Email Spam Classification Using Naïve Bayes

A PROJECT REPORT
FOR
Introduction to AI (AI101B)
Session (2024-25)

Submitted By

Imran Ahmad
202410116100092
Harshit Singh
202410116100089

Submitted in the partial fulfilment of the Requirements of the Degree of

MASTER OF COMPUTER APPLICATION
Under the Supervision of
Mr. Apoorv Jain
Assistant Professor

Submitted to
DEPARTMENT OF COMPUTER APPLICATIONS
KIET Group of Institutions, Ghaziabad
Uttar Pradesh-201206

(MARCH - 2025)

1. Introduction

Objective

To build a machine learning model that classifies emails as spam or not spam using the Naive Bayes algorithm.

Motivation

- Email spam is a major concern for users, organizations, and ISPs.
- Automating spam detection helps improve communication efficiency and cyber security.
- This project showcases how Natural Language Processing (NLP) and Machine Learning solve real-world problems.

2. Dataset Description

Source: Public dataset from GitHub (emails.csv)

- **Features**:
- text: Content of the email
- spam: Target label (1 = spam, 0 = not spam)
- Dataset Size: Approximately 5728 email samples

3. Methodology

A. Data Preprocessing

- Train-Test Split → 80% training, 20% testing.
- Using CountVectorizer to convert email text into numeric vectors.

B. Machine Learning Models Used

- Multinomial Naive Bayes :-
- Suitable for text classification problems.
- Assumes independence between features (bag-of-words model).

C. Evaluation Metrics

- Accuracy: Proportion of correctly predicted emails.
- Confusion Matrix: Comparison of actual vs predicted labels.
- Precision, Recall, F1-score: For spam classification effectiveness.

Python Code:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
from wordcloud import WordCloud

url = 'https://raw.githubusercontent.com/nikhilkr29/Email-Spam-Classifier-using-Naive-Bayes/main/emails.csv'
df = pd.read_csv(url)

print(df.head())
print("\nClass Distribution:")
print(df['spam'].value_counts())
plt.figure(figsize=(6,4))
sns.countplot(x='spam', data=df, palette='mako')
plt.title('Spam vs Not Spam Count')
plt.xticks([0, 1], ['Not Spam', 'Spam'])
plt.show()
spam_words = ' '.join(df[df['spam'] == 1]['text'])
spam_wc = WordCloud(width=600, height=400,
background_color='black').generate(spam_words)
plt.figure(figsize=(8,6))
plt.imshow(spam_wc, interpolation='bilinear')
plt.axis('off')
plt.title("Most Common Words in Spam Emails", fontsize=15)
plt.show()
ham_words = ' '.join(df[df['spam'] == 0]['text'])
ham_wc = WordCloud(width=600, height=400,
background_color='white').generate(ham_words)
plt.figure(figsize=(8,6))
plt.imshow(ham_wc, interpolation='bilinear')
plt.axis('off')
plt.title("Most Common Words in Non-Spam Emails", fontsize=15)
plt.show()
X = df['text']
y = df['spam']
vectorizer = CountVectorizer()
X_transformed = vectorizer.fit_transform(X)
X_train, X_test, y_train, y_test = train_test_split(X_transformed, y, test_size=0.2,
random_state=42)
model = MultinomialNB()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
print("\nAccuracy Score:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
```

