# 基于自然语言处理的网民情绪识别报告

# ——以微博推文为例

目录

[基于自然语言处理的网民情绪识别报告 1](#_Toc38995813)

[——以微博推文为例 1](#_Toc38995814)

[一、数据来源和说明 2](#_Toc38995815)

[1.1数据简介 2](#_Toc38995816)

[1.2数据说明 2](#_Toc38995817)

[1.3应用场景 3](#_Toc38995818)

[1.4问题描述 3](#_Toc38995819)

[二、数据分析与实验 3](#_Toc38995820)

[2.1文本分词 3](#_Toc38995821)

[2.2、视觉部分：计算机视觉——图像情感分析 4](#_Toc38995822)

[三、可视化部分 6](#_Toc38995823)

[四、附件部分 14](#_Toc38995824)

[附件1 14](#_Toc38995825)

[附件2 15](#_Toc38995826)

[附件三 20](#_Toc38995827)

说明：

1.数据来源及说明（网站）（余洁）

2.应用场景（描述一下可以应用于文字图片相结合的情感分析）（余洁，闵佳莉）

3.问题描述

4.数据分析与实验— 准确度，

可视化（闵佳莉）

算法—视觉（冯云浩）、

文字（余洁）（原理，流程，数学公式）

5.主要结论等

## 一、数据来源和说明

### 1.1数据简介

数据集来自于DataFountain, 数据集依据与“新冠肺炎”相关的230个主题关键词进行数据采集，抓取了2020年1月1日—2020年2月20日期间共计11万条微博数据，并对其中10万条数据进行人工标注，标注分为三类，分别为：1（积极），0（中性）和-1（消极）。

### 1.2数据说明

数据集以csv格式进行储存，包括nCoV\_100k.labled.csv 和nCov\_10k\_test.csv两个文件组成。

其中：nCoV\_100k.labled.csv：包含10万条用户标注的微博数据，具体格式如下：[微博id,微博发布时间,发布人账号,微博中文内容,微博图片,情感倾向]

微博id，格式为整型。

微博发布时间，格式为xx月xx日 xx:xx。

发布人账号，格式为字符串。

微博中文内容，格式为字符串。

微博图片，格式为url超链接，[]代表不含图片。

情感倾向,取值为{1,0,-1}。

nCov\_10k\_test.csv为1万条未标注微博数据，包含与“新冠肺炎”相关的90万条未标注的微博数据，具体格式如下：

[微博id,微博发布时间,发布人账号,微博中文内容,微博图片,微博视频]

微博id，格式为整型。

微博发布时间，格式为xx月xx日 xx:xx。

发布人账号，格式为字符串。

微博中文内容，格式为字符串。

微博图片，格式为url超链接，[]代表不含图片。

### 1.3应用场景

可以应用于微信朋友圈、QQ空间等网络平台，通过用户发布的文字、图片等数据，判断用户的情感，并对其进行分析，形成相应的认识，产生针对相关问题的解决方案。

### 1.4问题描述

2020年1月份和2月份是新型冠状病毒肺炎的初期，各地纷纷落实新冠疫情一级响应，全国人民纷纷待在家中，不给社会添乱。网民们在微博上就新冠肺炎发表了许多自己的态度和观点，通过大量的微博数据判定网民的情感倾向，分析网民在疫情期间的情绪状态具有重要意义。

## 二、数据分析与实验

### 2.1文本分词

由于中文文本没有天然的分割符，所以需要先将中文分割成空格连接的词语。在python语言中，中文分词一般选用jieba分词器。Jieba是一款优秀的第三方库，主要由三种模式：精确模式、全模式和搜索引擎模式。这里采用的是适合做文本分析的精确模式。

#### 2.1.1停用词处理

停用词是指在信息检索中，为节省存储空间和提高搜索效率，在处理自然语言数据（或文本）之前或之后会自动过滤掉某些字或词，这些字或词即被称为停用词。去除停用词可以大大减少特征词的数量，从而使得文本分类、预测的准确度更高。我们可以通过使用第三方的停用词表来处理停用词。这里我们使用的是哈工大的停用词表。

#### 2.1.2文本特征提取

文本特征提取是将文本数据转化成特征向量的过程。文本特征提取有许多方法，常见的有：词袋模型、TF-IDF模型等。这里采用的是词袋模型。

词袋模型将所有词语装进一个袋子里，不考虑其词法和语序的问题，即每个词语都是独立的，把每一个单词都进行统计，同时计算每个单词出现的次数。

#### 2.1.3文本分类模型

常用的文本分类模型有决策树、最近邻分类器和朴素贝叶斯分类器等。决策树学习采用的是自顶向下的递归方法，其基本思想是以信息熵为度量构造一颗熵值下降最快的树，到叶子节点处，熵值为0。其具有可读性、分类速度快的优点。最近邻分类器找出与测试样本的属性相对接近的k的训练样本，根据它们的类标号判定测试样本的类别。这里采用的朴素贝叶斯分类。

贝叶斯定理

每个记录用一个d维特征向量X = (x1, x2, …, xd)表示

假定有k个类y1, y2, …, yk.

给定X, X属于yj类的后验概率P(yj|X) 满足贝叶斯( Bayes)定理



MAP (maximum posteriori hypothesis, 最大后验假设)

将X指派到具有最大后验概率P(yj|X)的类yj，即将X指派到P(X|yj)P(yj) 最大的类yj

贝叶斯分类工作原理

给定一个未知的数据样本X, 分类法将预测X属于具有最高后验概率的类. 即, 未知的样本分配给类yj, 当且仅当



根据贝叶斯定理, 我们有



由于P(X) 对于所有类为常数, 只需要最大化P(X|yj)P(yj)即可。

贝叶斯分类器的特点:

对孤立的噪声点的鲁棒性，个别点对概率估计的影响很小。

容易处理缺失值，在估计概率时忽略缺失值的训练实例。

对不相关属性的鲁棒性，各类在不相关属性上具有类似分布。

类条件独立假设可能不成立，使用其他技术，如贝叶斯信念网络。

### 2.2、视觉部分：计算机视觉——图像情感分析

#### 2.2.1概述

本部分采用计算机视觉技术，将图片存储矩阵以像素点为单位，从每一个像素点中提取到H（色调）、S（饱和度）、V（亮度）值，将图片通过函数全部转化为５０＊５０的图片。以每张照片的２５００个HSV作为特征值，送入KNN分类器进行学习，最终对测试集进行归类。

