C 题:“智慧政务”中的文本挖掘应用

摘要

文本数据挖掘是利用某些方法比如自然语言处理(Natural language processing (NLP))技术把一堆没有结构的数据而处理成有结构的数据的一种人工智能技术,而处理后的这些有结构的数据可以作为机器学习和深度学习模型的输入,也可以直接分析这些数据产生想要的结果。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。本报告通过自然语言处理,实现对“智慧政务”中中文文本的分类处理,文本挖掘可以大致定义为一个知识密集型过程,在该过程中,用户通过使用一套分析工具随时间与文档集进行交互。以类似于数据挖掘的方式,文本挖掘试图通过识别和探索感兴趣的模式从数据源中提取有用的信息。本报告针对“智慧政务”中的文本挖掘应用的群众留言分类和热点问题挖掘两个问题设计出相应的算法,以得到测试数据和这类问题的解决方法。该算法利用中文文本分类的七个步骤进行算法设计,中文文本分类流程:1. 预处理,2. 中文分词,3. 结构化表示-构建词向量空间,4.权重策略-TF-IDF,5. 分类器 6. 评价。最核心的部分是程序编写,使用 Python 语言来实现总体需求,本报告最后分析了“智慧政务”中的文本挖掘应用的思路分析与实用价值,利用自然语言(NLP)技术使得政府部门在数据处理这一方面能够更加便捷准确,大大减轻了相关政府人员的工作量,这也成为未来数据分类处理的主要方式。

关键词:分类器;TF-IDF;NLP;热点挖掘;词频分析

Abstract

Text data mining is the use of certain methods such as Natural language processing (Natural language processing (NLP) technology to a pile of no structure data and processing data into a structure of a kind of artificial intelligence technology, and the structure of the processed these data can be as input of machine learning and deep learning model, also can analysis the data directly to produce the desired results. With the development of big data, cloud computing, artificial intelligence and other technologies, the establishment of intelligent government affairs system based on natural language processing technology has become a new trend of social governance innovation and development, which plays a great role in promoting the management level and efficiency of the government. Text mining can be roughly defined as a knowledge-intensive process in which users interact with document sets over time by using a set of analysis tools. In a manner similar to data mining, text mining attempts to extract useful information from data sources by identifying and exploring patterns of interest. This report designs the corresponding algorithm for the two problems of text mining application in "smart government", namely, the classification of public comments and the mining of hot issues, in order to get the test data and the solution of these problems. This algorithm USES seven steps of Chinese text classification to carry out algorithm design, Chinese text classification process: 1. Preprocessing, 2. Chinese word segmentation, 3. Structured representation - construction of word vector space, 4. Weight strategy - TF-IDF, 5. Classifier 6. Evaluation. Is the most core part of programming, using the Python language to realize the overall demand, the end of this report analyzes "smart government" the thinking of the text mining application analysis and the practical value, the use of natural language in data processing (NLP) technology has made the government department that on the one hand can more convenient and accurate, and greatly reduce the workload of relevant government personnel, it also become the main way of the future data classification process.

Key words: classifier； TF-IDF; NLP; hotspot mining; word frequency analysis

目录

1 前言.....4
1.1 问题的背景和意义.....4
1.2 本文的主要工作.....4
1.3 本文的组织结构.....4
2 文本挖掘.....4
2.1 什么是文本挖掘.....4
2.2 文本挖掘技术的发展.....5
2.3 文本挖掘主要步骤.....5
2.4 文本挖掘的关键技术.....5
2.5 文本挖掘应用前景.....6
3 解决问题.....6
3.1 群众留言分类.....6
3.2 热点问题挖掘.....9
3.3 文件夹说明.....11
4 实验结论.....11
4.1 程序测试.....11
4.2 样本数据测试结果.....14
5.参赛感想.....14
致谢.....15
参考文献.....16

