**阜阳师范大学**

**计算机与信息工程学院**

**《****数据分析与挖掘》期末项目**

**（项目名称： 舆论舆情对经济板块影响 ）**

**姓 名： 周子苏**

**学 号： 2019114054**

**班 级：数据科学与大数据技术**

**完成时间： 2021年12月20日**

阜阳师范大学计算机与信息工程学院

**项目主题**

### 摘 要：

### 为分析舆论舆情对经济板块的影响，通过爬取历史新闻，获取历史股票数据进而合成分析处理成板块走势数据。训练建立分类模型将新闻分类，用板块走势和新闻发布时间内容训练建立正负相关模型判断该新闻对板块为积极影响还是消极影响。通过分析发现舆论舆情对经济板块具有一定影响，该程序可实现实时新闻分类和实时新闻正负相关预测。

**关键词：**爬虫、建模、分类、正负相关预测、实时

**内容可参考以下目录：**

## 1 数据来源

网站链接：<https://finance.sina.com.cn/7x24/?tag=10>（新浪财经7\*24小时全球实时财经新闻直播A股）

网站链接：[https://finance.sina.com.cn/7x24/?tag=2](https://finance.sina.com.cn/7x24/?tag=10)（新浪财经7\*24小时全球实时财经新闻直播行业）

网站链接：[https://finance.sina.com.cn/7x24/?tag=5](https://finance.sina.com.cn/7x24/?tag=10)（新浪财经7\*24小时全球实时财经新闻直播市场）

网站链接：[https://finance.sina.com.cn/7x24/?tag=7](https://finance.sina.com.cn/7x24/?tag=10)（新浪财经7\*24小时全球实时财经新闻直播央行）

Tushare大数据金融开放社区

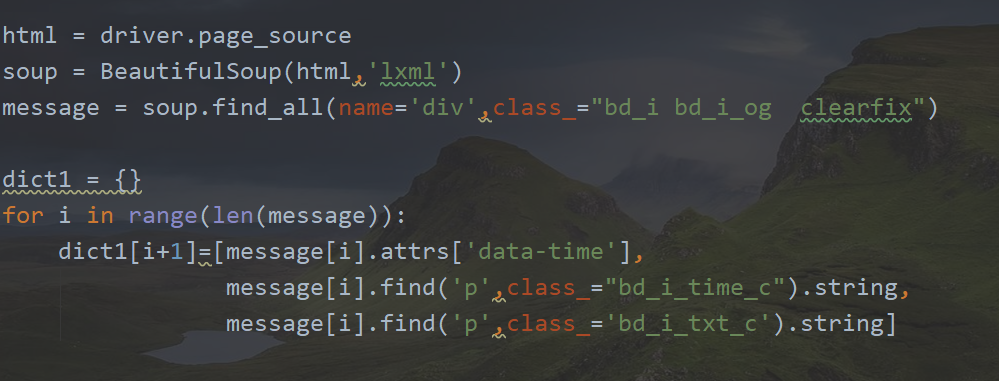
### 背景描述

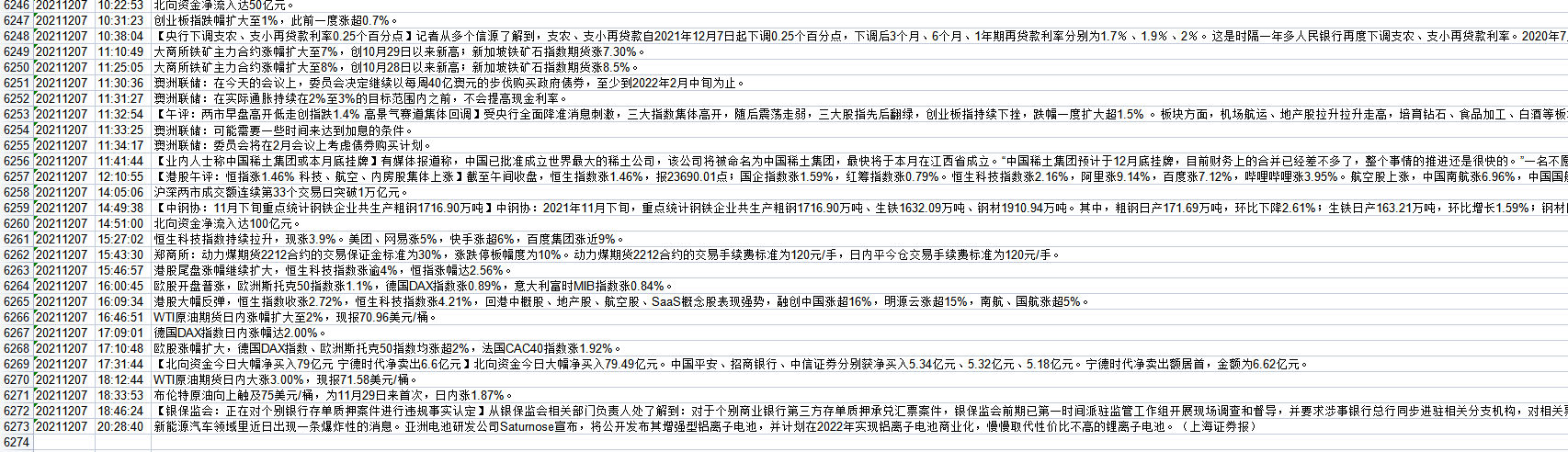
影响股票、板块的因素有很多，我国是社会主义市场经济，股市无法准确预测趋势。但万物皆数据，每次的交易日股市都会产生海量数据（包括涨跌幅，收盘价，成交额等），量化交易的兴起也证实了我们可以对金融进行数据化的分析和预测。其中分析舆论舆情对经济影响的项目论文作品很少，每年在知网上可查询的大概有两三百篇。本项目是为了证实“舆论舆情对经济板块具有影响”论点并探讨是否可以量化舆论舆情对经济的影响。

