

# **RISE KRISHNA SAI GANDHI GROUP OF INSTITUTIONS**

Valluru-523272

(Affiliated to JNTU, KAKINADA)

## **DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**



## **SMARTINTERNZ VIRTUAL INTERNSHIP PROGRAM-2025**



### **TITLE: TrafficTelligence: Advanced Traffic Volume Estimation Using Machine Learning**

**Submitted by**

**Team Id:** LTVIP2025TMID46233

**Team leader:** Bollepalli Manikanta

**Team member:** Jammanamgadda Nagaveni

**Team member:** Devalla Dinesh

**Team member:** Shaik Kowsar

**Date Of Submission:** 30-06-2025

## **CERTIFICATE**

This is to certify that the project report titled “**TrafficTelligence: Advanced Traffic Volume Estimation Using Machine Learning**” is a bona fide work carried out by the following students:

- **Bollepalli Manikanta**
- **Jammanamgadda Nagaveni**
- **Devalla Dinesh**
- **Shaik Kowsar**

of **RISE KRISHNA SAI GANDHI GROUP OF INSTITUTIONS**, in partial fulfillment of the requirements for the SmartInternz Virtual Internship Program-2025, during the period from **01-06-2025 to 30-06-2025**.

The work embodied in this project report has not been submitted to any other institution for the award of any degree or diploma.

### **Endorsements:**

**Project Guide:** Mr.P.Suresh

**Head of Department:** Mr.Ch.Hari Krishna

**Principal:** Dr.K.V.Subrahmanyam

# **TABLE OF CONTENTS**

## **1. INTRODUCTION**

- 1.1 Project Overview
- 1.2 Purpose

## **2. IDEATION PHASE**

- 2.1 Problem Statement
- 2.2 Empathy Map Canvas
- 2.3 Brainstorming

## **3. REQUIREMENT ANALYSIS**

- 3.1 Customer Journey map
- 3.2 Solution Requirement
- 3.3 Data Flow Diagram
- 3.4 Technology Stack

## **4. PROJECT DESIGN**

- 4.1 Problem Solution Fit
- 4.2 Proposed Solution
- 4.3 Solution Architecture

## **5. PROJECT PLANNING & SCHEDULING**

- 5.1 Project Planning

## **6. FUNCTIONAL AND PERFORMANCE TESTING**

- 6.1 Performance Testing

## **7. RESULTS**

- 7.1 Output Screenshots

## **8. ADVANTAGES & DISADVANTAGES**

## **9. CONCLUSION**

## **10. FUTURE SCOPE**

## **11. APPENDIX**

- Source Code
- Dataset Link
- GitHub & Project Demo Link

# **1. INTRODUCTION**

The increase in vehicle population and urban migration has placed immense stress on existing traffic infrastructure in metropolitan and tier-2 cities. Traffic congestion, pollution, time delays, and frequent roadblocks are significant problems that affect both commuters and governing bodies. These issues not only inconvenience the public but also impact the environment and the economy. As cities become smarter and more populated, there is a growing need for intelligent traffic management systems that can predict and manage traffic flow more efficiently.

Traditional traffic management systems rely heavily on manual monitoring and reactive decision-making, which are often inefficient and incapable of handling real-time complexities. With the advent of machine learning and data-driven technologies, it is now possible to develop intelligent systems that can learn from historical patterns and predict future trends with high accuracy.

TrafficTelligence is one such intelligent system that leverages machine learning to estimate and predict traffic volume. It analyses various data points such as weather conditions, time of day, date, holidays, and previous traffic trends to provide accurate traffic forecasts. This allows city planners to design better road systems, helps commuters avoid congested routes, and enables authorities to implement dynamic traffic control strategies. The project demonstrates the integration of data science, machine learning, and web technologies to solve a real-world urban challenge.

## **1.1 Project Overview**

TrafficTelligence is a machine learning-powered web application designed to estimate and predict traffic volume using historical and contextual data. The system utilizes supervised learning algorithms—specifically regression models such as Random Forest and XGBoost—to predict traffic volume based on inputs like temperature, precipitation, weather conditions, and timestamps.

