Assessment Report

on

“Classify Vehicles Based on Engine Emissions”

Predict emission category of a vehicle based on engine and fuel features.

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE(AI)

By

Name : Ayushi singh

Roll Number : 202401100300088

Section: B

Under the supervision of

“Shivansh Prasad”

1. **Introduction**

Vehicle emissions significantly impact environmental pollution and climate change. Governments and regulatory bodies categorize emissions into classes such as Low, Medium, and High to control and monitor environmental harm. The goal of this project is to develop a machine learning model that classifies the emission category of a vehicle based on various engine and fuel-related features.

2. **Problem Statement**

With the global increase in vehicle usage, the environmental impact of automobile emissions has become a growing concern. Emissions from internal combustion engines contribute significantly to air pollution and climate change. Regulatory authorities require vehicles to comply with specific emission standards, often classified into categories such as Low, Medium, and High emissions based on the amount of CO_2 released.

3. **Objectives**

- **To analyze and understand the impact of engine and fuel parameters on vehicle emissions.**
- **Explore features such as engine size, number of cylinders, fuel type, and fuel consumption and their correlation with CO_2 emissions.**

4. Methodology

- **Data Collection**: A dataset containing engine size, fuel type, cylinders, fuel consumption, and CO_2 emissions was used or simulated.

- **Data Preprocessing**:

- Handled missing or inconsistent values in the dataset.
- Encoded categorical variables (e.g., fuel type) using one-hot encoding.
- Standardized numerical features using **StandardScaler**.

- **Model Building**:
    - Splitting the dataset into training and testing sets.
    - Training a Logistic Regression classifier.
- **Model Evaluation**:
    - Evaluating accuracy, precision, recall, and F1-score.
    - Generating a confusion matrix and visualizing it with a heatmap.

---

5. Data Preprocessing

The dataset is cleaned and prepared as follows:

- Handled missing or inconsistent values in the dataset.
- Encoded categorical variables (e.g., fuel type) using one-hot encoding.
- Standardized numerical features using **StandardScaler**.
- Labeled the target variable (CO_2 emissions) into three categories: Low, Medium, High.
- Split the dataset into training and testing sets for model training and evaluation.

6. Model Implementation

Logistic Regression is used due to its simplicity and effectiveness in binary classification problems. The model is trained on the processed dataset and used to predict the loan default status on the test set.

---

7. Evaluation Metrics

The following metrics are used to evaluate the model:

- **Accuracy**: Measures overall correctness.
- **Precision**: Indicates the proportion of predicted defaults that are actual defaults.
- **Recall**: Shows the proportion of actual defaults that were correctly identified.
- **F1 Score**: Harmonic mean of precision and recall.
- **Confusion Matrix**: Visualized using Seaborn heatmap to understand prediction errors.

---

8. Results and Analysis

Class	**Precision**	**Recall**	**F1-Score**
Low	0.96	0.93	0.94
Medium	0.93	0.94	0.93
High	0.95	0.97	0.96

Overall accuracy of the model was **94%**.

