Research Trends International Journal of Technology and Innovation
An International Peer-Reviewed, Refereed & Open-Access Journal
ISSN : 3144-3273
Available Online at : https://rtijti.com/
doi : https://doi.org/10.5281/zenodo.23720386

A Data-Driven Approach to Intelligent Traffic Control and Congestion Classification

Prateek Rajput[1], Nikhil Jalodiya[2], Pranjali Bhagwat[3], Roshan Vyas[4], Sumit Sharma[5]
[1, 2, 3, 4] Students, [5] Faculty Guide, Department of Data Science, Oriental College of Technology, Bhopal, Madhya Pradesh, India
prateekrajput202122@gmail.com, nikhiljalodiya121105@gmail.com, pranjalibhagwat802@gmail.com, vyasroshan9000@gmail.com, sumitsharma@oct.ac.in

ARTICLE INFO

Article History:
Received : 25-10-2026
Accepted : 25-10-2026
Published : 26-10-2026

Publication Issue
Volume 1, Issue 4
October – December 2026

Page Number
34-38

ABSTRACT

Traffic congestion lengthens journeys, increases fuel use, and delays emergency vehicles. Fixed-time signals cannot adjust to demand that changes through the day, while fully adaptive systems need a lot of sensing hardware. This paper describes a prototype web-based Smart Traffic Management System that classifies congestion as LOW, MEDIUM, or HIGH and suggests a green time for each signal.

The system uses a React dashboard, a Node.js API, Supabase (PostgreSQL) for storage, and a Random Forest classifier. The model is trained on a simulated dataset of 50,000 rows covering 20 roads and signals in Bhopal. The inputs are the number of vehicles, average speed, previous traffic density, hour, day of week, weather, and event type. On a random 80/20 split the deployed model reached an accuracy of 88.22% and a macro F1 score of 88.51%. Logistic regression and a 200-tree Random Forest gave almost the same results, which shows that the simulated classes can be separated using only a few variables.

Because the data is simulated and the split is random, the results show that the pipeline works end to end, but they do not show any benefit on real roads. The system has not yet been tested with real sensor data.

Index Terms - Smart Traffic Management, Congestion Classification, Random Forest, Signal Timing, Emergency Vehicle Priority, Simulated Data

I. INTRODUCTION

Congestion lengthens journeys, increases fuel use, and slows down ambulances, fire engines, and police vehicles. Most signals in Indian cities still run on fixed-time plans, which give the same green time at 8 AM and at midnight. Adaptive control can overcome this, but most adaptive systems depend on dense sensing such as loop detectors and cameras at every approach, which is expensive for a medium-sized city [3], [4].

1.1 Background and Motivation

The aim of this project is to build a low-cost prototype that a city such as Bhopal could try before investing in sensing hardware. The prototype has to predict how congested a road is from a small set of inputs, suggest a reasonable green time, let a traffic officer override that suggestion, and give emergency vehicles priority along a chosen route. It also has to be honest about what a model trained on simulated data can and cannot prove.

1.2 Contributions

The work makes the following contributions:

- A Random Forest classifier that labels congestion as LOW, MEDIUM, or HIGH from traffic, time, and context inputs.
- Simple rules that map the predicted class to a suggested green time, which an authorised officer

Copyright © 2026 The Author(s) : This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/)

can change.

- A web-based system with dashboards, notifications, reports, role-based access, and an emergency override.
- A clear statement of what the simulated results show and what they do not.

The system does not claim to reduce congestion, delay, or emergency response time. Such claims can only be made with real data and controlled field trials.

1.3 Organisation of the Paper

Section II reviews related work. Section III describes the dataset and Section IV the system design. Section V gives the methodology and Section VI the results. Section VII explains signal timing and the emergency override, Section VIII lists the limitations, Section IX outlines future work, and Section X concludes the paper.

II. RELATED WORK

A. Model Predictive Control for Signals: Ye et al. surveyed model predictive control (MPC) methods for traffic signal control. MPC predicts traffic states over a horizon and optimises signal settings, but it needs an accurate traffic model and real-time measurements [3].

B. AI-Based Adaptive Signal Control: Agrahari et al. reviewed artificial intelligence based adaptive traffic signal control, covering fuzzy logic, reinforcement learning, and deep learning methods. Most of the reviewed systems are evaluated in simulation, and field deployment remains limited [4].