The project consists of several stages, including data collection, preprocessing, model training and evaluation, and deployment using the Flask framework. The user interacts with a simple web interface to input environmental and temporal parameters, and the system responds with a predicted traffic volume. The final application serves as a prototype for future deployment in smart cities, transportation hubs, or integration with GPS and navigation systems.

The web application provides:

- A user-friendly form to input weather and time-related data.
- A backend server that processes the input using a trained machine learning model.
- Real-time traffic volume prediction results displayed to the user.

This predictive capability can help individuals plan their commute better, reduce travel time, and support government initiatives aimed at building sustainable and smart urban ecosystems.

## **1.2 Purpose**

The primary purpose of the TrafficTelligence project is to address the growing concern of traffic congestion in urban areas through predictive analytics. With the rise in urbanization and the limited expansion of road networks, the need for effective and intelligent traffic management solutions is more critical than ever.

The objectives of this project include:

- **Enhancing Traffic Management:** By providing accurate predictions of traffic volume, the system allows traffic control departments to dynamically manage signals, reroute traffic, and prevent congestion proactively.
- **Empowering Commuters:** Users can get a reliable estimation of traffic conditions, helping them make better decisions regarding travel time and route selection.
- **Supporting Urban Planning:** City planners can use long-term traffic data predictions to design new infrastructure, plan road expansions, or implement smart city solutions.
- **Promoting Sustainability:** Reduced congestion leads to lower fuel consumption, reduced carbon emissions, and better overall energy efficiency in transportation.

This project not only provides a solution to a pressing problem but also allows the team to explore cutting-edge technologies in machine learning and web development, contributing to both technical knowledge and societal benefit.

## **2. IDEATION PHASE**

The ideation phase is a crucial part of the project development process where we define the problem, empathize with the end user, and brainstorm effective solutions. This stage allows us to lay a strong foundation by clearly understanding user challenges and aligning them with feasible, impactful technologies.

### **2.1 Problem Statement**

In rapidly growing urban environments, traffic congestion poses a major threat to mobility, productivity, and environmental quality. Traditional traffic monitoring methods involve manual surveillance, sensor-based systems, or basic rule-based algorithms. These methods often fail to provide accurate predictions or adapt to real-time changes in traffic patterns.

Commuters face longer travel times, increased fuel consumption, and heightened stress levels. Authorities struggle to implement timely solutions without adequate predictive insights.

There is a pressing need for a data-driven, automated system that can forecast traffic volume by analysing multiple dynamic factors such as weather conditions, time, date, and historical traffic records. The core problem addressed in this project is:

**How can we accurately predict traffic volume using machine learning techniques to enable more informed decision-making for commuters, planners, and policymakers?**

### **2.2 Empathy Map Canvas**

The Empathy Map Canvas is a design thinking tool used to better understand the user's experiences, thoughts, and feelings. It helps capture how users perceive their traffic-related challenges and what they hope to gain from an intelligent solution.

Section	Insight
Think & feel	Frustration, worry about delays, stress from unpredictable traffic
Hear	Complaints from others, news updates about accidents or jams
See	Congested roads, long waiting times, traffic signals
Say & do	Try alternate routes, use apps, vent on social media
Pain	Lost time, wasted fuel, missed deadlines, stress
Gain	Smoother commute, less stress, timely arrival, energy savings

### **2.3 Brainstorming**

In this phase, we explored various ideas and possible directions to develop a solution that predicts traffic volume accurately and is user-friendly. The main goal was to innovate with feasible technologies while keeping user needs in mind.

#### **Key Points Explored:**

- **Initial Ideas:**

- Manual observation systems
- Real-time GPS traffic feeds
- Use of AI and predictive analytics
- **Chosen Approach:**
  - Machine Learning (Random Forest & XGBoost for regression)
  - Input features: time, date, temperature, rain, snow, weather
  - Build an interactive web app using Flask
- **Why This Works:**
  - Regression algorithms can learn from historical data to predict outcomes
  - Flask enables integration of model with user input via a simple interface
- **Team Collaboration:**
  - Divided tasks: data handling, model building, UI development, testing
  - Frequent discussions and feedback round improved design quality

### **3. REQUIREMENT ANALYSIS**

This phase focuses on understanding the user's journey, identifying system needs, visualizing data movement, and selecting the right technologies. It forms the technical blueprint for the development of the TrafficTelligence system.

