

**《数学建模综合实训》课程论文**

姓 名 朱拂

学 号 202283270243

专 业 计算机科学与技术

班 级 22级 计科1班

教 师 汪佳玲

**二Ｏ二三年 十二月 二十七日**

基于Perceptron模型的垃圾短信识别算法

朱拂

南京信息工程大学计算机学院、网络空间安全学院，江苏 南京 210044

**摘要：**本文全面探讨了垃圾短信分类问题。作为背景，垃圾短信越来越多地困扰着人们的生活。在分类理论上，本文涉及统计学和机器学习，详细分析了CountVectorizer、逻辑回归、SVM等算法。实践上，本文基于scikit-learn和lazypredict工具，对多种算法在这个问题上的性能展开了全面分析，并选择Perceptron模型作为分类算法，进一步展开讨论。模型运行速度快、准确率高，且具有泛化能力。

**关键词：**机器学习、垃圾短信识别、模型评估、文本向量化、数学建模

# 绪论

## 背景介绍

随着信息时代的到来，通信业务的快速发展给人们的日常生活带来了便利，但也带来了一些负面影响，其中之一就是垃圾短信的泛滥。垃圾短信是指未经用户同意或请求，以营销、推销、广告等为目的，向不特定多数人发送的短信。垃圾短信不仅会干扰用户正常的生活，还会传播不良信息，影响社会秩序。

近年来，我国垃圾短信的数量呈逐年上升趋势。据统计，2022年我国垃圾短信总量超过了1000亿条。垃圾短信的泛滥给社会造成了严重的危害。具体表现在以下几个方面：

* 干扰用户正常生活。垃圾短信会频繁地出现在用户的手机上，严重影响用户的正常工作和学习。
* 传播不良信息。垃圾短信中常常包含色情、暴力、赌博等不良信息，会对青少年的身心健康造成不良影响。
* 损害个人隐私。垃圾短信中常常会包含用户的个人信息，会给用户的财产安全带来威胁。

因此，识别和过滤垃圾短信具有重要的实际意义和应用价值。识别和过滤垃圾短信可以有效地减少垃圾短信的数量，保护用户的合法权益，维护社会秩序。

## 本研究的目标

本研究的目标是建立一种有效的垃圾短信识别模型。具体来说，本研究将解决以下三个问题：

* 通过分析附件spamdata\_T1.csv中文本数据，确定垃圾短信的特征，并给出短信类别。
* 进一步分析附件spamdata\_T2.csv中垃圾短信及正常短信的特征，研究垃圾短信和正常短信的差异性。
* 建立识别垃圾短信的数学模型，并用于识别附件testdata.csv中的垃圾短信。

## 研究方法

本研究将采用以下方法来实现目标：

* 数据处理：对数据进行清洗、分词、去停用词等操作，并用Scikit的Counter向量化器对推文文本进行矢量化，将文本数据转换为数值特征。
* 模型评估：使用测试集对常用分类模型进行评估，以选择最佳模型。本研究将使用常见的评估指标，如准确度、精确度、召回率和F1得分等，评估LinearSVC、Perceptron、BernoulliNB等常用分类模型。
* 模型训练：采用机器学习方法训练垃圾短信识别模型。本研究将采用Perceptron

模型。

* 模型预测：本研究将使用选择的模型对testdata进行预测，判断是否为垃圾短信。

# 相关理论

## 文本向量化

文本向量化是将文本转换为数值向量的过程，以便计算机可以处理文本。文本向量化是机器学习和自然语言处理中的一个重要步骤，它在文本分类、文本聚类、文本检索等任务中具有广泛的应用。

文本向量化的方法有很多，常见的文本向量化方法包括：

1. 词袋模型：词袋模型将文本视为一个词的集合，每个词的出现次数作为该词的特征。
2. TF-IDF模型：TF-IDF模型将词袋模型的思想进行了改进，考虑了词频和逆文档频率两个因素。
3. Word2Vec模型：Word2Vec模型是基于神经网络的词嵌入模型，它可以学习词与词之间的语义关系。

