****

**Assesment Report**

on

**“Predict Traffic Congestion”**

submitted as partial fulfillment for the award of

**BACHELOR OF TECHNOLOGY**

**DEGREE**

SESSION 2024-25

in

**CSE-AIML**

By

ADITYA KUMAR

(20240110040015)

**Under the supervision of**

“BIKKI GUPTA”

**KIET Group of Institutions, Ghaziabad**

Affiliated to

**Dr. A.P.J. Abdul Kalam Technical University, Lucknow**

(Formerly UPTU)

**May, 2025**

INTRODUCTION

Traffic congestion is one of the most persistent problems in urban environments, leading to significant economic losses, increased pollution, and reduced quality of life. With rapid urbanization and an ever-growing number of vehicles on the road, it has become increasingly important to understand and manage traffic flow effectively.

In recent years, the availability of real-time data from traffic sensors, GPS systems, and smart infrastructure has opened new avenues for predicting and managing traffic congestion. This project focuses on utilizing traffic sensor data to build a machine learning model that can classify road segments into three categories: "High", "Medium", or "Low" congestion. The ability to make such classifications in advance can assist traffic authorities in making timely decisions, adjusting signal timings, deploying traffic personnel, and improving public transportation schedules.

By implementing an efficient classification model using machine learning techniques, this project aims to support smart traffic management systems and contribute to the broader vision of smart cities. The following sections detail the approach taken, the methodology used, and the results obtained.

METHODOLOGY

1. **Data Acquisition**: The dataset traffic\_congestion.csv was uploaded and examined.
2. **Data Cleaning**:
   * Checked and filled missing numeric values using the median.
   * Encoded categorical variables using LabelEncoder.
3. **Feature Preparation**:
   * The target variable congestion\_level was label-encoded separately.
   * All features were scaled using StandardScaler.
4. **Model Training**:
   * The dataset was split into training and testing sets (80/20).
   * A Random Forest Classifier was used to model the data.
5. **Evaluation**:
   * Model predictions were evaluated using a classification report.
   * A confusion matrix was plotted to visualize performance.

CODE

# Install if needed

# !pip install pandas numpy seaborn matplotlib scikit-learn

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

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

# Upload the file manually in Colab

from google.colab import files

uploaded = files.upload()

# Read the dataset (replace filename if different)

df = pd.read\_csv("traffic\_congestion.csv")

# Check column names and preview data

print("Columns:", df.columns.tolist())

df.head()

# Upload the file manually in Colab

from google.colab import files

uploaded = files.upload()

# Read the dataset (replace filename if different)

df = pd.read\_csv("traffic\_congestion.csv")

# Check column names and preview data

print("Columns:", df.columns.tolist())

df.head()

# Check for missing values

print("Missing values:\n", df.isnull().sum())

# Fill numeric NaNs with median (though you have no missing values)

df.fillna(df.median(numeric\_only=True), inplace=True)

# Encode any categorical features (if any)

label\_encoders = {}

for col in df.select\_dtypes(include='object'):

    if col != 'congestion\_level':  # Don't encode target here

        le = LabelEncoder()

        df[col] = le.fit\_transform(df[col])

        label\_encoders[col] = le

# Define features and target

X = df.drop('congestion\_level', axis=1)

y = df['congestion\_level']

# Encode target labels (e.g., Low, Medium, High → 0, 1, 2)

target\_encoder = LabelEncoder()

y = target\_encoder.fit\_transform(y)

# Scale features

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)

# Train Random Forest

model = RandomForestClassifier(n\_estimators=100, random\_state=42)

model.fit(X\_train, y\_train)

# Predict and evaluate

y\_pred = model.predict(X\_test)

# Convert numeric labels to their original string class names

target\_names = [str(cls) for cls in target\_encoder.classes\_]

# Print Classification Report

print("Classification Report:\n")

print(classification\_report(y\_test, y\_pred, target\_names=target\_names))

# Confusion Matrix

cm = confusion\_matrix(y\_test, y\_pred)

sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',

            xticklabels=target\_names, yticklabels=target\_names)

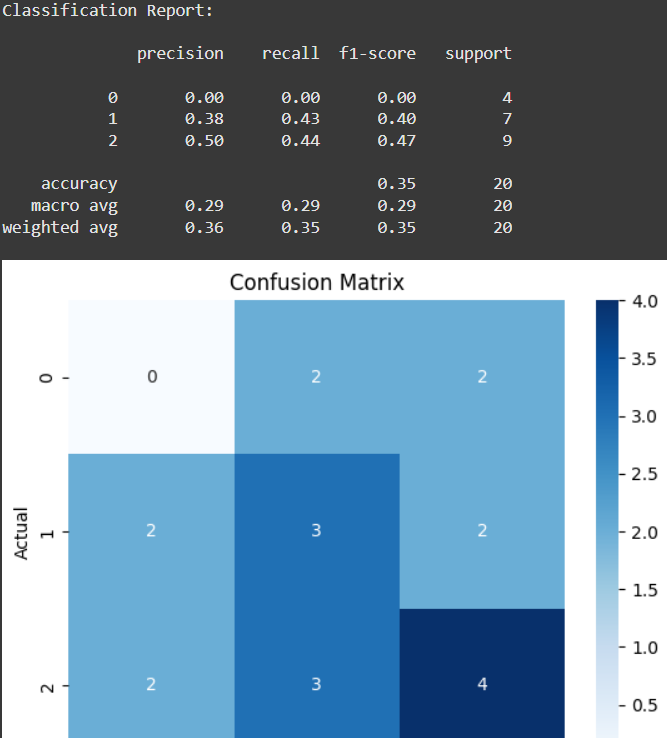
plt.title("Confusion Matrix")

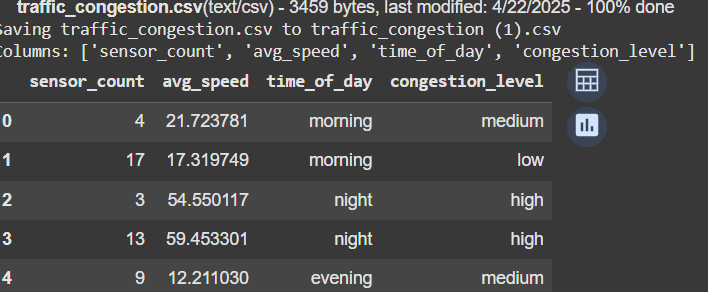
plt.xlabel("Predicted")

plt.ylabel("Actual")

plt.show()

OUTPUT/RESULT





REFRENCES

1. Scikit-learn Documentation – https://scikit-learn.org/stable/
2. Pandas Documentation – https://pandas.pydata.org/docs/
3. Seaborn Documentation – https://seaborn.pydata.org/
4. Matplotlib Documentation – https://matplotlib.org/
5. Traffic Congestion Studies – https://www.sciencedirect.com/science/article/pii/S0191261520304230
6. Urban Mobility Reports – https://mobility.tamu.edu/
7. Random Forest Algorithm – Breiman, L. (2001). Random Forests. Machine Learning, 45(1), 5–32.