

第八届“泰迪杯”数据挖掘挑战赛——

C 题：“智慧政务”中的文本挖掘应用

摘要：附件给出了收集自互联网公开来源的群众问政留言记录，及相关部门对部分群众留言的答复意见。本文主要利用自然语言处理和文本挖掘的方法解决群众留言分类、热点问题挖掘以及答复意见的评价的问题。针对这些问题，本文建立了分类模型、多项式贝叶斯和 LDA 主题模型。

关键词：分类模型、多项式贝叶斯、LDA 主题模型

目录

- 1、问题重述.....3
 - 1.1 问题背景.....3
 - 1.2 要解决的问题.....3
- 2、问题分析.....3
 - 2.1 问题一分析.....3
 - 2.2 问题二分析.....3
 - 2.3 问题三分析.....3
- 3、问题一的求解.....4
 - 3.1 数据选取.....4
 - 3.2 导入数据.....4
 - 3.3 自定义函数.....4
 - 3.4 绘制词频图.....5
 - 3.5 模型的建立与验证.....7
- 4、问题二的求解.....7
 - 4.1 排名前 5 热点问题.....7
 - 4.2 热点问题留言明细.....8
- 5、问题三的求解.....8
 - 5.1 导入数据.....8
 - 5.2 绘制词频图.....8

1、问题重述

1.1 问题背景

近年来，随着微信、微博、市长信箱、阳光热线等网络问政平台逐步成为政府了解民意、汇聚民智、凝聚民气的重要渠道，各类社情民意相关的文本数据量不断攀升，给以往主要依靠人工来进行留言划分和热点整理的相关部门的工作带来了极大挑战。同时，随着大数据、云计算、人工智能等技术的发展，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，对提升政府的管理水平和施政效率具有极大的推动作用。附件给出了收集自互联网公开来源的群众问政留言记录，及相关部门对部分群众留言的答复意见。请利用自然语言处理和文本挖掘的方法解决下面的问题。

1.2 要解决的问题

根据群众的留言数据，建立模型，将群众留言根据一级标签进行划分，划分成七大类。同时对群众留言数据进行分析归类，定义合理的热度评价指标，并给出评价结果，提炼出排名前五的热点问题，给出相应热点问题对应的留言信息。针对附件 4 对答复意见的质量给出一套评价方案。

2、问题分析

2.1 问题一分析

在处理网络问政平台的群众留言时，传统的留言分类流程基本为人工凭借经验对留言进行分类，然后将留言分派给相应的职能部门处理，但是人工处理的时候依旧存在大量的问题，例如工作量较大，效率较低，而且差错率较高等问题，因此需要通过建立一级标签分类模型来进行辅助分类，并对这个分类模型进行验证。

2.2 问题二分析

某一时段内群众几种反映的某一问题可以称之为热点问题，如“xxx 小区多位业主多次反映入夏以来小区楼下烧烤店深夜经营导致噪音和油烟扰民”。我们需要将这类反映较多、热度较高的问题提取出来，并将其反馈给有关部门及时处理。需要我们根据附件 3，将留言进行归类，定义合理的热度评价指标，并给出评价结果，按照表一的格式给出排名前 5 的热点问题，并将其保存为“热点问题表.xls”。按照表 2 的格式给出相应热点问题对应的留言信息，并保存为“热点问题留言明细表.xls”。

