**程序报告**

学号：2212998 姓名：胡博浩

1. **问题重述**

垃圾短信 (Spam Messages，SM) 是指未经过用户同意向用户发送不愿接收的商业广告或者不符合法律规范的短信。

大数据时代的到来使得大量个人信息数据得以沉淀和积累，但是庞大的数据量缺乏有效的整理规范。

在面对巨大的短信数据时，为了保证更良好的用户体验，如何从数据中挖掘出更多有意义的信息为人们免受垃圾短信骚扰成为当前亟待解决的问题。

**实验要求：**

1) 任务提供包括数据读取、基础模型、模型训练等基本代码

2) 参赛选手需完成核心模型构建代码，并尽可能将模型调到最佳状态

3) 模型单次推理时间不超过 10 秒

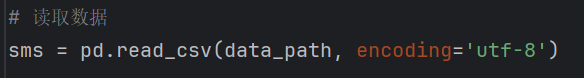
1. **设计思想**

1.设计思路：

本部分从读取数据开始，简单说明代码的逻辑思路，并给出相应优化的理由和方案。

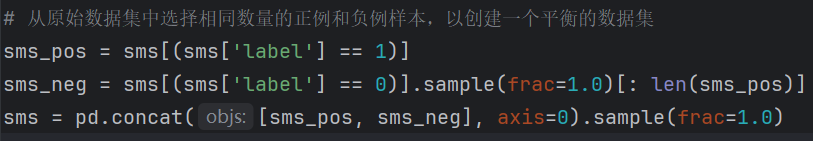
（1）数据读取：

根据 data path 利用 pandas 库读取 csv 格式的数据集，主要使用了其中的 pandas.read\_csv()。



（2）数据平衡化：

从原始数据集中选择相同数量的正例和负例样本，以创建一个平衡的数据集。这种操作解决了类别不平衡的分类问题，可以更好地训练分类模型并提高模型的性能。



（3）停用词表过滤：

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 Stop Words（停用词）。

本实验测试了四种中文常用的停用词表，分别是中文停用词表、哈工大停用词表、百度停用词表和四川大学机器智能实验室停用词库。经过对比分析，发现四川大学停用词表更适合本实验，因此最终选择它作为本实验的停用词表。

（4）文本向量化：

主要有CountVectorizer和TfidfVectorizer两种方法，经过测试，最终采用TfidfVectorizer方法。该方法使用了 TF-IDF 算法，在本质上 IDF 是一种试图抑制噪声的加权，并且单纯地认为文本频率小的单词就越重要，文本频率大的单词就越无用。

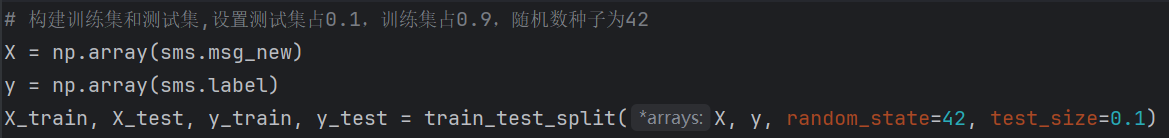
为了充分利用数据，尝试修改TfidfVectorizer中的ngram\_range参数。ngram\_range参数，可以控制特征的粒度和模型对文本的理解程度。经过测试，发现设为（1,2）效果最佳。即考虑单个词和相邻两个词的组合作为特征，进行特征提取。

（5）数据归一化：

主要有StandardScaler和MaxAbsScaler两种方法，目的是将数据标准化，防止因为数量级不同降低模型精度。由于MaxAbsScaler适用于稀疏数据集，契合文本数据的TF-IDF矩阵；且经过测试，MaxAbsScaler效果最佳。因此，最终选择MaxAbsScaler方法。

（6）数据划分：

本实验构建训练集和测试集,设置测试集占0.1，训练集占0.9，随机数种子为42。确保训练集和测试集随机划分。



（7）模型搭建：

经过查阅资料，在尝试众多可能的算法模型之后，发现先验为多项式分布的朴素贝叶斯可能更适合本实验。同时，为了纠正标准多项朴素贝叶斯分类器所做的“严重假设”，采用ComplementNB模型对数据集分类。

