概述

聚类任务是将样本集划分为几个没有交集的子集（也叫“簇”），通过划分，使不同的簇对应于实际中具有区别的概念（或者说结构）。

聚类属于无监督学习任务，因为它不使用样本的标签，只使用样本的特征。聚类的基本原则是“簇内样本相似度尽量高，簇间样本相似度尽量低”（这也是制定聚类性能指标的基本依据）。

在实际中，我们会遇到各种各样的聚类问题，它们对聚类结构有不同的假设，从而形成不同的聚类算法。目前主要的聚类算法有四类：基于原型的聚类、基于密度的聚类、层次聚类和谱聚类。下面分别介绍它们。

一、基于原型的聚类

基于原型的聚类算法，其基本思想是认为每个类别都有一个“聚类原型”，对于所有的样本，都依据其与聚类原型的关系来确定属于哪一个簇。基于原型的聚类包括KMeans算法和高斯混合模型等。

**1、KMeans算法**

KMeans聚类算法需要定义样本之间的“距离”。两个样本的距离越近，则样本间相似度越高，从而它们越有可能划分到同一个簇。KMeans算法对每个簇设定一个聚类原型（在KMeans算法中，又叫聚类中心），该中心是与样本同维度的向量，从而样本与聚类中心之间可以计算距离。每个样本将被划分到与其距离最近的聚类中心对应的簇中。

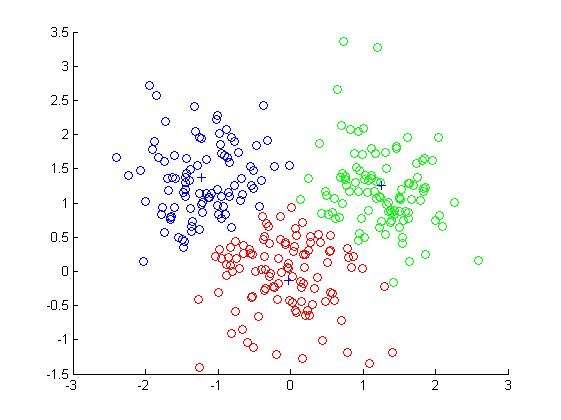


图1 KMeans算法示意图

图1为KMeans算法示意图，它是对2维的样本数据进行聚类的结果，类别数为3。图中，每个类别的聚类中心被标记为“+”。

下面我们给出KMeans算法的优化目标：

假设样本集最终划分为K个簇：，每个簇的中心是，则KMeans的优化目标为：



可以看出，它是想最小化每个簇中的所有样本与其聚类中心之间的均方误差，这个量的意义类似于“方差”，它从整体上衡量了一个簇内部的样本之间集中程度。

注意KMeans优化目标需要优化两个参数：一是簇的划分，二是每个簇的聚类中心。实际上，若簇的划分确定了，簇的聚类中心可以很容易地进行优化；若每个簇的聚类中心确定了，簇的划分也很容易优化。但是二者都未知，同时对它们优化就比较困难了。

