“智慧政务”中的文本挖掘应用

摘要

随着各种网络平台的层出不穷，人们有越来越多的方式去反映自己的想法和意见，这对于他们来说是快捷且方便的，但是与此同时，对于依靠人工来收集数据和分别类型的政府来说，这带来了巨大的挑战。为了让政府更好的去制定有效的公共政策，维护社会秩序，并且可以及时的为社会和民众提供宽泛而优良的公共服务和公共物品，形成有效调整社会关系和行为的制度及机制,从而促进国家持续发展的能力。实时的去明晰民众的思想和意愿就显得格外重要，其中非常主要的方式就是对民众的文本数据进行内在信息的数据挖掘和分析。而对于收集到的这些信息，会更有利于提升政府施行政务的能力和管理水平。本文将在数据挖掘技术的基本上对群众问政留言记录和相关部门对部分群众留言的答复意见的数据进行内在信息的挖掘和分析。

本队伍做 C 题的过程，第一步先运用 python 及 matlab 对数据进行提取，然后分析数据组成与类型等，再运用 excel 进行表格的整理，接着分析各数据量（人群、地点、热度、关注度等）之间的相关性并且从中找出其地点的规律，初步建立数学模型。

第二步，基于初步模型的建立及数据的分析，我们发现建立的地点模型对人群关注的新闻类型有影响，并改动我们的模型，再进行进一步的分析，观察这种影响是否具有持续性。

第三步，分析各个板块的新闻类型是否具有共通性，分析单个人群关注的新闻与地点的关系，因此，再次改进我们的模型。

目录

一、问题重述 3

1.1 群众留言分类.....3
1.2 热点问题挖掘.....3
1.3 答复意见的评价.....5

二、模型建立 5

2.1 群众留言分类.....6
2.1.1 数据预处理.....6
2.1.2 分词并去除停用词.....6
2.1.3 文本特征选择.....9
2.1.4 分类器的选择.....10
2.1.5 模型的选择.....11
2.1.6 模型评估.....13
2.1.7 代码.....15
2.2 热点问题挖掘.....18
2.2.1 问题识别.....18
2.2.3 问题归类.....18

一、问题重述

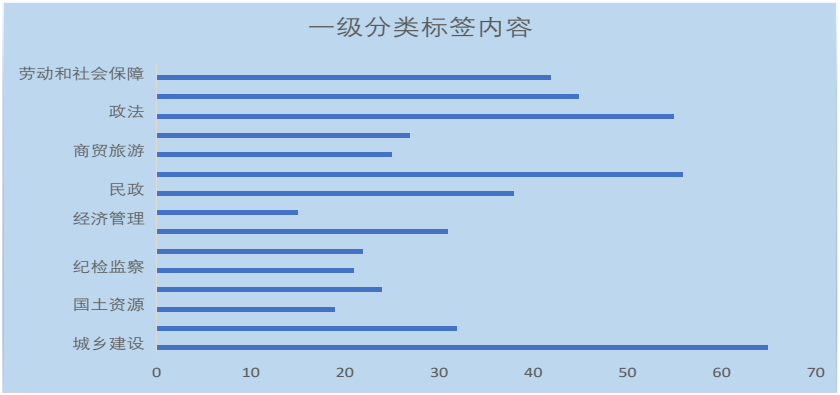
1.1 群众留言分类

对于网络搜集的群众留言，首先按照三级标签划分体系对留言进行初步分类，以便后续处理。由于人工进行再分类存在不同层度的问题，现要求根据已有数据，建立关于留言内容的一级标签分类模型。并用 F-Score 对模型方法进行评价:

$$F_1 = \frac{1}{\mathrm{n}} \sum_{\mathrm{i}=1} \frac{2\mathrm{P}i\mathrm{R}i}{Pi + Ri}$$