（8）构建PipeLine：

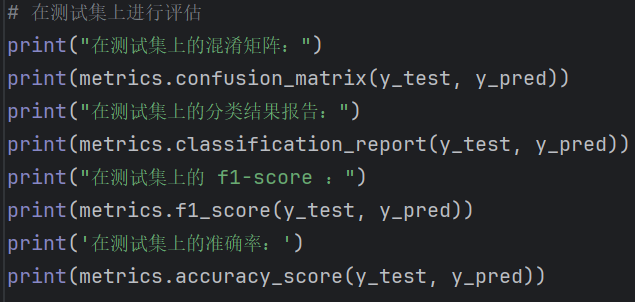
构建PipleLine可以将数据处理和数据分类结合在一起，这样输入原始的数据就可以得到分类的结果，方便直接对原始数据进行预测。



（9）训练测试：

在搭建好的PipLine中调用fit函数先对训练集进行训练，再在测试集观察训练情况。

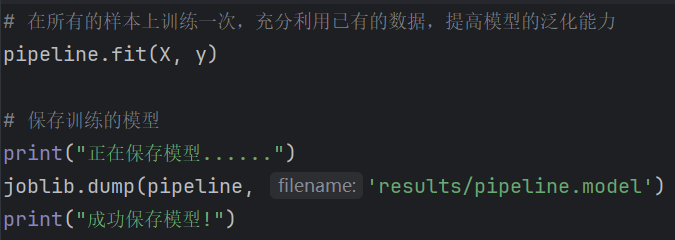
主要评测指标是f1-score和准确率accuracy\_score



（10）模型保存：

为了提高模型的泛化能力，在所有的数据上训练一次，充分利用已有的数据。

然后调用joblib中的dump函数将训练好的最终模型保存到result目录下。



2.局限性：

（1）朴素贝叶斯分类器的局限性：

朴素贝叶斯分类器假设特征之间相互独立，但在实际应用中，特征之间可能存在一定的相关性，这会影响模型性能。

而且，朴素贝叶斯分类器对输入数据的分布假设较为敏感，如果数据分布与其假设不符合，则模型性能可能会受到影响。

（2）TF-IDF方法的局限性：

TF-IDF 方法虽然能够有效地表示文本数据的重要性，但它忽略了词汇之间的顺序信息。对于某些文本分类任务，词汇顺序可能会对分类结果产生影响。

并且，TF-IDF方法对稀有词汇的处理可能不足，因为它主要基于词频和文档频率进行计算，而对于出现次数较少的词汇，其重要性可能被低估。

1. **代码内容**

1. main.py

import os  
  
os.environ["HDF5\_USE\_FILE\_LOCKING"] = "FALSE"  
  
# ---------- 停用词库路径，若有变化请修改 -------------  
stopwords\_path = './datasets/5f9ae242cae5285cd734b91e-momodel/scu\_stopwords.txt'  
  
  
# ---------------------------------------------------  
  
def read\_stopwords(stopwords\_path):  
 *"""  
 读取停用词库  
 :param stopwords\_path: 停用词库的路径  
 :return: 停用词列表，如 ['嘿', '很', '乎', '会', '或']  
 """* stopwords = []  
 # ----------- 请完成读取停用词的代码 ------------  
 with open(stopwords\_path, 'r', encoding='utf-8') as f:  
 stopwords = f.read()  
 stopwords = stopwords.splitlines()  
 # ----------------------------------------------  
  
 return stopwords  
  
  
# 读取停用词  
stopwords = read\_stopwords(stopwords\_path)  
  
# 加载训练好的模型  
import joblib  
  
# ------- pipeline 保存的路径，若有变化请修改 --------  
pipeline\_path = 'results/pipeline.model'  
# --------------------------------------------------  
pipeline = joblib.load(pipeline\_path)  
  
  
def predict(message):  
 *"""  
 预测短信短信的类别和每个类别的概率  
 param: message: 经过jieba分词的短信，如"医生 拿 着 我 的 报告单 说 ： 幸亏 你 来 的 早 啊"  
 return: label: 整数类型，短信的类别，0 代表正常，1 代表恶意  
 proba: 列表类型，短信属于每个类别的概率，如[0.3, 0.7]，认为短信属于 0 的概率为 0.3，属于 1 的概率为 0.7  
 """* label = pipeline.predict([message])[0]  
 proba = list(pipeline.predict\_proba([message])[0])  
  
