

南开大学

Python 机器学习

假新闻检测实验报告

姓名：姜志凯

学号：2011937

学院：网络空间安全学院

专业：信息安全

时间：2021.12.12

一、实验目的：

- 给定一个信息的标题、出处、相关链接以及相关评论，尝试辨别信息真伪
- 输入：信息来源、标题、超链接、评论、真伪标签（0：消息为真；1：消息为假）
- 输出：通过模型构建、训练、测试，得到模型预测的准确率（accuracy）、精确度（precision）、召回率（recall）、ROC 曲线以及 AUC

二、实验原理：

- 数据获取
<https://github.com/yaqingwang/WeFEND-AAAI20>
随本文件同 URL 提供
只使用有标签数据
- 数据读取
文件格式为 csv 格式
可以使用 Python 自带的文件读取方式，手动分列
可以使用 Pandas 库进行 csv 文件读取
文件读取代码可以参考上文提及的 git 仓库中代码
- 读取代码

```
with open(filename, 'r', encoding='utf') as f:  
    Import pandas as pd; dataset = pd.read_csv(filename)
```
- 方法选择
 - 特征工程：目的是最大限度地从原始数据中提取特征以供算法和模型使用
 文本预处理：包括数据清洗、数据分析、特征构建
 特征提取：使用 TF-IDF 进行特征构建
 TF：词频 IDF：逆向文本频率指数
 - 分类器：可选择贝叶斯分类器、支持向量机、随机森林等
 这里选择岭回归分类器
- 评测指标
二分类问题，指标包括：Accuracy、Precision、Recall、F-beta score、ROC、AUC

• 混淆矩阵

		Predicted condition	
		Positive (PP)	Negative (PN)
Actual condition	Total population = P + N		
	Positive (P)	True positive (TP), hit	False negative (FN), type II error, miss, underestimation
	Negative (N)	False positive (FP), type I error, false alarm, overestimation	True negative (TN), correct rejection

• 准确率、精确度、召回率

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{Precision} = \frac{tp}{tp + fp}$$

$$\text{Recall} = \frac{tp}{tp + fn}$$

• ROC曲线与

- $TPR = TP / (TP + FN)$
- $FPR = FP / (FP + TN)$
- 修改二分类的阈值，得到若干个(FPR, TPR)散点。散点组成的曲线为ROC曲线。

• AUC

- AUC为ROC曲线下的面积

三、实验内容：

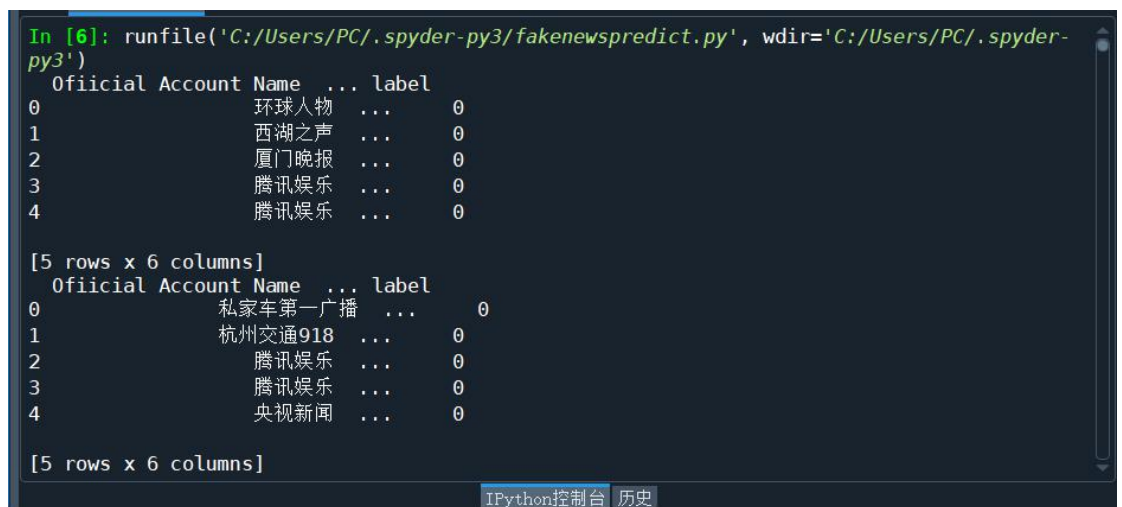
实验工具：本地 Spyder

➤ 数据导入

1、用 pandas 的 read_csv 函数读训练集和测试集，再将训练集和测试集中的 title、Ofiicial Account Name、Report Content 分别合并（求 tf-idf 用），分别得到 title、Ofiicial Account Name、Report Content 列