#### 2.2.2具体实现

首先，我们知道图像在计算机上的存储是基于矩阵方式的存储，虽然在cv2里有imread（）函数可以直接图片提取每个像素点的RGB参数，但它并不直接与色调、饱和度和亮度等人的主观感觉相对应，所以还需要把RGB参数转换为HSV（色调H、饱和度S、亮度V）。变换方法如下图所示：

变换方法如下：

        max= max ( R , G , B )；

        min= min ( R , G , B )；

        V= max ( R , G , B )；

        S= ( max – min ) / max；

        if( R = max )  H = ( G – B ) / ( max – min) \* 60；

        if( G = max )  H = 120 + ( B – R ) / ( max– min ) \* 60；

        if( B = max )  H = 240 + ( R – G ) / ( max– min ) \* 60；

        if( H < 0 )   H = H + 360；

        其中, R , G , B , S , V ∈[ 1,0 ] , H∈[ 0 , 360 ]

其中, R , G , B , S , V ∈[ 1,0 ] , H∈[ 0 , 360 ]

        在HSV空间中,H从0°到360°变化时,色调依次呈现为红、橙、黄、绿、青、蓝、紫、粉。将H分量根据视觉对颜色的心理感觉,分为不相等间隔的8份,饱和度S分成2份,亮度V分成1份,并根据色彩的不同范围进行量化,当V足够小(V<0. 15时),视觉感知的颜色基本上接近黑色,可以忽略H的影响,仅需一个量化值就可以表示。当S足够小，比如说s小于0.1时，视觉感知的颜色已经基本上接近灰度图像，这时候可以忽略H的影响。量化后的色调、饱和度和亮度值分别为H, S , V：

        当H的值在330– 360以及0 – 25时，H赋值为0，代表色彩粉色；

        当H的值在25– 41时，H赋值为1，代表色彩红色；

       当H的值在41– 75时，H赋值为2，代表色彩橙色；

        当H的值在75– 156时，H赋值为3，代表色彩黄色；

        当H的值在156– 201时，H赋值为4，代表色彩绿色；

        当H的值在201–272时，H赋值为5，代表色彩青色；

        当H的值在272 – 285时，H赋值为6，代表色彩蓝色；

        当H的值在285– 330时，H赋值为7，代表色彩紫色；

        当S的值在0.1– 0.65时，S赋值为0，代表暗色；

        当S的值在0.65– 1时，S赋值为1，代表明色；

        当V的值在0.15– 1时，V赋值为0；

        按照以上的量化级,把3个颜色分量合成为一维特征矢量:

        L= H\*Qs\*Qv+ S\*Qv+ V (1)

        其中:Qs和Qv分别是分量S和V的量化级数,取Qs=2,Qv= 1。因此,式(1)可以表示为:

        L=2H+ S+ V (2)

        这样,量化后的3个分量H、S、V依式(2)合为一个值,根据式(2),L的取值范围为[0,1,… ,15]。

但是，在量化的过程中，我们发现上面所说的方法存在巨大缺陷，即灰度图像是无法提取RGB的，对此，我们采用的方法是跳过这一像素点，将其色调、饱和度、亮度都设为０。接着，通过运用数据爬取技术，我们开始爬取图片，首先构建**ｒｅｓｐｏｎｓｅ**对象，然后使用ＢｙｔｅｓIO进行转化，使ｃｖ２里的Ｉｍａｇｅ函数可以正常代开网页链接，接着开始对图片开展向量化。

再对图片展开向量化完毕后，将其存储到ｃｓｖ文件里，接着再读取ｃｓｖ文件，重新加载数据，划分训练集和数据集，再进行预测。

但是因为HSV数据过于庞大，导致模型学习消耗时间过多，因此为简化起见，我们将HSV值进行每25项一平均，再送入模型学习。

#### 2.2.3、灰度方法

本部分除了像前文一样使用HSV值之外，还准备了一种使用灰度的方法。即先将图片灰度化，再将其转化为数组。接着再根据亮度将其进行二值化，二值化之后再压缩为向量，送入模型进行学习。 其他步骤与上文采用HSV的步骤一样。

#### 2.2.4、不足

针对在本次视觉实验中出现的不足，诸如第一次提取到HSV值后，迫于计算时间过长，因此不得不将数据进行平均。

然而，平均方式过于粗暴，应该使用卷积神经网络对数据进行处理。同时，在送入模型学习时，没有考虑到交叉验证，参数设置不优秀等，希望在以后的学习上，通过不断地学习改进。

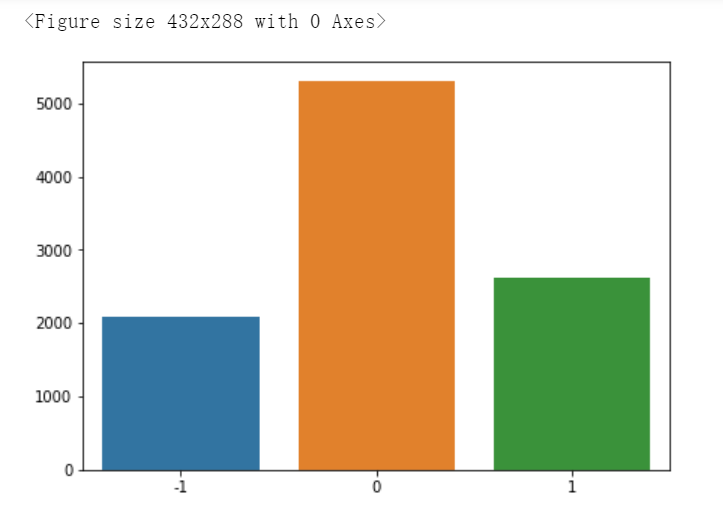
三、可视化部分：

本次探究报告中的可视化部分分为两部分：第一部分是对测试结果letresult.csv进行探究，第二部分是对训练集nCoV\_100k\_train.labled.csv进行探究。

测试结果letresult.csv探究

（1）本部分主要对letresult.csv按照情感进行分类，并统计各类情感类型的微博发布数量，以柱状图的形式进行呈现，便于更直观地观察疫情期间网民情绪总体分布情况，该图表命名为“微博发布数量按情感分类图”。

