SEPTEMBER 2025

PROJECT REPORT

SPAM MESSAGE CLASSIFIER

NLP + Machine Learning Project

DEVELOPED BY
Saiful Islam Rupom

Github Repository:
https://github.com/saiful-islam-rupom/spam-message-classifier.git

1. Introduction

With the rapid growth of digital communication, individuals and organizations face an increasing challenge in managing unwanted spam messages. Spam can include promotional SMS, phishing emails, or irrelevant text content, which can compromise productivity and sometimes security.

To address this issue, this project introduces a Spam Message Classifier that leverages Natural Language Processing (NLP) and Machine Learning (ML) to classify messages as Spam or Not Spam. The project applies preprocessing techniques, feature engineering, and supervised learning algorithms to build a reliable classification model. A Streamlit-based web application is also developed to make the model easily accessible and user-friendly.

2. Objectives

The main objectives of the project are:

- To design and implement an end-to-end machine learning pipeline for spam detection.
- To apply NLP preprocessing techniques for cleaning and normalizing text data.
- To identify the most effective machine learning algorithm for spam classification.
- To deploy the model in a web-based interface for real-time predictions.
- To provide a system that delivers reliable, interpretable, and interactive results for end-users.

3. Methodology

The project follows a structured workflow:

3.1 Data Collection & Exploration

- Dataset: spam_raw.csv from Kaggle.
- Initial steps included loading the dataset, handling duplicates, and checking for null values.
- Exploratory Data Analysis (EDA):
  - Visualized the distribution of spam vs. non-spam messages.
  - Assessed dataset balance to identify potential bias.
  - Analyzed message lengths, frequency of words, and patterns in spam messages.

3.2 Data Preprocessing

To prepare the dataset for modeling, the following NLP steps were applied:

- Lowercasing: Converted all text into lowercase for uniformity.
- Tokenization: Split text into individual tokens using the NLTK tokenizer.
- Stopword Removal: Removed common but uninformative words (e.g., “is”, “the”).
- Punctuation Removal: Stripped out symbols and special characters.
- Stemming: Applied PorterStemmer to reduce words to their root form (e.g., “running” → “run”).
- Reconstruction: Rebuilt processed tokens into cleaned sentences.

3.3 Feature Engineering

- Applied TF-IDF Vectorization to transform text into numerical features.
- Split the dataset into training and testing sets to evaluate performance.

3.4 Model Training & Evaluation

Three variants of Naive Bayes classifiers were tested:

1. GaussianNB
2. BernoulliNB
3. MultinomialNB (selected as the best-performing model)

- Evaluation Metrics: Accuracy and Precision Score were used to measure effectiveness.
- Results: The Multinomial Naive Bayes classifier achieved the best trade-off between accuracy and precision.
- The final trained model and TF-IDF vectorizer were serialized using Pickle (model.pkl, vectorizer.pkl).

3.5 Application Development

- Developed an interactive Streamlit web application for real-time classification.
- Key features:
  - User input text box for SMS/email messages.
  - Automatic preprocessing applied to input before prediction.
  - Predictions displayed as color-coded labels:
    - Red → Spam
    - Green → Not Spam

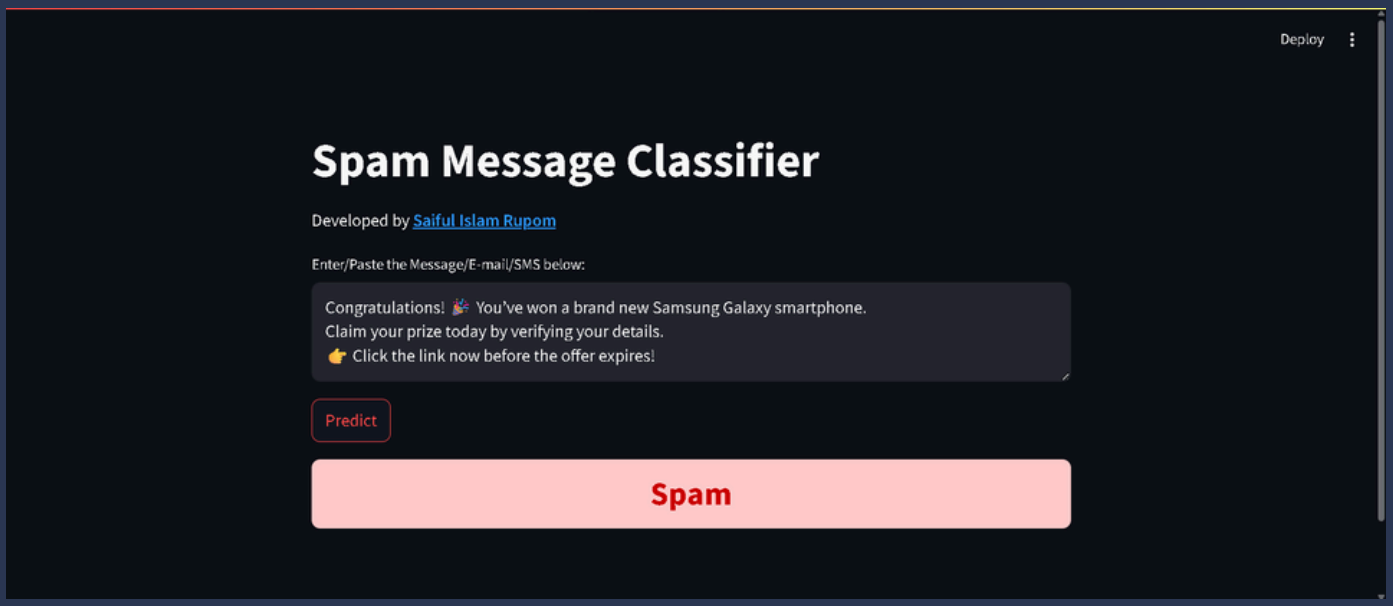
3.6 User Interaction & Output

- End-users can input any message into the app interface.
- The system processes the input through the trained pipeline.
- Final prediction is displayed instantly with visual emphasis.

4. Results & Discussion

- Model Performance:
  - The MultinomialNB classifier outperformed GaussianNB and BernoulliNB due to the categorical and sparse nature of TF-IDF features.
  - The model demonstrated high accuracy and reliable spam detection in test cases.
- Strengths:
  - Lightweight and efficient model suitable for real-time applications.
  - Clear and interpretable preprocessing pipeline.
  - Accessible web deployment using Streamlit.
- Limitations:
  - Model currently trained on English-language text only.
  - Limited to traditional ML algorithms; does not incorporate deep learning or contextual embeddings.

5. App Snapshot



Live App: Click to Try the App

6. Future Enhancements

To improve the system, the following future enhancements are proposed:

- Multilingual Support: Extend preprocessing and classification for multiple languages.
- Advanced Models: Experiment with deep learning architectures such as LSTMs, BERT, or Transformers for better contextual understanding.
- Explainability: Provide probability scores and explanations for each prediction.
- Enhanced UI/UX: Integrate charts, performance metrics, and dataset insights into the web app for a richer user experience.

7. Conclusion

The Spam Message Classifier successfully demonstrates the application of NLP and machine learning in a real-world text classification problem. Through structured preprocessing, feature engineering, and model evaluation, the project achieved a reliable spam detection system. The deployment as a Streamlit app ensures usability and accessibility for end-users.

This project highlights the potential of machine learning in addressing digital communication challenges and provides a foundation for further development into more advanced integrated systems and multilingual spam detection.