SPAM EMAIL DETECTION
PROJECT REPORT

Submitted By

NIKHIL V BHASKAR

INTRODUCTION

Spam emails are a prevalent issue in digital communication, posing risks such as phishing, fraud, and information theft. Detecting and filtering spam emails is essential to safeguard users from these threats. Traditional rule-based spam filters have limitations in adapting to evolving spam patterns, making machine learning (ML) a more effective solution.

Machine learning enables the development of intelligent spam email detection systems that can analyse and classify emails based on their content. Various ML algorithms, including logistic regression, multinomial Naive Bayes, and random forest classifiers, are commonly used to build robust spam classifiers. These models leverage features such as word frequencies, sender details, and email structure to differentiate spam from legitimate emails.

The objective of this project is to create a machine learning-based spam detection system that accurately identifies spam emails while minimizing false positives. By comparing the performance of different algorithms, the project aims to highlight the strengths and trade-offs of each approach, ultimately delivering an efficient and reliable spam detection solution.

EXPLORATORY DATA ANALYSIS (EDA)

Exploratory Data Analysis (EDA) is a crucial step in the data analysis process, aimed at summarizing the main characteristics of a dataset, oftenwith visual methods.

EDA is iterative and involves going back and forth between different steps to refine the understanding of the data. The goal is to make sense of the data, detect essential features, and generate questions or hypotheses for further analysis.

MACHINE LEARNING

Machine learning (ML) is a subset of artificial intelligence (AI) that focuses on building systems to learn from and make data-based decisions. Instead of being explicitly programmed to perform a task, these systems use algorithms to identify patterns and make predictions or decisions. Here are the key components and types of machine learning:

Types of machine learning:

- Supervised Learning: The model is trained on labelled data, where

  the input data and the corresponding output are provided. Forexample, regression (predicting continuous values) and classification (categorizing data into discrete classes) are used. Its applications are Spam detection, image classification, and medical diagnosis.

- Unsupervised Learning: The model is trained on unlabelled data and

  must find patterns and relationships within the data. For example, Clustering (grouping similar data points) and association (finding rules that describe large portions of data). Its applications are Customer segmentation, market basket analysis, and anomaly detection.

- Semi-supervised Learning: It combines a small amount of labelled

  data with many unlabeled data during training. Its applications are situations where acquiring labeled data is expensive or time- consuming, such as medical image analysis.

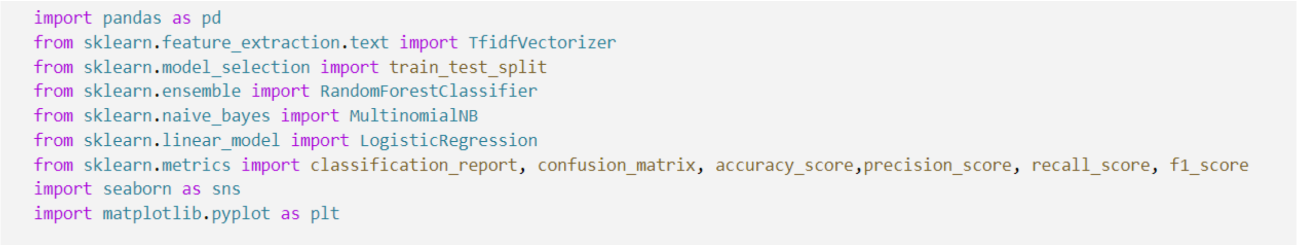
SPAM EMAIL DETECTION USING MACHINE LEARNING

Spam email detection using machine learning involves using computer programs to determine whether an email is spam or not. These programs analyze features like email content to make their prediction. They learn from numerous examples of emails labeled as spam or non-spam to improve their accuracy over time. By identifying patterns in the data, these programs can adapt to new and evolving spam tactics, providing a reliable solution to email filtering.

