VISVESVARAYA TECHNOLOGICAL UNIVERSITY

BELAGAVI, KARNATAKA

A Mini-Project Work Report

on

"TRAFFIC PREDICTION SYSTEM USING HYBRID MODELS"

Submitted in the partial fulfillment for the award of

BACHELOR OF ENGINEERING

in

INFORMATION SCIENCE AND ENGINEERING

Mr. Nitesh. K. Doddamani	USN: 1BY24IS410
Mr. K. Manoj	USN: 1BY24IS406
Mr. Moksha Pawan K	USN: 1BY23IS122
Ms. Mahalakshmi. A	USN: 1BY23IS104

Under the guidance of

Dr. Savitha. S

Assistant Professor

Department of ISE, BMSITM

BMS INSTITUTE OF TECHNOLOGY & MANAGEMENT

YELAHANKA, BENGALURU – 560119

DEPARTMENT OF INFORMATION SCIENCE & TECHNOLOGY

2025-2026

ಬಿ.ಎಂ.ಎಸ್. ತಾಂತ್ರಿಕ ಮತ್ತು ವ್ಯವಸ್ಥಾಪನಾ ಮಹಾವಿದ್ಯಾಲಯ
BMS INSTITUTE OF TECHNOLOGY & MANAGEMENT
(Autonomous Institution Under VTU)
Yelahanka, Bengaluru -560119

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Mini-Project (BCS506) entitled **"TRAFFIC PREDICTION SYSTEM USING HYBID MODELS"** is a Bonafide work carried out by **NITESH. K. DODDAMANI (1BY24IS410), K MANOJ (1BY24IS406), MOKSHA PAWAN K (1BY23IS122) & MAHALAKSHMI A (1BY23IS104)** in partial fulfillment for the award of **Bachelor of Engineering Degree in Information Science and Engineering** of the Visvesvaraya Technological University, Belagavi during the year 2025-26. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in this report. The mini-project report has been approved as it satisfies the academic requirements with respect to mini-project for the B.E. Degree.

Signature of the Guide	**Signature of the HOD**
Signature of the Co-ordinator	**Signature of Cluster Head**

ABSTRACT

This project proposes an intelligent Traffic prediction system using Hybrid Models that leverages multi-modal data to deliver accurate and context-aware congestion forecasting for modern smart cities. Unlike traditional models that rely solely on historical traffic patterns, this system integrates real-time weather data, public events, temporal indicators, and road-network relationships to capture real-world dynamics more effectively. The architecture combines machine learning models such as Random Forest, XGBoost, Extra Trees, and Linear Regression with deep-learning (Time series models) approaches like LSTM enabling the system to learn both tabular patterns and sequential behaviors. The final hybrid ensemble boosts accuracy, stability, and generalization. A user-friendly dashboard visualizes live conditions, predicted flow, congestion zones, and influencing factors, transforming raw predictions into actionable insights. The system provides a scalable, data-driven framework for improving urban mobility, supporting traffic authorities, and enhancing commuter decision-making.

ACKNOWLEDGEMENT

We are happy to write a mini-project report after completing it successfully. This mini-project would not have been possible without the guidance, assistance, and suggestions of many individuals. We would like to express our deep sense of gratitude to each one who has helped us to make this mini-project a success.

We heartily thank **Dr. Sanjay H A, Principal, BMS Institute of Technology & Management** for his constant encouragement and inspiration in taking up this mini-project.

We heartily thank **Dr. Surekha K B, Head of the Department, Information Science and Engineering & Dr. Narasimha Murthy M S, Associated Head, Cluster-4, BMS Institute of Technology & Management** for their constant encouragement and inspiration in taking up this mini-project.

We gratefully thank our mini-project guide**, Dr. Savitha S, Assistant Professor, Department of Information Science and Engineering,** for her encouragement and advice throughout the course of the mini-project work.

We heartily thank mini-project coordinators **Prof. Vinay Kumar Y B, Assistant professor and Prof. Syed Matheen Pasha, Assistant Professor, Department of Information Science and Engineering,** for their constant follow-up and advice throughout the mini-project work.