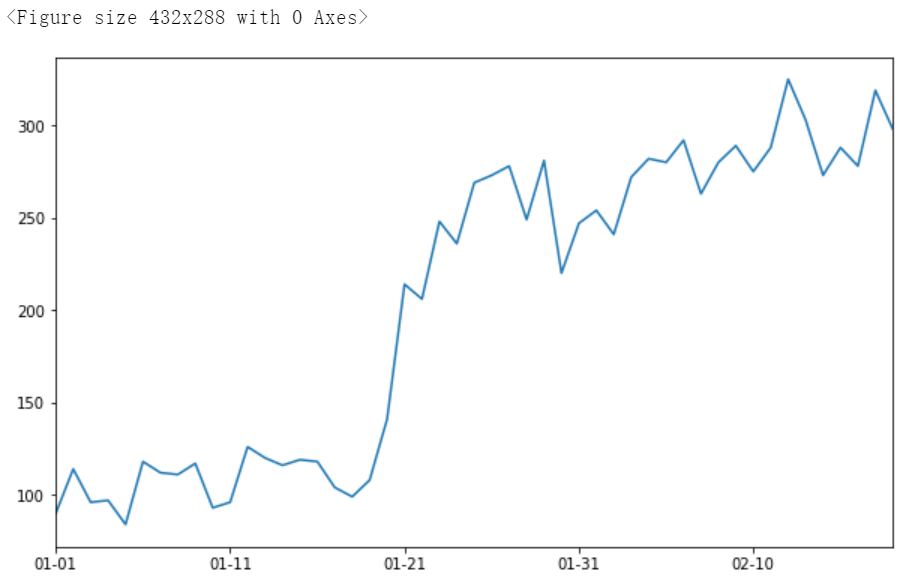
对图表进行观察，负向情感微博推文数量最少，中性情感微博推文数量最多，正向情感微博推文数量居中，但仍极大地少于中性数量，只有其一半左右的数量。



**微博发布数量按情感分类图**

（2）同时，我们将测试结果letresult.csv与测试集nCov\_10k\_test.csv进行合并，生成“合并后数据.csv”，便于统计随时间变化的微博推文数量情况，并以折线图的形式进行更完整的展示，该折线图命名为“微博发布数量随时间变化折线图”。

对该图表进行分析，在1月18日—1月20日左右微博发布数量陡增，与当时网民所处状态有关，正值疫情爆发阶段，大家都居家隔离，有更多的时间与精力进行微博推文等的发送，并在推文中发布体现自己情绪相关的内容。同时，我们也能看到，自微博推文数量陡增开始，后期整体水平都很高，并保持在200以上。

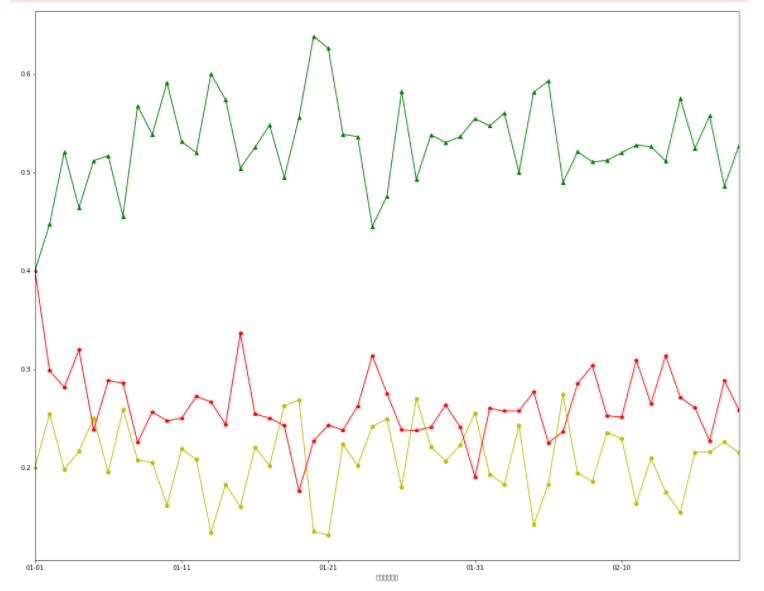


**微博发布数量随时间变化折线图**

（3）根据“合并后数据.csv”文件，我们计算出每日的各类情感推文数量占当日所有推文数量的比例，并绘制出折线图，便于更好地观察每日各类情感的占比变化及其趋势，该图命名为“情感占比随日期变化图”。

其中，黄色线条为负向情感占当日比重变动情况，绿色线条为中性情感占当日比重变动情况，红色线条为正向情感占当日比重变动情况。

整体图表情况体现出：各类情感每日占比随时间有所变化，但总体而言，中性情感比重每日均为最高，其中，1月20日左右占比最高，随时间的变化也会在0.4—0.7之间波动；正向情感多数时间均高于负向情感，1月1日左右占比最高，后面趋于平缓，主要处于0.2—0.3之间；负向情感整体处于最低状态，其中1月21日左右占比最低，整体水平维持在0.2左右。

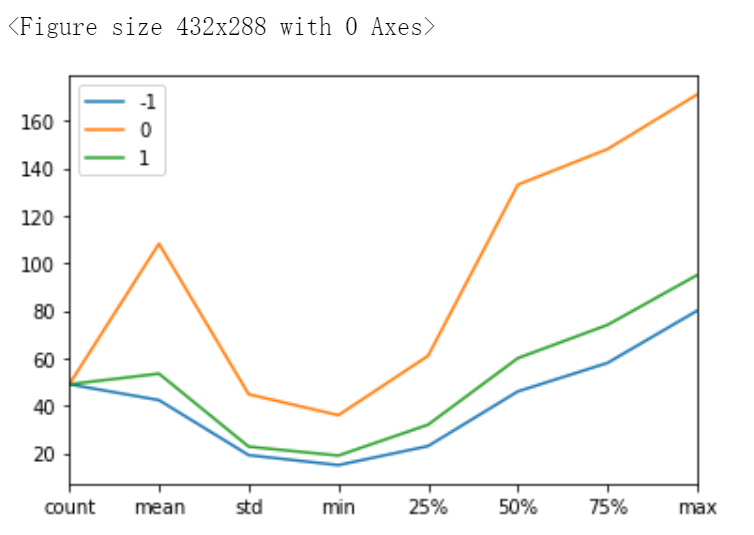


**情感占比随日期变化图**

（4）本部分根据情感极性（-1,0,1）三类进行统计，并得出整体描述，表格统计命名为“整体情况描述”，图片命名为“整体描述图”，如下所示：

|  |  |  |  |
| --- | --- | --- | --- |
| 情感极性 | -1.0 | 0.0 | 1.0 |
| count | 49.000000 | 49.000000 | 49.000000 |
| mean | 42.346939 | 108.204082 | 53.489796 |
| std | 19.177408 | 44.797219 | 22.763387 |
| min | 15.000000 | 36.000000 | 19.000000 |
| 25% | 23.000000 | 61.000000 | 32.000000 |
| 50% | 46.000000 | 133.000000 | 60.000000 |
| 75% | 58.000000 | 148.000000 | 74.000000 |
| max | 80.000000 | 171.000000 | 95.000000 |