对此我们采用“交替优化”的思想，即每次先固定一个参数，优化另一个参数，如此迭代地优化。

具体的步骤如下：

（1）随机选择样本集中的K个样本作为初始聚类中心（K值事先选定）；

（2）迭代，直至收敛：

（a）固定聚类中心，计算当前最优的簇的划分：计算每个样本与聚类中心的距离，并将样本划入与之距离最近的聚类中心对应的簇；

（b）固定簇划分，计算当前最优的聚类中心：各簇中的样本的平均值作为新的聚类中心。

KMeans的这种交替优化属于局部优化，不能得到全局最优值，因此不同的初始聚类中心可能得到不同的最优解。

**2、高斯混合模型**

高斯混合模型的思想是，假设样本集中的样本由几个不同的高斯分布生成。则每个高斯分布对应于一个簇，该簇由所有由该分布生成的样本组成。

这些高斯分布就是高斯混合模型的“聚类原型”。

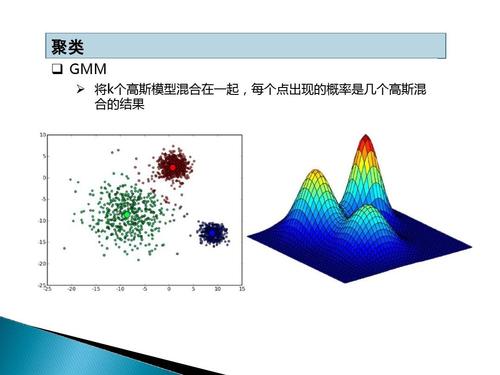


图2 高斯混合模型示意图

设总共有K个高斯分布，则整体的分布为：



以上就是高斯混合模型的具体形式，其中，分别是组成整体分布的K个高斯分布的系数、方差、期望。它们是模型的参数（或者说总体分布的参数）。

高斯混合模型对样本集的整体分布进行建模，模型可以给出每个样本的生成概率（或者说每个样本“出现的概率”），以及每个样本由任何一个高斯分布生成的概率。然后，高斯混合模型将每个样本划分到生成该样本的概率最大的高斯分布对应的簇中。

高斯混合模型属于含有隐变量的概率模型，其参数的求解无法使用极大似然估计，而要使用EM算法。

基于EM算法的高斯混合模型最优参数的求解步骤如下：

（1）初始化未知参数；

（2）迭代,直至收敛：

（a）E步：固定模型参数，计算每个样本由每个分布生成的概率：



表示第i个样本由第k个高斯分布产生的概率。可以看出，它不仅与

高斯分布有关，还与高斯分布的系数有关，一个高斯分布的系数越大，样本由其生成的概率就越大，另外注意，它是经过归一化的；

（b）M步：固定E步计算出的，计算模型的参数：







可以看出，高斯分布的期望和方差的计算类似于利用统计量估计的方法，但不同的是，其考虑了每个样本观察值的“权重”或说“重要性”，样本的权重是其由高斯分布产生的概率，该样本由该高斯分布产生的概率越大，该样本的重要性就越大；

而高斯分布的系数由所有样本由它产生的概率之和决定，概率越大，该高斯分布的系数就越大。

可以看出，高斯混合聚类的优化流程与KMeans的优化流程有相似之处。都属于“交替优化”。

高斯混合模型的优化流程可以由EM算法严格地推导出来，然而从它的流程本身上看，也具有很强的可解释性。

注意，上面给出的高斯混合模型，是以多元线性无关高斯分布为例的，而对于一般的情况，即多元高斯分布，其各元之间（即样本的各维特征之间）不一定独立，这时分布参数中的方差就要换成协方差。不过基本流程是不变的。

二、基于密度的聚类

基于密度的聚类算法从“密度”的角度来看待聚类问题，它认为样本数据的聚类结构是由密度决定的，密度越大的地方越容易形成簇。

注意，“密度”的意义也得基于“距离”的概念来体现。

DBSCAN是一种著名的基于密度的聚类算法。下面介绍该算法的原理：

**1、基本概念**

DBSCAN算法首先要定义几个概念：

给定数据集，定义如下5个基本概念：

（1）邻域：，即对给定的样本，其邻域是样本集中与其距离不大于的所有样本的集合；

（2）核心对象：若样本的邻域至少包含MinPts个样本，即，则称样本是一个核心对象（可见，核心对象是其附近“密度”足够大的样本）；

（3）密度直达：若样本在核心对象的邻域中，则称由密度直达；

（4）密度可达：若样本可由样本（通过密度直达关系）经过一系列的其他样本传递而达到，则称样本由密度可达（注意，这时，这一系列的中间样本都得是核心对象）；

（5）密度相连：对于样本和，若存在，使得和均由密度可达，则称和密度相连。

**2、簇的定义**

基于这些概念，DBSCAN给出了“簇”的定义：一个簇就是由密度可达关系导出的最大的密度相连的样本集合。也就是说，簇中任何两个样本都是密度相连的，并且簇中的任何样本密度可达的样本也在簇中。直观上看，这样定义的一个簇就是样本空间中一个密度较高的区域，它与周围密度较低的区域区分开来。