核心代码：

```
import pandas as pd
train_dataset=pd.read_csv(r'C:\Users\PC\Desktop\train.csv')
test_dataset=pd.read_csv(r'C:\Users\PC\Desktop\test.csv')
#数据合并
alltitle=list(pd.concat([train_dataset['Title'],test_dataset[
'Title']],axis=0))
allofiicialAccountName=list(pd.concat([train_dataset['Ofiicial
Account Name'],test_dataset['Ofiicial Account Name']],axis=0))
allcontent=list(pd.concat([train_dataset['Report
Content'],test_dataset['Report Content']],axis=0))
```



```
In [6]: runfile('C:/Users/PC/.spyder-py3/fakenewspredict.py', wdir='C:/Users/PC/.spyder-
py3')
Official Account Name ... label
0      环球人物      ...      0
1      西湖之声      ...      0
2      厦门晚报      ...      0
3      腾讯娱乐      ...      0
4      腾讯娱乐      ...      0

[5 rows x 6 columns]
Official Account Name ... label
0      私家车第一广播 ...      0
1      杭州交通918    ...      0
2      腾讯娱乐      ...      0
3      腾讯娱乐      ...      0
4      央视新闻      ...      0

[5 rows x 6 columns]
```

train_dataset 和 test_dataset 的前 5 行

2、考虑到只使用 title、Ofiicial Account Name、Report Content 会不会不够全面，我又根据提供的新闻网址爬虫，将新闻内容爬了下来

核心代码:

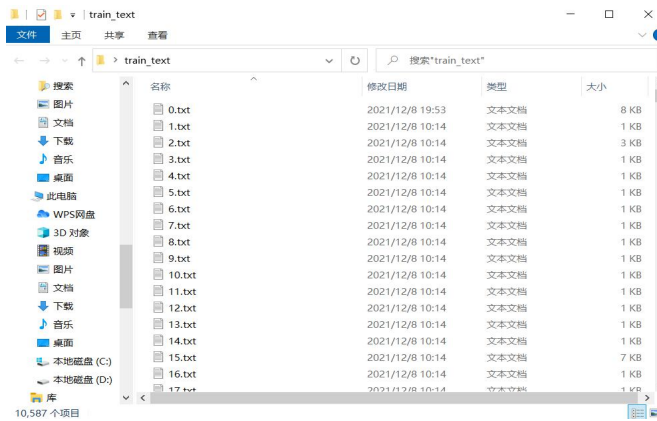
```
#爬虫得到所有新闻 text
import urllib.request
import random
from csv import reader
from bs4 import BeautifulSoup
from ua_info import ua_list

def request_html(url):
    headers={'User _Agent':'Mozilla/5.0 (Windows NT 10.0;
Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
67.0.3396.99 Safari/537.36'}
    request=urllib.request.Request(url,headers=headers)
    return request

def parse_html(html,f):
    soup=BeautifulSoup(html,'html.parser')
    line_name=soup.select('.rich_media_content > p > span'
)
    if(len(line_name)!=0):
        for item in line_name:
            temp =item.text
            f.write('\n'+temp)

def run():
    for i in range(10141):
        f=open('C:\\Users\\PC\\Desktop\\test_text'+str(i)+
'.txt','w',encoding='utf-8')
        url=test_dataset['News Url'][i]
        request=request_html(url)
        html=urllib.request.urlopen(request).read().decode
('utf-8')
        parse_html(html,f)
        f.close()

run()
```



