



SMART TRAFFIC CONGESTION PREDICTION USING MACHINE LEARNING

Santhana Preethi B S¹, Mrs. J. Steffi²

¹II MCA, Reg.No: 25081205300112025, Department of Computer Applications and Research Centre, Sarah Tucker College (Autonomous), Tirunelveli. ²Assistant Professor, Department of Computer Applications and Research Centre, Sarah Tucker College (Autonomous), Tirunelveli.

ABSTRACT :

Traffic congestion is a major challenge in urban transportation systems because changing vehicle density, road occupancy, speed, weather, traffic-signal conditions, accidents, and temporal patterns can alter congestion levels. This paper presents a machine learning-based Smart Traffic Congestion Prediction System that classifies traffic conditions into Low, Medium, and High congestion levels. The proposed system uses a dataset of 5,000 traffic records and eleven predictive attributes, including source, destination/designation, vehicle count, traffic speed, road occupancy, traffic-light state, weather condition, accident report, sentiment score, hour, and day-of-week information. Categorical attributes are encoded using one-hot encoding, while numerical attributes are passed through the preprocessing pipeline. A Random Forest classifier with 100 trees, balanced class weighting, and a fixed random state is trained using an 80:20 stratified train-test split. Experimental evaluation on 1,000 test records produces an accuracy of 93.20%, weighted precision of 93.25%, weighted recall of 93.20%, and weighted F1-score of 93.21%. The trained model is integrated into a Flask web application, with SQLite used to store prediction history and dashboard statistics. The system provides a practical interface for entering traffic conditions and obtaining congestion predictions with model confidence. The study demonstrates how a lightweight machine learning and web-based architecture can support data-driven traffic monitoring and congestion assessment.

Keywords—Traffic congestion prediction, machine learning, Random Forest, intelligent transportation systems, Flask, SQLite, traffic classification.

I. INTRODUCTION

Rapid urbanization, increasing vehicle ownership, limited road capacity, and changing travel patterns have made traffic congestion an important problem for intelligent transportation systems (ITS). Congestion increases travel time, fuel consumption, and operational inefficiency, while also creating challenges for traffic management and route planning. Recent research shows that traffic prediction has evolved from statistical approaches toward machine learning and deep learning methods capable of modelling nonlinear and temporal relationships in traffic data [1], [2].

Traffic conditions are influenced by several interacting variables rather than by vehicle volume alone. Road occupancy, average speed, signal state, weather, accidents, time of day, and day-of-week patterns can provide complementary information for identifying congestion states. Machine learning is useful in this setting because classification models can learn relationships among heterogeneous traffic attributes and map them to predefined congestion categories.

This work develops a Smart Traffic Congestion Prediction System using a Random Forest classifier. The system is designed as an end-to-end application: traffic data are prepared and transformed, the model is trained and evaluated, and the trained model is deployed through a Flask web interface. Prediction records are stored in SQLite so that recent predictions and summary statistics can be displayed through the dashboard.

The main contributions of this work are: (1) development of a multiclass traffic-congestion classifier using heterogeneous traffic features; (2) an 80:20 stratified evaluation protocol on 5,000 records; (3) analysis of class-wise performance and feature importance; and (4) integration of the trained model with a Flask and SQLite application for interactive prediction and history management.

II. LITERATURE REVIEW

Traffic prediction is an established research area within ITS. Reviews of traffic prediction methods describe the use of statistical models, traditional machine learning, and deep learning for forecasting traffic flow, speed, and congestion [1], [2]. Urban traffic prediction is particularly challenging because traffic conditions exhibit temporal and spatial variation and are affected by external factors such as weather, events, and human activity [2].

Shaygan et al. reviewed artificial-intelligence-based traffic prediction methods and discussed preprocessing, data types, prediction tasks, and emerging research directions [1]. Xie et al. examined machine-learning approaches for urban flow prediction from spatiotemporal data and highlighted the importance of temporal and contextual factors [2]. More recent surveys report increasing use of deep learning, graph-based learning, and attention-based approaches for complex traffic forecasting, while also noting practical issues such as data sparsity, generalization, computational cost, and deployment constraints [3].

Random Forest remains relevant for structured traffic data because it can model nonlinear relationships and interactions without requiring a highly complex neural architecture. Breiman introduced Random Forest as an ensemble learning method based on multiple decision trees and randomized feature selection [4]. Recent IEEE conference work has also investigated Random Forest and hybrid approaches for traffic prediction and management, including systems that classify congestion states from vehicle and temporal features [5], [6].

