Automated Email Classification Using GenAI-Enhanced Models

Submitted by

Sanmitraa V R

727723EUCS207

III CSE – ‘C’

Computer Science and Engineering

ABSTRACT

With the rapid growth of digital communication, organizations receive a large number of emails every day. Manually sorting these emails into categories such as Spam, Promotions, Support, and Personal is time-consuming and inefficient. Traditional rule-based email filtering systems are not effective in handling evolving language patterns, while conventional machine learning models require extensive tuning and feature engineering.

This project presents an **Automated Email Classification System** using **Natural Language Processing (NLP)** and **GenAI-enhanced models**. A real-world email dataset collected from Kaggle is used for training and testing. The email text is preprocessed and converted into numerical features using **TF-IDF vectorization** and embedding-based techniques. Classification models are trained to categorize emails into four classes: Spam, Promotions, Support, and Personal. The system performance is evaluated using metrics such as **confusion matrix, precision, recall, and F1-score**. The proposed system improves classification accuracy and supports scalable, real-time email automation.

1. INTRODUCTION

Email is one of the most widely used communication tools in enterprises and organizations. Every day, users receive a large number of emails that include advertisements, spam messages, customer support queries, and personal communication. Managing these emails manually reduces productivity and increases the chance of missing important messages.

Automated email classification helps in organizing emails into meaningful categories, making it easier for users to prioritize and respond efficiently. Recent advancements in **Natural Language Processing (NLP)** and **Generative Artificial Intelligence (GenAI)** have made it possible to build intelligent systems that understand the context and semantics of textual data. This project aims to utilize these technologies to develop an effective email classification system.

2. PROBLEM STATEMENT

An enterprise email system needs to automatically classify incoming emails into the following categories:

- Spam
- Promotions
- Support
- Personal

Rule-based email filtering techniques are inadequate for handling large-scale and dynamic email content. Traditional machine learning models struggle to adapt to new language patterns. Hence, an intelligent NLP-based system enhanced with GenAI techniques is required for accurate and scalable email classification.

3. OBJECTIVES OF THE PROJECT

The main objectives of this project are:

- To collect and prepare an email dataset for classification
- To preprocess and clean raw email text
- To convert text data into numerical representations using TF-IDF
- To apply embedding-based techniques for enhanced semantic understanding
- To train and evaluate multi-class classification models
- To visualize and analyze email data distributions
- To automate email classification for real-time use

4. SYSTEM REQUIREMENTS

4.1 Hardware Requirements

- Processor: Intel i3 or above
- RAM: Minimum 8 GB
- Storage: Minimum 20 GB

4.2 Software Requirements

- Python 3.x
- Pandas
- Scikit-learn
- Matplotlib
- NLTK
- Hugging Face Transformers
- fastText

5. DATASET DESCRIPTION

The dataset used in this project is collected from **Kaggle** and consists of real-world email messages. Initially, the dataset contains emails labeled as spam and non-spam (ham). To meet enterprise requirements, the non-spam emails are further classified into Promotions, Support, and Personal categories using keyword-based and semantic analysis.

Dataset Attributes

- **email_text**: Contains the subject and body of the email
- **category**: Represents the class label (Spam, Promotions, Support, Personal)

6. METHODOLOGY

6.1 Data Collection

The dataset is downloaded from Kaggle and loaded into the system using the Pandas library.

6.2 Data Preprocessing

Text preprocessing is performed to remove noise and improve feature extraction. The following steps are applied:

- Conversion of text to lowercase
- Removal of punctuation and numbers
- Removal of stopwords
- Token normalization

6.3 Feature Extraction

- **TF-IDF Vectorization** is used to convert text into numerical features based on word importance.
- **Embedding-based techniques** such as fastText and transformer models are used to capture semantic and contextual meaning of email text.

6.4 Classification

Machine learning classifiers are trained on the extracted features to classify emails into four categories.

7. SYSTEM ARCHITECTURE

The system follows a modular architecture consisting of:

1. Email Dataset Input
2. Text Preprocessing Module
3. Feature Extraction Module
4. Classification Model
5. Evaluation Module
6. Prediction Output

8. IMPLEMENTATION DETAILS

8.1 TF-IDF Vectorization

TF-IDF converts email text into vectors by considering both term frequency and inverse document frequency.

8.2 GenAI-Enhanced Embeddings

Transformer-based embeddings are used to generate contextual representations of emails, improving classification accuracy for complex language patterns.

9. EXPLORATORY DATA ANALYSIS AND VISUALIZATION

Exploratory Data Analysis (EDA) is performed to understand the characteristics of the dataset. The following visualizations are used:

- Email category distribution
- Email length distribution
- Most frequent words after preprocessing

These visualizations help in understanding class balance and text complexity.

10. MODEL TRAINING AND TESTING

The dataset is divided into training and testing sets. Classification models are trained using the training data and tested on unseen email data to evaluate performance.

11. PERFORMANCE EVALUATION

The performance of the system is evaluated using:

- Accuracy
- Precision

- Recall
- F1-Score
- Confusion Matrix

These metrics provide a detailed understanding of the model's effectiveness across multiple categories.

12. RESULTS AND DISCUSSION

The experimental results show that TF-IDF based models perform well for basic classification, while embedding-based and GenAI-enhanced models achieve higher accuracy due to better semantic understanding. The system successfully classifies emails into Spam, Promotions, Support, and Personal categories.