## 2 数据预处理

### 2.1数据爬取过程

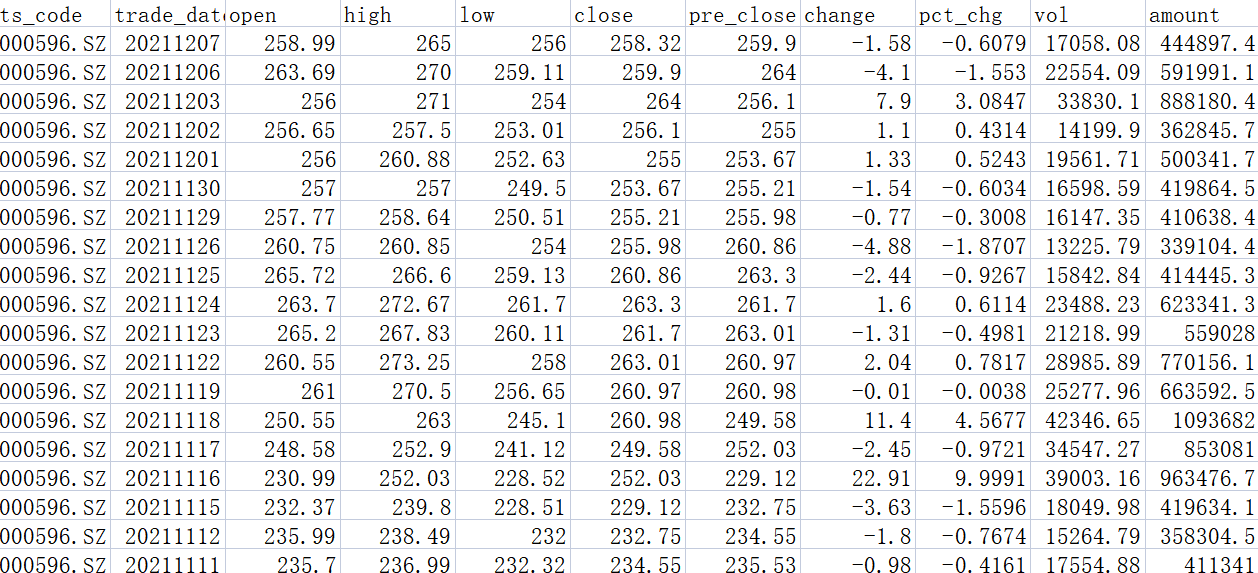
在爬取新浪财经24\*7新闻中发现它的数据是倒序的，也就是说它的第一条是最近最新的新闻，这个网页需要我们不断的下滑下滑才能看到历史新闻，可能我们只需要分析每次网页动态变化后所发生变化JSON信息和get-post传递的信息等寻找规律，这样可以更好的爬取，在花费一两个小时发现这个可能没有太多可用的信息，然后我采用了selenium自动化测试的第三方库，通过无界面的模拟浏览器让网页自己自动下滑下拉直到出现我们设置的时间节点后停止，接着分析出网页当前的HTML结构（所需信息被存放在JSON格式中），通过BS4精准定位提取一条新闻的日期、时间和内容，并使用xlwt三方库将这三类数据保证在excel中。这里我们没有用到request，正则re，虽然re正则匹配是非常高效的，但是当被匹配的信息很多很多时，re就显得不那么强大了，我这里采用的方法更高效一点！





