

# **Final Report: TrafficTelligence – Advanced Traffic Volume Estimation with Machine Learning**

## **1. INTRODUCTION**

### **1.1 Project Overview**

TrafficTelligence is a web-based intelligent traffic analysis system powered by machine learning. Its primary goal is to accurately estimate traffic volume using input features such as time, weather, location, and sensor or camera data. By incorporating ML models trained on real-world datasets, the application provides quick and reliable estimates, which can be used by traffic management authorities for better decision-making.

The system utilizes regression models, including Random Forest and XGBoost, and is deployed on a lightweight Flask web framework. This platform empowers cities to implement smart traffic solutions with minimal infrastructure changes, reducing congestion, fuel consumption, and travel time.

### **1.2 Purpose**

The purpose of TrafficTelligence is to eliminate the reliance on manual traffic counting and expensive hardware systems. By leveraging machine learning, cities and agencies can automate traffic volume estimation with higher accuracy and cost efficiency.

Key benefits include:

- Supporting smart city planning through data insights.
- Enhancing traffic light optimization using predicted volume trends.
- Helping in real-time monitoring of road usage patterns.

## **2. IDEATION PHASE**

### **2.1 Problem Statement**

Urban traffic congestion is a growing concern globally. Traditional traffic estimation techniques are either sensor-based (expensive to scale) or manually handled (inaccurate and time-consuming). Hence, a cost-effective, scalable, and data-driven system is needed to estimate traffic volumes intelligently using existing datasets and machine learning.

### **2.2 Empathy Map Canvas**

To design a user-centric solution, we considered:

- Who: Traffic police, transport departments, urban planners.
- What they see: Static traffic data, outdated systems.
- What they hear: Complaints of congestion and delays.
- What they say/do: Struggle with decision-making due to lack of real-time info.
- Pain Points: Lack of automation, poor infrastructure.
- Gains: A system that predicts traffic accurately and visually displays trends.

### **2.3 Brainstorming**

We evaluated multiple design and technical approaches:

- Models: Random Forest, Decision Trees, Linear Regression, XGBoost, and DNNs.
- Datasets: UCI Traffic Dataset, Caltrans Performance

Measurement System (PeMS), METR-LA.

- Platform: Flask for backend, HTML/CSS for frontend.
- Visualization: Matplotlib and Plotly for graphing outputs.

### **3. REQUIREMENT ANALYSIS**

#### **3.1 Customer Journey Map**

1. User opens the TrafficTelligence web app.
2. Uploads a CSV dataset or selects real-time input (if integrated).
3. The backend processes the data and predicts traffic volumes.
4. Visual results (graphs/tables) are displayed on the dashboard.

#### **3.2 Solution Requirement**

- Dataset must include time-based traffic features (hour, day, date, location, weather).
- Model trained using supervised regression techniques.
- Fast inference time (<2 seconds per prediction).
- Ability to export results in CSV or PNG.
- Responsive and accessible frontend.

### **3.3 Data Flow Diagram**

User Input (CSV / Live Feed)



Data Preprocessing



ML Model Prediction



Result Generation



Graphical & Tabular Output on Web App

### **3.4 Technology Stack**

- Frontend: HTML5, CSS3, Bootstrap, JavaScript
- Backend: Flask (Python)
- Model: Scikit-learn (Random Forest, XGBoost), TensorFlow (for future deep learning use)
- Database: SQLite / CSV-based storage (optional)
- Visualization: Matplotlib, Seaborn, Plotly