2.3 问题三分析

需要我们根据附件 4 相关部门对群众留言的答复意见，对答复的相关性、完整性、课解释性等角度进行评价，并对答复意见给出一套合理的评价方案，同时尝试实现它。

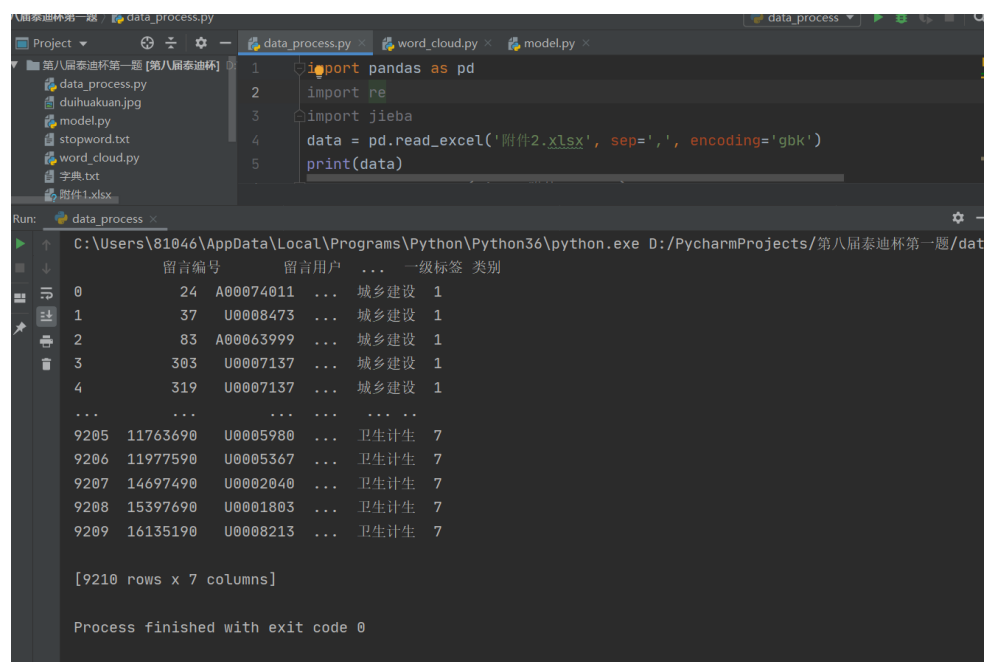
3、问题一的求解

3.1 数据选取

根据题意，传统的留言划分是人工进行划分，且大部分都是依靠经验处理，存在工作量大、效率低，且容错率高等问题。题目已给出附件 1 和附件 2。附件 1 中将留言一共划分了三个等级，分别为一级标签、二级标签、三级标签。而我们需要根据附件 1 的划分体系将附件 2 提供的数据，建立关于留言内容的以及标签分类模型。我们在附件 2 的基础上在最后面加上一列“类别”，利用 Excel 表格当中的 if...else...函数将“类别”列填充，若一级标签为“城市建设”则类别为 1，若一级标签为“环境保护”则类别为 2，以此类推，将类别列填充完整，一共七个类别。

3.2 导入数据

首先导入 jieba 库、re 库和 pandas 库，通过 Python 代码将附件 1 中数据导入 pycharm 并运行，得到的结果如图一所示。



```
1 import pandas as pd
2 import re
3 import jieba
4 data = pd.read_excel('附件2.xlsx', sep=',', encoding='gbk')
5 print(data)
```

	留言编号	留言用户	...	一级标签	类别
0	24	A00074011	...	城乡建设	1
1	37	U0008473	...	城乡建设	1
2	83	A00063999	...	城乡建设	1
3	303	U0007137	...	城乡建设	1
4	319	U0007137	...	城乡建设	1
...
9205	11763690	U0005980	...	卫生计生	7
9206	11977590	U0005367	...	卫生计生	7
9207	14697490	U0002040	...	卫生计生	7
9208	15397690	U0001803	...	卫生计生	7
9209	16135190	U0008213	...	卫生计生	7