## CounterVectorizer

CountVectorizer是scikit-learn中的一个文本特征提取工具，它将文本转换为词频矩阵。其算法步骤如下：

1. 首先，将文本数据分割为单独的单词或短语，称为“标记化”。将文本数据表示为一个矩阵X，其中每行表示一个文档，每列表示一个单词，矩阵中的每个元素表示第i个文档中第j个单词出现的次数。
2. 其次，CountVectorizer将所有文档中的单词进行计数，并将其存储在一个矩阵中。该矩阵的每行表示一个文档，每列表示一个单词。具体地，CountVectorizer通过以下公式计算文档中每个单词的出现次数：

其中表示单词j在文档i中出现的次数，是文档i中第k个单词出现的次数，表示文档i中第k个单词的索引，j表示待计数的单词的索引。

1. 然后，可以通过一些参数来控制CountVectorizer的行为，例如选择是否去掉停用词、是否进行词根化等。这些参数会影响到文本数据的标准化和处理方式。
2. 最后，可以使用所得到的矩阵作为输入来训练分类或聚类模型，或进行其他的文本分析任务。例如，可以使用朴素贝叶斯分类器对文档进行分类，其中每个文档的向量表示为一个多项式分布，其参数可以通过极大似然估计计算得到。

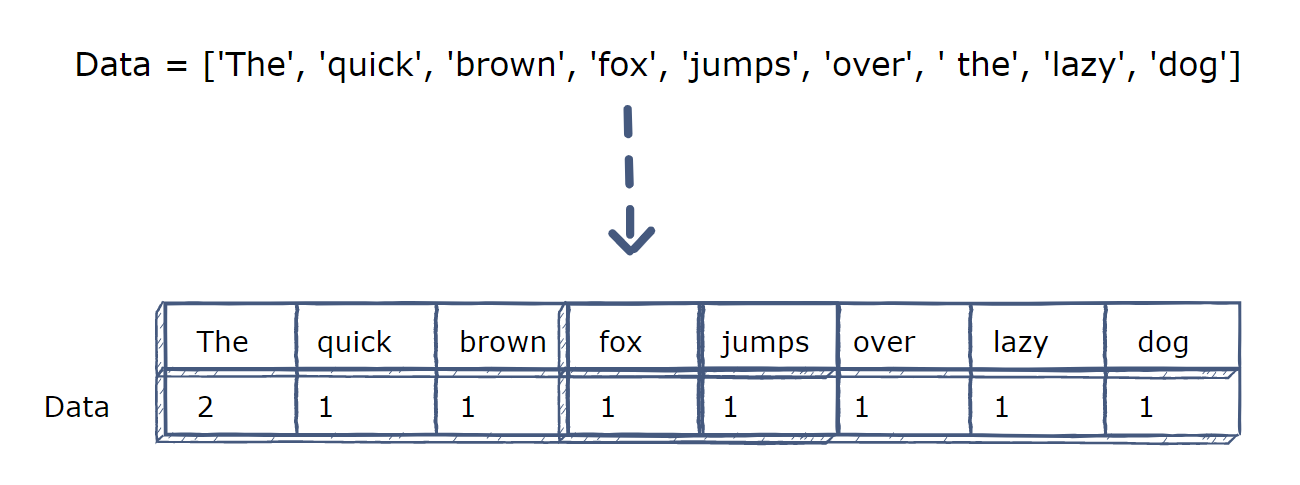


图1 CountVectorizer文本特征提取示例

## 逻辑回归算法

逻辑回归是一种常用的分类算法，它可以用于二分类和多分类问题。逻辑回归的核心思想是将线性函数转换为概率值，从而实现分类。在自然语言处理中，逻辑回归可以用于文本分类、情感分析等任务。

逻辑回归算法的步骤如下[1]：

1. 假设要将数据分为两类，将每个样本表示为一个特征向量x和标签y。其中x包含n个特征，y为0或1。
2. 定义一个代价函数，其目的是最小化分类错误的概率。通常使用对数损失函数作为代价函数。
3. 使用梯度下降或其他优化算法最小化代价函数，从而得到模型参数。
4. 根据模型参数，对新的数据进行分类。

