中国地址大学实验报告

题目：垃圾邮件分类实验

**学院：**计算机科学与技术

**班级：**

**学号：**

**姓名：**

**导师：**

2024年7月2日

**目录**

[**I. 介绍 3**](#_Toc170819541)

[**II. 实验背景 4**](#_Toc170819542)

[**III. 实验思路 5**](#_Toc170819543)

[**IV. 实验实施 6**](#_Toc170819544)

[**4.1 数据集** 6](#_Toc170819545)

[**4.2 模型实现** 6](#_Toc170819546)

[**V. 实验结果与讨论 9**](#_Toc170819547)

[**VI. 实验结论 11**](#_Toc170819548)

# 介绍

在数字时代，电子邮件已成为个人和职业用途的主要沟通方式。它快捷、高效，并让我们与世界各地的人保持联系。然而，这种便利也带来了一个缺点：垃圾邮件。

垃圾邮件，又称垃圾电子邮件，是通过电子邮件大量发送的未经请求的邮件。这些邮件通常包含广告，但也可能带有有害链接或附件，这些链接或附件可能会导致您进入钓鱼网站或在您的计算机上安装恶意软件。

垃圾邮件问题不仅仅是令人烦恼。它给个人和组织都带来了重大挑战和风险。对于个人而言，垃圾邮件会使邮件箱变得杂乱，难以找到重要邮件。此外，如果用户不小心点击了恶意链接或下载了有害附件，它们还可能导致安全漏洞。

对于企业来说，问题更加严重。垃圾邮件会导致员工工作效率下降，因为员工需要花时间整理收件箱。如果员工接触到恶意内容，垃圾邮件还会导致重大安全风险，包括数据泄露和财务损失。

尽管电子邮件服务提供商努力过滤垃圾邮件，但许多垃圾邮件仍然进入我们的收件箱。这就是机器学习发挥作用的地方。通过训练机器学习模型来识别垃圾邮件的特征，我们可以改进垃圾邮件过滤器，更好地保护用户免受不必要和潜在有害电子邮件的侵害。

在本实验中，将探讨如何使用机器学习来解决垃圾邮件问题。本次实验将实施几种机器学习算法，评估其性能，并讨论它们在垃圾邮件检测的实际环境中的影响。

# 实验背景

在数字通信领域，准确将电子邮件归类为“spam”（垃圾邮件）或“ham”（普通邮件）的分类任务至关重要。垃圾邮件不仅会扰乱我们的收件箱，使我们更难找到相关邮件，而且还会带来重大的安全风险。因此，该实验的目标是开发一种可以有效区分垃圾邮件和普通邮件的机器学习模型。

为了实现这一点，将利用一个包含大量电子邮件示例的数据集，每个示例都被标记为“spam”或“ham”。该数据集是训练我们的机器学习模型的基础。它为模型提供了垃圾邮件和普通邮件的示例，使其能够学习与每个类别相关的区别性特征和模式。

然而，这项任务并不像看起来那么简单。电子邮件由文本数据组成，文本数据是非结构化的，内容、风格和格式可能有很大差异。此外，垃圾邮件发送者经常使用复杂的策略来逃避检测，例如改变垃圾邮件相关单词的拼写或插入良性内容。因此，该项目的一个关键部分将涉及预处理和将文本数据转换为我们的机器学习模型可以理解和学习的格式。

此外，不仅要创建一个可以对电子邮件进行分类的模型，而且还要创建一个可以高度准确分类的模型。模型的有效性将根据其正确分类测试集中未见过的电子邮件的能力来评估。实现高准确率的模型将确保合法电子邮件不会被错误地标记为垃圾邮件，同时还能捕获最大数量的垃圾邮件。

