JSPM's
RAJARSHI SHAHU COLLEGE OF ENGINEERING
TATHAWADE, PUNE-33

DEPARTMENT OF COMPUTER ENGINEERING

Case Study

on

Email Spam Detection System

– Submitted by –

Name of Student: Adarsh Gangshettiwar

Roll No: CS3274

PRN No: RBTL24CS145

Class and Division: 3rd year B Division

Course: Machine Learning

1. Title

- **Email Spam Detection System**

2. Background/ Introduction

With the rapid growth of email communication, a significant amount of unwanted and malicious content such as spam, phishing, and advertising emails is transmitted daily. These spam emails not only clutter inboxes but also pose serious security threats like phishing attacks, malware distribution, and fraudulent schemes.

Machine Learning provides a more powerful and adaptive solution to this problem. By analyzing and learning from historical email data, Machine Learning algorithms can automatically identify patterns and classify incoming emails as spam or ham (non-spam).

In this project, we develop an Email Spam Detection System using classical Machine Learning techniques. Text data from emails is transformed into numerical features, and classification algorithms such as Naïve Bayes, Logistic Regression, or Support Vector Machines are trained to accurately detect spam. This approach enhances accuracy, adaptability, and scalability compared to traditional methods.

3. Problem Statement

The main challenge is to design and develop an efficient and accurate email spam detection system that can automatically classify incoming emails as spam or non-spam using machine learning techniques. The system should be able to handle a variety of email formats and textual content, ensuring a high detection rate with minimal false positives.

4. Objectives

- To collect and preprocess email data for spam classification.
- To apply suitable NLP techniques to clean and transform text data.
- To build and train machine learning models for classifying emails.
- To evaluate model performance using appropriate metrics.

5. Libraries required

- **NumPy** – For numerical operations
- **Pandas** – For data manipulation and analysis
- **Matplotlib / Seaborn** – For data visualization
- **Scikit-learn** – For machine learning algorithms and evaluation metrics
- **NLTK / re** – For text preprocessing (stopword removal, tokenization, etc)
- **TfidfVectorizer** – For feature extraction from text.

6. Approach/ Methodology

The project follows the Machine Learning workflow:

1. Data Collection:
    - Use publicly available datasets such as the "SMS Spam Collection" or "Enron Email Dataset".
2. Data Preprocessing:
    - Clean the email text by removing punctuation, numbers, and special characters.
    - Convert text to lowercase and remove stopwords.
    - Apply stemming or lemmatization.
3. Feature Extraction:
    - Use Bag of Words (BoW) or TF-IDF technique to convert text data into numerical features suitable for ML algorithms.
4. Model Building:
    - Train different classification models like Naïve Bayes, Logistic Regression, Support Vector Machine (SVM), or Random Forest.
5. Model Evaluation:
    - Evaluate models using metrics such as accuracy, precision, recall, F1-score, and confusion matrix.
6. Deployment / Demonstration:
    - Integrate the model into a simple Python script or web app to classify new emails.

7. Implementation

```
import pandas as pd
import numpy as np
import re
import string
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report
import joblib

# Step 1: Load Local Dataset
df = pd.read_csv("Email_Spam.csv")
print("Dataset Loaded:")
print(df.head())

# Step 2: Preprocessing
def clean_text(text):
    text = text.lower()
    text = re.sub(r'\d+', '', text)
    text = text.translate(str.maketrans('', '', string.punctuation))
    return text.strip()
df['cleaned_message'] = df['message'].apply(clean_text)
df['label_num'] = df['label'].map({'ham': 0, 'spam': 1})

# Step 3: Train-Test Split
```

```
X_train, X_test, y_train, y_test = train_test_split(
df['cleaned_message'], df['label_num'], test_size=0.2, random_state=42)

# Step 4: Feature Extraction
vectorizer = TfidfVectorizer()
X_train_tfidf = vectorizer.fit_transform(X_train)
X_test_tfidf = vectorizer.transform(X_test)

# Step 5: Model Training
model = MultinomialNB()
model.fit(X_train_tfidf, y_train)

# Step 6: Evaluation
y_pred = model.predict(X_test_tfidf)
print("\nAccuracy:", accuracy_score(y_test, y_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))

# Step 7: Save Model
joblib.dump(model, "spam_model.pkl")
joblib.dump(vectorizer, "tfidf_vectorizer.pkl")
print("Model and Vectorizer saved successfully!")
```

8. GitHub Link:

https://github.com/AdarshG07/Email-Spam-Detection-System.git

9.Results

Dataset Loaded:

	label	message
0	ham	Hey how are you?
1	spam	Win a $1000 Walmart gift card now!
2	ham	Are we still meeting tomorrow?
3	spam	Congratulations you have been selected for a f...
4	ham	Lunch at 1 PM?

Accuracy: 0.6666666666666666

Confusion Matrix:

[[1 1]

[0 1]]

Classification Report:

	precision	recall	f1-score	support
0	1.00	0.50	0.67	2
1	0.50	1.00	0.67	1
accuracy			0.67	3
macro avg	0.75	0.75	0.67	3
weighted avg	0.83	0.67	0.67	3

10.Conclusion

The Email Spam Detection System successfully demonstrates how machine learning and NLP can be applied to classify spam emails with high accuracy. The system is scalable and can adapt to new types of spam through continuous learning. With further improvements such as deep learning models or real-time email integration, this system can be effectively used in real-world email filtering applications.