基于“智慧政务”中的文本挖掘应用研究

摘 要

随着 5G 时代的到来以及大数据、云计算、人工智能的普及，网络问政平台已然成为政府了解民情民意的主要渠道。但是，各类民意相关的文本数据在不断攀升，这无疑极大加剧了人工留言划分和热点整理的相关部门的工作量。因此，建立自然语言处理和文本挖掘的服务系统，对群众留言进行高效、准确的分类和热点问题快速、有效的挖掘，对提高相关部门的服务效率和服务水平，将起到积极的推动作用。

本文旨在利用自然语言处理技术和文本挖掘方法，建立相关的分类模型、进行对民意热点问题的挖掘和政府留言的答复意见的评价。

针对问题一，对附件 2 给出的数据进行探索分析，发现附件 2 有七个一级分类标签，首先需要对留言主题和留言详情的数据分别进行了预处理，把分好后的词绘制词云，通过词云，可以直观地看出每一个一级分类特征词的分布情况和核心内容；再用向量化表示出来，构建朴素贝叶斯、KNN 算法和支持向量机 3 种分类模型去进行训练和测试，用 F-score 对 3 种分类模型进行评价，最后得出结论：朴素贝叶斯和支持向量机都具有不错的分类效果，有 85%以上的准确率，也出现了准确率高于 90%的情况。

针对问题二，对附件 3 的数据进行探索分析，发现有留言内容、留言时间、点赞数和反对数这四个关键信息，热度评价指标也是由这些参数共同决定。首先对留言主题的内容进行了预处理操作，再把每一个留言主题用向量表示出来，然后进行两两之间的相似度计算，本文用到的是余弦相似度算法，两个向量的夹角的余弦值越大，即相似度越高。最后，筛选出相似度大于 0.75 的问题进行归类，并根据定义的热点评价指标算出热度排名前 5 的热点问题，并给出对应的热点问题留言明细表。

针对问题三，对附件 4 进行探索分析，从答复的相关性、完整性和可解释性这三个角度定义答复意见质量评价指标，量化相关性、完整性和可解释性并计算得分，最后给出评价结果。

关键词：文本分类、朴素贝叶斯、支持向量机、自然语言处理、文本相似度

Abstract

With the advent of 5G era and the popularization of big data, cloud computing and artificial intelligence, the online political platform has become the main channel for the government to understand the people's conditions and public opinions. However, the text data related to all kinds of public opinions are constantly rising, which undoubtedly greatly intensifies the workload of the relevant departments of manual message division and hot topic sorting. Therefore, the establishment of a natural language processing and text mining service system, the efficient and accurate classification of public comments and the rapid and effective mining of hot issues will play a positive role in improving the service efficiency and service level of relevant departments.

The purpose of this paper is to use natural language processing technology and text mining methods to establish a relevant classification model, to carry out the mining of public opinion hot issues and to evaluate the comments of the government.

On problems, to explore analysis is given in annex 2 of the data found in annex 2 there are seven primary category labels, we first need to message subject and the message for details of data pretreatment, respectively draw points after word word cloud, through the word cloud, we can visually see the distribution of each level of classification of key and the core content; Reoccupy vectorization representation, to build simple bayesian, three kinds of KNN algorithm and support vector machine (SVM) classification model for training and testing, the F - score to evaluate three kinds of classification model, and finally draw the conclusion: naive bayes and support vector machine (SVM) have good classification effect, more than 85% accuracy, and the accuracy is higher than 90%.

For question 2, the data in annex 3 were explored and analyzed. It was found that there were four key information: message content, message time, thumb up number and opposition number. The heat evaluation index was also determined by these parameters. Firstly, the content of the message subject is preprocessed, and then each message subject is represented by a vector, and then the similarity between two is calculated. In this paper, the cosine similarity algorithm is used. The greater the cosine value of the Angle between two vectors is, the higher the similarity is. Finally, the problems with similarity greater than 0.75 were classified, and the top 5 hot spot problems were calculated according to the defined hot spot evaluation index, and the corresponding hot spot problem message list was given.

In view of question 3, annex 4 is explored and analyzed, and the quality evaluation index of reply comments is defined from the three aspects of relevance, completeness and interpretability of replies, and the relevance, completeness and interpretability are quantified, and the score is calculated. Finally, the evaluation result is given.

Key words: text classification, naive bayes, support vector machines, natural language processing, text similarity

目录

一、问题重述

1.1 问题背景

近年来，随着科学技术和互联网的高速发展，微信、微博、市长热线、阳光信箱等网络问政平台逐渐成为政府相关部门了解民意、汇聚民智、凝聚民气的重要渠道，使得民意民情更方便、更快捷、更有效地汇集到政府部门。但是，各类民意相关的文本数据在不断攀升，这无疑极大加剧了人工留言划分和热点整理的相关部门的工作量。随着 5G 时代的到来，大数据、云计算、人工智能等新兴技术也逐渐走进人们的生活，自然语言处理技术作为人工智能的一个重要领域也得到了飞速的发展，建立基于自然语言处理技术的智慧政务系统，将极大有利于社会治理，使政府解决民情民意更加高效，从而提高政府的管理水平和施政效率，对建立完善的城乡智慧政务系统有着十分重要的意义。

1.2 要解决的问题

（1）根据附件 2 的数据，建立关于留言内容的一级分类标签，并对所用的分类方法进行评价。
（2）根据附件 3 的数据，定义合理的评价热度指标，给出评价结果，挖掘出排名前五的热点问题，并给出相应的热点问题留言信息。
（3）根据附件 4 相关部门对留言的答复意见，从答复的相关性、完整性、可解释性对答复意见的质量进行合理的评价。

二、模型假设

根据实际的智慧政务系统，需要做出如下合理的假设：

1.假设附件所给的数据集都是完备可靠的；
2.假设群众留言内容都是真实存在的；
3.假设热点问题的热度与该类问题的留言数量和点赞数量呈正相关，即热点问题的留言数量越多热度排行越高，点赞数量越多热度排行越高。且不存在人为故意刷赞现象。

三、问题分析

3.1 问题一的分析

群众留言分类，传统的留言分类方法还是依靠人工根据经验处理，存在工作量大，效率低，差错率高等问题。针对问题一，对附件 2 给出的数据进行探索分析，发现附件 2 有七个一级分类标签，首先需要对留言主题和留言详情的数据分别进行了预处理，把分好后的词绘制词云，通过词云，可以直观地看出

每一个一级分类特征词的分布情况和核心内容；再用向量化表示出来，构建朴素贝叶斯、KNN 算法和支持向量机 3 种分类模型去进行训练和测试，最后用 F-score 对 3 种分类模型进行评价。

3.2 问题二的分析

热点问题挖掘，某一时段内群众集中反映的某一问题可称为热点问题，传统的热点问题归类还是依靠人工整理，效率低下，群众提出的问题无法及时解决。本文采用 python 工具，对附件 3 的留言主题进行预处理，把每一个留言主题用向量表示出来，计算两两之间的相似度。然后进行留言问题归类，并定义合理的热度评价指标，挖掘出热点问题，并给出评价结果。最后给出排名前 5 的热点问题表和热点问题留言详细表。

3.3 问题三的分析

答复意见评价，相关部门会对群众的留言问题进行评价，但是评价结果的质量参差不齐，导致有些群众所反映的问题无法及时解决。针对问题三，对附件 4 进行探索分析，从答复的相关性、完整性和可解释性这三个角度定义答复意见质量评价指标，并给出评价结果。

四、数据预处理方法介绍

4.1 正则表达式(Regular Expression)

由于附件的数据存在大量的标点符号和少量的特殊符号，以及根据题目题意选择是否去掉数字和英文。本文使用了正则表达式把这些字符去除掉，只保留中文文本，从而避免对分词结果的影响。

4.2 分词

由于在中文自然语言处理中，词是最小的能够独立活动的有意义的语言成分，词语之间没有明显的界限，所以进行中文自然语言处理通常是先将汉语文本中的字符串切分成合理的词语序列，然后再在此基础上进行分析处理。中文分词是中文信息处理的一个基础环节，已被广泛应用于中文文本处理、信息提取、文本挖掘等应用中。对于中文分词，分词工具有 jieba 分词、thulac、SnowNLP 等。本文采用 python 语言中的一个中文分词模块——jieba 分词，选择的理由是 jieba 分词实现原理比较完善，效果显著。Jieba 分词词库庞大，已有词库 60 万词以上，并且在不断更新。它支持自定义词典，许多情况下，需要对特定语境语意进行分词或者定义一些专有名词，这些词汇往往是词库没有的，jieba 分词很好地解决了这个问题，可以自行添加单词，确保分词有更高的准确率。