C. Deep Graph Models for Forecasting: Li et al. proposed the Diffusion Convolutional Recurrent Neural Network (DCRNN), which models traffic on a road graph and forecasts speed on sensor networks. Such models are accurate but need large sensor datasets and GPU training [5].

D. Tree Ensembles for Tabular Data: Random Forest [1] and gradient boosting such as XGBoost [6] remain strong baselines for tabular data. They train quickly on a CPU, need little tuning, and are easy to run inside a web back end. scikit-learn [2] provides standard tools to compare such models.

E. Microscopic Simulation: SUMO is an open-source microscopic simulator widely used to test signal control strategies by measuring delay and queue length before field trials [7].

A comparison of the related work is given in Table 1, arranged from the most recent study to the oldest.

Table 1. Comparative analysis of related work

Ref.	Year	Technique	Key Finding	Limitation
[4]	2024	Review of AI-based adaptive control	Learning methods adapt green time	Tested mainly in simulation
[3]	2019	MPC survey	Optimises signals over a horizon	Needs accurate model and live data
[5]	2018	DCRNN (graph + RNN)	Accurate network forecasts	Needs dense sensors, GPU
[7]	2018	SUMO simulator	Tests delay and queue length	Simulation only
[6]	2016	XGBoost	Strong on tabular data	More tuning than RF
[2]	2011	scikit-learn	Standard model comparison	Library, not a system
[1]	2001	Random Forest	Fast, stable ensemble	No time ordering

This project focuses on building a prototype that can be deployed and used by officers, rather than on the best possible forecasting accuracy. It uses simulated data for Bhopal.

III. DATASET

The dataset (data/bhopal_traffic_50000.csv) is simulated. It contains 50,000 rows for 20 roads and signals in Bhopal. The simulator writes one row every 15 minutes, starting at 00:00 on 1 January 2025, so the 50,000 consecutive time steps end at 19:45 on 5 June 2026. Each row belongs to one of the 20 roads, so a road has about 2,500 rows on average rather than a reading at every 15-minute step.

Each row has seven input features: number of vehicles, average speed (km/h), previous traffic density, hour, day of week, weather, and event type. The target is the congestion class (LOW, MEDIUM, or HIGH), which is generated by the simulator. A summary of the data is given in Table 2.

Table 2. Summary of the simulated dataset

Property	Value
Rows	50,000
Roads / signals	20
Date range	1 Jan 2025 00:00 to 5 Jun 2026 19:45
Time interval	15 minutes (one row per step, one road per row)
MEDIUM class	43.5%
LOW class	37.4%
HIGH class	19.1%
Average number of vehicles	150.1
Average speed	31.9 km/h

IV. SYSTEM OVERVIEW

The front end is built with React, TypeScript, TailwindCSS, and Vite. A Node.js API handles requests, and Supabase provides the PostgreSQL database that stores traffic and application data. The deployed model is written in JavaScript so that it runs inside the API without a separate Python service. The comparison models were trained offline in Python using scikit-learn [2]. The architecture is shown in Figure 1.

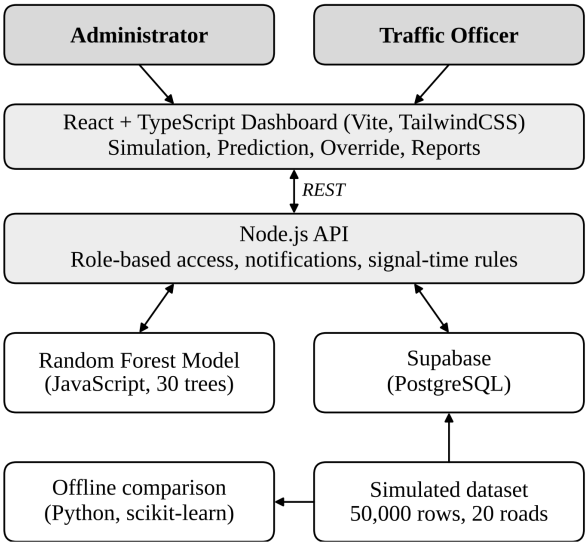


Figure 1. Architecture of the Smart Traffic Management System.

The system provides the following features:

- dashboard to view traffic data and run traffic simulations;
- congestion prediction and suggested signal time;
- emergency override along a selected route;
- notifications, analytics, and reports;
- road and intersection management.

Role-based access is enforced. Administrators manage roads and users, while traffic officers run simulations, manage signals, and handle emergencies. The working flow of a prediction is shown in Figure 2.

