Project Title: Text Analytics for Spam Detection Using Naïve Bayes

- Submitted By:
- Abhishek Kumar Verma

- Course:
- MBA-1st year

- Subject:
- hr/marketing

- Institution:
- [Axis institute of planning and management(723)]

- Date:
- [25-02-2026]

Abstract

- Spam messages have become a major problem in digital communication, causing inconvenience and potential security threats. The objective of this project is to develop a machine learning model that classifies SMS messages as spam or ham (legitimate messages). The Multinomial Naïve Bayes algorithm is used for classification due to its efficiency in handling text data. The SMS Spam Collection Dataset from the UCI Machine Learning Repository was used for training and testing the model. The dataset was preprocessed and converted into numerical form using the Bag of Words technique. The developed model achieved an accuracy of approximately 96–98%, demonstrating high effectiveness in spam detection.

- ***Introduction***

- With the rapid growth of digital communication, spam messages have become increasingly common. Spam messages are unsolicited and often promotional or fraudulent in nature. These messages can cause financial loss and privacy issues.
- Text analytics plays an important role in detecting spam messages by analyzing textual patterns and identifying suspicious content. Machine learning algorithms, particularly Naïve Bayes, are widely used in spam detection systems due to their simplicity and high performance.
- This project focuses on implementing a spam detection system using text analytics techniques and the Multinomial Naïve Bayes algorithm.

- ***Problem Statement***

- To develop a machine learning model that can automatically classify SMS messages as spam or ham using text analytics and the Naïve Bayes algorithm.

- ***Objectives***

- To collect and analyze SMS spam dataset
- To preprocess text data for machine learning
- To convert text data into numerical format
- To implement the Multinomial Naïve Bayes algorithm
- To evaluate the model performance using accuracy and other metrics

- ***Dataset Description***

- The dataset used in this project is the **SMS Spam Collection Dataset** from the UCI Machine Learning Repository.
- Dataset details:
- Total messages: 5,572
- Categories: Spam and Ham
- Format: CSV file
- Columns:
- v1 → Label (spam/ham)
- v2 → Message text
- The dataset size is approximately 600 KB and contains real-world SMS messages.

- ***Methodology***

- The following steps were followed:

- **Step 1: Data Collection**
- The dataset was downloaded and imported using Python.

- **Step 2: Data Preprocessing**
- Selected relevant columns (v1 and v2)
- Renamed columns to 'label' and 'message'
- Converted categorical labels (spam/ham) into numerical values (1/0)

- **Step 3: Text Vectorization**
- The Bag of Words technique was used to convert text into numerical features using Count Vectorizer.

- **Step 4: Splitting Dataset**
- The dataset was split into:
- 80% Training Data
- 20% Testing Data

- **Step 5: Model Training**

The Multinomial Naïve Bayes algorithm was trained on the training dataset

- **Step 6: Model Evaluation**
- The model was evaluated using:
- Accuracy
- Precision
- Recall
- F1-Score
- Confusion Matrix

- ***Algorithm Used – Multinomial Naïve Bayes***

- Naïve Bayes is a probabilistic classification algorithm based on Bayes' Theorem.
- Bayes' Theorem:
- P(A|B) = [P(B|A) × P(A)] / P(B)
- The algorithm assumes independence between features (words in text). The Multinomial variant is specifically designed for text classification problems and works well with word frequency data.
- Reasons for choosing Naïve Bayes:
- Fast computation
- Works well with text data
- Requires less training data
- High accuracy for spam detection

- ***Implementation Tools***

- Programming Language: Python
- Libraries Used:
- Pandas
- Scikit-learn
- Matplotlib
- Seaborn
- Development Environment: Command Prompt / VS Code

- ***Results***

- After training and testing the model:
- Accuracy achieved: ~96% to 98%
- The confusion matrix showed strong classification performance.
- Precision and recall values indicated reliable spam detection.
- (Insert Screenshot of Accuracy Here)
- (Insert Screenshot of Confusion Matrix Here)
- The model successfully classified most spam and ham messages correctly.

• ***Conclusion***

- This project successfully implemented a spam detection system using text analytics and the Multinomial Naïve Bayes algorithm. The model achieved high accuracy and demonstrated that Naïve Bayes is highly effective for text classification tasks. The system can be further improved using advanced natural language processing techniques.