DATASET

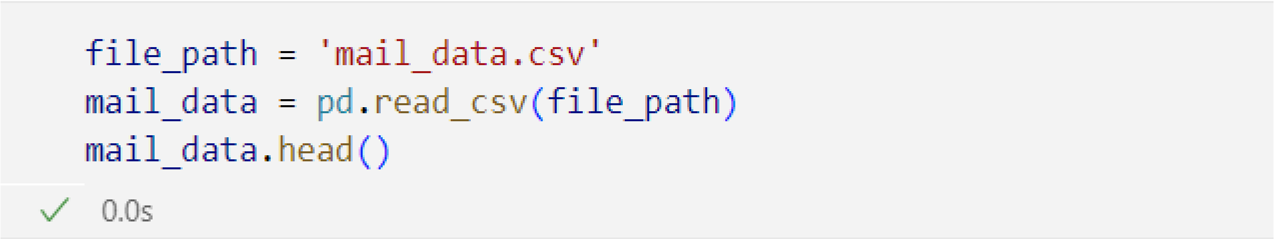
The dataset used for the Spam Email Detection project was sourced from Kaggle. The dataset consists of 5,572 observations (rows) and 2 variables (columns). The dataset includes one feature, Message, which contains the text content of the emails, and a target variable, Category, which indicates whether the email is spam (spam) or not (ham). This dataset is particularly useful for testing machine learning algorithms for binary classification tasks. By analysing the text data in the Message column, machine learning models can learn to classify emails accurately, making it a robust resource for developing and evaluating spam detection systems.

Importing libraries



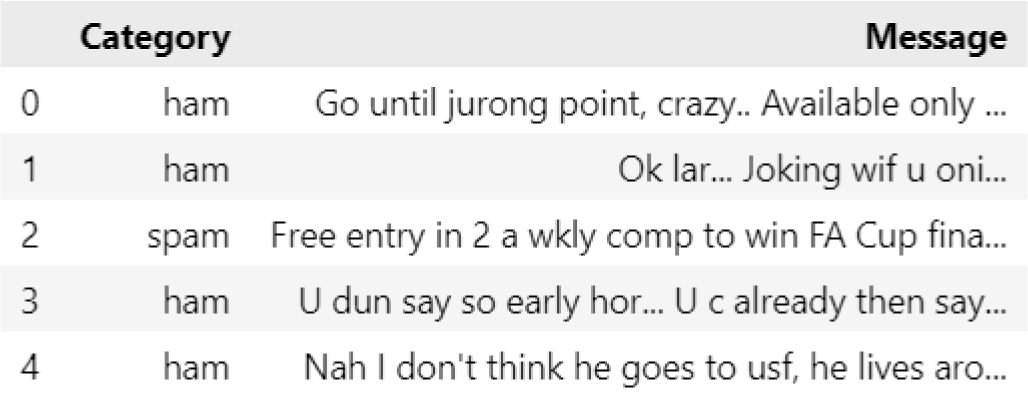
```
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score,precision_score, recall_score, f1_score
import seaborn as sns
import matplotlib.pyplot as plt
```

We start by reading the training dataset, which is in CSV format:



```
file_path = 'mail_data.csv'
mail_data = pd.read_csv(file_path)
mail_data.head()
```

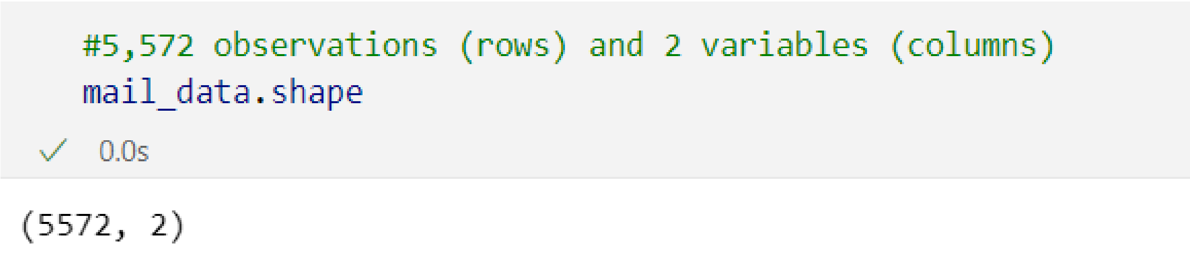
✓ 0.0s



	Category	Message
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...