Jieba 分词支持三种分词模式[1]：

（1）精确模式，试图将句子最精确地切开，适合文本分析；

（2）全模式，把句子中所有的可以成词的词语都扫描出来，速度非常快，但是不能解决歧义；

（3）搜索引擎模式，在精确模式的基础上，对长词再次切分，提高召回率，适合用于搜索引擎分词。

Jieba 分词的算法实现[2]：

（1）基于前缀词典实现高效的词图扫描，生成句子中汉字所有可能成词情况所构成的有向无环图；

（2）最短路径匹配法，采用了动态规划查找最大概率路径，找出基于词频的最大切分组合；

（3）对于未登录词，采用了基于汉字成词能力的隐马尔科夫模型，使用了 **Viterbi** 算法。

4.3 去停用词（stop words）

由于在文本处理中，存在大量的无意义的词，通常是一些单字，单字母，符号，语气词等等，比如中文中的“我，你，的，了，地，吗，啦，哈哈哈”等，这些对文本没有意义的词即为停用词。在本文中，根据具体的题目要求(如问题一的留言分类模型)还将一些对文本无作用的词进行删除,如人名（例如：黄涛），地名（例如：西地省），量词（例如：一个），副词（例如：或者）等词进行删除并添加到停用词表，进行去停用词的操作，从而避免它对文本，特别是短文本，产生较大的影响。

五、问题求解

5.1 问题一的求解过程

5.1.1 探索分析

打开附件 **2**，不难发现有留言主题、留言详情和一级分类这三个重要的信息。往下读取一级分类，发现了一共有 **7** 个一级分类标签，分别是城乡建设、环境保护、交通运输、教育文体、劳动和社会保障、商贸旅游、卫生计生七大类，本文分别就留言主题和留言详情建立两次分类模型去训练和测试分类结果。下面将为建立关于留言主题和留言详情的一级标签分类模型做准备，接下来做数据的预处理操作。

5.1.2 数据清洗

首先把附件 **2** 的留言主题和留言详情的数据分别导入 **python** 中，打印出来发现文本存在大量的标点符号和少量的特殊符号，如果不去除，这对分词的结果影响是很大的。如下图：

```
0                    A市西湖建筑集团占道施工有安全隐患
1              A市在水一方大厦人为烂尾多年，安全隐患严重
2                  投诉A市A1区苑物业违规收停车费
3            A1区蔡锷南路A2区华庭楼顶水箱长年不洗
4                A1区A2区华庭自来水好大一股霉味
                          ...
9205                 两孩子一个是一级脑瘫，能再生育吗？
9206    B市中心医院医生不负责任，做无痛人流手术后结果还是活胚芽
9207                 西地省二胎产假新政策何时出台？
9208                      K8县惊现奇葩证明！
9209              请问J4县卫计委社会抚养费到底该交多少钱？
Name: 留言主题, Length: 9210, dtype: object
```

图 1-1 留言主题数据的部分展示

本文使用正则表达式去除这些对文本无意义的符号，另一方面，对于第一题群众留言分类而言，需要选取对文本有代表性的特征词，而文本中出现的地点，数字，英文（如：A 市、K8 县、A1 区等）对文本分类没有多大实际的意义，不是本题想要的特征词。因此，只需要保留中文文本，并在下一个步骤自定义停用词表去除地点的中文词语（如：市、区、县、西地省等），算法如下：

```
data_theme = data_new['留言主题'].apply(lambda x:
                            re.sub("[^\u4E00-\u9FD5]+",'',x))

data_content = data_new['留言详情'].apply(lambda x:
                          re.sub("[^\u4E00-\u9FD5]+",'',str(x)))
```

打印结果如下：

```
0                    市西湖建筑集团占道施工有安全隐患
1              市在水一方大厦人为烂尾多年安全隐患严重
2                  投诉市区苑物业违规收停车费
3            区蔡锷南路区华庭楼顶水箱长年不洗
4                区区华庭自来水好大一股霉味
                          ...
9205                 两孩子一个是一级脑瘫能再生育吗
9206    市中心医院医生不负责任做无痛人流手术后结果还是活胚芽
9207                 西地省二胎产假新政策何时出台
9208                      县惊现奇葩证明
9209              请问县卫计委社会抚养费到底该交多少钱
Name: 留言主题, Length: 9210, dtype: object
```

图 1-2 使用正则表达式后的留言主题数据的部分展示

5.1.3 分词

由于在中文自然语言处理中，词是最小的能够独立活动的有意义的语言成

分，词语之间没有明显的界限，所以进行中文自然语言处理通常是先将汉语文本中的字符串切分成合理的词语序列，然后再在此基础上进行分析处理。Jieba 分词对于长文本，分词效率会比较高，本文采用 jieba 完整的技术路线和 python 强大的工具库，实现对长文本的分词操作。在 python 中导入 jieba 函数，在分词实现过程中，发现文本存在一些专有词汇，这些往往是 jieba 词库没有的，虽然 jieba 对新词有不错的辨别能力，但单独添加自定义词典以确保有更高的准确率。因此，本题加入了自定义词典 jieba.load_userdict(fenci)，自定义词典里有县卫计委，市中心医院，无痛人流手术等，分词的代码如下：

```
jieba.load_userdict(fenci)
data_cut = data_theme.apply(jieba.lcut)
```

图 1-3

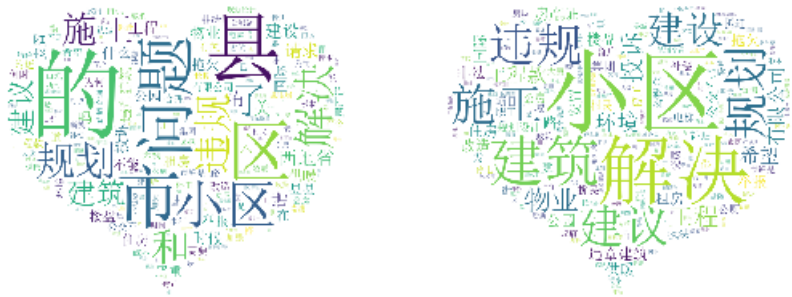
分完词的结果如下：

```
0                          [市，西湖，建筑，集团，占，道，施工，有，安全隐患]
1                        [市，在水一方，大厦，人为，烂尾，多年，安全隐患，严重]
2                                  [投诉，市区，苑，物业，违规，收停车费]
3                        [区，蔡锷，南路，区华庭，楼顶，水箱，长年，不洗]
4                                  [区区，华庭，自来水，好大，一股，霉味]
                                          ...
9205                     [两，孩子，一个，是，一级，脑瘫，能，再，生育，吗]
9206    [市中心医院，医生，不负责任，做，无痛人流手术，后，结果，还是，活，胚芽]
9207                           [西地省，二胎，产假，新，政策，何时，出台]
9208                                        [县惊现，奇葩，证明]
9209                     [请问，县卫计委，社会，抚养费，到底，该交，多少，钱]
Name: 留言主题, Length: 9210, dtype: object
```

图 1-4 分词后的留言主题数据的部分展示

5.1.4 去停用词

分完词后，发现还存在了大量对文本分类无意义的词，比如：量词（如一个，两，一股等）、地名（如市，区，县，西地省等）、语气词（的，了，呢，啊等）等等这些无意义的词出现的概率非常高，如果不去除会影响分类结果的准确率。为了更直观看出去除停用词的效果，使用了绘制词云的技术对比去除停用词前后的效果，如下图所示，图（a）是去除停用词之前的效果，图（b）是去除停用词之后的效果。显然，可以看出图（b）的特征词比图（a）的更具代表性，对分类的效果也会更好。



（a）　　（b）

由于不同的文本对应的语境语意也不同，对应无意义的词也会不一样，停用词表的选取不能疏忽大意。因此，本文对分完的词进行了词性分析，对一些无意义的词性进行删除并添加到了自定义停用词表，再进行去停用词的操作。

本文用了 20 分钟左右的时间对以下的词性进行分析[3]：

{'f', 'x', 'i', 'h', 'c', 'y', 'vg', 'ag', 'a', 'ng', 'u', 'nz', 's', 'vn', 'ad', 'n', 't', 'd', 'ug', 'tg', 'uj', 'zg', 'uz', 'g', 'v', 'an', 'q', 'l', 'ul', 'z', 'mq', 'p', 'nrfg', 'b', 'j', 'ud', 'ns', 'nt', 'm', 'nrt', 'nr', 'uv', 'df', 'k', 'r'}

n	名词
nr	人名
ns	地名
nt	机构团体
nz	其他专名
o	拟声词
p	介词
q	量词
r	代词
s	处所词
tg	时语素
t	时间词
u	助词
vg	动语素
v	动词
vd	副动词
vn	名动词
w	标点符号
x	非语素词

符号	词性
Ag	形语素
a	形容词
ad	副形词
an	名形词
b	区别词
c	连词
dg	副语素
d	副词
e	叹词
f	方位词
g	语素
h	前接成分
i	成语
j	简称略语
k	后接成分
l	习语词
m	数词
Ng	名语素