其中 Pi 为第 i 类的查准率， Ri 为第 i 类的查全率。

首先要清楚的是，一级标签虽然是对所有留言的大致分类，但其种类数目也不在少数，要想建立一级标签分类模型，就要对一级标签内容有所了解，详见下图 1。

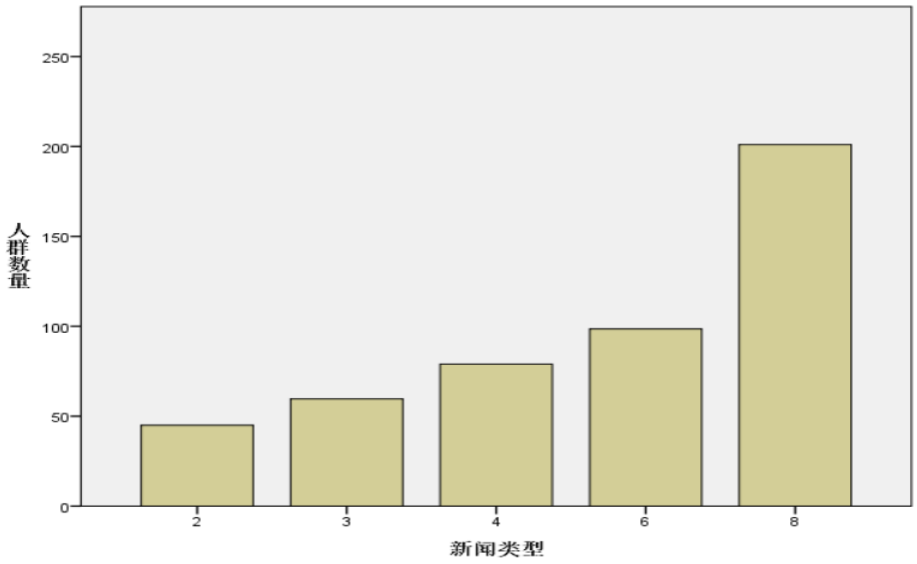


由图可以看出，不同的一级分类标签占比不同，除个别标签外数目较多外，其他标签数目相差不大。接下来根据附件 2 所给数据建立一级标签分类模型并用 F-Score 对模型进行评价。

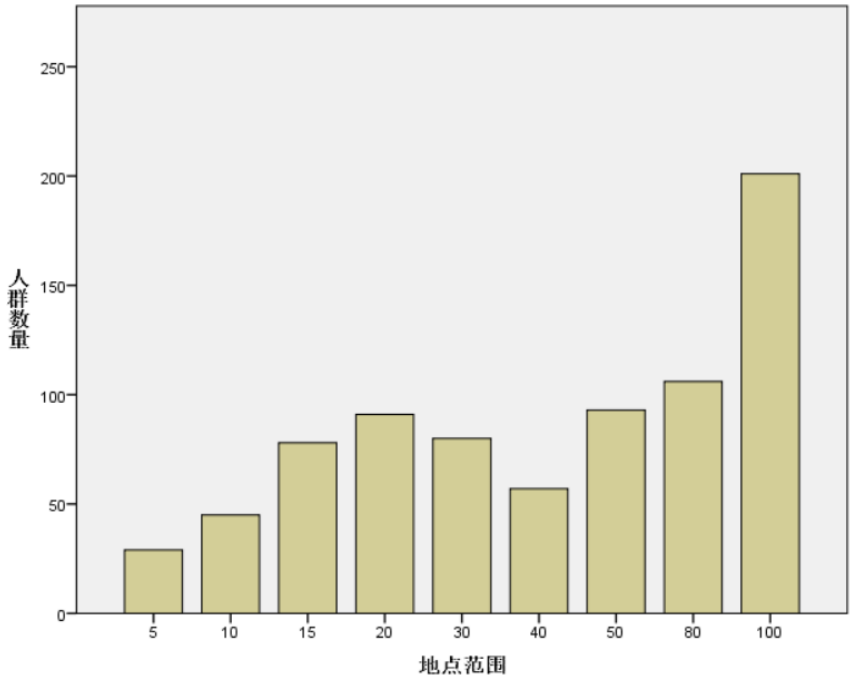
1.2 热点问题挖掘

利用附件 3 所给的的数据，找出在某一时段内有关特定地点或特定人群的留言并将其分类，既定义为同类问题。然后定义热度评价指标，得到相应的热度排名和热度指数。找出热度排名前 5 的问题，将其按照热度排名、问题 ID、

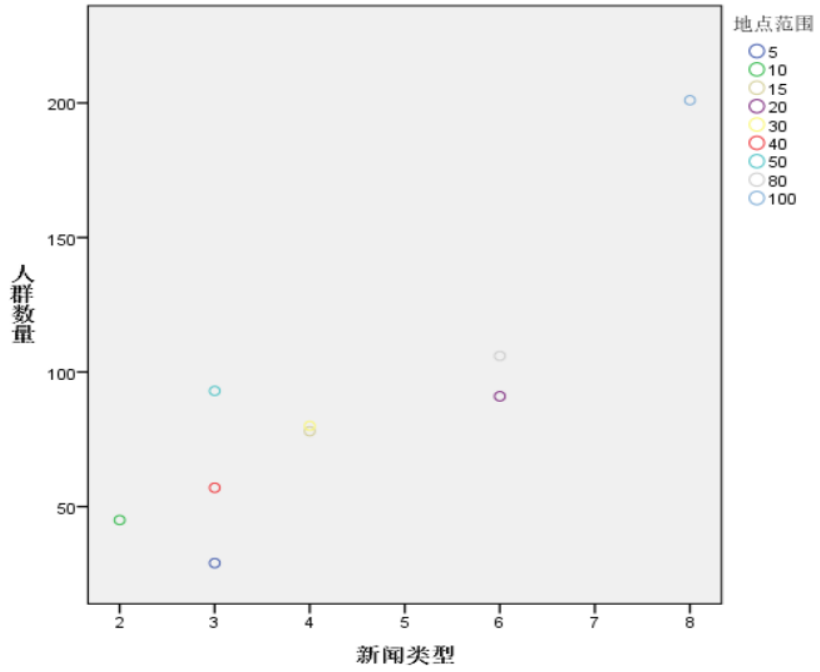
热度指数、时间范围、地点/人群、问题描述的顺序制作成“热点问题表.xls”。按照附件 3 中的内容再以问题 ID、留言编号、留言用户、留言主题、留言详情、点赞数、反对数的顺序为制作成“热点问题留言明细表.xls”。分析如下



