|  |  |
| --- | --- |
| **自然语言处理及应用**  **实验报告** | |
|  | |
|  |  |
| **名称** | 基于GMM和k-means的文本聚类 |
| **姓名** | 郭淇 |
| **班级** | 软件72班 |
| **学号** | 2174111345 |
| Email | 1325930068@qq.com |
| **日期** | 2021-4-8 |

1. **实验目的**

从一个有20个类别，每个类别1000个文本的20\_newsgroups数据集中抽取了三个（'rec.motorcycles', 'sci.electronics', 'talk.politics.guns'）（选择这三个类的原因是它们的主题差异比较大，聚类效果会比较好），采用GMM和k-means分别进行聚类，然后采用混淆矩阵给出聚类情况，并通过调整兰德系数、互信息和准确率进行评价比较（这三个指标都是越接近1越好）。

1. **实验环境**

Pycharm

Windows10

Python3.8

1. **实验方法**

从已有的20\_newsgroups数据集中抽取三个类别得到20\_newsgroups，首要步骤是遍历3\_newsgroups中所有的文本文件。此处可以调用os模块，利用os.walk的遍历方案进行文件的遍历。

得到文本文件内容的同时，需要对数据进行预处理，我的处理方案是首先剔除文本中的特殊字符，将非26个字母和’的字符用空格替换，对文本根据空格进行分词，对分词结果运用网上找来的停用词表进行筛选，停用词表采用百度停用词，然后得到word.txt为分词后的结果。

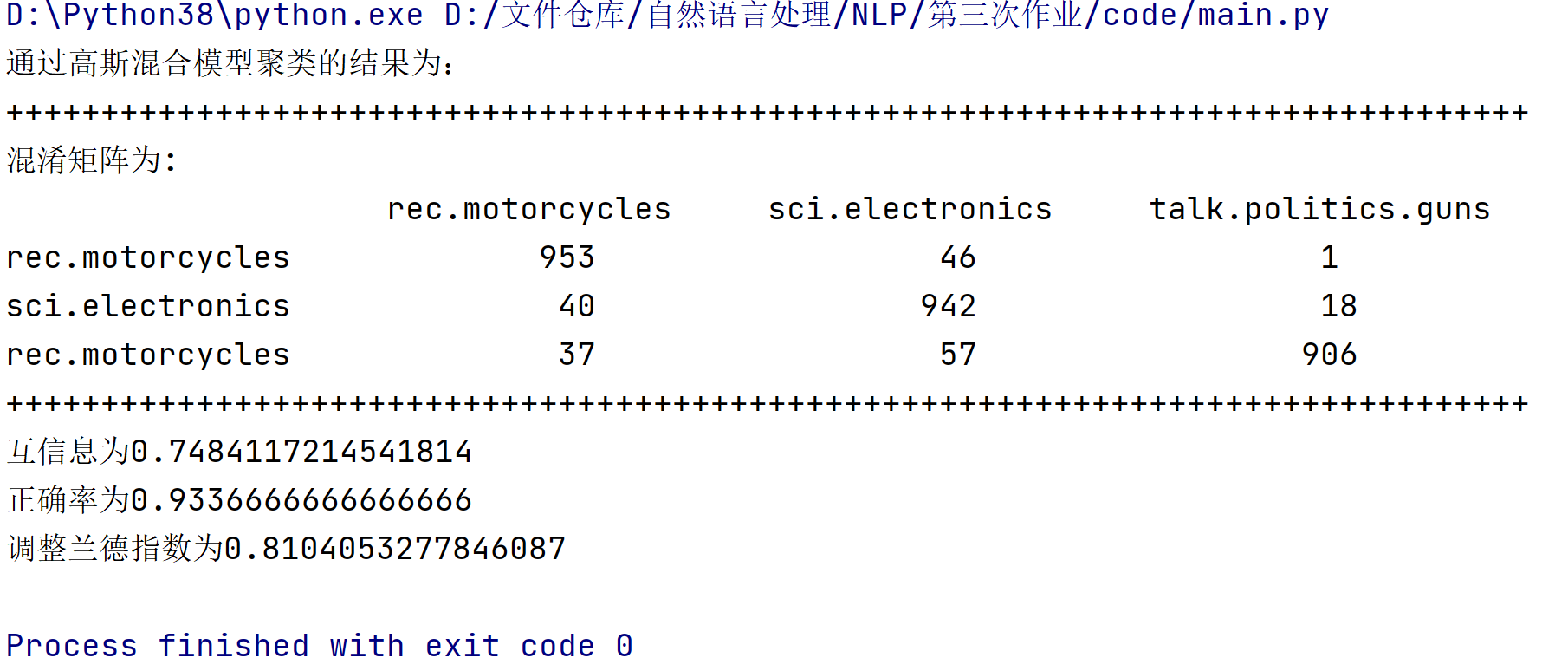
接着采用tfidf编码方式将文本数据转换为词向量，但是此词向量的维度为三万多维（是一个稀疏矩阵，维度太高，计算协方差不方便），采用了PCA将词向量的维度降到了2000维(保留了百分之九十五的信息)，将结果保存为word\_vectors.txt。