Special thanks to all the staff members of the **Information Science and Engineering** Department for their help and kind co-operation.

Lastly, we thank our parents and friends for their encouragement and support given to us to finish this mini-project work.

By,

NITESH. K. D
K MANOJ
MOKSHA PAWAN K
MAHALAKSHMI A

Declaration

We, hereby declare that the mini-project titled "**Traffic Prediction System Using Hybrid Models**" is a record of original mini-project work undertaken for partial fulfilment of Bachelor of Engineering in Information Science and Engineering of the Visvesvaraya Technological University, Belagavi during the year 2025-26. We have completed this mini-project work under the guidance of **Dr. Savitha S, Assistant Professor, Department of Information Science and Engineering.**

I also declare that this mini-project report has not been submitted for the award of any degree, diploma, associate ship, fellowship or other titles anywhere else.




USN:	Nitesh. K. D	K. Manoj	Moksha. P. K	Mahalakshmi. A
Name:	1BY24IS410	1BY24IS406	1BY23IS122	1BY23IS104

Signature

Examiner's Signature

1.

2.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Overview of Intelligent Traffic Prediction

Smart traffic forecasting allows cities to better understand and handle the flow of congestion, thus making the roads safer. The rapid growth of urban agglomerations and road networks puts daily pressure on traffic congestion, which often turns out to be unforeseeably delayed. Traditional approaches to managing traffic, especially those based on fixed rules and simple averages, cannot capture the dynamic and irregular nature of real-world traffic.

These complex patterns have, until recently, defied interpretation, but advances in machine learning and deep learning now offer powerful tools. Current data-driven methods utilize large datasets continuously updated to capture spatial and temporal relationships in traffic. With this information, municipalities are able to forecast congestion hotspots, identify peak travel periods, and recognize anomalous surges. Accordingly, the impact of incidents becomes more accurately understood, thus enabling smarter navigation systems, real-time traffic monitoring, and long-term planning for sustainable urban mobility.

The paper proposes a hybrid traffic prediction framework that fuses various machine learning methods: Random Forest, XGBoost, Extra Trees, and Linear Regression with deep learning sequence models using LSTM. The goal is to construct a system that would allow for the study of structured features of the traffic against patterns varying in time. The joint architecture facilitates prediction at the following levels: citywide traffic dynamics and route congestion analysis. The code is organized modularly, with the main directories named app/, pages/, models/, and src/ for clarity in development and extensibility.

1.2 Need for Hybrid AI Models in Smart Mobility

Urban traffic behaviour arises from the interaction of many factors, including variability in weather, incidents, peak-period changes, public gatherings, and seasonal variations. Because of the nonlinear nature of these influences and without a single inherently suited descriptor that can characterize all system features as they may evolve over time, hybrid modelling has become one of the successful approaches in delivering a reliable and precise forecast of traffic. Models such as Random Forest, XGBoost, and Extra Trees are especially good at discovering complex spatial relationships and identifying the most influential features out of large datasets, hence providing robust general-purpose

predictions and enabling anomaly detection or sudden changes in traffic flows. Linear Regression, while simpler, provides transparency into interpreting broad linear trends present in the data.

However, these temporal dependencies, which also include cyclic congestion at the same time every day and long-range trends repeated after weeks or months, cannot be learned by ML models alone. The project uses an LSTM network specified for sequential time-series data analysis. An LSTM network learns the evolving nature of traffic conditions over time to make better predictions of future states. In this way, the system combines the complementary strengths of ML and DL techniques into a single hybrid ensemble. Benefits include the following:

a) Machine learning models for feature-based insights extraction and structured data handling.
b) Deep learning models are used for learning temporal behaviours and long-term dependencies.
c) Ensemble integration, which stabilizes the predictions and increases the overall accuracy.

In the overall context of smart mobility, such hybrid systems enable better route optimization and reduce traffic congestion, improve the planning of emergency responses and enable strategic city decisions development. With increasing modernization in urban ecosystems, such hybrid approaches to AI represent a solid basis for traffic management solutions that are scalable and adaptable.