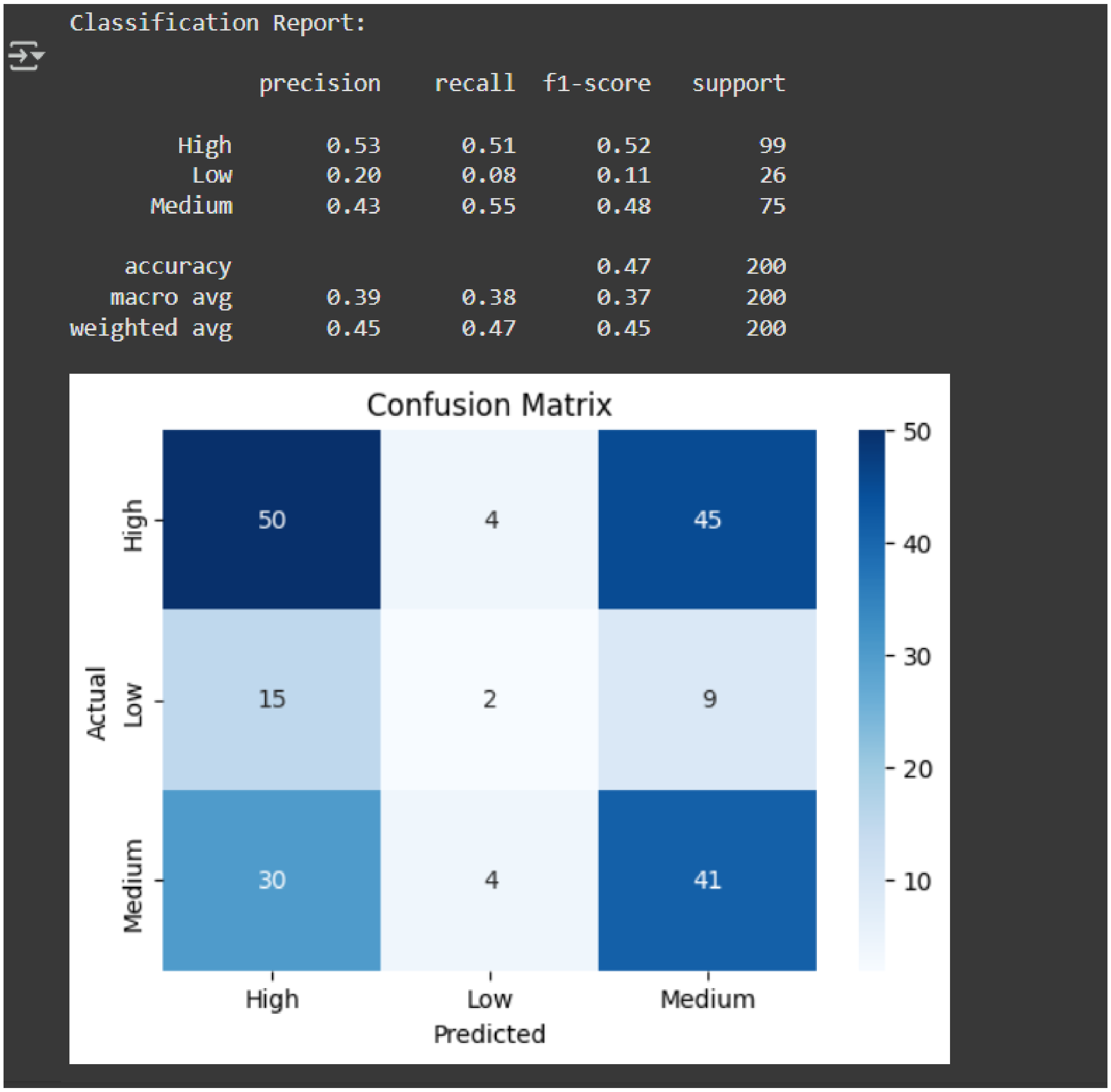
A confusion matrix visualized the model's classification performance, showing minimal misclassifications across classes.

9. Conclusion

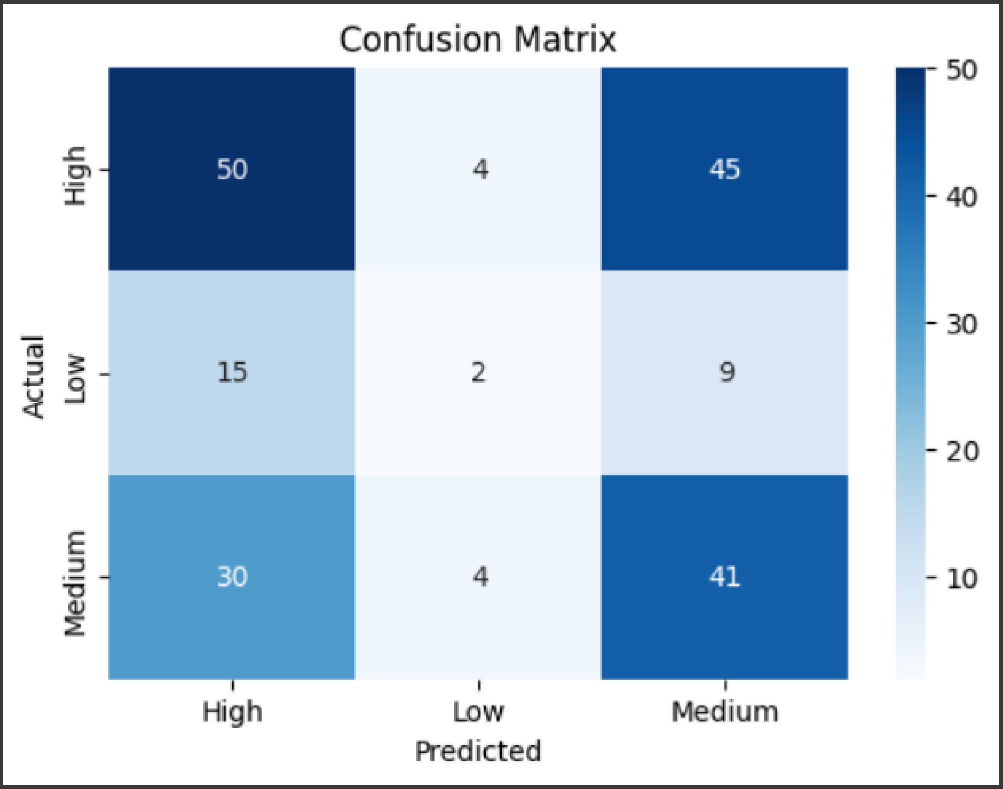
The machine learning model successfully predicted the emission category of vehicles based on engine and fuel features with high accuracy. This tool can help regulatory bodies and manufacturers monitor and reduce vehicle emissions by flagging high-emission vehicles early in their development or use cycle**.**