通过Tushare大数据免费金融社区，使用Tushare第三方库直接获取白酒板块“贵州茅台 600519.SH”、“山西汾酒 600809.SH”、“泸州老窖 000568.SZ”、“五粮液 000858.SZ”、“洋河股份 002304.SZ”、“酒鬼酒 000799.SZ”、“古井贡酒 000596.SZ”、“水井坊 600779.SH”、“今世缘 603369.SH”、“迎驾贡酒 603198.SH”；房地产开发板块“保利发展 600048.SH”、“金地集团 600383.SH”、“新地控股 601155.SH”、“绿地控股 600606.SH”、“万业企业 600641.SH”、“新湖中宝 600208.SH”、“西藏城投 600773.SH”、“上海临港 600848.SH”、“陆家嘴 600663.SH”、“中国建筑 601668.SH”;银行板块“招商银行 600036.SH”、“兴业银行 601166.SH”、“平安银行 000001.SZ”、“工商银行 601398.SH”、“宁波银行 002142.SZ”、“交通银行 601328.SH”、“浦发银行 600000.SH”、“农业银行 601288.SH”、“民生银行 600016.SH”、“上海银行 601229.SH”历史数据信息，并保存在CSV文件中。





### 2.2数据清理

### Tushare得来的数据没有需要处理的地方，只要把数据读取与顺序给改变一下就行了。这里可以直接采用excel的排序就可以了或者用代码列表逆转一下[::-1]，个人认为在处理实际问题的时候并不用强求于必须用哪种方法，最方便最高效的就是最好的！新闻数据需要处理的地方比较多，比如说我们是对白酒、房地产开发、银行三个板块进行分析，所以我们需要从6272条新闻中提取可以用来分析的新闻数据。那么如何分别新闻内容呢？这里就涉及到了分类建模这一部分，我们放在后面解释说明！



### 2.3数据集成

## 我们需要对各个股票进行集成然后形成一个板块的数据，我们还需要对提取出来的新闻汇总形成一个可训练可测试的数据集。以上工作为数据分析挖掘建模做准备！

### 2.4 简单的情感分析

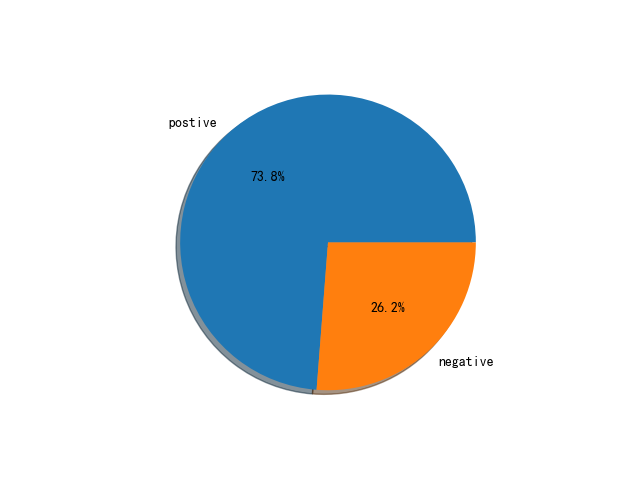
在进行下一个部分工作之前，我们做了一个简单的情感分析：

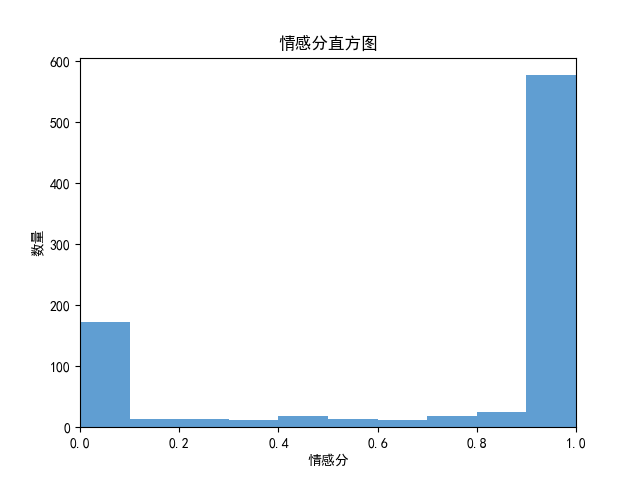
进行情感分析前先说明一点，情感分析中最重要的一点就是语料库，因为情感分析的原理把一段话或者一篇文章切片成一个个的词组或字，然后根据语料库的词语或者某些计算公式去判断这些片段的情感倾向。我这里用的snownlp库如果是用于在线评论的话是比较精准的，但是用于财经新闻的话，就不是那么准确。但是还是可以用的！

data1['emotion'] = data1['contents'].apply(lambda x:SnowNLP(x).sentiments) #核心语句！！！

