**程序报告**

学号： 2112060 姓名：孙蕗

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）

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

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。

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**）

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

1. 文本向量化可以选择 CountVectorizer 或者 TfidfVectorizer，适当调节里面的参数，如 ngram\_range。本次实验中文本向量化使用了TfidfVectorizer。
2. 更换更好的停用词库，放在 results 目录下。本次实验中使用的停用词库为哈工大停用词表、百度停用词表、四川大学机器智能实验室停用词库、中文停用词表的汇总。
3. 尝试进行数据进行归一化，可以采用 StandardScaler 或者 MaxAbsScaler。本次实验中使用MaxAbsScaler进行数据归一化。
4. 适当调节分类器的参数，提高模型的表现
5. **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）

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

import os

os.environ["HDF5\_USE\_FILE\_LOCKING"] = "FALSE"

# ---------- 停用词库路径，若有变化请修改 -------------

stopwords\_path = r'scu\_stopwords.txt'

#stopwords\_path = r'stopwords\_all.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 ------------

#----------------------------------------------

return stopwords

# 读取停用词

stopwords = read\_stopwords(stopwords\_path)

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

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

from sklearn.pipeline import Pipeline

from sklearn.feature\_extraction.text import CountVectorizer

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.naive\_bayes import BernoulliNB

from sklearn.naive\_bayes import MultinomialNB

from sklearn.naive\_bayes import ComplementNB

from sklearn import preprocessing

#from sklearn.preprocessing import StandardScaler

#from sklearn.preprocessing import MaxAbsScaler

# pipline\_list用于传给Pipline作为参数

pipeline\_list = [

# --------------------------- 需要完成的代码 ------------------------------

# = = = = = = 以下代码仅供参考 = = = = = =

('cv',TfidfVectorizer(token\_pattern=r"(?u)\b\w+\b", stop\_words=stopwords,ngram\_range=(1, 4)) ),

('max\_abs\_scaler',preprocessing.MaxAbsScaler()),

('classifier', MultinomialNB()),

# = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =

# ------------------------------------------------------------------------

]

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

# 搭建 pipeline

pipeline = Pipeline(pipeline\_list)

# 训练 pipeline

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

# 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

from sklearn import metrics

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))

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

# 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，请将模型保存在 results 目录下

from sklearn.externals import joblib

pipeline\_path = 'results/pipeline.model'

joblib.dump(pipeline, pipeline\_path)

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

# 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]

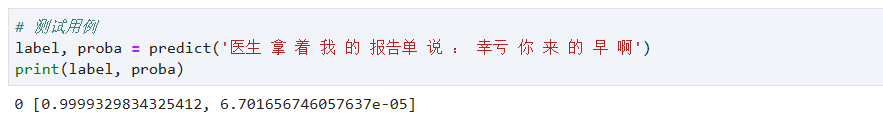
proba = list(pipeline.predict\_proba([message])[0])