Exploratory Data Analysis (EDA)

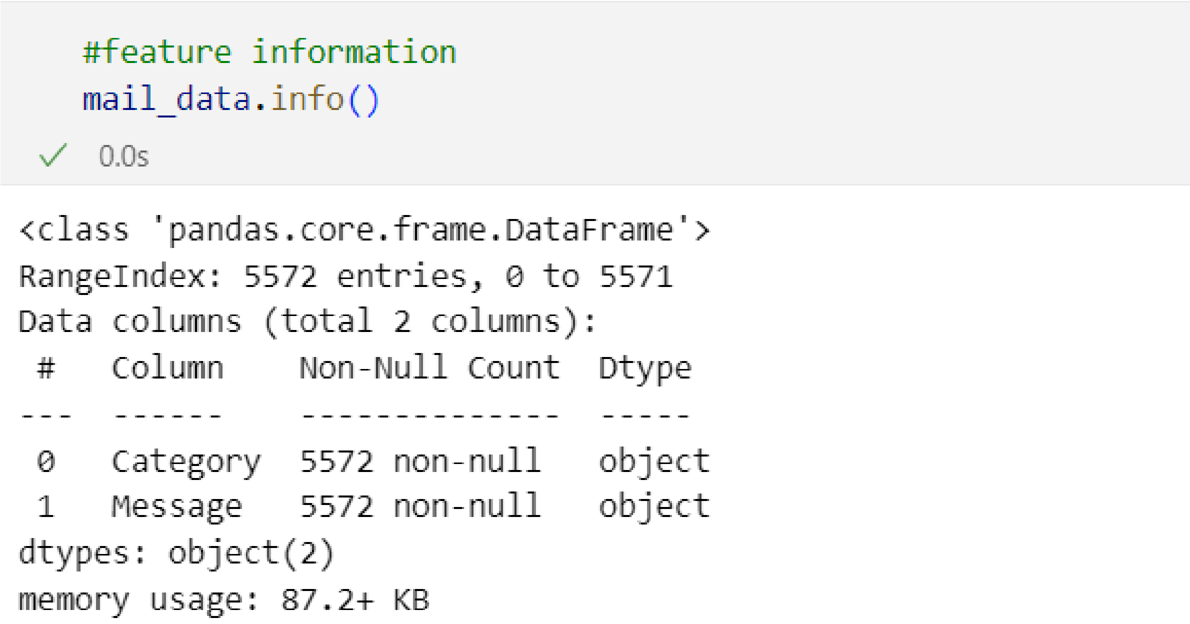
To get more information about the dataset:



```
#5,572 observations (rows) and 2 variables (columns)
mail_data.shape
```

✓ 0.0s

```
(5572, 2)
```

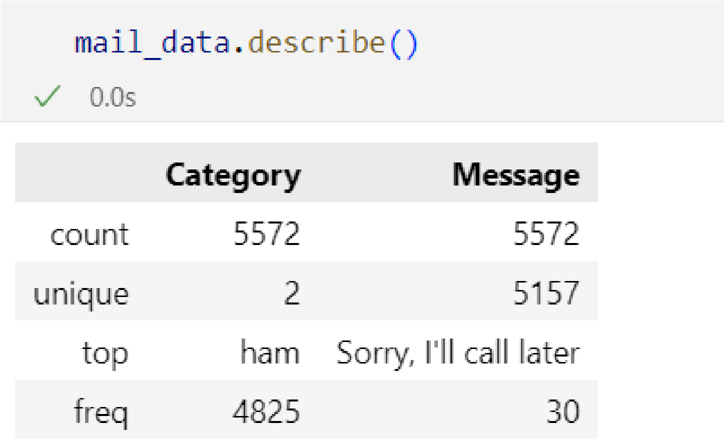


```
#feature information
mail_data.info()
```

✓ 0.0s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 2 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   Category  5572 non-null   object
 1   Message   5572 non-null   object
dtypes: object(2)
memory usage: 87.2+ KB
```

To understand the dataset:

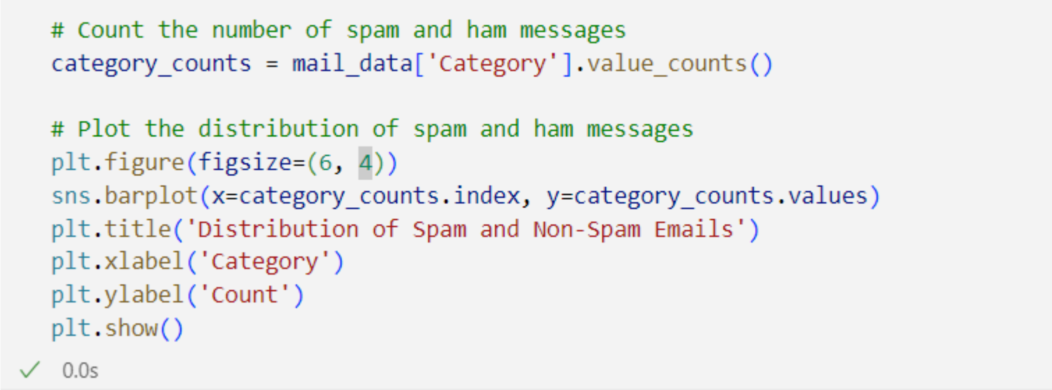