总而言之，该项目提供了一个将机器学习技术应用于实际问题的激动人心的机会。从该项目中获得的见解可能会增强垃圾邮件过滤器的有效性，从而改善全球用户的电子邮件体验。

# 实验思路

我们将使用分类任务算法来决解提出的垃圾邮箱检测问题，因为我们要预测一个分类值：一封电子邮件是 “垃圾邮件” 还是 “非垃圾邮件”。具体来说，在实验中将使用支持向量机 (SVC) 算法作为本次实验的模型。

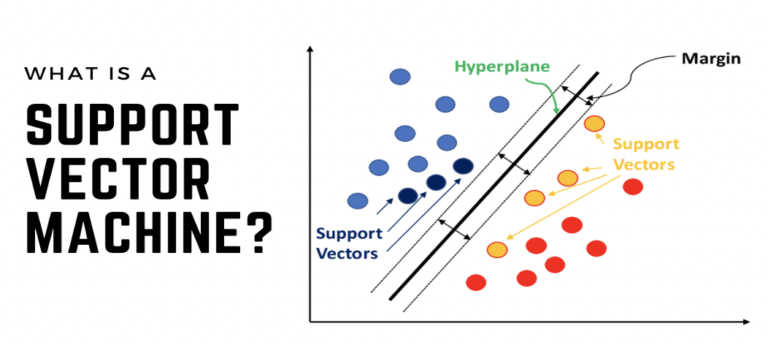
SVC是支持向量机（SVM）的一个具体应用形式，专门用于分类问题。在此实验中，我们将数据集中的每个数据项绘制在 N 维空间中（其中 N 是您拥有的特征数），每个特征的值是特定坐标的值。然后，通过找到能够很好地区分两个类的超平面来执行分类，如图3.1。

图 3.1 支持向量机（SVM）

SVC 算法特别具有优势，因为它能够处理高维数据，当维数大于样本数时，它非常有效。此外，它用途广泛，因为可以为决策函数指定不同的核函数。

使用 Python 中的 sci-kit-learn 库来实现该算法，该库为数据分析和建模提供了简单而有效的工具。它包含各种机器学习算法，包括 SVC，并允许轻松实现和测试这些模型。

在本实验设计中，将在一组电子邮件训练集上训练 SVC 模型，每封电子邮件都标记为“spam”或“ham”。训练完成后，将在另一组电子邮件测试集上测试该模型，并根据其正确分类电子邮件的准确性评估其性能。

# 实验实施

**4.1 数据集**

构成我们实验主干的数据集由两个不同的列组成： “Category” 和 “Message” 。 “Category” 列在我们的数据集中起着至关重要的作用。它将每封电子邮件归类为 “spam” (垃圾邮件) 或 “ham” （非垃圾邮件），为我们的机器学习模型提供学习预测的基本标签。另一方面，“Message”列包含电子邮件消息的实际内容。我们的模型将分析和学习这些原始文本数据，以进行预测。

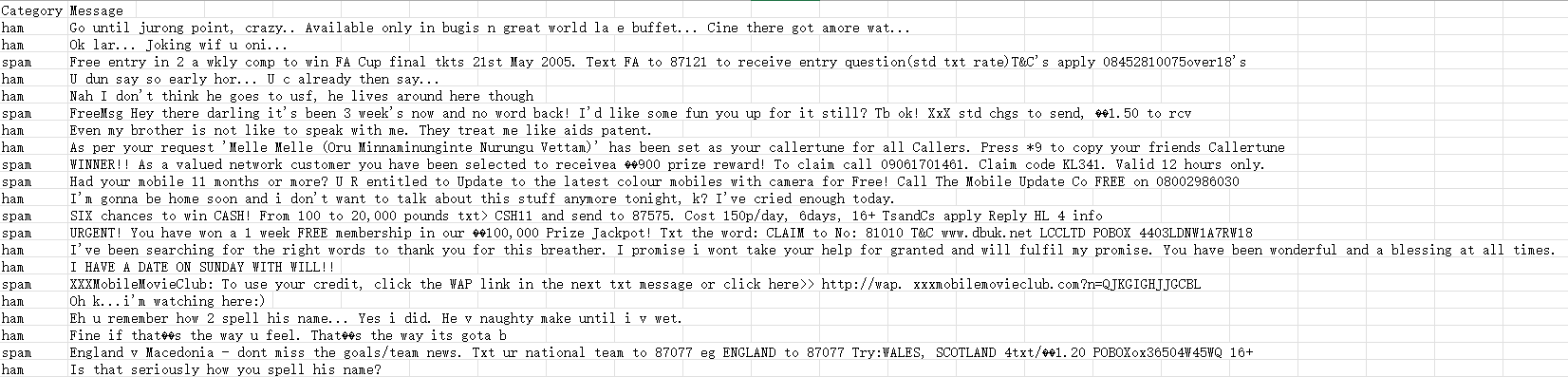
该数据集包含 5572 条的电子邮件记录。此数据集将作为我们机器学习模型的训练集的提供坚实的基础，使其能够学习区分垃圾邮件和非垃圾邮件的各种模式和特征，具体数据集如图4.1。