以上是调用snownlp库的情感分析部分最核心的部分！

我们还绘制了两张由情感分析得出来的图：





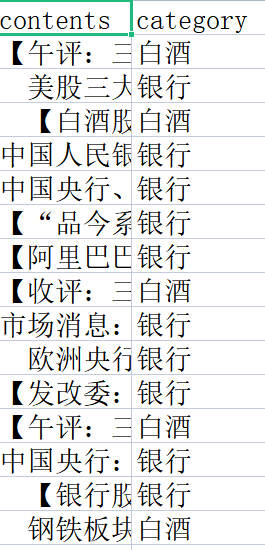
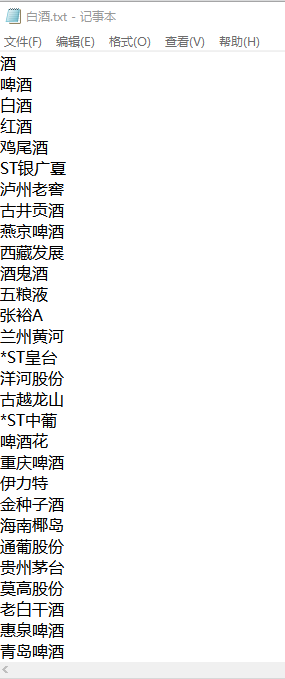
## 3 数据分析

### 3.1数据分析/数据挖掘方法

在建立模型之前我们需要对数据贴标签，这样才能进行训练。无论是新闻分类还是正负相关预测我们选择的都是[朴素贝叶斯分类器](https://blog.csdn.net/cymy001/article/details/79052366?ops_request_misc=%7B%22request%5Fid%22%3A%22164008007016780265420260%22%2C%22scm%22%3A%2220140713.130102334..%22%7D&request_id=164008007016780265420260&biz_id=0&utm_medium=distribute.pc_search_result.none-task-blog-2~all~sobaiduend~default-2-79052366.first_rank_v2_pc_rank_v29&utm_term=MultinomialNB" \t "https://so.csdn.net/so/_blank)模型，在一般 的机器学习中，朴素贝叶斯模型和高斯贝叶斯模型对文本处理效果更加显著。我们先对新闻贴类别标签，编写程序“手动”贴标签。这里我们自己创建使用了三个语料库：白酒.txt、房地产开发.txt、银行.txt。语料库内容主要是与类别相关的名词和股票名词，以白酒.txt为例，其中内容有：酒、啤酒、白酒、红酒、鸡尾酒、ST银广夏、泸州老窖、古井贡酒、五粮液、贵州茅台等（如图一所示）。

经过语料库初步筛选分类后只剩下了三个类别共800多条可利用的新闻，造成这个现象的主要原因是语料库不够丰富无法覆盖到100%且基数新闻不够！这里前期工作没有做好导致后面训练出来的模型分类准确率不高，不能称之为人工智能，应为人工智障！以下为贴类别标签的代码：

def read\_newscontects(path,x,y,z):  
 path\_1 = os.getcwd()[:36]  
 data = pd.DataFrame(pd.read\_csv(path,encoding='gbk'))  
 data = data.dropna()  
 data\_contects = list(data['内容'])  
 list\_test = []  
 for i in range(len(data\_contects)):  
 mark1 = 1  
 for a in x:  
 if a in data\_contects[i]:  
 list\_test.append([data\_contects[i],'白酒'])  
 mark1 = 0  
 break  
 for b in y:  
 if b in data\_contects[i]:  
 list\_test.append([data\_contects[i],'房地产开发'])  
 mark1 = 0  
 break  
 for c in z:  
 if c in data\_contects[i]:  
 list\_test.append([data\_contects[i],'银行'])  
 mark1 = 0  
 break  
 if mark1:  
 list\_test.append([data\_contects[i],'其他'])  
  
 f = open(path\_1+'数据/新闻/标签新闻2.csv','w',encoding='utf-8',newline="")  
 csv\_writer = csv.writer(f)  
 csv\_writer.writerow(["contents", "category"])  
 for i in list\_test:  
 csv\_writer.writerow(i)  
 f.close()  
 data = pd.DataFrame(pd.read\_csv(path\_1+'数据/新闻/标签新闻2.csv',encoding='utf-8'))  
 return data