return label, proba

1. **实验结果**

（实验结果，必填）

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



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

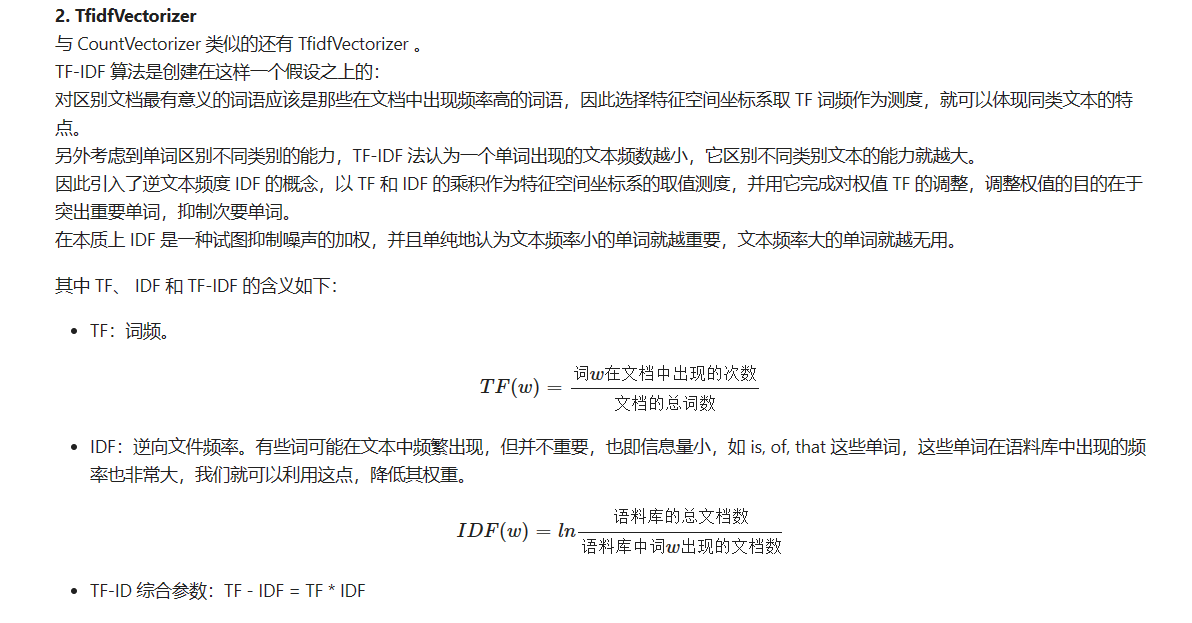


1. **总结**

（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**）

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

1. 更换了停用词库，使用的停用词库为哈工大停用词表、百度停用词表、四川大学机器智能实验室停用词库、中文停用词表的汇总。
2. 文本向量化使用了TfidfVectorizer，调节了ngram\_range参数。



参数：

input：string{'filename', 'file', 'content'}

如果是'filename'，序列作为参数传递给拟合器，预计为文件名列表，这需要读取原始内容进行分析

如果是'file'，序列项目必须有一个”read“的方法（类似文件的对象），被调用作为获取内存中的字节数

否则，输入预计为序列串，或字节数据项都预计可直接进行分析。

encoding：string， ‘utf-8’by default

如果给出要解析的字节或文件，此编码将用于解码

decode\_error: {'strict', 'ignore', 'replace'}

如果一个给出的字节序列包含的字符不是给定的编码，指示应该如何去做。默认情况下，它是'strict'，这意味着的UnicodeDecodeError将提高，其他值是'ignore'和'replace'

strip\_accents: {'ascii', 'unicode', None}

在预处理步骤中去除编码规则(accents)，”ASCII码“是一种快速的方法，仅适用于有一个直接的ASCII字符映射，"unicode"是一个稍慢一些的方法，None（默认）什么都不做

analyzer：string，{'word', 'char'} or callable

定义特征为词（word）或n-gram字符，如果传递给它的调用被用于抽取未处理输入源文件的特征序列

preprocessor：callable or None（default）

当保留令牌和”n-gram“生成步骤时，覆盖预处理（字符串变换）的阶段

tokenizer：callable or None(default)

当保留预处理和n-gram生成步骤时，覆盖字符串令牌步骤

ngram\_range: tuple(min\_n, max\_n)

要提取的n-gram的n-values的下限和上限范围，在min\_n <= n <= max\_n区间的n的全部值

stop\_words：string {'english'}, list, or None(default)

如果未english，用于英语内建的停用词列表

如果未list，该列表被假定为包含停用词，列表中的所有词都将从令牌中删除

如果None，不使用停用词。max\_df可以被设置为范围[0.7, 1.0)的值，基于内部预料词频来自动检测和过滤停用词

lowercase：boolean， default True

在令牌标记前转换所有的字符为小写

token\_pattern：string

正则表达式显示了”token“的构成，仅当analyzer == ‘word’时才被使用。两个或多个字母数字字符的正则表达式（标点符号完全被忽略，始终被视为一个标记分隔符）。

max\_df： float in range [0.0, 1.0] or int, optional, 1.0 by default

当构建词汇表时，严格忽略高于给出阈值的文档频率的词条，语料指定的停用词。如果是浮点值，该参数代表文档的比例，整型绝对计数值，如果词汇表不为None，此参数被忽略。

min\_df：float in range [0.0, 1.0] or int, optional, 1.0 by default

当构建词汇表时，严格忽略低于给出阈值的文档频率的词条，语料指定的停用词。如果是浮点值，该参数代表文档的比例，整型绝对计数值，如果词汇表不为None，此参数被忽略。

max\_features： optional， None by default

如果不为None，构建一个词汇表，仅考虑max\_features--按语料词频排序，如果词汇表不为None，这个参数被忽略

vocabulary：Mapping or iterable， optional

也是一个映射（Map）（例如，字典），其中键是词条而值是在特征矩阵中索引，或词条中的迭代器。如果没有给出，词汇表被确定来自输入文件。在映射中索引不能有重复，并且不能在0到最大索引值之间有间断。