第一种：如上图可知人群人数与新闻类型大部分成正相关，说明人群人数与新闻类型有关系，则更受人欢迎的新闻类型人群数则会更多，则新闻类型是影响人群人数的一个特征。



第二种如上图可知人群数量与地点范围呈波动趋势，有一定关系的同时并不能说明有直接的联系，但也不可排除这种情况，模型重新建立。



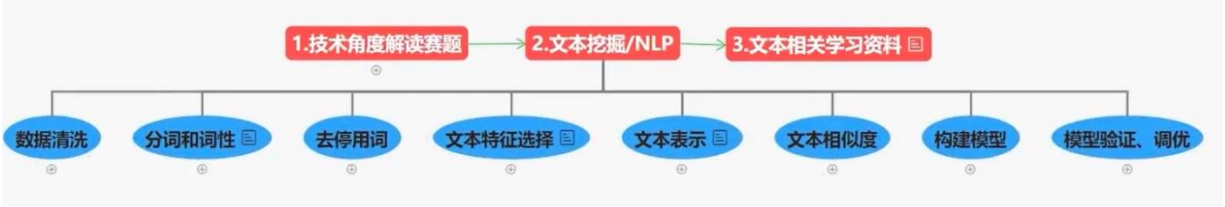
如上图将三者共同建立模型成分散式，没有直接关系但人群数量与新闻类型有关，则模型成立。

1.3 答复意见的评价

附件 4 为相关部门对留言的答复意见，从答复的相关性即答复意见是否与问题相关、完整性即是否满足某种规范、可解释性即答复意见中内容的相关解释等角度对答复意见的质量给出一套评价方案，并尝试实现。其中不同的答复意见内容多少不一，不便于后续的分析和评价，所以先将答复内容用数学方法进行特征提取，接着考虑如何对三个指标进行量化，以便总结出评价方案并实现。

二、模型建立

建模步骤



2.1 群众留言分类

2.1.1 数据预处理

通过人工观察数据发现，评论中夹杂许多特殊符号，对于本案例挖掘目标而言，这类数据本身没有实质性帮助，可以采用正则表达式去除文本中除字母，数字，汉字以外的所有符号。

2.1.2 分词并去除停用词

- **分词**

分词是文本信息处理的基础环节，是将一个单词序列切分成一个一个单词的过程。准确的分词可以极大的提高计算机对文本信息的是被和理解能力。相反，不准确的分词将会产生大量的噪声，严重干扰计算机的识别理解能力，并对这些信息的后续处理工作产生较大的影响。

汉语的基本单位是字，由字可以组成词，由词可以组成句子，进而由一些句子组成段、节、章、篇。可见，如果需要处理一篇中文语料，从中正确的识别出词是一件非常基础且重要的工作。

然而，中文以字为基本书写单位，词与词之间没有明显的区分标记。中文分词的任务就是把中文的序列切分成有意义的词，即添加合适的词串使得所形成的词串反映句子的本意，例子如表 0-1 所示。

表 0-1 中文分词例子

操作	内容
输入	我帮小明打饭
输出	我 帮 小明 打饭

当使用基于词典的中文分词方法进行中文信息处理时不得不考虑未登录词

的处理。未登录词指词典中没有登录过的人名、地名、机构名、译名及新词语等。当采用匹配的办法来切分词语时，由于词典中没有登录这些词，会引起自动切分词语的困难。常见的未登陆词有命名实体，如“张三”、“北京”、“联想集团”、“酒井法子”等；专业术语，如“贝叶斯算法”、“模态”、“万维网”；新词语，如“卡拉 OK”“美刀”、“啃老族”等。