（图 二） （图 一）

在对新闻内容正负相关贴标签的时候，我想过很多计算公式用来对其进行量化，

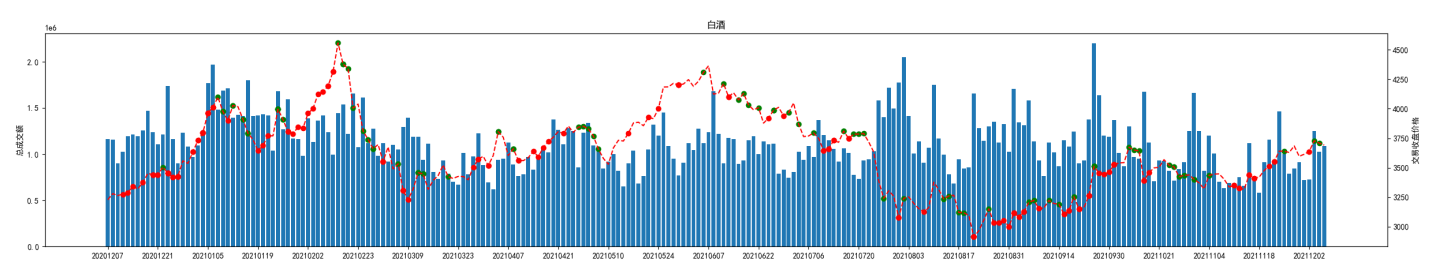
比如说：

方法1—平均斜率法

方法2—分时比例算法

方法3—分日权重法 **

通过比较最终选择了分日权重法，因为它浅显易懂而且效果非常可观！



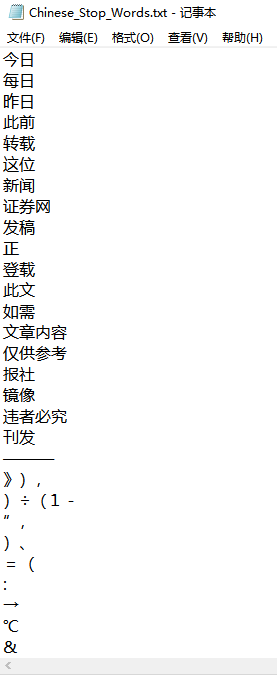
如图所示：红点代表该新闻为positive（积极正相关），绿点代表该新闻为negitive（消极负相关），分日权重法尤其在每个拐点的时候大显神威！以下为贴正负相关标签的代码：

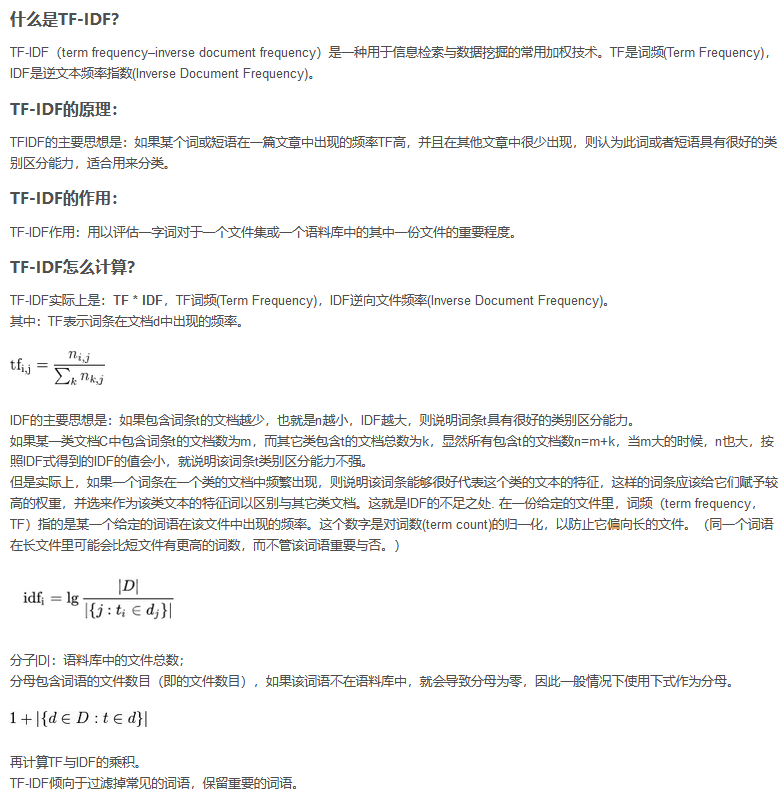
def Weight\_calculation(data,x,y,z):  
 list\_date,list\_category = list(data['date']),list(data['category'])  
 x\_date,x\_price = list(x['日期']),list(x['价格'])  
 y\_date,y\_price = list(y['日期']),list(y['价格'])  
 z\_date,z\_price = list(z['日期']),list(z['价格'])  
 list\_emotion = [0 for i in range(len(list\_date))]  
 for i in range(len(list\_date)):  
 if list\_category[i] == '白酒':  
 try:  
 x\_index = x\_date.index(int(date\_weekday(list\_date[i])))  
 weight = calculation(x\_index, x\_price, x\_date)  
 if weight > 0:  
 list\_emotion[i] = 1  
 elif weight < 0:  
 list\_emotion[i] = -1  
 except:  
 print(str(int(list\_date[i])) + '这个日期找不到！')  
 continue  
  