逻辑回归算法的数学表达式如下：

给定一个样本的特征向量x和标签y，假设模型参数为w，b，那么该样本属于类别1的概率为：

属于类别0的概率为：

其中，是线性函数，将其输入到Sigmoid函数中，得到属于类别1的概率。对于多分类问题，可以使用softmax函数将线性函数转换为概率分布。

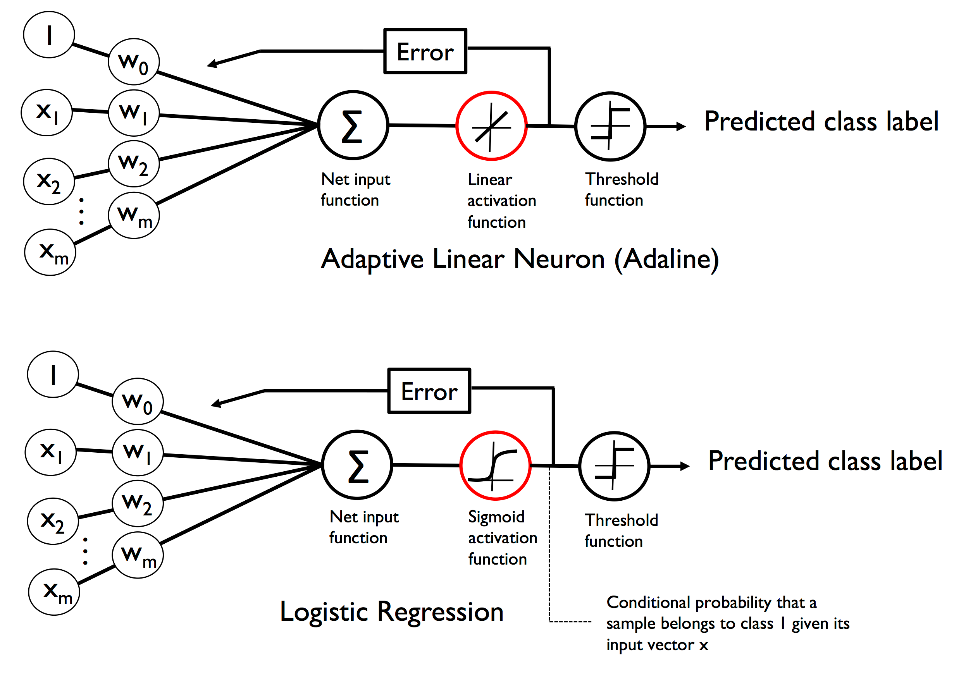


图2 逻辑回归算法原理图示

## SVM算法

支持向量机（Support Vector Machine，SVM）是一种监督学习模型，用于分类和回归。SVM 的基本思想是找到一个能够将不同类别的样本点尽可能分开的超平面。在垃圾短信识别中，可以将短信中的词语作为特征，然后使用 SVM 进行训练和预测。

SVM算法的步骤如下[2]：

1. 假设要将数据分为两类，将每个样本表示为一个特征向量x和标签y。其中x包含n个特征，y为-1或1。
2. 构建一个超平面，使其能够最大化不同类别之间的间隔。通常使用带有正则化项的目标函数作为代价函数。
3. 使用梯度下降或其他优化算法最小化代价函数，从而得到超平面参数。
4. 根据超平面参数，对新的数据进行分类。

SVM算法的数学表达式如下：

给定一个样本的特征向量x和标签y，假设超平面参数为w，b，那么该样到超平面的距离为：  
分类决策函数为：  
其中，||w||是向量w的范数，sign函数是符号函数，即f(x)为1或-1。通过最大化间隔，可以得到优化问题的目标函数：