另外，中文分词还存在切分歧义问题，如“当结合成分子时”这个句子可以有以下切分方法：“当/结合/成分/子时”，“当/结合/成/分子/时”，“当/结/合成/分子/时”，“当/结/合成分/子时”。

可以说，中文分词的关键问题为：切分歧义的消解和未登录词的识别。

词典匹配是分词最为传统也最为常见的一种办法。匹配方式可以为正向（从左到右）或逆向（从右到左）。对于匹配中遇到的多种分段可能性（segmentation ambiguity），通常会选取分隔出来词的数目最少的。

很明显，这种方式对词表的依赖很大，一旦出现词表中不存在的新词，算法是无法做到正确的切分的。但是词表匹配也有它的优势，比如简单易懂，不依赖训练数据，易于纠错等等。

还有一类方法是通过语料数据中的一些统计特征（如互信息量）去估计相邻汉字之间的关联性，进而实现词的切分。这类方法不依赖词表，特别是在对生词的发掘方面具有较强的灵活性，但是也经常会有精度方面的问题。

分词最常用的工作包是 jieba 分词包，jieba 分词是 python 写成的一个分词开源库，专门用于中文分词，其有三条基本原理，即实现所采用技术。

（1） 基于 trie 树结构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（DAG）。jieba 分词自带了一个叫做 dict.txt 的词典，里面有 2 万多条词，包含了词条出现的次数（这个次数是作者自己基于人民日报语料等资源训练得出来的）和词性。trie 树是有名的前缀树，若一个词语的前面几个字一样，表示该词语具有相同的前缀，可以使用 trie 树来存储，trie 树存储方式具有查找速度快的优势。后一句的“生成句子中汉字所有可能成词情况所构成的有向无环图”意思是给定一个待切分的句子，生成一个如图 0-1 所示的有向无环图。

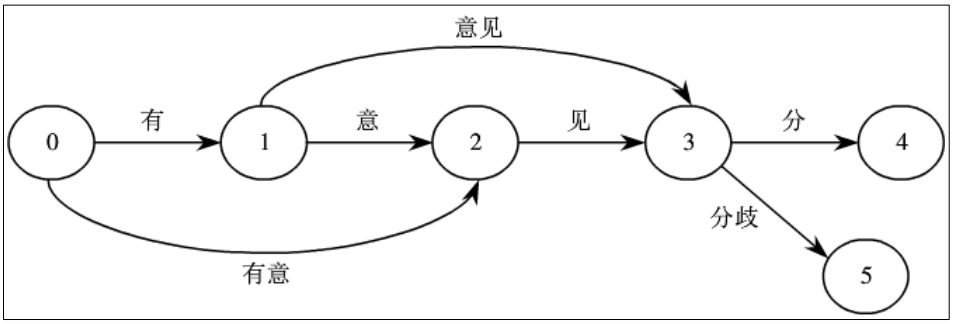


图 0-1 “有意见分歧”切分生成的有向无环图

（2）采用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。先查找待分词句子中已经切分好的词语，再查找该词语出现的频率，然后根据动态规划查找最大概率路径的方法，对句子从右往左反向计算最大概率（反向是因为汉语句子的重心经常落在右边，从右往左计算，正确率要高于从左往右计算，这个类似于逆向最大匹配），最后得到最大概率的切分组合。

（3）对于未登录词，采用 HMM 模型，使用了 Viterbi 算法，将中文词汇按照 BEMS 四个状态来标记。其中 B 是 begin，表示开始位置；E 是 end，表示结束位置；M 是 middle，表示中间位置；S 是 singgle，表示单独成词的位置。HMM 模型采用(B,E,M,S)这四种状态来标记中文词语，比如北京可以标注为 BE，即北/B 京/E，表示北是开始位置，京是结束位置，中华民族可以标注为 BMME，就是开始、中间、中间和结束。

- **去除停用词**

停用词（Stop Words），词典译为“电脑检索中的虚字、非检索用字”。在 SEO 搜索引擎中，为节省存储空间和提高搜索效率，搜索引擎在索引页面或处理搜索请求时会自动忽略某些字或词，这些字或词即被称为停用词。

停用词一定程度上相当于过滤词（Filter Words），区别是过滤词的范围更大一些，包含情色、政治等敏感信息的关键词都会被视做过滤词加以处理，停用词本身则没有这个限制。通常意义上，停用词大致可分为如下两类。

一类是使用十分广泛，甚至是过于频繁的一些单词。比如英文的“i”、“is”、“what”，中文的“我”、“就”等，这些词几乎在每个文档上均会出现，查询这样的词无法保证搜索引擎能够给出真正相关的搜索结果，因此无法缩小搜索范围来提高搜索结果的准确性，同时还会降低搜索的效率。因此，在搜索的时候，Google 和百度等搜索引擎会忽略掉特定的常用词，如果使用了太多的停用词，