The present work differs in emphasis by combining a compact multiclass congestion classifier with an application-oriented deployment pipeline. Instead of treating model evaluation and web deployment as separate tasks, the proposed system connects dataset preparation, preprocessing, prediction, storage, and dashboard visualization into one workflow.

III. RESEARCH GAP AND OBJECTIVES

Existing traffic prediction studies demonstrate the effectiveness of machine learning and deep learning, but many systems focus primarily on forecasting traffic flow or speed. A practical student-scale intelligent transportation application also requires a consistent classification target, interpretable structured inputs, reproducible evaluation, and an accessible interface. In addition, models developed for one dataset may not directly generalize to another traffic environment; therefore, reported performance must be interpreted in the context of the dataset and evaluation protocol.

The proposed work addresses this application-level gap by using a balanced three-class congestion target and combining traffic, environmental, temporal, and contextual variables in a single Random Forest pipeline. The system also records prediction outputs so that the web application can present historical and summary information.

- To classify traffic conditions into LOW, MEDIUM, and HIGH congestion levels.
- To preprocess categorical and numerical traffic attributes using a reproducible pipeline.
- To evaluate the trained classifier using accuracy, precision, recall, F1-score, and a confusion matrix.
- To identify influential input attributes using aggregated Random Forest feature importance.
- To deploy the trained model through a Flask web application with SQLite-based prediction history.

IV. PROPOSED METHODOLOGY

The methodology follows the same research-oriented sequence used in the reference paper: dataset preparation, preprocessing, feature selection, train-test splitting, model development, evaluation, model saving, and web-based prediction. The complete workflow is shown conceptually in Fig. 1.

Proposed Smart Traffic Congestion Prediction Architecture

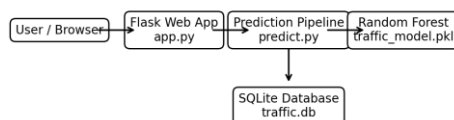


Fig. 1. Proposed architecture of the Smart Traffic Congestion Prediction System.

A. Dataset Preparation

The project dataset contains 5,000 records and 15 columns. The raw/project data include Timestamp, Source, Designation, Vehicle_Count, Traffic_Speed_kmh, Road_Occupancy_%, Traffic_Light_State, Weather_Condition, Accident_Report, Sentiment_Score, hour, day_of_week, day_of_week_num, congestion_score, and congestion_level. The target variable is congestion_level with three classes: LOW, MEDIUM, and HIGH.

The class distribution is nearly balanced: 1,667 HIGH records, 1,667 LOW records, and 1,666 MEDIUM records. This distribution reduces the risk that a classifier can obtain high aggregate performance simply by favouring a dominant class.

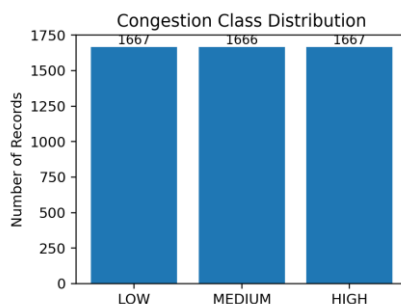


Fig. 2. Distribution of the three congestion classes in the project dataset.

B. Feature Selection and Engineering

The model uses eleven predictive attributes: Source, Designation, Vehicle_Count, Traffic_Speed_kmh, Road_Occupancy_%, Traffic_Light_State, Weather_Condition, Accident_Report, Sentiment_Score, hour, and day_of_week_num. Source, Designation, Traffic_Light_State, and Weather_Condition are categorical features. The remaining attributes are treated as numerical features. Temporal information is represented by hour and day_of_week_num. These features allow the classifier to learn recurring traffic patterns across different times of day and days of the week. The project pipeline also retains source and designation fields so that the web application can associate a prediction with a selected journey context.

C. Data Preprocessing

A ColumnTransformer is used to apply one-hot encoding to the categorical features while passing numerical features through the pipeline. OneHotEncoder is configured with handle_unknown='ignore', allowing the deployed application to process previously unseen categorical values without failing during transformation. The preprocessing and classifier are saved together as a single scikit-learn Pipeline.

D. Train-Test Splitting

The dataset is divided into 80% training data and 20% testing data using train_test_split with random_state=42 and stratification by congestion_level. Consequently, 4,000 records are used for training and 1,000 records are reserved for testing. Stratification preserves the approximate class proportions in both subsets.