其中，第一项是正则化项，第二项是损失函数，C是正则化系数，用于平衡间隔和误分类的权重。

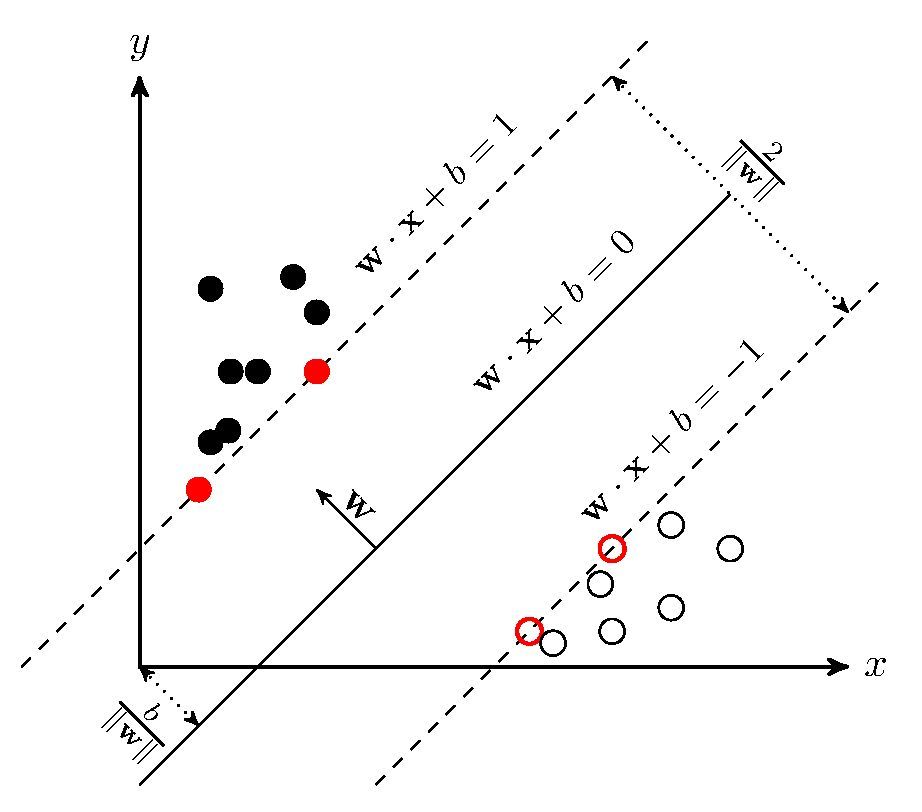


图3 SVM算法步骤原理图示

# 垃圾短信特征分析

人工标注spamdata\_T1.csv，观察垃圾短信和非垃圾短信的区别。注意到垃圾短信有以下特征：

1. 商业推广相关的垃圾短信中，很多有引号。
2. 很多垃圾短信含有Free、Call、Click等词汇。
3. 包含链接的很多是垃圾短信。
4. 有些垃圾短信含有大写单词。

总体上，只要含有特定字符或词汇，就可以认为是垃圾短信。也就是说，识别垃圾短信不需要发现词汇之间的关系。这使得词袋模型更有可能成功。

# 性能分析与模型选择

## 文本分词和向量化

我们使用CountVectorizer来向量化文本，因为它在分类问题中有不错的的性能[3]。在scikit-learn中，已经有很好的封装和实现。我们可以用以下代码来创建一个CountVectorizer。

vectorizer = CountVectorizer()

可以用stop\_words参数来选用停用词，也就是去除文本中常见、无意义的词。具体来说，

vectorizer = CountVectorizer(stop\_words='english')

会删去英语中的停用词，包括the, is, of, and, to等。我们会比较去停用词前后模型性能的差异。

完整的向量化实现如下。

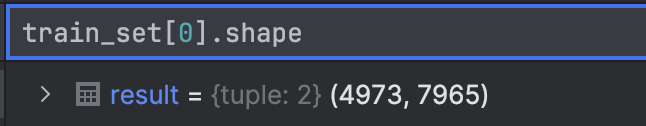
def read\_train\_dataset(to\_predict: str = None):  
 data\_iter = train\_data\_iter()  
 if to\_predict is not None:  
 data\_iter = itertools.chain(data\_iter, [(-1, to\_predict)])  
 data\_iter = itertools.tee(data\_iter)  
 return (csr\_matrix(vectorizer.fit\_transform(row[1] for row in data\_iter[0])),  
 list(row[0] for row in data\_iter[1]))