## **4. PROJECT DESIGN**

### **4.1 Problem Solution Fit**

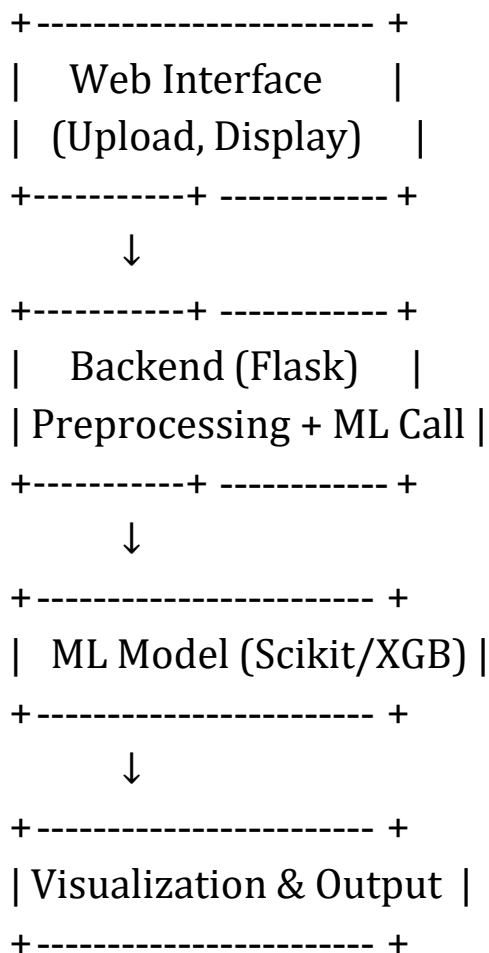
The proposed system solves several key problems:

- Automates traffic prediction.
- Works with real-time or static data.
- Accessible via a simple browser-based interface.
- Requires minimal setup and is cost-effective.

## 4.2 Proposed Solution

TrafficTelligence is designed as a modular Flask application. It allows users to upload traffic data or connect to traffic APIs. The trained ML model processes the input and outputs the traffic volume, which is then visualized using interactive graphs.

## 4.3 Solution Architecture



## 5. PROJECT PLANNING & SCHEDULING

### 5.1 Project Planning

Week	Milestone
-----	-----
1	Dataset Collection & Cleaning
