距离

当p取无穷时的极限情况下，可以得到切比雪夫距离

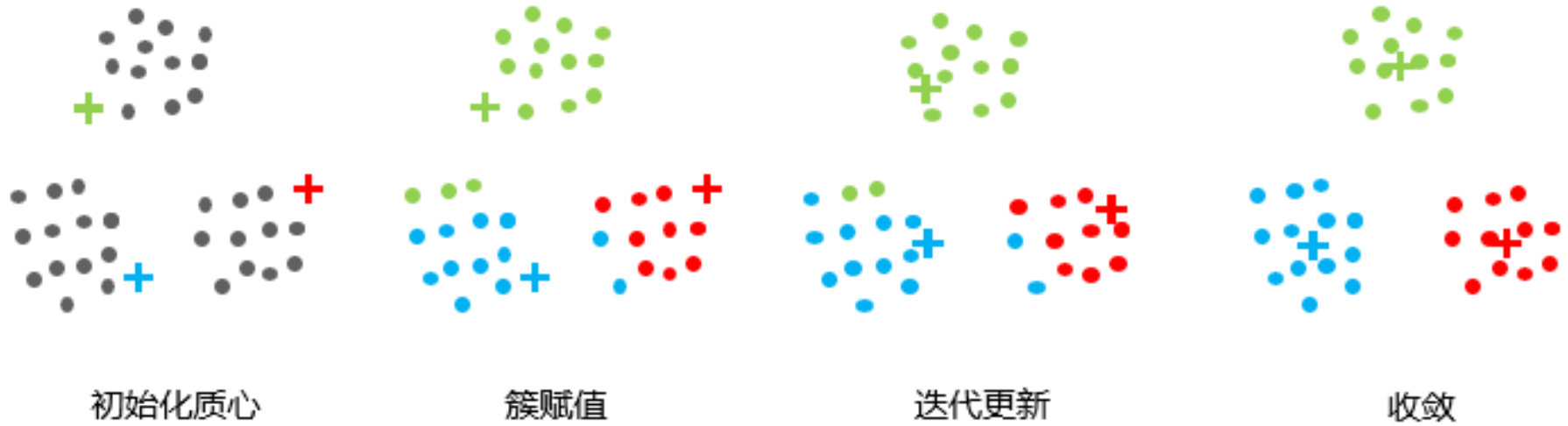
$$d(x,y) = \left(\sum_i |x_i - y_i|^p\right)^{\frac{1}{p}}$$

欧氏距离：$d(x,y) = \left(\sum_i |x_i - y_i|^2\right)^{\frac{1}{2}}$

2.K-means聚类

K-means算法流程

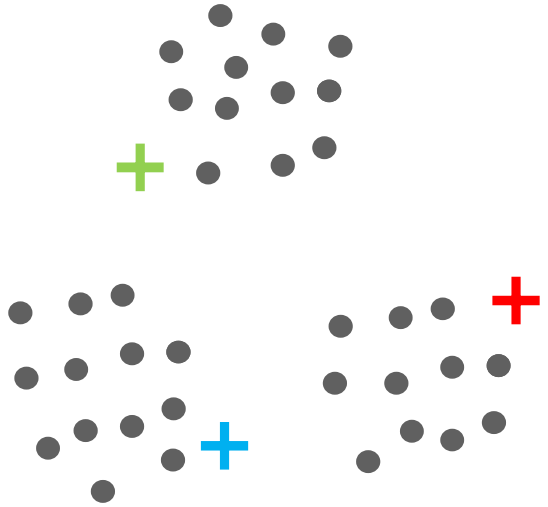
1. 选择K个点作为初始质心。
2. 将每个点指派到最近的质心，形成K个簇。
3. 对于上一步聚类的结果，进行平均计算，得出该簇的新的聚类中心。
4. 重复上述两步/直到迭代结束：质心不发生变化。



2.K-means聚类

K-means算法流程

首先，初始化称为簇质心的任意点。初始化时，必须注意簇的质心必须小于训练数据点的数目。因为该算法是一种迭代算法，接下来的两个步骤是迭代执行的。

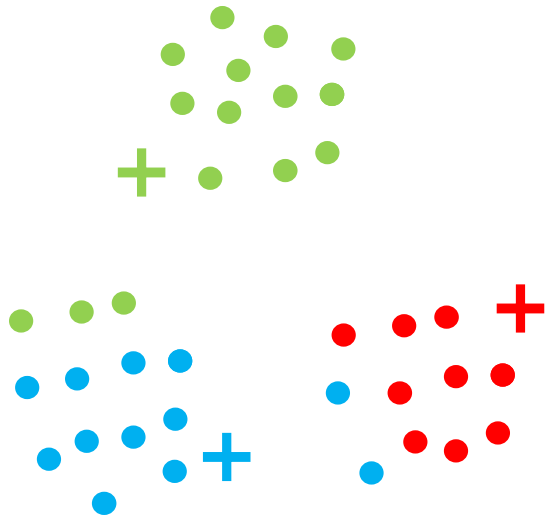


初始化质心

2.K-means聚类

K-means算法流程

初始化后，遍历所有数据点，计算所有质心与数据点之间的距离。现在，这些簇将根据与质心的最小距离而形成。在本例中，数据分为3个簇($K = 3$)。

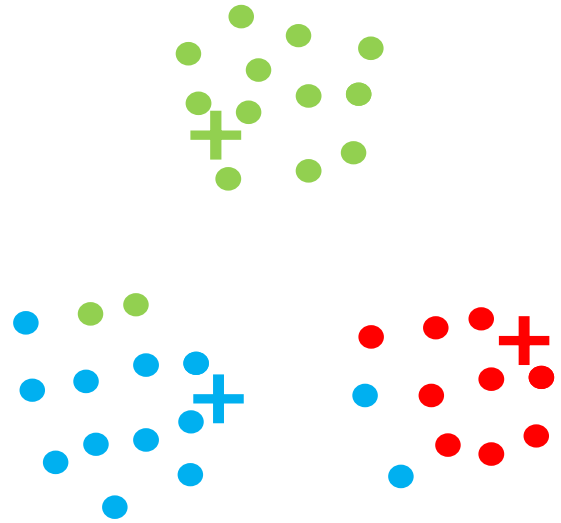


簇赋值

2.K-means聚类

K-means算法流程

第三步：移动质心，因为上面步骤中形成的簇没有优化，所以需要形成优化的簇。为此，我们需要迭代地将质心移动到一个新位置。取一个簇的数据点，计算它们的平均值，然后将该簇的质心移动到这个新位置。对所有其他簇重复相同的步骤。



迭代更新

2.K-means聚类

K-means算法流程

优化

上述两个步骤是迭代进行的，直到质心停止移动，即它们不再改变自己的位置，并且成为静态的。一旦这样做，k-均值算法被称为收敛。

K-均值的代价函数（又称**畸变函数 Distortion function**）为：

$$J(c^{(1)},\ldots,c^{(m)},\mu_1,\ldots,\mu_K)=\frac{1}{m}\sum_{i=1}^{m}\|X^{(i)}-\mu_{c^{(i)}}\|^2$$