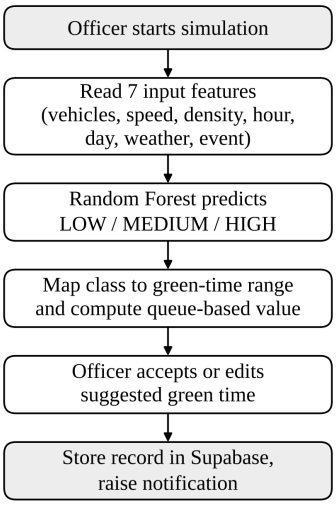


Figure 2. Working flow of congestion prediction and signal-time suggestion.

V. METHODOLOGY

5.1 Deployment Model

The deployed model is a Random Forest [1] with 30 trees, a maximum depth of 10, and a minimum leaf size of 20. It was trained on a random 80/20 split, which gives 40,000 rows for training and about 10,000 rows for testing. The small number of trees keeps the model file light enough to load in the Node.js API.

5.2 Evaluation Metrics

Let TP_c, FP_c, and FN_c be the true positives, false positives, and false negatives for class c, and N the number of test rows. The metrics are

$$\text{Accuracy} = (\Sigma_c\ TP_c) / N \quad (1)$$

$$P_c = TP_c / (TP_c + FP_c) \quad (2)$$

$$R_c = TP_c / (TP_c + FN_c) \quad (3)$$

$$F1_c = 2\ P_c\ R_c / (P_c + R_c) \quad (4)$$

Macro precision, recall, and F1 are the plain averages of the per-class values over the three classes, so that the smaller HIGH class carries the same weight as the larger MEDIUM class.

5.3 Comparison Models

A majority-class baseline, logistic regression, a decision tree (maximum depth 8), and a Random Forest with 200 trees were trained on the same split for comparison. These models use different hyperparameters from the deployed model.

VI. RESULTS

6.1 Deployed Model

The test-set results of the deployed model are given in Table 3. MEDIUM was the hardest class to predict. No LOW row was predicted as HIGH or the other way round, and most errors were between neighbouring classes.

Table 3. Results of the deployed Random Forest

Metric	Value
Rows trained	40,000
Rows tested	~10,000
Trees	30
Accuracy	88.22%
Macro precision	88.84%
Macro recall	88.21%
Macro F1	88.51%

6.2 Model Comparison

Table 4 and Figure 3 show that every trained model clearly beat the majority-class baseline. Logistic regression came within 0.3 percentage points of the Random Forest models. This means the simulated data is nearly linearly separable on the main input features, and the tree models are not finding any complex interactions.

Table 4. Comparison of models on the same test split

Model	Accuracy	Macro F1
Majority class	43.5%	20.2%
Logistic regression	88.1%	88.3%
Decision tree (max depth 8)	87.0%	87.3%
Random Forest (30 trees, deployed)	88.22%	88.51%
Random Forest (200 trees)	88.4%	88.6%

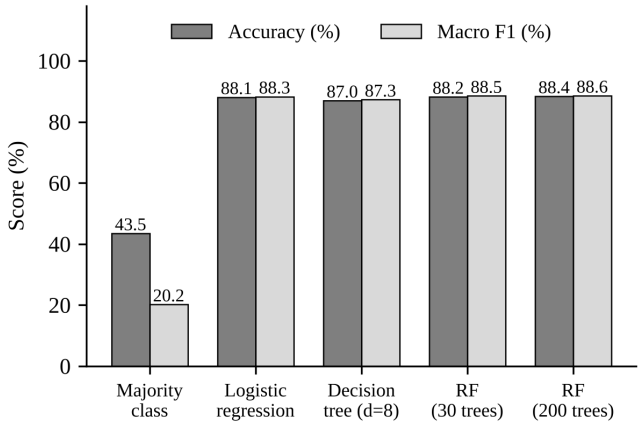


Figure 3. Accuracy and macro F1 of the compared models.

6.3 Feature Importance

The feature importances of the Random Forest are shown in Figure 4. Number of vehicles, previous density, and average speed together account for about 97.6% of the importance, while hour, day of week, event, and weather contribute very little. On real roads, time of day, weather, and events have a strong effect on traffic, so this pattern is a result of how the data was generated and is unlikely to carry over.