图 1-5

最后，把无意义的词添加到了停用词表（见附件），如方位词（'f'）、副词（'d'）、介词（'p'）等，如下图所示：

{'里', '中', '东侧', '北侧', '之前', '北', '以后', '之间', '东', '以下', '以上', '末', '内', '背后', '以前', '内侧', '南边', '上面', '中部', '最后', '沿江', '周边', '外围', '北岸', '前面', '右', '两边', '附中', '西侧', '两岸', '内', '南方', '沿线', '四期', '下游', '前后', '最近', '末端', '东南角', '西北角', '下', '两旁', '附', '北方', '西', '西部', '面前', '后面', '以北', '里面', '周围', '左岸', '附近', '南岸', '南', '后', '顶上', '之际', '前', '中间', '外来', '下方', '期间', '初中', '东半', '前上', '以南', '东北侧', '侧面', '南部', '下面', '上下', '内部', '上', '两间', '东南部', '东部', '之外', '两侧', '西北边', '外面', '初', '之下', '初中部', '沿街', '之后', '旁', '北面', '足部', '旁边', '中外', '外', '上游'}

方位词（'f'）

{'乱', '又', '真正', '高速', '经常', '一致', '同样', '尽早', '按时', '肆意', '一并', '频发', '最', '太', '再', '才', '进度', '一直', '便', '就', '最新', '竟然', '还', '将', '私自', '一次性', '至今', '同步', '一边', '重复', '正常', '约', '尽快', '屡屡', '未', '已经', '不', '必须', '只', '到底', '故意', '不当', '好好', '却', '己', '被迫', '长期', '也', '都', '随口', '随意', '不许', '大', '大同', '一幼'}

副词（'d'）

{'除了', '以', '从', '至', '为', '被', '在', '与', '对', '把', '向', '按', '因', '通过', '给', '关于', '由', '经过', '用'}

介词（'p'）

去除停用词操作代码如下：

```
with open(wordpath,'r') as f:
    f.seek(0)
    stop = f.read()
stop = stop.split()
stop = [' ']+stop
f.close()
```

图 1-6

```
data_after = data_cut.apply(lambda x:
                [i for i in x if i not in stop])
```

图 1-7

5.1.5 绘制词云

去除完停用词后，基于留言主题和留言详情分别绘制出 7 个一级分类的词云。词云是对文本中出现频率较高的特征词去进行可视化，在 python 中，使用 World-Cloud 库可以完成词云的绘制[4]，并且可以进行个性化的处理，本文绘制词云选择爱心作为背景图。代码如下：

```
import matplotlib.pyplot as plt
from wordcloud import WordCloud
from PIL import Image
import numpy as np
import itertools
num = pd.Series(list(itertools.chain(*list(data_after)))).value_counts()
pic = np.array(Image.open("D:\\taidibei\\love2.png"))
wc = WordCloud(font_path = 'C:\\Windows\\Fonts\\simsun.ttc',
               background_color = 'white',
               mask = pic)
wc2 = wc.fit_words(num)
plt.imshow(wc2)
plt.axis('off')
plt.show()
```

绘制出来的词云效果如下：

城乡建设

环境保护

交通运输

教育文体

劳动和社会保障　　商贸旅游

卫生计生

上图为基于留言主题的七个一级分类。
通过词云，可以清楚地看到每一个一级分类特征词的分布情况和核心内容。

5.1.6 文本向量化表示

由于计算机不能像人类一样去理解语言文字，而计算机可以理解向量的形式，需要把非结构化的数据转化成结构化的数据。因此把分好的特征词去向量化，本文选取的特征权重是词频 **TF**，所以需要计算每一个特征词的词频，最后用词频向量表示出来，便于下一步构建分类模型。代码如下：

```
'''文本向量化'''
tmp = data_after.apply(lambda x:' '.join(x))
print(tmp)
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer().fit(tmp)#生成词频词典
print(cv.vocabulary_)
cv_data = cv.transform(tmp)#词频词典，向量化
```

图 1-8

5.1.7 构建分类模型

文本分类的方法可以分成三大类，分别是基于统计的方法，基于连接的方法，还有一种是基于规则的方法。本文采用的是基于统计的方法，选择的分类算法有朴素贝叶斯、**KNN** 算法及支持向量机。

① 朴素贝叶斯原理[5]：

朴素贝叶斯算法利用贝叶斯定理在留言分类中有广泛应用，是文本分类最为精确的算法。它的原理[6]是设 $x=\{a_1,a_2,\cdots,a_m\}$ 为一个待分类项，每一个 **a** 为 **x** 的特征属性，有类别集合 $C=\{y_1,y_2,\cdots,y_n\}$ 计算每个 **y** 在 **x** 基础的概率分布

$P(y_1|x),P(y_2|x),\cdots,P(y_n|x)$ 如果 $P(y_k|x)=\max\{P(y_1|x),P(y_2|x),\cdots,P(y_n|x)\}$，则 $x\in y_k$

然后可以计算每个类别在训练集中的出现频率及每个特征属性划分对每个类别的条件概率估计，根据贝叶斯定理有

$$P(y_i\left|x\right.)=\frac{P(x\left|y_i\right.)P(y_i)}{P(x)} \quad (1)$$

分母对于所有类别为常数所以将分子最大化，又各特征属性是条件独立的,因此有

$$P(x\left|y_i\right.)P(y_i)=P(y_i)\prod_{j=1}^{m}P(a_j\left|y_i\right.) \quad (2)$$

② KNN 算法原理[6]：

KNN 算法的基本思路是在给定测试对象，然后计算它与训练集中每个对象的欧氏距离，选定距离最近的 k 个训练对象,根据它们的类别判定测试对象所属的类别。具体的算法步骤：计算测试数据与各个训练数据之间的距离;接着按照距离的递增关系进行排序;然后选取距离最小的 K 个近邻点并确定前 K 个点所在类别的出现频率;最后返回前 K 个点中出现频率最高的类别作为测试数据的预测分类。

③ 支持向量机原理[7]：

支持向量机（SVM）是基于统计学习理论的结构风险最小化原则基础上的机器学习算法。它属于二分类算法，可以支持线性和非线性的分类，本文采用的是线性的核函数。支持向量机的基本思想是寻找最优分类面使正负类之间的分类间隔最大。

假设训练集为$(x_i,y_i),i=1,2,\cdots,l,x\in R^n$， $y\in\{+1,-1\},l$ 为样本数，n 为维数。当线性可分时，最优分类超平面为：$wx+b=0$ （1）

此时分类间隔为$\frac{2}{\|w\|}$，当$\|w\|$值最小时，分类间隔最大。把问题描述为求解下述约束性优化问题：

$$\min\|w\|^2/2$$

$$s.t.\quad y_i(w\bullet x_i+b)-1\geq 0,i=1,2,\cdots,l \quad (2)$$

当训练集线性不可分时，需要引入非负松弛变量$\xi_i,i=1,2\cdots,l$，求解最优分类面问题为:

$$\begin{aligned}&\min\|w\|^2/2+C\sum_{i=1}^{l}\xi_i\\&s.t.\quad y_i(w\bullet x_i+b)\geq 1-\xi_i,\xi_i\geq 0,i=1,2,\cdots,l\end{aligned} \quad (3)$$

其中 C 为惩罚参数,C 越大表示对错误分类的惩罚越大。通过 Lagrange 乘子法求解上述优化问题,可得最优决策函数为:

$$f(x)=\mathrm{sgn}\left[\sum_{i=1}^{l}y_ia_i(x\bullet x_i)+b\right] \quad (4)$$

其中 a 为 Lagrange 系数。在对输入测试集 x 进行测试时,由式(4) 确定 x 的所属类别。上述优化问题的解必须满足：

$$a_i(y_i(w \bullet x + b) - 1) = 0 \qquad (5)$$

因此,对于多数样本 a_i 将为零,只有支持向量的 a_i 不为零,它们通常在全体样本中所占的比例很少。即仅需要少量支持向量即可完成正确的样本分类。

非线性分类问题时,支持向量机通过核函数 $K(x_i \bullet x_j)$ 将样本 x 映射到某个高维空间 H,然后在 H 中对原始问题进行线性划分。根据 Mercer 条件,此时相应的最优决策函数变为:

$$f(x) = \mathrm{sgn}[\sum_{i=1}^{l} y_i a_i K(x \bullet x_i) + b] \qquad (6)$$

算法实现代码：

```
#贝叶斯分类器
model_nb = MultinomialNB().fit(cv_train,y_train)
#knn算法分类器
model_knn = KNeighborsClassifier().fit(cv_train,y_train)
#支持向量机分类器
model_svc = LinearSVC().fit(cv_train,y_train)
```

图 1-9

5.1.8 模型评估

三种分类模型构建好之后，把留言主题和留言详情的数据分别分成训练集和测试集，即对留言主题的内容训练测试一次，对留言详情的内容也训练测试一次，本文选取 **85%**为训练集，**15%**为测试集。首先对训练集进行训练，接着再拿测试集的数据进行测试，得出分类的预测结果，最后还要对模型进行评估，选出分类效果最好的模型。通常使用 **F-Score** 对分类方法进行评价，计算公式如下：

$$F_1 = \frac{1}{n}\sum_{i=1}^{n}\frac{2P_iR_i}{P_i + R_i},$$

其中 P_i 为第 **i** 类的查准率，即准确率**(precision)**，R_i 为第 **i** 类的查全率,即召回率**(recall)**。

代码如下：

```python
from sklearn.naive_bayes import MultinomialNB      #朴素贝叶斯
from sklearn.svm import LinearSVC                  #支持向量机
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split

#

cv_train,cv_test,y_train,y_test = train_test_split(
                cv_data,data_new['一级标签'],
                test_size = 0.15,random_state=123
                )

#贝叶斯分类器
model_nb = MultinomialNB().fit(cv_train,y_train)
#knn算法分类器
model_knn = KNeighborsClassifier().fit(cv_train,y_train)
#支持向量机分类器
model_svc = LinearSVC().fit(cv_train,y_train)

# ''' '''
from sklearn.metrics import classification_report,confusion_matrix
y_pre_nb = model_nb.predict(cv_test)
y_pre_knn = model_knn.predict(cv_test)
y_pre_svc = model_svc.predict(cv_test)
print("朴素贝叶斯:")
print(classification_report(y_true=y_test,y_pred=y_pre_nb))
print("knn算法:")
print(classification_report(y_true=y_test,y_pred=y_pre_knn))
print("scv支持向量机:")
print(classification_report(y_true=y_test,y_pred=y_pre_svc))
```