binary：boolean， False by default

如果未True，所有非零计数被设置为1，这对于离散概率模型是有用的，建立二元事件模型，而不是整型计数

dtype：type， optional

通过fit\_transform()或transform()返回矩阵的类型

norm：'l1', 'l2', or None,optional

范数用于标准化词条向量。None为不归一化

use\_idf：boolean， optional

启动inverse-document-frequency重新计算权重

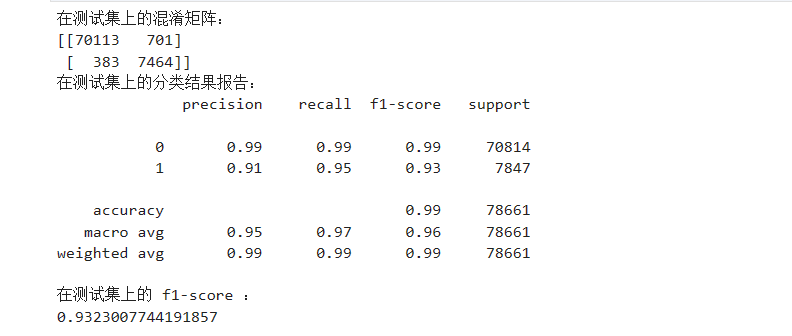
smooth\_idf：boolean，optional

通过加1到文档频率平滑idf权重，为防止除零，加入一个额外的文档

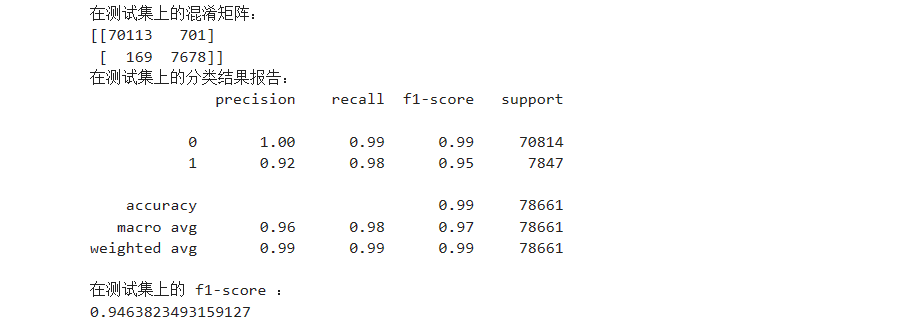
sublinear\_tf：boolean， optional

应用线性缩放TF，例如，使用1+log(tf)覆盖tf

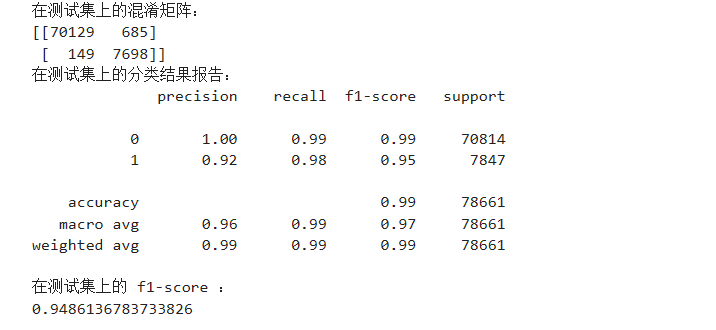
1. ngram\_range=(1, 1)（默认）



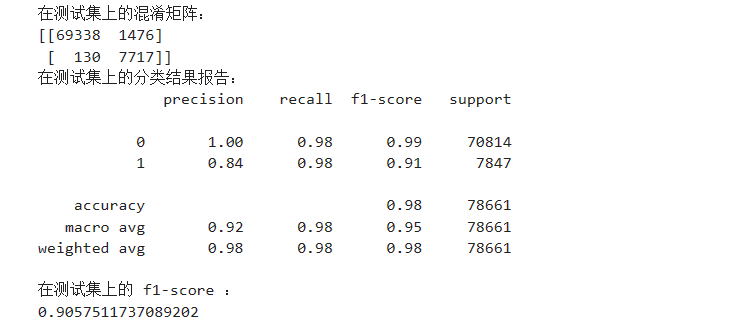
1. ngram\_range=(1, 2)



1. ngram\_range=(1, 3)



1. ngram\_range=(2, 3)



1. MaxAbsScaler进行数据归一化

通过除以每个特征中的最大绝对值，将每个特征单独重新缩放到范围 [-1, 1]。它不会移动/居中数据，因此不会破坏任何稀疏性。