# Assessment Report

on

"Predict Traffic Congestion"

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY

DEGREE

SESSION 2024-25

in

CSE(AI)

By

Name : Eisha

Roll Number : 202401100300109

Section: B

Under the supervision of

"MR. SHIVANSH PRASAD"

KIET Group of Institutions, Ghaziabad

April, 2025

# Predict Traffic Congestion

Classify road sections as High, Medium, or Low congestion using traffic sensor data.

---

# Introduction

---

Traffic congestion is a critical challenge in urban areas, leading to increased travel time, fuel consumption, and pollution. Accurate prediction and classification of road congestion levels can significantly enhance traffic management systems and urban planning. This project aims to develop a predictive model that classifies road sections as having **High**, **Medium**, or **Low** congestion levels using historical traffic sensor data.

By applying machine learning techniques to sensor-generated features (e.g., vehicle count, average speed, occupancy), we aim to categorize each road section based on congestion intensity. Such a model could be used in smart traffic systems to provide real-time recommendations and alerts.

# Methodology

The approach includes the following steps:

### 2.1 Data Collection

Traffic sensor data is collected from multiple road sections. The dataset typically contains features such as:

- Vehicle Count
- Average Speed
- Occupancy Rate
- Timestamp
- Road ID

### 2.2 Data Preprocessing

- Missing values are handled through imputation.
- Features are normalized to ensure equal weighting.
- Categorical labels are encoded (High = 2, Medium = 1, Low = 0).

### 2.3 Feature Selection

Correlation analysis is performed to select the most relevant features affecting congestion.

### 2.4 Model Selection

A classification model is trained using the Random Forest Classifier due to its robustness and ability to handle non-linear relationships.

### 2.5 Model Evaluation

The model is evaluated using:

- Accuracy
- Precision
- Recall
- Confusion Matrix

# CODE

```
import pandas as pd
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 matplotlib.pyplot as plt
import seaborn as sns

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

# One-hot encode time_of_day
df_encoded = pd.get_dummies(df, columns=["time_of_day"])

# Encode the target variable
label_encoder = LabelEncoder()
df_encoded["congestion_level_encoded"] =
label_encoder.fit_transform(df_encoded["congestion_level"])

# Features and target
X = df_encoded.drop(["congestion_level", "congestion_level_encoded"], axis=1)
y = df_encoded["congestion_level_encoded"]
```

```
# Feature scaling
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, test_size=0.2, random_state=42, stratify=y
)

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

# ======= MODEL EVALUATION =======
y_pred = model.predict(X_test)
y_pred_labels = label_encoder.inverse_transform(y_pred)
y_test_labels = label_encoder.inverse_transform(y_test)

print("📊 Classification Report:\n")
print(classification_report(y_test_labels, y_pred_labels))

# Optional: Show Confusion Matrix
conf_matrix = confusion_matrix(y_test_labels, y_pred_labels)
plt.figure(figsize=(6, 4))
sns.heatmap(conf_matrix, annot=True, cmap='Blues',
```