```
mail_data.describe()
```

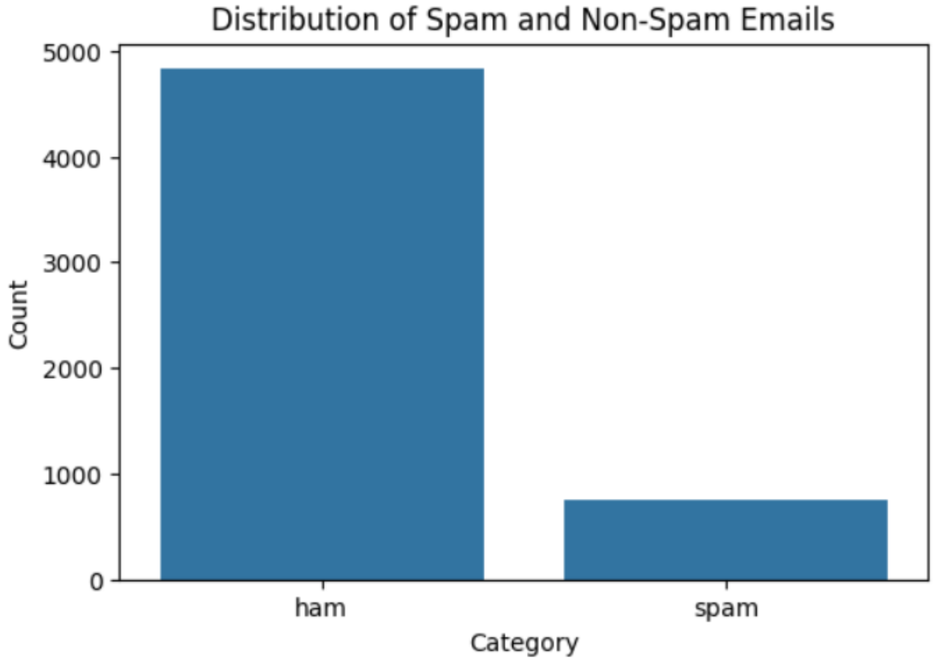
✓ 0.0s

	Category	Message
count	5572	5572
unique	2	5157
top	ham	Sorry, I'll call later
freq	4825	30

Data Visualization



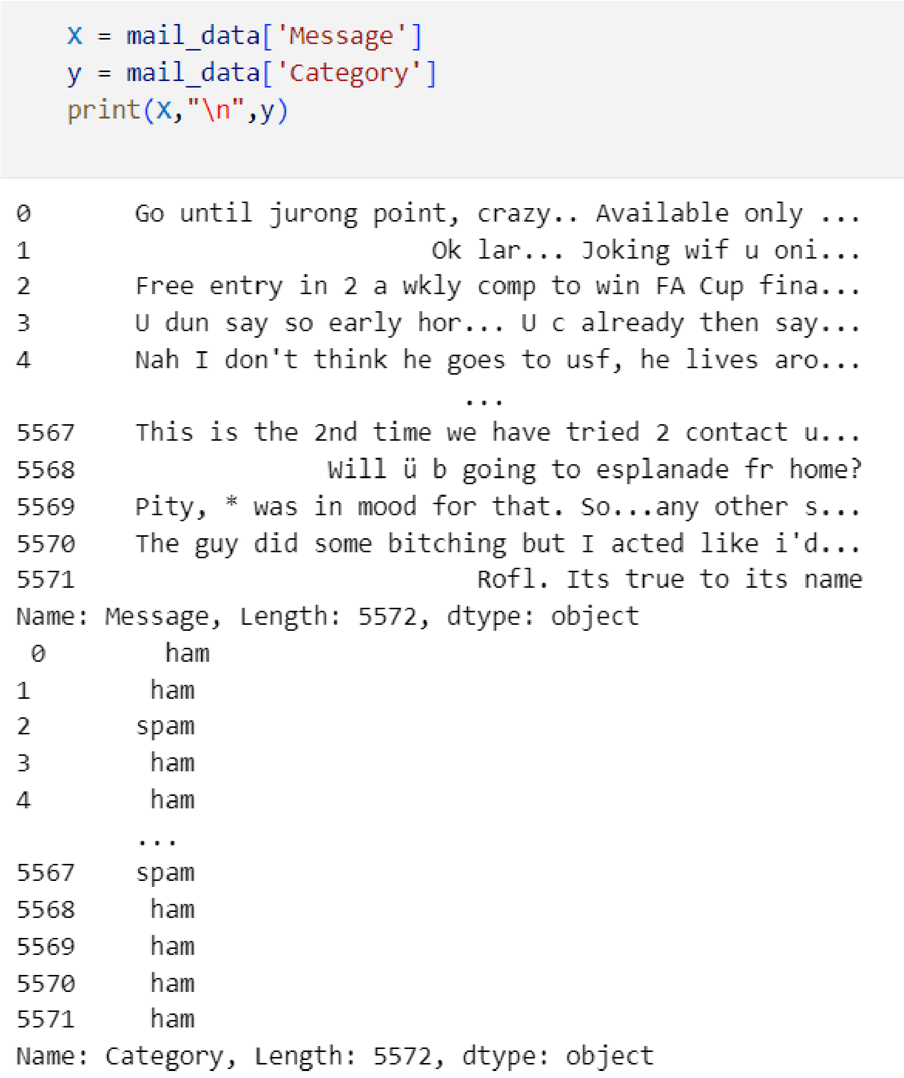
```
# Count the number of spam and ham messages
category_counts = mail_data['Category'].value_counts()