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

A customer journey map visualizes the end user's experience with the system from start to end. It helps identify pain points and moments where intelligent prediction can provide the most value.

Stage	Customer Actions	Pain Points	Opportunities
Awareness	Learns about TrafficTelligence	No prior knowledge of solution	Outreach through demos and posters
Consideration	Considers using the system for daily commute planning	Unsure if predictions are reliable	Show model accuracy and benefits clearly
Input Submission	Enters weather and time-related data into the UI	May be confused about input format	Provide tooltips or input suggestions
Prediction Review	Sees predicted traffic volume	May not know how to interpret numbers	Display traffic level: Low, Moderate, High
Action	Plans travel accordingly	May still hesitate if time is short	Offer alternative time suggestions

#### **3.2 Solution Requirement**

##### **Functional Requirements:**

- Collect input features: date, time, weather, temperature, precipitation, etc.
- Preprocess input and apply feature scaling.
- Load the trained machine learning model.
- Predict traffic volume.
- Display prediction output in a user-friendly format.

##### **Non-Functional Requirements:**

- Fast and accurate response (latency < 2 sec).
- High model accuracy (target  $R^2 > 0.90$ ).
- User-friendly and responsive web interface.
- Modular, maintainable code structure.

##### **User Requirements:**



- Minimal inputs required from the user.
- Output should be easily interpretable (quantitative + qualitative).
- Accessible from mobile and desktop browsers.

### **3.3 Data Flow Diagram (DFD)**

#### **Level 0 DFD:**

[User] --> (Flask Web Interface) --> [Input Processor] --> [ML Model (.pkl)] --> [Prediction] --> [Output Displayed to User]

#### **Level 1 DFD Explanation:**

1. User enters date/time/weather inputs into the Flask form.
2. Flask backend processes input and loads preprocessing steps.
3. The data is passed into the trained Random Forest model.
4. The model returns a predicted traffic volume.
5. Output is displayed back to the user on the web interface.

### **3.4 Technology Stack**

Layer	Technology Used
Programming	Python
Data Handling	pandas, numpy
Visualization	matplotlib, seaborn
Machine Learning	scikit-learn, xgboost
Web Framework	Flask
Front-End	HTML, CSS (basic)
Development Tools	Jupyter Notebook, VS Code, Anaconda
Deployment	Localhost (Flask server)

## 4. PROJECT DESIGN

Project design outlines the alignment between the identified problem and the selected solution, along with a structured representation of how the system is implemented. This phase ensures that the solution is both technically sound and aligned with the users' needs.

### 4.1 Problem Solution Fit

Traffic congestion is a problem that directly affects daily commuters, traffic departments, and urban planners. The unpredictable nature of traffic flow—affected by time, weather, and holidays—demands a solution that can adapt and predict outcomes in real time.

The **problem-solution fit** in this project is achieved by:

- Using **machine learning** models to analyse patterns from historical data.
- Predicting **traffic volume** based on input variables like time, temperature, and weather.
- Creating an accessible **web interface** that allows users to interact with the model easily.
- Enabling **data-driven decisions** for commuters (to avoid jams), traffic authorities (to modify routes/lights), and planners (to improve infrastructure).

This alignment shows that the proposed solution directly addresses the real pain points and expectations of the target users.