 elif list\_category[i] == '房地产开发':  
 try:  
 y\_index = y\_date.index(int(date\_weekday(list\_date[i])))  
 weight = calculation(y\_index, y\_price, y\_date)  
 if weight > 0:  
 list\_emotion[i] = 1  
 elif weight < 0:  
 list\_emotion[i] = -1  
 except:  
 print(str(int(list\_date[i])) + '这个日期找不到！')  
 continue  
  
 elif list\_category[i] == '银行':  
 try:  
 z\_index = z\_date.index(int(date\_weekday(list\_date[i])))  
 weight = calculation(z\_index, z\_price, z\_date)  
 if weight > 0:  
 list\_emotion[i] = 1  
 elif weight < 0:  
 list\_emotion[i] = -1  
 except:  
 print(str(int(list\_date[i])) + '这个日期找不到！')  
 continue  
 return list\_emotion  
  
def calculation(index,list\_price,list\_date):  
 if index <= len(list\_date)-5:  
 weight = 0.4\*(float(list\_price[index+1])-float(list\_price[index]))+0.3\*(float(list\_price[index+2])-float(list\_price[index]))+0.2\*(float(list\_price[index+3])-float(list\_price[index]))+0.1\*(float(list\_price[index+4])-float(list\_price[index]))  
 elif index == len(list\_date)-4:  
 weight = 0.4 \* (float(list\_price[index + 1]) - float(list\_price[index])) + 0.3 \* (  
 float(list\_price[index + 2]) - float(list\_price[index])) + 0.2 \* (float(list\_price[index + 3])-float(list\_price[index]))  
 elif index == len(list\_date)-3:  
 weight = 0.4 \* (float(list\_price[index + 1]) - float(list\_price[index])) + 0.3 \* (  
 float(list\_price[index + 2]) - float(list\_price[index]))  
 elif index == len(list\_date)-2:  
 weight = 0.4 \* (float(list\_price[index + 1]) - float(list\_price[index]))  
 else:  
 weight = 0  
 return weight  
  
def date\_weekday(date):  
 date = str(int(date))  
 y = date[:4]  
 m = date[4:6]  
 d = date[6:]  
 x = calendar.weekday(int(y),int(m),int(d))  
 if int(x) == 5:  
 return str(int(date)-1)  
 elif int(x) == 6:  
 return str(int(date)-2)  
 else:  
 return date  
  
def Digitization(path1,path2,path3,data):  
 path\_1 = os.getcwd()[:36]  
 data1 = pd.DataFrame(pd.read\_csv(path1, encoding='gbk'))  
 data2 = pd.DataFrame(pd.read\_csv(path2, encoding='gbk'))  
 data3 = pd.DataFrame(pd.read\_csv(path3, encoding='gbk'))  
 list\_emotion = Weight\_calculation(data,data1,data2,data3)  
 data\_news = pd.read\_csv(path\_1 + '数据/新闻/时间标签新闻.csv', encoding='gbk')  
 data\_news['emotion'] = list\_emotion  
 data\_news.to\_csv(path\_1 + '数据/新闻/时间标签新闻.csv', mode='w', index=False)

标签贴好后我们还需要对新闻内容做两个处理：一是对新闻内容进行分割，将每一个新闻分割成多个由列表储存的词语；二是对词语列表转换为向量格式，模型无法对文字进行分析，所以我们把中文转换成数字，这也是一种量化的过程，然后提供给模型进行训练。接下来我们依次进行解读。