图 1-10

测试结果如下：

留言主题

测试集为 15%，一共有 1382 个数据

朴素贝叶斯:

	precision	recall	f1-score	support
交通运输	0.90	0.73	0.80	99
劳动和社会保障	0.83	0.94	0.88	289
卫生计生	0.95	0.72	0.82	123
商贸旅游	0.86	0.75	0.80	185
城乡建设	0.78	0.90	0.84	300
教育文体	0.86	0.85	0.86	240
环境保护	0.86	0.83	0.84	146
accuracy			0.84	1382
macro avg	0.86	0.82	0.84	1382
weighted avg	0.85	0.84	0.84	1382

图 1-11

knn算法:

	precision	recall	f1-score	support
交通运输	0.85	0.61	0.71	99
劳动和社会保障	0.37	0.84	0.52	289
卫生计生	0.74	0.46	0.56	123
商贸旅游	0.85	0.39	0.53	185
城乡建设	0.76	0.51	0.61	300
教育文体	0.60	0.62	0.61	240
环境保护	0.98	0.27	0.43	146
accuracy			0.56	1382
macro avg	0.73	0.53	0.57	1382
weighted avg	0.69	0.56	0.56	1382

图 1-12

scv支持向量机:

	precision	recall	f1-score	support
交通运输	0.89	0.83	0.86	99
劳动和社会保障	0.87	0.88	0.87	289
卫生计生	0.85	0.85	0.85	123
商贸旅游	0.82	0.76	0.79	185
城乡建设	0.79	0.86	0.83	300
教育文体	0.83	0.83	0.83	240
环境保护	0.87	0.79	0.83	146
accuracy			0.84	1382
macro avg	0.84	0.83	0.84	1382
weighted avg	0.84	0.84	0.84	1382

图 1-13

由上述结果可知：朴素贝叶斯、KNN 和支持向量机的预测加权准确率分别为 0.85、0.69、0.84，加权召回率分别为 0.84、0.56、0.84，加权 F1 值分别为 0.84、0.56、0.84。因此选择朴素贝叶斯和支持向量机构建分类模型对分类效果更好，准确率更高。经过实践还发现，当选取训练集为 90%，选取测试集为 10%时，出来的分类模型中，支持向量机的准确率有所提高。

下面给出 7 个一级分类分类效果的条形统计图，直观地反映每一种一级分类的准确率。

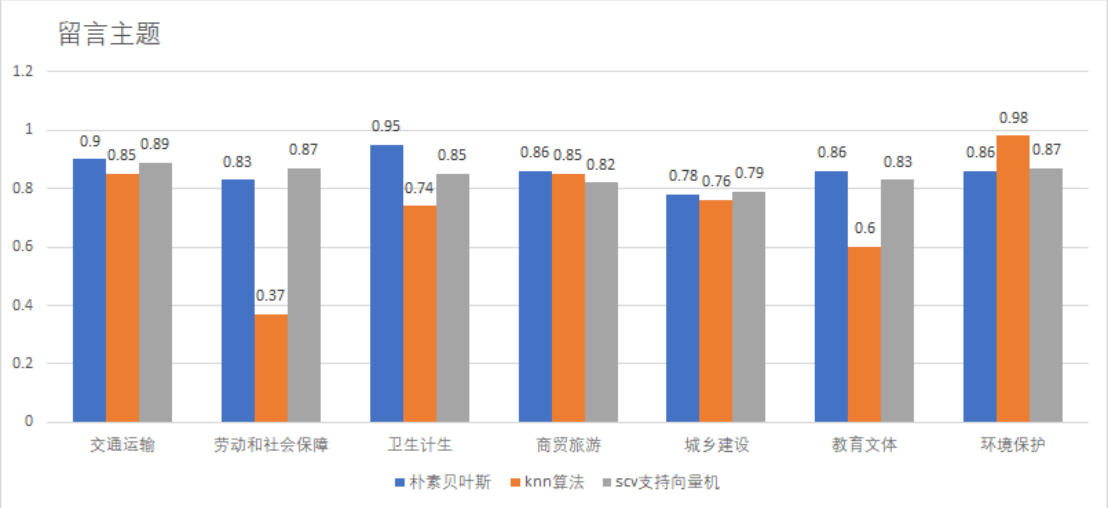


图 1-14

由上图可以看出朴素贝叶斯和支持向量机对每一种一级分类的分类准确率较高也比较平稳，朴素贝叶斯算法有两个一级分类也达到了 90%以上的准确率，分别为交通运输（准确率：0.9）和卫生计生（准确率：0.95）；KNN 算法对每一种一级分类的分类准确率波动较大，最高为环境保护（准确率：0.98），最低为劳动和社会保障（准确率：0.37）

留言内容

测试集为 15%，一共有 1382 个数据

朴素贝叶斯:

	precision	recall	f1-score	support
交通运输	0.81	0.53	0.64	99
劳动和社会保障	0.87	0.93	0.90	289
卫生计生	0.89	0.83	0.86	123
商贸旅游	0.88	0.77	0.82	185
城乡建设	0.80	0.89	0.84	300
教育文体	0.89	0.92	0.90	240
环境保护	0.89	0.93	0.91	146
accuracy			0.86	1382
macro avg	0.86	0.83	0.84	1382
weighted avg	0.86	0.86	0.86	1382

图 1-15

knn算法:

	precision	recall	f1-score	support
交通运输	0.57	0.39	0.47	99
劳动和社会保障	0.71	0.80	0.75	289
卫生计生	0.33	0.65	0.44	123
商贸旅游	0.42	0.64	0.51	185
城乡建设	0.75	0.53	0.62	300
教育文体	0.81	0.67	0.73	240
环境保护	0.89	0.38	0.53	146
accuracy			0.61	1382
macro avg	0.64	0.58	0.58	1382
weighted avg	0.67	0.61	0.61	1382

图 1-16

scv支持向量机:

	precision	recall	f1-score	support
交通运输	0.89	0.75	0.81	99
劳动和社会保障	0.89	0.93	0.91	289
卫生计生	0.85	0.91	0.88	123
商贸旅游	0.84	0.80	0.82	185
城乡建设	0.81	0.88	0.84	300
教育文体	0.94	0.89	0.91	240
环境保护	0.93	0.86	0.89	146
accuracy			0.87	1382
macro avg	0.88	0.86	0.87	1382
weighted avg	0.87	0.87	0.87	1382

图 1-17

由上述结果可知：朴素贝叶斯、KNN 和支持向量机的预测加权准确率分别

为 0.86、0.67、0.87，加权召回率分别为 0.86、0.61、0.87，加权 F1 值分别为 0.86、0.61、0.87。因此选择朴素贝叶斯和支持向量机构建分类模型对分类效果更好，准确率更高。因此，无论是测试留言主题还是测试留言详情，都是选择朴素贝叶斯和支持向量机构建分类模型对分类效果更好，准确率更高。

下面给出 7 个一级分类分类效果的条形统计图，直观地反映每一种一级分类的准确率。

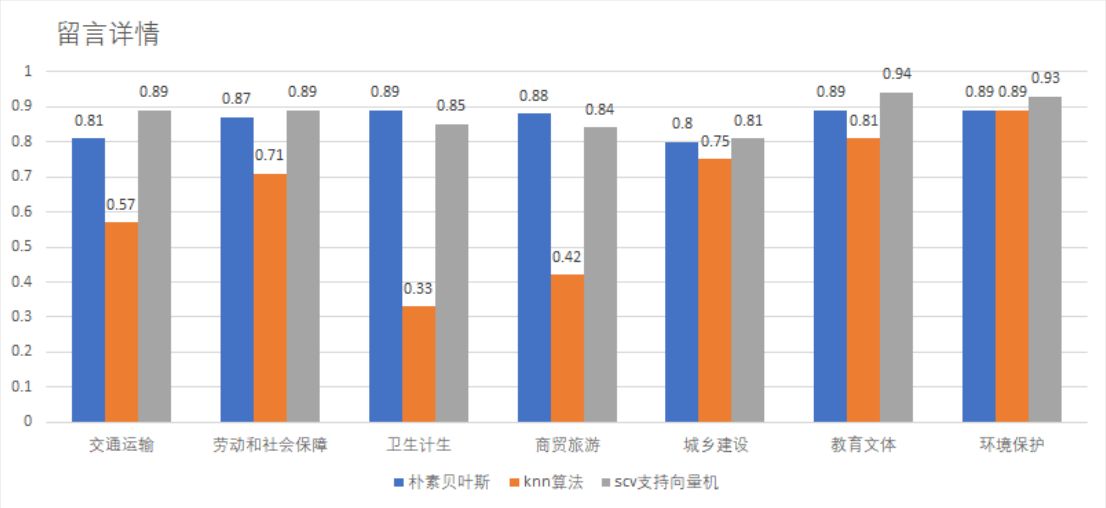


图 1-18

由上图可以看出朴素贝叶斯和支持向量机对每一种一级分类的分类准确率较高也比较平稳，支持向量机算法有两个一级分类也达到了 90%以上的准确率，分别为教育文体（准确率：0.94）和环境保护（准确率：0.93）；KNN 算法对每一种一级分类的分类准确率波动较大，最高为环境保护（准确率：0.89），最低为卫生计生（准确率：0.33）