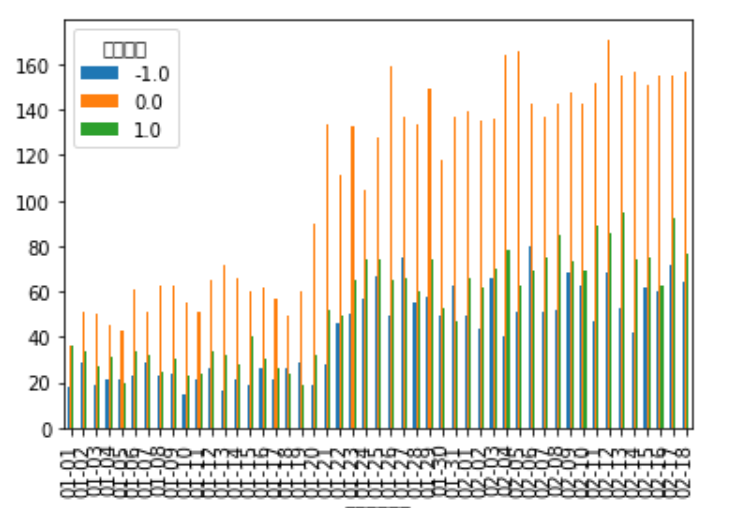
**整体情况描述**



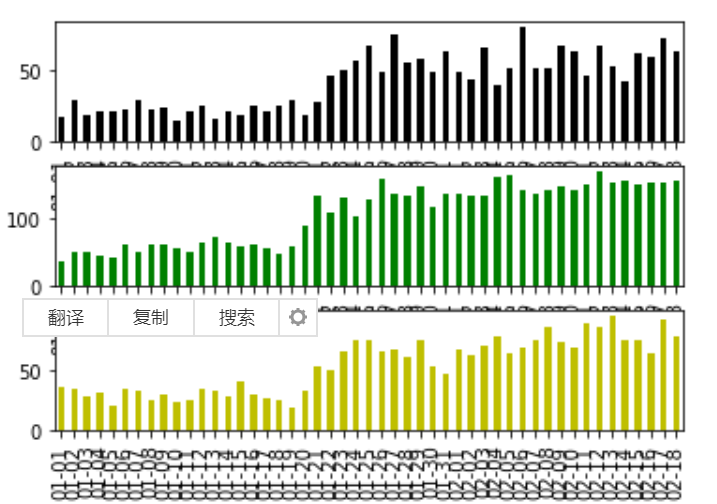
**整体描述图**

（5）我们按照日期统计每天各类情感数量，并制作为三张柱状图，保存为“按日期分类的情感分布柱状图”，同时，我们也提供了按照三种情感类型分别统计的图表，命名为“按日期分别分类的情感分布柱状图”。

1月20日之后，三类情感微博推文每日发布数量与1月20日前相比，是之前的两倍左右，从这个角度来看，各类情感微博推文整体特征相一致。



**按日期分类的情感分布柱状图**



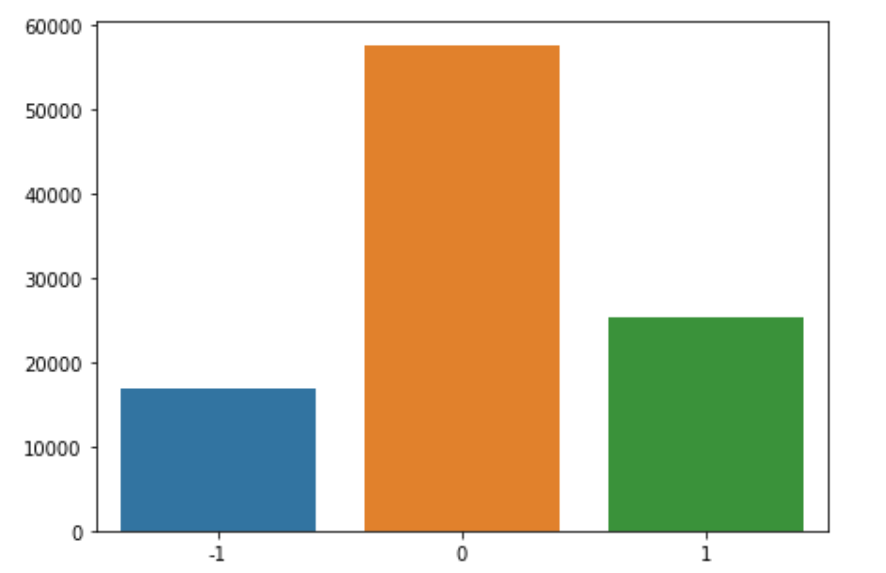
**按日期分类的情感分布柱状图**

训练集nCoV\_100k\_train.labled.csv探究

（1）本部分主要对nCoV\_100k\_train.labled.csv按照情感进行分类，并统计各类情感类型的微博发布数量，以柱状图的形式进行呈现，便于更直观地观察疫情期间网民情绪总体分布情况，该图表命名为“微博发布数量按情感分类图2”。

对图表进行观察，负向情感微博推文数量最少，中性情感微博推文数量最多，正向情感微博推文数量居中，但仍极大地少于中性数量，只有其一半左右的数量。

与第一部分进行比较，两者的分布特征总体上一致。

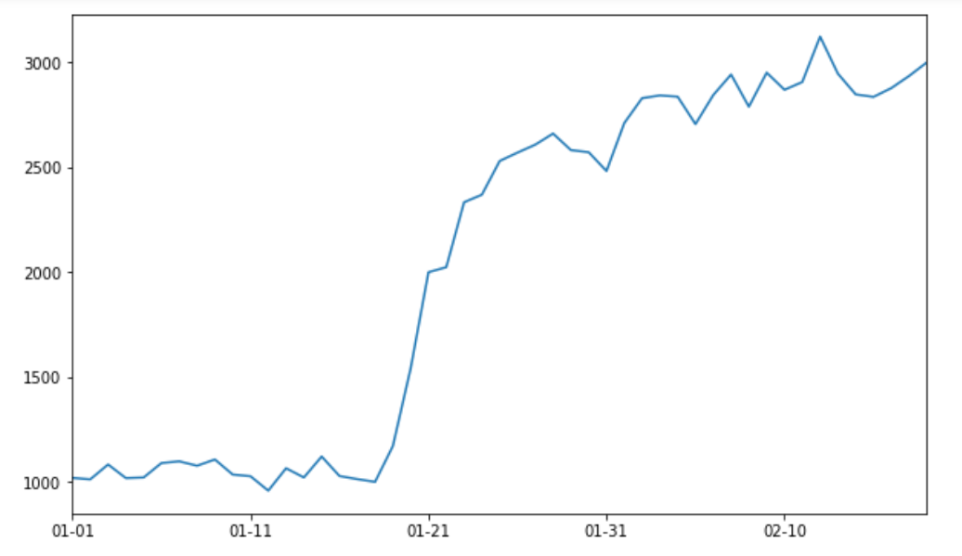


**微博发布数量按情感分类图2**

（2）同时，为了便于统计随时间变化的微博推文数量情况，我们以折线图的形式进行更完整的展示，该折线图命名为“微博发布数量随时间变化折线图2”。

对该图表进行分析，在1月18日—1月20日左右微博发布数量陡增，疫情期间多数网民都居家隔离，有更多的时间与精力进行微博推文等的发送，并在推文中发布体现自己情绪相关的内容。同时，我们也能看到，自微博推文数量陡增开始，后期整体水平都很高，并保持在2500以上。

