**程序报告**

学号： 2113997 姓名：齐明杰

1. **问题重述**

（简单描述对问题的理解，从问题中抓住主干，必填）

====================================================================

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。随着手机的普及，垃圾短信在日常生活日益泛滥，已经严重的影响到了人们的正常生活娱乐，乃至社会的稳定。据 360 公司 2020 年第一季度有关手机安全的报告提到，360 手机卫士在第一季度共拦截各类垃圾短信约 34.4 亿条，平均每日拦截垃圾短信约 3784.7 万条。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；在面对量级如此巨大的短信数据时，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。

根据一段中文文本（200个中文字符以内），预测这段文本是否为垃圾短信。

注意：可以在准好的接口中实现的朴素贝叶斯模型（若使用可以修改除predict外的函数接口），也可以训练其他模型，但需要满足predict函数的输入输出符合格式要求

1. **设计思想**

（所采用的方法，有无对方法加以改进，该方法有哪些优化方向（参数调整，框架调整，或者指出方法的局限性和常见问题），伪代码，理论结果验证等… **思考题，非必填**）

====================================================================

**所采用的方法**：本实验采用了朴素贝叶斯分类器（Naive Bayes）对垃圾短信进行识别。朴素贝叶斯分类器是一种基于贝叶斯定理的分类算法，它假设特征之间相互独立。在本实验中，我们采用了一种简单的特征提取方法，即TF-IDF。同时，为了去除一些无关紧要的词汇，我们引入了停用词库。

**优化方向**：本实验的方法虽然较为简单，但在很多应用场景中表现出良好的效果。这种方法的优化方向主要包括：

1、参数调整：可以尝试调整TF-IDF和朴素贝叶斯分类器的参数，以获得更好的性能。

2、框架调整：可以尝试使用其他的特征提取方法和分类算法，如词袋模型、word2vec等特征提取方法，以及支持向量机、决策树等分类算法。

**方法局限性**：由于朴素贝叶斯分类器假设特征之间相互独立，这在实际应用中并不总是成立。此外，TF-IDF方法未考虑词汇之间的顺序，这可能会影响分类效果。  
 **伪代码**：

读取数据

分割训练集和测试集

构建pipeline（包括TF-IDF特征提取、朴素贝叶斯分类器）

训练pipeline

保存模型

计算测试集上的性能指标

1. **代码内容**

（能体现解题思路的主要代码，有多个文件或模块可用多个"===="隔开，必填）

**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)

# ----------------- 导入相关的库 -----------------

from sklearn import metrics

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

from sklearn.naive\_bayes import MultinomialNB

from sklearn.pipeline import Pipeline

from sklearn.feature\_extraction.text import CountVectorizer

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.preprocessing import MaxAbsScaler

# pipline\_list用于传给Pipline作为参数

pipeline\_list = [

('tfidf', TfidfVectorizer(token\_pattern=r"(?u)\b\w+\b", stop\_words=stopwords)),

#('MaxAbsScaler', MaxAbsScaler()),

('clf', MultinomialNB())

#('tfidf', TfidfVectorizer(ngram\_range=(1,2),token\_pattern=r"(?u)\b\w+\b", stop\_words=stopwords)),

#('MaxAbsScaler', MaxAbsScaler()),

#('clf', MultinomialNB())

#('tfidf', TfidfVectorizer(ngram\_range=(1,3),token\_pattern=r"(?u)\b\w+\b",stop\_words=stopwords)),

#('clf', MultinomialNB(alpha=0.99))

]

# 加载训练好的模型

from sklearn.externals 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

**train.py**

====================================================================

# 导入相关的包

import warnings

warnings.filterwarnings('ignore')

import pandas as pd

import numpy as np

from sklearn.externals import joblib

from sklearn import metrics

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

from sklearn.naive\_bayes import MultinomialNB

from sklearn.pipeline import Pipeline

from sklearn.feature\_extraction.text import CountVectorizer

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.preprocessing import MaxAbsScaler

# 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')

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)

