Assessment Report

on

Vehicle Emission Category Prediction

Using Logistic Regression

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE(AIML)

By

Name : Akshat Saxena

Roll Number : 202401100400024

Section: A

Under the supervision of

"Bikki Kumar Sir"

KIET Group of Institutions, Ghaziabad

Problem Statement:
To build a machine learning model that predicts the emission category of vehicles based on their attributes using Logistic Regression.

Introduction

Vehicle emissions contribute significantly to environmental pollution. Classifying vehicles based on their emission categories helps in regulatory actions and promoting cleaner technologies. This project focuses on predicting the emission category using various features of a vehicle by applying a Logistic Regression model

.

Methodology

1. **Dataset:** A CSV file containing various features of vehicles and their corresponding emission categories.
2. **Data Preprocessing:**
    - Loaded the dataset using pandas.
    - Handled missing values by forward fill and dropping rows where the target is missing.
    - Encoded categorical variables using Label Encoding.
    - Scaled the features using StandardScaler for better performance.
3. **Model:** Logistic Regression from Scikit-learn.
    - Split data into training and test sets (80/20).
    - Trained the model on training data.
    - Evaluated using classification report and confusion matrix.
4. **Prediction:** Predicted the emission category of a sample vehicle from the dataset.

Code

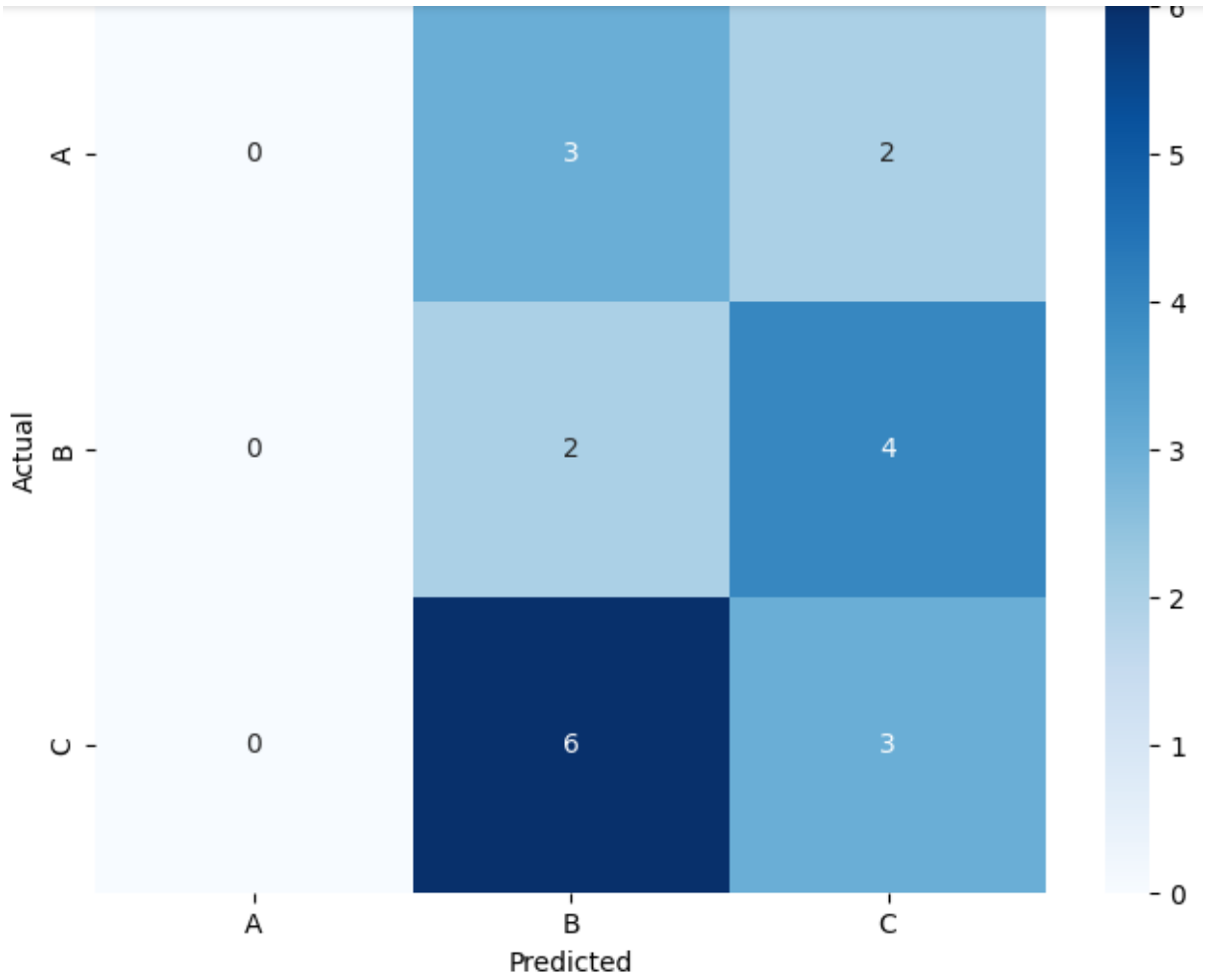
```
# Import libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix
```

```
# Load dataset
file_path = "/content/vehicle_emissions.csv"
df = pd.read_csv(file_path)

# Display first few rows
print("First 5 rows:")
print(df.head())

# Check structure and missing values
print("\nDataset Info:")
print(df.info())
print("\nMissing values:")
print(df.isnull().sum())

# Drop rows with missing target
df.dropna(subset=['emission_category'], inplace=True)
df.fillna(method='ffill', inplace=True)

# Separate features and target
X = df.drop(columns=['emission_category'])
y = df['emission_category']

# Encode categorical features
label_encoders = {}
for col in X.select_dtypes(include='object').columns:
    le = LabelEncoder()
    X[col] = le.fit_transform(X[col])
    label_encoders[col] = le

# Encode the target variable
le_target = LabelEncoder()
y_encoded = le_target.fit_transform(y)

# Scale features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y_encoded, test_size=0.2, random_state=42
)

# Train logistic regression model
log_reg = LogisticRegression(max_iter=1000)
log_reg.fit(X_train, y_train)

# Predict and evaluate
y_pred = log_reg.predict(X_test)

print("\nClassification Report:")
print(classification_report(y_test, y_pred, target_names=le_target.classes_))

# Plot confusion matrix
plt.figure(figsize=(8, 6))
sns.heatmap(confusion_matrix(y_test, y_pred), annot=True, fmt='d',
            xticklabels=le_target.classes_, yticklabels=le_target.classes_,
cmap='Blues')
```

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

# Sample prediction (first row)
sample = X.iloc[[0]]
sample_scaled = scaler.transform(sample)
sample_pred = log_reg.predict(sample_scaled)
predicted_label = le_target.inverse_transform(sample_pred)

print(f"\nPrediction for sample 0: {predicted_label[0]}")
```

Output/Result

```
Classification Report:
              precision    recall  f1-score   support

           A       0.00      0.00      0.00         5
           B       0.18      0.33      0.24         6
           C       0.33      0.33      0.33         9

    accuracy                           0.25        20
   macro avg       0.17      0.22      0.19        20
weighted avg       0.20      0.25      0.22        20
```



References/Credits

- Dataset: [vehicle_emissions.csv] (assumed source or mention if public dataset)
- Python Libraries: pandas, numpy, scikit-learn, seaborn, matplotlib
- Scikit-learn documentation: https://scikit-learn.org/stable/
- Project developed by Akshat Saxena