5.2 问题二的求解过程

5.2.1 探索分析

打开附件 3，发现有留言主题、留言详情、点赞数和反对数这四个关键信息。往下读取数据，一共有 4326 条留言主题，本文将根据某一特定地点或特定人群反映的某些问题对留言主题进行归类，本文的热度评价指标定义由该热点问题的留言数、点赞数、反对数和时间这些参数指标共同决定，然后根据定义出来的热度评价指标选出热度排行前 5 的热点问题，最后给出相应热点问题对应的留言信息。接下来先进行数据的预处理。

5.2.2 数据清洗

首先把附件 3 的留言主题的数据导入 python 中，打印出来发现文本还是存在大量的标点符号和少量的特殊符号，如果不去除，这对分词的结果影响是很大的。因此使用正则表达式去除这些对文本无意义的符号，另一方面，第二题热点问题挖掘[9]与第一题群众留言分类有所不同，对于热点问题的识别，还是需要保留具体的地点和特定的人群，所以不能使用正则表达式直接去掉除中文文本以外的字符，因此本文从 CSDN 博客找到了去除特殊符号的代码[10]：

```
string = re.sub("[\s+\.\!\/_,$%^*(+\"\']+|[+——！，。？、~@#￥%……&*（）]",
"",corpus)
```

将其应用到本文的代码中，经过多次尝试，最终把所有的标点符号和特殊字符去掉，代码如下：

```
theme_content_list.append(re.sub(r"[ⅠⅡ３＃
\-\s+\.\!\/_,$%^*(+\"\')\?\[\]-·\\]+|[+——()?【】“”！，。？、~@#￥%……
&*（）《》 ]+","",All_theme_content[i]))
```

5.2.3 分词

由于要保留具体的地名和地点，而 jieba 分词对地点的分词能力很差，比如(A 市，A3 区，三号线，13 栋等）这些词是 jieba 词库没有的，因此在去除特殊符号之前还添加了自定义词典，代码如下：

```
def file_add_all_word():
    file_add_word = open(addpath,'a+')
    add_text_1=[]
    add_text_2=[]
    global add_text_all
    add_text_2=re.findall("[a-zA-z]*[0-9]+[区|市|县|号|号线|路|栋|
期]+|[a-zA-Z]+[区|市|县|号|号线|路|栋|期]",str(All_theme_content),flags=0)
    add_text_1=re.findall("[一|二|三|四|五|六|七|八|九|十|]+[区|市|县|号|
号线|路|栋|期]+",str(All_theme_content),flags=0)
    add_text_all=list(set(add_text_1+add_text_2))
    for key in add_text_all:
        file_add_word.write(key)
        file_add_word.write("\n")
    file_add_word.close()
```

此外还在自定义词典里添加了两个词，分别是'西地省'和'魅力之城'，代码如下：

```
for i in range(len(add_text_all)):
        jieba.add_word(add_text_all[i])
    jieba.add_word('西地省')
```

jieba.add_word('魅力之城')

最后进行分词操作。

5.2.4 去停用词

与第一题群众留言分类一样，分完词还是存在了大量对文本分类无意义的词，如果不去除会影响下一步骤的结果，因此还是需要去除停用词，根据数据的语境语意和题目要求，进行了词性分析，把一些无意义的词添加到了自定义停用词表（见附件），再进行去停用词的操作。

去除停用词后，效果如下：

```
0                    [A3区，一米阳光，婚纱，艺术摄影，合法，纳税]
1          [咨询，A6区，道路，命名，规划，初步，成果，公示，城乡，门牌]
2                    [A7县，春华，镇金鼎村，水泥路，自来水，到户]
3                  [A2区，步行街，古道，住户，卫生间，粪便，外排]
4          [A市，A3区，国际，社区，三期，空地，夜间，施工，噪音，扰民]
                              ...
4321           [A市，经济，学院，寒假，过年，组织，学生，工厂，工作]
4322                 [A市，经济，学院，组织，学生，外出，打工]
4323                    [A市，经济，学院，强制，学生，实习]
4324                 [A市，经济，学院，强制，学生，外出，实习]
4325               [A市，经济，学院，体育，学院，变相，强制，实习]
Name: 留言主题, Length: 4326, dtype: object
```

图 2-1 留言主题去除停用词后的部分展示

5.2.5 绘制词云

去除完停用词后，本文基于留言主题的内容绘制出词云。通过词云，可以直观地看出留言内容的分布情况和核心内容。绘制出来的词云效果如下：

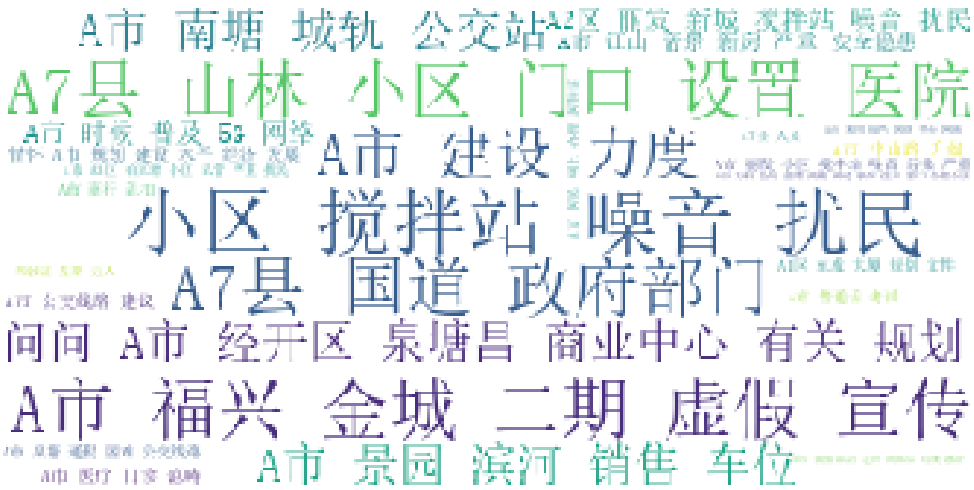


图 2-2

5.2.6 文本向量化表示

文本向量化，首先通过一个实际例子直观反映文本是如何转换为向量的[8]。

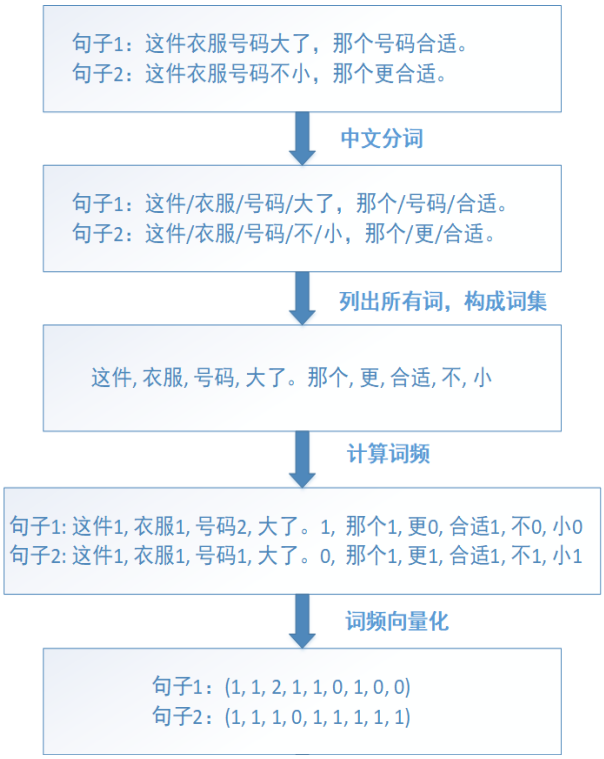


图 2-3

与第一题群众留言分类类似，把分好的特征词基于词频去向量化，把留言主题的每一句文本用词频向量表示出来，得到一个 6656x4326 的稀疏矩阵，其

中 6656 是矩阵的行数，代表着 6656 个特征词；4326 为矩阵的列数，代表着 4326 个留言主题。代码如下：

```
tmp = data_after.apply(lambda x:' '.join(x))
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer().fit(tmp)
cv_data = cv.transform(tmp)

np_data = cv_data.A#文本特征词矩阵化--稀疏矩阵
```