13. ADVANTAGES OF THE SYSTEM

- Reduces manual email sorting
- Improves productivity
- Scalable for enterprise use
- Handles evolving language patterns
- Supports real-time automation

14. APPLICATIONS

- Enterprise email management systems
- Customer support automation
- Spam filtering systems
- Marketing email analysis

15. CONCLUSION

This project successfully demonstrates an automated email classification system using NLP and GenAI-enhanced models. By combining traditional text vectorization techniques with modern embedding approaches, the system achieves accurate and scalable email categorization. The proposed approach is suitable for real-world enterprise deployment.

16. FUTURE ENHANCEMENTS

- Real-time email stream integration
- Online learning for continuous improvement
- Multi-language email classification
- Integration with enterprise email servers

Source Code:

```
# ---------- 1. Import Libraries ----------
import pandas as pd
import numpy as np
import re

import nltk
from nltk.corpus import stopwords

import matplotlib.pyplot as plt
from collections import Counter

from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
```

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, classification_report, f1_score

# ---------- 2. Download Stopwords ----------
nltk.download("stopwords")

# ---------- 3. Load Dataset ----------
df = pd.read_csv("spam_ham_dataset.csv")

# Drop unnecessary columns
df = df.drop(columns=["Unnamed: 0", "label_num"])

# Rename columns
df = df.rename(columns={
    "text": "email_text",
    "label": "category"
})

print("Dataset Loaded Successfully")
print(df.head())

# ---------- 4. Convert Spam/Ham into 4 Categories ----------
def map_category(text, label):
    text = text.lower()

    if label == "spam":
        return "Spam"
    elif any(word in text for word in ["offer", "sale", "discount", "free", "deal"]):
        return "Promotions"
```

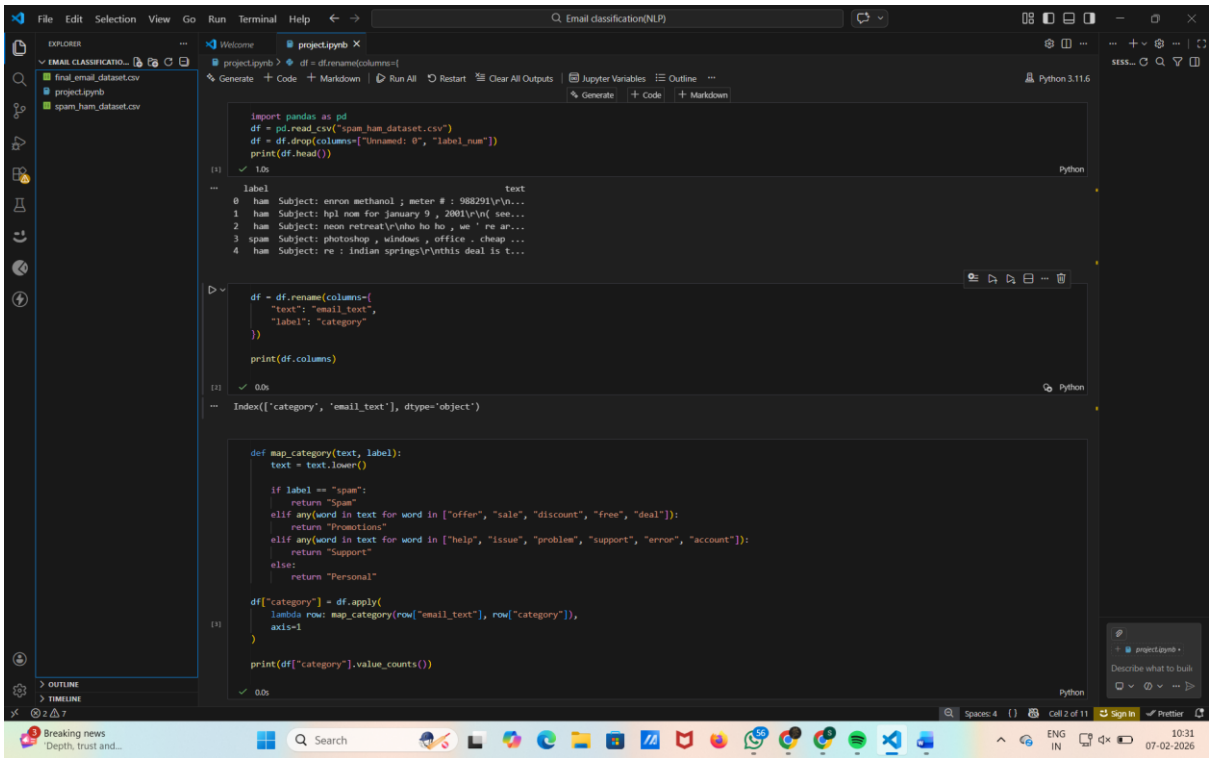
```
    elif any(word in text for word in ["help", "issue", "problem", "support", "account",
"error"]):
        return "Support"
    else:
        return "Personal"

df["category"] = df.apply(
    lambda row: map_category(row["email_text"], row["category"]),
    axis=1
)

print("\nCategory Distribution:")
print(df["category"].value_counts())

# ---------- 5. Text Preprocessing ----------
stop_words = set(stopwords.words("english"))

def clean_text(text):
    text = text.lower()
    text = re.sub(r"[^a-z\s]", "", text)
    words = text.split()
    words = [word for word in words if word not in stop_words]
    return " ".join(words)

df["clean_text"] = df["email_text"].apply(clean_text)