1 前言

1.1 问题的背景和意义

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。

1.2 本文的主要工作

本文主要通过文本挖掘技术方法之一朴素贝叶斯分类器进行 C 题的问题的分析与解决方案，通过自然语言处理中文文本分类的方法得到测试数据结果，再对得出数据进行分析，对群众留言分类问题和热点问题挖掘得出结果数据，对刚发的准确性和实用性进行评价。

1.3 本文的组织结构

本文主要分为五章:
第一章，　主要介绍问题背景与本文的工作和目的，以及结构安排。
第二章，　介绍了文本挖掘的概念和运用发展，还有文本挖掘技术的发展方向。
第三章，　解决赛题的分析思路和步骤与具体分类方法。
第四章，　程序运行结果与准确度分析，数据汇总。
第五章，　参赛感想，对今后数据挖掘方面的学习思路。

2 文本挖掘

2.1 什么是文本挖掘

文本挖掘是指从大量文本数据中抽取事先未知的、可理解的、最终可用的知识的过程，同时运用这些知识更好地组织信息以便将来参考。直观的说，当数据挖掘的对象完全由文本这种数据类型组成时，这个过程就称为文直观的说，当数据挖掘的对象完全由文本这种数据类型组成时，这个过程就称为文本挖掘[1]。文本挖掘也称为文本数据挖掘。简单来说，文本挖掘是将文本信息转化为可利用的数据的知识

2.2 文本挖掘技术的发展

文本数据挖掘处理的数据类型是文本数据，属于数据挖据的一个分支，与机器学习、自然语言处理、数理统计等学科具有紧密联系。文本挖掘作为数据挖掘的一个分支学科，其底层技术包括机器学习、数理统计、自然语言处理等领域的技术方法。进阶技术是文本挖掘的基本技术，面向不同的应用，分为五大类：信息抽取、文本分类、文本聚类、文本数据压缩、文本数据处理[2]。

应用领域，文本挖掘最终的目的如其定义中所描述的，信息访问与知识发现，信息访问包括信息检索、信息浏览、信息过滤和信息报告，知识发现包括数据分析和数据预测[3]。

2.3 文本挖掘主要步骤

文本挖掘的过程来自各种数据源的文本数据通过挖掘处理到最终用户主要经过三个过程：文档预处理、特征信息提取和数据挖掘。

（1）文档预处理。当来自各种信息源的文档到达服务器时，首先对文档进行过滤，对文档的类型进行鉴别。根据文档可能类型的特征可分为：结构化文档和非结构化文档。过滤器对不同类型的文档提供不同的文本过滤方法。对于结构化文档，过滤器把文档分成各自的组成部分如：标题、摘要、主要内容、参考目录等。在这一步骤中，不同形式的文档（word、PDF、图片、图像等）都用 XML 语言转化成新的相同（或相似）的形式，例如（标题）、（作者）、（摘要）和（全文）等。而对于非结构化的文档，必须要通过语言预处理，把它转化为可用算术分析的形式，以便在下一步骤中能对文档进行自动的特征信息提取。它能利用语法知识把句子分解出基本部分，包括名词、动词、形容词、日期、货币、数字等，并从标题或摘要或全部文档中选出新的关键词。

（2）特征信息提取。特征信息的提取使非结构化数据转化成可以直接记录在数据库中的结构化数据，这为下一步骤的挖掘处理做了充分的准备。特征提取主要是识别文本中代表其特征的词项。提取的特征大部分是文本集中表示的概念，这些概念包含着重要的信息，因此要提前定义哪些信息必须被抽取和被怎样抽取，这需要有较好的专业知识。目前使用的方法主要有向量空间模型和布尔模型两种，其中向量空间模型是近年来应用较多并且效果较好的方法之一。

（3）模式评估与表示为最后一个环节，是利用已经定义好的评估指标对获取的知识或模式进行评价。如果评价结果符合要求，就存储该模式以备用户使用；否则返回到前面的某个环节重新调整和改进，然后再进行新一轮的发现。