图 2-4

```
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]]
```

图 2-5 稀疏矩阵展示（含有很多个零）

5.2.7 相似度计算

上述已经把附件 3 的每一句留言主题用向量表示了出来，本文采用的相似度算法是余弦相似度：通过一个向量空间中两个向量夹角的余弦值作为衡量两个文本之间的差异，显然，相似度的大小就在 0~1 之间，夹角越小，余弦值越大，表示相似度也就越大，反之，相似度就越小。计算公式为：

$$\cos\theta = \frac{\sum_{i=1}^{n}(x_i \times y_i)}{\sqrt{\sum_{i=1}^{n}(x_i)^2} \times \sqrt{\sum_{i=1}^{n}(y_i)^2}} = \frac{a \bullet b}{\|a\| \times \|b\|}$$

在上述的例子中，两个句子的余弦值计算为[8]：

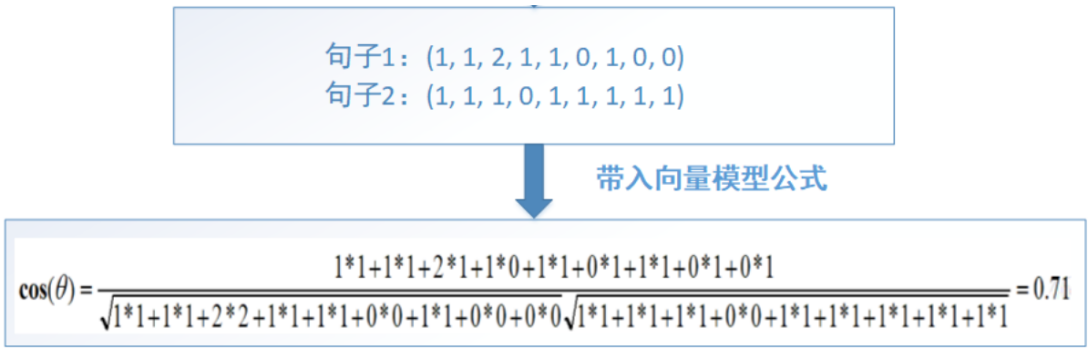


图 2-6

因此这两个句子的相似度为 71%。

本题做了一个循环，计算两两主题之间的相似度，算法实现如下：

```
'''
    求相似度矩阵
'''
cos_zhi = np.zeros((4326,4326))
mol = []#计算模

for i in range(4326):
    s = sum(np_data[i])
    mol.append(math.sqrt(s))

for i in range(4326):
    for j in range(i+1,4326):
        if mol[i] == 0 or mol[j] == 0:
            continue
        else:
            neiji = np.dot(np_data[i],np_data[j])
            mo_ji = mol[i]*mol[j]
            cos_zhi[i][j] = cos_zhi[j][i] = neiji/mo_ji
np.savetxt(r'f:\余弦值.txt',cos_zhi)

endtime = time.time()
print("耗时: ",endtime-starttime,'秒',sep='')
```

图 2-7

结果得到一个 **4326x4326** 的矩阵，结果如下：

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	1
0	0	0	0	0	0.129099	0.144338	0	0	0	0.129099	0	0.117851	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0.136083	0	0	0	0
3	0	0	0	0	0	0	0.188982	0	0	0	0	0	0	0	0
4	0.129099	0	0	0	0	0.223607	0	0.141421	0.141421	0.3	0	0.182574	0	0.358569	0.10
5	0.144338	0	0	0	0.223607	0	0.176777	0	0	0.111803	0	0.204124	0	0	0
6	0	0	0	0.188982	0	0.176777	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0.141421	0	0	0	0.2	0.141421	0	0	0	0.169031	0.44
8	0	0	0	0	0.141421	0	0	0.2	0	0	0	0	0	0.169031	0.14
9	0.129099	0	0	0	0.3	0.111803	0	0.141421	0	0	0	0.182574	0	0.119523	0.21
10	0	0	0.136083	0	0	0	0	0	0	0	0	0	0	0	0
11	0.117851	0	0	0	0.182574	0.204124	0	0	0	0.182574	0	0	0	0.109109	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0.358569	0	0	0.169031	0.169031	0.119523	0	0.109109	0	0	0.12
14	0	0	0	0	0.105409	0	0	0.447214	0.149071	0.210819	0	0	0	0.125988	0
15	0	0	0	0	0	0.158114	0	0	0	0	0	0.129099	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0.408248	0	0	0	0	0	0	0	0.166667	0	0	0	0
18	0	0.141421	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0.119523	0	0	0.169031	0.169031	0	0	0	0	0.142857	0.12
20	0	0.111803	0	0	0.111803	0	0	0	0	0	0	0	0	0	0
21	0	0	0.182574	0	0	0	0	0	0	0	0.149071	0	0	0	0
22	0	0	0	0.154303	0	0.144338	0.204124	0	0	0	0	0.117851	0	0	0

图 2-8

5.2.8 留言问题归类

最后，筛选出相似度大于 **0.75** 的列索引，把每一行的元素作为一个集合，去求交集，若交集非空，即为一类问题，最后把这一类留言问题求并集，得出这类留言问题的留言总数，作循环，把所有留言问题进行归类。代码如下：

```
# 获得每行cos_zhi>=0.75的列索引,将其放在similar[]中
for i in range(4326):
    similar.append(np.where(cos_zhi[i]>=0.75))
# 把similar的元素转为集合对象
for i in range(len(similar)):
    set_similar.append(set(similar[i][0]))
# 求交，求相似的
for i in range(4326):
    tmp_set = set_similar[i]
    if i in is_in or len(tmp_set) == 0:
        continue
    else:
        for j in range(0,4326):
            s = tmp_set & set_similar[j]
            if len(s)>0:
                tmp_set = tmp_set|set_similar[j]
                is_in.append(j)
    if len(tmp_set)>0:
        samekind.append(tmp_set)

# 再次求并
samekind_last = []
is_in2 = []
for i in range(len(samekind)):
    tmp = samekind[i]
    if i in is_in2:
        continue
    else:
        for j in range(0,len(samekind)):
            if len(tmp&samekind[j])>0:
                tmp = tmp|samekind[j]
                is_in2.append(j)
        if tmp == samekind[i]:
            samekind_last.append(tmp)
```

图 2-9

5.2.9 定义热度评价指标

（1）符号含义

Score	热度评价得分
A	某一类问题的留言数
x	点赞数
y	反对数
t0	定义的初始时间
t1	该类问题留言的平均时间

本文定义的初始时间为 2017 年 1 月 1 日 0 时 0 秒，z=x-y(点赞数－反对数)

（2）定义热度评价指标

定义的热度评价得分的计算公式为：Score=该类问题的留言数+ln(点赞数－反对数)+$\frac{\text{该类问题留言的平均时间}-\text{定义的初始时间}}{\text{一个季度的时间}}$

最终的计算公式为：

$$z=\begin{cases}1, x \le y \\ x-y, x>y\end{cases} \quad ①$$

$$Score = A + \ln z + \frac{t_1 - t_0}{90} \qquad ②$$

（3）热度评价指标的算法解释：

热点问题的评价指标由该热点问题的留言数、点赞数、反对数和时间这些参数指标共同决定；

z 表示点赞数－反对数，本文采用取自然对数来削弱这部分的权重。若点赞数小于或等于反对数，则取 z 为 1；

时间是一个非常重要的参数，本文采用的算法对于问题的得分不会因为时间的更迭而减少，但是新的留言问题会比旧的留言问题得分更高。一个季度的时间为 3 个月，即 90 天。

5.2.10 评价结果

（1）构建得分模型

代码如下：

```
# 建立模型
Score = []
for i in range(len(samekind_last)):
    length = len(samekind_last[i])
    z = 0 #点赞数-反对数
    t0 = datetime.datetime.strptime('2017-1-1 00:00:00',
                                    '%Y-%m-%d %H:%M:%S')
    time_list = []
    for key in samekind_last[i]:
        t = datetime.datetime.strptime(data_time[key],'%Y-%m-%d %H:
        time_list.append(t)

    day = 0
    for j in range(len(time_list)):
        day+=(time_list[j]-t0).days
    day=day/length
    for key in samekind_last[i]:
        z = z + data_agree[key] - data_disagree[key]
    if z<=0:
        z = 1
    Sc = length + math.log(z,math.e) + day/90
    Score.append(Sc)
```

图 2-10

（2）热点问题排行

根据得分，给出排名前五的热点问题：

```
Score_sort = Score[:]
Score_sort.sort()#热度排序
top_key = [] #获得热度前五的索引
top_zhishu = [] #获得热度前五的热度指数
top_huati = [] #获得热度前五的话题
for i in range(5):
    top_key.append(Score.index(Score_sort[-i-1]))
for key in top_key:
    top_zhishu.append(Score[key])
for key in top_key:
    top_huati.append(samekind_last[key])
```

图 2-11

结果如下：

```
[39.550436898026504,
 39.48074855288878,
 35.51925282639305,
 22.238883423610886,
 20.482828282828283]
```

