TrafficTelligence: Advanced Traffic Volume Estimation with Machine Learning

Team ID:

LTVIP2025TMID44576

Team Members:

1. Anchi Jagan Yadav
2. B Lalitha
3. Lingala Anusha
4. T M Kiran Kumar

Phase 1: Advanced Traffic Volume Estimation with Machine Learning

Problem statement: 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.

Proposed Solution: To address the challenges of increasing traffic congestion, inefficient traffic control, and lack of accurate real-time insights, Traffic Telligence proposes a machine learning-driven, data-centric system capable of accurately estimating and predicting traffic volumes using a combination of historical data, weather conditions, and event-related context.

Target Users: Traffic Telligence is designed to serve a diverse range of users involved in traffic control, urban planning, commuting, and intelligent infrastructure

Expected Outcome: Traffic Telligence will deliver accurate traffic volume predictions, enabling real-time congestion management and dynamic traffic control. It will support smarter urban planning through data-driven insights. Commuters will benefit from optimized routes, reduced travel time, and improved navigation experiences.

Phase 2: Requirements Analysis

Technical Requirements:

- Python
- Xgboost, Rondam forest
- HTML
- Jupyter Note Book

Functional Requirements

- Holiday, temp, rain, snow
- Display result

Constraints & Challenges:

- Model accuracy depends on data set quality

- Ensuring low-latency processing for real-time traffic predictions is technically demanding.
- Hardware Limitations
- Real-Time Processing

Phase 3: Project Design

- **System Architecture Diagram:**

User
⇩
Enter details
⇩
Flask Backend
⇩
Transfer Learning Model
(e.g., Random Forest)
⇩
Prediction
⇩
Result Display

User Flow:

User opens the web application
⇩
Click on "Get Started" button
⇩
Enter the details
⇩
Click on "Predict"
⇩
system displays traffic volume

UI/UX Considerations:

- Simple enter details
- Predict Button
- Result section.

Phase 4: Project Planning (Agile Methodologies)

- **Sprint Planning**

  - ✓ Week 1: Dataset collection & preprocessing
  - ✓ Week 2: Model training and tuning
  - ✓ Week 3: Flask integration
  - ✓ Week 4: Frontend + Testing + deployment

- **Task Allocation:**

  - ✓ Member A: Model training
  - ✓ Member B: Flask backend
  - ✓ Member C: Frontend UI
  - ✓ Member D: Documentation & testing

- **Timeline & Milestones:**

  - ✓ Milestone 1: Dataset ready (week 1)
  - ✓ Milestone 2: Model trained (week 2)
  - ✓ Milestone 3: Web integration (week 3)

- ✓ Milestone 4: Testing + Report (week 4)

Phase 5: Project Development

- **Technology stack used:**

  Python Flask, train_test_split ,Random forest, HTML

- **Development Process:**
  - ❖ Trained Random forest on rotten dataset
  - ❖ Created app.py with prediction route
  - ❖ HTML templates for UI
  - ❖ Uploaded image saved and pre-processes
  - ❖ Model predicts and result shown on predict.html

- **Challenges & Fixes:**
  - ❖ File not saving correctly – fixed with os. path. join ()
  - ❖ Styling issues fixed via HTML templates updates

Phase 6: Functional & Performance testing

- **Test cases Executed:**
  - ❖ Uploaded valid / invalid image formats
  - ❖ Checked correct predictions for known test images
  - ❖ UI responsiveness and error handling

- **BUG Fixes & improvements:**
  - ❖ Fixed data not found bug
  - ❖ Added image preview and file name display
  - ❖ Improved styling for better UX

- **Final Validation:**
  - ❖ Project meets objectives of predict traffic Telligence
  - ❖ Easy for non- technical users