**Spam Email Detection**

**1. Introduction**

This Project demonstrates the process of building a spam email detection system. The primary objective is to classify emails as either spam or non-spam (ham) based on their content. This is achieved using machine learning techniques on textual data.

**2. Dataset**

- **Source**: A dataset containing labeled email messages.

* **Key Features**:
  + text: The content of the email.
  + label: Categorical labels (spam or ham).
  + label\_num: Numeric representation of the labels (1 for spam, 0 for ham).
* **Size**: 5,171 records with 4 columns.
* **Data Preparation**:
  + Removed newline characters within the email content.
  + Verified that there are no missing values.
  + Checked for and handled duplicate entries.

**3. Data Preprocessing**

- **Text Normalization**:

* Converted text to lowercase.
* Removed punctuation and special characters.
* Tokenized text and removed stopwords using NLTK.
* Applied stemming using PorterStemmer to reduce words to their root forms.

**4. Feature Engineering**

- **Vectorization**:

* Used CountVectorizer to convert text data into a bag-of-words representation.
* Generated a sparse matrix of word counts for use in machine learning models.

**5. Model Training**

- **Algorithm**: Random Forest Classifier (RFR).

* **Data Split**:
  + 80% for training, 20% for testing.
* **Hyperparameters**: Default settings were used unless specified otherwise.

**6. Evaluation**

- **Metrics**:

* Accuracy Score: Assesses overall correctness.
* Precision: Measures the ratio of true positives to all predicted positives.
* Recall: Evaluates the model’s ability to find all relevant instances.
* F1 Score: Harmonic mean of precision and recall.
* **Confusion Matrix**:
  + Visualized to understand true positives, true negatives, false positives, and false negatives.

**7. Conclusion**

The notebook successfully builds a spam detection system using a Random Forest Classifier. The evaluation metrics demonstrate its effectiveness, and the following improvements can be explored:

* Use of advanced NLP techniques like TF-IDF or word embeddings.
* Experimenting with different classifiers such as Naive Bayes or deep learning models.
* Hyperparameter tuning for the Random Forest model to optimize performance.