CHAPTER 2

LITERATURE SURVEY

2.1 Critical Analysis of Literature

There are many different papers collected, but most of the traffic prediction research is based on either traditional machine learning models, such as Random Forest, XGBoost, Support Vector Regression, and Linear Regression, or deep learning approaches, notably Long Short-Term Memory networks. However, in most of these studies, only historical traffic data from a single city is used for prediction, and external factors like weather, events, public holidays, or usual public transport schedules, and incidents are generally ignored. Historically, time-based congestion patterns have not been captured fully by such approaches. Congestion data collection in real time through sensors can be relatively cost-effective; however, there exists a gap between research models and real-world implementations in smart cities.

2.2 Implications and Conclusion

It also shows that traffic prediction cannot rely on any single modeling approach in urban areas. Machine learning algorithms have the capability for understanding features, while deep learning models, especially LSTM, do a great job in learning temporal dynamics. There is a scope for hybrid systems that integrate several methodologies rather than relying on a single methodology. The proposed Traffic Prediction System for Smart Cities Using Hybrid Models will try to fill this gap by:

a) Integrating machine learning and deep learning models under a single framework.
b) Incorporating supplementary inputs like weather and temporal information.
c) Enabling decision-making for the benefit of users and authorities through an interactive dashboard presenting actionable insights.

2.3 Existing Systems

Several research works and deployments in cities are illustrative of the employment of single-model approaches: machine learning or deep learning applied to traffic signal control and traffic prediction in specific road segments or intersections. These models mainly leverage historical sensor data from a single city; multiple algorithms are not merged, and external factors like weather and events are also excluded quite often. User-friendly dashboards for real-time decision support are also missing. Thus, state-of-the-art systems are not sufficient for the realization of smart-city concepts and meeting the requirements of authorities and users for accurate, context-aware, real-time traffic predictions for larger networks.

SL. NO	Paper Title	Methodology/Technology Used	Outcomes	Limitations	Future Work
1	A Machine Learning Approach for Predictive Analysis of Traffic Flow	ML, DL, CNN	Timely Predictions	Not Robust, Limited History	Data Quality, Preprocessing
2	Experimental Analysis of Time Series Models for Traffic Flow Prediction	LSTM, RF, XGBoost	Accuracy: 95.83%	Limited Datasets	Integration of Real-time Factors
3	Real Time Traffic flow Prediction Using ML Algorithms	LSTM, SVR, ANN	ANN 98.2%	No Real-time Deployment	Hybrid Model, Real-time Integration
4	Traffic Prediction Using Machine Learning	LSTM, RF	Acc: 95.83% (LSTM), 94.86% (RF)	Limited Datasets	Richer Data Sources
5	Application of Deep Learning Models for Traffic Flow Prediction	LSTM, SVR, DT, KNN	LSTM: 88.91% (RÂ²)	Smaller Datasets	Diverse Real-world Data
6	Review of Dynamic Traffic Flow Prediction Methods	LSTM, GNN, CNN-LSTM, ARIMA	LSTM/GNN > Classical	CNN-LSTM slow	Heterogeneous Data Integration
7	Road Traffic Congestion Prediction using AI/ML	CNN-RNN Hybrid, RL, XGBoost	Hybrid Improved Accuracy	Generalization Challenges	Multi-modal/Spatial-Temporal Fusion
8	Ensemble Learning Algorithms for Traffic Flow	Bagging Ensemble, RF, LSTM, LR	Acc: 93.52%	No Image/Video Data	Add Sensor/Image Data
9	Deep Learning for Traffic	Bayesian DL, GCN, Transformer	Better Uncertainty	Data Hungry	Probabilistic Forecasting, Attention

