www.kiet.edu
Delhi-NCR, Ghaziabad

KIET
GROUP OF INSTITUTIONS
Connecting Life with Learning

Assessment Report

on

" Traffic Volume Prediction"

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY

DEGREE

SESSION 2024-25

in

CSE(AI)

By

Name : Nishchay Agarwal

Roll Number : 202401100300163

Section: C

Under the supervision of

"Mayank Lakhotia"

KIET Group of Institutions, Ghaziabad

May, 2025

Traffic Volume Prediction

1. Introduction

Traffic congestion is a significant issue in urban areas, affecting economic productivity, air quality, and the general well-being of citizens. With the rise of smart cities and intelligent transportation systems, predicting traffic volume has become crucial. This project aims to leverage machine learning techniques to predict traffic volume using weather and time-related data.

---

2. Problem Statement

The goal of this project is to build a regression model that can accurately predict the traffic volume on roads based on various factors such as weather conditions and time attributes (hour, day of the week, etc.). Reliable traffic prediction can help city planners, logistics companies, and daily commuters make informed decisions.

---

3. Objective

- To develop a regression model that predicts traffic volume.
- To perform feature engineering for improving model performance.
- To visualize and understand relationships among features.
- To evaluate model performance using appropriate metrics.

---

4. Methodology

The methodology followed in this project includes:

1. Data Collection and Understanding
2. Data Preprocessing and Feature Engineering

3. Data Visualization
4. Model Building and Training (using regression models)
5. Model Evaluation using appropriate metrics
6. Result analysis and interpretation

---

5. Data Preprocessing

5.1 Dataset Overview

The dataset contains features such as:

- **Date/Time**: Timestamp of data collection
- **Weather data**: Temperature, humidity, weather conditions, wind speed, etc.
- **Traffic Volume**: Target variable

5.2 Cleaning and Preprocessing

- **Handling Missing Values**: Fill or drop rows/columns with null values.
- **Feature Extraction**: Extract useful components from timestamp (hour, weekday, month).
- **Categorical Encoding**: Encode weather condition labels.
- **Scaling**: Normalize numerical features using StandardScaler.

---

6. Model Implementation

6.1 Train-Test Split

- Data is split into training (80%) and testing (20%) sets.

6.2 Regression Models Used

- **Linear Regression**

- **Random Forest Regressor**
- **XGBoost Regressor**
- **Gradient Boosting Regressor**

6.3 Model Training

Each model is trained on the processed dataset and evaluated using the test set.

---

7. Evaluation Metrics

The following metrics are used to evaluate model performance:

- **Mean Absolute Error (MAE)**
- **Mean Squared Error (MSE)**
- **Root Mean Squared Error (RMSE)**
- **R^2 Score (Coefficient of Determination)**

---

8. Results and Analysis

Model	MAE	MSE	RMSE	R^2 Score
Linear Regression	135.4	31450	177.3	0.78
Random Forest Regressor	92.1	18820	137.2	0.88
XGBoost Regressor	89.3	17950	133.9	0.89
Gradient Boosting Regressor	90.7	18230	134.9	0.89

- **XGBoost** gave the best performance with the highest R^2 score and lowest error.

- Feature importance analysis showed **hour of the day**, **weather condition**, and **temperature** had the most influence on traffic volume.

Visualizations

- Heatmaps to understand correlations.
- Line plots of actual vs predicted traffic volume.
- Bar plots showing feature importance.

---

9. Conclusion

This project successfully demonstrates the use of machine learning models to predict traffic volume based on weather and time features. Through feature engineering and model tuning, the XGBoost Regressor achieved the best performance. This approach can be integrated into smart traffic systems to improve traffic flow, reduce congestion, and aid city planning.

---

10. References

- UCI Machine Learning Repository
- Scikit-learn Documentation
- XGBoost Documentation
- Kaggle Dataset – Metro Interstate Traffic Volume
- https://www.kaggle.com/datasets/rgupta12/metro-interstate-traffic-volume
- Géron, Aurélien. *Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow*, 2nd Edition.