2.4 文本挖掘的关键技术

文本挖掘的主要支撑技术：自然语言处理和机器学习由于处理的对象是半结构化或非结构化的文档，自然语言处理技术成为实现生物医学文本挖掘的主要技术手段。

（1）自然语言处理技术自然语言处理是主要研究人与计算机交际中的语言问题的一N学科。“自然语言处理要研制表示语言能力语言应用的模型，建立计算机框架来实现这样的语言模型，提出相应的方法来不断完善这样的语言模型，根据这样的语言模型设计各种实用

系统，并探讨这些实用系统的评测技术，更简单直观的说法，就是采用计算机技术来研究和处理自然语言。由于自然语言处理是一个多边缘的交叉学科,除语言学外还涉及计算机科学、数学、统计学、电子工程、心理学、哲学以及生物学等知识领域，它是在各个相关学科的交融和协作中逐渐成长起来的。在历史上，自然语言处理曾经在计算机科学、电子工程、语言学和心理认知语言学等不同的领域分别进行过研究。1956 年以前，人们主要进行自然语言处理的基础性研究工作[4]。

（2）机器学习方法机器学习研究计算机怎样模拟或实现人类的学习行为以获取新的知识或技能重新组织已有的知识结构使之不断改善自身的性能。它是人工智能领域的一个重要分支。机器学习从研究人类学习行为出发，研究一些基本方法（如：归纳、一般化、特殊化、类比等）去认识客观世界，获取各种知识和技能，以便对人类的认识规律进行探索，深入了解人类的各种学习过程，借助于计算机科学和技术原理建立各种学习模型，从而为计算机系统赋予学习能力。为了实现这一目的的理论、方法和工程构成了机器学习的主要任务[5]。此外，机器学习还有另一个基本目标，就是从理论上探索一些人类尚未发现的新学习方法和途径。学习能力是智能行为的一个非常重要的特征，但至今对学习的机理尚不清楚。人们曾对机器学习给出各种定义[6]。

2.5 文本挖掘应用前景

利用文本挖掘技术处理大量的文本数据，无疑将会给政府部门和社会发展带来巨大的贡献。因此，目前对于文本挖掘的需求非常强烈，文本挖掘技术应用前景广阔。

3 解决问题

3.1 群众留言分类

3.1.1 问题分析

在处理网络问政平台的群众留言时,工作人员首先按照一定的划分体系（参考附件 1 提供的内容分类三级标签体系）对留言进行分类，以便后续将群众留言分派至相应的职能部门处理。目前，大部分电子政务系统还是依靠人工根据经验处理，存在工作量大、效率低，且差错率高等问题。请根据附件 2 给出的数据，建立关于留言内容的一级标签分类模型。

按照题目要求可了解问题需要进行中文文本分类的分类处理,其中按照朴素贝叶斯分类算法进行训练分类器，对原始数据的分词，建立空间向量，训练分类器。然后使用 F-Score 对分类方法进行评价，计算 F1 的值，其中 Pi 为第 i 类的查准率，Ri 为第 i 类的查全率。

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

3.1.2 解决方法

在源程序中 NB.py 是朴素贝叶斯分类器。该源文件里面有对原始数据的分词，建立空间向量，训练分类器。其功能为将未知类别留言进行分类，并将分类好的留言移动到相应类别文件中。中文文本分类的中文文本分类流程：

1. 预处理
2. 中文分词
3. 结构化表示-构建词向量空间
4. 权重策略-TF-IDF
5. 分类器
6. 评价[7]