## 模型性能分析

我们使用lazypredict工具，测试scikit-learn中每种经典模型的性能。具体来说，我们从spamdata\_T2.csv中读取标记好的数据，为两列。第一列（train\_set[0]）是向量化过的文本。第二列是0或1，分别表示垃圾短信和正常短信。代码如下。

train\_set = read\_train\_dataset()  
x\_train, x\_test, y\_train, y\_test = train\_test\_split(train\_set[0], train\_set[1], train\_size=0.2, random\_state=42)  
clf = LazyClassifier(predictions=True)  
models, \_ = clf.fit(x\_train.toarray(), x\_test.toarray(), y\_train, y\_test)  
print(models.to\_string())

检查train\_set，它是4973个7965维的稀疏矩阵。



运行这个测试，结果如下。

表1 各个模型在未去停用词的情况下准确率（Accuracy）、平衡准确率（Balanced）、ROC AUC、F1得分（F1 Score）汇总

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Model | Accuracy | Balanced Accuracy | ROC AUC | F1 Score | Time Taken |
| Perceptron | 0.97 | 0.94 | 0.94 | 0.97 | 0.61 |
| PassiveAggressiveClassifier | 0.97 | 0.93 | 0.93 | 0.97 | 0.72 |
| LinearSVC | 0.98 | 0.93 | 0.93 | 0.98 | 4.45 |
| BaggingClassifier | 0.96 | 0.89 | 0.89 | 0.96 | 2.64 |
| AdaBoostClassifier | 0.95 | 0.87 | 0.87 | 0.95 | 3.72 |
| XGBClassifier | 0.95 | 0.86 | 0.86 | 0.95 | 0.74 |
| LGBMClassifier | 0.96 | 0.86 | 0.86 | 0.96 | 2.72 |
| DecisionTreeClassifier | 0.95 | 0.85 | 0.85 | 0.95 | 0.93 |
| GaussianNB | 0.88 | 0.84 | 0.84 | 0.89 | 0.76 |
| ExtraTreesClassifier | 0.95 | 0.83 | 0.83 | 0.95 | 0.60 |
| NearestCentroid | 0.95 | 0.82 | 0.82 | 0.94 | 1.57 |
| RidgeClassifierCV | 0.95 | 0.81 | 0.81 | 0.94 | 1.51 |
| RidgeClassifier | 0.95 | 0.81 | 0.81 | 0.94 | 1.12 |
| LogisticRegression | 0.94 | 0.79 | 0.79 | 0.94 | 1.03 |
| RandomForestClassifier | 0.93 | 0.76 | 0.76 | 0.92 | 0.67 |
| ExtraTreeClassifier | 0.92 | 0.75 | 0.75 | 0.91 | 0.49 |
| LinearDiscriminantAnalysis | 0.92 | 0.72 | 0.72 | 0.90 | 0.74 |
| SGDClassifier | 0.63 | 0.66 | 0.66 | 0.69 | 3.86 |
| CalibratedClassifierCV | 0.90 | 0.63 | 0.63 | 0.87 | 16.13 |
| SVC | 0.89 | 0.59 | 0.59 | 0.85 | 13.94 |
| LabelSpreading | 0.22 | 0.55 | 0.55 | 0.19 | 1.39 |
| LabelPropagation | 0.22 | 0.55 | 0.55 | 0.19 | 1.33 |
| QuadraticDiscriminantAnalysis | 0.21 | 0.54 | 0.54 | 0.18 | 5.32 |
| BernoulliNB | 0.86 | 0.50 | 0.50 | 0.80 | 0.93 |
| KNeighborsClassifier | 0.86 | 0.50 | 0.50 | 0.80 | 0.63 |
| DummyClassifier | 0.86 | 0.50 | 0.50 | 0.80 | 0.46 |