设训练集为：$\{x^{(1)},x^{(2)},x^{(3)},\ldots,x^{(m)}\}$，簇划分$C=\{C_1,C_2,\cdots,C_K\}$，用$\mu_1,\mu_2,\ldots,\mu_K$来表示聚类中心

其中$\mu_{c^{(i)}}$代表与$x^{(i)}$最近的聚类中心点。

我们的的优化目标便是找出使得代价函数最小的 $c^{(1)}$,$c^{(2)}$,...,$c^{(m)}$和 $\mu_1,\mu_2,\ldots,\mu_K$ 。

2.K-means聚类

K-means优化过程

记k个簇中心为$\mu_1, \mu_2, \ldots, \mu_k$，每个簇的样本数目为$N_1$, N_2,..., N_k

使用平方误差作为目标函数:

$$J(\mu_1, \mu_2, \cdots \mu_k) = \frac{1}{2}\sum_{j=1}^{K}\sum_{i=1}^{N_j}(x_i - \mu_j)^2$$

对关于从$\mu_1, \mu_2, \cdots \mu_k$的函数求偏导，这里的求偏导是对第$j$个簇心$\mu_j$求的偏导。故而其驻点为:

$$\frac{\partial J}{\partial \mu_j} = -\sum_{i=1}^{N_j}(x_i - \mu_j) \overset{令}{\rightarrow} 0 \Rightarrow \mu_j = \frac{1}{N_j}\sum_{i=1}^{N_j} x_i$$

推导:

$$\frac{\partial J}{\partial \mu_j} = \frac{\partial \frac{1}{2}\sum_{j=1}^{k}\sum_{i=1}^{N_j}(x_i - \mu_j)^2}{\partial \mu_j}$$

$$= \frac{\partial \frac{1}{2}\sum_{i=1}^{N_j}(x_i - \mu_j)^2}{\partial \mu_j}$$

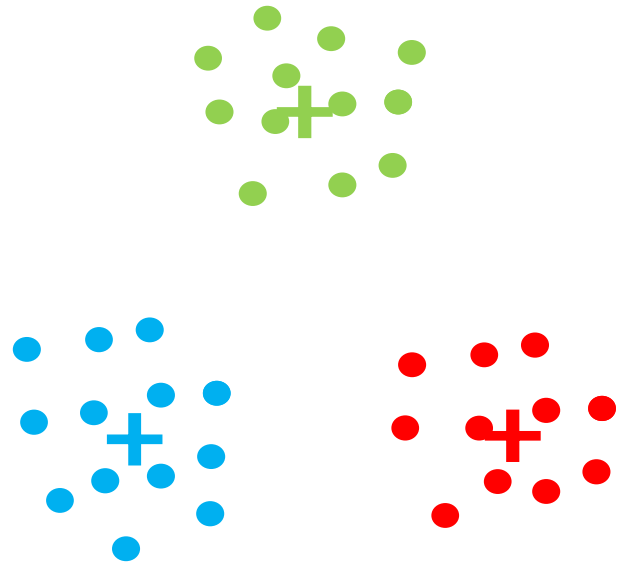
$$= \sum_{i=1}^{N_j}(x_i - \mu_j)\cdot(-1)$$

$$= -\sum_{i=1}^{N_j}(x_i - \mu_j)$$

2.K-means聚类

K-means算法流程

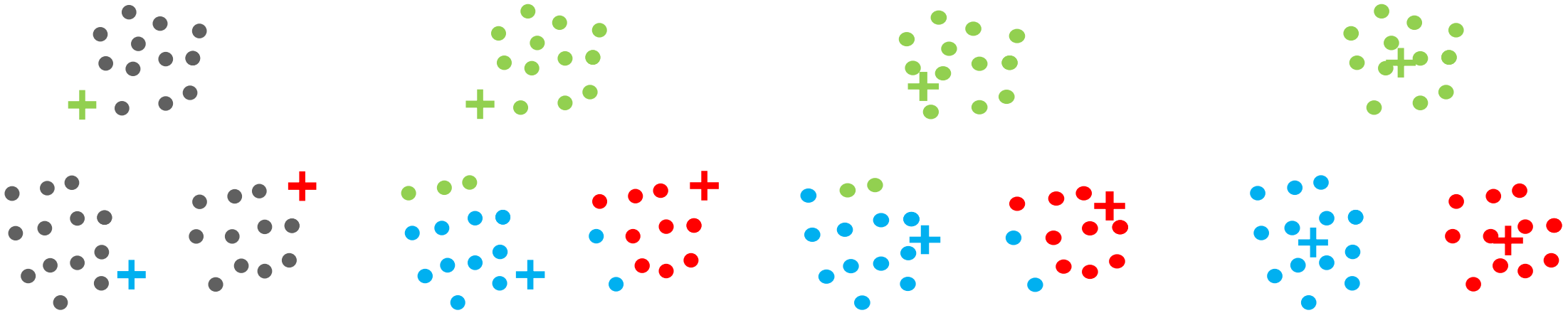
现在，这个算法已经收敛，形成了清晰可见的不同簇。该算法可以根据簇在第一步中的初始化方式给出不同的结果。