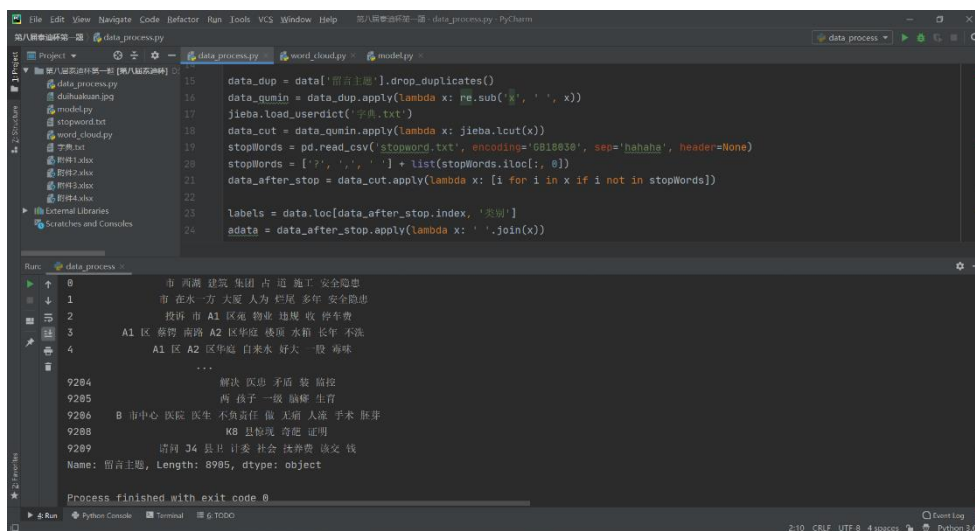
[9210 rows x 7 columns]