	Probabilistic Prediction				
10	Hybrid LSTM + Graph Neural Network	LSTM, GNN, OSM, Google Maps API	Hotspot Detection	Real-time Data Dependent	Scalability, Transfer Learning
11	Traffic Prediction Analysis using Twitter & ML	DL, social data, weather	Sensor & Social Data	Reliance on Social Stream	Sensor/Video Feed Integration
12	Real-Time Bengaluru City Traffic Prediction	CNN, LSTM, SVR	CNN: 74.48%, LSTM: Lower	Sensor/Crowd Limited	Video Sensor Fusion
13	Intelligent Transport System for Urban Traffic Prediction	XGBoost, LSTM, GRU	Reduced Travel Time/Emissions	Validation Needed	IoT/Edge/Cloud Integration
14	Spatio-temporal Evolutional Graph Neural Network	STGCN, ASTGCN, STSGCN	Low RMSE	Needs Multi-modal Data	Semantic Graph Updating
15	Spatio-Temporal Graph CNN for Traffic Forecasting	STGCN	High Accuracy, Low RMSE	Computational Complexity	Multi-task, Multi-agent Prediction
16	Improved Traffic Congestion Prediction Using ML/XGBoost	ML, XGBoost	Acc: 90%	Needs Robust Validation	Larger/Smart City Datasets
17	Leveraging ML to Predict Traffic Jams	Advanced ML	Accident & Congestion Risk	Data Complexity	Multi-agent/Routing Expansion
18	ML Adaptive Traffic Prediction & Control	Edge-ML	Real-time Optimization	Sparse Ground-truth Data	Multi-city Implementation
19	Traffic Flow Prediction w/ Multi-	DL, Attention	Higher Accuracy	Needs More Benchmarking	Transfer Learning

	dimensional Feature Input				
20	Traffic Forecasting Using GNNs	GNNs, LSTM, ARIMA	Up to 15% Higher Accuracy	High Computational Cost	Hybrid GNNs, Deployment
21	Enhancing Urban Traffic Congestion Prediction Through Hybrid Deep Learning	Hybrid DL (LSTM + Ensemble)	Novel hybrid ensemble; high accuracy	Computational overhead	Scalable for city-wide deployment
22	Hybrid Deep Learning-Based Traffic Congestion Control in IoT	DL, Arithmetic Optimization, LSTM, Deep CNN	Improved congestion control (IoT context)	IoT-sensor limitations	Integration with real-time traffic feeds
23	Urban Traffic Prediction Using Hybrid XGBoost-LSTM Model	XGBoost, LSTM	Accuracy: 93%, high precision	Data imbalance, external feature reliance	Integration with city sensors
24	Transfer Learning in Traffic Prediction with Graph Neural Networks	Transfer Learning, GNN	Multi-city/region test, accuracy boost	Domain adaptation challenges	Robust deployment for smart cities
25	Extreme Gradient Boosting Algorithm Based Urban Daily Traffic Prediction	XGBoost	Prediction > 90% accuracy for daily peak	Limited to daily-level, city-specific	Benchmark to more granular intervals

Table 2.1 Literature review

CHAPTER 3

REQUIREMENT ANALYSIS

In this respect, requirement analysis is a crucial, foundational stage in intelligent systems engineering, especially those whose enablement relies on advanced computation approaches like machine learning, deep learning, and real-time analytics. In ITS, good traffic prediction is impossible without a well-defined set of functional capabilities, system constraints, and environmental specifications, besides robust predictive models. The aim of this phase will be to ensure that the system architecture, design decisions, and strategies for implementation comply with stakeholder expectations, operational objectives, and performance benchmarks. A hybrid traffic prediction framework, which embodies several machine learning models including Random Forest, XGBoost, Extra Trees, Linear Regression, along with a deep learning-based LSTM architecture, requires requirement analysis as an indispensable means for their full coordination, efficient data flow, and reliable model inference.

Considering that the system works by the integration of interactive visualization dashboards, modular execution scripts, and multilevel prediction capabilities, that is city-level and route-based forecasting, the requirement analysis necessarily has to be extended toward algorithmic performance, user interaction, scalability, deployment environment, and real-world constraints such as data variability and system load.

This section delineates the functional requirements that comprehensively define the operational system behavior, together with the software and hardware resources required to support data processing, model computation, and user interface rendering. All these, in summary, set a structured basis for the design, development, and evaluation of the hybrid traffic prediction system, in order to guarantee that the final solution is correct, scalable, maintainable, and suitable for smart mobility applications in dynamic urban contexts.