1.预处理：在 NB.py 中导入 jieba.jar 库(可能需要自己去官网下载安装，有些 python 版本不自带)，进行中文分词，jieba 是优秀的中文分词第三方库，中文文本需要通过分词获得单个的特征词。2. 中文分词：在本地创建 corpus ，seg 文件夹，corpus_path 是未分词语料库路径 ,seg_path 是分词后语料库存储路径。通过程序实现获取每个目录（类别）下所有的文件，其中子目录的名字就是类别名，例如：train_corpus/art/21.txt 中，'train_corpus/'是 corpus_path，'art'是 catelist 中的一个成员。然后拼出分类子目录的路径如： train_corpus/art/ 。拼出分词后存贮的对应目录路径如：train_corpus_seg/art/ ，判断是否存在分词目录，如果没有则创建该目录，获取未分词语料库中某一类别中的所有文本。然后遍历类别目录下的所有文件，拼出文件名全路径如：train_corpus/art/21.txt，读取文件内容。图 1Python 目录下需要创建的文件夹，图 2 是按照附件 2 中的类别创建分类文件夹

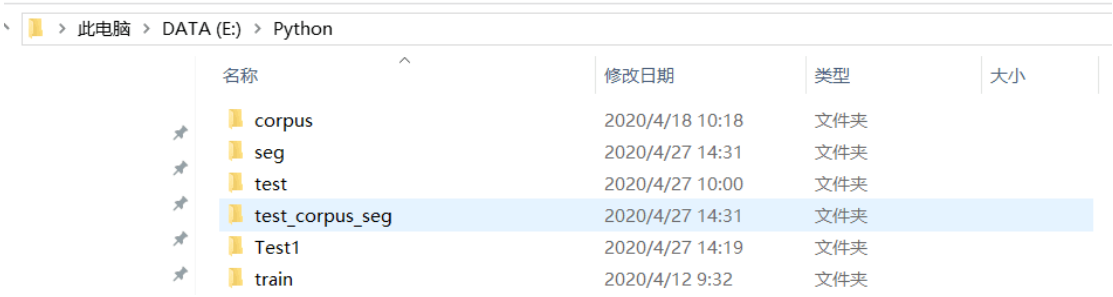


图 1

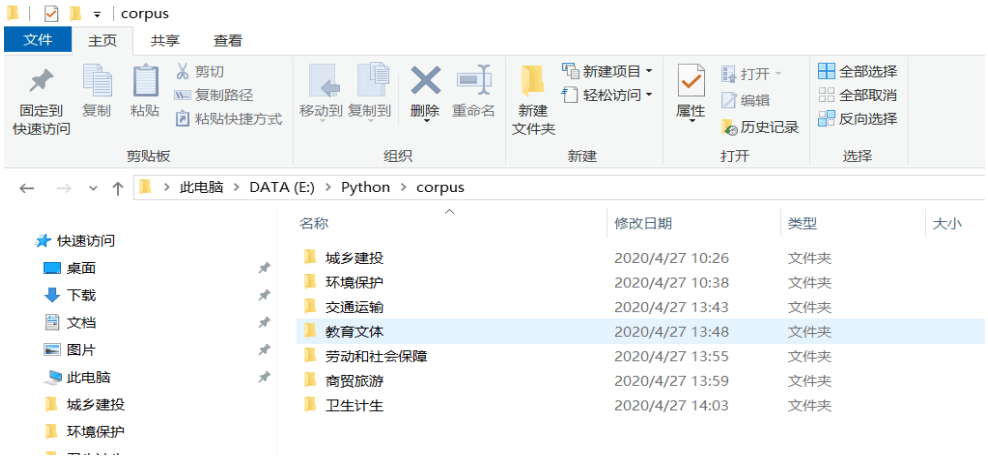


图 2

此时，content 里面存贮的是原文本的所有字符，例如多余的空格、空行、回车等等，接下来，我们需要把这些无关痛痒的字符统统去掉，变成只有标点符号做间隔的紧凑的文本

