山东大学计算机科学与技术学院

数据挖掘实验报告

实验题目： Homework 3: 聚类算法		学号： 201834882
日期：2018.12.23	班级： 18 级专硕班	姓名： 谢升军
Email：1640893020@qq.com		

实验目的：测试 sklearn 中 K-means，Affinity Propagation,Mean_shift,Spectral,AgglomerativeClustering,DBSCAN,Birch 等七种聚类算法在 tweets 数据集上的聚类效果。并用 NMI 作为评价指标。

硬件环境：win10 系统的 8g 内存计算机

软件环境：Anaconda3，Spyder

实验步骤与内容：

一、 算法

1、K-Means：

①需要选取 k 个初始质心作为初始 cluster 对每个样本点；

②计算得到距其最近的质心，将其类别标为该质心所对应的 cluster； 重新计算 k 个 cluster 对应的质心；

③重复②直到质心不再发生变化。

2、Affinity Propagation：

AP 聚类算法是基于数据点间的"信息传递"的一种聚类算法。与

k-均值算法不同，AP 算法不需要在运行算法之前确定聚类的个数。AP 算法寻找的聚类中心点是数据集合中实际存在的点，作为每类的代表。

3、Mean-Shift：

Mean-Shift 聚类法可以自动确定 k 的个数，下面简要介绍一下其算法流程：

①随机确定样本空间内一个半径确定的高维球及其球心;

②求该高维球内质心，并将高维球的球心移动至该质心处;

③重复②，直到高维球内的密度随着继续的球心滑动变化低于设定的阈值，算法结束。

4、Spectral：

谱聚类主要思想是把所有的数据看做空间中的点，这些点之间可以用边连接起来。距离较远的两个点之间的边权重值较低，而距离较近的两个点之间的边权重值较高，通过对所有数据点组成的图进行切图，让切图后不同的子图间边权重和尽可能的低，而子图内的边权重和尽可能的高，从而达到聚类的目的。

5、Ward hierarchical：

层次聚类试图在不同的“层次”上对样本数据集进行划分，一层一层地进行聚类。自底向上的凝聚方法（agglomerative hierarchical clustering）是先将所有样本的每个点都看成一个簇，然后找出距离最小的两个 cluster 进行合并，不断重复到预

期 cluster 或者其他终止条件。其中，ward 是一种链接方式。

Agglomerative：

使用自底向上的凝聚方法的层次聚类，除 ward 链接方法外还有 complete 和 average。

6、DBSCAN：

DBSCAN 是一种基于密度的聚类算法，这类密度聚类算法一般假定类别可以通过样本分布的紧密程度决定。同一类别的样本，他们之间的紧密相连的，也就是说，在该类别任意样本周围不远处一定有同类别的样本存在。通过将紧密相连的样本划为一类，这样就得到了一个聚类类别。通过将所有各组紧密相连的样本划为各个不同的类别，则我们就得到了最终的所有聚类类别结果。

7、BIRCH

BIRCH 的全称是利用层次方法的平衡迭代规约和聚类（Balanced Iterative Reducing and Clustering Using Hierarchies），名字实在是太长了，不过没关系，其实只要明白它是用层次方法来聚类和规约数据就可以了。刚才提到了，BIRCH 只需要单遍扫描数据集就能进行聚类，那它是怎么做到的呢？

BIRCH 算法利用了一个树结构来帮助我们快速的聚类，这个数结构类似于平衡 B+树，一般将它称之为聚类特征树(Clustering Feature Tree，简称 CF Tree)。这颗树的每一个

节点是由若干个聚类特征(Clustering Feature，简称 CF)组成。每个节点包括叶子节点都有若干个 CF，而内部节点的 CF 有指向孩子节点的指针，所有的叶子节点用一个双向链表链接起来。

二、 算法参数

1、K-means:

--- n_clusters: 指定 K 的值

--- max_iter: 对于单次初始值计算的最大迭代次数

--- n_init: 重新选择初始值的次数

--- init: 制定初始值选择的算法

--- n_jobs: 进程个数，为-1 的时候是指默认跑满 CPU

2 AffinityPropagation:

--- damping: 衰减系数，默认为 0.5

--- convergence_iter: 迭代次后聚类中心没有变化，算法结束，默认为 15

--- max_iter: 最大迭代次数，默认 200

--- copy: 是否在元数据上进行计算，默认 True，在复制后的数据上进行计算

--- preference: S 的对角线上的值

--- affinity: 相似度(S)矩阵，默认为 euclidean（欧氏距离）矩阵

3 MeanShift:

--- bandwidth: float，高斯核函数的带宽，如果没有给定，则使用 sklearn.cluster.estimate_bandwidth 自动估计带宽

--- seeds: array，我理解的 seeds 是初始化的质心，如果为 None 并且 bin_seeding=True，就用 clustering.get_bin_seeds 计算得到 --- bin_seeding: boolean，在没有设置 seeds 时起作用，如果 bin_seeding=True，就用 clustering.get_bin_seeds 计算得到质心，如果 bin_seeding=False，则设置所有点为质心