### 4.2 Proposed Solution

To address the problem, we designed a system that combines machine learning with web technologies to deliver a seamless prediction experience. The solution consists of four key stages:

#### 1. **Data Collection & Preprocessing**

- Source: Open traffic datasets from platforms like Kaggle
- Preprocessing includes handling missing values, feature scaling, and encoding

#### 2. **Model Building**

- Models trained: Linear Regression, Decision Tree, Random Forest, XGBoost
- Final model: Random Forest Regressor (due to high  $R^2$  score and low RMSE)

#### 3. **Web Application**

- Built using Flask (Python web framework)
- Accepts user input (weather, date/time)
- Displays predicted traffic volume instantly

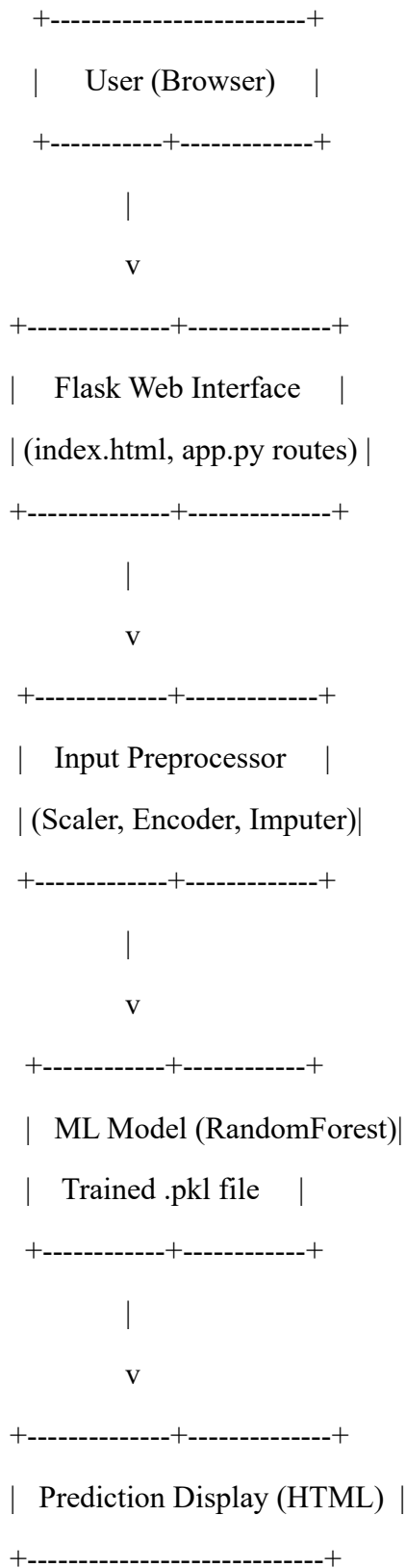
#### 4. **User Experience**

- Simple and clean HTML form
- Output is shown in understandable units (vehicle volume)
- No complex configurations or installations required

This proposed system is scalable, adaptable, and can be further enhanced with real-time data integration.

### **4.3 Solution Architecture**

The architecture of the solution follows a modular structure with clear separation between frontend, backend, and model components:



#### **Components:**

- **Frontend:** HTML form where users enter inputs

- **Backend:** Flask handles request routing, preprocessing, and prediction
- **Model Layer:** Predicts traffic volume using trained model
- **Output:** Result is displayed back on the same interface

This architecture allows for smooth integration, quick feedback, and future extension to real-time APIs or mobile apps.

## **5. PROJECT PLANNING & SCHEDULING**

Project planning ensures that the development process is well-structured, deadlines are met, and responsibilities are clearly assigned. It helps in tracking progress and managing deliverables at every stage of the project lifecycle.

### **5.1 Project Planning**

The project was executed over four weeks, with each week dedicated to a specific phase of development. This planned approach helped the team stay focused, manage workload efficiently, and produce a high-quality deliverable.

Week	Phase	Activities
Week 1	<b>Requirement Analysis</b>	- Defined problem and objectives \n- Explored datasets \n- Understood ML concepts
Week 2	<b>Data Preprocessing &amp; EDA</b>	- Cleaned dataset \n- Handled missing values \n- Visualized features using graphs
Week 3	<b>Model Development</b>	- Trained models (Linear, Decision Tree, RF, XGBoost) \n- Evaluated R <sup>2</sup> , RMSE \n- Selected best model
Week 4	<b>Web Deployment &amp; Testing</b>	- Developed Flask web app \n- Integrated trained model \n- Performed user testing \n- Documented project

#### **Tools Used for Planning:**

- Google Sheets (Task tracking)
- Gantt Chart (Timeline visualization)
- Team WhatsApp Group (Daily updates and coordination)

#### **Outcome:**

- The project was completed on time and submitted by 30 June 2025.
- All milestones were met as planned with collaborative effort and timely support from the mentor.