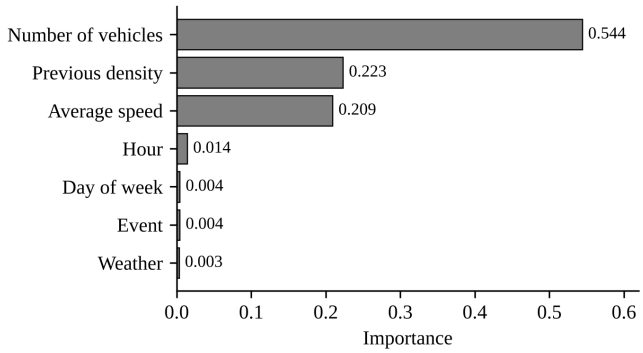


Figure 4. Feature importance of the Random Forest model.

6.4 Interpretation

Since the data is simulated, these results show how well the model recovers the simulator's labels, not how the system would perform on real roads. A random split can also overstate performance, because rows from neighbouring time steps can fall in both the training and test sets. Future work will evaluate the model on later time periods and on roads not seen during training.

VII. SIGNAL TIMING AND EMERGENCY OVERRIDE

7.1 Suggested Green Time

Each predicted class is mapped to a green-time range: LOW 40 to 65 s, MEDIUM 65 to 90 s, and HIGH 90 to 120 s. Within the range, a value g_q is computed from a simple queue estimate, and the suggested green time is kept inside the class limits as

$$g = \min(\max(g_q, g_{min,c}), g_{max,c}) \quad (5)$$

where $g_{min,c}$ and $g_{max,c}$ are the limits for the predicted class c. These ranges are chosen by the team and have not been validated on real intersections. An authorised officer can change the suggested value from the dashboard.

7.2 Emergency Override

During an emergency an officer selects a route and the type of vehicle to prioritise (ambulance, fire engine, or police). Signals along the route are set to green for 90 s

or more, and a critical notification is created. Routes are selected manually; there is no automatic routing or live GPS tracking. The effect of the override on emergency response time has not been measured.

VIII. LIMITATIONS

- The congestion labels are produced by the simulator's own rules, so the accuracy shows how well the model learns those rules, not how well it predicts real traffic.
- The random split may overstate performance because of the time dependency between rows.
- The signal-time ranges and queue calculation have not been tested on real intersections, and no delay or queue-length measurements were recorded.
- Some road and location details on the map are approximate.
- The system has not been tested at city scale with real sensors and signal controllers, and it does not show that congestion on real roads would fall.

IX. FUTURE WORK

- Train and test the model on real traffic data from Bhopal.
- Evaluate on unseen roads and later time periods instead of a random split.
- Test the model without the previous-density feature to measure how much it depends on past values.
- Forecast congestion 15 to 60 minutes ahead and compare with gradient boosting [6] and recurrent models [5].
- Compare the suggested signal plans with fixed-time plans in SUMO [7] using delay and queue length.
- Add automatic emergency routing and live vehicle location.

X. CONCLUSION

This paper presented a prototype system that classifies congestion from simulated traffic data for Bhopal, suggests signal timings, and provides a manual emergency override through a web dashboard. The deployed Random Forest reached 88.22% accuracy and 88.51% macro F1 on a random test split, and logistic regression performed almost the same. Because the data is simulated and the split is random, the results confirm that the pipeline works but do not show any benefit on real roads. Real data, time-aware and road-aware evaluation, and controlled field trials are needed before drawing conclusions about traffic flow in Bhopal.

ACKNOWLEDGMENT

The authors thank the Department of Data Science, Oriental College of Technology, Bhopal, for academic support.

REFERENCES

[1] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5-32, 2001, doi: 10.1023/A:1010933404324.

[2] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825-2830, 2011.

[3] B.-L. Ye et al., "A survey of model predictive control methods for traffic signal control," *IEEE/CAA Journal of Automatica Sinica*, vol. 6, no. 3, pp. 623-640, 2019, doi: 10.1109/JAS.2019.1911471.

[4] A. Agrahari et al., "Artificial intelligence-based adaptive traffic signal control system: A comprehensive review," *Electronics*, vol. 13, no. 19, Art. no. 3875, 2024.

[5] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *Proc. Int. Conf. Learning Representations (ICLR)*, 2018.

[6] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, 2016, pp. 785-794, doi: 10.1145/2939672.2939785.

[7] P. A. Lopez et al., "Microscopic traffic simulation using SUMO," in *Proc. 21st IEEE Int. Conf. Intelligent Transportation Systems (ITSC)*, 2018, pp. 2575-2582, doi: 10.1109/ITSC.2018.8569938.