内容。删除换行，删除空行、多余的空格，为文件内容分词，最后将处理后的文件保存到分词后语料目录。后面对训练集进行分词。3. 结构化表示-构建词向量空间：创建 bunch 向量，import _pickle as pickle 导入 cPickle 包并且取一个别名 pickle，from sklearn.datasets import base 正确写法（from sklearn.datasets.base import Bunch 错误写法）。读取文件内容 ， 将 bunch 存储到 wordbag_path 路径中。对测试集进行 Bunch 化操作。4. 权重策略-TF-IDF：然后进入权重策略-TF-IDF 部分，TF-IDF（term frequency - inverse document frequency）是一种用于信息检索与数据挖掘的常用加权技术，常用于挖掘文章中的关键词，而且算法简单高效，常被工业用于最开始的文本数据清洗。

```
if __name__ == '__main__':
stopword_path = r"E:/python/train/train_word_bag/hlt_stop_words.txt"
bunch_path = r"E:/python/train/train_word_bag/train_set.dat"
space_path = r"E:/python/train/train_word_bag/tfdifspace.dat"
vector_space(stopword_path,bunch_path,space_path)
```

5. 朴素贝叶斯分类器，读取 bunch 对象，导入训练集，导入测试集，训练分类器：输入词袋向量和分类标签，alpha:0.001 alpha 越小，迭代次数越多，精度越高。如果进行多次交叉检验，可以发现朴素贝叶斯分类器在这个数据集上能够达到 80%以上的准确率。如果你亲自测试一下，会发现 KNN 分类器在该数据集上只能达到 60%的准确率，相信你对朴素贝叶斯分类器应该能够刮目相看了。且要知道，情感分类这种带有主观色彩的分类准则，连人类都无法达到 100%准确。

```
clf = MultinomialNB(alpha=0.001).fit(train_set.tdm, train_set.label)
```

Date.py 是将 excel 文件数据加载进 Python 中进行数据处理。开始导入 pandas 包，Pandas 是一个开源的，BSD 许可的库，为 Python 编程语言提供高性能，易于使用的数据结构和数据分析工具。由于九千多行无法显示完全。要几百行几百行的输出，通过更改 start 和 end 的值来确定输出。也可通过写一个 Logger 类将控制台输出重定向到一个文档文件，这时便不会出现显示不完全的情形，但不能马上在文本文件中看到输出，需要先保存你所更改的 python 文件，这是由于输出缓存问题，本文不做过多的描述。

```
workbook=xlrd.open_workbook(r'D:\desktop\01040730kg73\C 题全部数据\附件
2.xlsx')
table =workbook.sheets()[0]     # 通过索引打开……
all_content = []
start=5 #开始的行
end=1200  #结束的行
rows=end-start
```

Build.py 是对文档中的每条留言建立一个文本文档，并且重命名的，例如 0.txt，1.txt，2.txt……

```
arr = txt.split('\n') # 用//分割内容
print(arr) # 得到: ['\nabc1', 'abc2', 'abc3', '\nabc4', 'abc5', 'abc6\n']
for i, v in enumerate(arr): # 遍历 arr 每个元素生成一个文件
    if v == '': continue # 跳过内容为空的
    intab = "?*/\|.:><"
    outtab = "         "
    trantab = str.maketrans(intab, outtab)
    v=v.replace("\"","")
    title = v.translate(trantab)
```

```
    with open(r'E:\Python\test\未知\\'+str(title)+ '.txt',
'w',encoding='utf-8') as f:
        f.write(v)
```

图 3 是分类结果的示例样式，城乡建设类别里面的留言内容。

图 3

3.2 热点问题挖掘

3.2.1 问题分析与解决方法

留言的热点问题，简单来说就是在所有留言中，把一段时间内，同一地点，留言内容相似的问题，汇总出来后，对每个问题的次数进行统计，得出热度最高的五个热度问题，并且根据模板内容导出到“热点问题表.xls”和“热点问题留言明细表.xls”

步骤就是先通过 HotType.py 这个对每个时段的留言进行统计，然后通过 Build.py 对每个时段的每条留言建立一个文本文档，再通过 NB.py 进行分类。最后在通过 FindHot.py 找热点问题并建表。图 4 是各个时间段的留言数汇总表。

