**Traffic Volume Estimation - Detailed Documentation**

**Team ID : PNT2025TMID12884**

**Team Leader : Marisetti Ruchita**

**Team member : Pinisetti Devi Sri Satyasai**

**Team member : Naga Praneetha Bokka**

**Team member : Veera Venkata Durga Praveen Kumar Karnidi**

**Project Title : TrafficTelligence: Advanced Traffic Volume Estimation with Machine Learning**

**Introduction**

TrafficTelligence is an advanced system that uses machine learning algorithms to estimate and predict traffic volume with precision. By analyzing historical traffic data, weather patterns, events, and other relevant factors, TrafficTelligence provides accurate forecasts and insights to enhance traffic management, urban planning, and commuter experiences.

**Use Case Scenarios**

**1. Dynamic Traffic Management**

Provides real-time traffic volume estimations to adjust signal timings and optimize traffic flow.

**2. Urban Development Planning**

Assists city planners in designing road networks, public transit systems, and infrastructure based on future traffic projections.

**3. Commuter Guidance and Navigation**

Helps commuters and navigation apps avoid congested areas using predictive traffic analytics.

**Technical Architecture**

* **UI for Input**
* **Backend Model for Prediction**
* **Frontend Output Display**

**Project Flow**

1. **User Inputs Data** via UI
2. **Model Predicts Traffic Volume**
3. **Result Displayed on Screen**

**Step-by-Step Implementation**

**1. Data Collection**

* Collected from open sources like Kaggle, UCI, etc.
* Dataset used: weatherAus.csv
* Columns:
  + Holiday, Temp, Rain, Snow, Weather, Date, Time, Traffic Volume (Target)

**2. Data Pre-processing**

**a. Import Libraries**

* Numpy
* Pandas
* Matplotlib
* Seaborn
* Scikit-learn

**b. Handle Missing Values**

* Fill numeric columns using **mean**
* Fill categorical columns using **mode**

**c. Data Visualization**

* **Correlation Heatmap**
* **Pair Plot**
* **Box Plot**

**d. Feature Engineering**

* Convert Date and Time into separate features (Year, Month, Day, Hour, etc.)

**e. Splitting Variables**

* X: Independent variables
* y: Target variable (Traffic Volume)

**f. Feature Scaling**

* Normalize data using StandardScaler

**g. Train-Test Split**

* 80% training data
* 20% testing data

**3. Model Building**

**Models Used:**

* Linear Regression
* Decision Tree
* Random Forest (Best Performance)
* KNN
* SVM
* XGBoost

**Model Evaluation Metrics**

* **R2 Score**
* **RMSE (Root Mean Square Error)**

**Selected Model:**

**Random Forest Regressor**

* R2 Score ≈ 0.97
* Lowest RMSE

**4. Save the Model**

Used pickle to serialize the trained model:

python

import pickle

with open('model.pkl', 'wb') as f:

pickle.dump(model, f)

**5. Application Building (Web App)**

**Frontend (HTML)**

* index.html file with input form

**Backend (Flask - app.py)**

* Load model
* Take user input
* Feature scale the inputs
* Predict traffic volume
* Render prediction on web page

**Flask Code Highlights**

python

from flask import Flask, render\_template, request

import pickle

app = Flask(\_\_name\_\_)

model = pickle.load(open('model.pkl', 'rb'))

@app.route('/')

def home():

return render\_template('index.html')

@app.route('/predict', methods=['POST'])

def predict():

# Get input values, scale, predict

...

return render\_template('index.html', prediction\_text=f"Estimated Traffic Volume is {output}")

**6. Run the App**

**Steps:**

1. Open Anaconda Prompt
2. Navigate to app directory
3. Run: python app.py
4. Open: http://127.0.0.1:5000/ in browser

**7. Output**

* User sees a prediction:  
  **“Estimated Traffic Volume is XXXX units”**

**Screenshots of outputs:**