**3、算法流程**

DBSCAN算法求解样本集的簇的思路是：先找出所有的核心对象，然后对每个核心对象，找出由其密度可达的所有样本构成的集合。

具体的算法流程略。

可以看出，基于密度的聚类算法的超参数有：邻域大小和核心对象最小包含的样本数MinPts。直观上看，这两个参数决定了密度的大小。

基于密度聚类算法的优点是：

（1）在完成聚类的同时，能够顺便识别出异常点；

（2）能够用来考察数据本身的可分性。

三、层次聚类

从最朴素的观点来看，对一个样本集聚类的过程就是将相似的样本聚在一起的过程，也是将不相似的样本区分开的过程。层次聚类用递归的、层次化的思想来具体实现这一过程。

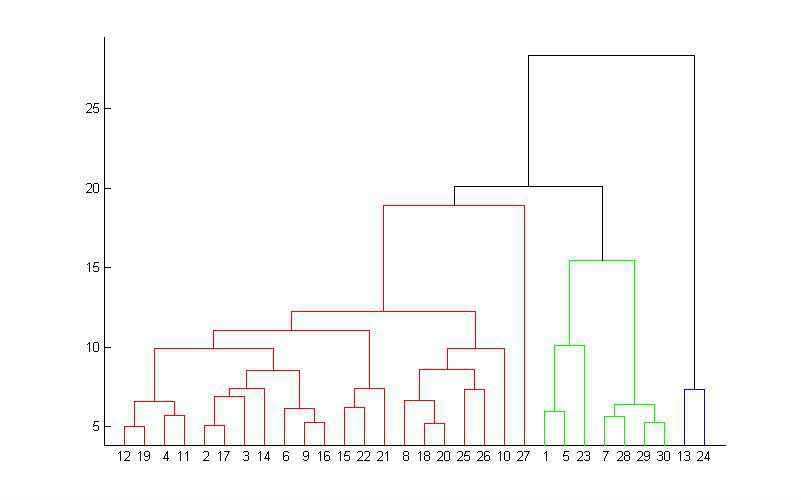


图3 层次聚类示意图

如图3为层次聚类的示意图，可以看出，层次聚类把样本的聚类结构看成是一个层次结构。在实际操作中，采用“自底向上”或者“自顶向下”的策略建立层次结构。下面分别介绍这两种策略。

（1）AGNES：基于自底向上的聚合策略。

它先将样本集中每个样本都作为一个初始聚类簇，然后迭代，每一次找出距离最近的两个簇进行合并，直至合成整个样本集，或者达到预设的簇个数。

那么如何找出两个最近的簇呢？这就需要定义“簇与簇之间的距离”，一般有三种定义方式：最大距离，最小距离和平均距离，表达式分别如下：







给定了簇之间距离的定义，AGNES算法就能够顺利执行啦。

（2）基于最小生成树（MST）的自顶向下的层次聚类：

最小生成树是一种图模型。图模型将样本集对应成一个图，样本集中的样本对应于图中的节点，样本间的距离对应于相应节点之间的边的权重。最小生成树是在样本集的图的基础上，找到连接所有的节点，并且使边的权重之和最小的图——它是一个树结构。