Process finished with exit code 0

图一

3.3 自定义函数

我们将导入的数据中“留言主题”列经过一系列数据预处理操作（分词、去重、删停用词）等基础操作，将数据进行处理，得到清洗后较为干净的数据，如图二所示：

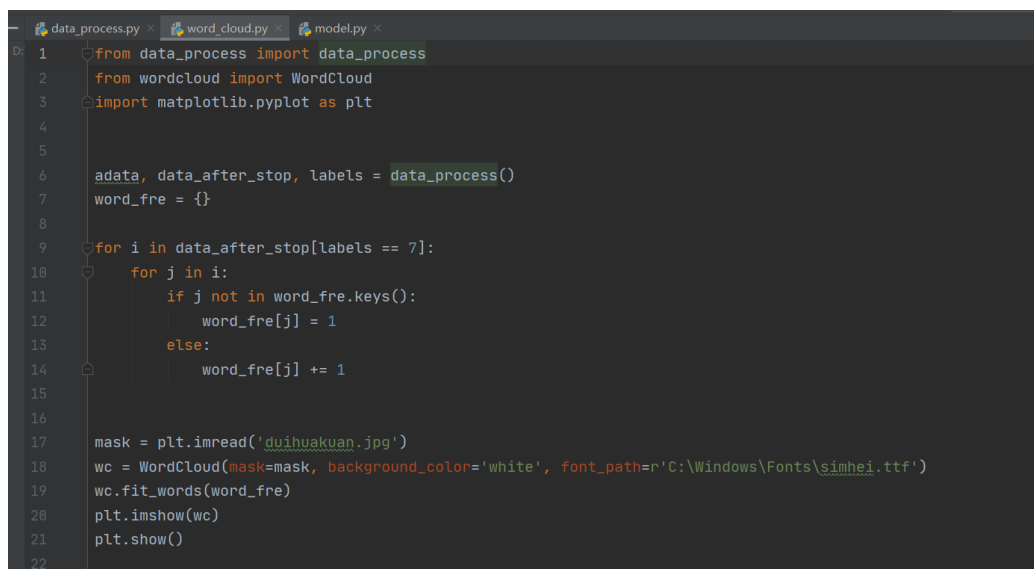


图二

然后我们开始进行自定义函数，将处理好的数据等代码包装起来得到一个名为 data_process 的自定义函数。

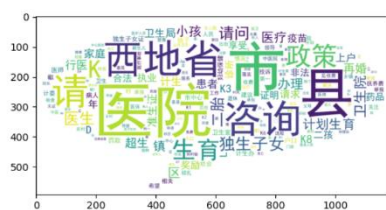
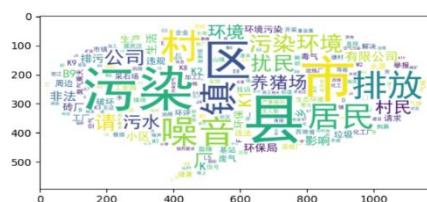
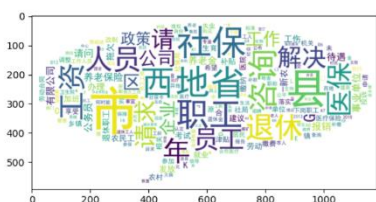
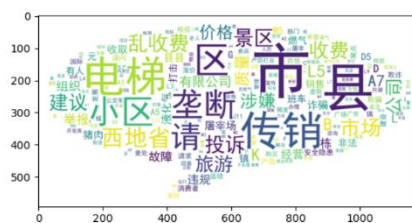
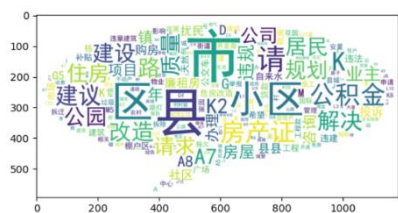
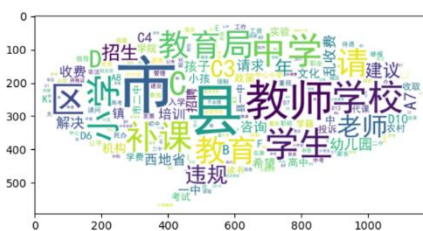
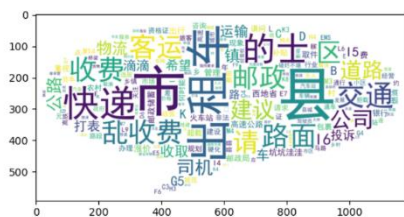
3.4 绘制词频图

有了自定义函数，建立一个新 file 绘制词频图，首先导入 data_process、wordcloud 和 matplotlib 库，利用一个 for 循环将附件 2 中的数据进行词频统计，如图三。