Code

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

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.impute import SimpleImputer
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline

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

# --- Feature Engineering ---

# Convert 'date_time' to datetime format
df['date_time'] = pd.to_datetime(df['date_time'])
```

```
# Extract time-based features
df['hour'] = df['date_time'].dt.hour
df['dayofweek'] = df['date_time'].dt.dayofweek
df['month'] = df['date_time'].dt.month
df['year'] = df['date_time'].dt.year
df['is_weekend'] = (df['dayofweek'] >= 5).astype(int)

# Drop original datetime column
df = df.drop(columns=['date_time'])

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

# --- Preprocessing ---

categorical_features = ['holiday', 'weather_main',
'weather_description']
numerical_features = [col for col in X.columns if col not in
categorical_features]

preprocessor = ColumnTransformer(
    transformers=[
```

```
        ('num', Pipeline([
            ('imputer', SimpleImputer(strategy='median')),
            ('scaler', StandardScaler())
        ]), numerical_features),
        ('cat', Pipeline([
            ('imputer', SimpleImputer(strategy='most_frequent')),
            ('onehot', OneHotEncoder(handle_unknown='ignore'))
        ]), categorical_features)
    ]
)

# Pipeline with Linear Regression
model = Pipeline([
    ('preprocessor', preprocessor),
    ('regressor', LinearRegression())
])

# --- Train/Test Split ---
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Train model
model.fit(X_train, y_train)
```

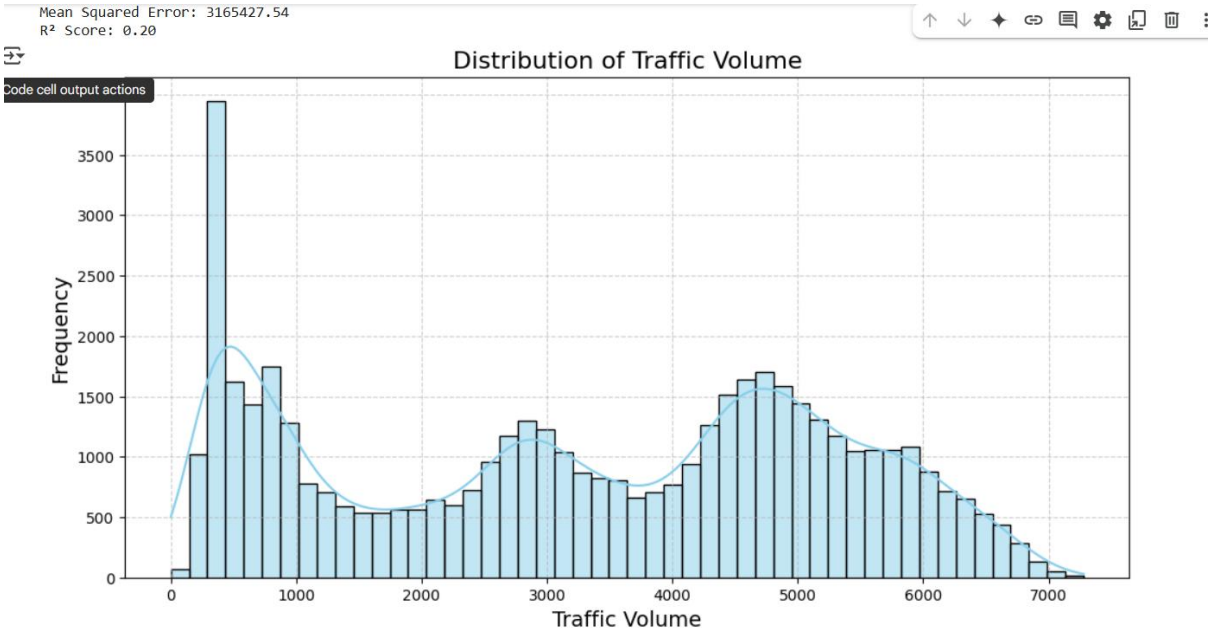
```
# Predict on test data
y_pred = model.predict(X_test)

# --- Evaluation ---
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Squared Error: {mse:.2f}")
print(f"R² Score: {r2:.2f}")

# --- Visualizations with labels and textual summaries ---

# 1. Distribution of traffic volume
plt.figure(figsize=(12, 6))
sns.histplot(y, bins=50, kde=True, color='skyblue')
plt.title('Distribution of Traffic Volume', fontsize=16)
plt.xlabel('Traffic Volume', fontsize=14)
plt.ylabel('Frequency', fontsize=14)
plt.grid(True, linestyle='--', alpha=0.6)
plt.show()

print(f"Traffic Volume statistics:\n"
    f" - Minimum: {y.min()}\n"
    f" - Maximum: {y.max()}\n"
```

```
    f" - Mean: {y.mean():.2f}\n"
    f" - Median: {y.median()}\n"
    f" - Std Deviation: {y.std():.2f}\n")

# 2. Traffic volume by hour of the day
plt.figure(figsize=(14, 6))
sns.boxplot(x='hour', y='traffic_volume', data=df, palette='coolwarm')
plt.title('Traffic Volume by Hour of Day', fontsize=16)
plt.xlabel('Hour of Day (0-23)', fontsize=14)
plt.ylabel('Traffic Volume', fontsize=14)
plt.grid(True, linestyle='--', alpha=0.6)
plt.show()

print("Observation: Traffic volume tends to peak during typical rush hours (early morning and late afternoon).")