E. Random Forest Classifier

The prediction model is a RandomForestClassifier with 100 decision trees, random_state=42, class_weight='balanced', and parallel execution enabled. Random Forest is an ensemble method that combines the outputs of multiple decision trees. For this structured traffic dataset, the ensemble can represent nonlinear relationships between traffic variables and congestion classes while remaining comparatively straightforward to deploy.

For an input vector x , the forest aggregates the predictions of its trees. In classification, the predicted class can be represented as the class receiving the largest number of tree votes. The implementation additionally uses predict_proba() to obtain class probabilities, and the maximum class probability is displayed as a confidence value in the application.

F. Evaluation Metrics

The classifier is evaluated using accuracy, weighted precision, weighted recall, and weighted F1-score. Accuracy is the proportion of correctly classified test instances. Precision measures the proportion of predicted instances of a class that are correct, while recall measures the proportion of actual instances of a class that are correctly identified. The F1-score is the harmonic mean of precision and recall. A confusion matrix is used to inspect class-specific errors.

V.EXPERIMENTAL RESULTS AND DISCUSSION

A. Overall Model Performance

The trained model was evaluated on the 1,000-record test set. The project reports an accuracy of 93.20%, weighted precision of 93.25%, weighted recall of 93.20%, and weighted F1-score of 93.21%. These values indicate that the model correctly identifies most congestion states on the held-out test data.

Metric	Value
Dataset size	5,000
Training samples	4,000
Testing samples	1,000
Accuracy	93.20%
Weighted Precision	93.25%
Weighted Recall	93.20%
Weighted F1-score	93.21%

TABLE I. OVERALL PERFORMANCE OF THE RANDOM FOREST MODEL.

B. Class-wise Evaluation

Table II presents the class-wise evaluation. LOW achieves the highest recall among the three classes, while MEDIUM has comparatively lower precision and F1-score. The confusion matrix shows that most errors occur between adjacent congestion categories, particularly between MEDIUM and HIGH, rather than between LOW and HIGH.

Class	Precision	Recall	F1-score	Support
HIGH	0.95	0.92	0.94	334
LOW	0.96	0.96	0.96	333
MEDIUM	0.89	0.91	0.90	333

TABLE II. CLASS-WISE PERFORMANCE ON THE TEST SET.

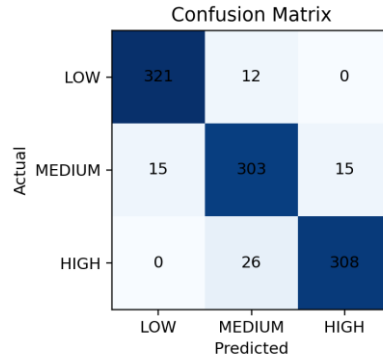


Fig. 3. Confusion matrix obtained from the current 5,000-record project dataset and the 80:20 stratified split.

C. Feature Importance

Aggregated Random Forest feature importance indicates that Road_Occupancy_% contributes the largest share of model importance (approximately 0.366), followed by Vehicle_Count (0.274) and Traffic_Speed_kmh (0.183). Sentiment_Score, hour, day_of_week_num, weather condition, traffic-light state, and accident report contribute smaller shares. Source and Designation have near-zero aggregate importance in the fitted model. Feature importance is model-specific and should not be interpreted as proof of causal influence.

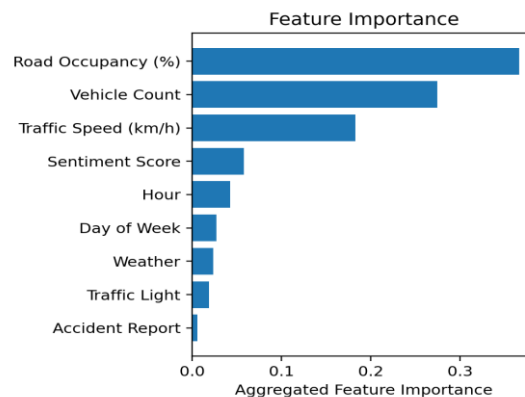


Fig. 4. Aggregated feature importance of the trained Random Forest model.

D. Discussion

The results show that the proposed classifier can distinguish the three congestion categories with high performance on the project test set. The strong contribution of road occupancy, vehicle count, and speed is consistent with the interpretation of congestion as a condition associated with road utilization and movement speed. However, the evaluation is based on one project dataset and one fixed train-test split; therefore, the reported 93.20% accuracy should not be interpreted as universal performance for all cities or traffic environments.

The medium class produces more confusion with HIGH and LOW than the other classes. This is plausible for boundary cases where traffic conditions lie near category transitions. Further research can examine probability calibration, threshold strategies, additional spatial features, and time-series models to improve robustness.