```
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(5,4))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=['Not Spam',
'Spam'], yticklabels=['Not Spam', 'Spam'])
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()
sample = ["Congratulations! You have won a free iPhone. Click here to claim it."]
sample_transformed = vectorizer.transform(sample)
print("\nPrediction (1 = Spam, 0 = Not Spam):",
model.predict(sample_transformed)[0])
```

4. Results & Discussion

Spam vs Not Spam Count Plot

WordClouds for visualizing common words in spam and non-spam emails

Confusion Matrix Heatmap to evaluate classification results

Sample Results:

Accuracy Score: ~98%

Prediction Example:

Input: “Congratulations! You have won a free iPhone.”

Output: Spam (1)

WordClouds for visualizing common words in spam and non-spam emails

Confusion Matrix Heatmap to evaluate classification results

Sample Results:

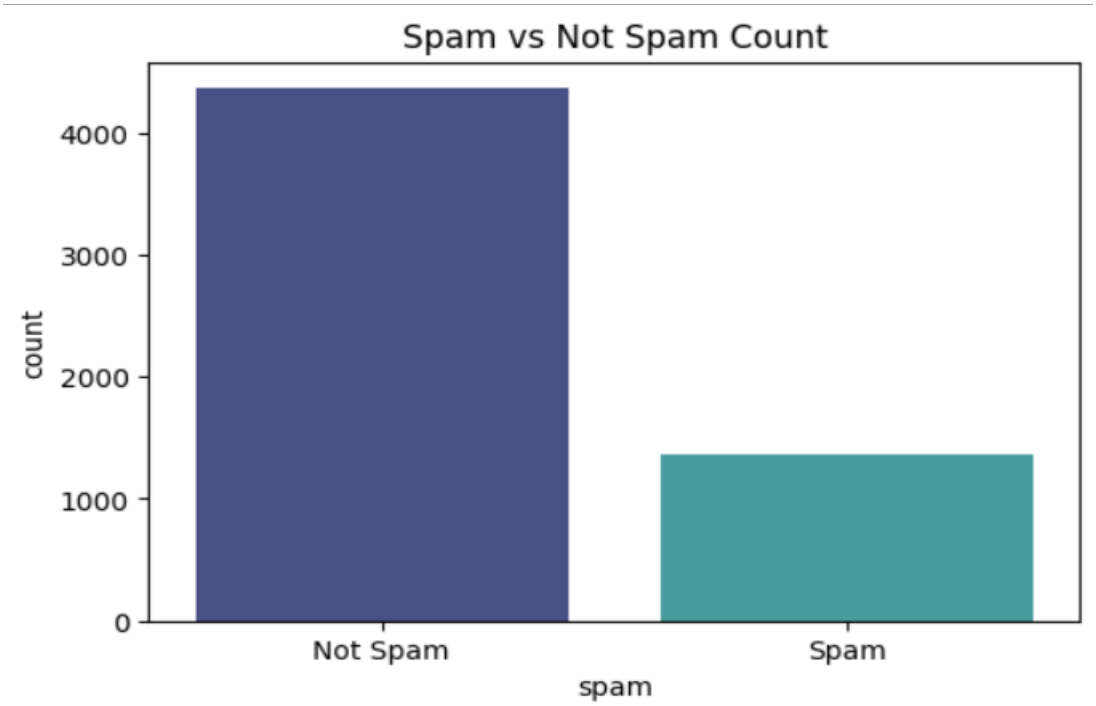