# ---------- 6. Visualization ----------

# Category Distribution
```

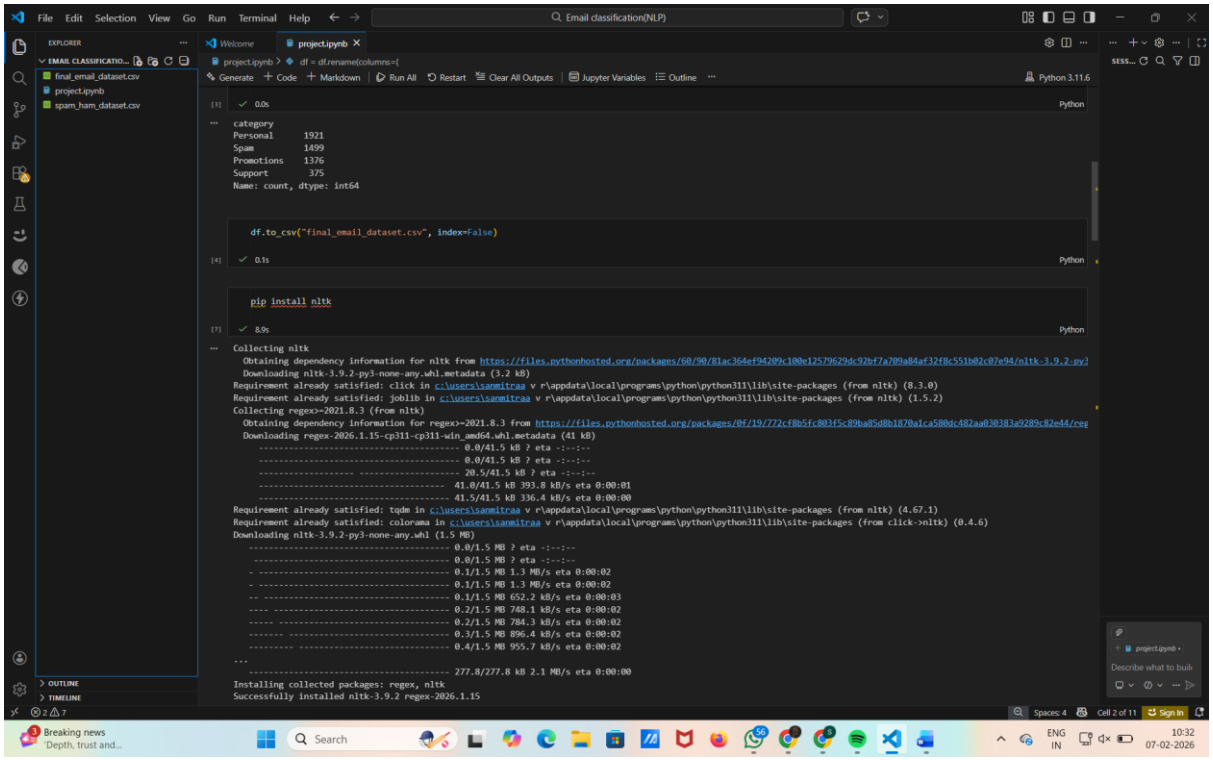
```
plt.figure()
df["category"].value_counts().plot(kind="bar")
plt.title("Email Category Distribution")
plt.xlabel("Category")
plt.ylabel("Number of Emails")
plt.show()

# Email Length Distribution
df["text_length"] = df["clean_text"].apply(lambda x: len(x.split()))

plt.figure()
plt.hist(df["text_length"], bins=30)
plt.title("Email Length Distribution")
plt.xlabel("Number of Words")
plt.ylabel("Frequency")
plt.show()

# Most Common Words
all_words = " ".join(df["clean_text"]).split()
common_words = Counter(all_words).most_common(10)
words, counts = zip(*common_words)

plt.figure()
plt.bar(words, counts)
plt.title("Top 10 Most Common Words")
plt.xlabel("Words")
plt.ylabel("Frequency")
plt.show()
```

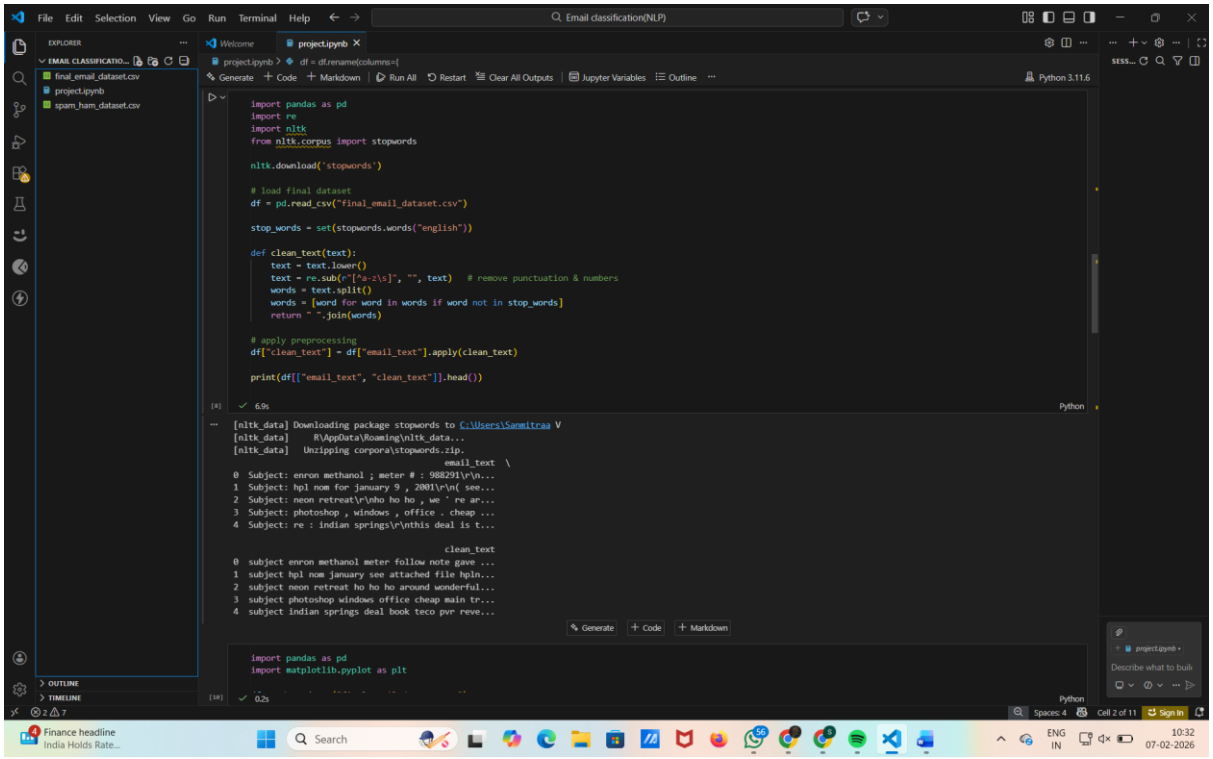
```
# ---------- 7. Train-Test Split ----------
X = df["clean_text"]
y = df["category"]