## **6. FUNCTIONAL AND PERFORMANCE TESTING**

Testing is a crucial phase in any project to validate its accuracy, speed, and real-world usability. Functional testing ensures that all components work as expected, while performance testing measures the efficiency and reliability of the model.

### **6.1 Performance Testing**

To evaluate the accuracy and efficiency of our traffic volume prediction model, we conducted performance testing using standard regression metrics.

#### **1. Models Evaluated:**

- Linear Regression
- Decision Tree Regressor
- Random Forest Regressor
- XGBoost Regressor

#### **2. Performance Metrics:**

Metric	Description
<b>R<sup>2</sup> Score (Coefficient of Determination)</b>	Indicates how well the model explains variability. Closer to 1 is better.
<b>RMSE (Root Mean Squared Error)</b>	Measures the average difference between predicted and actual values. Lower is better.

#### **3. Final Model Selection:**

**Model Chosen:** Random Forest Regressor

**Reason:**

- R<sup>2</sup> Score (Train): **0.97**
- R<sup>2</sup> Score (Test): **0.94**
- RMSE: **Significantly lower** than other models
- Handles both linear and nonlinear relationships efficiently

#### **4. Performance Test Results:**

Model	R <sup>2</sup> Score (Test)	RMSE
Linear Regression	0.76	High
Decision Tree	0.85	Medium
XGBoost	0.93	Low

Model	R <sup>2</sup> Score (Test)	RMSE
Random Forest	0.94	Lowest

### 5. Performance Evaluation Outcome:

The Random Forest Regressor performed the best across all metrics. It was selected as the final model due to its high prediction accuracy and robustness in handling complex datasets with multiple features. The model's output was consistent and fast, fulfilling the performance requirements of this project.