 return label, proba

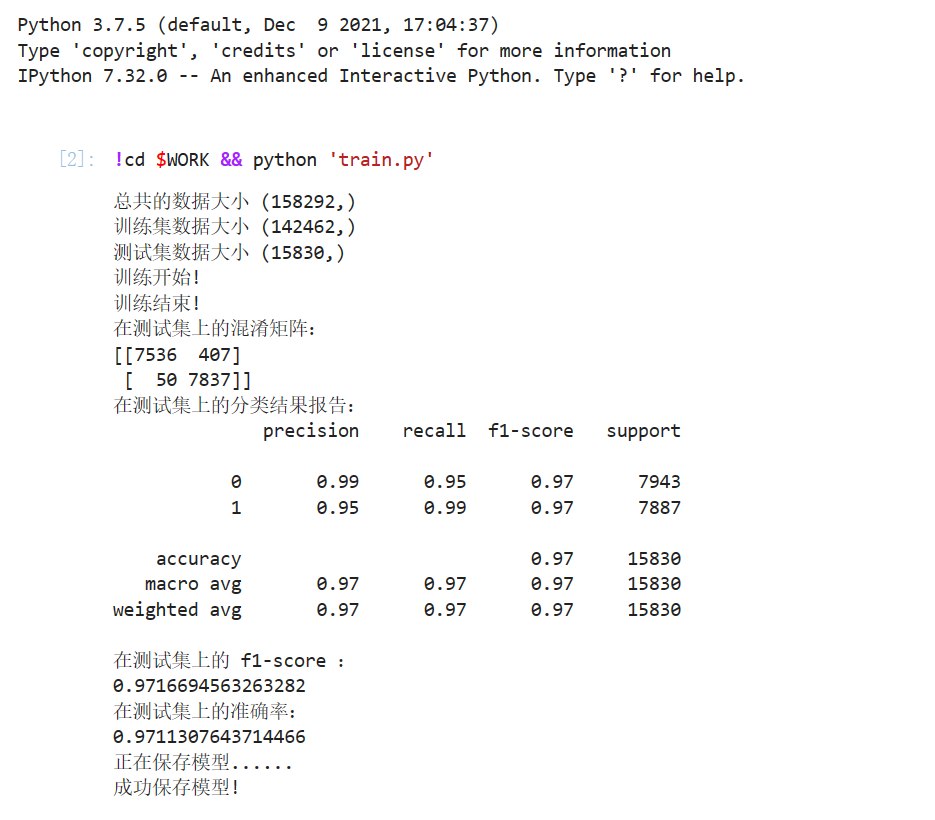
2.train.py

# 导入相关的包  
import warnings  
  
warnings.filterwarnings('ignore')  
import pandas as pd  
import numpy as np  
import joblib  
from sklearn import metrics  
from sklearn.model\_selection import train\_test\_split  
from sklearn.naive\_bayes import ComplementNB  
from sklearn.pipeline import Pipeline  
from sklearn.feature\_extraction.text import TfidfVectorizer  
from sklearn.preprocessing import MaxAbsScaler  
  
# 停用词库路径  
stopwords\_path = './datasets/5f9ae242cae5285cd734b91e-momodel/scu\_stopwords.txt'  
  
  
def read\_stopwords(stopwords\_path):  
 *"""  
 读取停用词库  
 :param stopwords\_path: 停用词库的路径  
 :return: 停用词列表  
 """* stopwords = []  
 with open(stopwords\_path, 'r', encoding='utf-8') as f:  
 stopwords = f.read()  
 stopwords = stopwords.splitlines()  
 return stopwords  
  
  
# 读取停用词  
stopwords = read\_stopwords(stopwords\_path)  
  
# 数据集的路径  
data\_path = "./datasets/5f9ae242cae5285cd734b91e-momodel/sms\_pub.csv"  
  
# 读取数据  
sms = pd.read\_csv(data\_path, encoding='utf-8')  
  
# 从原始数据集中选择相同数量的正例和负例样本，以创建一个平衡的数据集  
sms\_pos = sms[(sms['label'] == 1)]  
sms\_neg = sms[(sms['label'] == 0)].sample(frac=1.0)[: len(sms\_pos)]  
sms = pd.concat([sms\_pos, sms\_neg], axis=0).sample(frac=1.0)  
  
# 构建训练集和测试集,设置测试集占0.1，训练集占0.9，随机数种子为42  
X = np.array(sms.msg\_new)  
y = np.array(sms.label)  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, random\_state=42, test\_size=0.1)  
print("总共的数据大小", X.shape)  
print("训练集数据大小", X\_train.shape)  
print("测试集数据大小", X\_test.shape)  
print("训练开始!")  
  