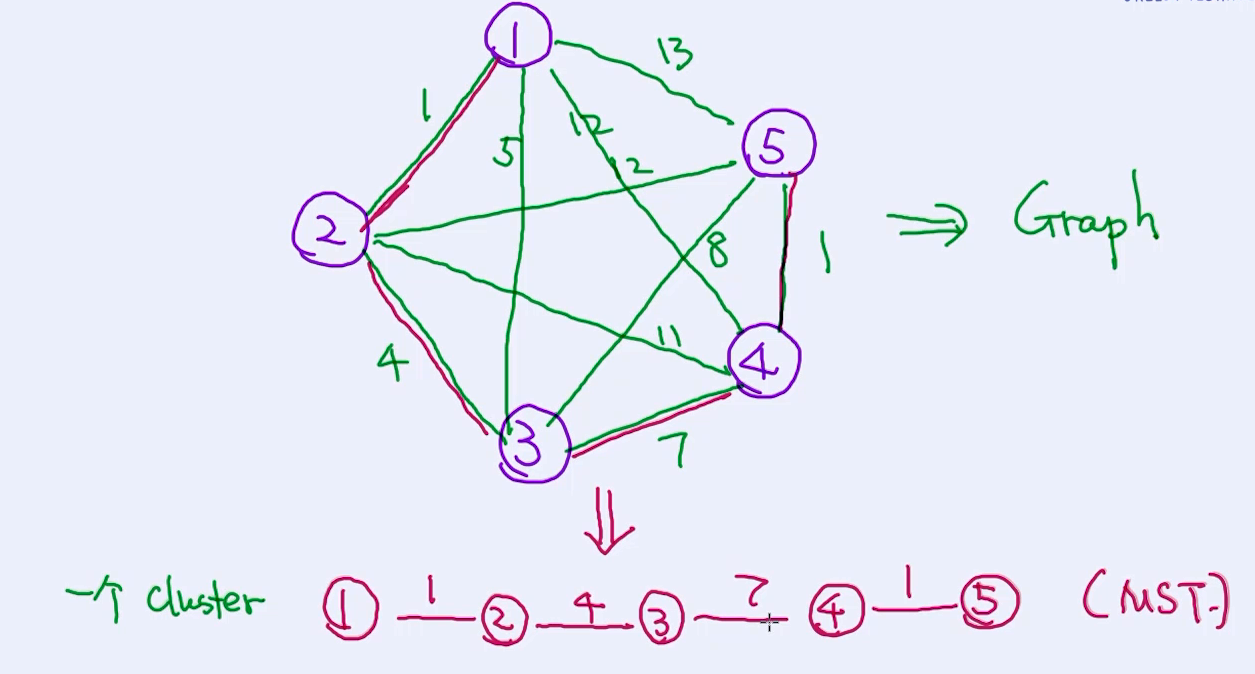


图4 最小生成树示意图

对样本集求出最小生成树以后，如何利用最小生成树来划分簇呢？

首先，最小生成树对应于整个样本集，它作为一个原始的簇，然后切断最小生成树中最大权重对应的边，将最小生成树分成两部分，每一部分对应于一个簇；然后不断重复这个过程，就递归地将样本集划分成多个簇。

可以看出，层次聚类不需要像KMeans那样，事先指定聚类的簇的个数和聚类中心，它首先形成一个代表样本集的聚类结构的层次结构，然后我们再从这个结构中决定具体的划分成几个簇。

四、谱聚类

谱聚类的基本思路是利用图论中的“切图”进行聚类，它与基于最小生成树的层次聚类具有相似之处。

首先，它将数据建模成一个图：将每个样本看成是一个“顶点”，顶点之间连接的“边”上的权值设置为样本之间的相似度。这样，聚类问题就变成了“切图”问题：将整个样本集对应的图分割成若干子图，以满足关于权值的某些条件。