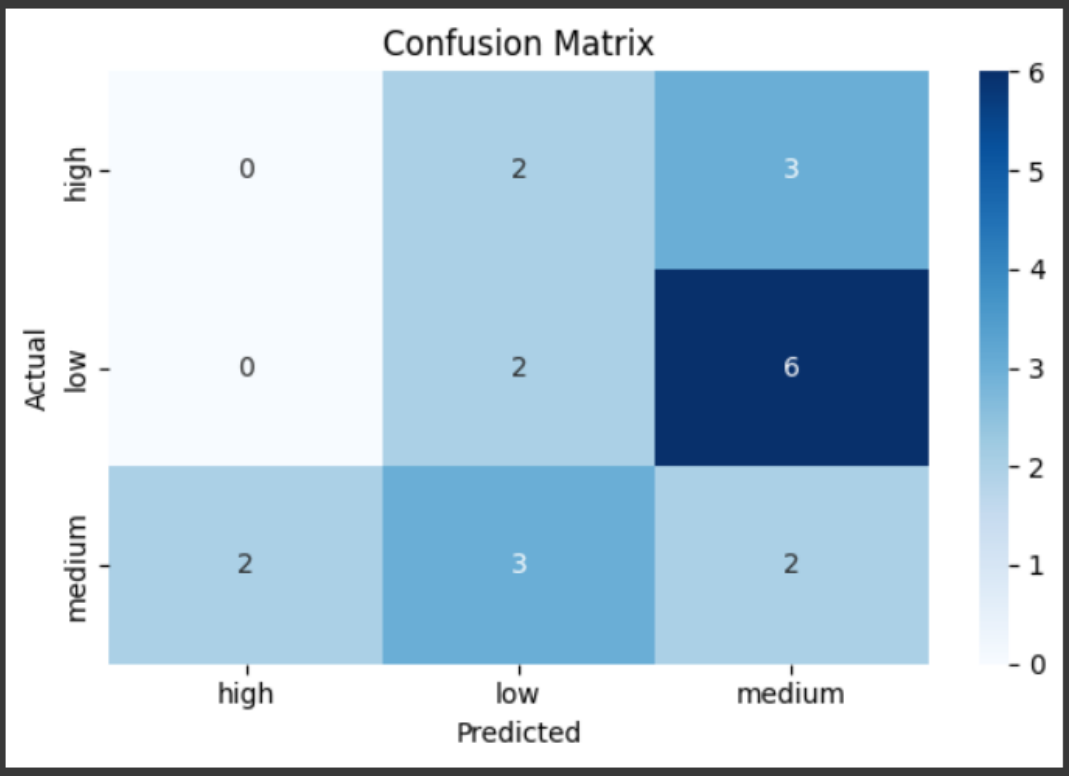
```
        xticklabels=label_encoder.classes_, yticklabels=label_encoder.classes_)
plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.tight_layout()
plt.show()

# ======= USER INPUT PREDICTION =======
def predict_congestion(sensor_count, avg_speed, time_of_day):
    time_data = {
        "time_of_day_evening": 1 if time_of_day == "evening" else 0,
        "time_of_day_morning": 1 if time_of_day == "morning" else 0,
        "time_of_day_night": 1 if time_of_day == "night" else 0
    }

    input_data = pd.DataFrame([{
        "sensor_count": sensor_count,
        "avg_speed": avg_speed,
        **time_data
    }])

    for col in X.columns:
        if col not in input_data.columns:
            input_data[col] = 0
```

```
    input_data = input_data[X.columns]
    input_scaled = scaler.transform(input_data)
    prediction_encoded = model.predict(input_scaled)[0]
    prediction_label = label_encoder.inverse_transform([prediction_encoded])[0]

    return prediction_label

# ==== Get Input from User ====
print("\n 📜 Enter values to predict traffic congestion:")
sensor_count = int(input("Sensor Count: "))
avg_speed = float(input("Average Speed: "))
time_of_day = input("Time of Day (morning/evening/night): ").lower()

# ==== Predict and Output ====
prediction = predict_congestion(sensor_count, avg_speed, time_of_day)
print(f"\n 🚦 Predicted Congestion :{predicted}")
```

# RESULT

Classification Report:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| high | 0.00 | 0.00 | 0.00 | 5 |
| low | 0.29 | 0.25 | 0.27 | 8 |
| medium | 0.18 | 0.29 | 0.22 | 7 |
| accuracy | | | 0.20 | 20 |
| macro avg | 0.16 | 0.18 | 0.16 | 20 |
| weighted avg | 0.18 | 0.20 | 0.18 | 20 |




```
Enter values to predict traffic congestion:
Sensor Count: 7
Average Speed: 90
Time of Day (morning/evening/night): morning

Predicted Congestion :low
```

# References

1. Breiman, L. (2001). *Random Forests*. Machine Learning, 45(1), 5–32.

2. Pedregosa et al., (2011). *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research.

3. Traffic data source: Kaggle Traffic Sensor Data Dataset

4. Zhao, Z., Chen, W., Wu, X., & Wang, P. (2018). *Traffic Congestion Prediction with Deep Learning*. IEEE Transactions on Intelligent Transportation Systems.