在对新闻内容进行分割时我们需要一个分割语料库，这里我们采用了网上专门对经济文章分割的文本文件：Chinese\_Stop\_Words.txt。该文本内容基本上收录了金融专业中文语法中所有断句的词语、符号标点等（如图所示），进行分割的时候程序还需要一个名为“jieba”第三方库，专门用来分割语句的工具库。Jieba库是我们在进行文本分析时经常使用到的库，很好用很强大！尤其是在进行词频统计、绘制词云的时候！以下是新闻内容分割的代码：

def separate\_words(data):  
 content = list(data['contents']) # 将文本内容转换为list格式  
 # 读入停用词表  
 x = os.getcwd()[:36]  
 stopwords = pd.read\_csv(x+'Chinese\_Stop\_Words.txt', index\_col=False, sep="\t", quoting=3, names=['stopword'],encoding='GBK') # list  
 stopwords = list(stopwords['stopword'])  
 print("正在分词,请耐心等候......")  
 contents\_clean = []  
 all\_words = []  
  
 for line in content:  
 current\_segment = jieba.lcut(line) # jieba分词  
 current\_segment = [x.strip() for x in current\_segment if x.strip() != ''] # 去掉分词后出现的大量空字符串  
 if len(current\_segment) > 1 and current\_segment != "\r\n":  
 line\_clean = []  
 for word in current\_segment:  
 if word in stopwords:  
 continue  
 line\_clean.append(word)  
 all\_words.append(str(word))  
 contents\_clean.append(line\_clean)  
 print('------------分词完成-----------')  
 return contents\_clean, all\_words



在对词语列表转换为向量格式时，我们使用的是TfidfVectorizer函数，它是机器学习sklearn.feature\_extraction.text中的专门用于文本转向量数据化的方法。其原理是：TfidfVectorizer分别是由两种方法 CountVectorizer 与 TfidfTransformer 的结合，其处理文本语言的主要中心思想也就是 TF-IDF (词频-逆文档频率)，TfidfVectorizer 的使用相当于先调用了 CountVectorizer 方法，然后再调用 TfidfTransformer 方法。

转为向量格式代码如下：

def format\_transform(x): #x是数据集（训练集或者测试集）  
 words =[]  
 for line\_index in range(len(x)):  
 try:  
 words.append(" ".join(x[line\_index]))  
 except:  
 print("数据格式有问题")  
 return words

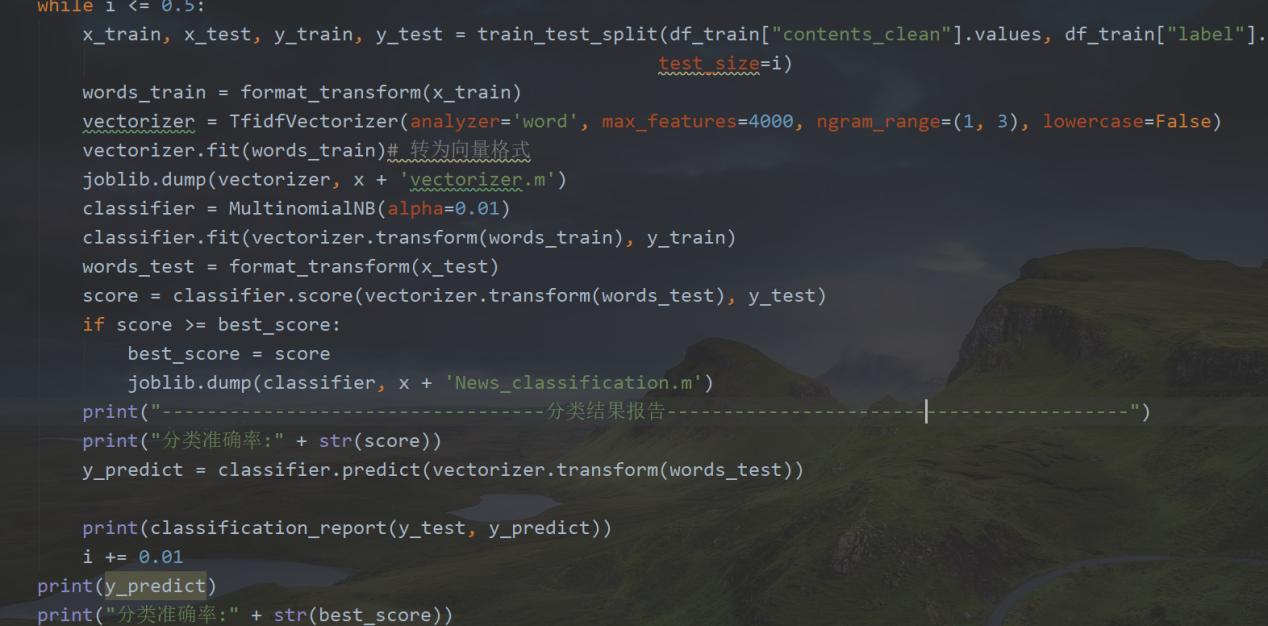
words\_train = format\_transform(x\_train)  
vectorizer = TfidfVectorizer(analyzer='word', max\_features=4000, ngram\_range=(1, 3), lowercase=False)  
vectorizer.fit(words\_train)# 转为向量格式  
joblib.dump(vectorizer, x + 'vectorizer.m')