图 2-12

(排名前五的热点问题得分)

排名第一的索引：

{90,288,320,452,704,732,788,1297,1446,1484,1546,1779,1953,2015,2347,2660,2831,2938,3052,3390,3718,3733,3841,4040,4139,4149,4299}

排名第二的索引：

{52, 645,664,725,774,1355,1516,1580,1690,1742,1862,2375,2499,2751,3013, 3096,3182,3278, 3281,3377,3469,3941,3955,3964,4020,4271}

排名第三的索引：{70,79,107,634,834,1052,1091,1238,1799,2095,2138,2156,2321, 2470,2709,2810,3077,3323,3923,4019}

排名第四的索引：{909, 2107, 2803, 2969, 3301, 3455, 3470, 4066, 4201, 4284}

排名第五的索引：{49, 66, 388, 1102, 1112, 1686, 2195, 2414, 2759, 3310, 3638}

（数字代表附件 3 的文本的位置）

下图给出问题 1~5 的热度得分统计图：

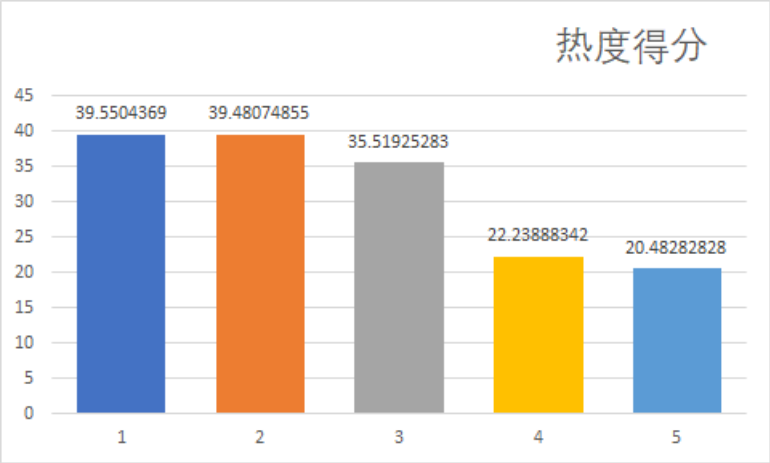


图 2-13

问题 1：A 市武广新城伊景园滨河苑小区捆绑车位销售

问题 2：A 市人才购房补贴、住房公积金贷款以及购房资格相关问题

问题 3：A 市 A2 区丽发新城小区附近搅拌站噪音扰民和污染环境

问题 4：加快 A 市国家中心城市的建设

问题 5：A 市 A3 区西湖街道茶场村五组拆迁规划

（详情见附件的热点问题表.xls 和热点问题留言明细表.xls）

5.3 问题三的求解过程

5.3.1 探索分析

打开附件 4，发现有留言主题、留言详情和答复意见这三个关键信息。往下

读取数据，一共有 2816 条答复意见。本文将根据留言详情和答复意见，从答复的相关性、完整性、可解释性这三个角度对答复意见的质量定义质量评价指标，最后给出评价结果。

5.3.2 答复意见质量评价指标[11]

（1）首先需要对相关性、完整性、可解释性进行量化。

① 相关性的解释及量化指标

相关性：答复意见的内容是否与内容相关，是否存在答非所问或者答复的是别的问题。

量化指标：本文采用的方法是计算留言详情与答复意见的相似度，求留言详情与答复意见共有的特征词，根据共有的特征词的数量的多少来决定相关性的大小。

② 完整性的解释及量化指标

完整性：是否满足某种答复的规范，存在一套答复标准，有开头，有内容也要有结尾。

量化指标：本文采用的方法是按照答复内容的结构选择了一些规范词：（网友，您好，已收悉，答复如下，建议，感谢，年，月，日）等等，详情见附录（完整性_规范词），通过答复意见的内容里规范词出现的种数多少以及答复意见内容的多少来衡量完整性的大小。

③ 可解释性的解释及量化指标

可解释性：答复意见中内容的相关解释，有没有相关的法律法规或者方针政策的支撑。

量化指标：本文采用的方法是挖掘答复内容中是否有引用法律法规或者某种政策来决定可解释性的大小。

（2）定义答复意见质量评价指标

① 相关性得分

若留言详情与答复意见共有的特征词大于或等于 10，则为 10 分；

若留言详情与答复意见共有的特征词在 3~9 之间，则为对应的分数 3~9 分；

若留言详情与答复意见共有的特征词在 0~2 之间，则为 0 分。

② 完整性得分

完整性得分由两部分组成：

第一是 $\frac{\text{答复意见内容的字数}}{\text{留言详情内容的字数}}$ 的比值，若大于等于 2，则得分固定为 2x10=20 分，若小于 2，则得分为相应的比值 x10。

第二是规范词的种数，（完整性_规范词）里一共列了 20 个，即答复内容出现多少种即为多少分，满分为 20 分。

③ 可解释性得分

若答复内容种引用了法律法规或者某种政策，即得 30 分，反之，即得 0 分。

答复意见质量得分公式：

$$Score = \text{相关性得分} \times 3 + \text{完整性得分} + \text{可解释性得分}$$

5.3.3 数据的预处理[12]

由于计算相关性得分需要提取特征词，所以需要对留言详情和答复意见的数据进行预处理，去除标点符号、特殊符号和一些无意义的词。与第二题热点问题挖掘的预处理方法类似，用正则表达式去符号、再分词和去停用词，此处不再赘述。

5.3.4 相关性得分计算

去完停用词之后，开始计算留言详情和答复意见的相同特征词的个数，即把每一个留言详情与对应的答复意见求交集，最后算出每一个答复意见的相关性得分，代码实现如下：

```
# ∑交集元素----相关性
count_connect = []
for i in range(len(data_connect)):
    if len(data_connect[i])<=2:
        s = 2
    elif 3<=len(data_connect[i])<=9:
        s = len(data_connect[i])
    else:
        s = 10
    count_connect.append(s)
```

图 3-1

结果如下：

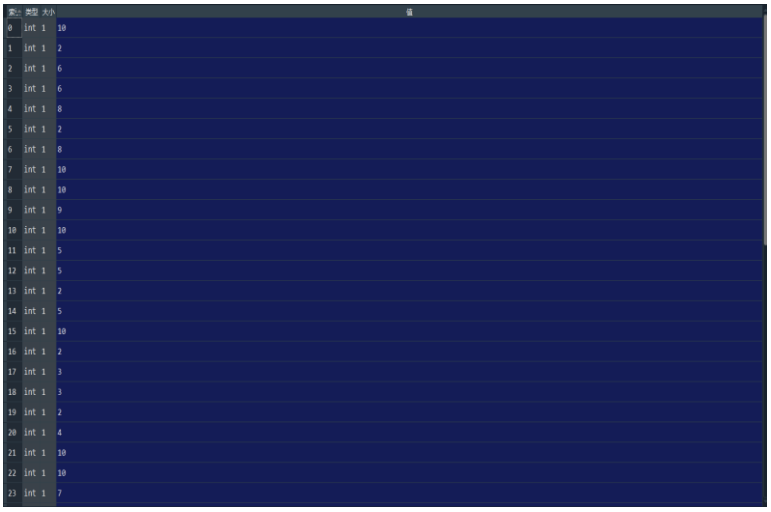


图 3-2

可见，相关性得分为满分的答复意见是比较多的。

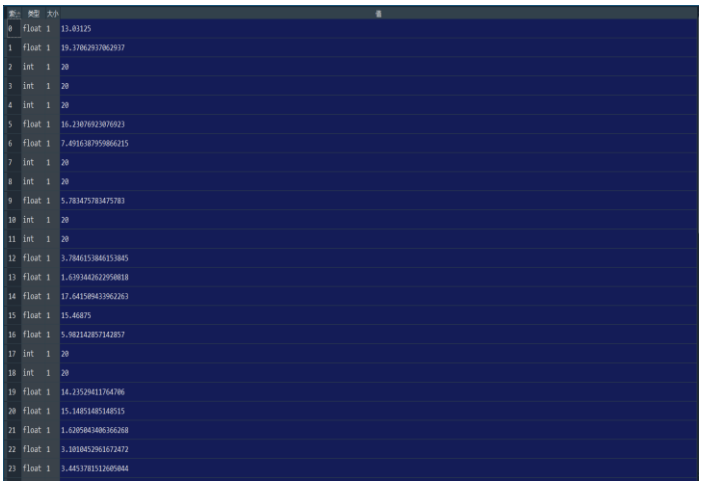
5.3.5 完整性得分计算

计算第一部分的得分，代码如下：

```python
# 回复数/留言数
wanzheng = []
for i in range(len(data_answer)):
    tmp = (len(data_answer[i])/len(data_content[i]))*10
    if tmp>=20:
        tmp = 20
    wanzheng.append(tmp)
```