Accuracy Score: ~98%

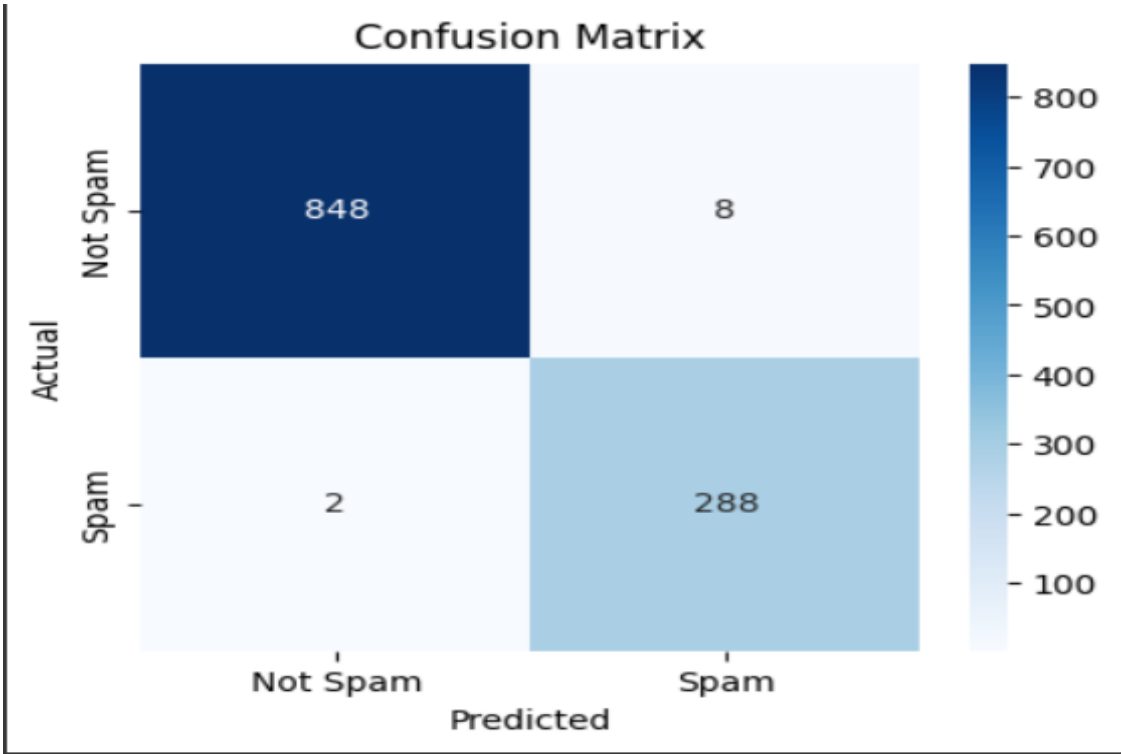
Prediction Example:

Input: “Congratulations! You have won a free iPhone.”

Output: Spam (1)

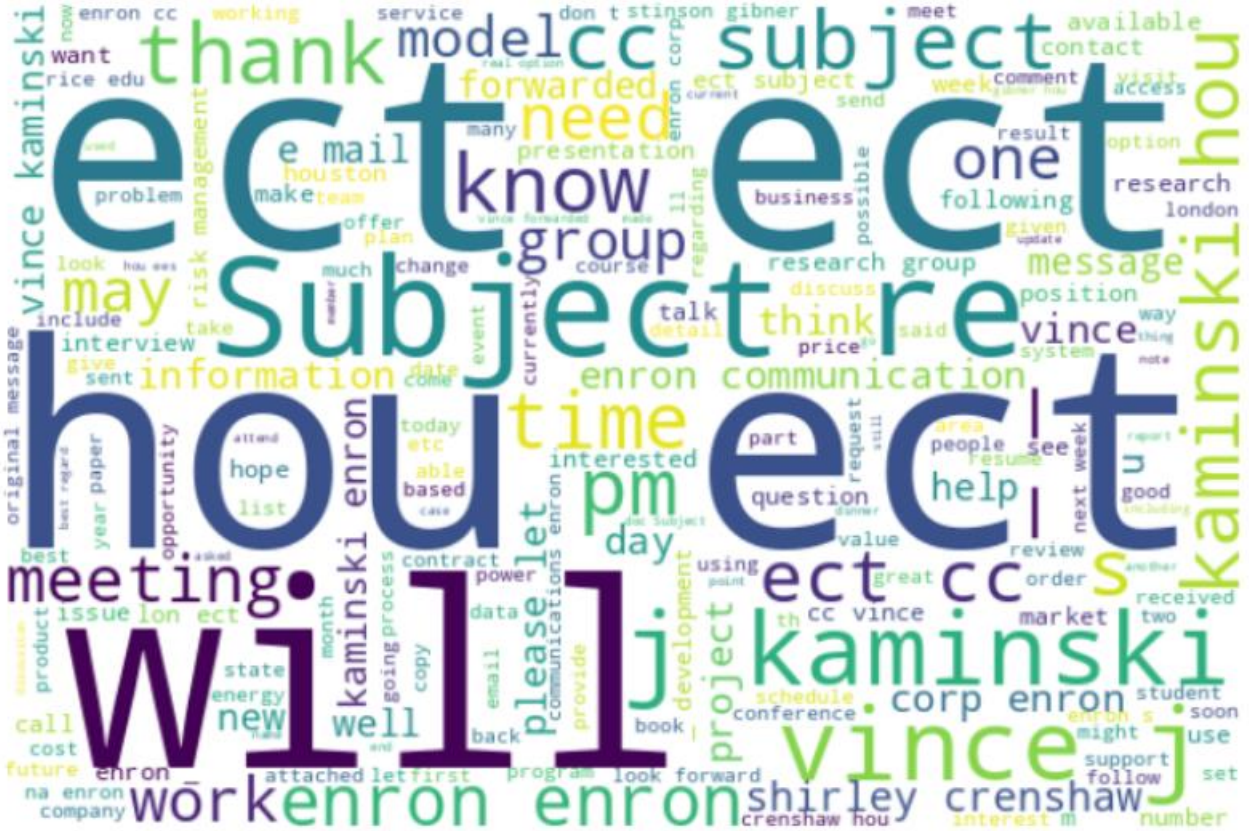
Output Screenshots






```
Prediction (1 = Spam, 0 = Not Spam): 1
```

Most Common Words in Non-Spam Emails



Most Common Words in Spam Emails

5. Conclusion & Future Scope

Conclusion

- Naive Bayes is a powerful algorithm for classifying text-based data like emails.
- With minimal preprocessing, it provides high accuracy and fast predictions.

Future Scope

- Implement TF-IDF Vectorization for improved feature weighting.
- Compare with other models like Logistic Regression, SVM, or XGBoost.
- Add more data fields like sender info or metadata for enhanced detection.

6. References

- Scikit-learn Documentation.
- GitHub: Email Spam Dataset.
- WordCloud & Seaborn libraries.
- Research papers on spam detection with ML.