与第一部分相比较，训练集图表部分相对更平滑，而第一部分图表则相对变化更多，较为崎岖，同时，两者的整体分布曲线特征基本一致。



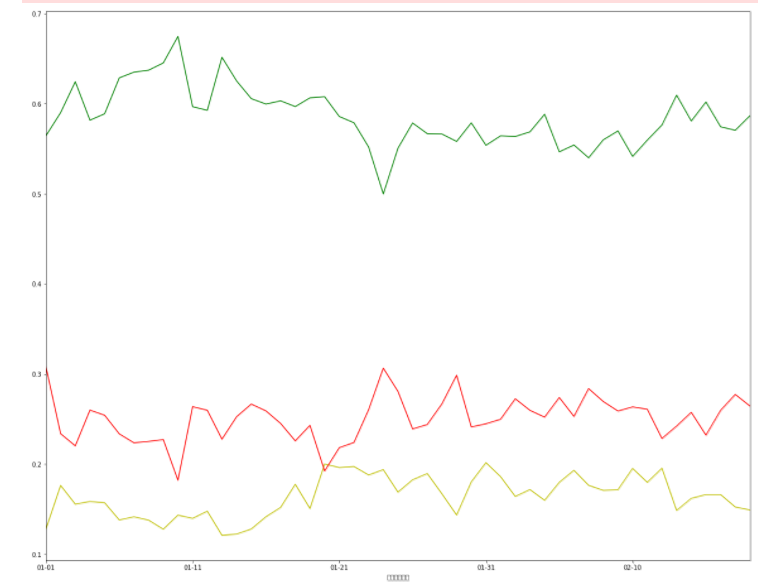
**微博发布数量随时间变化折线图2**

（3）我们根据csv数据计算出每日的各类情感推文数量占当日所有推文数量的比例，并绘制出折线图，便于更好地观察每日各类情感的占比变化及其趋势，该图命名为“情感占比随日期变化图2”。

其中，黄色线条为负向情感占当日比重变动情况，绿色线条为中性情感占当日比重变动情况，红色线条为正向情感占当日比重变动情况。

整体图表情况体现出：各类情感每日占比随时间有所变化，但总体而言，中性情感比重每日均为最高，其中，1月10日左右占比最高，随时间的变化也会在0.5—0.6之间波动；正向情感多数时间均高于负向情感，1月1日左右占比最高，后面趋于平缓，主要处于0.2—0.3之间；负向情感整体处于最低状态，其中1月13日—1月14日左右占比最低，整体水平维持在0.15左右。

与第一部分相比较，两者整体特征相符，但同时，本部分相较于第一部分，各曲线更为平缓，变动幅度相对更小。

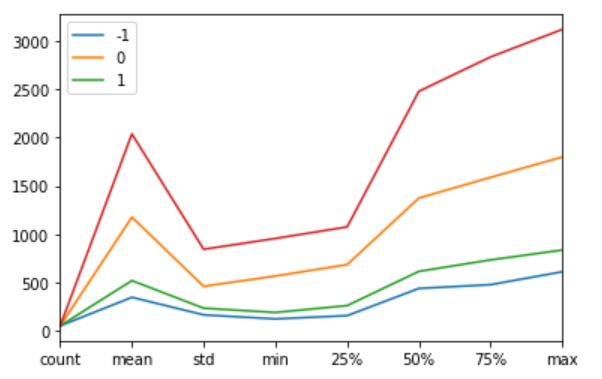


**情感占比随日期变化图2**

（4）本部分根据情感极性（-1,0,1）三类进行统计，并得出整体描述，表格统计命名为“整体情况描述2”，图片命名为“整体描述图2”，如下所示：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 情感倾向 | -1 | 0 | 1 | Col\_sum |
| count | 49.000000 | 49.000000 | 49.000000 | 49.000000 |
| mean | 344.938776 | 1175.877551 | 518.183673 | 2039.000000 |
| std | 162.937796 | 459.100689 | 232.299257 | 843.010256 |
| min | 121.000000 | 566.000000 | 188.000000 | 955.000000 |
| 25% | 155.000000 | 684.000000 | 259.000000 | 1075.000000 |
| 50% | 437.000000 | 1374.000000 | 614.000000 | 2481.000000 |
| 75% | 476.000000 | 1589.000000 | 733.000000 | 2837.000000 |
| max | 610.000000 | 1800.000000 | 835.000000 | 3123.000000 |