收敛

2.K-means聚类

K-means算法流程总结

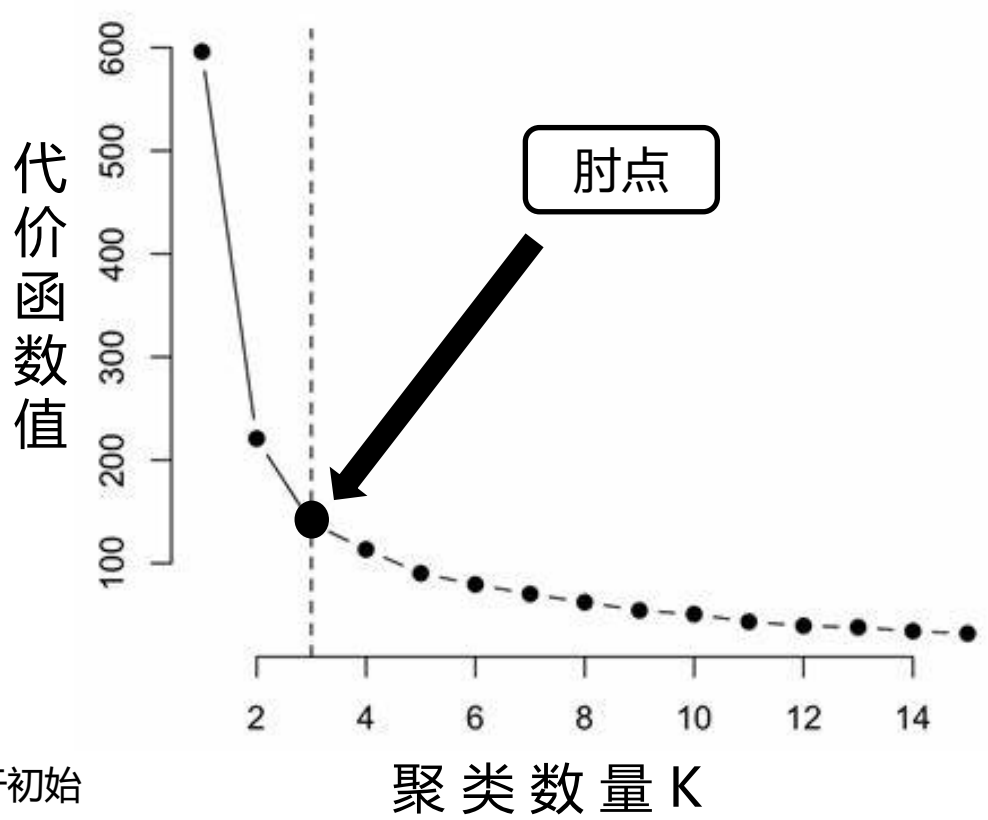


初始化质心　　簇赋值　　迭代更新　　收敛

2.K-means聚类

K值的选择

现在我们需要找到簇的数量。通常通过“肘部法则”进行计算。我们可能会得到一条类似于人的肘部的曲线。右图中，代价函数的值会迅速下降，在$K = 3$的时候达到一个肘点。在此之后，代价函数的值会就下降得非常慢，所以，我们选择$K = 3$。这个方法叫“肘部法则”。



K-均值的一个问题在于，它有可能会停留在一个局部最小值处，而这取决于初始化的情况。

为了解决这个问题，我们通常需要多次运行**K-均值**算法，每一次都重新进行随机初始化，最后再比较多次运行**K-均值**的结果，选择代价函数最小的结果。

2.K-means聚类

K-means的优点

鲁棒性高；

速度快、易于理解、效率高；

计算成本低、灵活性高；

如果数据集是不同的，则结果更好；

可以产生更紧密的簇；

重新计算质心时，簇会发生变化。

2.K-means聚类

K-means的缺点

需要预先指定簇的数量；

如果有两个高度重叠的数据，那么它就不能被区分，也不能判断有两个簇；

欧几里德距离可以不平等的权重因素，限制了能处理的数据变量的类型；

有时随机选择质心并不能带来理想的结果；

无法处理异常值和噪声数据；

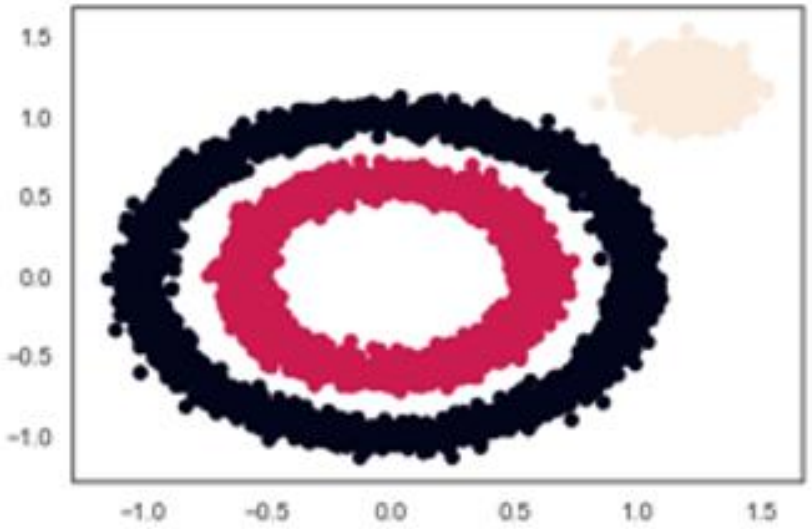
不适用于非线性数据集；

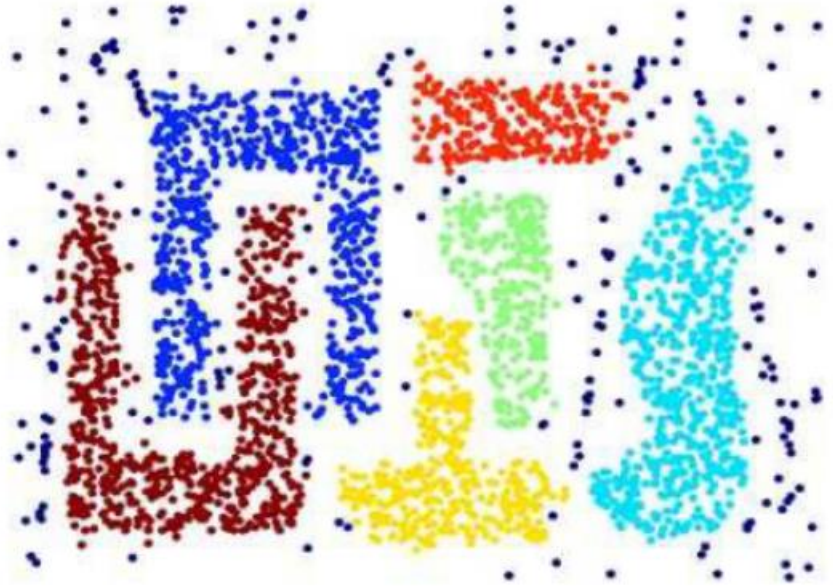
对特征尺度敏感；

如果遇到非常大的数据集，那么计算机可能会崩溃。

3.密度聚类和层次聚类

密度聚类-DBSCAN





背景知识：如果 S 中任两点的连线内的点都在集合 S 内，那么集合 S 称为凸集。反之，为非凸集。

密度聚类-DBSCAN

DBSCAN密度聚类

与划分和层次聚类方法不同，DBSCAN(Density-Based Spatial Clustering of Applications with Noise)是一个比较有代表性的基于密度的聚类算法。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簇**，并可**在噪声的空间数据库中发现任意形状的聚类**。

密度：空间中任意一点的密度是以该点为圆心，以**扫描半径**构成的圆区域内包含的点数目。

密度聚类-DBSCAN

DBSCAN使用**两个超参数**:

扫描半径 (eps)和最小包含点数(minPts)来获得簇的数量，而不是猜测簇的数目。

- **扫描半径 (eps)：**

用于定位点/检查任何点附近密度的距离度量，即扫描半径。

- **最小包含点数(minPts)：**

聚集在一起的最小点数（阈值），该区域被认为是稠密的。

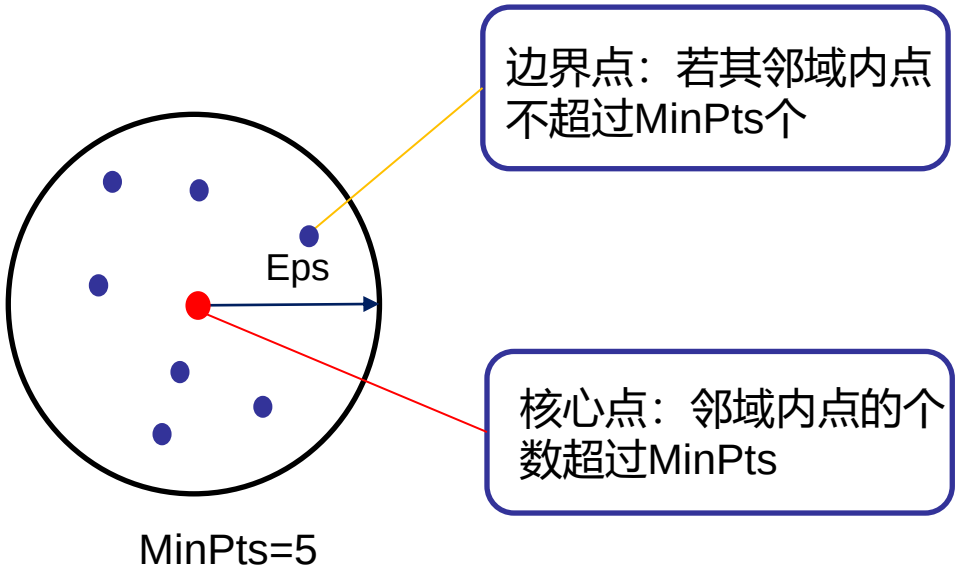
密度聚类-DBSCAN

DBSCAN算法将数据点分为三类：

1.**核心点**：在半径Eps内含有超过MinPts数目的点。

2.**边界点**：在半径Eps内点的数量小于MinPts，但是落在核心点的邻域内的点。

3.**噪音点**：既不是核心点也不是边界点的点。



3.密度聚类和层次聚类

DBSCAN密度聚类的算法流程

1.将所有点标记为核心点、边界点或噪声点;

2. 如果选择的点是核心点，则找出所有从该点出发的密度可达对象形成簇;

3. 如果该点是非核心点，将其指派到一个与之关联的核心点的簇中;

4. 重复以上步骤，直到所点都被处理过

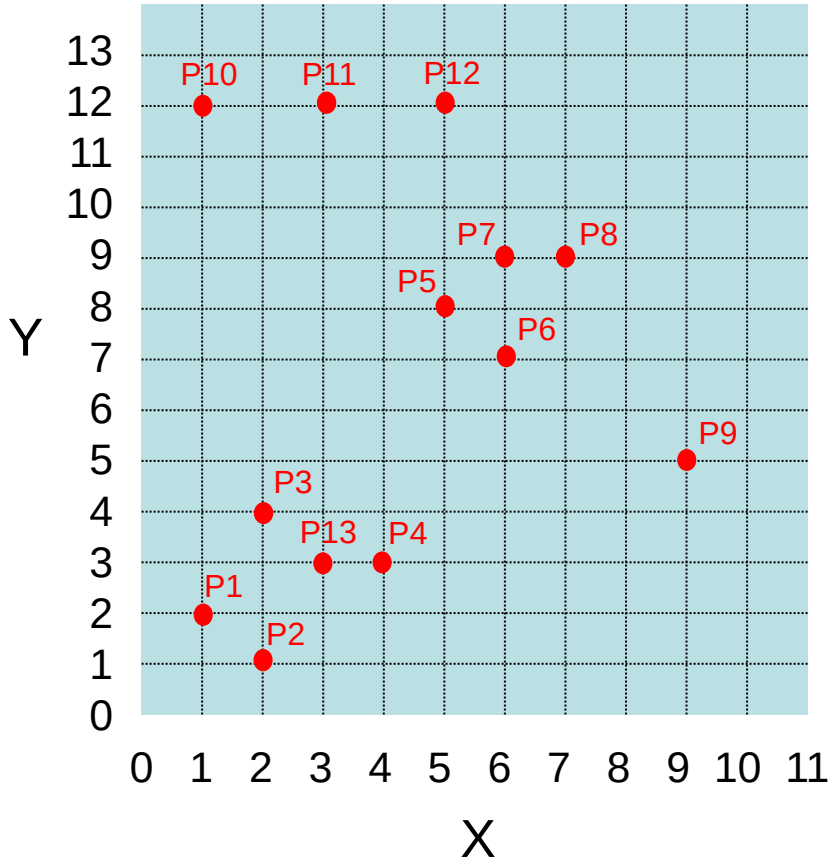
举例：有如下13个样本点，使用DBSCAN进行聚类

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13
X	1	2	2	4	5	6	6	7	9	1	3	5	3
Y	2	1	4	3	8	7	9	9	5	12	12	12	3

密度聚类-DBSCAN

DBSCAN密度聚类的算法流程

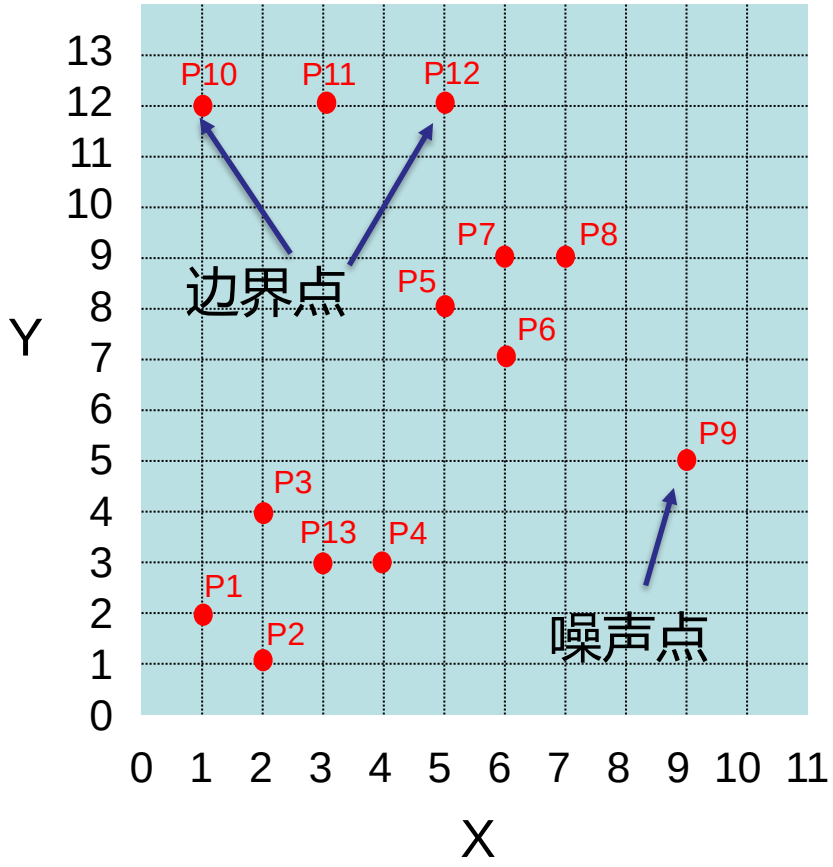
- 对每个点计算其邻域Eps=3内的点的集合。
- 集合内点的个数超过MinPts=3的点为核心点。