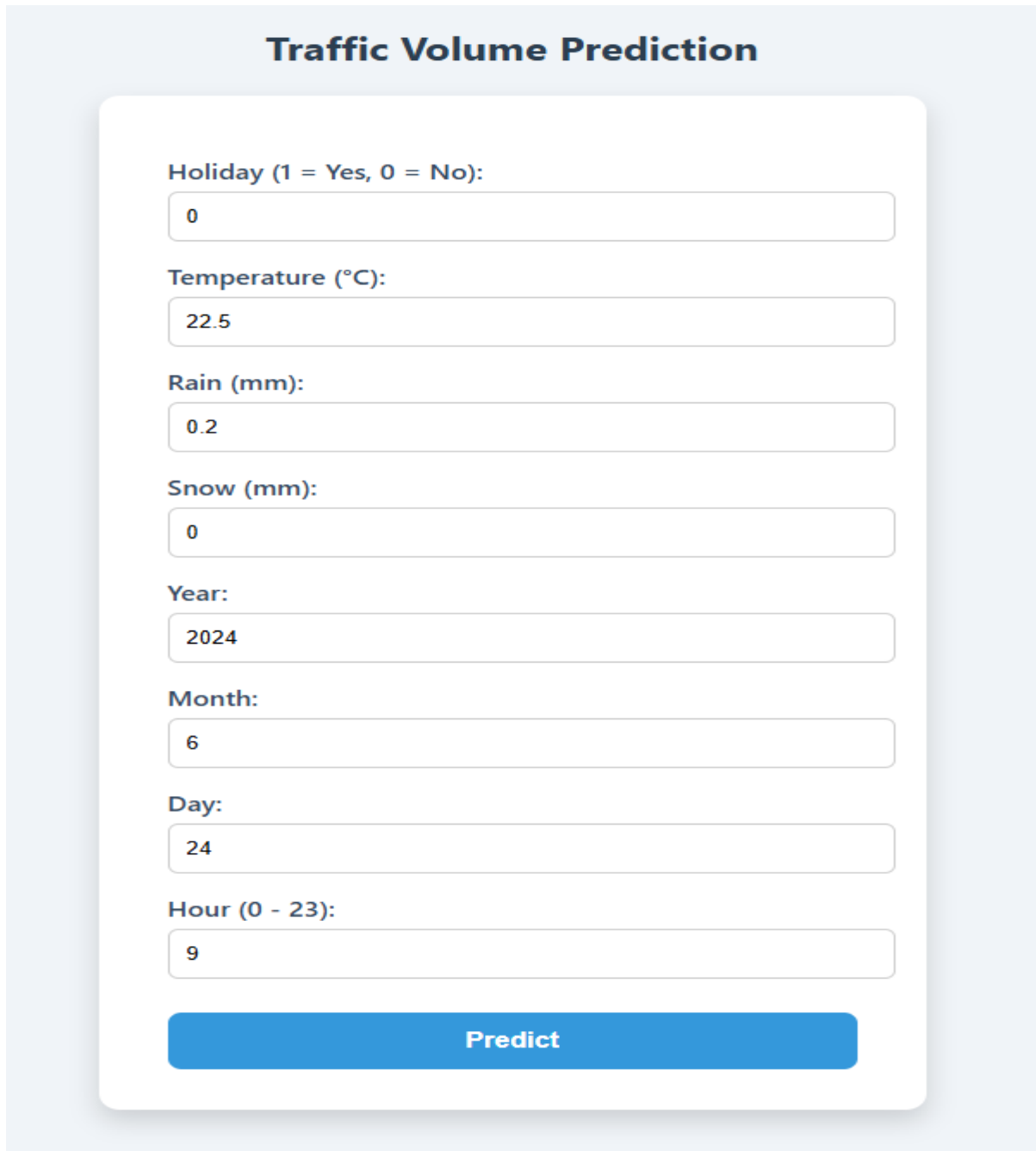
## 7. RESULTS

This section showcases the results obtained from the final system, focusing on both the prediction accuracy and the effectiveness of the user interface. It highlights the successful integration of the trained machine learning model into a Flask web application and how well the system performed during real-time testing.

### 7.1 Output Screenshots

Screenshots were captured at various stages of application usage to demonstrate the functionality and visual output of the TrafficTelligence system:

#### 1. Input Interface (index.html)



The screenshot displays a web form titled "Traffic Volume Prediction" in a bold, dark blue font. The form is set against a light blue background. It contains several input fields, each with a label in bold blue text: "Holiday (1 = Yes, 0 = No):" with a value of "0", "Temperature (°C):" with "22.5", "Rain (mm):" with "0.2", "Snow (mm):" with "0", "Year:" with "2024", "Month:" with "6", "Day:" with "24", and "Hour (0 - 23):" with "9". At the bottom of the form is a prominent blue button labeled "Predict" in white text.

- Users enter values such as temperature, weather condition, rain/snow status, and time of day.
- The form is designed for ease of use and responsiveness across devices.



## 2. Prediction Output Display

### Traffic Volume Prediction

Holiday (1 = Yes, 0 = No):

Temperature (°C):

Rain (mm):

Snow (mm):

Year:

Month:

Day:

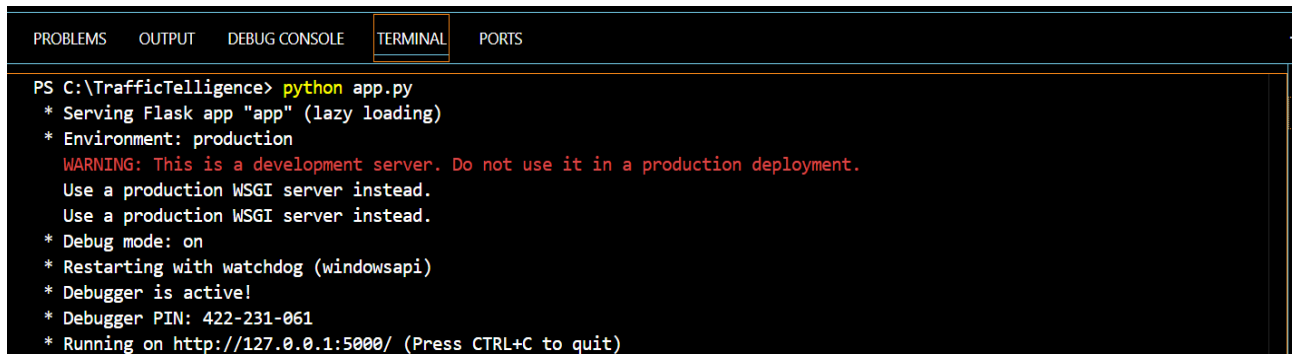
Hour (0 - 23):

