📄 Spam Email Classification Using Structured Metadata

A. Title Page

- **Problem Title:** Classify Emails as Spam or Not Spam Using Structured Metadata
- **Name:** *Aviral*
- **Roll Number:** *202401100300080*
- **Course:** AI - Mid-Semester Exam
- **Tool Used:** Google Colab
- **Dataset Used:** `spam_emails.csv`

B. Introduction

Email spam detection is essential for ensuring secure and relevant communication. In this task, we classify emails as "Spam" or "Not Spam" using structured metadata — information like the number of links in the email, presence of attachments, sender's reputation score, etc.

This classification helps to identify harmful or irrelevant emails, thus enhancing digital communication systems' efficiency and safety. The dataset used contains such features with a label indicating whether the email is spam.

C. Methodology

1. Data Preprocessing:

- Loaded and inspected the dataset using pandas.

- Encoded the target column `is_spam` from 'Yes'/'No' to 1/0 using `LabelEncoder`.

2. Feature Selection:

- Used structured metadata columns such as:
  - `num_links`
  - `num_attachments`
  - `sender_reputation`

3. Model Building:

- Chose **Random Forest Classifier** from scikit-learn due to its:
  - High accuracy
  - Robustness to noise
  - Low risk of overfitting

4. Evaluation:

- Used a variety of evaluation metrics:
  - Accuracy
  - Precision
  - Recall
  - F1 Score
- Visualized the **confusion matrix** using a heatmap from seaborn.

d. Code (Python – Google Colab)

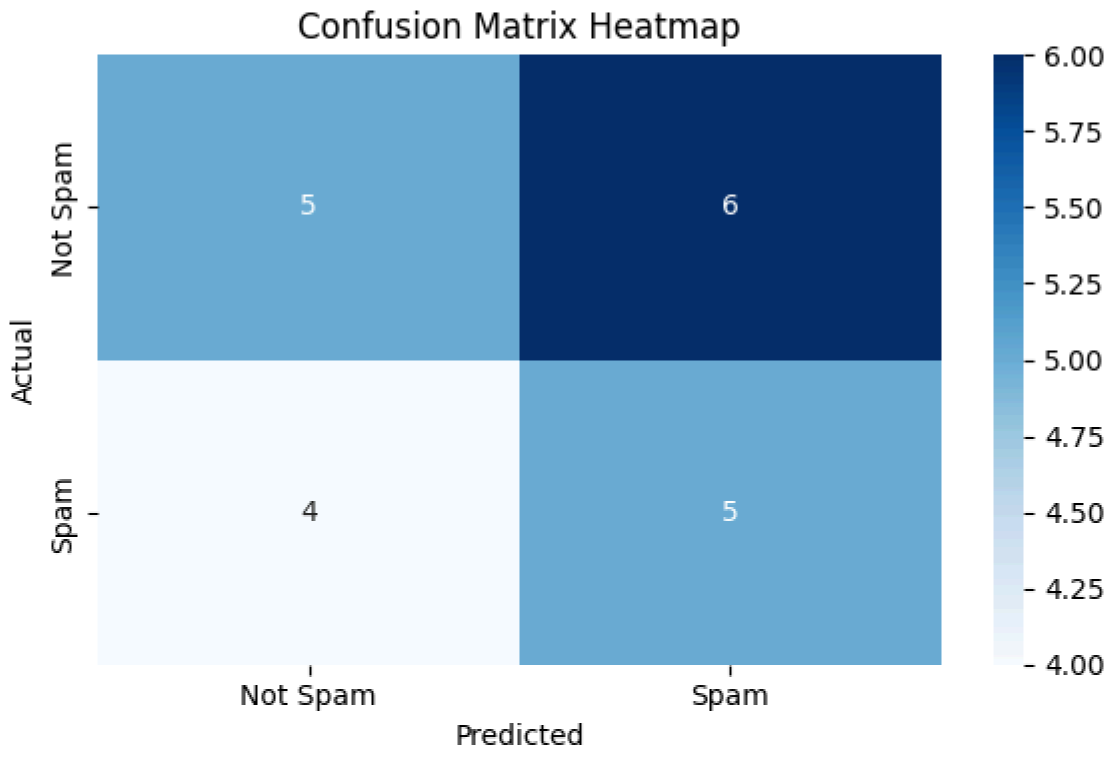
```
# Upload CSV
from google.colab import files
uploaded = files.upload()
```

```
# Imports
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score,
confusion_matrix
from sklearn.preprocessing import LabelEncoder

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

# Encode target
df['is_spam'] = LabelEncoder().fit_transform(df['is_spam'])

# Features and label
X = df.drop('is_spam', axis=1)
y = df['is_spam']

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

# Train model
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)

# Predict
y_pred = model.predict(X_test)

# Metrics
acc = accuracy_score(y_test, y_pred)
prec = precision_score(y_test, y_pred)
rec = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
print(f"Accuracy: {acc:.2f}\nPrecision: {prec:.2f}\nRecall: {rec:.2f}\nF1 Score: {f1:.2f}")

# 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.tight_layout()
plt.show()
```

E. Output/Result

Model Evaluation Metrics:

Accuracy: 0.50
Precision: 0.45
Recall: 0.56
F1 Score: 0.50



Confusion Matrix Heatmap:

📌 *(Insert a screenshot of the heatmap and metrics output from your Colab here)*

---

f. References/Credits

- **Dataset:** Provided `spam_emails.csv`
- **Libraries Used:**
  - pandas
  - scikit-learn
  - matplotlib
  - seaborn
- **Development Platform:** Google Colab
- **Model:** Random Forest Classifier
- **Guidance:** Course instruction and AI faculty resources