密度聚类-DBSCAN

DBSCAN密度聚类的算法流程

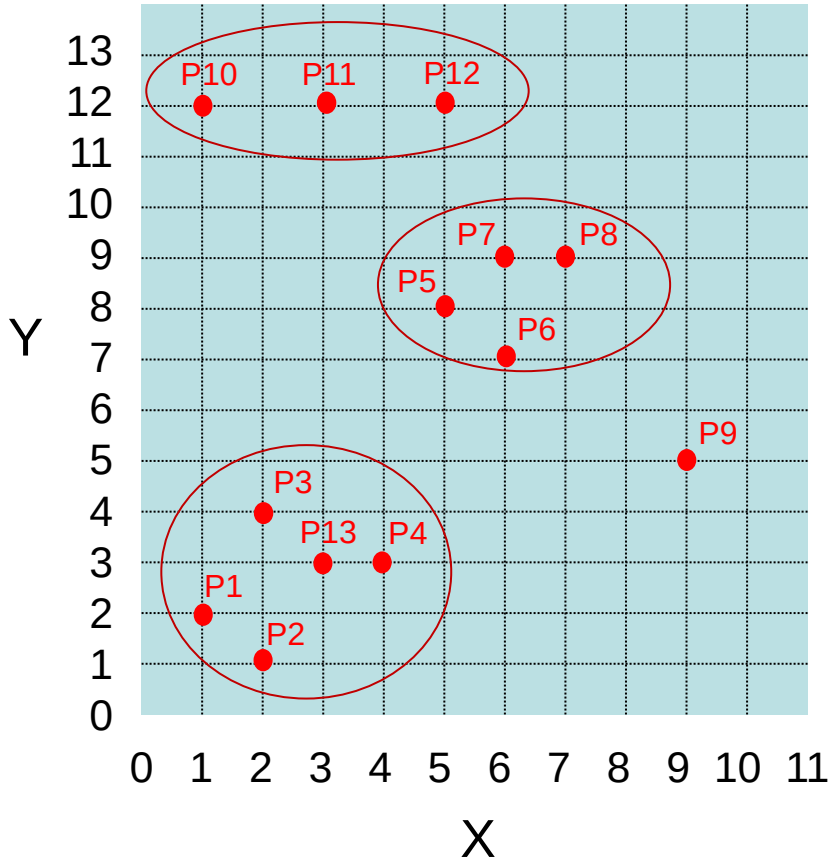
- 查看剩余点是否在核点的邻域内，若在，则为边界点，否则为噪声点。



密度聚类-DBSCAN

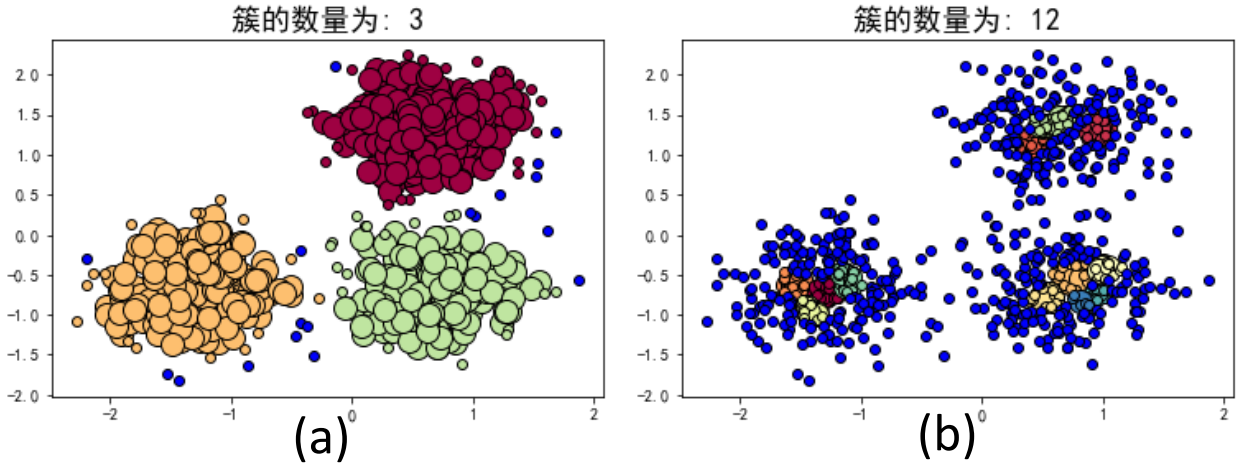
DBSCAN密度聚类的算法流程

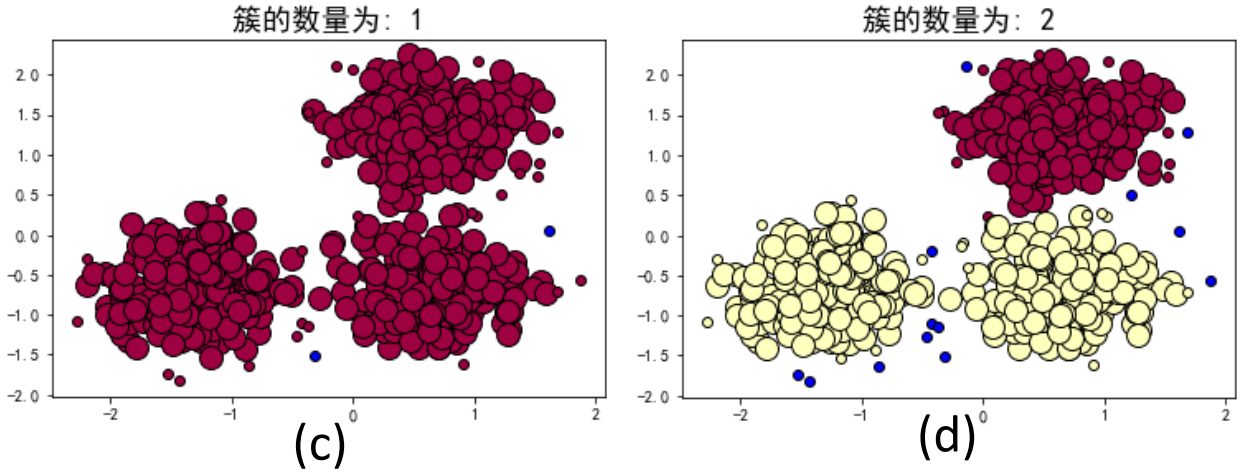
- 将距离不超过Eps=3的点相互连接，构成一个簇，核心点邻域内的点也会被加入到这个簇中。



密度聚类-DBSCAN

DBSCAN的超参数





DBSCAN超参数案例

图片编号		(a)	(b)	(c)	(d)
评价指标	超参数	eps=0.3 minPts=10	eps=0.1 minPts=10	eps=0.4 minPts=10	eps=0.3 minPts=6
估计的簇的数量		3	12	1	2
估计的噪声点		18	516	2	13
同一性		0.9530	0.3128	0.0010	0.5365
完整性		0.8832	0.2489	0.0586	0.8623
V-measure		0.9170	0.0237	0.0020	0.6510
ARI		0.9517	0.2673	0	0.5414
轮廓系数		0.6255	-0.3659	0.0611	0.3845