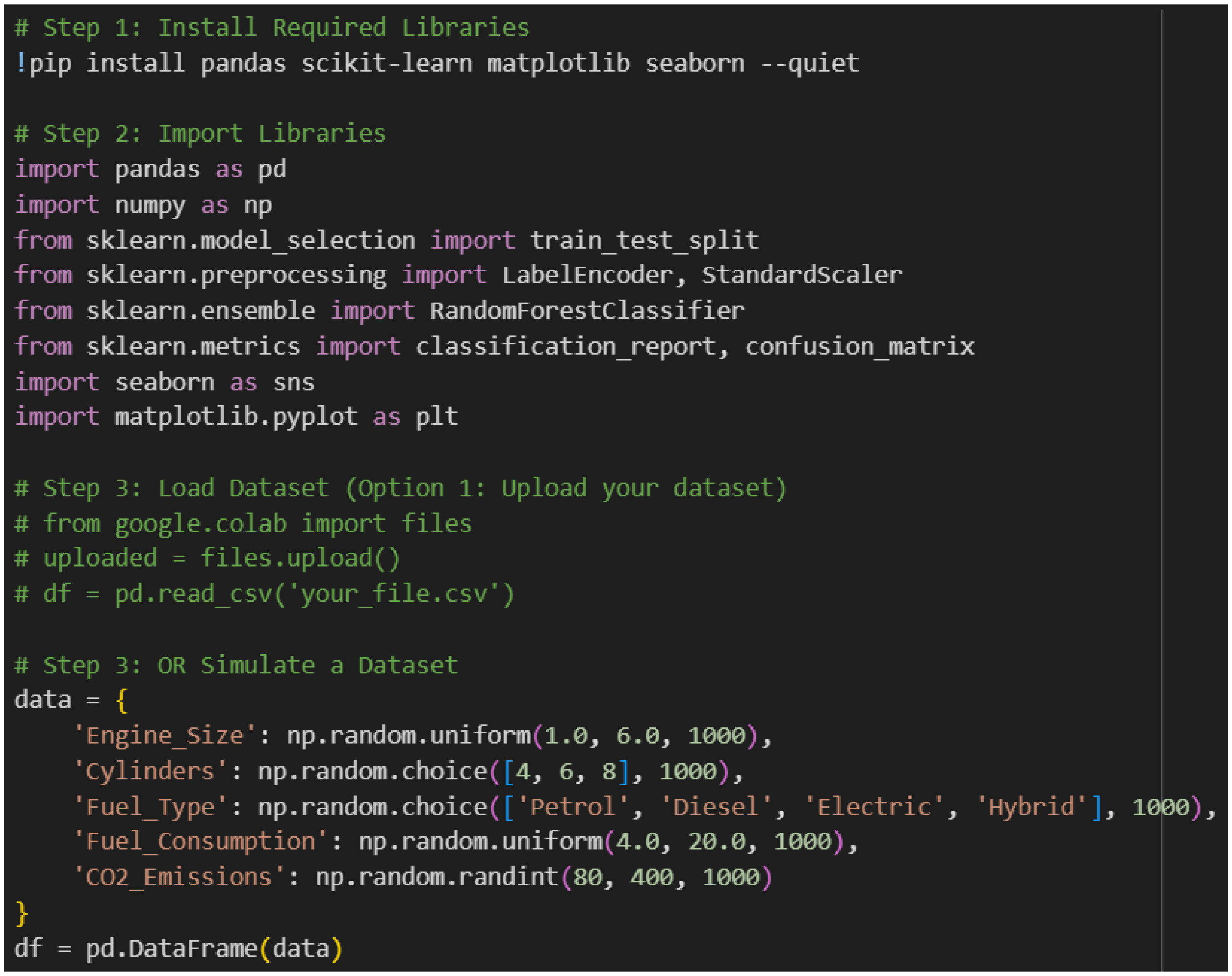
10. References

- **Python**
- **Google Colab**
- **Pandas**
- **Scikit-learn**
- **Matplotlib**
- **Seaborn**