--- min_bin_freq: int，clustering.get_bin_seeds 的参数，设置的最少质心个数

4 SpectralClustering:

--- n_clusters: 代表我们在对谱聚类切图时降维到的维数，同时也是最后一步聚类算法聚类到的维数。也就是说 scikit-learn 中的谱聚类对这两个参数统一到了一起。简化了调参的参数个数。虽然这个值

是可选的，但是一般还是推荐调参选择最优参数。

--- affinity: 也就是我们的相似矩阵的建立方式。可以选择的方式有三类，第一类是 'nearest_neighbors'即 K 邻近法。第二类是 'precomputed'即自定义相似矩阵。第三类是全连接法，可以使用各种核函数来定义相似矩阵，还

可以自定义核函数。最常用的是内置高斯核函数'rbf'。其他比较流行的核函数有‘linear’即线性核函数,

‘poly’即多项式核函数, ‘sigmoid’即sigmoid核函数。如果选择了这些核函数， 对应的核函数参数在后面有单独的参数需要调。 affinity 默认是高斯核'rbf'。一般来说，相似矩阵推荐使用默认的高斯核函数。

--- gamma: 如果我们在affinity参数使用了多项式核函数'poly'，高斯核函数‘rbf’， 或者'sigmoid'核函数，那么我们就需要对这个参数进行调参。

--- n_neighbors: 如果我们 affinity 参数指定为'nearest_neighbors'即K邻近法，则我们可以通过这个参数指定KNN 算法的 K 的个数。默认是 10.我们需要根据样本的分布对这个参数进行调参。如果我们affinity不使用'nearest_neighbors'，则无需理会这个参数。

--- n_init: 即使用 K-Means 时用不同的初始值组合跑K-Means 聚类的次数，这个和K-Means 类里面n_init 的意义完全相同，默认是 10，一般使用默认值就可以。如果你的n_clusters 值较大，则可以适当增大这个值。

5 AgglomerativeClustering:

--- n_clusters: 一个整数，指定分类簇的数量

--- connectivity: 一个数组或者可调用对象或者 None，用于指定连接矩阵

--- affinity: 一个字符串或者可调用对象，用于计算距离。可以为:

'euclidean', 'l1', 'l2', 'mantattan', 'cosine', 'precomputed', 如果 linkage='ward'，则 affinity 必须为'euclidean'

--- linkage: 一个字符串，用于指定链接算法

'single': 单链接single-linkage,采用dmindmin

'complete': 全链接 complete-linkage 算法，采用 dmaxdmax

'average': 均连接 average-linkage 算法，采用 davg

'ward': 最小化被合并的 clusters 的方差

6 DBSCAN:

--- min_samples: DBSCAN 算法参数，即样本点要成为核心对象所需要的 ϵ-邻域的样本数阈值。默认值是 5,一般需要通过在多组值里面选择一个合适的阈值。通常和 eps 一起调参。在 eps 一定的情况下， min_samples 过大，则核心对象会过少，此时簇内部分本来是一类的样本可能会被标为噪音点，类别数也会变多。反之 min_samples 过小，则会产生大量的核心对象，导致类别数过少。

7 Birch:

--- threshold: 即叶节点每个 CF 的最大样本半径阈值

T，它决定了每个 CF 里所有样本形成的超球体的半径阈值。一般来说 threshold 越小，则 CF Tree 的建立阶段的规模会越大，即 BIRCH 算法第一阶段所花的时间和内存会越多。但是选择多大以达到聚类效果则需要通过调参决定。默认值是 0.5.如果样本的方差较大，则一般需要增大这个默认值。

--- branching_factor: 即 CF Tree 内部节点的最大 CF 数 B，以及叶子节点的最大 CF 数 L。这里 scikit-learn 对这两个参数进行了统一取值。也就是说，branching_factor 决定了 CF Tree 里所有节点的最大 CF 数。默认是 50。如果样本量非常大，比如大于 10 万，则一般需要增大这个默认值。选择多大的 branching_factor 以达到聚类效果则需要通过和 threshold 一起调参决定

--- n_clusters: 即类别数 K，在 BIRCH 算法是可选的，如果类别数非常多，我们也没有先验知识，则一般输入 None，此时 BIRCH 算法第 4 阶段不会运行。但是如果我们有类别的先验知识，则推荐输入这个可选的类别值。默认是 3，即最终聚为 3 类。

三、 实验结果

1-K-means ： 0.7887

2-AffinityPropagation：0.7831

3-MeanShift：0.7265

4-SpectralClustering：0.6839

5-AgglomerativeClustering：0.9004

6-DBSCAN：0.7542

7-Birch：0.7861

可以发现聚类效果 AgglomerativeClustering 最好，SpectralClustering 的效果最差。

结论分析与体会：

要想取得好的实验结果必须耐心的调整参数。