RV UNIVERSITY
School of Computer Science and Engineering
Bengaluru – 560059

Machine Learning For Cyber Security

VII Semester

Student Name:	Hussaina Mustafa
USN:	1RVU22BSC035
Academic Year	2025 - 2026

Email Spam Detection Using Machine Learning

1. Introduction

Email spam has become a major problem for individuals and organizations, causing security risks, loss of productivity, and potential financial fraud. Detecting spam automatically using machine learning provides an effective solution to filter unwanted emails. This project focuses on building a **spam detection system** using machine learning models trained on a preprocessed email dataset. The system also provides a web interface for real-time classification of PDF emails.

2. Dataset Description

The dataset used is the **TREC 2007 Public Corpus (preprocessed)**, which contains labeled emails indicating spam (1) or not spam (0). Each email record consists of the following fields:

- label – Indicates SPAM (1) or NOT SPAM (0)
- subject – Email subject
- email_to – Recipient email address
- email_from – Sender email address
- message – Email content

Dataset shape: 75,419 emails × 5 columns

The dataset is already partially preprocessed; however, additional cleaning was performed to normalize the text for better model performance.

3. Data Preprocessing

The following preprocessing steps were applied to the email messages:

1. **Lowercasing:** All text was converted to lowercase to ensure uniformity.
2. **Removing numbers:** Any numerical digits were removed to focus on textual content.
3. **Removing punctuation:** Punctuation marks were removed as they do not contribute meaningfully to classification.

4. **Whitespace normalization:** Extra spaces and newlines were removed to produce clean, consistent text.

After preprocessing, the cleaned text was stored in a new column clean_text.

Next, the text data was transformed into numerical features using **TF-IDF (Term Frequency–Inverse Document Frequency)**:

- Maximum 5000 features
- English stopwords removed

This allows the machine learning models to quantify the importance of words across all emails.

4. Model Training and Evaluation

Several machine learning algorithms were trained and compared:

- **Naive Bayes (MultinomialNB)** – Assumes word frequencies follow a multinomial distribution.
- **Logistic Regression** – A linear model suitable for binary classification.
- **Support Vector Machine (LinearSVC)** – Maximizes the margin between classes.
- **Random Forest** – Ensemble of decision trees for robust predictions.
- **K-Nearest Neighbors (KNN)** – Classifies based on similarity to neighbors.

The dataset was split into training (80%) and testing (20%) sets, stratified by labels to maintain class distribution.

Evaluation Metrics:

- Accuracy
- Precision
- Recall
- F1-score

Each model was trained and evaluated on the test set. The model with the highest **F1-score** was chosen as the best performing model.

Example Results (Best Model – Random Forest):

Metric	Value
Accuracy	0.9940
Precision	0.9948
Recall	0.9962
F1 Score	0.9955

The trained model, TF-IDF vectorizer, and model info (algorithm, evaluation metrics, dataset shape) were saved using pickle for deployment.

5. Web Application Workflow

A **Streamlit-based web application** was developed to allow real-time classification of PDF emails. The workflow of the application is as follows:

1. The user uploads one or more PDF emails through the interface.
2. Each PDF is parsed using **PyPDF2** to extract text.
3. Extracted text is cleaned using the same preprocessing steps as the training data.
4. The TF-IDF vectorizer transforms the cleaned text into numerical features.
5. The trained best model predicts whether each email is SPAM or NOT SPAM.
6. Results are displayed in a table along with a **horizontal bar chart** showing the number of SPAM vs NOT SPAM emails.
7. The app also displays a summary of the dataset shape, algorithm used, and evaluation metrics of the best model.

Enhancements for User Experience:

- Horizontal bar chart for intuitive visualization

- Color-coded labels (Red for SPAM, Green for NOT SPAM)
- Clear display of best model and evaluation metrics

6. Implementation Requirements

- **Frontend:** Streamlit (Python) with interactive visualizations using Plotly
- **Backend:** Python, handling PDF parsing, preprocessing, and predictions
- **Libraries Used:** pandas, numpy, scikit-learn, PyPDF2, pickle, plotly, re, string

7. Conclusion

This project successfully demonstrates **automated email spam detection** using machine learning. The final system provides:

- High accuracy and F1-score in detecting spam emails
- Ability to handle real-time PDF uploads
- User-friendly interface with visual feedback
- Comparison of multiple algorithms to select the best performing model

The solution can be extended in the future to include more sophisticated text preprocessing, deep learning-based models, and integration with email servers for live monitoring.