Spam Detection Model Report

1. Introduction

With the growing number of digital communications, spam messages have become a significant problem. This project aims to develop a machine learning model that can accurately classify SMS messages as either "spam" or "ham" (non-spam). The solution involves natural language processing (NLP) techniques and a Naive Bayes classification model to detect spam messages effectively.

2. Dataset

Source:
The dataset used is the SMS Spam Collection, a well-known corpus containing 5,572 labeled messages collected for spam detection research.

Structure:
- v1: Label of the message (ham or spam)
- v2: Message content (raw text)

Preprocessing:
- Unnecessary columns were dropped.
- Columns were renamed for clarity: label (spam or ham) and text (the actual message).

3. Text Preprocessing

To prepare the text data for modeling, a cleaning function was implemented. The steps include:
- Lowercasing: Standardizing text by converting all characters to lowercase.
- Punctuation Removal: Stripping all punctuation to reduce noise.
- Stopword Removal: Eliminating common English words that do not contribute to classification (e.g., “the”, “and”, “is”).
- Stemming: Reducing words to their base or root form using the Porter Stemmer.

A new column, clean_text, was added to the dataset containing the cleaned version of each message.

4. Feature Extraction

To convert text data into numerical format for machine learning:
- TF-IDF Vectorization was applied.

- Maximum of 3000 features were used to limit dimensionality and reduce overfitting.
- The resulting output is a feature matrix representing each message numerically.

5. Model Building

Algorithm Used:
- Multinomial Naive Bayes: A simple yet effective algorithm suitable for text classification problems.

Process:
- The dataset was split into 80% training and 20% testing using train_test_split.
- The Naive Bayes model was trained on the training data.

6. Evaluation Metrics

The model was evaluated using the following metrics:

- Accuracy: ~98%
- Precision: High
- Recall: High
- F1-Score: High

Additional outputs included:
- Confusion Matrix: Showing true positives, true negatives, false positives, and false negatives.
- Classification Report: Detailed per-class performance.

7. Prediction Function

A utility function predict_message(msg) was implemented to:
- Clean and preprocess a new message.
- Vectorize the message using the trained TF-IDF vectorizer.
- Predict and return the label (Spam or Ham).

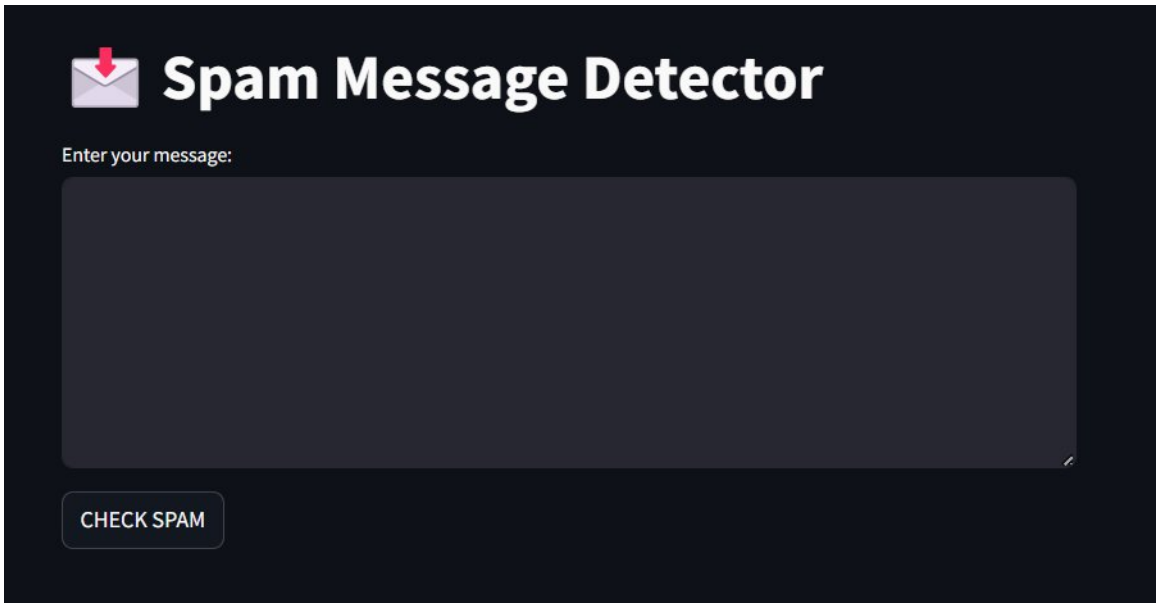
Example Predictions:
"Congratulations! You won a free iPhone. Click here to claim." → Spam
"Hey, are we still on for lunch today?" → Ham