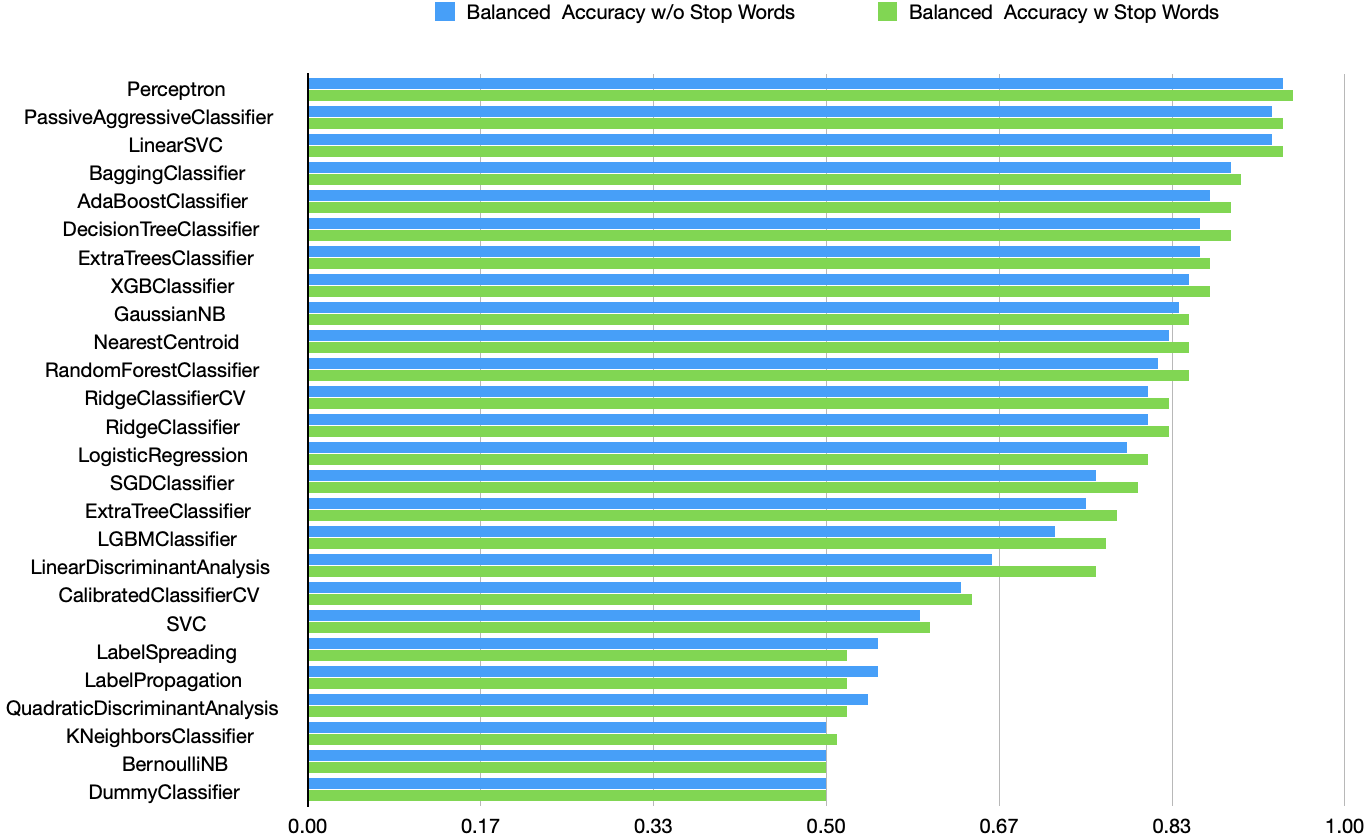
现在，我们去除停用词，重新运行这个测试。

表2 各个模型在去停用词的情况下准确率（Accuracy）、平衡准确率（Balanced）、ROC AUC、F1得分（F1 Score）汇总

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Model | Accuracy | Balanced Accuracy | ROC AUC | F1 Score | Time Taken |
| Perceptron | 0.97 | 0.95 | 0.95 | 0.97 | 0.60 |
| PassiveAggressiveClassifier | 0.97 | 0.94 | 0.94 | 0.97 | 0.76 |
| LinearSVC | 0.98 | 0.94 | 0.94 | 0.98 | 4.87 |
| BaggingClassifier | 0.96 | 0.90 | 0.90 | 0.96 | 2.72 |
| AdaBoostClassifier | 0.96 | 0.89 | 0.89 | 0.96 | 4.10 |
| XGBClassifier | 0.96 | 0.89 | 0.89 | 0.96 | 1.17 |
| LGBMClassifier | 0.95 | 0.87 | 0.87 | 0.95 | 0.90 |
| DecisionTreeClassifier | 0.95 | 0.87 | 0.87 | 0.95 | 0.72 |
| GaussianNB | 0.91 | 0.85 | 0.85 | 0.91 | 0.72 |
| ExtraTreesClassifier | 0.96 | 0.85 | 0.85 | 0.96 | 2.41 |
| NearestCentroid | 0.96 | 0.85 | 0.85 | 0.96 | 0.58 |
| RidgeClassifierCV | 0.95 | 0.83 | 0.83 | 0.95 | 1.55 |
| RidgeClassifier | 0.95 | 0.83 | 0.83 | 0.95 | 1.13 |
| LogisticRegression | 0.95 | 0.81 | 0.81 | 0.94 | 0.81 |
| RandomForestClassifier | 0.95 | 0.80 | 0.80 | 0.94 | 1.35 |
| ExtraTreeClassifier | 0.93 | 0.78 | 0.78 | 0.92 | 0.50 |
| LinearDiscriminantAnalysis | 0.85 | 0.77 | 0.77 | 0.86 | 4.34 |
| SGDClassifier | 0.93 | 0.76 | 0.76 | 0.92 | 0.71 |
