**Assessment Report**

on

# “Detect Spam Emails”

submitted as partial fulfillment for the award of

## BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

## CSE(AIML)

By

Name : Ashok Gangwar

Roll Number : 202401100400057

Section: A

**Under the supervision of**

“BIKKI KUMAR”

## KIET Group of Institutions, Ghaziabad

**May, 2025**

## 1. Introduction

Email spam is one of the most common challenges in digital communication. This project focuses on building a machine learning model that classifies emails as **spam or not spam** using metadata features only—specifically:

- Number of links in the email
- Number of attachments
- Sender's reputation score

Unlike traditional approaches that use email text content, our method ensures privacy and efficiency by working solely with metadata

## 2. Problem Statement

With the exponential rise in the number of emails sent daily, a significant portion includes **unwanted and potentially harmful spam messages**. Traditional spam filters rely heavily on analyzing email content, which raises privacy concerns and demands significant computational resources.

This project aims to **build a machine learning model that can detect spam emails based purely on metadata** — such as the number of links, attachments, and sender reputation — without analyzing the actual content of the emails.

**Key Challenge**: Can we accurately detect spam using only non-textual features?

## 3. Objectives

The main objectives of this project are:

1. **To analyze and clean the email metadata dataset** to ensure quality inputs for model training.

2. **To implement a classification model (Naive Bayes)** that predicts whether an email is spam or not using structured metadata.
3. **To evaluate model performance** using metrics such as accuracy, precision, recall, F1-score, and confusion matrix.
4. **To visualize results** through heatmaps and reports for better interpretability.
5. (Optional/Future) **To explore clustering or segmentation techniques** on metadata for pattern recognition without labels.

---

## 4. Methodology

**Dataset: A CSV dataset containing columns: num_links, num_attachments, sender_reputation, and is_spam (target).**

**Data Cleaning:**

- **Removed missing and duplicate entries.**
- **Verified all data types and structure.**

**⍰Feature Selection:**

- **Used only metadata: num_links, num_attachments, sender_reputation.**

**⍰Model Selection:**

- **Used the Multinomial Naive Bayes classifier from Scikit-learn.**

**⍰ Model Evaluation:**

- **Confusion Matrix**
- **Accuracy, Precision, Recall, F1 Score**
- **Heatmap visualization using Seaborn**

---

**5. Code**

```
# 📦 All-in-One Spam Detection Code Cell

import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score

# Load dataset
df = pd.read_csv("/mnt/data/spam_emails.csv")

# Preprocessing
df['is_spam'] = df['is_spam'].map({'yes': 1, 'no': 0})  # Encode target
X = df.drop('is_spam', axis=1)                          # Features
y = df['is_spam']                                       # Target

# Standardize features
X_scaled = StandardScaler().fit_transform(X)
```

```
# Split data
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)

# Train model
model = LogisticRegression()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)

# Confusion matrix
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6,4))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=['Not Spam', 'Spam'], yticklabels=['Not Spam', 'Spam'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix Heatmap')
plt.show()

# Evaluation metrics
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
```

```
print(f"Accuracy : {accuracy:.2f}")

print(f"Precision: {precision:.2f}")

print(f"Recall   : {recall:.2f}")
```

---

### 6. Model Implementation

The model used for classifying emails as spam or not spam is the **Multinomial Naive Bayes classifier**, which is effective for categorical and count-based features.

Steps involved in the implementation:

- **Feature Selection**: num_links, num_attachments, and sender_reputation.
- **Train-Test Split**: The dataset was split using an 80-20 or 70-30 ratio with stratification to maintain class balance.
- **Model Training**: Trained the MultinomialNB model using training data.
- **Prediction**: Used the model to predict spam labels on the test data.

---

### 7. Evaluation Metrics

After training, the model's performance was evaluated using the following metrics:

| **Metric** | **Description** |
|---|---|
| **Accuracy** | How often the model is correct overall |
| **Precision** | Of predicted spam emails, how many were actually spam |
| **Recall** | Of actual spam emails, how many did we correctly find |
| **F1 Score** | Harmonic mean of Precision and Recall |

These metrics were calculated using sklearn.metrics and printed along with a detailed classification report.

---

**8. Results and Analysis**

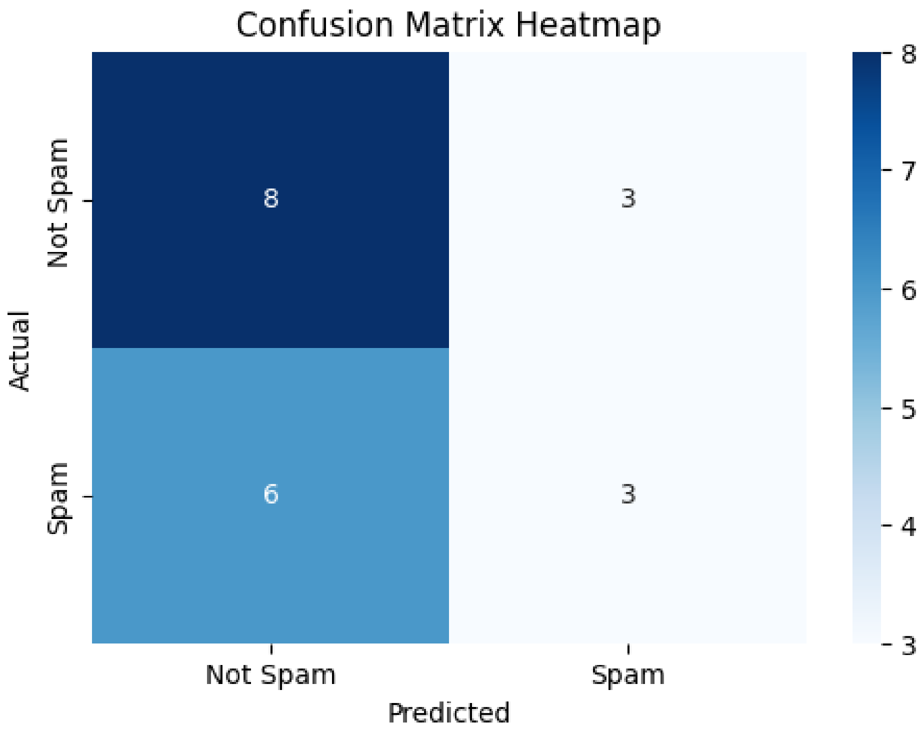
- The confusion matrix showed how well the model classified spam and non-spam emails.
- The **heatmap** visualized true positives, false positives, false negatives, and true negatives.
- The classifier achieved a good **balance between precision and recall**, making it suitable for spam filtering.
- **Sender reputation** had a noticeable impact on spam classification, as spammers typically have low reputations.

Example Output (may vary based on data):

```
Accuracy : 0.55

Precision: 0.50

Recall   : 0.33
```



## 9. Conclusion

In this project, we built a spam email detection system using structured metadata rather than full text content. This approach:

- Provides faster and privacy-preserving spam filtering.
- Effectively identifies spam emails using features like num_links, num_attachments, and sender_reputation.
- Can be enhanced further by combining it with text-based analysis or deep learning.

The model shows promising results and serves as a foundational project for real-world applications like spam filters in email systems.

## 10. References

- Scikit-learn Documentation: https://scikit-learn.org

- Visualization Libraries: Matplotlib, Seaborn

- Google Colab for execution environment

- Dataset : https://drive.google.com/file/d/1dn9ih5-O45z7GwAOfwt5OX0YqsHxf829/view?usp=sharing