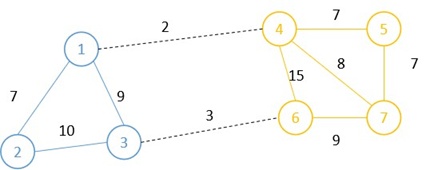


图5 “切图”示意图

如图5为“切图”的示意图，显然，切掉1-4边和3-6边能够得到最优的切图。

**1、图理论的基本概念**

设有一个顶点数为N的图（对应于样本集中样本个数为N），有：

（1）无向图：边上的权重与两顶点的方向无关，即

（2）顶点的“度”：与顶点相连的所有边的权重之和，即

（3）图的度矩阵：是一个对角矩阵，对角线上的值为每个顶点的度值



（4）图的邻接矩阵：记为W，矩阵的第i行第j列的值为第i个顶点与第j个顶点的权重

（5）拉普拉斯矩阵：L=D－W。

**2、谱聚类的思路**

根据图理论的基本概念和谱聚类的基本思想，我们来研究如何实现谱聚类：

（1）首先对样本集构造相应的图：

构建样本集的图时，通常不会保留样本集中所有样本之间的边，因为那太多了；并且权值太低的边对整个问题来说影响极小。因此每个顶点通常只保留权重最

高的k条边（k值是事先选定的）。也就是说，每个样本只与其临近的几个顶点相连。

（2）然后确定切图的目标函数：

最自然的想法是，使切图切掉的边的权重尽量小（因为权重代表相似度，切掉的边的权重越低，表明分割的两个子图的相似度就越低），对于二分切图，其目标函数为：



其中，A和B是两个子图，表示连接顶点i和顶点j的权重。可以看出，二分切图的目标是使两个子图之间所有边（也就是切掉的边）的权重之和最小。

然后定义多分切图的目标函数为：



其中，是的补集。

但是这样定义目标函数有一个问题：基于该目标函数最小化的切图会倾向于使其中一个或几个子图的规模尽量小，甚至成为一个单个的顶点（因为图的规模越小，它与其补集的连接的边越少，这些边的权重之和就越可能比较小）。这与我们聚类的最根本的目标不符。这个问题与决策树的基于信息增益的目标函数的定义问题类似，其改进的思路也类似。改进的具体方法主要有两种：Ratio-cut和N-cut：



其中是子图的顶点数（或者说子集的元素个数）。可以看出，Ratio-cut消除了子图的规模对我们的最根本的目标的影响，因此是一个比较合适的目标。



其中，。显然，vol也可以很好地衡量一个子图的规模。

（3）然后，对目标函数进行转换，转换成适合求解的形式：（以Ratio-cut为例）

第一步，将目标函数转换成矩阵表示形式，先从二分切图问题入手，设有图V，其

其拉普拉斯矩阵为L，我们要求其一个最优的二分切图，其优化问题为：



其中V是整个图。对子图A，定义指示向量：



其每个分量对应于整个图的一个顶点，取值为：



则容易证明：



其中L是整个图的拉普拉斯矩阵，表示图V的顶点数量。这样，就把二分切图的目标函数写成了矩阵形式。

此外，容易证明任何指示向量具有如下两个性质：

（a）f与全1向量正交，即；

（b）f的模等于，即

为了对目标函数进行扩展，我们根据f的性质，将优化问题写成如下的约束优化问题：



该目标函数是一个离散的优化问题，具有NP难度。因此，我们将优化问题放松为一个连续问题：



（4）最后，求解目标函数：

上面得到的连续优化问题，是求解一个最优的向量f，这里省略推导步骤，直接给出结论：其最优解为拉普拉斯矩阵L的第二小的特征值对应的特征向量（L的最小的特征值是0，对应的特征向量是1）。

但是别忘了，刚才我们是将离散优化问题转换成了连续优化问题。这里我们求出了连续优化问题的最优向量f，但是f不一定具有指示向量的意义（也就是说f不一定对应于图V的一个切分），因此必须要将f再转化成具有指示向量意义的向量。

那么如何实现呢？

一个合理的方法是通过聚类来实现。即将之前求出的连续优化问题的最优解f的每个分量看成是一个样本，然后将这些样本聚成两类C和，根据聚类结果，决定分量的值（进而决定子图的划分）：



对于多分切图问题，可以类比二分切图问题，定义指示向量，导出（连续）优化问题：



（注意，由于这里是多分切图，即多分类问题，所以优化目标不再是指示向量，而是指示矩阵（H））。

多分切图的优化问题的求解过程也与二分切图类似。注意，在多分切图的连续优化问题中，最优的指示矩阵H的值是L的最小的K个特征值对应的特征向量构成的矩阵。

**3、谱聚类的流程**

根据谱聚类的思路，可以得到具体的谱聚类算法流程。这里省略流程。