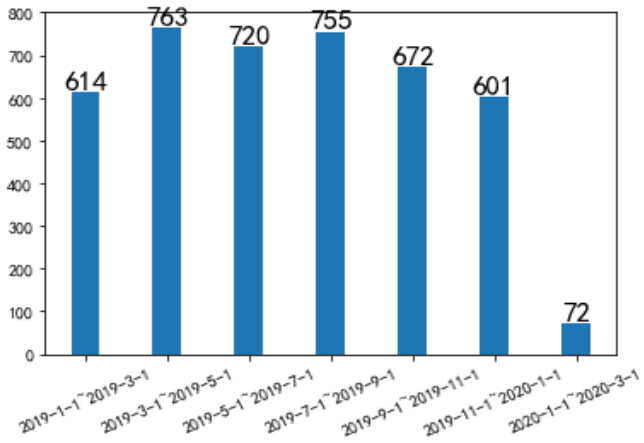


图 4

各时间段各类留言统计图见附件“各时段各类留言统计.doc”。

3.2.2 源程序设计

HotType.py 是将留言安装时间段分类，分出每个时间段中的每一类留言的个数，通过改变 start 和 end 的值进行每个时间段为两个月的筛选，筛选出每一类问题的数量。

```
try:
     dev_create_time = datetime.strptime(table.cell(i, 3).value,
"%Y/%m/%d %H:%M:%S").strftime("%Y-%m-%d %H:%M:%S")
    except:
     dev_create_time = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    arr1.append(dev_create_time)
    #print(table.cell(i, 5).value,end=' ')
f=pd.date_range(start='2018-5-1',end='2018-7-1',freq='s')
arr=[]
arr.append(f)
for i in range(len(arr)):
    for j in range(len(arr1)):

        if any(arr[i]==arr1[j]):#元素相等时

           print(table.cell(j+1, 2).value)
```

Build.py 和 NB.py 是前一个问题的同样方法，FindHot.py 源程序是对热点留言问题进行筛选，

```
for file_path in file_list:  # 遍历类别目录下的所有文件

    fullname = class_path + file_path  # 拼出文件名全路径

    content = readfile(fullname)  # 读取文件内容

    content = content.replace(" ", "")
    arr.append(content)
print(Counter(arr))
```

table.py 是对热点问题建立表

```
worksheet = workbook.add_sheet('data')

#生成 excel 文件

def generate_excel1(rec_data1):
    row = 0
    col = 0
    for item in (rec_data1):

        # 使用 write_string 方法，指定数据格式写入数据

             worksheet.write(row, col, str(item['sku_id']))
```

```
            worksheet.write(row, col + 1, item['sku_title'])
            worksheet.write(row, col + 2, str(item['id_1']))
            worksheet.write(row, col + 3, item['id_1_doc'])
            worksheet.write(row, col + 4, str(item['id_2']))
            worksheet.write(row, col + 5, item['id_2_doc'])

    workbook.save('热点问题表.xls')

if __name__ == '__main__':

    rec_data1 = [{'sku_id':u'热度排名','id_1':u'热度指数','id_2':u'地点/人群

','id_2_doc':u'问题描述','sku_title':u'问题 ID','id_1_doc':u'时间范围\n'}]

    generate_excel1(rec_data1)
```

3.3 文件夹说明

corpus 是附件 2 各类的数据，seg 是 corpus 分词后的数据，train 是存储训练数据的一些特征向量文件，test 是测试数据，test_corpus_seg 是测试数据分词后的保留文件夹，Test1 是存放测试数据特征向量空间文件。Type 是未知类别测试数据使用 NB.py 分类后保存的路径。