# Plot the distribution of spam and ham messages
plt.figure(figsize=(6, 4))
sns.barplot(x=category_counts.index, y=category_counts.values)
plt.title('Distribution of Spam and Non-Spam Emails')
plt.xlabel('Category')
plt.ylabel('Count')
plt.show()
```



The above visualization indicates that our training dataset is imbalanced.The number of ham mail is almost 4 times more than that of spam mails.

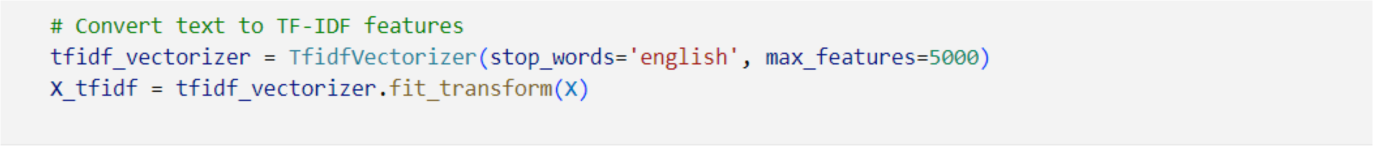
BUILDING MODEL

First, we need the split the data into Message(X) and Category (y). Then we have to convert the test message into TF-IDF features. Then we split the dataset into training and testing sets using the train_test_split function.

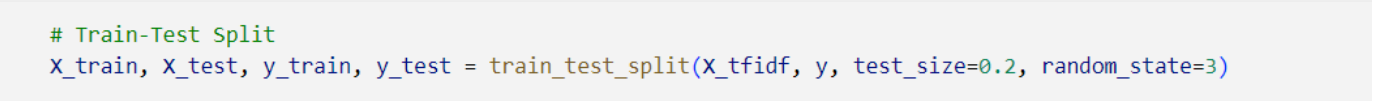


```
X = mail_data['Message']
y = mail_data['Category']
print(X,"\n",y)
```

```
0       Go until jurong point, crazy.. Available only ...
1                           Ok lar... Joking wif u oni...
2       Free entry in 2 a wkly comp to win FA Cup fina...
3       U dun say so early hor... U c already then say...
4       Nah I don't think he goes to usf, he lives aro...
                              ...
5567    This is the 2nd time we have tried 2 contact u...
5568                 Will ü b going to esplanade fr home?
5569    Pity, * was in mood for that. So...any other s...
5570    The guy did some bitching but I acted like i'd...
5571                           Rofl. Its true to its name
Name: Message, Length: 5572, dtype: object
 0        ham
1        ham
2       spam
3        ham
4        ham
        ...
5567    spam
5568     ham
5569     ham
5570     ham
5571     ham
Name: Category, Length: 5572, dtype: object
```

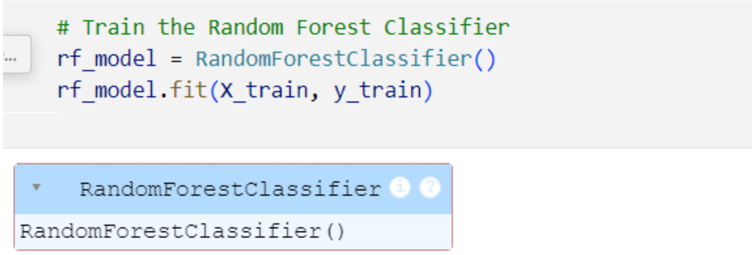