**整体情况描述2**

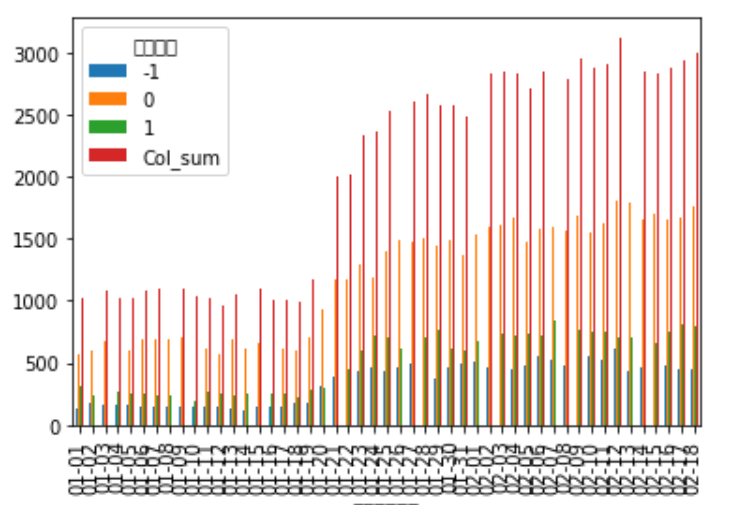


**整体描述图2**

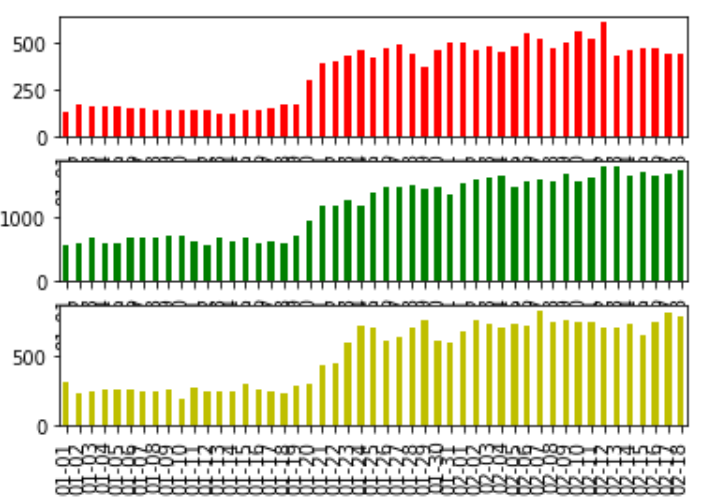
（5）我们按照日期统计每天各类情感数量，并制作为三张柱状图，保存为“按日期分类的情感分布柱状图2”，同时，我们也提供了按照三种情感类型分别统计的图表，命名为“按日期分别分类的情感分布柱状图2”。

1月20日之后，三类情感微博推文每日发布数量与1月20日前相比，是之前的两倍左右，从这个角度来看，各类情感微博推文整体特征相一致。

与第一部分相比较，前后两部分总体变化相符，变化趋势一致。



**按日期分类的情感分布柱状图2**



**按日期分类的情感分布柱状图2**

## 四、附件部分

### 附件1

数据来源：<https://www.datafountain.cn/competitions/423/datasets>

### 附件2

代码部分：

import pandas as pd

import numpy as np

from sklearn.impute import SimpleImputer

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import classification\_report

from sklearn.neighbors import KNeighborsClassifier

from sklearn.tree import DecisionTreeClassifier

from sklearn.naive\_bayes import GaussianNB

from sklearn.feature\_extraction.text import CountVectorizer

import jieba

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from PIL import Image

%matplotlib inline

def chinese\_word\_cut(mytext):

return " ".join(jieba.cut(mytext))

def get\_custom\_stopwords(stop\_words\_file):

with open(stop\_words\_file) as f:

stopwords = f.read()

stopwords\_list = stopwords.split('\n')

custom\_stopwords\_list = [i for i in stopwords\_list]

return custom\_stopwords\_list

a=pd.read\_csv("nCoV\_100k\_train.labled.csv",encoding="ANSI").astype(str)

z=pd.read\_csv("nCov\_10k\_test.csv",encoding="ANSI").astype(str)

test=z[['微博中文内容']]

x = a[['微博中文内容']]

y = a.情感倾向

x['cutted\_comment'] = x.微博中文内容.apply(chinese\_word\_cut)

test['cutted\_comment'] = test.微博中文内容.apply(chinese\_word\_cut)

x\_train, x\_none, y\_train, y\_none = train\_test\_split(x, y, test\_size=1)

stop\_words\_file = "stopwords.txt"

stopwords = get\_custom\_stopwords(stop\_words\_file)

max\_df = 0.8 # 在超过这一比例的文档中出现的关键词（过于平凡），去除掉。

min\_df = 0 # 在低于这一数量的文档中出现的关键词（过于独特），去除掉。

vect = CountVectorizer(max\_df = max\_df,

min\_df = min\_df,

token\_pattern=u'(?u)\\b[^\\d\\W]\\w+\\b',

stop\_words=frozenset(stopwords))

from sklearn.naive\_bayes import MultinomialNB

nb = MultinomialNB()

from sklearn.pipeline import make\_pipeline

pipe = make\_pipeline(vect, nb)

from sklearn.model\_selection import cross\_val\_score#cross\_val\_score

print(cross\_val\_score(pipe, x\_train.cutted\_comment, y\_train, cv=5, scoring='accuracy').mean())

pipe.fit(x\_train.cutted\_comment, y\_train)

result=pipe.predict(test.cutted\_comment)

ID=z['微博id']

rsep=pd.Series(result.tolist())

print(type(rsep))

print(type(ID))

print(rsep.shape)

print(ID.shape)

end = pd.DataFrame(list(zip(ID,rsep)))

print(end)

end.to\_csv('letresult.csv',header=None)

c = pd.read\_csv(r"letresult.csv",header=None,names=["测试数据id","情感极性"])

Data = []

for i in c["情感极性"]:

Data.append(i)

num1 = Data.count(-1)

num2 = Data.count(0)

num3 = Data.count(1)

attr = ["-1","0","1"]

v1 = [num1,num2,num3]

fig = plt.figure()

plt.figure(figsize= (7,5))

sns.barplot(data=c,x=attr,y=v1,ci = None)

plt.show()

plt.savefig("F:/微博发布数量按情感分类图.png")

df=pd.read\_csv('letresult.csv',header=None,names=["测试数据id","情感极性"])

data1=pd.read\_csv('nCov\_10k\_test.csv',encoding = 'ANSI')

df1=data1.join(df)

df1=df1.drop('测试数据id',axis=1)

df1.to\_csv('合并后数据.csv',encoding='utf-8',index=False)

data = pd.read\_csv('合并后数据.csv',encoding='utf-8')

data[u'微博发布时间'] = data[u'微博发布时间'].astype(str)

data[u'微博发布时间'] = data[u'微博发布时间'].apply(lambda x :x[0:2]+"-"+x[3:5])

data.to\_csv('合并后数据.csv',index=False, encoding='utf-8')

t = pd.read\_csv('合并后数据.csv')

time = t.微博发布时间.unique()

time\_counts = t["微博发布时间"].value\_counts()#找出微博发布时间及各个日期发布微博数量

time\_counts.sort\_index(inplace=True)

plt.figure(figsize= (10,6))

time\_counts.plot(kind = "line")###############微博发布数量随时间变化折线图###############

plt.savefig("F:/微博发布数量随时间变化折线图.png")

time\_mood\_groupby = t.groupby(["微博发布时间","情感极性"],sort=True)

grouped\_time\_mood = time\_mood\_groupby.aggregate(np.count\_nonzero).微博id.unstack()

grouped\_time\_mood['Col\_sum'] = grouped\_time\_mood.apply(lambda x: x.sum(), axis=1)

bad = grouped\_time\_mood[-1]/grouped\_time\_mood["Col\_sum"]