4 实验结论

4.1 程序测试

4.1.1 源代码片段

NB.py 部分代码

```
import jieba

# 配置 utf-8 输出环境

import imp
imp.reload(sys)

# 保存至文件

def savefile(savepath, content):
    with open(savepath, "wb") as fp:
        content=content.encode()
        fp.write(content)

# 读取文件
```

```
def readfile(path):
    with open(path, "rb") as fp:
        content = fp.read()
    return content
def corpus_segment(corpus_path, seg_path):
#corpus_path 是未分词语料库路径

#seg_path 是分词后语料库存储路径

    catelist = os.listdir(corpus_path)# 获取 corpus_path 下的所有子目录

# 获取每个目录（类别）下所有的文件

#其中子目录的名字就是类别名，例如：train_corpus/art/21.txt 中，'train_corpus/'是

corpus_path，'art'是 catelist 中的一个成员

    for mydir in catelist:

#这里 mydir 就是 train_corpus/art/21.txt 中的 art（即 catelist 中的一个类别）
```

Date.py 部分代码

```
import pandas as pd

# 设置显示的最大列、宽等参数，消掉打印不完全中间的省略号

# pd.set_option('display.max_columns', 1000)

pd.set_option('display.width', 3000)#加了这一行那表格的一行就不会分段出现了

# pd.set_option('display.max_colwidth', 1000)
# pd.set_option('display.height', 1000)

#显示所有列

pd.set_option('display.max_columns', None)

#显示所有行

pd.set_option('display.max_rows', None)
import xlrd
from datetime import datetime
from xlrd import xldate_as_tuple

workbook=xlrd.open_workbook(r'D:\desktop\01040730kg73\C 题全部数据\附件

2.xlsx')

table =workbook.sheets()[0]     # 通过索引打开

all_content = []
```

```
start=5 #开始的行

end=1200  #结束的行

rows=end-start
for i in range(start,end):
    row_content = []
    for j in range(table.ncols):

        ctype = table.cell(i, j).ctype             # 获取单元格返回的数据类型

        cell_value = table.cell(i, j).value         # 获取单元格内容

        if ctype == 2 and cell_value % 1 == 0:      # 是否是数字类型

            cell_value = int(cell_value)

        elif ctype == 3:                            # 是否是日期

            date = datetime(*xldate_as_tuple(cell_value, 0))
            cell_value = date.strftime('%Y/%m/%d %H:%M:%S')

        elif ctype == 4:                            # 是否是布尔类型

            cell_value = True if cell_value == 1 else False
        row_content.append(cell_value)
    all_content.append(row_content)
    print(table.cell(i, 2).value)
```

HotType.py 部分代码

```
table =workbook.sheets()[0]     # 通过索引打开

all_content = []
arr1=[]

start=1 #开始的行

end=4327 #结束的行

rows=end-start
for i in range(start,end):
    row_content = []
    for j in range(table.ncols):

        ctype = table.cell(i, j).ctype             # 获取单元格返回的数据类型

        cell_value = table.cell(i, j).value         # 获取单元格内容

        if ctype == 2 and cell_value % 1 == 0:      # 是否是数字类型
```

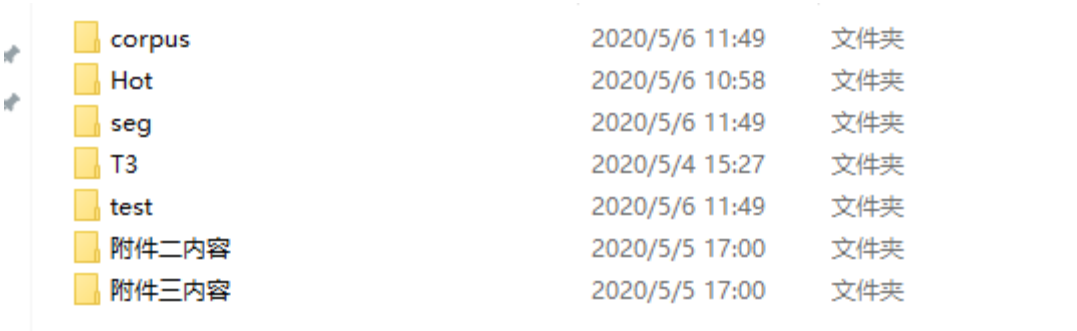
```
            cell_value = int(cell_value)
        elif ctype == 3:                                # 是否是日期
            date = datetime(*xldate_as_tuple(cell_value, 0))
            cell_value = date.strftime('%Y/%m/%d %H:%M:%S')
        elif ctype == 4:                                # 是否是布尔类型
            cell_value = True if cell_value == 1 else False
        row_content.append(cell_value)
```