VI. SYSTEM IMPLEMENTATION

A. Web Application

The trained model is integrated into a Flask application. The main route displays the input interface, while the /predict route receives traffic information through a POST request, validates numerical ranges, invokes predict_congestion(), stores the result, and renders the prediction result page. The application also provides dashboard and API functionality for summary and historical traffic information.

B. Prediction Module

The prediction module converts submitted form values into a Pandas DataFrame with the same feature names expected by the trained pipeline. The model returns the congestion class through `predict()` and the maximum class probability through `predict_proba()`. The returned object contains `congestion_level` and confidence values.

C. Database and Dashboard

SQLite is used as the persistent storage layer. The predictions table stores timestamp, source, designation, vehicle count, speed, road occupancy, traffic-light state, weather condition, accident report, sentiment score, temporal features, predicted congestion level, and confidence. The dashboard retrieves recent predictions and summary statistics, allowing the application to show the operational history of model outputs.

D. End-to-End Workflow

The deployed workflow is: user enters source, destination/designation, and traffic attributes → Flask validates the input → prediction pipeline preprocesses categorical values → Random Forest predicts congestion → confidence is calculated → prediction is stored in SQLite → result page displays the predicted congestion level → dashboard/history provide previously stored predictions. This architecture separates the user interface, prediction logic, trained model, and data storage components.

VII. CONCLUSION

This paper presented a Smart Traffic Congestion Prediction System using machine learning and a Flask-based web application. A dataset of 5,000 traffic records was used with eleven predictive features covering traffic volume, speed, road occupancy, traffic-light state, weather, accidents, sentiment, location context, and temporal information. The Random Forest classifier was trained using an 80:20 stratified split and achieved 93.20% accuracy, 93.25% weighted precision, 93.20% weighted recall, and 93.21% weighted F1-score on the 1,000-record test set.

The system demonstrates an end-to-end approach in which model development is connected to practical deployment. The SQLite database maintains prediction history, while the Flask interface provides an accessible mechanism for generating and reviewing congestion predictions. The results support the feasibility of using structured machine learning models for traffic-congestion classification within a lightweight web application. The reported performance remains specific to the dataset and evaluation protocol used in this study.

VIII. FUTURE WORK

Future development can extend the current system in several directions. First, real-time traffic feeds, GPS data, IoT sensors, and road-network information can be incorporated to support continuous prediction. Second, spatial relationships between connected roads can be modelled explicitly rather than representing location only through categorical fields. Third, temporal forecasting models such as LSTM, GRU, temporal convolution, transformers, or graph neural networks can be evaluated against the current Random Forest baseline [3], [6].

Further work can also include hyperparameter optimization, probability calibration, cross-validation across multiple geographic areas, explainable-AI techniques such as SHAP, and monitoring for dataset drift. A mobile interface, map-based visualization, and alert mechanism could make the system more suitable for operational traffic-management scenarios. Such extensions should be evaluated on independent datasets and under realistic deployment conditions.

IX. REFERENCES

- [1] M. Shaygan, C. Meese, W. Li, X. G. Zhao, and M. Nejad, "Traffic prediction using artificial intelligence: Review of recent advances and emerging opportunities," *Transportation Research Part C: Emerging Technologies*, vol. 145, Art. no. 103921, 2022, doi: 10.1016/j.trc.2022.103921.
- [2] P. Xie, T. Li, J. Liu, S. Du, X. Yang, and J. Zhang, "Urban flow prediction from spatiotemporal data using machine learning: A survey," *Information Fusion*, vol. 59, pp. 1–12, 2020, doi: 10.1016/j.inffus.2020.01.002.
- [3] "A comprehensive review of traffic prediction: From traditional machine learning to AutoML," *Neurocomputing*, vol. 666, Art. no. 132148, 2026, doi: 10.1016/j.neucom.2025.132148.
- [4] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.
- [5] "AI and ML Based Traffic Prediction System to Improve City Commutes," in *Proc. 2025 2nd Int. Conf. on Artificial Intelligence, Metaverse, and Cybersecurity (ICAMAC)*, Dubai, UAE, 2025, doi: 10.1109/ICAMAC67779.2025.11398412.
- [6] "Machine Learning-Based Traffic Flow Prediction and Management," in *Proc. 2025 6th Int. Conf. on Inventive Research in Computing Applications (ICIRCA)*, Coimbatore, India, 2025, doi: 10.1109/ICIRCA65293.2025.11089788.