图三

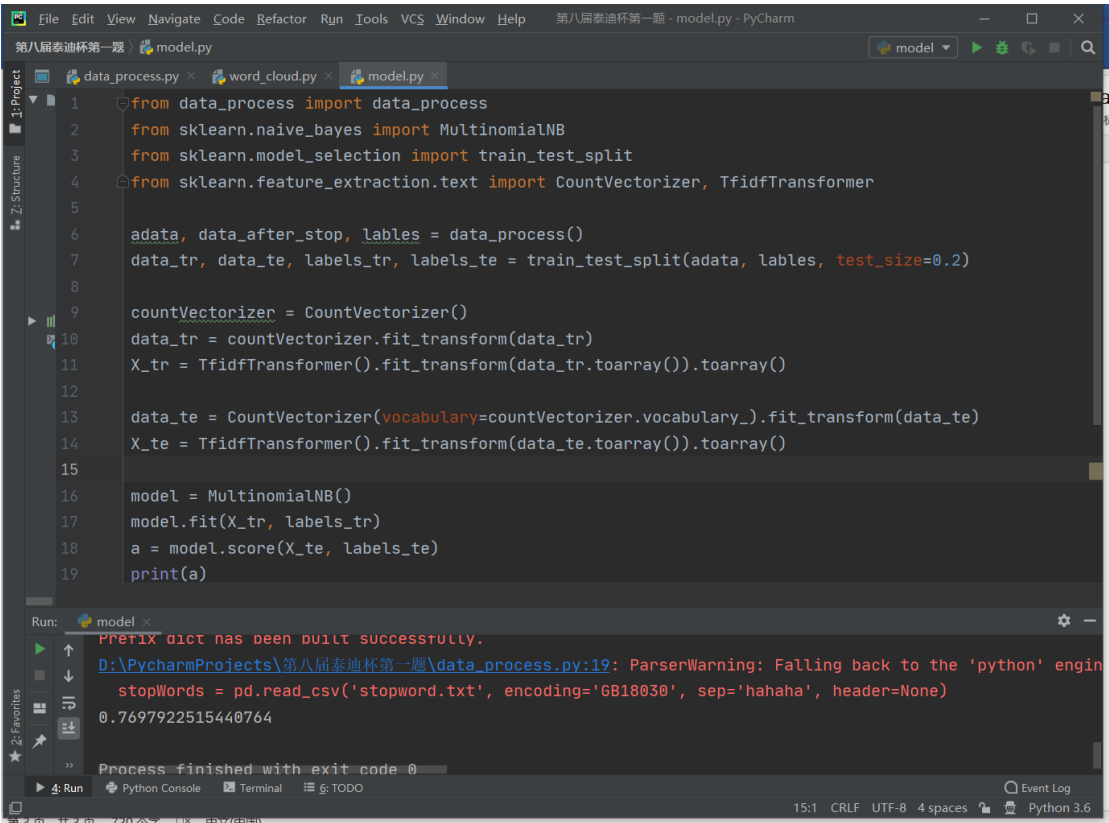
因为有七个类别，所以我们可以绘制出七张词频图。如下图：



我们可以从图中清晰的看出每个类别提到次数最多的词，而这些词就是这个类别的特征，我们只需要利用这些特征建立模型就能较为准确地将群众留言进行分类了。

3.5 模型的建立与验证

建立新模块，导入 data_process、sklearn.naive_bayes 中的多项式贝叶斯，导入 sklearn.model_selection 中的 train_test_split，导入 sklearn.feature_extraction.text 中的 CountVectorizer 和 TfidfTransformer，通过一系列代码建立出关于留言内容的一级标签分类模型。建立好模型后我们使用测试数据进行验证，验证结果为 0.769，接近 1，说明模型较为符合。



```
1 from data_process import data_process
2 from sklearn.naive_bayes import MultinomialNB
3 from sklearn.model_selection import train_test_split
4 from sklearn.feature_extraction.text import CountVectorizer, TfidfTransformer
5
6 adata, data_after_stop, labels = data_process()
7 data_tr, data_te, labels_tr, labels_te = train_test_split(adata, labels, test_size=0.2)
8
9 countVectorizer = CountVectorizer()
10 data_tr = countVectorizer.fit_transform(data_tr)
11 X_tr = TfidfTransformer().fit_transform(data_tr.toarray()).toarray()
12
13 data_te = CountVectorizer(vocabulary=countVectorizer.vocabulary_).fit_transform(data_te)
14 X_te = TfidfTransformer().fit_transform(data_te.toarray()).toarray()
15
16 model = MultinomialNB()
17 model.fit(X_tr, labels_tr)
18 a = model.score(X_te, labels_te)
19 print(a)
```

Run: model x
PREFIX dict has been built successfully.
D:\PycharmProjects\第八届泰迪杯第一题\data_process.py:19: ParserWarning: Falling back to the 'python' engine
stopWords = pd.read_csv('stopword.txt', encoding='GB18030', sep='hahaha', header=None)
0.7697922515440764
Process finished with exit code 0

图十一

通过这个模型我们可以将群众的留言信息用电脑划分成七个类别。

4、问题二的求解

我们将某一时段内群众集中反映的某一问题称为热点问题。及时发现热点问题，有助于相关部门进行有针对性地处理，提升服务效率。

4.1 排名前 5 热点问题

我们根据附件 3 将某一时段内反映特定地点或特定人群问题的留言进行归类，定义合理的热度评价指标，并给出评价结果，如附件“热点问题表.xls”，在此我们给出排名前 5 的热点问题。