3.1 Functional Requirements

Functional requirements specify the operations, computing functions, and interactions that the system should provide. The hybrid traffic prediction system will be realized as a scalable architectures capable of data processing, inference, visualization, and model analysis. These are some of the requirements that govern core functionalities within the system:

FR1: Data Acquisition and Preprocessing

a) This will provide support for the traffic data set ingestion in CSV or JSON format.
b) It shall perform noise removal and other pre-processing operations such as feature encoding, Normalization, scaling, time indexing and outlier detection are some of the steps involved in preprocessing.
c) Preprocessing pipelines - e.g. preprocessor.pkl, label_encoder.pkl shall be automatically loaded and used in both training and prediction phases.
d) The system shall produce clean, structured data in suitable formats for consumption by ML and DL models.

FR2: Machine Learning-Based Traffic Prediction

a) Load in the /models directory all ML models, including Random Forest, XGBoost, Extra Trees, and Linear Regression.
b) It shall then make the prediction using every ML algorithm and assess the performance of each using stored metrics files: model_results_ml.csv.
c) The system shall provide interpretability through feature importance analysis where applicable.

FR3: Deep Learning-Based Time-Series Forecasting

a) The LSTM model (lstm_model.keras) grabs time-based trends from traffic data with this method - using patterns it finds step by step.
b) The system sets up time windows, moves through input steps one by one, yet runs model forecasts using deep learning inside /src.
c) The LSTM bit predicts traffic further ahead, meanwhile dumping performance stats into model_results_dl.csv.

FR4: Hybrid Ensemble Prediction

a) The system shall perform the integration of ML and DL outputs through the hybrid ensemble mechanism.
b) It shall point out the best performing model using metadata files: best_model_info.json, the final selected model is saved as best_model_selected.pkl.
c) The predictions made by an ensemble will have better accuracy and reliability compared to individual models.

FR5: City-Level Traffic Forecasting

a) The module 1_City_Traffic_Prediction.py shall predict congestion levels, traffic density variation, and temporal traffic surges across zones of cities.
b) It shall present results through graphical outputs to aid decision-makers in interpretation.

FR6: Route-Level Traffic Prediction

a) The system shall, using the script, make forecasts of traffic intensities along given routes.
b) It shall provide the facility of feeding source and destination by the user and give congestion prediction for segmented road ways.
c) Route-based results will provide navigation guidance and micro-level traffic insights.

FR7: Analytical Dashboard and Visualization

a) The analytics dashboard, 3_Analytics_Dashboard.py, should display visual insights including model comparisons, historical traffic trends, peak-hour graphs, and temporal heatmaps.
b) The UI shall be interactive, generated using Streamlit or equivalent frameworks, and integrated through app.py.
c) They are displayed as bullet points Real-time responsiveness by the dashboard to user queries.

FR8: Model Evaluation and Reporting

a) The system shall use MAE, RMSE, accuracy, among other metrics, to evaluate the prediction models. And R^2 score.
b) Results will be stored in structured files; CSV format and should be displayed within dashboard modules.
c) The system shall support continuous model improvement by allowing training on an updated datasets.

FR9: User Interface and System Interaction

a) The system shall provide an intuitive UI to select prediction categories, upload datasets, and result visualization.
b) All outputs will be rendered in a user-friendly, visual coherent manner of presentation for technical and non-technical users.

3.2 Software and Hardware Requirements

Proper software tools are needed to implement a hybrid ML-DL traffic prediction model. Frameworks and hardware for computation. This will discuss the basics and recommended operating modes for maximum efficiency.

3.2.1 Software Requirements

Category	Specification
Operating System	Windows 10/11, Ubuntu 20.04+, or equivalent Linux environment
Programming Language	Python 3.9 or later
Core Libraries	NumPy, Pandas, Scikit-Learn, TensorFlow/Keras, Matplotlib, Seaborn, XGBoost
Visualization Framework	Streamlit, Plotly, Matplotlib
Development Tools	Visual Studio Code, Jupyter Notebook, GitHub
Dataset Formats Supported	CSV, JSON
Execution Environment	Virtual Environment (venv) with PIP dependency management
Project Scripts	app.py, main.py, and ML–DL modules under /src

Table 3.1 Software requirements

These tools make it possible to automate preprocessing, model training, prediction, and visualization.