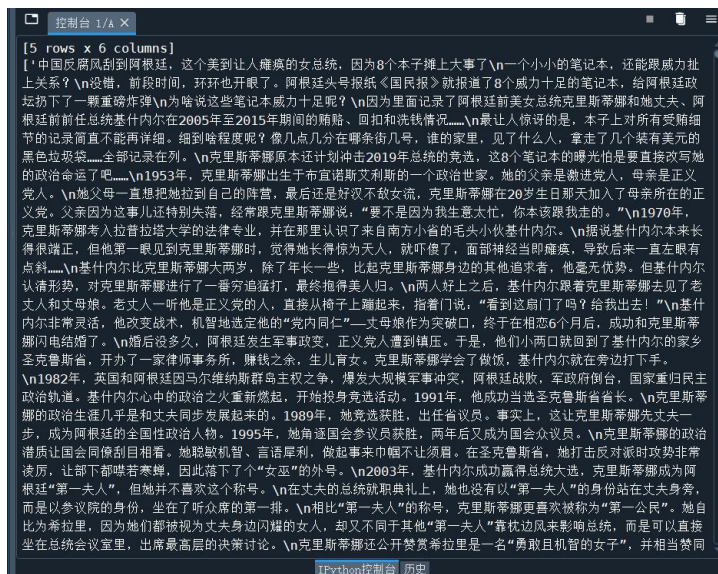
3、读爬下来的文件内容并将 train、test 合并

核心代码：

```
path='C:\\Users\\PC\\Desktop\\'
train_text=[]
for i in range(10587):

    temp=open(path+'train_text\\'+str(i)+'.txt','r',encoding='utf-8',errors='ignore')
        text=temp.read()
        train_text.append(text)
test_text=[]
for i in range(10141):

    temp=open(path+'test_text\\'+str(i)+'.txt','r',encoding='utf-8',errors='ignore')
        text=temp.read()
        test_text.append(text)
#将 train 和 test 合到一起
alltext=train_text
alltext.extend(test_text)
```



部分 text

4、将 title、Ofiicial Account Name、Report Content 和 text 合成一个字符串，便于分词，进而计算 tfidf 特征值

核心代码：

```
coll=[]
for i in range(20728):

s=str(all0fiicialAccountName[i])+str(alltitle[i])+str(alltext[i])+
str(allcontent[i])
    coll.append(s)
print(coll)
```

至此，得到了由每一个新闻的 title、Ofiicial Account Name、Report、Content 和 text 合到一起的字符串组成的数组 coll

➤ 中文分词

1、百度找到停用词表，读入程序

核心代码：

```
stop_word=[]
stopwordfile=open('C:\\Users\\PC\\Desktop\\stop_word.txt','r',
encoding='utf-8',errors='ignore')
f=stopwordfile.read()
for word in f:
    stop_word.append(word)
```

2、利用 jieba 对每个新闻进行中文分词，并除去停用词表里的词

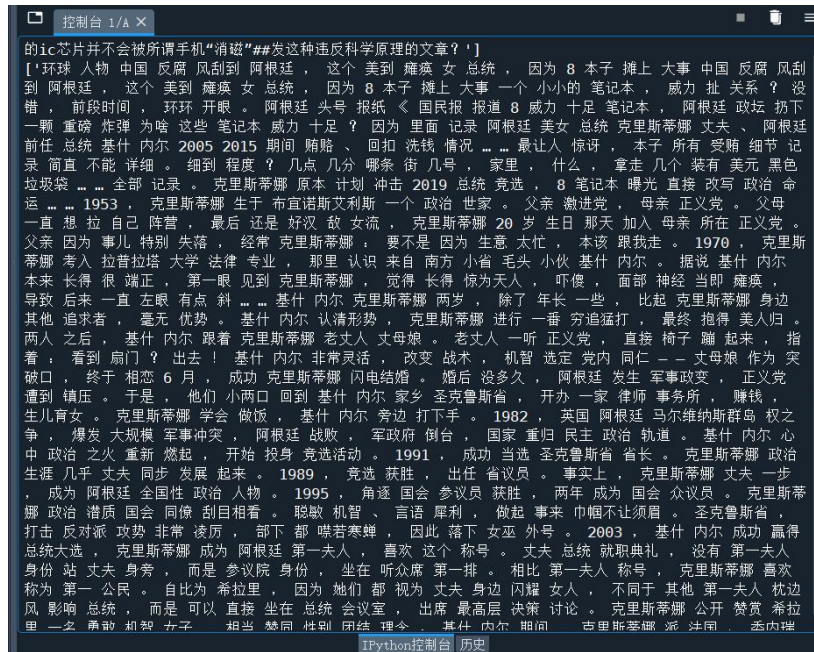
核心代码：

```
import jieba
for text in coll:
    words=jieba.cut(text)
    s=''
```

```

for word in words:
    if word not in stop_word:
        if s!='':
            s=s+' '+word
        else:
            s=word
cutcoll.append(s)