4.2 热点问题留言明细

由附件“热点问题表.xls”可知热点问题的 ID，因此，我们给出相应热点问题对应的留言信息，并保存为“热点问题留言明细表.xls”。

5、问题三的求解

5.1 导入数据

通过附件 4，给出的数据，将他通过 python 语句导入 pycharm 中，此方法类似于问题一，导入 pandas、re、wordcloud、matplotlib、jieba 库，对数据进行预处理（去重、分词）等操作，并将处理好的数据导出命名为 pos_com.csv。

```
1 import pandas as pd
2 import re
3 from wordcloud import WordCloud
4 import matplotlib.pyplot as plt
5 import jieba
6 data = pd.read_excel('附件4.xlsx', sep=',', encoding='gbk')
7
8 data1 = data['留言详情']
9 data1 = data1.dropna().apply(lambda x: re.sub('\n', '', x)) # 删除换行符
10 data1 = data1.apply(lambda x: re.sub('&[a-zA-Z]+', '', x))
11
12 data1.drop_duplicates() # 去重
13 data_cut = data1.apply(jieba.lcut)
14 stop_word = pd.read_csv('stopword.txt', sep='haha')
15 stopWords = ['市'] + list(stop_word.iloc[:, 0])
16
17 data_after_stop = data_cut.apply(lambda x: [i for i in x if i not in list(stopWords)])
18 messge_ = data_after_stop.apply(lambda x: ' '.join(x))
19 messge_.to_csv('pos_com.csv')
```

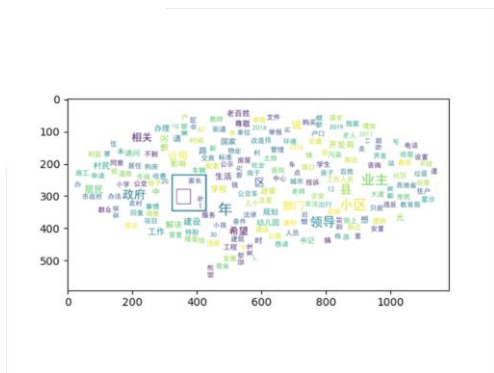
图十二

5.2 绘制词频图

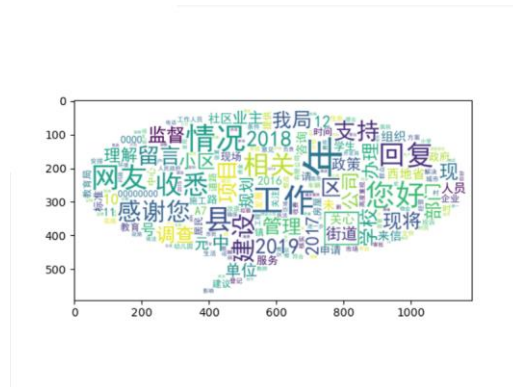
创建字典，利用一个 for 循环将数据遍历，如图十三，制作词频统计图，左图为留言详情词频统计，右图为答复意见词频统计。可以从图中仔细观察得答复意见和留言内容的相关性还是很接近的。

```
23 dic = {}
24 for i in data_after_stop:
25     for j in i:
26         if j not in dic.keys():
27             dic[j] = 1
28         else:
29             dic[j] += 1
30
31 mask = plt.imread('duihuakuan.jpg')
32 wc = WordCloud(font_path=r'C:\Windows\Fonts\simhei.ttf', mask=mask, background_color='white')
33 wc.fit_words(dic)
34 plt.imshow(wc)
35 plt.show()
```

图十三



图十四



图十五

5. 3LDA 主题模型的建立与测试

首先导入 pandas、gensim.corpora 中的 Dictionary、gensim.models 中的 LdaModel，导入完以上几种库之后，将之前保存的数据加载进来，利用 dictionary 生成一个字典然后将它代入模型当中测试得出结果。

```
1 import pandas as pd
2 from gensim.corpora import Dictionary
3 from gensim.models import LdaModel
4
5 pos_com = pd.read_csv('pos_com.csv', index_col=0)
6 # print(pos_com)
7 pos_com.columns = ['comment']
8 mid = list(pos_com['comment'].str.split(' '))
9 print(mid)
10 dictionary = Dictionary(mid)
11 bow = [dictionary.doc2bow(comment) for comment in mid]
12
13 pos_model = LdaModel(corpus=bow, id2word=dictionary, num_topics=3) # 构建LDA主题模型
14 a = pos_model.print_topic(1) # 打印主题
15 print(a)
```

图十六

这里再一次验证了答复意见的相关性和完整性，答复意见对相关内容来说，可以说是很接近了，而结果也显示了关于答复意见的词语使用频率，对于每一类的意见都有进行相关的答复。

LDA 主题模型对答复意见列进行了很好的评价，对每个关键词都有着统计，可以看出这个词语在这个语句当中的使用率，可以得知答复意见还是相当完整的。