3.2.2 Hardware Requirements

Component	Minimum Specification	Recommended Specification
Processor	Intel Core i5 / AMD Ryzen 3	Intel Core i7 / AMD Ryzen 5 or higher
RAM	8 GB	16–32 GB (for LSTM training and data-heavy operations)
Storage	2–5 GB free disk space	SSD storage for fast I/O and model loading
GPU	Not required	NVIDIA CUDA-enabled GPU for accelerated DL computation
Display	1080p Standard Monitor	High-resolution display for dashboard analytics

Table 3.2 Hardware requirements

These hardware specifications guarantee stable model performance. They allow for faster training and smooth dashboard rendering.

CHAPTER 4

SYSTEM DESIGN

4.1 Overview of System Design

The modular architecture design will provide flexibility, model interoperability, and other such features within the system. Extensibility: The design follows a layered approach where every layer performs a set of functions critical for correct traffic prediction.

a) Data Ingestion Layer

- It processes raw data input, fetching sets of traffic data either from local sources or those uploaded by the user.
- It supports CSV and JSON formats for correct loading of historical and real-time data.

b) Preprocessing & feature engineering layer

- This component shall do data cleaning, handle missing values, normalization of data, label encoding, and finally the generation of sequences for time-series modeling.
- The preprocessing artifacts of preprocessor.pkl and label_encoder.pkl are loaded dynamically.

c) Machine Learning Prediction Layer

- This layer unifies the classical ML algorithms such as Random Forest, XGBoost, Extra Trees, and Linear Regression. Each model takes in engineered features and outputs predictive results.
- The stored models under the /models directory are loaded for inference.

d) Deep learning prediction layer

- The LSTM model, lstm_model. keras, which consists of the core of this DL component, takes in sequential.
- Data to capture temporal relationships and output multi-step traffic forecasts.

e) Hybrid ensemble layer

- The ensemble module integrates the ML and DL predictions to come up with an optimized final prediction of the traffic.
- Model selection information best_model_info.json assures that the top models performing Contribute to the final output.

f) Application and Visualization Layer

- This layer includes an interactive dashboard and user interface app.py, pages/.
- Visual analytics grants users the ability to view predictions, historical trends, congestion levels, and model comparisons.

g) Execution & Integration Layer

- The scripts under /src ensure that all modules integrate well for training, model evaluation, and saving.
- This Layer keeps modular execution and system-level control.

Overall, the system design makes sure complex predictive computations are abstracted behind a user. A friendly, efficient, scalable architecture.

4.2 Data Collection

Data collection forms the very foundation of the traffic prediction system, considering the performance of machine learning and deep learning models, which heavily rely on the quality, diversity, and representativeness of the dataset.

a) Data Source

- Traffic data can either be obtained from open-source data sets, simulation-generated data sets, or city level transportation data repositories.
- Normally, a dataset is described by features related to traffic volume, vehicle count, speed, lane occupancy, timestamps, weather condition, and road identifiers.

b) Data Types Collected

- Temporal Data: Time-stamped traffic flow values that can be used for LSTM-sequence learning.
- Spatial Data: Route-level and city-level identifiers to enable localized predictions.
- Environmental Factors : Weather, peak-hour indicators, holiday/event patterns.
- Historical Records: These are multi-day or multi-week traffic logs that allow for the extraction of trends.

c) Data storage and organization

- Collected data are stored in the /data directory and formatted according to standardized preprocessing formats.
- Following initial processing, files such as ML_train_data.csv and ML_test_data.csv are created for training and test datasets.

d) Data Integrity and Preprocessing Requirements

- Data cleaning to remove noisy or incomplete entries.
- Handling of missing values
- Normalization or scaling of ML models.
- Creation of time series windows for LSTM input.
- Label encoding for categorical variables.

e) Dataset Usage in the System

- ML models run on structured, feature-engineered data.
- LSTM models leverage sequential temporal data in order to understand long-term dependencies.
- Analytics dashboards visualize data that has been aggregated and processed to yield insights.