图 4.1 mail\_data.csv 数据集

数据集可以在此[网页](https://drive.google.com/drive/folders/1bRuM2RJ3CGD5_PtnIr7jVbC5kOJW5o6x)下载。

**4.2 模型实现**

对于模型的实现，将其分为3个步骤：数据预处理、训练和评估。

**4.2.1 数据预处理**

这是机器学习流程中的第一个最关键的步骤之一。在此阶段，为模型准备数据。对于垃圾邮件检测问题，这涉及 “Message” 列中的原始文本数据转换为机器学习模型可以理解的数字格式。可以使用标记化、删除Stop word和 TF-IDF 矢量化等技术。此外，还需要将 “Category” 列转换为二进制标签，例如将 “spam” 转换为 1， “ham” 转换为 0。

**步骤1：导入数据集**

数据预处理流程的第一步是导入数据集。使用Python 中的pandas库，pandas为数据分析和操作提供了强大的数据结构。

*#Import libraries*

import numpy as np

import pandas as pd

*#Import Dataset*

df = pd.read\_csv("./mail\_data.csv")

df.shape

**步骤2：定义特征和目标变量**

加载数据集后，定义特征和目标变量。在本实验中，特征 X 是“Message”列表，包含电子邮件的文本。目标变量 Y 是“Category”列，它表示每封电子邮件是“垃圾邮件”还是“普通邮件”。我们还将“类别”列转换为 0（表示“垃圾邮件”）和 1（表示“普通邮件” ）作为 Y 目标标签。

*# Select Only Data that is not null*

data = df.where(pd.notnull(df), '')

*# Change spam and ham cateogry to 0 and 1*

data.loc[data['Category'] == 'spam', 'Category',] = 0

data.loc[data['Category'] == 'ham', 'Category',] = 1

X = data['Message']

Y = data['Category']

**步骤3：拆分训练和测试集**

下一步是将数据集拆分为训练集和测试集。这是可以使用 sklearn.model\_selection 模块中的 train\_test\_split 函数完成的。本次实验选着分配 80% 的数据用于训练模型，其余 20% 用于测试其性能。另外，将 random\_state 设置为0，以确保测试后可重复地训练模型，并得到相同的正确率。

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

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, *test\_size*=0.2, *random\_state*=0)

**步骤4：特征提取**

拆分数据后，我们需要将文本数据转换为机器学习模型可以理解的数字格式。这个过程称为特征提取。我们使用 sklearn.feature\_extraction.text 模块中的 TfidfVectorizer 来实现此目的。我们将 min\_df 设置为 1，将 stop\_words 设置为“english”，将 lowercase 设置为 True。这会将我们的电子邮件文本转换为 TF-IDF 特征矩阵，从而有效地将文本数据转换为数字数据。

*# Transform Data to Feature Vector*

from sklearn.feature\_extraction.text import TfidfVectorizer

feature\_extraction = TfidfVectorizer(*min\_df* = 1, *stop\_words* = 'english', *lowercase* = True)