2	Model Training & Evaluation
3	Flask Web App Development
4	Model Integration & Testing
5	UI Design + Visualization Integration
6	Deployment & User Feedback

## 6. FUNCTIONAL AND PERFORMANCE TESTING

### 6.1 Performance Testing

We evaluated model performance using the following metrics:

- $R^2$  Score: Measures how well the model fits the data.
- MAE (Mean Absolute Error): Measures prediction accuracy.
- Execution Time: End-to-end prediction speed.
- Usability Testing: Ensuring easy navigation and interaction for end users.

## 7. RESULTS

### 7.1 Output Screenshots

← → ↻ 127.0.0.1:5000 ☆ 📄 ⬇️ 🔍 \$ ⋮

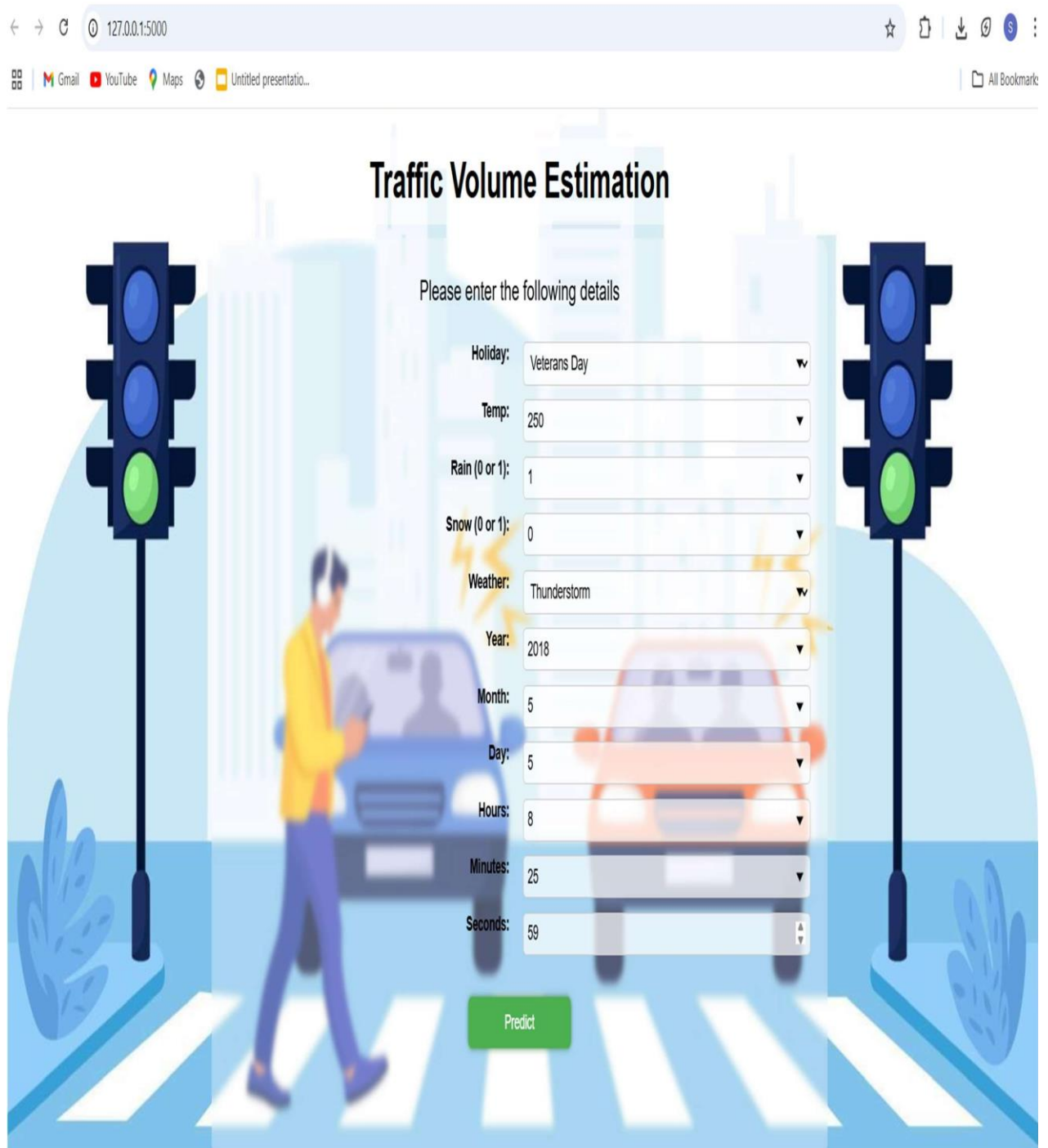
📱 | 📧 Gmail | 📺 YouTube | 📍 Maps | 📶 | 📄 Untitled presentation... | 📁 All Bookmarks