然后再使用GMM算法和K-means算法进行聚类。

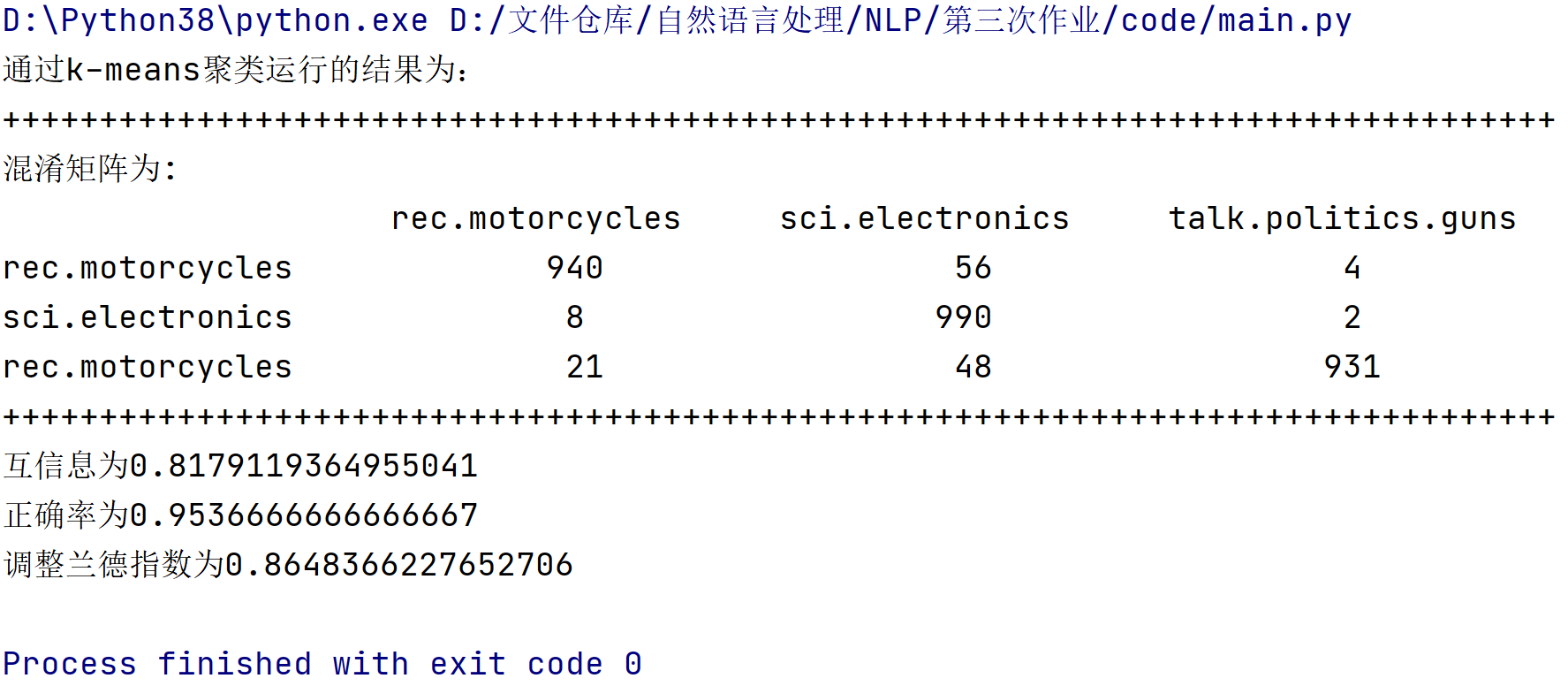
最后采用调整兰德系数、互信息和准确率进行评估。

1. **实验结果**

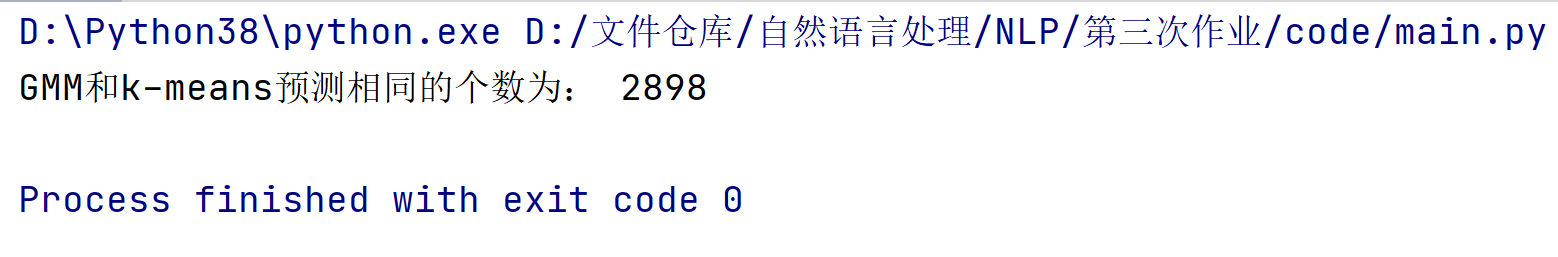
GMM的运行结果为：



k-means的结果为：



对于所有的文本聚类结果，k-means和GMM预测一样标签的个数为：



结果分析：

（1）从点的划分情况来看，GMM和k-means的聚类结果有较强的的相似性，在3000个文本数据中，它们对2898文本的聚类结果是一样的。

（2）从评价指标的各方面来看，整体上，k-means模型在文本聚类上的性能比GMM要好，一个可能的原因是文本数据服从的是隐狄利克雷分布，而不是严格的高斯分布。

（3）k-means和GMM受初始化的影响都比较大，我们需要采用一些启发式的方法进行初始化，避免出现聚类较差的结果（如k-means的初始聚类中心要尽可能远一点。）

1. **遇到问题及解决思路**

（1）采用tf-idf转换为词向量后，维度太高，为三万多维，在GMM中计算协方差矩阵时出现了数据存储量太大的情况，观察词向量发现其为稀疏矩阵（大量元素都为0），因此采用的解决方法是利用PCA将维度降到2000维，此时保留了原始数据95%的信息。

（2）在计算逆矩阵时出现了奇异值矩阵的错误，采用的解决方法是使用SVD得到矩阵的近似逆矩阵（np.linalg.pinv）。

（3）k-means和GMM受初始值的影响过大，每次求解的结果差异太大，此时采用的方法是固定初始化的随机性（np.random.seed(n)），通过固定随机种子，使得代码可以复现，经过调参找到最好的随机种子。