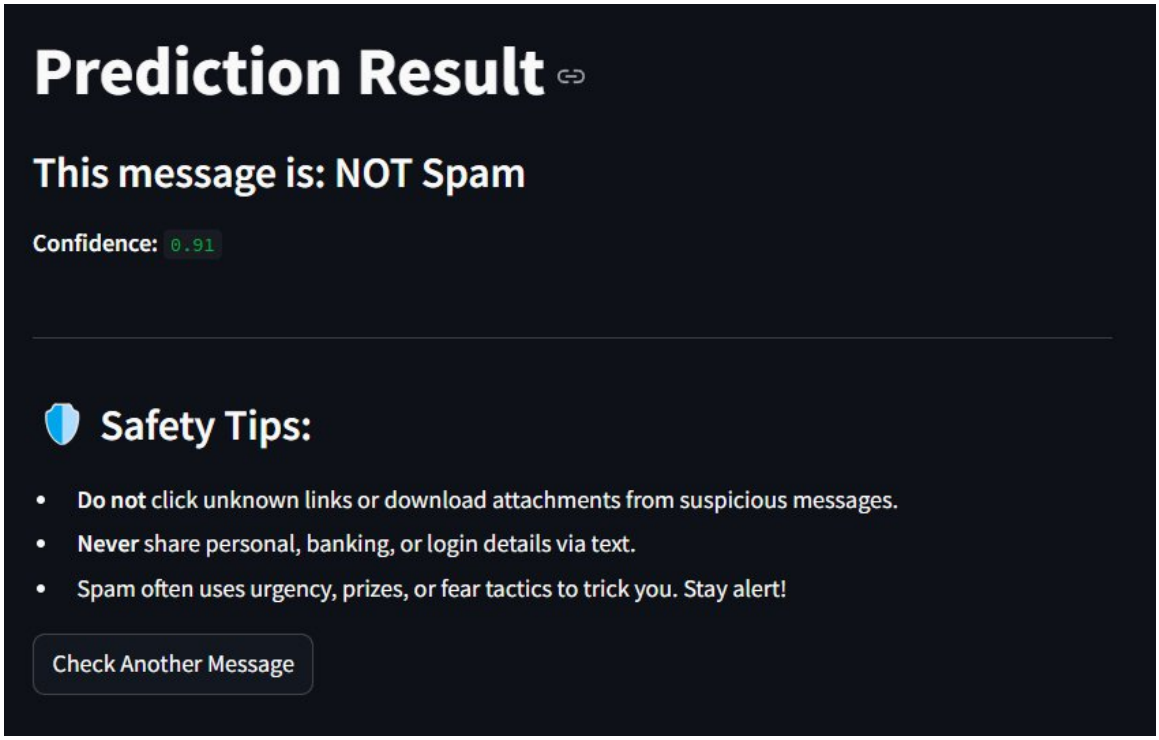
8. Model Persistence

The trained model and vectorizer were saved using Python's pickle module:
- spam_model.pkl: Contains the trained Naive Bayes model.
- vectorizer.pkl: Contains the TF-IDF vectorizer.

These files can be reused in a web app or any production environment for real-time spam detection.

9. UI using Streamlit





10. Conclusion

This project successfully demonstrates how machine learning and NLP can be combined to develop a robust spam detection system. The model shows high accuracy and is efficient for deployment. Future improvements may include:

- Testing with more diverse datasets.
- Exploring deep learning approaches.
- Enhancing the front-end user interface.