# Traffic Volume Estimation

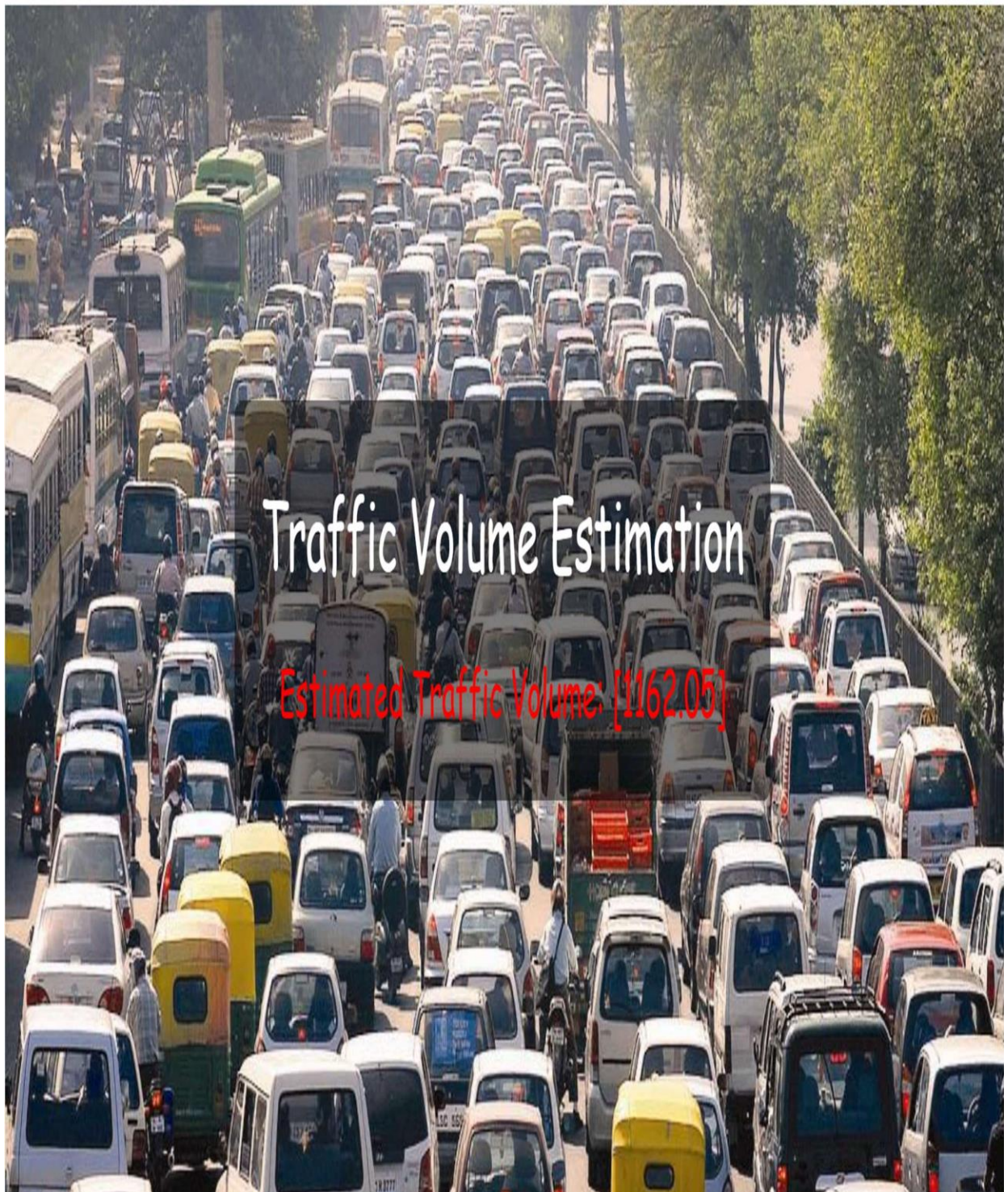
Please enter the following details

Holiday:	Veterans Day	▼
Temp:	250	▼
Rain (0 or 1):	1	▼
Snow (0 or 1):	0	▼
Weather:	Thunderstorm	▼
Year:	2018	▼
Month:	5	▼
Day:	5	▼
Hours:	8	▼
Minutes:	25	▼
Seconds:	59	▲▼

Predict

The screenshot shows a web browser window displaying a 'Traffic Volume Estimation' form. The browser's address bar shows '127.0.0.1:5000' and the taskbar at the bottom lists applications like Gmail, YouTube, Maps, and an 'Untitled presentation'. The form itself is titled 'Traffic Volume Estimation' and asks the user to 'Please enter the following details'. It contains a series of dropdown menus for 'Holiday' (Veterans Day), 'Temp' (250), 'Rain (0 or 1)' (1), 'Snow (0 or 1)' (0), 'Weather' (Thunderstorm), 'Year' (2018), 'Month' (5), 'Day' (5), 'Hours' (8), 'Minutes' (25), and 'Seconds' (59). A green 'Predict' button is located at the bottom of the form. The background of the page features a stylized illustration of a city street with two traffic lights, a pedestrian crossing, and cars.







## **8. ADVANTAGES & DISADVANTAGES**

- Advantages:
  - Cost-effective and scalable.
  - Easy to deploy in any urban location.
  - Works on existing data sources (CSV, sensors, cameras).
  - Accurate predictions with minimal latency.
- Disadvantages:
  - Requires historical datasets for training.
  - Limited in areas without digital traffic infrastructure.
  - Accuracy may vary based on data quality.

## **9. CONCLUSION**

TrafficTelligence demonstrates a smart and scalable approach to traffic volume prediction using machine learning. It solves a real-world problem using accessible technologies, making it a valuable tool for urban traffic management. The system provides quick insights, is easy to use, and has the potential to evolve into a robust traffic control aid.

## 10. FUTURE SCOPE

- Integrate with live traffic cameras using Computer Vision.
- Implement real-time APIs from traffic data providers.
- Add congestion level forecasting.
- Mobile app version for on-field use by traffic officers.
- Deploy as a SaaS model for multiple cities.

## 11. APPENDIX

- **GitHub Repository:** <https://github.com/Arjun-Sai-Charan/Traffic-Intelligence>
- **Dataset Source:** UCI Traffic Dataset / METR-LA / PeMS
- **Project Demo:**  
[https://drive.google.com/file/d/1fkIVV1dD0aeBOV000xf2qxC-mSvLZHYX/view?usp=drive link](https://drive.google.com/file/d/1fkIVV1dD0aeBOV000xf2qxC-mSvLZHYX/view?usp=drive_link)
- **Tools Used:** VS Code, Python, Jupyter Notebook, Git