X\_train\_features = feature\_extraction.fit\_transform(X\_train)

X\_test\_features = feature\_extraction.transform(X\_test)

Y\_train = Y\_train.astype('int')

Y\_test = Y\_test.astype('int')

**4.2.2 模型训练**

一旦我们的数据经过预处理，我们就可以继续训练我们的模型。在这个阶段，我们将使用支持向量分类 (SVC) 算法从我们的训练数据中学习。该模型将学习将电子邮件消息的数字表示与其相应的标签（垃圾邮件或非垃圾邮件）相关联。我们将使用 Python 中的 sci-kit-learn 库来实现 SVM 算法。

**步骤1：模型声明**

sklearn 库提供对支持向量分类器 (SVC) 模型的支持。只需从 sklearn.svm 导入 SVC 模型，就可以完成模型声明。

*# Support Vector Classification Model*

from sklearn.svm import SVC

model = SVC()

**步骤2：训练模型**

从 sklearn.svm 导入支持向量分类器 (SVC) 模型后，下一步是使用我们的训练数据训练模型。这就是拟合函数发挥作用的地方。fit（）函数是sklearn提供的一种训练模型的方法。它根据给定的数据调整模型的参数，试图最小化模型预测与实际值之间的差异。本实验中，将两个参数传递给 fit 函数：X\_train\_features 和 Y\_train。

X\_train\_features 是我们训练集的特征矩阵。该矩阵中的每一行对应于一封电子邮件，每一列对应于一个特征（在我们的例子中，是特定单词的 TF-IDF 分数）。 fit 函数使用此矩阵来学习区分垃圾邮件和非垃圾邮件的模式。

另一方面，Y\_train 是一个包含训练集标签的向量。该向量中的每个条目要么是“垃圾邮件”，要么是“普通邮件”，对应于 X\_train\_features 中相同位置的电子邮件。拟合函数使用该向量来了解每封电子邮件的正确分类，从而指导其学习过程。

model.fit(X\_train\_features, Y\_train)

**步骤3：预测**

使用fit()函数训练机器学习模型后，可以使用predict()函数对新的、未见过的数据(X\_test\_features)进行预测。在我们的垃圾邮件检测任务中，这意味着将电子邮件分类为“垃圾邮件”或“正常邮件”。predict()函数将特征矩阵作为输入，其中每一行对应于一封电子邮件，每一列对应于一个特征。

Y\_predict\_testing = model.predict(X\_test\_features)

**4.2.3 模型的评估**

训练完模型后，最后一步是评估其性能。将模型对训练期间未见过的一组电子邮件测试集进行预测得到（Y\_predict\_testing）。然后，可以将这些预测与实际标签(Y\_test)进行比较，以确定模型的准确性。还可以使用诸如Precision、Recal和 F1 分数等指标进行更全面的评估。

from sklearn import metrics

*# Calculate metrics*

accuracy = metrics.accuracy\_score(Y\_test, Y\_predict\_testing)

precision = metrics.precision\_score(Y\_test, Y\_predict\_testing)

recall = metrics.recall\_score(Y\_test, Y\_predict\_testing)

f1\_score = metrics.f1\_score(Y\_test, Y\_predict\_testing)

print(*f*"Accuracy: {accuracy}")

print(*f*"Precision: {precision}")

print(*f*"Recall: {recall}")

print(*f*"F1 Score: {f1\_score}")

