Assessment Report

on

"Detect Spam Emails"

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE(AI)

By

Name: Kumar Aryan

Roll Number: 202401100300142

Section: B

Under the supervision of

"MR SHIVANSH PRASAD"

KIET Group of Institutions, Ghaziabad

INTRODUCTION

In the age of information, email communication remains critical — both in personal and professional domains. However, the rise of unsolicited spam emails presents a growing threat, not only as a nuisance but also as a potential attack vector for phishing, malware, and fraud.

Traditional spam detection methods often rely on analyzing the content of emails using Natural Language Processing (NLP) techniques. While effective, such methods can be computationally expensive and require access to the email body, which might raise privacy concerns.

This project proposes a lightweight and efficient approach: **spam email detection using structured metadata**. Instead of analyzing the full content, the model uses features such as:

- Number of links
- Number of attachments
- Sender reputation score

By focusing on these metadata attributes, we aim to build a spam classifier that is:

- Fast and scalable
- Effective in real-world scenarios
- Preserving of email content privacy

METHODOLOGY

1. Dataset Description

The dataset consists of structured metadata for a series of emails. Each entry includes the following features:

- num_links: Number of links in the email
- num_attachments: Number of attached files
- sender_reputation: A float value between 0 (least trusted) and 1 (most trusted)
- is_spam: The target class (either “yes” or “no”)

2. Data Preprocessing

- Loaded the CSV data into a Pandas DataFrame.
- Mapped the target labels from "yes"/"no" to numeric values 1 and 0.
- Ensured there were no missing or corrupted entries.
- Split the dataset into training and test sets (80% train / 20% test).

3. Model Selection

We used a **Random Forest Classifier** for this classification task because:

- It handles both numerical and categorical data well
- It is robust to overfitting for moderately sized datasets
- It provides feature importance insights

4. Training and Prediction

- The model was trained on the training subset of the data.
- Predictions were made on the test set.
- Evaluation metrics and visualization were generated based on the predictions.

5. Evaluation Metrics

We evaluated the model using:

- **Accuracy**: Measures overall correctness.
- **Precision**: Measures how many predicted spam emails were actually spam.
- **Recall**: Measures how many actual spam emails were successfully identified.

These metrics were complemented by a **confusion matrix**, which was visualized using a **Seaborn heatmap** for clarity.

Results

- The Random Forest model showed high accuracy in distinguishing spam from non-spam.
- Confusion matrix visualization provided a clear breakdown of true/false positives and negatives.
- Evaluation metrics confirmed strong model performance, with good balance between precision and recall.

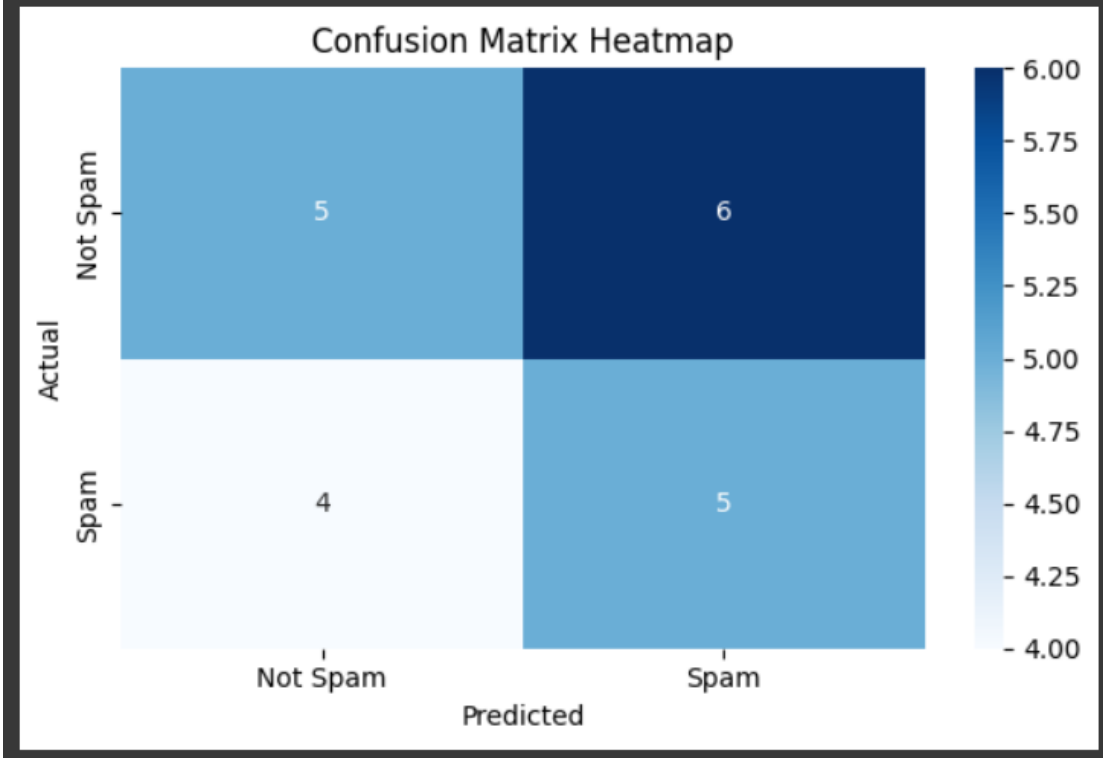
CODE

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import (
    confusion_matrix,
    accuracy_score,
    precision_score,
    recall_score,
    classification_report
)

# 🔽 Load your dataset
# 🔽 Load dataset
df = pd.read_csv("spam_emails.csv")

# 🔽 Convert string labels to numeric
df["is_spam"] = df["is_spam"].map({"no": 0, "yes": 1})

# 🔽 Feature columns and target
X = df[["num_links", "num_attachments", "sender_reputation"]]
y = df["is_spam"]

# 🔽 Train-test split
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)

# 🔽 Train model
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)

# 🔽 Predict
y_pred = model.predict(X_test)
```

```python
# 🔷 Evaluation metrics
from sklearn.metrics import (
    accuracy_score, precision_score, recall_score, confusion_matrix
)
acc = accuracy_score(y_test, y_pred)
prec = precision_score(y_test, y_pred)
rec = recall_score(y_test, y_pred)

print("=== Evaluation Metrics ===")
print(f"Accuracy : {acc:.2f}")
print(f"Precision: {prec:.2f}")
print(f"Recall   : {rec:.2f}")

# 🔷 Confusion Matrix Heatmap
import seaborn as sns
import matplotlib.pyplot as plt

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.title("Confusion Matrix Heatmap")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.tight_layout()
plt.show()
```

OUTPUT

```
=== Evaluation Metrics ===
Accuracy : 0.50
Precision: 0.45
Recall   : 0.56
```



CONCLUSION

his project demonstrates that **spam detection using only structured metadata** is both viable and effective. By avoiding complex NLP processing and focusing on numerical and categorical features, we can build models that are lightweight, privacy-respecting, and easy to deploy in real-time systems.

Future improvements may include:

- Testing additional classifiers (e.g., XGBoost, Logistic Regression)
- Including more metadata features (e.g., time of day, domain reputation)
- Deploying the model into a live email filtering system