```python
# Convert text to TF-IDF features
tfidf_vectorizer = TfidfVectorizer(stop_words='english', max_features=5000)
X_tfidf = tfidf_vectorizer.fit_transform(X)
```



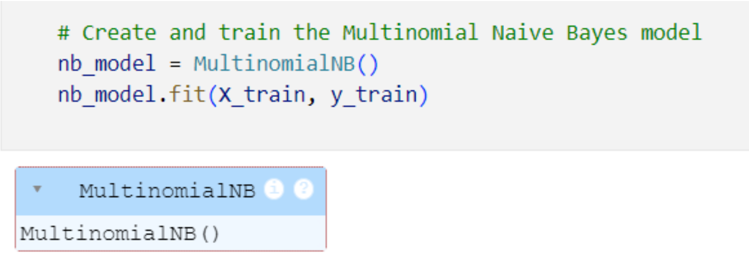
```python
# Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(X_tfidf, y, test_size=0.2, random_state=3)
```

Tested it with 3 models:



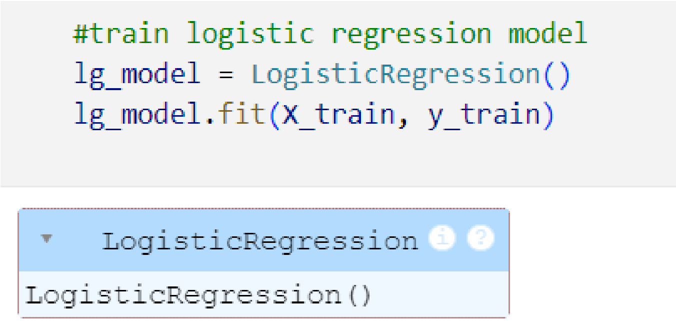
```python
# Train the Random Forest Classifier
rf_model = RandomForestClassifier()
rf_model.fit(X_train, y_train)
```

```
RandomForestClassifier
RandomForestClassifier()
```



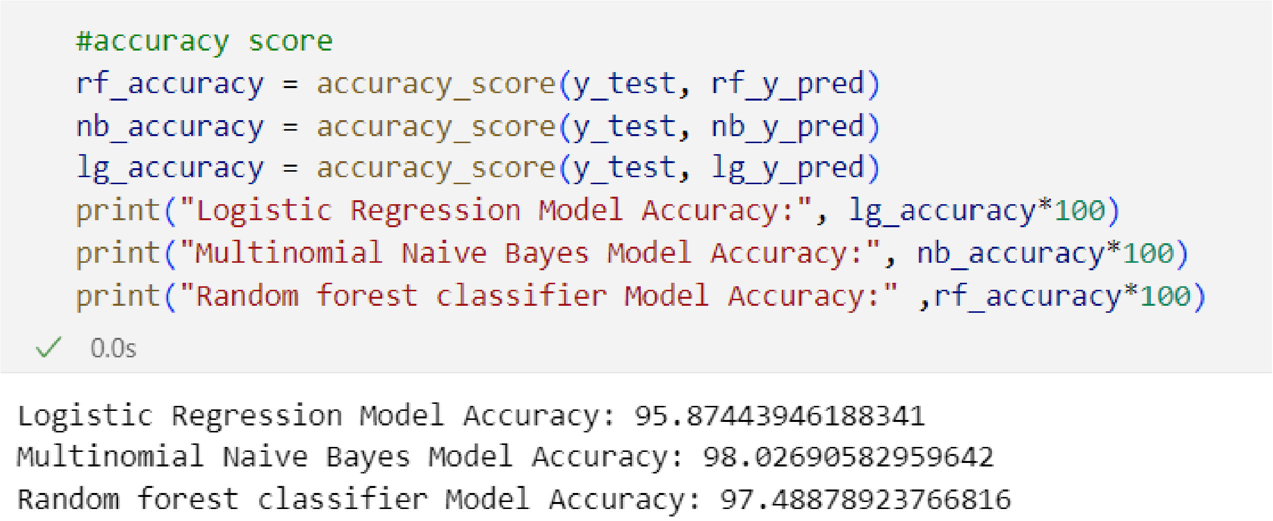
```python
# Create and train the Multinomial Naive Bayes model
nb_model = MultinomialNB()
nb_model.fit(X_train, y_train)
```