mid = grouped\_time\_mood[0]/grouped\_time\_mood["Col\_sum"]

good = grouped\_time\_mood[1]/grouped\_time\_mood["Col\_sum"]

plt.figure(figsize= (20,16))

bad.plot(kind = "line",color = "y",marker='o',)

mid.plot(kind = "line", color = "g",marker='^')

good.plot(kind = "line", color = "r",marker='p')

plt.show()

plt.savefig("F:情感占比随日期变化图.png")#占当日情感的比重

time\_mood\_groupby = t.groupby(["微博发布时间","情感极性"],sort=True)

grouped\_time\_mood = time\_mood\_groupby.aggregate(np.count\_nonzero).微博id.unstack()

b = grouped\_time\_mood.describe()

print(b)

b.plot()

plt.legend(['-1','0','1'],loc="upper left")

plt.show()

plt.savefig("F:/整体描述图.png")

plt.figure(figsize= (20,16))

a = grouped\_time\_mood.plot(kind = "bar")

plt.savefig("F:/按日期分类的情感分布柱状图.png")

plt.figure(figsize= (20,16))

fig, axes = plt. subplots( 3, 1)

a = grouped\_time\_mood[-1].plot(kind = "bar",ax= axes[0],color = "k")

b = grouped\_time\_mood[0].plot(kind = "bar",ax= axes[1],color = "g")

c = grouped\_time\_mood[1].plot(kind = "bar",ax= axes[2],color = "y")

plt.savefig("F:/按日期分别分类的情感分布柱状图.png")

c = pd.read\_csv(r"nCoV\_100k\_train.labled.csv",usecols=['微博id','微博发布时间','情感倾向'],encoding = "ANSI")

Data = []

for i in c["情感倾向"]:

Data.append(i)

num1 = Data.count("-1")

num2 = Data.count("0")

num3 = Data.count("1")

attr = ["-1","0","1"]

v1 = [num1,num2,num3]

fig = plt.figure()

plt.figure(figsize= (7,5))

sns.barplot(data=c,x=attr,y=v1,ci = None)

plt.show()

plt.savefig("F:/微博发布数量按情感分类图2.png")

c[u'微博发布时间'] = c[u'微博发布时间'].astype(str)

c[u'微博发布时间'] = c[u'微博发布时间'].apply(lambda x :x[0:2]+"-"+x[3:5])

time = c.微博发布时间.unique()

time\_counts = c["微博发布时间"].value\_counts()#找出微博发布时间及各个日期发布微博数量

time\_counts.sort\_index(inplace=True)

time\_counts = time\_counts[:-1]

plt.figure(figsize= (10,6))

time\_counts.plot(kind = "line")###############微博发布数量随时间变化折线图###############

plt.savefig("F:/微博发布数量随时间变化折线图2.png")

time\_mood\_groupby = c.groupby(["微博发布时间","情感倾向"],sort=True)

grouped\_time\_mood = time\_mood\_groupby.aggregate(np.count\_nonzero).微博id.unstack()

for i in ['-','-2','10','4','9','·']:

grouped\_time\_mood = grouped\_time\_mood.drop(i,axis = 1)

grouped\_time\_mood = grouped\_time\_mood[:-1]

grouped\_time\_mood['Col\_sum'] = grouped\_time\_mood.apply(lambda x: x.sum(), axis=1)

bad = grouped\_time\_mood["-1"]/grouped\_time\_mood["Col\_sum"]

mid = grouped\_time\_mood["0"]/grouped\_time\_mood["Col\_sum"]

good = grouped\_time\_mood["1"]/grouped\_time\_mood["Col\_sum"]

plt.figure(figsize= (20,16))

bad.plot(kind = "line",color = "y")#,marker='o',)

mid.plot(kind = "line", color = "g")#,marker='^')

good.plot(kind = "line", color = "r")#,marker='p')

plt.show()

plt.savefig("F:情感占比随日期变化图2.png")#占当日总数的比重

#############图名称中2代表按照训练集进行的划分

b = grouped\_time\_mood.describe()

print(b)

b.plot()

plt.legend(['-1','0','1'],loc="upper left")

plt.show()

plt.savefig("F:/整体描述图2.png")

plt.figure(figsize= (20,16))

a = grouped\_time\_mood.plot(kind = "bar")

plt.savefig("F:/按日期分类的情感分布柱状图2.png")

plt.figure(figsize= (20,16))

fig, axes = plt. subplots( 3, 1)

a = grouped\_time\_mood["-1"].plot(kind = "bar",ax= axes[0],color = "r")

b = grouped\_time\_mood["0"].plot(kind = "bar",ax= axes[1],color = "g")

c = grouped\_time\_mood["1"].plot(kind = "bar",ax= axes[2],color = "y")

plt.savefig("F:/按日期分别分类的情感分布柱状图2.png")

### 附件三

#### 灰度方法

**import** requests **as** req  
**from** PIL **import** Image  
**from** io **import** BytesIO  
**import** skimage.io **as** io  
**import** matplotlib.pyplot **as** plt  
  
**from** PIL **import** Image  
**import** pandas **as** pd  
**import** numpy **as** np  
**import** urllib.request  
**import** cv2  
  
  
  
  
**def** getfeel(wangzhi):  
  
 response = urllib.request.urlopen(wangzhi)  
 cat\_img = response.read()  
 **with** open(**"new.png"**, **'wb'**) **as** f:  
 f.write(cat\_img)  
 img = cv2.imread(**'new.png'**)  
 img = Image.open(**"new.png"**)  
 img = img.convert(**"L"**)  
 *# 将图片转换为数组形式，元素为其像素的亮度值* img = img.resize((30, 30))  
 *# 二值化* img = img.point(**lambda** x: 1 **if** x > 120 **else** 0)  
 *# 将图片转换为数组形式，元素为其像素的亮度值* img\_array = np.asarray(img)  
 features\_vector = img\_array.reshape(img\_array.shape[0] \* img\_array.shape[1])  
 **return** features\_vector  
  