这个案例中，当：

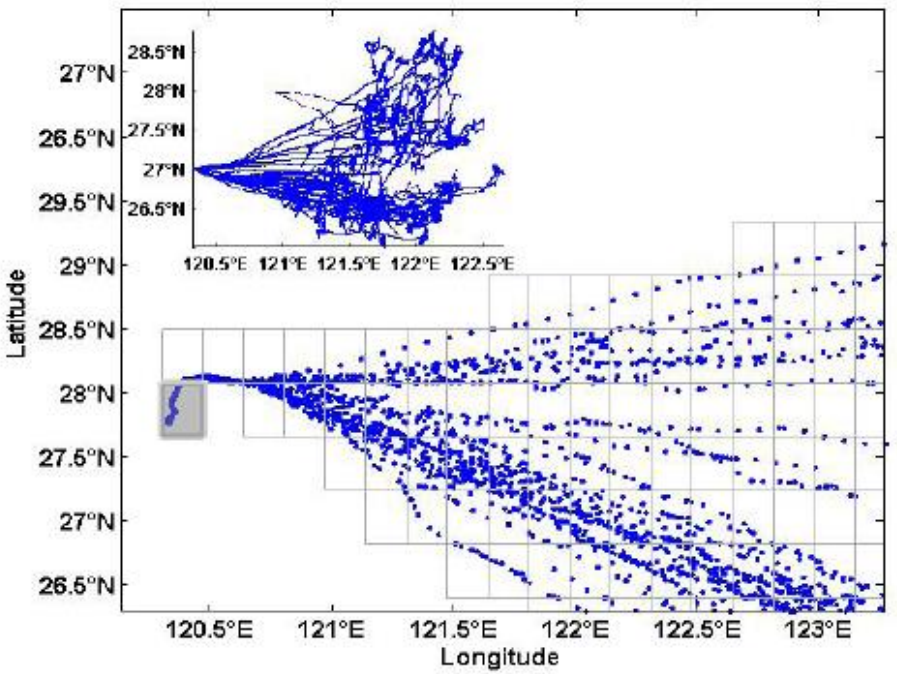
eps=0.3，minPts=10的时候，

DBSCAN达到最优效果。

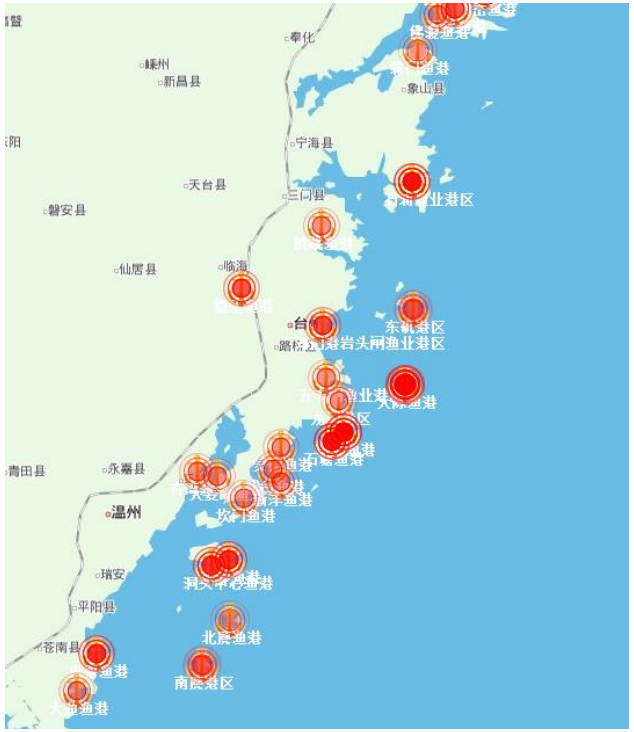
密度聚类应用

通过单拖船轨迹推算港口范围

单拖船的作业规律比较清晰，出港后，全速驶向作业区域，在作业区域拖网作业，一个航次结束，全速驶向渔港，我们设计了一种基于DBSCAN和K-means的混合FindPort算法