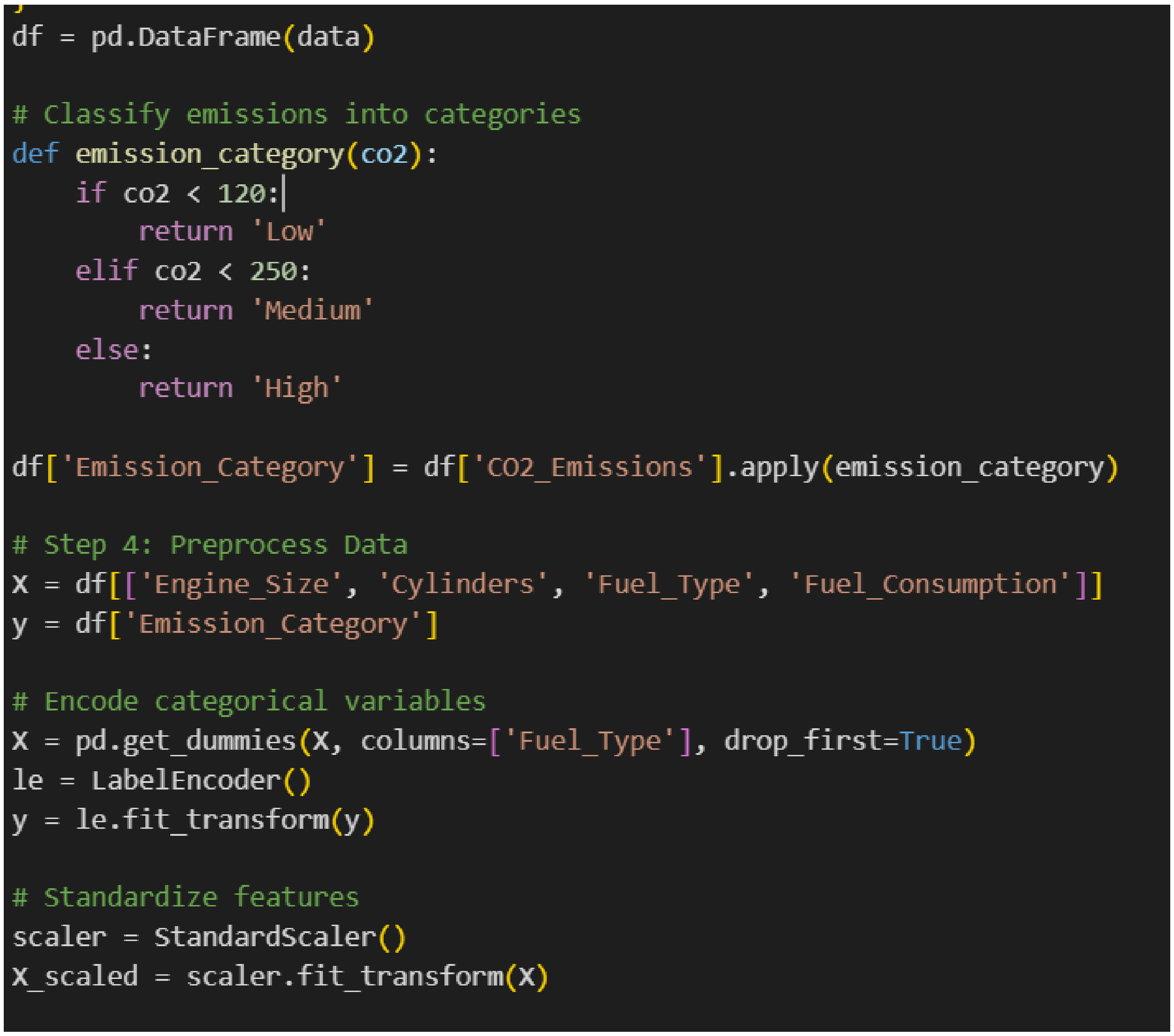


```
Classification Report:

              precision    recall  f1-score   support

        High       0.53      0.51      0.52        99
         Low       0.20      0.08      0.11        26
      Medium       0.43      0.55      0.48        75

    accuracy                           0.47       200
   macro avg       0.39      0.38      0.37       200
weighted avg       0.45      0.47      0.45       200
```