# 实验结果与讨论

经过训练和测试后，接下来的工作是展示 SVC 模型的性能,并讨论SVC模型在测试集上的表现。

**5.1 模型准确率**

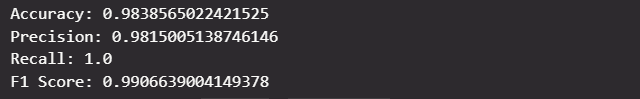
本次实验， SVC 模型的评估指标结果如图5.1。

图 5.1 SVC模型的评估指标

可以根据上述指标分析支持向量分类器（SVC）模型的性能：

* **Accuracy(0.9839)** ：这是正确预测总数与预测总数的比率。准确率 0.9839 意味着您的模型正确预测了测试集中约 98.39% 的电子邮件类别（垃圾邮件或非垃圾邮件）。这是一个很高的准确率，表明您的模型表现良好。
* **Precision(0.9815)** ：这是真实阳性预测 (正确分类为垃圾邮件的垃圾邮件) 与阳性预测总数 (所有电子邮件被分类为垃圾邮件) 的比率。准确率为 0.9815 意味着您的模型分类为垃圾邮件的电子邮件中约 98.15% 实际上是垃圾邮件。这表明您的模型在预测电子邮件是垃圾邮件时非常可靠。
* **Recall（1.0）** ：这是真实阳性预测与实际阳性总数（所有实际垃圾邮件）的比率。召回率为 1.0 表示您的模型正确识别了测试集中的所有垃圾邮件。这是一个出色的召回率，表明您的模型非常擅长检测垃圾邮件。
* **F1 Score(0.9907)** ：F1 分数是精确度和召回率的调和平均值，在这两个指标之间取得平衡。0.9907 的 F1 分数非常高，表明您的模型在精确度和召回率之间取得了良好的平衡。

总之， SVC模型在垃圾邮件分类任务上表现非常出色。它可以准确地对电子邮件进行分类，可靠地识别垃圾邮件，并且不会遗漏任何实际的垃圾邮件。高 F1 分数表明准确率和Recall之间取得了良好的平衡。

**5.2模型比较​**

除了分析SVC模型的表现之外，还将SVC 模型与其它流行的机器学习模型进行了比较,如表5.1。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 模型 | Accuracy | Precision | Recall | F1 Score |
| SVC | 0.9839 | 0.9815 | 1.0 | 0.9907 |
| Logistic Regression | 0.9668 | 0.9636 | 0.9989 | 0.9809 |
| Naïve Bayes | 0.9757 | 0.9725 | 1.0 | 0.9860 |
| Decision Tree | 0.9721 | 0.9733 | 0.9947 | 0.9839 |
| Random Forest | 0.9820 | 0.9804 | 0.9989 | 0.9896 |
| KNN | 0.9031 | 0.8984 | 1.0 | 0.9464 |

表 5.1 模型评价指标比交表

总之，虽然所有模型在垃圾邮件分类任务上都表现良好，但 SVM 模型脱颖而出，成为本次实验的最有效模型。然而，值得注意的是，这些结果是基于用于该实验的特定数据集。用于不同的数据集或现实世界数据时，这些模型的性能可能会有所不同。因此，测试多个模型并选择最适合当前特定任务和数据集的模型始终是一个好主意。

# 实验结论

本次实验证明了机器学习算法在处理此类任务方面的潜力和有效性。支持向量分类器 (SVC) 以及其他流行模型都表现出较高的准确率，表明它们能够有效区分“垃圾邮件”和“普通”电子邮件。尤其是 SVC，它表现出了强大的性能，进一步证明了该算法在处理文本分类问题方面的多功能性和强大功能。准确率、召回率和 F1 分数都非常高，这表明该模型不仅准确，而且预测可靠。

但需要注意的是，本次实验使用的数据集是专门为本次实验而准备的测试数据集。虽然模型在这个数据集上表现良好，但现实世界的情况可能会出现更复杂、更多样化的垃圾邮件实例。因此，虽然实验结果效果很好，但模型在现实世界数据上的表现可能会有所不同。在未来的工作中，在真实数据上测试这些模型将会大有裨益，这将提供对其性能和稳健性的更全面了解。此外，探索其他机器学习算法和调整超参数可能会带来更好的结果。

总之，这个实验强调了机器学习在增强垃圾邮件检测方面的潜力，为更安全、更高效的电子邮件通信铺平了道路。