| CalibratedClassifierCV | 0.90 | 0.64 | 0.64 | 0.87 | 17.55 |
| SVC | 0.89 | 0.60 | 0.60 | 0.86 | 14.96 |
| LabelSpreading | 0.18 | 0.52 | 0.52 | 0.12 | 1.53 |
| LabelPropagation | 0.18 | 0.52 | 0.52 | 0.12 | 1.35 |
| QuadraticDiscriminantAnalysis | 0.18 | 0.52 | 0.52 | 0.11 | 4.79 |
| BernoulliNB | 0.86 | 0.51 | 0.51 | 0.80 | 0.68 |
| KNeighborsClassifier | 0.86 | 0.50 | 0.50 | 0.80 | 0.94 |
| DummyClassifier | 0.86 | 0.50 | 0.50 | 0.80 | 0.47 |

比较这两次测试的结果。去停用词后，大部分模型表现比之前更好，平均增长1.94%。可以认为，去停用词对这个场景有作用。

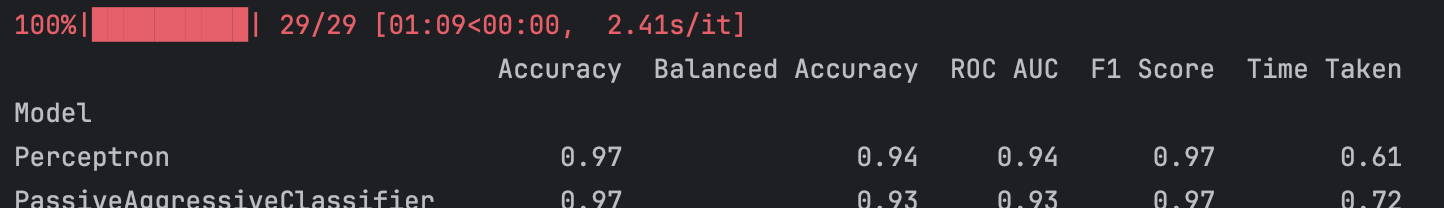
图3 各个模型在去停用词前后平衡准确率(Balanced)的对比



## 模型选择

根据上面的测试，我们应该选择Perceptron模型来做预测。其它可能选择的模型是LinearSVC和PassiveAggressiveClassifier。相比于PassiveAggressiveClassifier，Perceptron在去停用词前后的准确性都更高，耗时稍更短。相比于LinearSVC，Perceptron平衡准确率更高，且耗时是它的十二分之一。

总体来说，CountVectorizer向量化的方法是十分有效的。平衡准确率最高达到95%。耗时也很短，测试的每个迭代只需要2.41秒（Apple M1 Pro 16G）。



# 垃圾短信识别

read\_train\_dataset函数采用了一个参数to\_predict。带有这个参数的调用会将要预测的文本追加在CountVectorizer的末尾一并向量化。在推理的时候，先从其它部分进行训练，再对末尾的向量进行预测。具体实现如下。

def predict(sms: str):  
 clf = Perceptron()  
 data\_set = read\_train\_dataset(to\_predict=sms)  
 train\_set\_x, train\_set\_y = data\_set[0][0:-2], data\_set[1][0:-2]  
 clf.fit(train\_set\_x, train\_set\_y)  
 return clf.predict(data\_set[0][-1])[0]