# 搭建 pipeline  
pipeline = Pipeline([  
 ('tf', TfidfVectorizer(ngram\_range=(1, 2), token\_pattern=r"(?u)\b\w+\b", stop\_words=stopwords)),  
 ('MaxAbsScaler', MaxAbsScaler()),  
 ('classifier', ComplementNB())  
])  
  
# 训练 pipeline  
pipeline.fit(X\_train, y\_train)  
print("训练结束!")  
  
# 对测试集的数据集进行预测  
y\_pred = pipeline.predict(X\_test)  
  
# 在测试集上进行评估  
print("在测试集上的混淆矩阵：")  
print(metrics.confusion\_matrix(y\_test, y\_pred))  
print("在测试集上的分类结果报告：")  
print(metrics.classification\_report(y\_test, y\_pred))  
print("在测试集上的 f1-score ：")  
print(metrics.f1\_score(y\_test, y\_pred))  
print('在测试集上的准确率：')  
print(metrics.accuracy\_score(y\_test, y\_pred))  
  
# 在所有的样本上训练一次，充分利用已有的数据，提高模型的泛化能力  
pipeline.fit(X, y)  
  
# 保存训练的模型  
print("正在保存模型......")  
joblib.dump(pipeline, 'results/pipeline.model')  
print("成功保存模型!")

1. **实验结果**

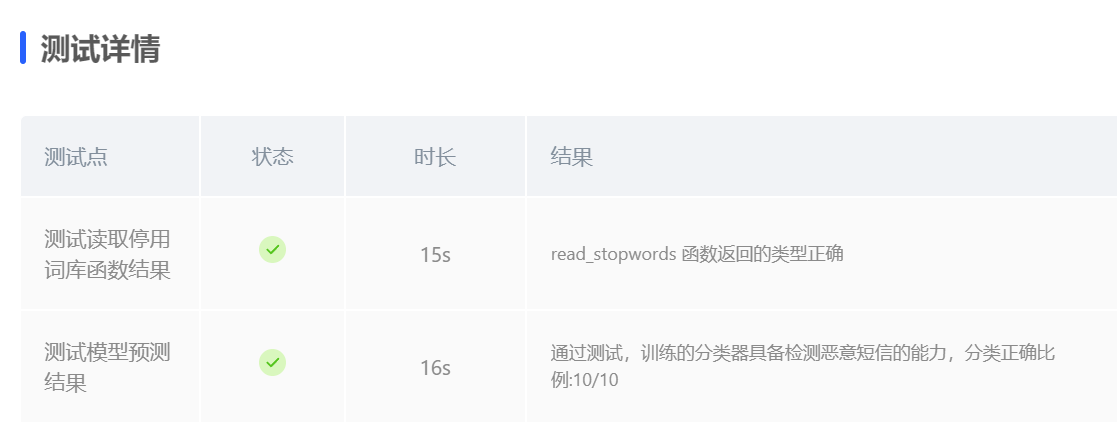
1.训练测试：

运行train.py，在测试集上进行验证，准确率到达了0.97，是一个较为有效的分类器：

****

2.系统测试：

在Mo环境下进行代码运行测试，对于给的示例短信集分类正确比例100%：



1. **总结**

本实验使用朴素贝叶斯分类器来垃圾短信进行识别，主要进行了以下几个优化：

a）对数据集进行了平衡化处理，保证正例和负例样本的数量相同，以更好地训练分类模型；

b）在构建模型的过程中，使用TF-IDF来对数据进行处理；

c）使用基于多项式的朴素贝叶斯模型ComplementNB，纠正标准多项朴素贝叶斯分类器所做的“严重假设”；

根据实验结果，该方法在测试集上的f1-score和准确率表现良好，证明了优化方案的有效性，成功得到了一个能够很好地对垃圾短信进行分类的模型。

遇到的困难主要是在特征提取方法选择上可能存在不足，TF-IDF虽然简单但可能无法充分表达语义信息。超参数和框架的选择也可能不够合理，需要进一步优化。

改进方向包括进一步优化特征提取方法、调整分类算法和模型超参数，例如使用网格搜索、随机搜索等方法进行更详细的调参，以提高模型性能。另外，选择更合适的停用词库也是一个重要的改进方向。

总之，通过这次实验，我不仅建立了一个实用的模型，还深入了解了自然语言处理和二分类问题的基本流程，学习了一些关键的细节，比如特征提取、模型选择、参数调优等等，对我理解人工智能和机器学习的原理和应用起了很大帮助。