```



分词后的部分新闻

至此，数据处理完毕。

➤ 进行 tf-idf 特征构建，得到文档-词矩阵 features

1、核心代码：（对 TfidfVectorizer 函数进行了参数优化）

```

from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer=TfidfVectorizer(min_df=3,ngram_range=(1,2),use_idf=1,s
mooth_idf=1,sublinear_tf=1)
intifeatures=vectorizer.fit_transform(cutcoll)

```

2、对特征矩阵进行 svd、lda 降维

核心代码：

```

from sklearn.decomposition import TruncatedSVD,
LatentDirichletAllocation
svd=TruncatedSVD(n_components=100,random_state=1400)
lda=LatentDirichletAllocation(n_components=10,random_state=13
37,n_jobs=6)
#svd 降维
svdfeatures=svd.fit_transform(intifeatures)
svdfeatures=pd.DataFrame(svdfeatures)
#lda 降维
ldafeatures=lda.fit_transform(intifeatures)
ldafeatures=pd.DataFrame(ldafeatures)

```



```

#两个降维结果合并
features=[]
features.append(svdfeatures)
features.append(ldafeatures)
features=pd.concat(features,axis=1)
print(features)

```

```

      0      1      2  ...      7      8      9
0    0.038453  0.087315 -0.051132  ...  0.250881  0.002854  0.002858
1    0.007711  0.018381 -0.009517  ...  0.458071  0.020698  0.020698
2    0.030293  0.072302 -0.042293  ...  0.959867  0.004459  0.004459
3    0.016106  0.028197  0.001933  ...  0.224314  0.013882  0.013880
4    0.022846  0.043725 -0.003486  ...  0.858187  0.015757  0.015757
...      ...      ...      ...      ...      ...
20723  0.071030  0.008019 -0.015753  ...  0.259611  0.545744  0.024330
20724  0.012534  0.025555 -0.012258  ...  0.549077  0.023276  0.023276
20725  0.023905  0.044351 -0.020962  ...  0.025774  0.025763  0.025762
20726  0.031802  0.072777 -0.035599  ...  0.019364  0.019360  0.019361
20727  0.014806  0.034716 -0.020295  ...  0.219251  0.013661  0.013661

[20728 rows x 110 columns]

```

降维后的特征矩阵

➤ 岭回归进行训练和预测

1、训练：参数优化：alpha：正则强度，必须是一个正浮点数。正则化改善了问题的条件，并减少了估计的方差。较大的值表示更强的正则化；normalize，归一化。fit_intercept 设置为 False 时，将忽略此参数。如果为 True，则将在回归之前通过减去均值并除以 12-范数来对回归变量 X 进行归一化。

核心代码：

```

#岭回归分类器进行训练和预测
from sklearn.linear_model import RidgeClassifier
clf=RidgeClassifier(alpha=1.0,normalize=True)
clf.fit(features[:train_dataset.shape[0]],train_dataset['label'])

```

2、预测：

核心代码：

```

from sklearn.metrics import precision_recall_fscore_support
from sklearn.metrics import accuracy_score
y_predict=clf.predict(features[10587:])
print(accuracy_score(test_dataset['label'], y_predict))
print(precision_recall_fscore_support(test_dataset['label'],
y_predict,average='binary'))

```

➤ 绘制 ROC 曲线

1、数据准备：

核心代码：

```

#ROC 曲线数据
from sklearn.metrics import roc_curve, auc
fpr, tpr, threshold=roc_curve(test_dataset['label'],
y_predict,pos_label=0)
roc_auc=auc(fpr, tpr)

