2. K-means

1.原理

1.1 算法过程

1.初始化k个质心，作为初始的k个簇的中心点，k为人工设定的超参数；

2.然后对于每一个样本分别计算其k个质心的距离，并将样本点归于最近的一类中。

3.重新计算质心，即将每一类中的所有点取平均值。

4.重复上述过程直到达到预定的迭代次数或质心不再发生明显变化

1.2损失函数

$$SSE = \sum_{k=1}^{K} \sum_{p \in C_k} |p - m_k|^2$$

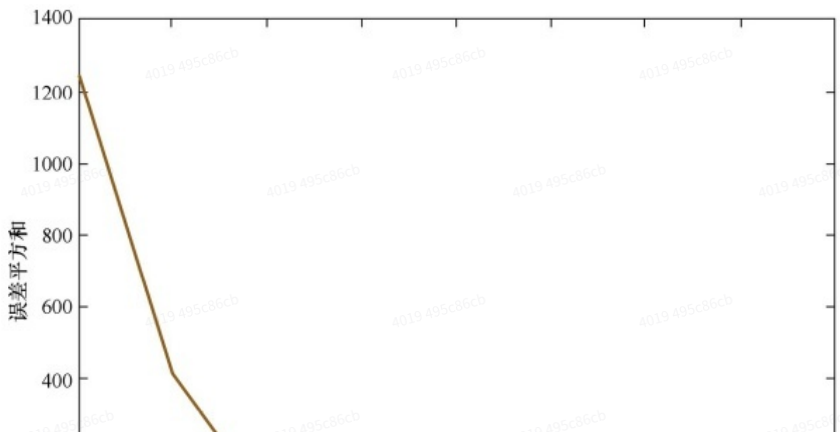
其中，K是聚类数量，p是样本，mk是第k个聚类的中心点。SSE越小，说明样本聚合程度越高。

1.3怎么确定聚类数量K（聚类如果不清楚有多少类，有什么方法？）

和评估分类或回归的方式一样，选择某个metric或某些metrics下最好的k，例如sse（其实就是kmeans的损失函数了），轮廓系数.

k的大小调参，手工方法，手肘法为代表。

手肘法其实没什么特别的，纵轴是聚类效果的评估指标，根据具体的问题而定，如果聚类是作为单独的任务存在则使用sse或轮廓系数这类无监督的metric作为纵坐标，然后找到metric最好并且k最小的结果对应的k为最终的选择；（我们说的学习曲线）



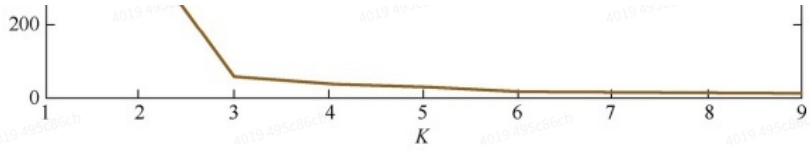


图5.3　K均值算法中K值的选取：手肘法

1.4k-means的缺点，怎么解决?

1. 对异常样本很敏感，簇心会因为异常样本被拉得很远

解决方法即做好预处理，将异常样本剔除或修正

2. k值需要事先指定，有时候难以确定

解决方法即针对k调参。

3. 只能拟合球形簇

对于流形簇等不规则的簇或是存在簇重叠问题的复杂情况等，效果较差。

解决方法，换算法。

4. 无法处理离散特征，缺失特征

3.DBSCAN

各个目标簇是由一群密集的数据点构成的，这些密集数据点构成的簇被稀疏区域分割，基于密度的聚类的最终目的是从稀疏区域中发现簇，并将稀疏区域中的孤立的样本点当作异常点；

基于密度的聚类，通过引入密度的概念，打破了基于划分的聚类的限制，可以拟合任意形状的簇。

1.原理

DBSCAN的算法流程：

Step1.（定义超参，半径，邻域下的最少样本）

定义超参数eplison表示**半径（即eplison-邻域）**，

minpoints表示eplison-**邻域下的最少样本数量**

Step2.（遍历样本）

从数据集中依次选择一个样本点进入后续的处理，直到所有样本遍历完毕；

Step3.确定核心点

对于选定的样本点s，做3件事：

（1）标记为“**已访问**”；

（2）**计算以样本s为中心，eplison为半径的内的样本数量neighbor_points**

（3）

如果neighbor_points中的points的数量小于minpoints，则样本s标记为**噪声点**，进入step2继续遍历；**（邻域内样本点过于少，就是离散点，标记位噪音）**

如果neighbor_points中的points的数量大于等于minpoints，则样本s标记为**核心点**。

为核心点s生成一个聚类簇**cluster_s**，此时cluster_s是一个**空簇**。neighbor_points中的每一个point都是样本s的**边界点**，他们和核心点s之间的关系是**密度直达**的；

（只要是在核心点半径内的点都是密度直达的，否则就是不可达）

（但是并非是只要在核心点的密度可达范围内就是该核心点对应簇的成员，原因是它有可能也是其他核心点的密度可达的点，这时候就是先到先得，标记就会发生作用，只能添加未标记的样本到自己的簇内，已经标记的样本就不行）

Step4. 确定核心点对应的簇内样本

先将核心点s放入cluster_s中，然后遍历neighbor_points中的每一个neighbor：

如果neighbor被访问过，则这个neighbor只有可能有两种情况：

1）**它是一个噪声点，不属于之前的任何一个cluster**，则加入cluster_s中，不再是噪声点了，而是核心点s的边界点（与核心点s**密度直达**）；

2）它是其它cluster内的点，则别人家的东西不能动，跳过；

如果neighbor没被访问过，显然neighbor不是噪声点（都没访问过怎么可能被标注），也不可能是别的cluster内的点（都没被访问过怎么可能是别的clsuter的点），则：

1）先标记为**“已访问”**，免得被别的cluster抢走了；

2）判断neighbor是否为核心点，

如果不是，则neighbor直接并入cluster_s中，

如果是，就与该neighbor的簇内样本进行合并，得到一个更大的簇内样本。

即**neighbor_points=neighbor_points+neighbor的neighbor_points（注意这里是递归计算的），然后继续step4进行遍历**；

2.怎么解决kmeans缺点的

1. K-means对异常样本很敏感

DBSCAN定义了密度的计算方式，不涉及到任何的平均这种鲁棒性较差的计算方式，对异常样本不敏感，还能检测异常样本呢；

2. K值需要事先指定；

DBSCAN不需要指定簇的数量；算法迭代过程中自然而然产生最优的k个聚类簇；

3. 只能拟合球形簇，

基于密度的聚类可以拟合任意形状的簇，这也归功于密度的计算方式，基于密度的聚类本身不对聚类簇的形状有任何的假设；

4. 无法处理离散特征，缺失特征：缺失特征要插补，离散特征可以换离散特征的距离度量方法，

基于密度的聚类算法可以灵活搭配各种不同的distance的度量方式；