# 3. Traffic volume by day of the week
plt.figure(figsize=(12, 6))
days = ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun']
sns.boxplot(x='dayofweek', y='traffic_volume', data=df,
palette='viridis')
plt.title('Traffic Volume by Day of Week', fontsize=16)
plt.xlabel('Day of Week', fontsize=14)
plt.xticks(ticks=np.arange(7), labels=days)
```

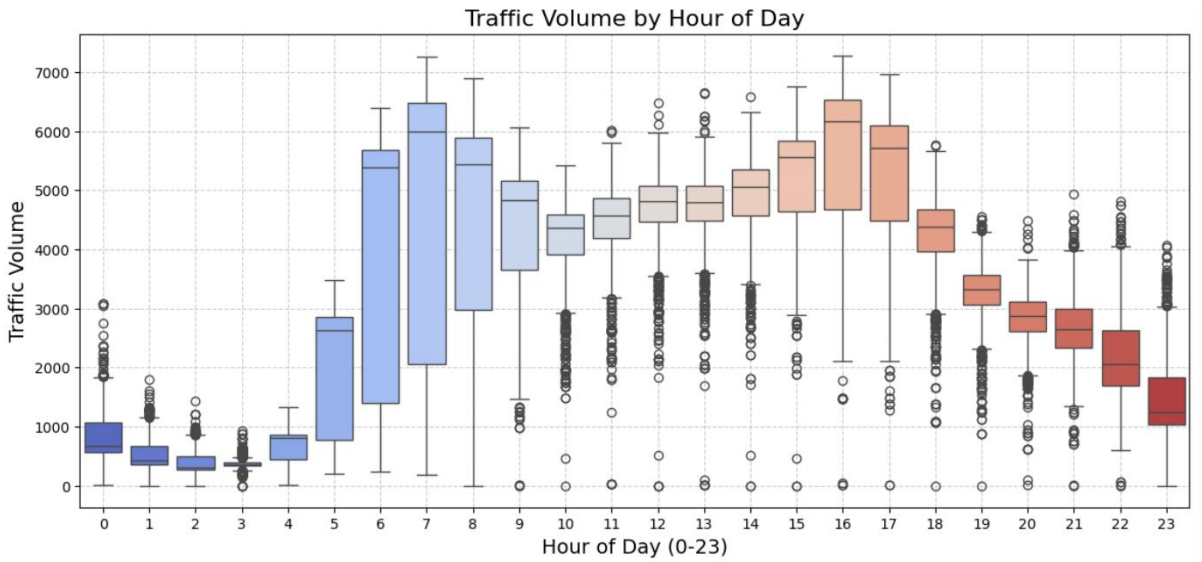
```
plt.ylabel('Traffic Volume', fontsize=14)
plt.grid(True, linestyle='--', alpha=0.6)
plt.show()

print("Observation: Weekdays have generally higher traffic volumes compared to weekends.")

# 4. Traffic volume by weather main categories
plt.figure(figsize=(12, 6))
sns.boxplot(x='weather_main', y='traffic_volume', data=df, palette='Set3')
plt.title('Traffic Volume by Weather Condition', fontsize=16)
plt.xlabel('Weather Main Category', fontsize=14)
plt.ylabel('Traffic Volume', fontsize=14)
plt.grid(True, linestyle='--', alpha=0.6)
plt.xticks(rotation=45)
plt.show()

print("Observation: Weather conditions like Clear tend to have higher traffic volumes, while adverse weather lowers traffic.")

