

**PYTHON实验报告**

**Challenging Project：Fake News Detection**

****

学 院 计算机学院

专 业 计算机科学与技术

学 号 2212410

姓 名 刘俊彤

班 级 0935

1. 问题描述

数据集是中文微信消息，包括微信消息的Official Account Name、Title、News Url、Image Url、Report Content、label。Title是微信消息的标题，label是消息的真假标签（0是real消息，1是fake消息）。训练数据保存在train.news.csv。需要先统计分析训练数据，并根据Title文字训练模型，然后在test.news.csv上测试，结果生成csv文件。

整体思路：将训练数据中的Title、Official Account Name、Report Content信息整合，去停用词，作为x\_train，label作为y\_train。同样处理测试数据，得到x\_test。用x\_train、y\_train训练模型，后用x\_test预测得到y\_test，生成csv文件。

1. 数据处理（附关键代码）
2. pandas读入训练集数据和测试数据，并把训练数据打乱

train\_df=pd.read\_csv('train.news.csv',encoding='utf-8')  
train\_df=train\_df.dropna()#如果有空数据则删除行  
train\_df=shuffle(train\_df)#打乱  
test\_df=pd.read\_csv('test.feature.csv',encoding='utf-8')

1. 将训练数据中的Title、Official Account Namet合并，去停用词

#分词+去除停用词  
# 获取停用词集合  
def get\_stopwords():  
 stopwords = [line.strip() for line in open('stop\_words.txt','r',encoding='utf-8').readline()]  
 return stopwords  
def cutsentences(sentences):  
 stop\_words=get\_stopwords()  
 cutwords=list(jieba.lcut\_for\_search(sentences))#分词  
 lastwords=''  
 for word in cutwords:  
 if word not in stop\_words:  
 lastwords+=word+' '  
 return lastwords  
  
t=pd.DataFrame(train\_df.astype(str))  
train\_df['alldata']=t['Ofiicial Account Name']+t['Title']  
t=pd.DataFrame(train\_df.astype(str))  
train\_df['alldata']=t['alldata'].apply(cutsentences)  
train\_alldata=train\_df['alldata']  
x\_train=train\_alldata  
y\_train=np.asarray(train\_df['label'])

相同处理得到x\_test

1. 向量化

from sklearn.feature\_extraction.text import TfidfVectorizer  
transfer=TfidfVectorizer()  
x\_train=transfer.fit\_transform(x\_train)  
x\_test=transfer.transform(x\_test)

1. 转为密集矩阵

x\_train=x\_train.toarray()  
x\_test=x\_test.toarray()

1. 方法介绍（附关键代码）
2. 高斯贝叶斯模型 AUC：0.7223

from sklearn.naive\_bayes import GaussianNB  
model = GaussianNB()  
model.fit(x\_train, y\_train)  
y\_test=model.predict(x\_test)

1. 伯努利贝叶斯模型 AUC：0.7348

from sklearn.naive\_bayes import BernoulliNB  
clf = BernoulliNB()  
clf.fit(x\_train, y\_train)  
BernoulliNB(alpha=1.0, binarize=0.0, class\_prior=None, fit\_prior=True)  
y\_test=clf.predict(x\_test)

1. 随机森林分类模型 AUC：0.7359

from sklearn.ensemble import RandomForestClassifier  
model = RandomForestClassifier(n\_estimators=10, random\_state=123,n\_jobs=-1)  
model.fit(x\_train, y\_train)  
y\_test=model.predict(x\_test)

1. SVM AUC：0.7573

from sklearn import svm  
clf=svm.LinearSVC(dual=True)  
clf.fit(x\_train,y\_train)  
y\_test=clf.predict(x\_test)

1. XGBoost AUC:0.7016

#XGBoost  
import xgboost as xgb  
model=xgb.XGBClassifier()  
model.fit(x\_train,y\_train)  
y\_test=model.predict(x\_test)

1. 随机森林回归模型

from sklearn.ensemble import RandomForestRegressor  
model = RandomForestRegressor(n\_estimators=30,random\_state=1,n\_jobs=-1,oob\_score=False)  
model.fit(x\_train, y\_train)  
y\_test=model.predict(x\_test)

1. n\_estimators=100 AUC：0.7657
2. n\_estimators=10 AUC：0.7676
3. n\_estimators=30 AUC：0.7638
4. n\_estimators=50 AUC:0.7699

主要使用随机森林回归模型，下面是模型的学习结果：

随机森林模型原理：

随机森林是一种基于集成学习的监督式机器学习算法。其结合了多个相同类型的算法，即多个决策树。

每个决策树都是一个模型，其单独对模型进行训练，每个决策树最终的结果都会有较高的方差，但组合成随机森林时，由于每个树都对训练数据进行了完美训练，组合到一起的随机森林的方差就会大大降低。

具体实现流程是：1.从数据集随机选择n个样本子集，2.基于此样本子集构建决策树，3.选择算法中所需要的颗数，重复1、2步骤。（Bootstrapping算法，即有限的样本中多次重复抽样，从而形成足够数量的新样本），最终每棵树都会有对应的预测y值，随机森林会对每个数据取所有预测结果的平均值来计算最终值。（取均值的过程为Bagging）

关于过拟合问题，查阅资料没有对随机森林回归模型是否会过拟合的标准答案，但在本次项目中，n\_estimator=10-100没有出现明显过拟合情况。

参数分析：

决策树个数：n\_estimators:随机抽样生成的树的颗数，默认为100，值越大模型越复杂

决策树最大深度：max\_depth：即生成树的最大深度，越小树结构越简单，可以一定程度上减小过拟合可能。

最小样本分离数：min\_sample\_split：即至少几个样本生成一个随机数

最小叶子结点样本数：min\_sample\_leaf：在叶子结点处最少的样本数

最大分离特征数：max\_features：生成决策树时，考虑到的特征数目的最大值。当这个参数的值设定为1时，表示在生成决策树时，考虑所有的特征；当这个参数的值设定为一个介于0和1之间的数值时，表示在生成决策树时，会从所有的特征中随机选择一定比例的特征进行考虑

是否进行随机抽样：bootstrap

随机抽样的随机数种子：random\_state

1. 实验结果

最终AUC值最高的是随机森林回归模型，参数为：

n\_estimators=50,

random\_state=1,

n\_jobs=-1,

oob\_score=True

其他参数使用默认值  
且数据处理时只用Title和Ofiicial Account Name，最终AUC为0.8476。

1. 实验心得

通过本次大作业，我初步学习了机器学习，并在解决问题的过程中，在不断查阅资料和与同学交流下，尝试学习了贝叶斯、随机森林、SVM、XGBoost等模型，完成了简单的机器学习任务。但过程以及结果还有很多不足：很多模型学习的很浅显，没有仔细研究各个参数的作用，所以大多数模型的参数都是用的默认参数，如果进一步学习，优化参数，结果可能会更好；另一方面是尝试的模型数量还是比较少，比如bert模型听说跑出来效果很好，但我还没有学习，LSTM模型尝试学习但最后没成功，需要学习的东西还有很多，要继续努力。