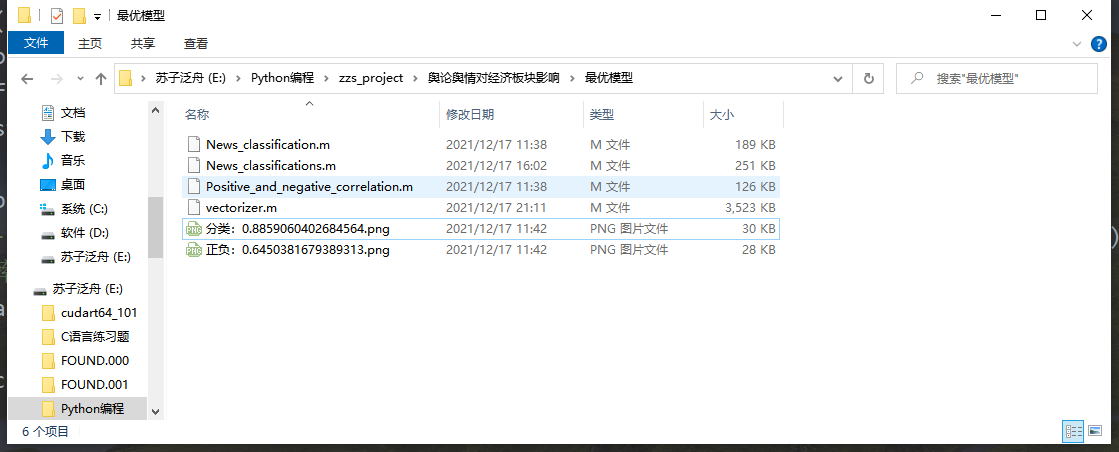
标签贴好后我们就开始了对两个模型进行训练测试跑分，然后保存模型。这里我们需要对贝叶斯原理再说明一下：首先，要明白贝叶斯统计方式与统计学中的频率概念是不同，从频率的角度出发，即假定数据遵循某种分布，我们的目标是确定该分布的几个参数，在某个固定的环境一下做模型。而贝叶斯则是根据实际的推理方式来建模。我们拿到的数据，来更新模型对某事件即将发生的可能性的预测结果。在贝叶斯统计学中，我们使用数据来描述模型，而不是使用模型来描述数据。

贝叶斯定理旨在计算P(A|B)的值，也就是在已知B发生的条件下，A发生的概率是多少。大多数情况下，B是被观察事件，比如“昨天下雨了”，A为预测结果“今天会下雨”。对数据挖掘来说，B通常是观察样本个体，A为被预测个体所属类别。所以，说简单一点，贝叶斯就是计算的是：B是A类别的概率。

IMG_256

训练测试保存模型代码如下：





### 3.2结果展示

在最后的the\_end.py中我们实现了实时监控爬取新浪财经7\*24小时全球实时财经新闻，并在得到最新的新闻后对其进行分类和正负相关的预测！这里我们插入一段程序运行的视频来进行结果展示。



存在的问题：目前存在最大的问题就是数据量少，准确率低和类别设置太少了,导致整个程序并没有到达预期的效果！

## 4 结论

舆论舆情对经济板块存在一定影响，且通过一系列量化操作可对其新闻影响力进行预测分析。

该程序已实现了大致的框架，基本完成了第一阶段的目标和要求。我们还需要很长时间来对其进行增强创新和完善，接下来对该课题和程序的阶段性计划和目标大致说明一下：

第二阶段：爬取更多的历史新闻数据，增加可分类的新闻板块，提高模型准确性。

第三阶段：优化模型运行时间内存，提高效率。增加一个可提供影响等级的正负相关性模型，使舆论舆情对经济板块影响真正做到量化数据化！

第四阶段：打包整理所有程序代码文件并设计实现GUI界面进行交互，完成一个可执行文件工程。

第五阶段：设计实现与金融交易平台对接的api接口，依据程序分析提供出来的舆论舆情影响等级实现股票自动买入卖出，完成量化交易！