有可能无法得到精确的结果，甚至可能得到大量毫不相关的搜索结果。

另一类是文本中出现频率很高，但实际意义又不大的词。这一类词主要包括了语气助词、副词、介词、连词等，通常自身并无明确意义，只有将其放入一个完整的句子中才有一定作用的词语。常见的有“的”、“在”、“和”、“接着”等，例如“泰迪教育研究院是最好的大数据知识传播机构之一”这句话中的“是”、“的”就是两个停用词。

经过分词后，评论由一个字符串的形式变为多个由文字或词语组成的字符串的形式，可判断评论中词语是否为停用词。根据上述停用词的定义整理出停用词库，并根据停用词库去除评论中的停用词。

2.1.3 文本特征选择

已提出的文本分类特征选择方法比较争，常用的方法有:文档频率(Document Frequency,DF)、信息增益(Information Gain, IG)、卡方(x2)校验(CHI)和互信息(Mutual Information,MI)等方法。

另外特征抽取也是一种特征降维技术，特征抽取通过将原始的特征进行变换运算，形成新的特征。

特征权重主要是经典的 TF-IDF 方法及其扩展方法，主要思路是一个词的重要度与在类别内的词频成正比，与所有类别出现的次数成反比。

这里采用 TF-IDF 和卡方(x2)校验的方法。

- **TF-IDF**

TF-IDF 的主要思想是：如果某个词或短语在一篇文章中出现的频率 TF 高，并且在其他文章中很少出现，则认为此词或者短语具有很好的类别区分能力，适合用来分类。TFIDF 实际上是：TF * IDF，TF 词频(Term Frequency)，IDF 逆向文件频率(Inverse Document Frequency)。

TF-IDF 的原始公式为：

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}}$$

$$idf_i = \log \frac{|D|}{|\{j : t_i \in d_j\}|}$$

公式中各个变量的含义在此不再赘述，与网上的 TF-IDF 公式一致。但是第二个公式由于分母可能为 0，因此此次我们采用平滑处理后的 TF-IDF 公式：

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}}$$

$$idf_i = \log \frac{|D|+1}{1+|\{j : t_i \in d_j\}|} + 1$$

作为源码实现所依据的公式。

- **卡方(x2)校验**

卡方检验是一种用途很广的计数资料的假设检验方法。它属于非参数检验的范畴，主要是比较两个及两个以上样本率(构成比）以及两个分类变量的关联性分析。其根本思想就是在于比较理论频数和实际频数的吻合程度或拟合优度问题

2.1.4 分类器的选择

这里选择朴素贝叶斯分类器：

朴素贝叶斯分类器最适合用于基于词频的高维数据分类器，最典型的应用如垃圾邮件分类器等,准确率可以高达 95%以上。这里我们使用的是 sklearn 的朴素贝叶斯分类器 MultinomialNB，我们首先将 review 转换成词频向量,然后将词频向量再转换成 TF-IDF 向量,还有一种简化的方式是直接使用 TfidfVectorizer 来生成 TF-IDF 向量(正如前面生成 features 的过程)，这里我们还是按照一般的方式将生成 TF-IDF 向量分成两个步骤：1.生成词频向量. 2.生成 TF-IDF 向量。最后我们开始训练我们的 MultinomialNB 分类器。

2.1.5 模型的选择

接下来我们尝试不同的机器学习模型,并评估它们的准确率，我们将使用如下四种模型:

- Logistic Regression(逻辑回归)
- (Multinomial) Naive Bayes(多项式朴素贝叶斯)
- Linear Support Vector Machine(线性支持向量机)
- Random Forest(随机森林)

