KIET GROUP OF INSTITUTIONS

Connecting Life with Learning

www.kiet.edu

Delhi-NCR, Ghaziabad

Assessment Report

on

"Traffic Volume Prediction"

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY

DEGREE

SESSION 2024-25

in

CSE(AI) B

By

Gitika Pal (202401100300112)

Ayaan Vashu (202401100300082)

Divyanshi Tyagi (202401100300106)

Deepanshu Pal (202401100300095)

Under the supervision of

"Shivansh Prashad"

Introduction

🚦 Traffic Volume Prediction Based on Weather and Time

In urban cities like Bangalore, traffic congestion is a critical issue impacting daily life, environment, and economy. Predicting traffic volume can help in better urban planning, smart traffic signal control, and optimized public transport scheduling.

This project focuses on building a **regression model** that predicts **traffic volume** based on weather conditions and time features (like month and weekday). We use a real-world dataset containing traffic and weather metrics from various areas in Bangalore.

Key objectives:

- Predict traffic volume using machine learning.
- Understand which features most influence traffic.
- Enable area-specific predictions for better planning.

Methodology

The following methodology was used:

- **1. Data Loading and Cleaning**
  - The dataset was loaded using pandas.
  - Missing values were handled by removing rows with null critical entries (Date, Traffic Volume, Area Name).
  - The 'Date' column was converted to datetime format.
- **2. Feature Engineering**
  - Extracted time-based features: **month** and **weekday**.
  - Applied one-hot encoding to convert categorical **Weather Conditions** into numerical features.
  - Final features included traffic indicators (e.g., **Average Speed**, **Congestion Level**), weather, and time data.
- **3. Model Building**
  - Features were scaled using StandardScaler.
  - A **Random Forest Regressor** was trained on 80% of the dataset.
  - Model evaluation was done using **R^2 Score** and **Mean Absolute Error**.
- **4. Visualization**
  - Plotted actual vs. predicted traffic volume to check prediction spread.
  - Created a feature importance chart to highlight which factors most affect traffic.
- **5. Area-Specific Prediction**
  - Enabled user to input an area name.
  - Predicted traffic volume based on average conditions of that area.
  - Suggested alternatives if area name was not found.

Code

```
import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import difflib

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler

from sklearn.ensemble import RandomForestRegressor

from sklearn.metrics import mean_absolute_error, r2_score


# Load dataset

df = pd.read_csv('/content/drive/MyDrive/Banglore_traffic_Dataset.csv')


# Convert 'Date' to datetime and drop missing key values

df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

df.dropna(subset=['Date', 'Traffic Volume', 'Area Name'], inplace=True)


# Extract time-based features

df['month'] = df['Date'].dt.month

df['weekday'] = df['Date'].dt.weekday


# One-hot encode weather

df = pd.get_dummies(df, columns=['Weather Conditions'], drop_first=True)


# Define feature columns dynamically

base_features = [

  'Average Speed', 'Travel Time Index', 'Congestion Level',

  'Road Capacity Utilization', 'Environmental Impact',

  'Public Transport Usage', 'month', 'weekday'

]

weather_features = [col for col in df.columns if col.startswith('Weather Conditions_')]

features = base_features + weather_features

target = 'Traffic Volume'


# Drop missing data for model training
```

```
df = df[features + ['Area Name', 'Date', target]].dropna()


# Split and scale data
X = df[features]
y = df[target]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)


scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)


# Train Random Forest model
model = RandomForestRegressor(random_state=42)
model.fit(X_train_scaled, y_train)


# Evaluate model
y_pred = model.predict(X_test_scaled)
print("\n📈 Model Evaluation Metrics:")
print("Mean Absolute Error:", round(mean_absolute_error(y_test, y_pred), 2))
print("R² Score:", round(r2_score(y_test, y_pred), 4))


# Plot: Actual vs Predicted
plt.figure(figsize=(10, 6))
plt.scatter(y_test, y_pred, alpha=0.3, color='blue')
plt.xlabel("Actual Traffic Volume")
plt.ylabel("Predicted Traffic Volume")
plt.title("Actual vs Predicted Traffic Volume")
plt.grid(True)
plt.show()


# Plot: Feature Importance
importances = model.feature_importances_
plt.figure(figsize=(10, 6))
sns.barplot(x=importances, y=features)
plt.title("Feature Importance (Random Forest)")
plt.xlabel("Importance")
```

```
plt.ylabel("Feature")
plt.show()

# 🚦 Area-specific prediction
area_input = input("\nEnter an area name (e.g., Koramangala, Whitefield, M.G. Road): ").strip()
area_df = df[df['Area Name'].str.lower() == area_input.lower()]

if area_df.empty:
    print(f"\n❌ No data found for area: '{area_input}'.")

    # Suggest close matches
    possible_areas = df['Area Name'].dropna().unique()
    close_matches = difflib.get_close_matches(area_input, possible_areas, n=3, cutoff=0.5)
    if close_matches:
        print(" ❓ Did you mean:", ", ".join(close_matches))
else:
    # Predict traffic volume
    area_features = area_df[features].mean().values.reshape(1, -1)
    area_features_scaled = scaler.transform(area_features)
    predicted_volume = model.predict(area_features_scaled)[0]

    print(f"\n✅ Predicted Traffic Volume for *{area_input}*: {int(predicted_volume)} vehicles")
    actual_avg = area_df['Traffic Volume'].mean()
    print(f" 📊 Actual Average Volume in data: {int(actual_avg)} vehicles")

    # Show sample data for area
    print("\n 📋 Sample traffic and weather data for this area:\n")
    display_columns = ['Date', 'Area Name', 'Traffic Volume', 'Environmental Impact', 'Public Transport Usage'] + weather_features
    print(area_df[display_columns].head(10).to_string(index=False))
```