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)

# ---------- 8. TF-IDF Vectorization ----------
tfidf = TfidfVectorizer(max_features=5000)

X_train_tfidf = tfidf.fit_transform(X_train)
X_test_tfidf = tfidf.transform(X_test)

# ---------- 9. Train Model ----------
model = LogisticRegression(max_iter=1000)
model.fit(X_train_tfidf, y_train)

# ---------- 10. Prediction ----------
y_pred = model.predict(X_test_tfidf)

# ---------- 11. Evaluation ----------

print("\nConfusion Matrix:")
print(confusion_matrix(y_test, y_pred))

print("\nClassification Report:")
print(classification_report(y_test, y_pred))
```

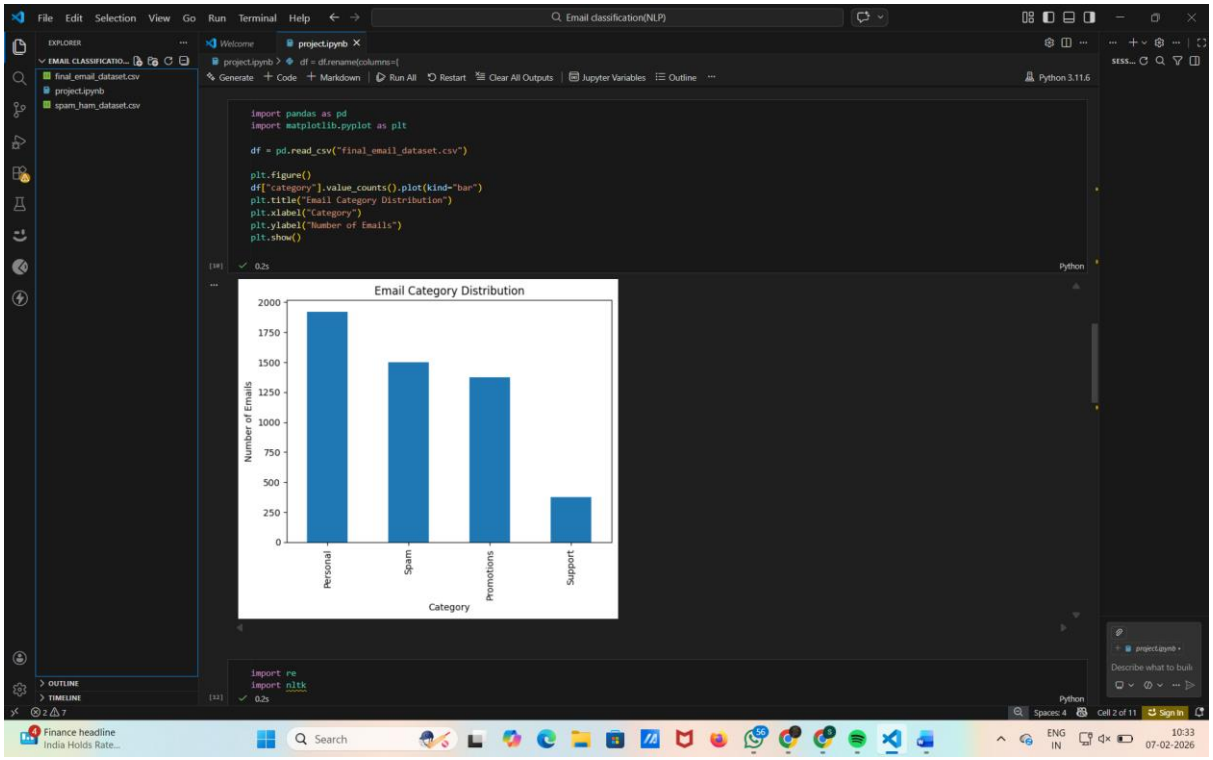
```
f1 = f1_score(y_test, y_pred, average="weighted")
print("Weighted F1 Score:", f1)

# ---------- 12. Real-Time Email Prediction ----------
def predict_email(email):
    email = clean_text(email)
    vector = tfidf.transform([email])
    prediction = model.predict(vector)
    return prediction[0]