```


2、绘制曲线：

核心代码：

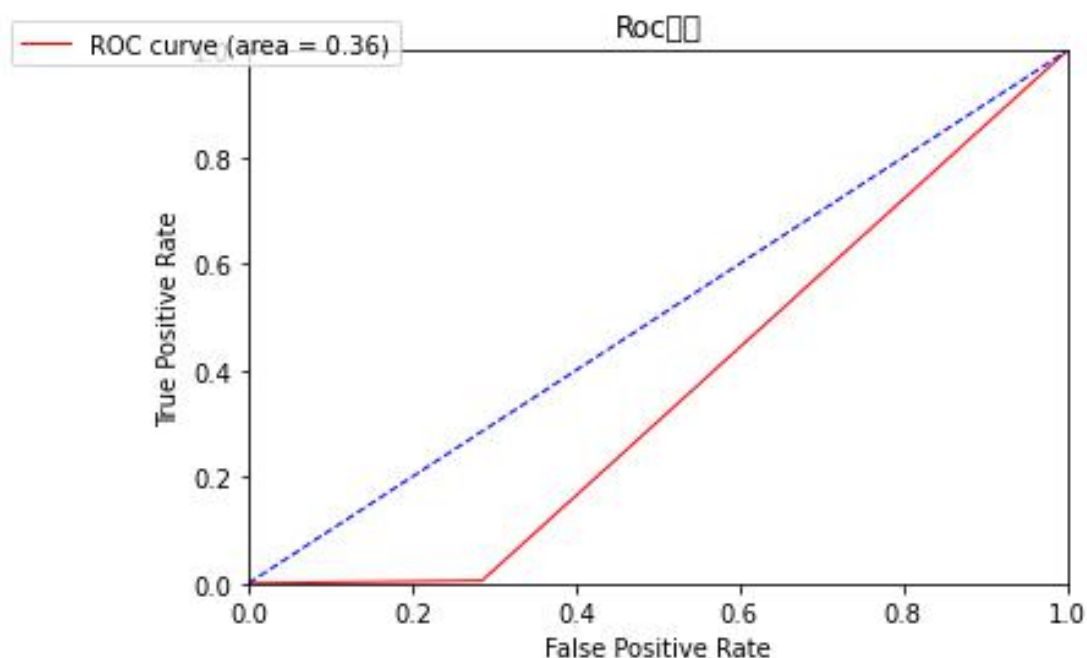
```
#绘制曲线
import matplotlib.pyplot as plt
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Roc 曲线')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.0])
plt.plot(fpr, tpr, color='r', linestyle='-', linewidth=1.0, label=
'ROC curve (area = %0.2f)' % roc_auc)
plt.plot([0, 1], [0, 1], color='b', linestyle='--',
linewidth=1.0)
plt.legend(loc='lower right', bbox_to_anchor=(0.2, 0.95))
plt.show()
```

四、实验结果：

Accruacy、Precision、Recall、F-beta score、support 分别为

```
[20728 rows x 110 columns]
0.8918252637806923
(0.9175704989154013, 0.2854251012145749, 0.4354091610910962, None)
```

ROC 及 AUC



五、实验分析及感悟：

特征构建时，我一并加入了 Report Content 进行特征构建，但事实上，评论均为消极，对模型的准确性并无提高，反而可能会降低。也正因如此，没有考虑情感分析；而且有的新闻已经删除，找不到原文，有的原文和标题一样，这些都会

影响训练的模型的性能。

建议不要用本地编译器，很费内存，CPU 很难受，运行慢，容易卡顿，建议用 anaconda 或飞桨，有网页版的

这是我第一次做机器学习，从一无所知到有所了解，感觉做的还行吧，网上查了好多资料，哪不会就学哪，一步一步完成了这个实验，学到挺多东西，提升挺多。本来想多用几个分类器，看看不同的效果，还想用其他特征构建一下，看看效果，但都不了了之，还是受限于不会，希望以后能继续学习相关知识，做出更强的机器学习模型。