  
  
all\_list = []  
each\_list = []  
trade = pd.read\_csv(**"bftest.csv"**)  
pic = trade[**"微博图片"**]  
user\_id = trade[**"微博id"**]  
count = 0  
**for** pic\_http **in** pic:  
 user\_id\_pic = user\_id[count]  
 count = count + 1  
 **if** pic\_http != **"[]"**:  
 pic\_http = pic\_http[1:-1]  
 each\_list.append(user\_id\_pic)  
 each\_list.append(pic\_http.split(**","**))  
 all\_list.append(each\_list)  
 each\_list = []  
result=[]  
i=0  
**for** users **in** all\_list:*#进入每一个ID（行）* linshi=[]  
 pict = users[1]*#进入图片地址单元格* u\_name = str(users[0])  
 print(type(u\_name))  
 i = pict[0]  
 pict\_tip = i.replace(**","**, **""**)  
 pict\_tip = i.replace(**"'"**, **""**) *# 网址* new\_str = **""** file = open(**"test.csv"**, **'a'**)  
 **for** i **in** range(0, len(pict\_tip)):  
 **if** pict\_tip[i] != **'"'**:  
 new\_str = new\_str + pict\_tip[i]  
 **try**:  
 h=getfeel(new\_str)  
  
 a = h.tolist()  
 i = 0  
 **while** i < 900:  
 b = a[i]  
 c = str(b)  
 file.write(c)  
 file.write(**','**)  
 i = i + 1  
 file.write(u\_name)  
 file.write(**'\n'**)  
 **except**:  
 print(**"wrong"**)  
 *#print(h)*

#### HSV

1. **from** PIL **import** Image  
   **import** pandas **as** pd  
   **import** numpy **as** np  
   **import** urllib.request  
   **import** cv2  
   **import** requests **as** req  
   **from** io **import** BytesIO  
     
   **def** rgb2hsv(r, g, b):  
    r, g, b = r/255.0, g/255.0, b/255.0  
    mx = max(r, g, b)  
    mn = min(r, g, b)  
    df = mx-mn  
    **if** mx == mn:  
    h = 0  
    **elif** mx == r:  
    h = (60 \* ((g-b)/df) + 360) % 360  
    **elif** mx == g:  
    h = (60 \* ((b-r)/df) + 120) % 360  
    **elif** mx == b:  
    h = (60 \* ((r-g)/df) + 240) % 360  
    **if** mx == 0:  
    s = 0  
    **else**:  
    s = df/mx  
    v = mx  
    **return** h, s, v  
     
   **def** getfeel(url):  
    response = req.get(url)  
    image = Image.open(BytesIO(response.content))  
    im=image.resize((50,50))  
    imgl = [[0 **for** row **in** range(50)] **for** col **in** range(50)]  
    img\_array=im.load()  
    **for** i **in** range(50):  
    **for** j **in** range (50):  
    **try**:  
    r, g, b = img\_array[i, j]  
    h, s, v = rgb2hsv(r, g, b)  
    **if** h > 300 **and** h <= 360 **or** h > 0 **and** h <= 25:  
    h = 0  
    **elif** h > 25 **and** h <= 41:  
    h = 1  
    **elif** h > 41 **and** h <= 75:  
    h = 2  
    **elif** h > 75 **and** h <= 156:  
    h = 3  
    **elif** h > 156 **and** h <= 201:  
    h = 4  
    **elif** h > 201 **and** h <= 272:  
    h = 5  
    **elif** h > 272 **and** h <= 285:  
    h = 6  
    **elif** h > 285 **and** h <= 330:  
    h = 7  
    **if** s > 0.1 **and** s < 0.65:  
    s = 0  
    **elif** s >= 0.65 **and** s <= 1:  
    s = 1  
    v = 0  
    imgl[i][j] = 2 \* h + s + v  
    **except**:  
    imgl[i][j] =0  
    print(**'gray'**)  
    ff = [[0 **for** row **in** range(50)] **for** col **in** range(2)]  
    **for** he **in** range(0, 50):  
    zong = 0  
    **for** su **in** range(0, 50):  
    zong = zong + imgl[he][su]  
    zong = zong / 50.0  
    ff[0][he] = zong  
    **for** su **in** range(0, 50):  
    heng = 0  
    **for** he **in** range(0, 50):  
    heng = heng + imgl[he][su]  
    heng = heng / 50.0  
    ff[1][su] = heng  
    **return** ff  
     
   all\_list = []  
   each\_list = []  
   trade = pd.read\_csv(**"it.csv"**)  
   pic = trade[**"微博图片"**]  
   user\_id = trade[**"微博id"**]  
   user\_qing = trade[**"情感倾向"**]  
   count = 0  
   **for** pic\_http **in** pic:  
    user\_id\_pic = user\_id[count]  
    user\_qing\_pic = user\_qing[count]  
    count = count + 1  
    **if** pic\_http != **"[]"**:  
    pic\_http = pic\_http[1:-1]  
    each\_list.append(user\_id\_pic)  
    each\_list.append(pic\_http.split(**","**))  
    each\_list.append(user\_qing\_pic)  
    all\_list.append(each\_list)  
    each\_list = []  
   result=[]  
   i=0  
   **for** users **in** all\_list:*#进入每一个ID（行）* linshi=[]  
    pict = users[1]*#进入图片地址单元格* u\_name = users[0]  
    u\_qing = users[2]  
    i = pict[0]  
    pict\_tip = i.replace(**","**, **""**)  
    pict\_tip = i.replace(**"'"**, **""**) *# 网址* new\_str = **""  
    for** i **in** range(0, len(pict\_tip)):  
    **if** pict\_tip[i] != **'"'**:  
    new\_str = new\_str + pict\_tip[i]*#提取网页* **try**:  
    h=getfeel(new\_str)  
    print(h)  
     
    file = open(**"secai.csv"**, **'a'**)  
    a = h  
    i = 0  
    m = 0  
    **for** i **in** range(0,2):  
    **for** m **in** range(0, 50):  
    b = a[i][m]  
    c = str(b)  
    file.write(c)  
    file.write(**','**)  
    m=m+1  
    i = i + 1  
    file.write(u\_qing)  
    file.write(**'\n'**)  
    file.close()  
    **except**:  
    print(**"can not fing picture"**)

#### 3验证预测

**import** pandas **as** pd  
**import** numpy **as** np  
**from** sklearn.impute **import** SimpleImputer  
**from** sklearn.model\_selection **import** train\_test\_split  
**from** sklearn.metrics **import** classification\_report  
**from** sklearn.neighbors **import** KNeighborsClassifier  
**from** sklearn.tree **import** DecisionTreeClassifier  
**from** sklearn.naive\_bayes **import** GaussianNB  
**from** sklearn.feature\_extraction.text **import** CountVectorizer  
  
*#先对模型进行训练*a=pd.read\_csv(**"secai.csv"**)  
  
x=a.iloc[:,:99]  
y=a.情感倾向  
x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.1)  
print(**'Start training knn'**)  
x\_f=x\_train  
y\_f=x\_train.astype(**'int'**)  
dt = DecisionTreeClassifier().fit(x\_f, y\_f)  
print(**'Training done'**)  
y\_pre = dt.predict(x\_test)