# Example Test
sample_email = "Huge discount offer on electronics, limited time only!"
print("\nSample Email Prediction:")
print(predict_email(sample_email))
```

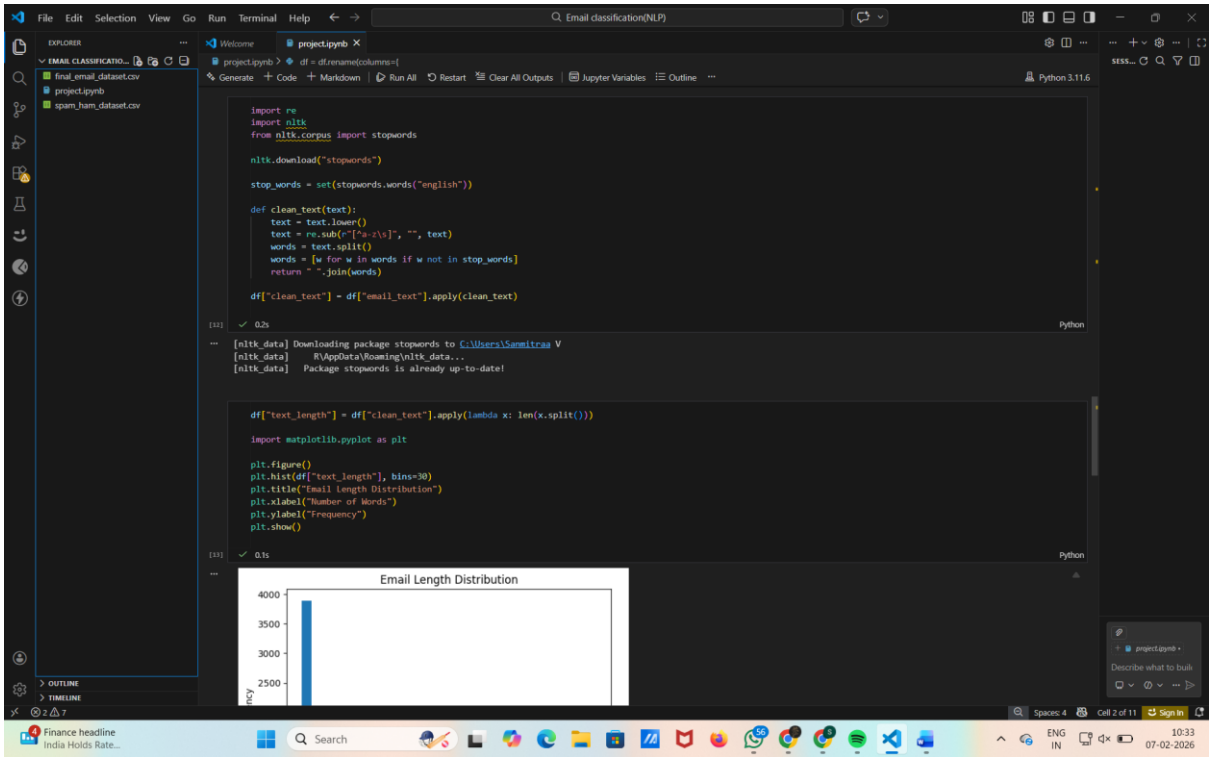
Screenshots:

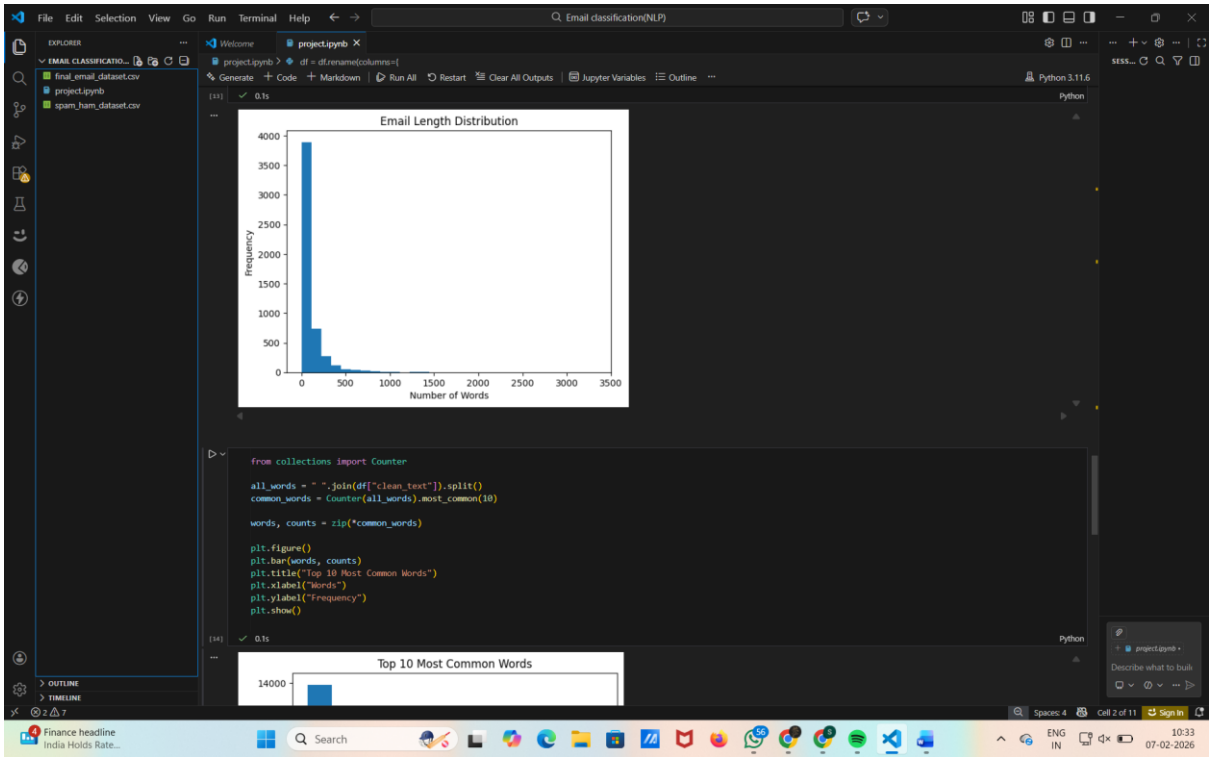


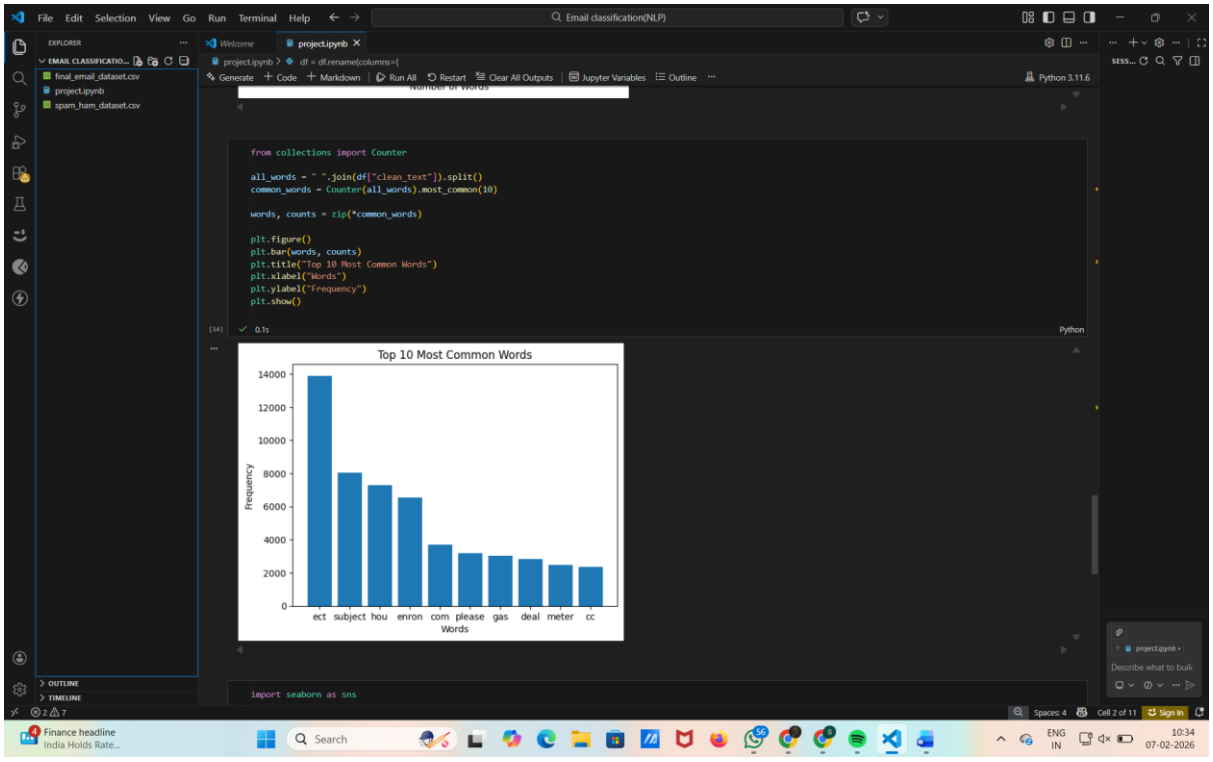

File Edit Selection View Go Run Terminal Help
Email classification(NLP)
EXPLORER
EMAIL CLASSIFICATIO...
final_email_dataset.csv
project.ipynb
spam_ham_dataset.csv
Welcome
project.ipynb
df = df.rename(columns={
Generate
Code
Markdown
Run All
Restart
Clear All Outputs
Jupyter Variables
Outline
Python 3.11.6
0.0s
Python
category
Personal 1921
Spam 1499
Promotions 1376
Support 375
Name: count, dtype: int64
df.to_csv("final_email_dataset.csv", index=False)
0.1s
Python
pip install nltk
8.9s
Python
Collecting nltk
Obtaining dependency information for nltk from https://files.pythonhosted.org/packages/60/90/81ac364ef94209c100e12579629dc92bf7a709a84af32f8c551b02c07e94/nltk-3.9.2-py3
Downloading nltk-3.9.2-py3-none-any.whl.metadata (3.2 kB)
Requirement already satisfied: click in c:\users\sanmitraa v r\appdata\local\programs\python\python311\lib\site-packages (from nltk) (8.3.0)
Requirement already satisfied: joblib in c:\users\sanmitraa v r\appdata\local\programs\python\python311\lib\site-packages (from nltk) (1.5.2)
Collecting regex>=2021.8.3 (from nltk)
Obtaining dependency information for regex>=2021.8.3 from https://files.pythonhosted.org/packages/0f/19/772cf8b5fc803f5c89ba85d8b1870a1ca580dc482aa030383a9289c82e44/reg
Downloading regex-2026.1.15-cp311-cp311-win_amd64.whl.metadata (41 kB)
0.0/41.5 kB ? eta -:--:--
0.0/41.5 kB ? eta -:--:--
20.5/41.5 kB ? eta -:--:--
41.0/41.5 kB 393.8 kB/s eta 0:00:01
41.5/41.5 kB 336.4 kB/s eta 0:00:00
Requirement already satisfied: tqdm in c:\users\sanmitraa v r\appdata\local\programs\python\python311\lib\site-packages (from nltk) (4.67.1)
Requirement already satisfied: colorama in c:\users\sanmitraa v r\appdata\local\programs\python\python311\lib\site-packages (from click->nltk) (0.4.6)