图 3-3

得分结果如下：



	索引	类型	大小
0	float	1	13.03125
1	float	1	19.170629370629377
2	int	1	20
3	int	1	20
4	int	1	20
5	float	1	16.230769230769234
6	float	1	7.491638795986622
7	int	1	20
8	int	1	20
9	float	1	5.783475783475783
10	int	1	20
11	int	1	20
12	float	1	3.7846153846153845
13	float	1	1.6393442622950818
14	float	1	17.641509433962263
15	float	1	15.46875
16	float	1	5.982142857142857
17	int	1	20
18	int	1	20
19	float	1	14.235294117647058
20	float	1	15.148514851485151
21	float	1	1.6205043406366268
22	float	1	3.1010452961672472
23	float	1	3.4453701512605044

图 3-4

计算第二部分的得分，代码如下：

```python
# 规范词
list_norm_word=[]
#获取规范词
with open(norm_path,"a+",encoding="UTF-8") as norm_file:
    norm_file.seek(0)
    norm_word = norm_file.read()
    list_norm_word = norm_word.split()
list_count = [] #记录完整性得分表
for i in range(len(data_answer)):
    count = 0
    for j in range(len(list_norm_word)):
        if list_norm_word[j] in data_answer[i]:
            count+=1
    list_count.append(count)
```

图 3-5

得分结果如下：

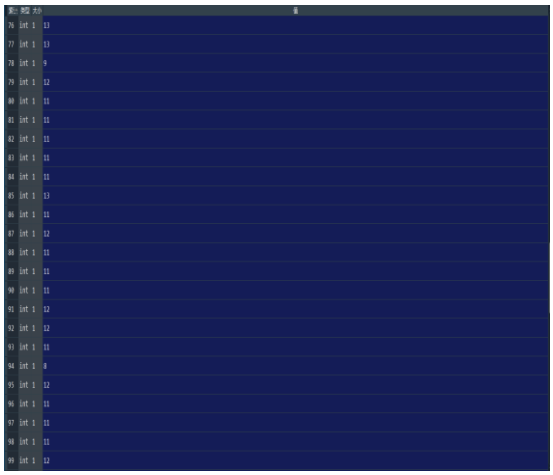


图 3-6

5.3.6 可解释性得分计算

若答复内容种引用了法律法规或者某条规定，如《大气污染防治法》、《A市公益性岗位管理办法》，即会出现书名号，或者出现了政策的字眼，则说明具有可解释性，得 30 分，反之，则认为得 0 分。

代码实现如下：

```
# 可解释性
list_count_jieshi=[]  #可解释性得分表
for i in range(len(data_answer)):
    if ('政策' in data_answer[i]) or ('《' in data_answer[i]):
        list_count_jieshi.append(30)
    else:
        list_count_jieshi.append(0)
```

图 3-7

结果得分如下：

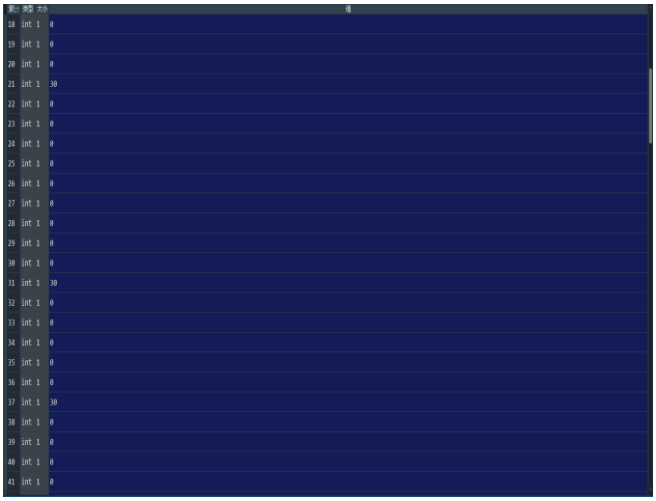


图 3-8

5.3.7 评价结果

（1）最后，用答复意见质量得分公式：

$$Score = 相关性得分 \times 3 + 完整性得分 + 可解释性得分$$

进行求和计算。
代码如下：

```
# 计算回复质量得分
Score_reply = []
for i in range(len(data_answer)):
    s = list_count_jieshi[i] + list_count[i] + wanzheng[i] + count_connect[i]
    Score_reply.append(s)
```

图 3-9

部分得分结果如下：

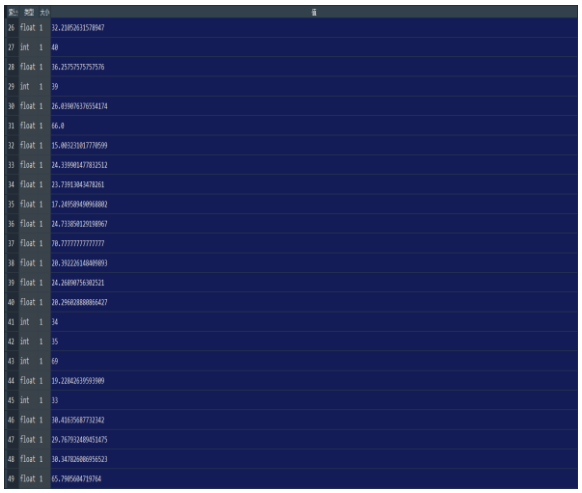


图 3-10

本文把答复意见质量得分放进附件 4，根据答复质量等级可以直观地看出每一个问题答复质量的好坏。详情见作品附件（新附件 4）
（2）答复质量等级
优秀：高于 70 分，有 27 个
良好：高于 50 分，有 343 个
合格：高于 30 分，有 1366 个
因此，答复意见质量的合格率=1366/2816=48.5%

六、总结

本文通过 python 工具，利用自然语言技术和文本挖掘技术建立文本分类模型、热点问题热度评价模型和答复意见质量评价模型。首先对附件 2、附件 3、附件 4 的数据进行探索分析和预处理，为建立模型做准备。

针对问题一，预处理之后，把分好后的词绘制词云，通过词云，可以直观地看出每一个一级分类特征词的分布情况和核心内容；再用向量化表示出来，构建朴素贝叶斯、KNN 算法和支持向量机 3 种分类模型去进行训练和测试，用

F-score 对 3 种分类模型进行评价，最后得出结论：朴素贝叶斯和支持向量机都具有不错的分类效果，有 85%以上的准确率，也出现了准确率高于 90%的情况。由于我们的知识有限，后续我们将继续研究，优化分类模型和建立更多的分类器进行训练和测试，争取有更好的表现。

针对问题二，把每一个留言主题用向量表示出来，然后进行两两之间的相似度计算，本文用到的是余弦相似度算法，两个向量的夹角的余弦值越大，即相似度越高。最后，筛选出相似度大于 0.75 的问题进行归类，并根据定义的热点评价指标算出热度排名前 5 的热点问题。总体来看，余弦相似度算法准确率和效率都有不错表现，后续我们也将尝试实现其他的相似度算法，争取有更好的准确率。

针对问题三，本文只考虑了三个角度，即从答复的相关性、完整性和可解释性这三个角度定义答复意见质量评价指标，量化相关性、完整性和可解释性并计算得分，最后给出评价结果。本文使用评价模型算法较为简单，考虑的角度不够全面，量化指标也有瑕疵，具有一定局限性。后续我们将优化我们的评价模型，争取有更好的表现。

致谢：十分感谢泰迪杯官方给出关于“智慧政务”中的文本挖掘应用的赛题。随着 5G 时代的到来，大数据和人工智能也得到了迅猛的发展，建立基于自然语言处理技术的智慧政务系统已经是社会治理创新发展的新趋势，文本挖掘和文本分类也成为大数据和人工智能的重要课题。在这次比赛的过程中，我们队伍也学习到了很多知识，身心也得到了成长，在未来的学习或工作道路上，我们将脚踏实地，继续努力，不断学习与进取，争取有更大的进步！

七、参考文献

[1]祝永志,荆静.基于 Python 语言的中文分词技术的研究[J].通信技术,2019,52(07):1612-1619.
[2]https://blog.csdn.net/meiqi0538/article/details/80213431
[3]https://blog.csdn.net/pengjunlee/article/details/90241312
[4]严明,郑昌兴.Python 环境下的文本分词与词云制作[J].现代计算机(专业版),2018(34):86-89.
[5]张帆. 贝叶斯算法在校园留言板垃圾过滤中的应用研究[D].郑州大学,2016.
[6]祁小军,兰海翔,卢涵宇,丁蕾锭,薛安琪.贝叶斯、KNN 和 SVM 算法在新闻文本分类中的对比研究[J].电脑知识与技术,2019,15(25):220-222.
[7]王道明,鲁昌华,蒋薇薇,肖明霞,李必然.基于粒子群算法的决策树 SVM 多分类方法研究[J].电子测量与仪器学报,2015,29(04):611-615.
[8]https://www.cnblogs.com/liangjf/p/8283519.html
[9]吴柳,程恺,胡琪.基于文本挖掘的论坛热点问题时变分析[J].软件,2017,38(04):47-51.
[10]https://blog.csdn.net/nlite827109223/article/details/62237733
[11]赵妍妍,秦兵,刘挺.文本情感分析[J].软件学报,2010,21(08):1834-1848.
[12]丁森华,邵佳慧,李春艳,杨枝蕊.文本情感分析方法对比研究[J].广播电视信息,2020(04):92-96.