Output

Chart for traffic volume

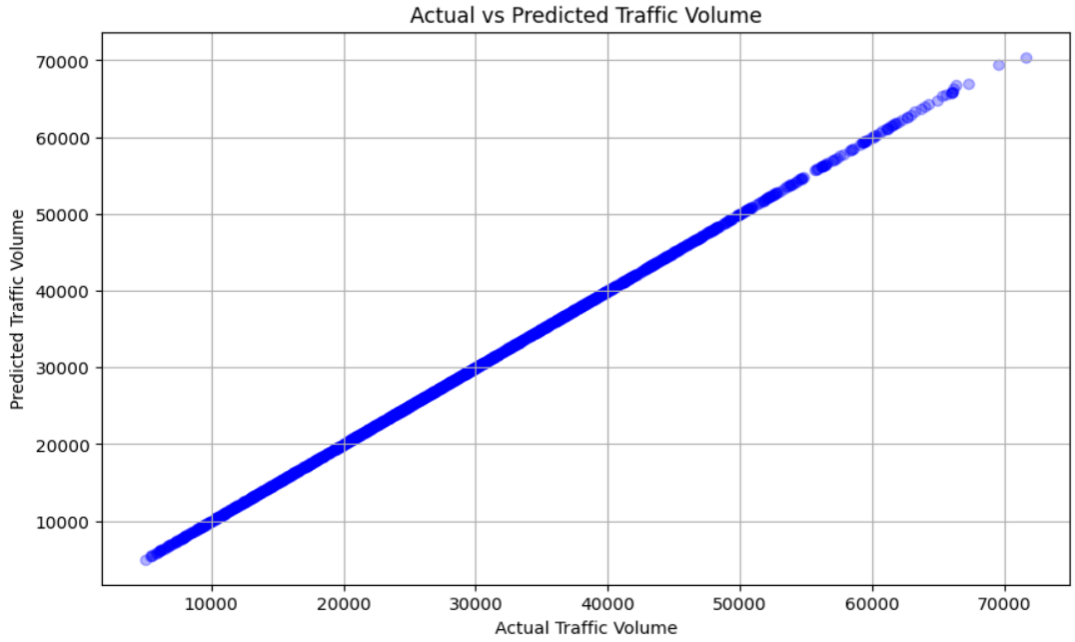
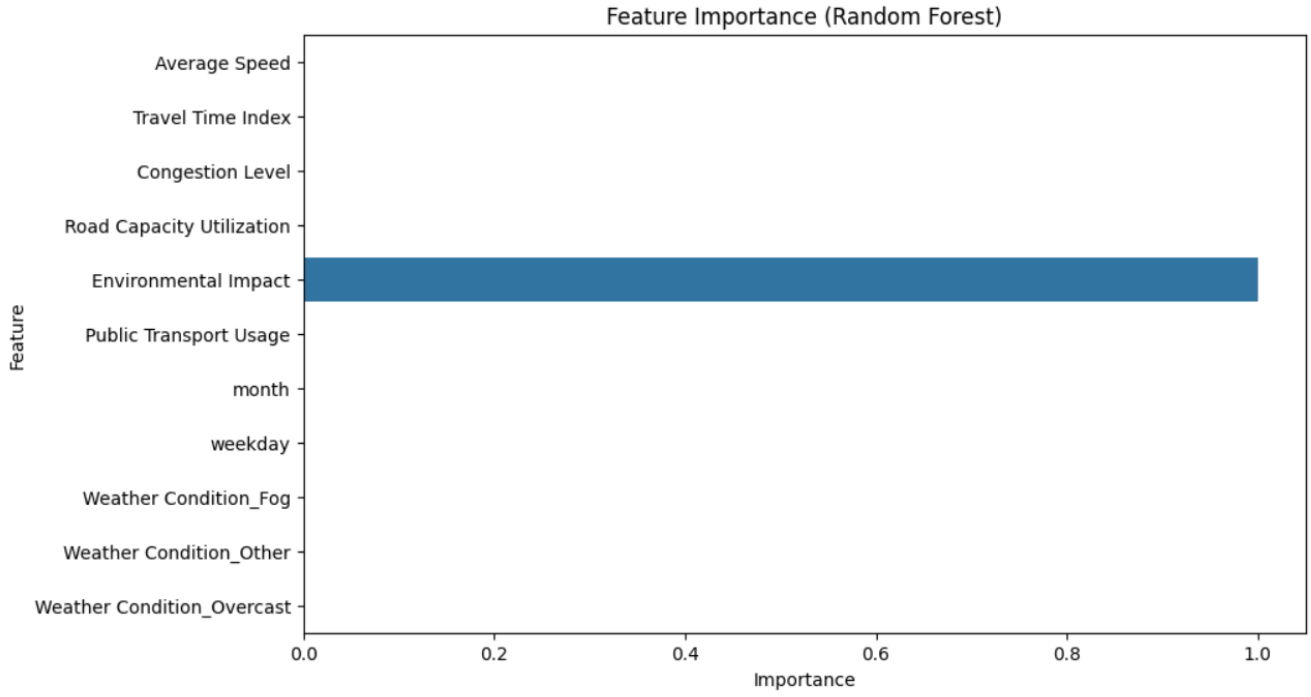


Chart for the environmental impact



Credit/References

- **Dataset:** Bangalore Traffic Volume Dataset – *(Source: Your Google Drive or dataset provider, if known. E.g., Kaggle or Open Government Data Platform)*
- **Libraries used:**
    - pandas, numpy – for data manipulation
    - matplotlib, seaborn – for visualizations
    - scikit-learn – for machine learning models and preprocessing
- **Images (if any):** Unsplash/Pixabay/Google Maps or the data visualization plots created using matplotlib.