Downloading nltk-3.9.2-py3-none-any.whl (1.5 MB)
0.0/1.5 MB ? eta -:--:--
0.0/1.5 MB ? eta -:--:--
0.1/1.5 MB 1.3 MB/s eta 0:00:02
0.1/1.5 MB 1.3 MB/s eta 0:00:02
0.1/1.5 MB 652.2 kB/s eta 0:00:03
0.2/1.5 MB 748.1 kB/s eta 0:00:02
0.2/1.5 MB 784.3 kB/s eta 0:00:02
0.3/1.5 MB 896.4 kB/s eta 0:00:02
0.4/1.5 MB 955.7 kB/s eta 0:00:02
277.8/277.8 kB 2.1 MB/s eta 0:00:00
Installing collected packages: regex, nltk
Successfully installed nltk-3.9.2 regex-2026.1.15
OUTLINE
TIMELINE
Spaces: 4
Cell 2 of 11
Sign In
Breaking news
'Depth, trust and...
Search
ENG
IN
10:32
07-02-2026

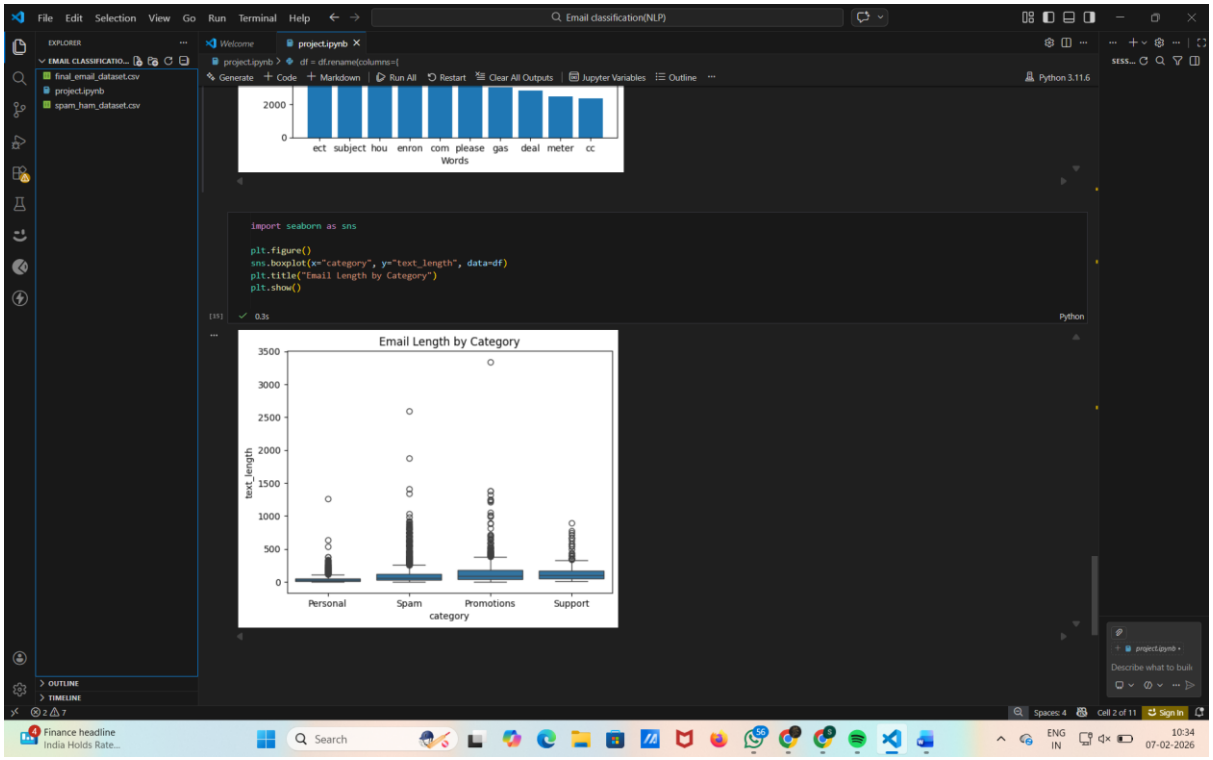

File Edit Selection View Go Run Terminal Help
Email classification(NLP)
EXPLORER
EMAIL CLASSIFICATIO...
final_email_dataset.csv
project.ipynb
spam_ham_dataset.csv
Welcome
project.ipynb
df = df.rename(columns={
Generate
Code
Markdown
Run All
Restart
Clear All Outputs
Jupyter Variables
Outline
Python 3.11.6
import pandas as pd
import re
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')
# load final dataset
df = pd.read_csv("final_email_dataset.csv")
stop_words = set(stopwords.words("english"))
def clean_text(text):
text = text.lower()
text = re.sub(r"[^a-z\s]", "", text) # remove punctuation & numbers
words = text.split()
words = [word for word in words if word not in stop_words]
return " ".join(words)
apply preprocessing
df["clean_text"] = df["email_text"].apply(clean_text)
print(df[["email_text", "clean_text"]].head())
6.9s
Python
[nltk_data] Downloading package stopwords to C:\Users\Sanmitraa V
[nltk_data] R\AppData\Roaming\nltk_data...
[nltk_data] Unzipping corpora\stopwords.zip.
email_text \
0 Subject: enron methanol ; meter # : 988291\r\n...
1 Subject: hpl nom for january 9 , 2001\r\n(see...
2 Subject: neon retreat\r\nho ho ho , we ' re ar...
3 Subject: photoshop , windows , office . cheap ...
4 Subject: re : indian springs\r\nthis deal is t...
clean_text
0 subject enron methanol meter follow note gave ...