典型的单拖船一年的轨迹图



FindPort算法计算的渔港图

层次聚类

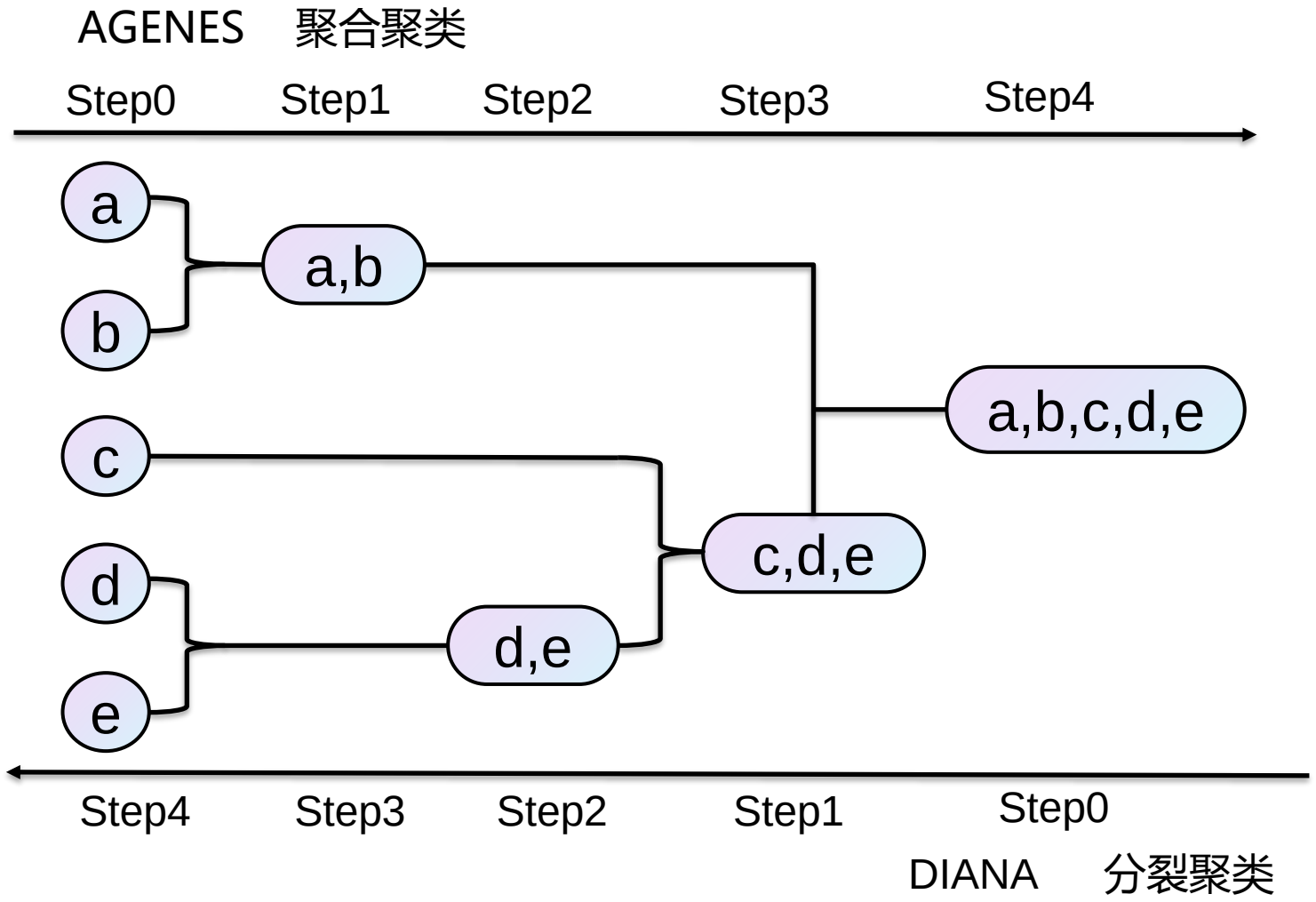
层次聚类

- 层次聚类假设簇之间存在层次结构，将样本聚到层次化的簇中。
- 层次聚类又有聚合聚类（自下而上）、分裂聚类（自上而下）两种方法。
- 因为每个样本只属于一个簇，所以层次聚类属于硬聚类。

背景知识：如果一个聚类方法假定一个样本只能属于一个簇，或簇的交集为空集，那么该方法称为硬聚类方法。

如果一个样本可以属于多个簇，或簇的交集不为空集，那么该方法称为软聚类方法。

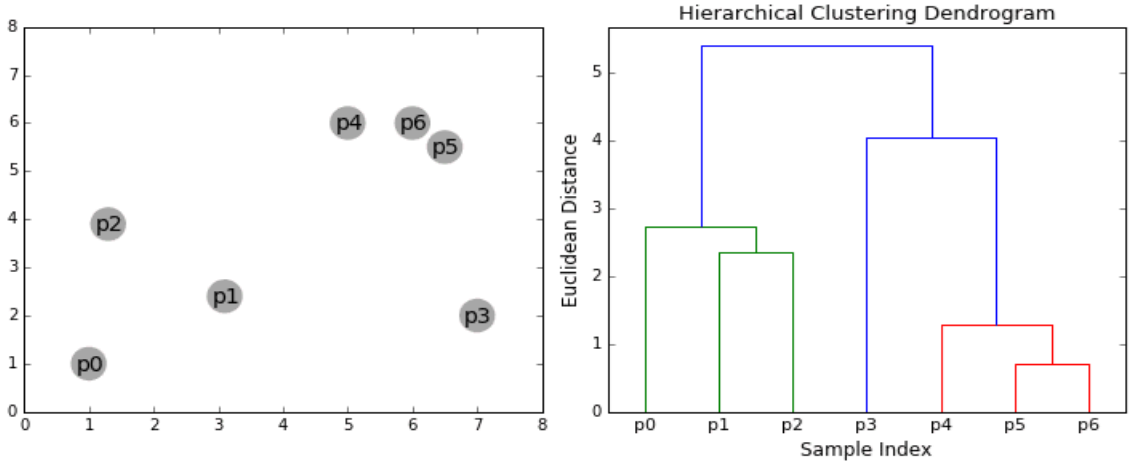
层次聚类

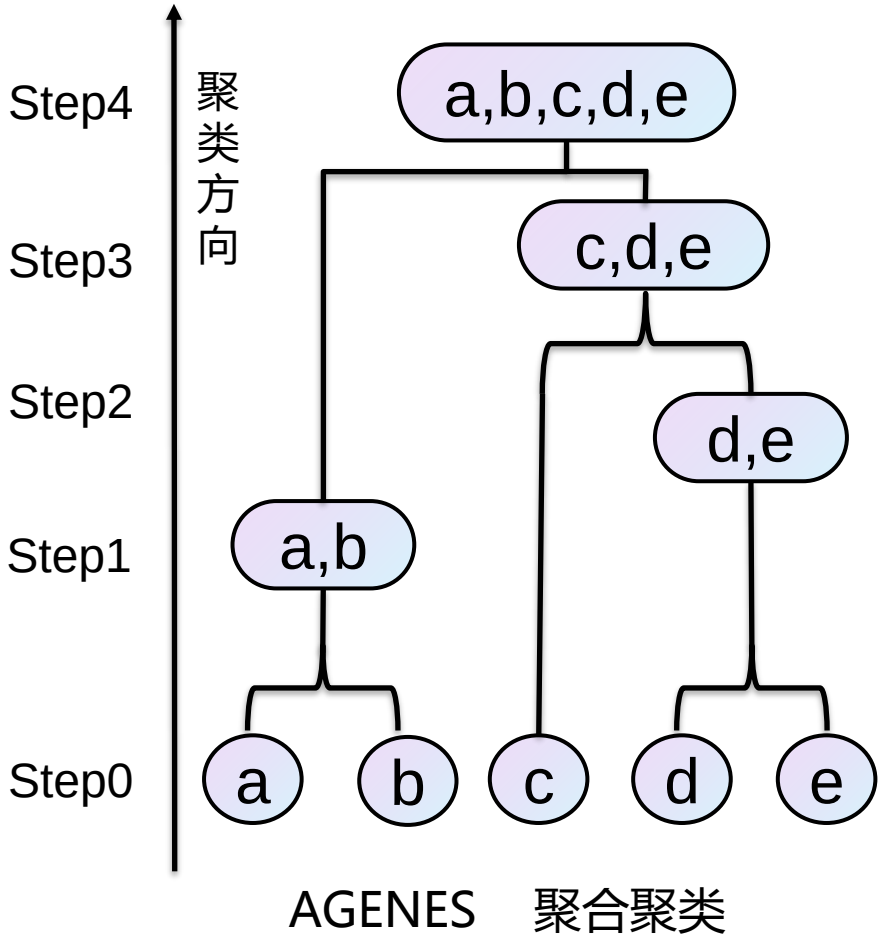

AGENES 聚合聚类
Step0
Step1
Step2
Step3
Step4
a
b
a,b
c
d
e
d,e
c,d,e
a,b,c,d,e
Step4
Step3
Step2
Step1
Step0
DIANA 分裂聚类