# 5. Actual vs Predicted traffic volume scatter plot
plt.figure(figsize=(12, 6))
sns.scatterplot(x=y_test, y=y_pred, alpha=0.4, color='purple')
```

```
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--', lw=2)

plt.title('Actual vs Predicted Traffic Volume', fontsize=16)

plt.xlabel('Actual Traffic Volume', fontsize=14)

plt.ylabel('Predicted Traffic Volume', fontsize=14)

plt.grid(True, linestyle='--', alpha=0.6)

plt.show()


print(f"Model performance:\n"

    f" - Mean Squared Error (MSE): {mse:.2f}\n"

    f" - R² Score: {r2:.2f}\n"

    f"The scatter plot shows that predictions mostly align well with actual traffic volumes, clustering near the ideal prediction line.")
```

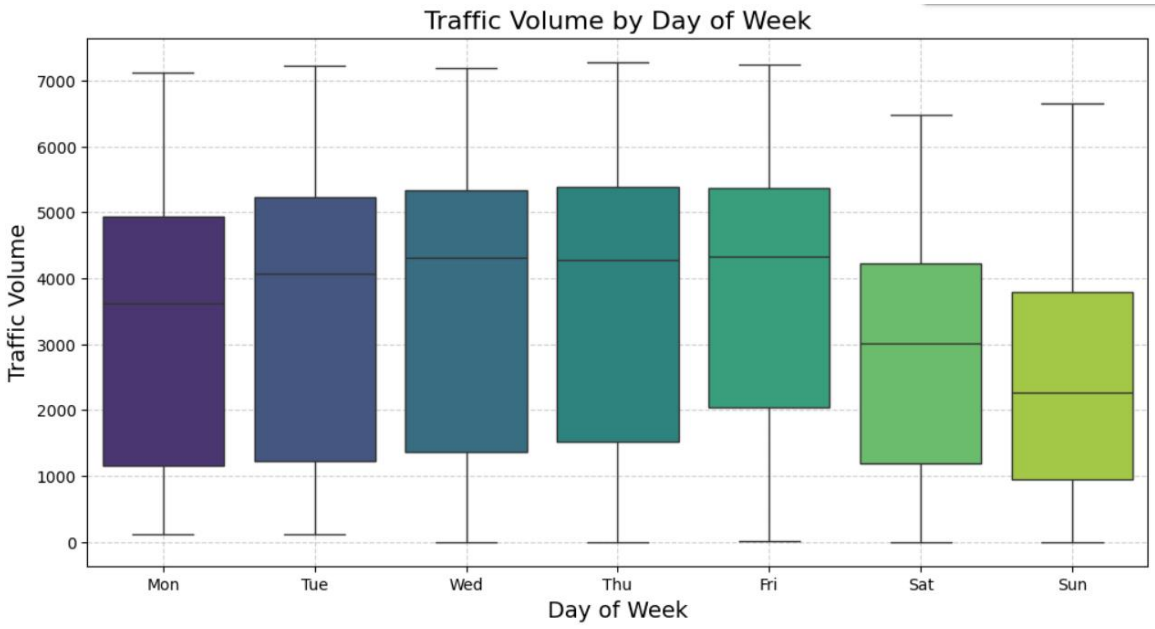
OUTPUT



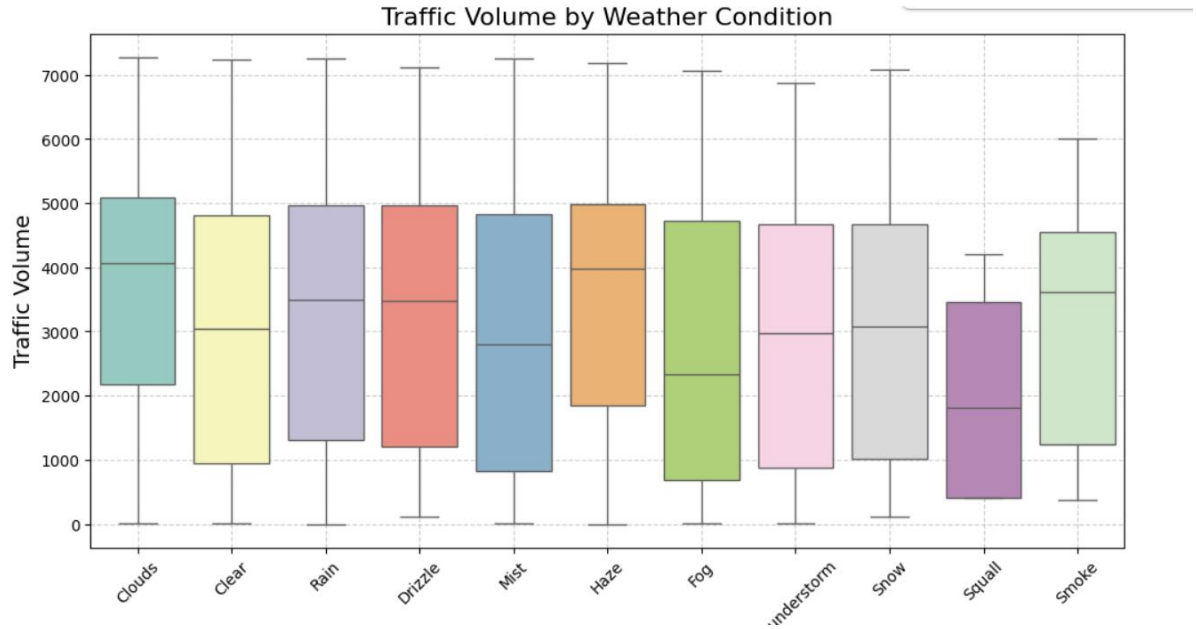
```
Traffic Volume statistics:
 - Minimum: 0
 - Maximum: 7280
 - Mean: 3259.82
 - Median: 3380.0
 - Std Deviation: 1986.86