1 subject hpl nom january see attached file hpln...
2 subject neon retreat ho ho ho around wonderful...
3 subject photoshop windows office cheap main tr...
4 subject indian springs deal book teco pvr reve...
Generate
Code
Markdown
import pandas as pd
import matplotlib.pyplot as plt
0.2s
Python
OUTLINE
TIMELINE
Spaces: 4
Cell 2 of 11
Sign In
Finance headline
India Holds Rate...
Search
ENG
IN
10:32
07-02-2026


File Edit Selection View Go Run Terminal Help
Email classification(NLP)
EXPLORER
EMAIL CLASSIFICATIO...
final_email_dataset.csv
project.ipynb
spam_ham_dataset.csv
Welcome
project.ipynb
df = df.rename(columns={
Generate
Code
Markdown
Run All
Restart
Clear All Outputs
Jupyter Variables
Outline
Python 3.11.6
import pandas as pd
import matplotlib.pyplot as plt
df = pd.read_csv("final_email_dataset.csv")
plt.figure()
df["category"].value_counts().plot(kind="bar")
plt.title("Email Category Distribution")
plt.xlabel("Category")
plt.ylabel("Number of Emails")
plt.show()
0.2s
Python
Email Category Distribution
Number of Emails
2000
1750
1500
1250
1000
750
500
250
0
Personal
Spam
Promotions
Support
Category
import re
import nltk
0.2s
Python
OUTLINE
TIMELINE
Finance headline
India Holds Rate...
Search
ENG
IN
10:33
07-02-2026


File Edit Selection View Go Run Terminal Help
Email classification(NLP)
EXPLORER
EMAIL CLASSIFICATIO...
final_email_dataset.csv
project.ipynb
spam_ham_dataset.csv
Welcome
project.ipynb
df = df.rename(columns={
Generate
Code
Markdown
Run All
Restart
Clear All Outputs
Jupyter Variables
Outline
Python 3.11.6
import re
import nltk
from nltk.corpus import stopwords
nltk.download("stopwords")
stop_words = set(stopwords.words("english"))
def clean_text(text):
text = text.lower()
text = re.sub(r"[^a-z\s]", "", text)
words = text.split()
words = [w for w in words if w not in stop_words]
return " ".join(words)
df["clean_text"] = df["email_text"].apply(clean_text)
0.2s
Python
[nltk_data] Downloading package stopwords to C:\Users\Sanmitraa V
