Spam Detection with Naive Bayes

Overview

This project is a simple **Spam Message Classifier** built using **Natural Language Processing (NLP)** and **Machine Learning**.
It uses **TF-IDF vectorization** and a **Multinomial Naive Bayes (MNB)** model to classify SMS messages as either:

- **Ham (Not Spam)**
- **Spam**

The dataset used is the **SMS Spam Collection Dataset**.

Dataset

- File: `spam.csv`
- Columns used:
  - **v1** → Target (ham/spam)
  - **v2** → Message text

During preprocessing:

- `v1` → renamed to `target`
- `v2` → renamed to `text`

Exploratory Data Analysis (EDA)

Before training the model, some analysis was done on the dataset:

1. **Class Distribution**
   - Ham messages: much higher than Spam messages (dataset is imbalanced).
2. **Message Length**
   - Spam messages are usually longer and contain more "offer/urgent" words.
   - Ham messages are often shorter and casual.
3. **Duplicates and Missing Values**
   - Duplicate rows and empty rows were removed to clean the dataset.

This EDA helped to understand the imbalance and gave insights into how spam differs from ham.

Preprocessing Steps

1. Convert all text to **lowercase**.
2. **Tokenize** text into words.
3. Remove **non-alphanumeric characters**.
4. Remove **stopwords** and **punctuation**.
5. Apply **stemming** using `PorterStemmer`.
6. Convert text into numerical features using **TF-IDF Vectorizer**.

Model

- **Algorithm**: Multinomial Naive Bayes (MNB)
- **Vectorizer**: TF-IDF (max_features=3000)
- **Train/Test Split**: 80/20 (with stratification to handle class imbalance)

Evaluation Metrics

The model prints out:

- Accuracy
- Confusion Matrix
- Precision

How to Use

Train the Model

Run the Python script to train the Naive Bayes model on the dataset.

Make Predictions

Use the function `predict_spam(message)` to classify new messages.