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

"""

with open(stopwords\_path, 'r', encoding='utf-8') as f:

stopwords = f.read()

stopwords = stopwords.splitlines()

return stopwords

# 停用词库路径

stopwords\_path = './datasets/5f9ae242cae5285cd734b91e-momodel/scu\_stopwords.txt'

# 读取停用词

stopwords = read\_stopwords(stopwords\_path)

# 构建训练集和测试集

X = np.array(sms.msg\_new)

y = np.array(sms.label)

pipeline = Pipeline([

('tfidf', TfidfVectorizer(token\_pattern=r"(?u)\b\w+\b", stop\_words=stopwords)),

#('MaxAbsScaler', MaxAbsScaler()),

('clf', MultinomialNB())

#('tfidf', TfidfVectorizer(ngram\_range=(1,2),token\_pattern=r"(?u)\b\w+\b", stop\_words=stopwords)),

#('MaxAbsScaler', MaxAbsScaler()),

#('clf', MultinomialNB())

#('tfidf', TfidfVectorizer(ngram\_range=(1,3),token\_pattern=r"(?u)\b\w+\b",stop\_words=stopwords)),

#('clf', MultinomialNB(alpha=0.99))

])

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!")

pipeline.fit(X\_train, y\_train)

print("训练结束!")

print("正在保存权重......")

joblib.dump(pipeline, 'results/pipeline.model')

print("成功保存权重!")

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))

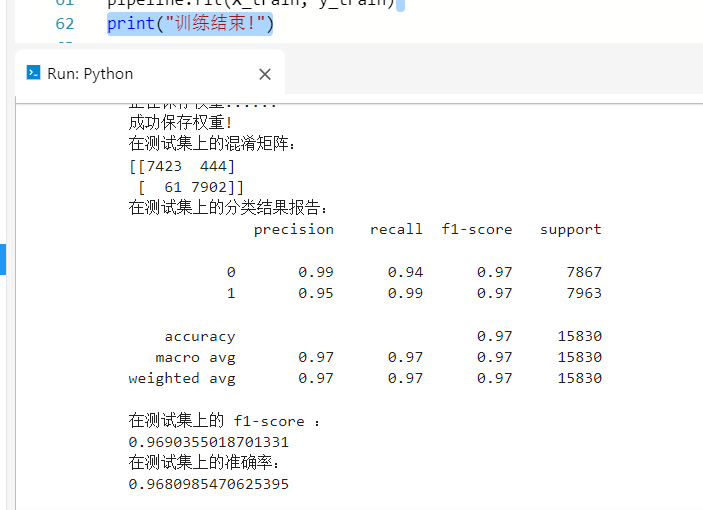
print('在测试集上的准确率：')

print(metrics.accuracy\_score(y\_test, y\_pred))

1. **实验结果**

（实验结果，必填）

====================================================================





1. **总结**

（自评分析（是否达到目标预期，可能改进的方向，实现过程中遇到的困难，从哪些方面可以提升性能，模型的超参数和框架搜索是否合理等），**思考题，非必填**）

====================================================================

本实验通过采用朴素贝叶斯分类器和TF-IDF特征提取方法，实现了对垃圾短信的识别。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，达到了预期目标。

实验过程中遇到的困难主要是特征提取方法的选择，TF-IDF方法虽然简单，但可能无法充分表达短信中的语义信息。为了提升性能，可以从以下方面进行优化：

1、采用更复杂的特征提取方法，如word2vec等。

2、尝试不同的分类算法，如支持向量机、决策树等。

3、调整模型的超参数，如朴素贝叶斯分类器的平滑参数，TF-IDF的ngram参数等。

在实验中，模型的超参数和框架选择可能不够合理，可以通过网格搜索、随机搜索等方法进行更详细的调参，以找到性能更优的模型配置。此外，本实验中采用的停用词库可能不是最佳选择，可以尝试使用其他停用词库或通过自定义停用词库来进一步提高分类性能。

总之，本实验实现了一个垃圾短信识别的基本框架，并在测试集上取得了令人满意的结果。通过进一步优化特征提取方法、调整分类算法以及调整模型超参数，有望进一步提高垃圾短信识别的性能。