我们用测试数据集testdata.csv中的数据测试这个程序。在这个测试中，此方法与人工标注结果完全一致。可以认为，这个方法具有泛化能力。

表3 测试数据集的模型预测与人工标注

|  |  |  |
| --- | --- | --- |
| Text | Mine | Human |
| Will ü b going to esplanade fr home? | ham | ham |
| Hope you are having a great new semester. Do wish you the very best. You are made for greatness. | ham | ham |
| Yeah my usual guy's out of town but there're definitely people around I know | ham | ham |
| What I'm saying is if you haven't explicitly told nora I know someone I'm probably just not gonna bother | ham | ham |
| Yar lor he wan 2 go c horse racing today mah, so eat earlier lor. I ate chicken rice. U? | ham | ham |
| Would you like to see my XXX pics they are so hot they were nearly banned in the uk! | spam | spam |
| Nah can't help you there, I've never had an iphone | ham | ham |
| WELL DONE! Your 4\* Costa Del Sol Holiday or £5000 await collection. Call 09050090044 Now toClaim. SAE, TCs, POBox334, Stockport, SK38xh, Cost£1.50/pm, Max10mins | spam | spam |
| Well I'm going to be an aunty! | ham | ham |
| This weeks SavaMob member offers are now accessible. Just call 08709501522 for details! SavaMob, POBOX 139, LA3 2WU. Only £1.50/week. SavaMob - offers mobile! | spam | spam |
| Sorry, I'll call you  later. I am in meeting sir. | ham | ham |
| 100 dating service cal;l 09064012103 box334sk38ch | spam | spam |
| I will reach ur home in  &lt;#&gt;  minutes | ham | ham |
| Haha... Hope ü can hear the receipt sound... Gd luck! | ham | ham |
| Al he does is moan at me if n e thin goes wrong its my fault&al de arguments r my fault&fed up of him of himso y bother? Hav 2go, thanx.xx | ham | ham |
| Not thought bout it... || Drink in tap & spile at seven. || Is that pub on gas st off broad st by canal. || Ok? | ham | ham |
| My sister in law, hope you are having a great month. Just saying hey. Abiola | ham | ham |
| CHEERS FOR CALLIN BABE.SOZI CULDNT TALKBUT I WANNATELL U DETAILS LATER WENWECAN CHAT PROPERLY X | ham | ham |
| Super msg da:)nalla timing. | ham | ham |
| Was the farm open? | ham | ham |
| I'm nt goin, got somethin on, unless they meetin 4 dinner lor... Haha, i wonder who will go tis time... | ham | ham |
| Babe ! How goes that day ? What are you doing ? Where are you ? I sip my cappuccino and think of you, my love ... I send a kiss to you from across the sea | ham | ham |
| Your account has been credited with 500 FREE Text Messages. To activate, just txt the word: CREDIT to No: 80488 T&Cs www.80488.biz | spam | spam |
| have \* good weekend. | ham | ham |
| My phone | ham | ham |
| I've sent ü my part.. | ham | ham |
| Okay, good, no problem, and thanx! | ham | ham |
| Thanks  and ! Or bomb and date as my phone wanted to say! | ham | ham |
| Aight do you still want to get money | ham | ham |
| I noe la... U wana pei bf oso rite... K lor, other days den... | ham | ham |

# 参考文献

[1] Choudhary, Kailash and Ruby Beniwal. “Xplore Word Embedding Using CBOW Model and Skip-Gram Model.” *2021 7th International Conference on Signal Processing and Communication (ICSC)* (2021): 267-270.

[2] Shah, Kanish et al. “A Comparative Analysis of Logistic Regression, Random Forest and KNN Models for the Text Classification.” *Augmented Human Research* 5 (2020): 1-16.

[3] Baby, Anusuya. “Detecting deceptive reviews using text classification.” *Research Gate* (2023): 13-14.