Assessment Report

on

“Classify Emails as Spam or Not Spam Using Structured Metadata”

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE(AI)

By

Name: Briteshwar Kumar

Roll Number: 202401100300091

Section: B

Under the supervision of

“Shivansh Prasad”

KIET Group of Institutions, Ghaziabad

May, 2025

1. 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.

2. 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.

1. **Train-Test Split:** The dataset is split into 80% training and 20% testing sets.
2. **Model Training:** A RandomForestClassifier is trained on the training data.
3. **Model Evaluation:**
   - A confusion matrix is used to visualize classification performance.
   - Accuracy, precision, recall, and F1-score are reported.

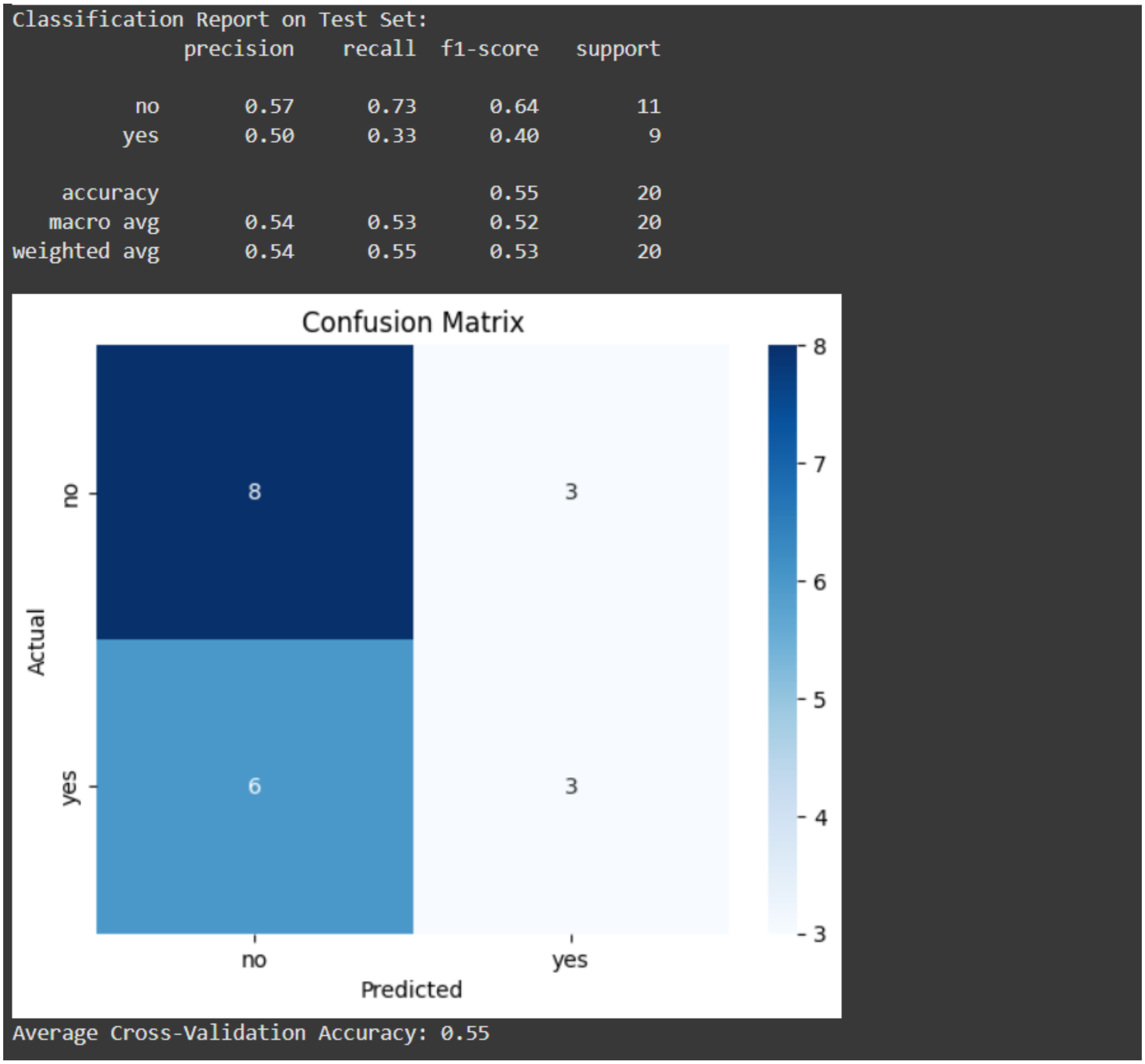
3. CODE

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix

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

# Encode target labels
label_encoder = LabelEncoder()
df['is_spam_encoded'] = label_encoder.fit_transform(df['is_spam'])  # 'no' -> 0, 'yes' -> 1

# Visualize feature distributions
sns.pairplot(df, hue='is_spam', vars=['num_links', 'num_attachments', 'sender_reputation'])
plt.suptitle("Feature Distributions by Spam Type", y=1.02)
plt.show()

# Features and labels
X = df[['num_links', 'num_attachments', 'sender_reputation']]
y = df['is_spam_encoded']

# Feature scaling
```

```
scaler = StandardScaler()
X_scaled = scaler.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)

# Initialize and train Logistic Regression model
model = LogisticRegression(random_state=42)
model.fit(X_train, y_train)

# Predict and evaluate
y_pred = model.predict(X_test)
print("Classification Report on Test Set:")
print(classification_report(y_test, y_pred, target_names=label_encoder.classes_))

# Confusion matrix
conf_matrix = confusion_matrix(y_test, y_pred)
sns.heatmap(conf_matrix, annot=True, fmt="d", cmap="Blues",
            xticklabels=label_encoder.classes_,
            yticklabels=label_encoder.classes_)
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()
```

```
# Cross-validation accuracy

cv_scores = cross_val_score(model, X_scaled, y, cv=5, scoring='accuracy')

printf("Average Cross-Validation Accuracy:", round(cv_scores.mean(), 2))
```

4. Output



```
Classification Report on Test Set:
              precision    recall  f1-score   support

          no       0.57      0.73      0.64        11
         yes       0.50      0.33      0.40         9

    accuracy                           0.55        20
   macro avg       0.54      0.53      0.52        20
weighted avg       0.54      0.55      0.53        20
```

```
Average Cross-Validation Accuracy: 0.55
```

6. References

● Dataset: Provided spam_emails.csv

● Libraries Used: ○ pandas ○ scikit-learn ○ matplotlib ○ seaborn

● Development Platform: Google Collab

● Model: Random Forest Classifier

● Guidance: Course instruction and AI faculty resources