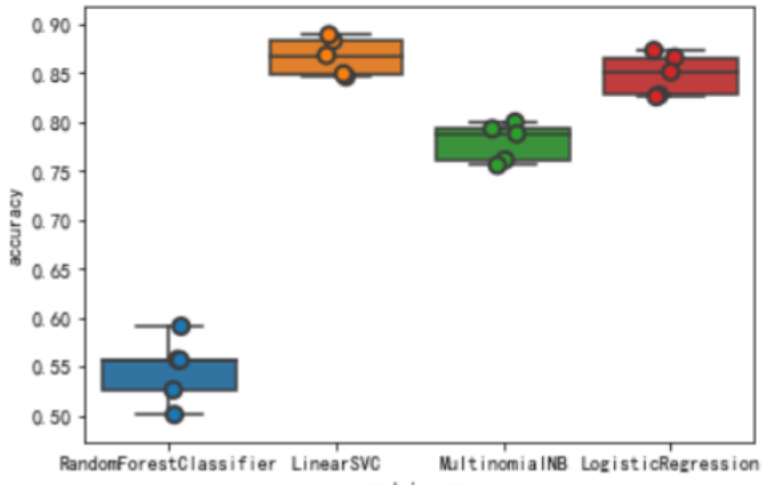
代码：

```
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.svm import LinearSVC
from sklearn.model_selection import cross_val_score

models = [
    RandomForestClassifier(n_estimators=200, max_depth=3, random_state=0),
    LinearSVC(),
    MultinomialNB(),
    LogisticRegression(random_state=0),
]
CV = 5
cv_df = pd.DataFrame(index=range(CV * len(models)))
entries = []
for model in models:
    model_name = model.__class__.__name__
    accuracies = cross_val_score(model, features, labels, scoring='accuracy', cv=CV)
```

```
    for fold_idx, accuracy in enumerate(accuracies):
        entries.append((model_name, fold_idx, accuracy))
cv_df = pd.DataFrame(entries, columns=['model_name', 'fold_idx', 'accuracy'])

import seaborn as sns
sns.boxplot(x='model_name', y='accuracy', data=cv_df)
sns.stripplot(x='model_name', y='accuracy', data=cv_df,
              size=8, jitter=True, edgecolor="gray", linewidth=2)
plt.show()
```



从可以箱体图上可以看出随机森林分类器的准确率是最低的，因为随机森林属于集成分类器(有若干个子分类器组合而成)，一般来说集成分类器不适合处理高维数据(如文本数据)，因为文本数据有太多的特征值，使得集成分类器难以应付，另外三个分类器的平均准确率都在 80%以上。其中线性支持向量机的准确率最高。

代码：

```
cv_df.groupby('model_name').accuracy.mean()
```

结果：

```
model_name
LinearSVC                0.867026
LogisticRegression       0.848626
MultinomialNB            0.779505
RandomForestClassifier   0.546457
Name: accuracy, dtype: float64
```

我们看到线性支持向量机的平均准确率达到了 86.7%，其次是逻辑回归和朴素贝叶斯。

2.1.6 模型评估

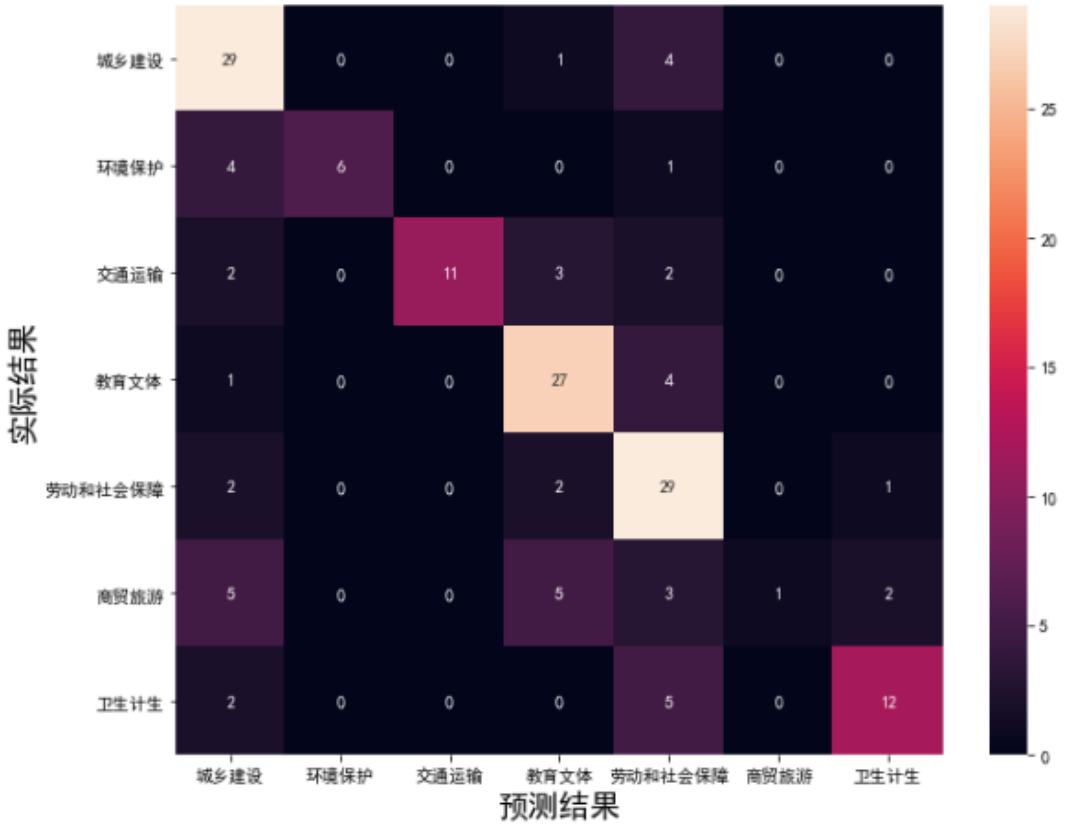
下面我们就针对平均准确率最高的 LinearSVC 模型，我们将查看混淆矩阵，并显示预测标签和实际标签之间的差异。