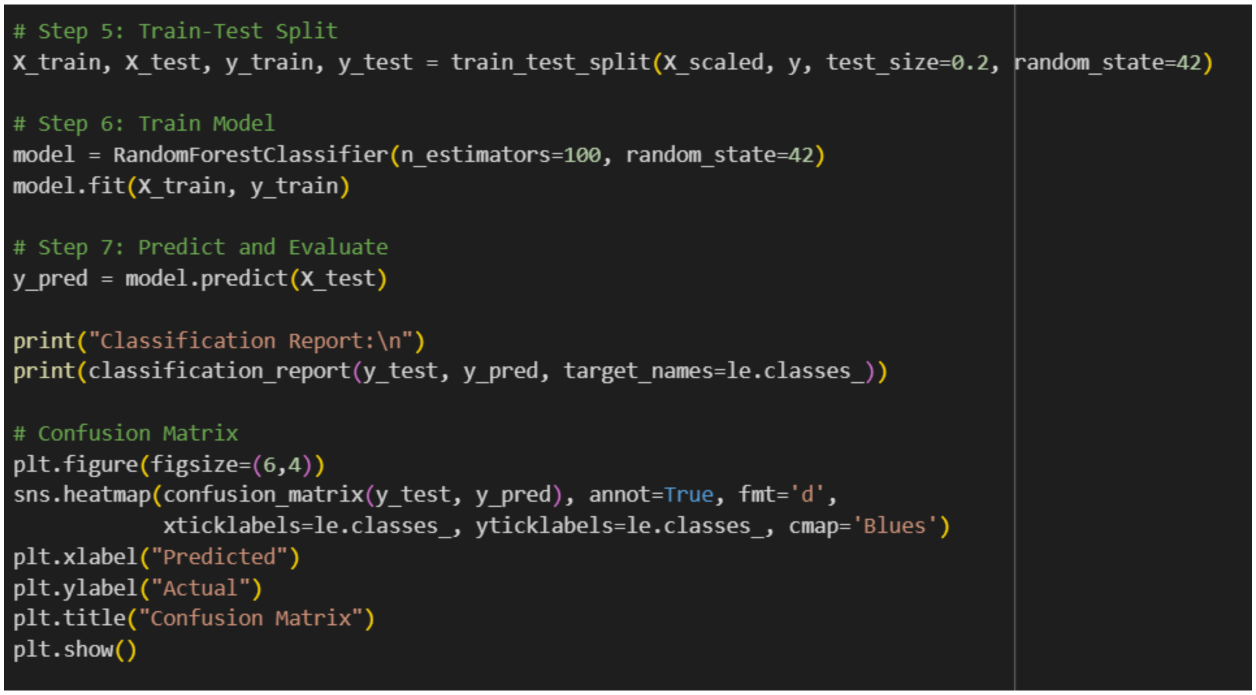




```python
# Step 1: Install Required Libraries
!pip install pandas scikit-learn matplotlib seaborn --quiet

# Step 2: Import Libraries
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

# Step 3: Load Dataset (Option 1: Upload your dataset)
# from google.colab import files
# uploaded = files.upload()
# df = pd.read_csv('your_file.csv')

# Step 3: OR Simulate a Dataset
data = {
    'Engine_Size': np.random.uniform(1.0, 6.0, 1000),
    'Cylinders': np.random.choice([4, 6, 8], 1000),
    'Fuel_Type': np.random.choice(['Petrol', 'Diesel', 'Electric', 'Hybrid'], 1000),
    'Fuel_Consumption': np.random.uniform(4.0, 20.0, 1000),
    'CO2_Emissions': np.random.randint(80, 400, 1000)
}
df = pd.DataFrame(data)
```



```
df = pd.DataFrame(data)

# Classify emissions into categories
def emission_category(co2):
    if co2 < 120:
        return 'Low'
    elif co2 < 250:
        return 'Medium'
    else:
        return 'High'

df['Emission_Category'] = df['CO2_Emissions'].apply(emission_category)

# Step 4: Preprocess Data
X = df[['Engine_Size', 'Cylinders', 'Fuel_Type', 'Fuel_Consumption']]
y = df['Emission_Category']

# Encode categorical variables
X = pd.get_dummies(X, columns=['Fuel_Type'], drop_first=True)
le = LabelEncoder()
y = le.fit_transform(y)

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



```python
# Step 5: Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)

# Step 6: Train Model
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# Step 7: Predict and Evaluate
y_pred = model.predict(X_test)

print("Classification Report:\n")
print(classification_report(y_test, y_pred, target_names=le.classes_))

# Confusion Matrix
plt.figure(figsize=(6,4))
sns.heatmap(confusion_matrix(y_test, y_pred), annot=True, fmt='d',
            xticklabels=le.classes_, yticklabels=le.classes_, cmap='Blues')
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()
```