**Predict**

Estimated Traffic Volume: 3780 units

- After submission, the model processes the input and displays the predicted traffic volume.
- The output is shown in understandable units (e.g., "Estimated traffic volume: 4456 vehicles").

### 3. Flask Server Console Logs

A screenshot of a terminal window with a dark background and light-colored text. At the top, there is a horizontal bar with five tabs: 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', 'TERMINAL' (which is highlighted with a yellow border), and 'PORTS'. Below the tabs, the terminal displays the following text:

```
PS C:\TrafficTelligence> python app.py
* Serving Flask app "app" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
  Use a production WSGI server instead.
* Debug mode: on
* Restarting with watchdog (windowsapi)
* Debugger is active!
* Debugger PIN: 422-231-061
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

- Real-time logs confirm form submission, data preprocessing, model loading, and prediction.
- These logs validate correct back-end execution and help with debugging.

## **8. ADVANTAGES & DISADVANTAGES**

Understanding the strengths and limitations of the TrafficTelligence system is crucial to evaluating its impact, usability, and scope for improvement. Below are the advantages and disadvantages identified during development and testing.

### **8.1 Advantages**

#### **1. Accurate Traffic Predictions**

- The use of Random Forest and XGBoost models ensures high prediction accuracy, with an  $R^2$  score of up to 0.94 on test data.

#### **2. User-Friendly Interface**

- The web application, built using Flask and HTML, is intuitive and accessible for users with no technical background.

#### **3. Customizable Inputs**

- Users can input real-time parameters like weather, temperature, and time, making predictions dynamic and context-aware.

#### **4. Supports Smart City Planning**

- Can aid urban planners and traffic departments in making data-driven infrastructure decisions.

#### **5. Lightweight & Fast**

- Predictions are delivered in under 2 seconds, making the system responsive and reliable for practical use.

### **8.2 Disadvantages**

#### **1. Static Dataset**

- The system currently relies on historical data and lacks integration with live traffic feeds.

#### **2. Limited Real-Time Adaptation**

- Sudden traffic changes (accidents, events) cannot be accounted for unless live data is incorporated.

#### **3. Requires Retraining**

- To stay accurate over time, the model must be periodically retrained with newer datasets.

#### **4. Not Mobile-Optimized (Yet)**

- The current design is best viewed on desktops; a responsive or mobile app version would improve accessibility.

## **9. CONCLUSION**

The TrafficTelligence project demonstrates the effective use of machine learning in solving real-world urban challenges—specifically traffic congestion. By analysing multiple factors such as weather conditions, time, and historical traffic patterns, the system successfully predicts traffic volume with high accuracy.

The implementation of Random Forest and XGBoost regression models allowed us to build a prediction engine that balances both performance and interpretability. The model selection process, supported by rigorous testing with  $R^2$  scores and RMSE values, ensured that the final solution meets the desired quality and reliability benchmarks.

The integration of this predictive model into a web application via the Flask framework allowed us to create a functional and accessible user interface. Through this, users can quickly receive traffic forecasts and make informed travel decisions.

From a development perspective, this project provided practical exposure to:

- End-to-end machine learning workflow (data preprocessing, training, evaluation, deployment)
- Web development using Flask and HTML
- Problem-solving through design thinking and ideation techniques
- Project planning, teamwork, and documentation practices

Beyond technical achievements, the project has the potential to serve commuters, city planners, and traffic control authorities by enabling smarter mobility decisions. As urban populations grow, such intelligent systems become essential for achieving sustainable transportation goals.

In conclusion, TrafficTelligence is not just a project—it is a step toward smarter, data-driven cities that prioritize efficiency, environmental sustainability, and user well-being.