代码：

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix
#训练模型
model = LinearSVC()
X_train, X_test, y_train, y_test, indices_train, indices_test = train_test_split(features, labels, data1. 一 级 分 类 , test_size=0.33, stratify=labels, random_state=0)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)

#生成混淆矩阵
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei']
plt.rcParams['axes.unicode_minus'] = False
conf_mat = confusion_matrix(y_test, y_pred)
fig, ax = plt.subplots(figsize=(10,8))
sns.heatmap(conf_mat, annot=True, fmt='d',
```

xticklabels=cat_id_df.一级分类.values,
yticklabels=cat_id_df.一级分类.values)
plt.ylabel('实际结果',fontsize=18)
plt.xlabel('预测结果',fontsize=18)
plt.show()



混淆矩阵的主对角线表示预测正确的数量,除主对角线外其余都是预测错误的数量.从上面的混淆矩阵可以看出"环境保护"类预测最准确,没有预测错误。“城乡建设”预测的错误数量教多。

多分类模型一般不使用准确率(accuracy)来评估模型的质量,因为accuracy 不能反应出每一个分类的准确性,因为当训练数据不平衡(有的类数据很多,有的类数据很少)时，accuracy 不能反映出模型的实际预测精度,这时候我们就需要借助于 F1 分数、ROC 等指标来评估模型。

2.1.7 代码

```
import pandas as pd
import jieba.analyse
import re
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.naive_bayes import MultinomialNB
from sklearn.feature_selection import chi2
from sklearn.svm import LinearSVC
import numpy as np

data=pd.read_excel('D:/python/附件 2.xlsx',encoding='gbk')
#print(data.isnull().sum())
data1=data[['留言详情','一级标签']]
data1['cat_id'] = data1['一级标签'].factorize()[0]
cat_id_df = data1[['一级标签','cat_id']].drop_duplicates().sort_values('cat_id').reset_index(drop=True)
cat_to_id = dict(cat_id_df.values)
id_to_cat = dict(cat_id_df[['cat_id', '一级标签']].values)
data1.sample(5)
```

	留言详情	一级标签	cat_id
5585	\n \n 本人是一个尿毒症患者，需要长期透析治疗。现在报销是通过特殊门...	劳动和社会保障	4
978	\n\t\t\t\t\t\n\t\t\t\t\t K市正在创建国家文明城市，城市建设越来越...	城乡建设	0
4335	\n \n 3.22日教育部已正式给省政府下文，合并E市学院和E市医专，...	教育文体	3
6405	\n \n 尊敬的易书记：您好！ 为什么这次A1区政府机关、街道在职在...	劳动和社会保障	4
7906	\n\t\t\t\t\t\n\t\t\t\t\t尊敬的市领导：A9市开心广场小区5栋（喜悦楼...	商贸旅游	5

#过滤字母,数字，汉字以外的所有符号

```
p=re.compile(u"[^a-zA-Z0-9\u4E00-\u9FA5]")
data1['留言详情']=data1['留言详情'].apply(lambda x:re.sub(p,'',x))
print(data1.head())
```

```
                                                留言详情  一级标签  cat_id
0  A3区大道西行便道未管所路口至加油站路段人行道包括路灯杆被圈西湖建筑集团燕子山安置房项目施工...  城乡建设       0
1  位于书院路主干道的在水一方大厦一楼至四楼人为拆除水电等设施后烂尾多年用护栏围着不但占用人行道...  城乡建设       0
2  尊敬的领导A1区苑小区位于A1区火炬路小区物业A市程明物业管理有限公司未经小区业主同意利用业...  城乡建设       0
3  A1区A2区华庭小区高层为二次供水楼顶水箱长年不洗现在自来水龙头的水严重霉味大家都知道水是我...  城乡建设       0
4  A1区A2区华庭小区高层为二次供水楼顶水箱长年不洗现在自来水龙头的水严重霉味大家都知道水是我...  城乡建设       0
```

#分词，并过滤停用词