```
MultinomialNB
MultinomialNB()
```



```python
#train logistic regression model
lg_model = LogisticRegression()
lg_model.fit(X_train, y_train)
```

```
LogisticRegression
LogisticRegression()
```



```
#accuracy score
rf_accuracy = accuracy_score(y_test, rf_y_pred)
nb_accuracy = accuracy_score(y_test, nb_y_pred)
lg_accuracy = accuracy_score(y_test, lg_y_pred)
print("Logistic Regression Model Accuracy:", lg_accuracy*100)
print("Multinomial Naive Bayes Model Accuracy:", nb_accuracy*100)
print("Random forest classifier Model Accuracy:" ,rf_accuracy*100)
```

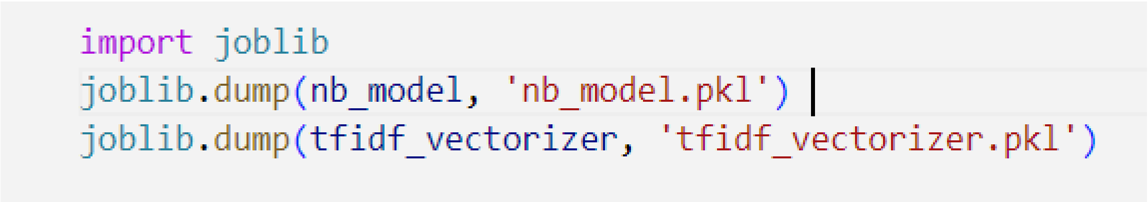
✓ 0.0s

```
Logistic Regression Model Accuracy: 95.87443946188341
Multinomial Naive Bayes Model Accuracy: 98.02690582959642
Random forest classifier Model Accuracy: 97.48878923766816
```

Among the models tested, the Multinomial Naïve Bayes model performed the best with an accuracy 98.02%. This indicates that the model correctly classifiedaround 98% of all cases in the test set.

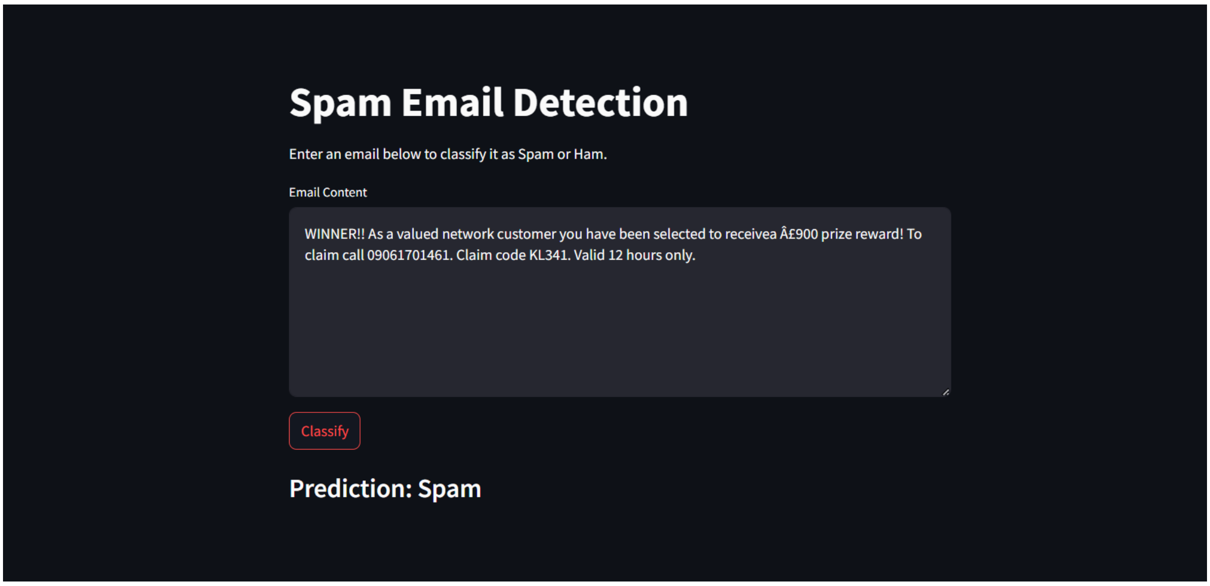
Saving Model – Multinomial Naïve Bayes

Also saving the tdidf vectorizer