- ***Future Scope***

- Use TF-IDF instead of Bag of Words
- Apply Deep Learning models (LSTM, Neural Networks)
- Deploy as a Web Application
- Integrate into real-time messaging platforms

- ***References***

- UCI Machine Learning Repository – SMS Spam Collection Dataset
- Scikit-learn Documentation
- Python Official Documentation

Command Prompt - py spam_detection.py

```
D:\>Spam_Detection_Text_Analytics
'Spam_Detection_Text_Analytics' is not recognized as an internal or external command,
operable program or batch file.

D:\>cd Spam-Detection-Text-Analytics

D:\Spam-Detection-Text-Analytics>dir
 Volume in drive D is DATA
 Volume Serial Number is BABD-ACAE

 Directory of D:\Spam-Detection-Text-Analytics

02/25/2026  06:48 PM    <DIR>          .
02/25/2026  10:01 AM               895 README.md.txt
02/25/2026  10:13 AM                34 requirements.txt.txt
02/25/2026  09:54 AM           503,663 spam.csv
02/25/2026  07:07 PM             1,388 spam_detection.py
               4 File(s)        505,980 bytes
               1 Dir(s)  343,902,007,296 bytes free

D:\Spam-Detection-Text-Analytics>py spam_detection.py

Accuracy: 0.9838565022421525

Classification Report:
              precision    recall  f1-score   support

           0       0.99      0.99      0.99       965
           1       0.96      0.92      0.94       150

    accuracy                           0.98      1115
   macro avg       0.97      0.96      0.96      1115
weighted avg       0.98      0.98      0.98      1115
```

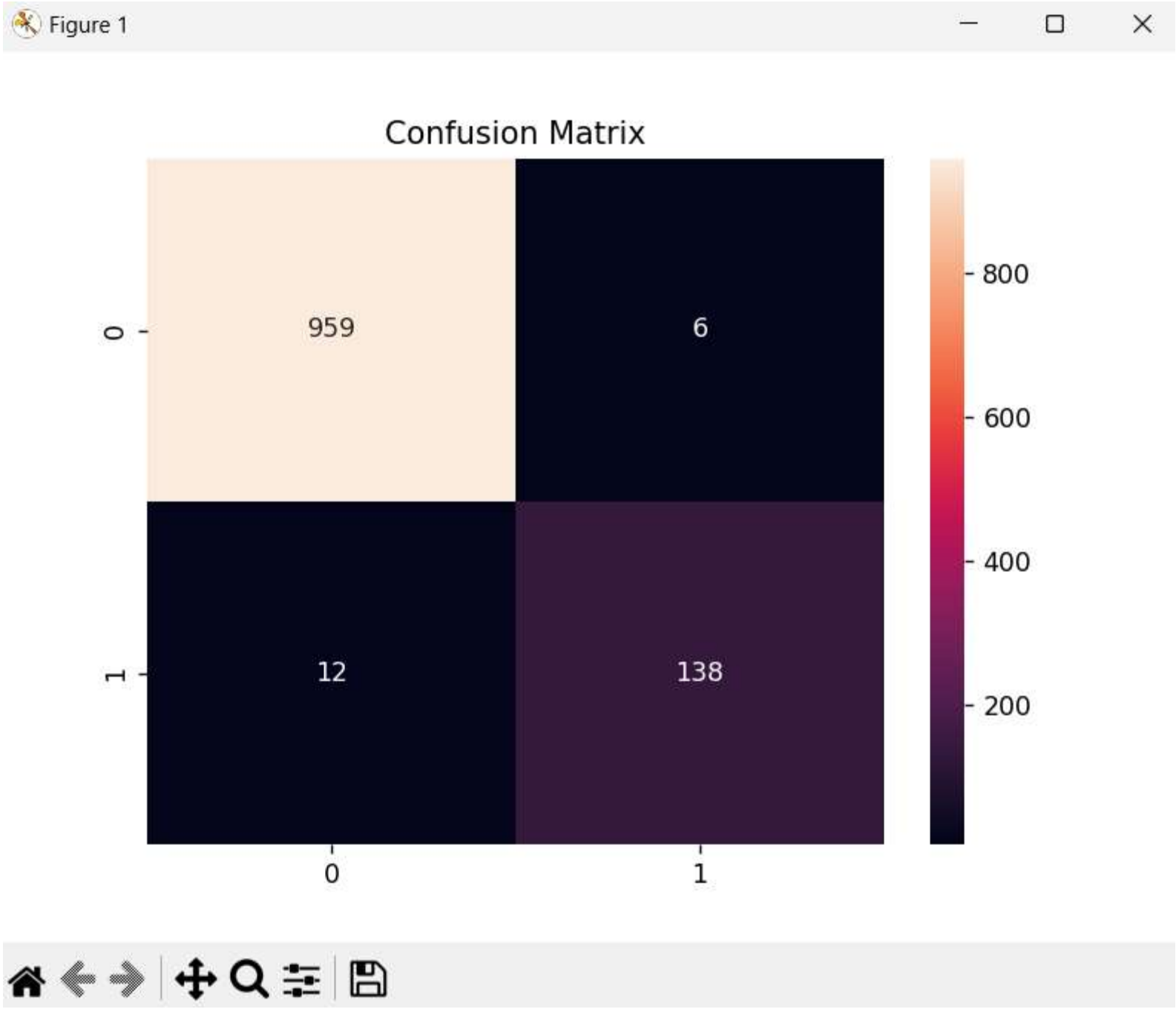
Figure 1
Confusion Matrix
959
6
12
138
0
1
0
1
800
600
400
200