4.2 样本数据测试结果

根据题目要求训练分类器，即把附件 2 中数据每类按照 7：3 的比例，进行分类器的训练和性能测试。根据 F-Score 方法评价分类器，得到的数据如图。

```
精度:0.866
召回:0.660
f1-score:0.682
```

热点问题留言分类得到了“热点问题表.xls”和“热点问题留言明细表.xls”。并对算法进行多次改善得到最终结果。下图为得出所有数据结果。



5.参赛感想

作为计算机专业的学生，以前也没有接触过数据挖掘这类问题，对机器学习和自然语言处理仅仅停留在知道名字的程度，并未有过深入了解。报名时候忐忑，从开始的无从入手，到后来通过自学 Python 语言和 TF-IDF 权重策略以及对文本挖掘算法的深入了解，慢慢的再对朴素贝叶斯分类器，机器学习，数据挖掘等一系列相关知识有了一些理解，最后对此次赛题进行充分的分析，并且慢慢开始有了解决问题的思路。从对这一方面知识畏惧，然后是感到这一类的困难，再到后来的了解了一些片面，到最后进行问题的解决，这个过程无疑是令人自豪的，做到了走出舒适区，迎接新领域与新知识。这种感觉是比生活在舒适区兴奋很多的，让人更加喜欢参加比赛，而不是畏惧停滞不前，解决未知的问题是十分有趣的过程，但时间一长就容易产生疲倦感，所以时刻保持自己的热度很重要。耐心也很重要，在一次又一次的训练分类器，才能在成功的时候感到快乐。我们的成果和论文也许不是很优秀，但是这

个过程是开心的，这份学习经历也是值得的，我们的团队合作也是让人开心的，一起学习，共同进步才是最重要的事。

致谢

首先，在这里我要感谢我们小组的导师，导师的指导与监督，让我们才能做出最后的成果。其次感谢 CSDN 组织，在 CSDN 中找到了各种大佬的学习笔记与重点内容，很多问题都是在 CSDN 中找到答案,同时还要感谢大赛组委会给我们这么好的机会,让我们能够展示自己，展示能力，并且我们以后还会多多参加这类比赛，来充实自己，我们会继续学习数据挖掘这一方面的内容，对这一方面有更深的学习计划。

最后，感谢我们小组的每一个人的付出，尽管还是有很多不完美之处，但这都是我们辛勤奋斗的结果。

参考文献

[1]康东. 中文文本挖掘基本理论与应用[D].苏州大学,2014.
[2]彭俊杰. 中文短文本表示及分类的研究与实现[D].河南大学,2012.
[3]尹凯. 基于深度学习的网络新闻文本分类研究[D].山西财经大学,2019.
[4]姚奇峰,杨连贺.数据挖掘经典分类聚类算法的研究综述[J].现代信息科技,2019,3(24):86-88. [5]李俊. 句子语义相似度计算方法研究及其应用[D].浙江工业大学,2016.
[6]李旭然,丁晓红.机器学习的五大类别及其主要算法综述[J].软件导刊,2019,18(07):4-9.
[7]黄承慧,印鉴,侯昉.一种结合词项语义信息和TF-IDF方法的文本相似度量方法[J].计算机学报,2011,34(05):856-864.