```
import joblib
joblib.dump(nb_model, 'nb_model.pkl')
joblib.dump(tfidf_vectorizer, 'tfidf_vectorizer.pkl')
```

Deployment of model using Streamlit



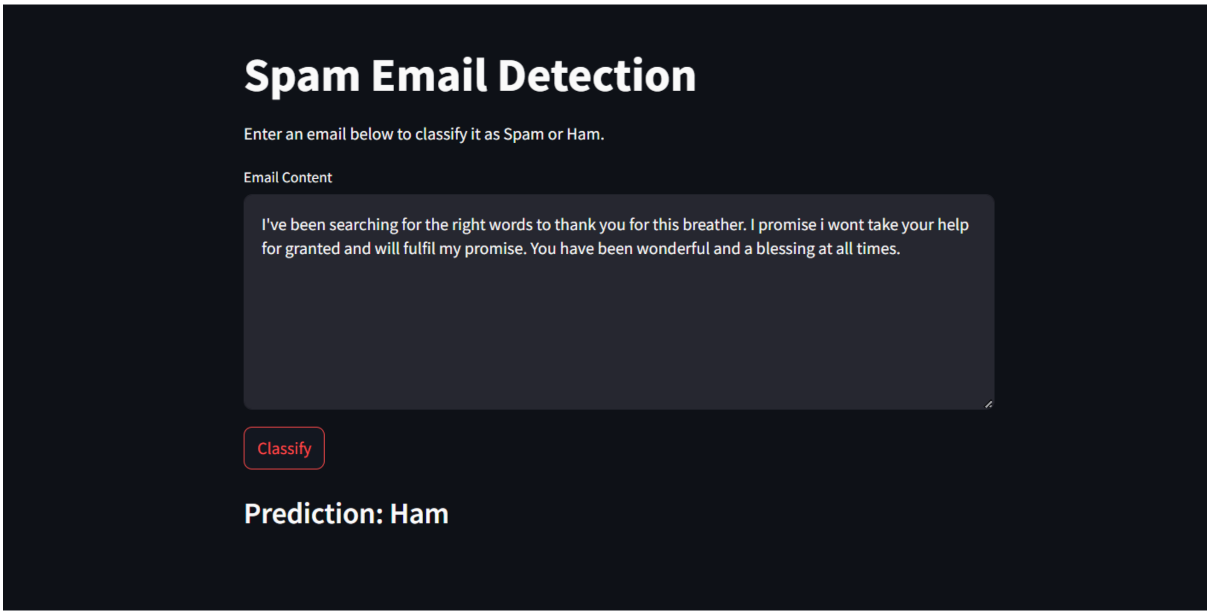
Spam Email Detection

Enter an email below to classify it as Spam or Ham.

Email Content

WINNER!! As a valued network customer you have been selected to receivea Â£900 prize reward! To claim call 09061701461. Claim code KL341. Valid 12 hours only.

Prediction: Spam



Spam Email Detection

Enter an email below to classify it as Spam or Ham.

Email Content

I've been searching for the right words to thank you for this breather. I promise i wont take your help for granted and will fulfil my promise. You have been wonderful and a blessing at all times.

Prediction: Ham

SUMMARY

The analysis utilized a Multinomial Naive Bayes classifier to classify emails as spam or non-spam (ham) using their textual content. The dataset contained 5,572 email records, split into spam and non-spam categories. After training and evaluation, the model achieved an impressive accuracy of 98.03%, showcasing its effectiveness in distinguishing between the two classes.

The model exhibited a recall of 1.00 for the ham class (non-spam), meaning it correctly identified all legitimate emails without any false negatives. For the spam class, the recall was 0.86, indicating that 86% of actual spam emails were correctly classified as spam, though a small percentage of spam emails went undetected.

The combination of 0.98 precision for ham, 1.00 precision for spam, and the recall metrics underscores the model's strong ability to accurately classify emails with minimal errors. These results demonstrate that the Multinomial Naive Bayes model is highly effective for spam detection, striking a balance between catching spam emails and avoiding misclassification of legitimate emails. This makes it a reliable solution for real-world email filtering applications.