[nltk_data] R\AppData\Roaming\nltk_data...
[nltk_data] Package stopwords is already up-to-date!
df["text_length"] = df["clean_text"].apply(lambda x: len(x.split()))
import matplotlib.pyplot as plt
plt.figure()
plt.hist(df["text_length"], bins=30)
plt.title("Email Length Distribution")
plt.xlabel("Number of Words")
plt.ylabel("Frequency")
plt.show()
0.1s
Python
Email Length Distribution
4000
3500
3000
2500
OUTLINE
TIMELINE
Finance headline
India Holds Rate...
Search
ENG
IN
10:33
07-02-2026


Email Length Distribution
Frequency
Number of Words
from collections import Counter
all_words = " ".join(df["clean_text"]).split()
common_words = Counter(all_words).most_common(10)
words, counts = zip(*common_words)
plt.figure()
plt.bar(words, counts)
plt.title("Top 10 Most Common Words")
plt.xlabel("Words")
plt.ylabel("Frequency")
plt.show()
Top 10 Most Common Words


from collections import Counter
all_words = " ".join(df["clean_text"]).split()
common_words = Counter(all_words).most_common(10)
words, counts = zip(*common_words)
plt.figure()
plt.bar(words, counts)
plt.title("Top 10 Most Common Words")
plt.xlabel("Words")
plt.ylabel("Frequency")
plt.show()
Top 10 Most Common Words
Frequency
ect subject hou enron com please gas deal meter cc
Words
import seaborn as sns


File Edit Selection View Go Run Terminal Help
Email classification(NLP)
EXPLORER
EMAIL CLASSIFICATIO...
final_email_dataset.csv
project.ipynb
spam_ham_dataset.csv
Welcome
project.ipynb
Generate
Code
Markdown
Run All
Restart
Clear All Outputs
Jupyter Variables
Outline
Python 3.11.6
2000
0
ect subject hou enron com please gas deal meter cc
Words
import seaborn as sns
plt.figure()
sns.boxplot(x="category", y="text_length", data=df)
plt.title("Email Length by Category")
plt.show()
0.3s
Python
Email Length by Category
text_length
3500
3000
2500
2000
1500
1000
500
0
Personal
Spam
Promotions
Support
category
OUTLINE
TIMELINE
Spaces: 4
Cell 2 of 11
Sign In
Finance headline
India Holds Rate...
Search
ENG
IN
10:34
07-02-2026