层次聚类-聚合聚类

聚合聚类

- 开始将每个样本各自分到一个簇；
- 之后将相距最近的两簇合并，建立一个新的簇；
- 重复此操作直到满足停止条件；
- 得到层次化的类别。

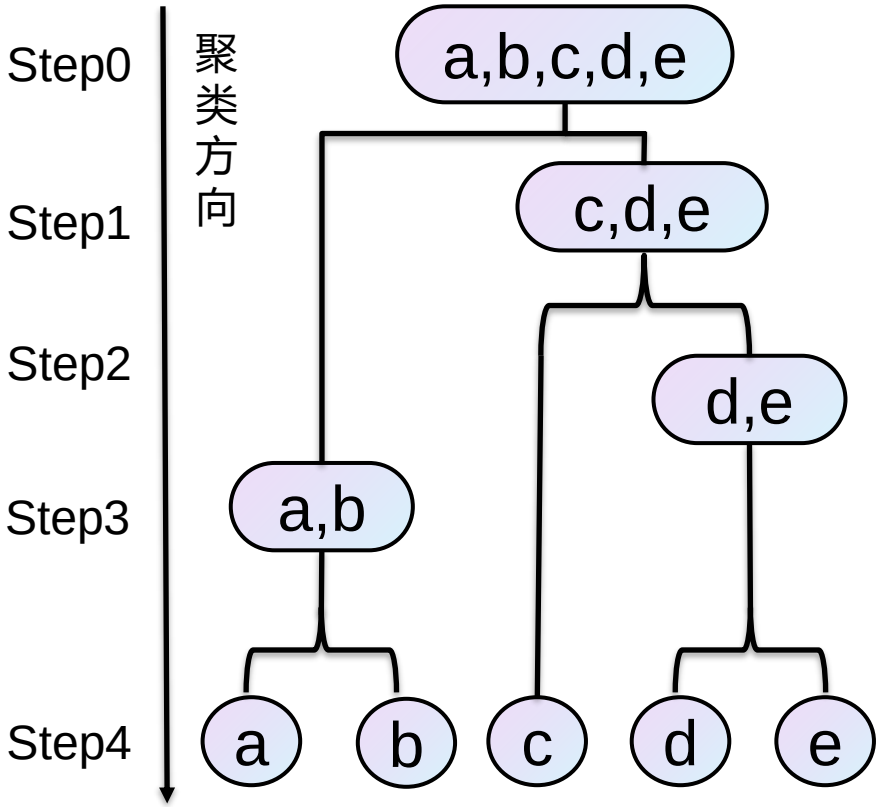




层次聚类-分裂聚类

分裂聚类

- 开始将所有样本分到一个簇；
- 之后将已有类中相距最远的样本分到两个新的簇；
- 重复此操作直到满足停止条件；
- 得到层次化的类别。



DIANA 分裂聚类

4. 聚类的评价指标

4. 聚类的评价指标

(1) 均一性：p

类似于精确率，一个簇中只包含一个类别的样本，则满足均一性。其实也可以认为就是正确率(每个聚簇中正确分类的样本数占该聚簇总样本数的比例和)

$$p = \frac{1}{k}\sum_{i=1}^{k}\frac{N(C_i == K_i))}{N(K_i)}$$

(2) 完整性：r

类似于召回率，同类别样本被归类到相同簇中，则满足完整性;(每个聚簇中正确分类的样本数占该类型的总样本数比例的和)

$$r = \frac{1}{k}\sum_{i=1}^{k}\frac{N(C_i == K_i))}{N(C_i)}$$

(3) V-measure:

均一性和完整性的加权平均

$$V = \frac{(1+\beta^2) * pr}{\beta^2 * p + r}$$

4. 聚类的评价指标

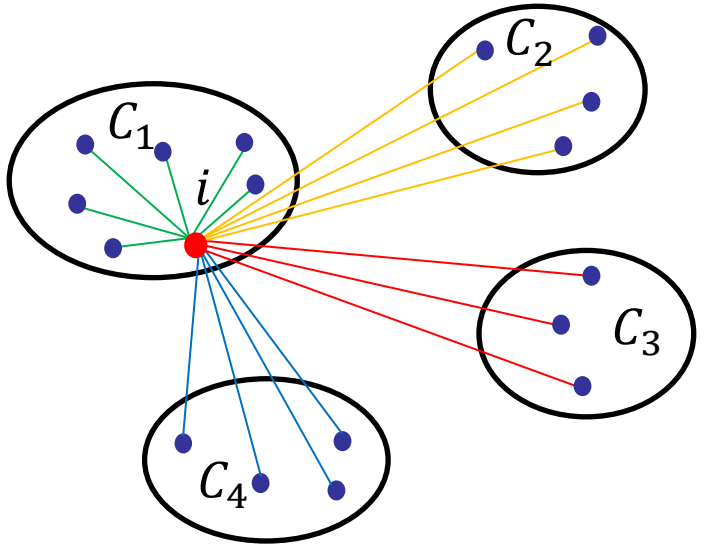
(4) 轮廓系数

样本i的轮廓系数：$$s(i) = \frac{b(i) - a(i)}{max\{a(i), b(i)\}}$$

簇内不相似度:计算样本i到同簇其它样本的平均距离为$a(i)$，应尽可能小。

簇间不相似度:计算样本i到其它簇C_j的所有样本的平均距离b_{ij}，应尽可能大。

轮廓系数$s(i)$值越接近1表示样本i聚类越合理，越接近-1，表示样本i应该分类到另外的簇中，近似为0，表示样本i应该在边界上;所有样本的$s(i)$的均值被成为聚类结果的轮廓系数。



假设数据集被拆分为4个簇，样本i对应的$a(i)$值就是所有C_1 中其他样本点与样本i的距离平均值;

样本对应的$b(i)$值分两步计算，首先计算该点分别到C_2、 C_3和C_4中样本点的平均距离，然后将三个平均值中的最小值作为$b(i)$的度量.

4. 聚类的评价指标

(5).调整兰德系数(ARI, Adjusted Rnd Index)

数据集S共有N个元素， 两个聚类结果分别是：

$$X = \{X_1, X_2, \ldots, X_r\}, Y = \{Y_1, Y_2, \ldots, Y_s\}$$

X和Y的元素个数为：

$$a = \{a_1, a_2, \ldots, a_r\}, b = \{b_1, b_2, \ldots, b_s\}$$

C	Y_1	Y_2	$\cdots$	Y_s	sum
X_1	n_{11}	n_{12}	$\cdots$	n_{1s}	a_1
X_2	n_{21}	n_{22}	$\cdots$	n_{2s}	a_2
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
X_r	n_{r1}	n_{r2}	$\cdots$	n_{rs}	a_r
sum	b_1	b_2	$\cdots$	b_s	N

记： $n_{ij} = |X_i \cap Y_i|$

$$ARI = \frac{\sum_{i,j} C_{n_{ij}}^2 - \left[\left(\sum_i C_{a_i}^2\right) \cdot \left(\sum_i C_{b_i}^2\right)\right] / C_n^2}{\frac{1}{2}\left[\left(\sum_i C_{a_i}^2\right) + \left(\sum_i C_{b_i}^2\right)\right] - \left[\left(\sum_i C_{a_i}^2\right) \cdot \left(\sum_i C_{b_i}^2\right)\right] / C_n^2}$$

ARI取值范围为$[-1,1]$，值越大意味着聚类结果与真实情况越吻合。从广义的角度来讲，ARI衡量的是两个数据分布的吻合程度

参考文献

1. 《统计学习方法》，清华大学出版社，李航著，2019年出版
2. 《机器学习》，清华大学出版社，周志华著，2016年出版
3. Christopher M. Bishop, Pattern Recognition and Machine Learning, Springer-Verlag, 2006
4. 《人工智能概论》，北京联合大学，彭涛
5. 《机器学习》，邹伟


谢 谢!