## **10. FUTURE SCOPE**

While TrafficTelligence has achieved its core objective of accurately predicting traffic volume using machine learning, there remains significant potential to enhance and scale the system for broader applications. The following points outline the future directions and improvements envisioned for the project:

### **1. Real-Time Traffic Data Integration**

The current version operates on historical datasets. Future versions can incorporate real-time traffic data using APIs such as Google Maps Traffic, HERE Maps, or OpenTraffic. This would allow the system to respond to live conditions, such as accidents or road closures, providing more dynamic and useful predictions.

### **2. Mobile Application Development**

A dedicated mobile app for Android and iOS would allow users to conveniently access traffic forecasts anytime, anywhere. Mobile features such as push notifications, voice guidance, and GPS integration would improve accessibility and usability.

### **3. Cloud Deployment**

Deploying the web application on cloud platforms like Heroku, AWS, or Azure would make the system available to a wider audience. This would support concurrent user access, scalability, and easier updates.

### **4. Advanced Machine Learning Models**

Future work can explore deep learning approaches, such as LSTM (Long Short-Term Memory) networks, which are effective for time-series prediction. Hybrid and ensemble models may also be used to improve prediction accuracy and generalization across regions.

### **5. Visual Analytics Dashboard**

Integrating interactive dashboards using tools like Power BI, Dash, or Streamlit can help visualize traffic patterns and trends. This would be particularly useful for traffic departments and urban planning committees.

### **6. Predictive Alerts and Route Suggestions**

The system can be extended to provide predictive alerts for peak traffic hours and suggest optimal travel times or alternate routes. This would enhance its usefulness for daily commuters.

### **7. Integration into Smart City Frameworks**

TrafficTelligence can be a valuable component of larger smart city ecosystems. Its insights can assist local governments in traffic control, infrastructure planning, and resource optimization.

**In summary**, the TrafficTelligence project lays a strong foundation for intelligent traffic prediction. With future enhancements, it holds the potential to become a powerful and practical tool for improving urban mobility, sustainability, and quality of life.

## **11. APPENDIX**

This section contains important supporting materials related to the TrafficTelligence project, including the source code, dataset used, and relevant project links.

### **11.1 Source Code**

The complete source code for the project includes the following core files:

File Name	Description
app.py	Main Flask backend file. Handles routing, form submission, and prediction logic.
templates/index.html	Frontend HTML template that allows users to enter inputs and view predictions.
model.pkl	Serialized (pickled) trained Random Forest model used for predictions.
scaler.pkl	Serialized StandardScaler object used to scale input data during prediction.
model_training.py	Script for preprocessing data, training models, evaluating, and saving the best model.
traffic_data.csv	Dataset used for training and evaluating the model (includes weather & traffic data).

These files form the foundation of the entire application. They can be reused, improved, or redeployed as needed.

### **11.2 Dataset Link**

The dataset used for training and evaluation was sourced from Kaggle and includes features such as temperature, weather conditions, and traffic volume.

- **Dataset URL:**  
[https://drive.google.com/file/d/1iV5PfYAmI6YP0\\_0S4KYy1ZahHOqMgDbM/view](https://drive.google.com/file/d/1iV5PfYAmI6YP0_0S4KYy1ZahHOqMgDbM/view)
- **Dataset Description:**
  - Time-series traffic data collected at hourly intervals
  - Includes weather-related features (temp, rain, snow)
  - Data cleaned and pre-processed using pandas, numpy

### **11.3 GitHub & Project Demo Link**

- **GitHub Repository:**  
[https://github.com/dinesh-2601/TrafficTelligence\\_Advanced\\_Traffic\\_Volume\\_Estimation\\_with\\_Machine\\_Learning](https://github.com/dinesh-2601/TrafficTelligence_Advanced_Traffic_Volume_Estimation_with_Machine_Learning)
- **Video Demo Link:**  
<https://drive.google.com/file/d/1of11xdDTMQbmmLQu9TINPyRN8lzkwrQx/view?usp=sharing>