```
stopwords=pd.read_csv('D:/python/chineseStopWords.txt',index_col=Fals
e,quoting=3,sep='\t',names=['stopword'], encoding='gbk')
data1['留言详情'] = data1['留言详情'].apply(lambda x: " ".join([w for
w in list(jieba.cut(x)) if w not in stopwords]))
print(data1.head())
```

```
                                                留言详情  一级标签  cat_id
0  A3 区 大道 西行 便道 未管 所 路口 至 加油站 路段 人行道 包括 路灯 杆 被 圈...  城乡建设       0
1  位于 书院 路 主干道 的 在水一方 大厦 一楼 至 四楼 人为 拆除 水电 等 设施 后 ...  城乡建设       0
2  尊敬 的 领导 A1 区苑 小区 位于 A1 区 火炬 路 小区 物业 A 市程明 物业管理...  城乡建设       0
3  A1 区 A2 区华庭 小区 高层 为 二次 供水 楼顶 水箱 长年 不洗 现在 自来水 龙...  城乡建设       0
4  A1 区 A2 区华庭 小区 高层 为 二次 供水 楼顶 水箱 长年 不洗 现在 自来水 龙...  城乡建设       0
```

#文本特征选择

```
tfidf = TfidfVectorizer(norm='l2', ngram_range=(1, 2))
features = tfidf.fit_transform(data1.留言详情)
labels = data1.cat_id
print(features.shape)
print('------------------------------')
print(features)
N = 10

for 一级分类,cat_id in sorted(cat_to_id.items()):
    features_chi2 = chi2(features, labels == cat_id)
    indices = np.argsort(features_chi2[0])
```

```
    feature_names = np.array(tfidf.get_feature_names())[indices]
    unigrams = [v for v in feature_names if len(v.split(' ')) == 1]
    bigrams = [v for v in feature_names if len(v.split(' ')) == 2]
    print("# '{}':".format(一级标签))
    print("    . Most correlated unigrams:\n                .
{}".format('\n       . '.join(unigrams[-N:])))
    print("    . Most correlated bigrams:\n                .
{}".format('\n       . '.join(bigrams[-N:])))

#生成训练集
X_train, X_test, y_train, y_test = train_test_split(data1['留言详情'],
data1['cat_id'], random_state = 0)
count_vect = CountVectorizer()
X_train_counts = count_vect.fit_transform(X_train)
tfidf_transformer = TfidfTransformer()
X_train_tfidf = tfidf_transformer.fit_transform(X_train_counts)
clf = MultinomialNB().fit(X_train_tfidf, y_train)

#预测函数 myPredict
def myPredict(sec):
    format_sec=" ".join([w for w in list(jieba.cut(sec)) if w not in
stopwords])
    pred_cat_id=clf.predict(count_vect.transform([format_sec]))
    print(id_to_cat[pred_cat_id[0]])

#检验
myPredict('假如 L4 县现在所有的财富(包括私人和单位)是一千亿人民币，如何
让它变多，只有通过贸易顺差来实现。而事实上，L4 县的主要出口只有煤炭和劳
务，而太多方面都在进口(手机，饮料，电器，服装等等等等)，光我自己从事的
```

行业每年各个店加起来要 2000 多万，其它行业数字更大，综合来说进口应该是大于出口的，也就是说一千亿人民币是逐年在减少的。希望 L4 县的真正人才们多做些能增加出口的事让一千亿变多起来,而不是只热衷于从外面进货赚钱。 ’)

结果：

商贸旅游

2.2 热点问题挖掘

2.2.1 问题识别

命名实体识别（Named Entity Recognition，简称 NER)，又称作“专名识别”，是指识别文本中具有特定意义的实体，主要包括人名、地名、机构名、专有名词等。通常包括两部分：（1）实体边界识别；（2） 确定实体类别（人名、地名、机构名或其他)。

命名实体识别通常是知识挖掘、信息抽取的第一步，被广泛应用在自然语言处理领域。主要有两种命名实体识别的方法。

NLTK：由宾夕法尼亚大学计算机和信息科学使用 python 语言实现的一种自然语言工具包，其收集的大量公开数据集、模型上提供了全面、易用的接口，涵盖了分词、词性标注(Part-Of-Speech tag, POS-tag)、命名实体识别(Named Entity Recognition, NER)、句法分析(Syntactic Parse)等各项 NLP 领域的功能。

NER 将文本中的实体按类标记出来，例如人名，公司名，地区，基因和蛋白质的名字等。

NER 基于一个训练而得的 Model（模型可识别出 Time, Location, Organization, Person, Money, Percent, Date）七类属性，其用于训练的数据即大量人工标记好的文本，理论上用于训练的数据量越大，NER 的识别效果就越好。

2.2.3 问题归类

应用 TF-IDF 方法，详情如 2.1.3 文本特征选择