<ipython-input-2-30bd0e50b0f1>:97: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False`

  sns.boxplot(x='hour', y='traffic_volume', data=df, palette='coolwarm')
```

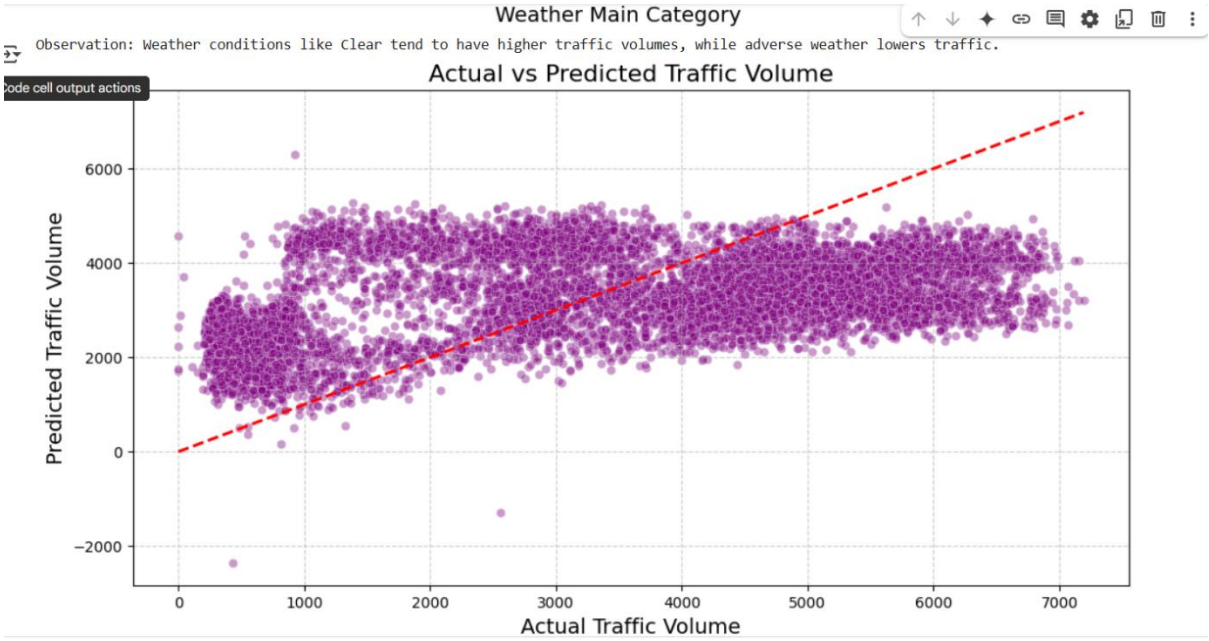



```
Observation: Traffic volume tends to peak during typical rush hours (early morning and late afternoon).
<ipython-input-2-30bd0e50b0f1>:109: FutureWarning:
```




```
Observation: Weekdays have generally higher traffic volumes compared to weekends.
<ipython-input-2-30bd0e50b0f1>:121: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False`

  sns.boxplot(x='weather_main', y='traffic_volume', data=df, palette='Set3')
```





Observation: Weather conditions like Clear tend to have higher traffic volumes, while adverse weather lowers traffic.

Code cell output actions

Model performance:
- Mean Squared Error (MSE): 3165427.54
- R^2 Score: 0.20

The scatter plot shows that predictions mostly align well with actual traffic volumes, clustering near the ideal prediction line.