Therefore, the strategy for data collection gives the system high-quality inputs, enabling the accuracy and reliability of the traffic prediction over both ML and DL components.

4.2.1 Historical Traffic Dataset

This research involves large and historical traffic flow datasets taken from publicly accessible repositories like Kaggle, data.gov.in, among others. The datasets shall form the basis for training in both the machine learning and deep learning parts of the model. Common characteristics involved in these datasets include the following:

a) Traffic flow or vehicle volume with timestamps.
b) Identifiers of road segments or routes.
c) Basic environmental indicators, where available.
d) Traffic density or congestion level annotations.

These structured datasets enable traditional machine learning models such as Random Forest, XGBoost, Extra Trees, and Linear Regression to learn static and nonlinear relationships in the data of traffic. In the case of deep learning, these datasets are used to extract temporal sequences that will allow the LSTM model to learn traffic flow patterns across minutes, hours, and days. The pre-processed datasets are maintained at the /data directory and include ML_train_data.csv, ML_test_data.csv, and final_dataset_preprocessed.csv for use systematically in model training, evaluation, and comparative analysis.

4.2.2 Real-Time Weather Data

Including environmental factors in traffic prediction, the system retrieves real-time weather conditions. Data are gathered from WeatherAPI.com by means of the WEATHERAPI key set during development. The weather conditions have a great impact on the behaviors of traffic, especially in terms of rainfall, fog, and changes in humidity. The Weather API returns structured and reliable data on:

a) Temperature
b) Humidity
c) Rainfall/ precipitation intensity
d) Percentage of cloud cover
e) Visibility range

Wind speed and direction (optional) These live weather parameters form a set of dynamic input features to the prediction process, enabling the system to dynamically adjust the predictions according to the ambient conditions. The Weather API improves the real-time accuracy of the short-term congestion forecast and route risk assessment.

4.2.3 Geolocation, Routing, and Traffic Flow Data (TomTom APIs)

This system uses geospatial information to make route-based traffic predictions and provide real-time insights for navigation using the TomTom API ecosystem via the TOMTOM_API_KEY set in the back end. TomTom offers a set of APIs that provide accurate route mapping, monitoring of congestion in real time, and geocoding functionality. Interactions with TomTom API services occur as follows:

a) TomTom Search API - Geocoding

- Translates user-specified place names into geographic coordinates: latitude and longitude.
- Performs exact locational lookups for origin and destination points.
- Ensures accuracy in route segment mapping during predictions.

b) TomTom Routing API

- Identifies origin–destination pairs and applies them to create optimized driving routes.
- Provides detailed route attributes, including:
  - Estimated travel duration
  - Distance
  - Route geometry (shape points)
  - Alternative routes

- Supports real-time, route-specific congestion predictions computations.

c) TomTom Traffic Flow API

- Live traffic speed data, including:
  - Current speed
  - Free-flow speed
  - Congestion index
  - Traffic density indicators
- Serves as a real-time signal for providing accurate congestion forecasts.

The integration of the TomTom APIs allows the system to provide route-specific predictions at a city level, inclusive of congestion estimation and map-based insights, hence making the model more applicable to smart mobility applications.

CHAPTER 5

METHODOLOGY

This approach explains in detail a clear and specific route for developing, constructing, instructing, and deploying an intelligent hybrid model for traffic prediction. It combines various sources of information, intelligent learning models, deep learning architectures, and real-time API services to provide traffic forecasting for city-wide scales or single routes. Merging static feature learning with the detection of temporal relationships increases the accuracy of predictions. The methodology proceeds along several large-scale dimensions.

5.1 Data Acquisition Pipeline

The tool pulls in data from several sources and keeps its accuracy with added variables to make good predictions.

a) Historical Traffic Dataset

Source: Kaggle datasets, with some elements drawn from data.gov.in. In all, these datasets cover details of traffic, timing, and road data, providing minute-by-minute granular movement. The aim is to construct an algorithm that would be able to detect movements and learn the evolution of its pattern.

b) WeatherAPI.com Real-Time Weather Data

Live measurements include temperature, humidity, precipitation, cloud cover, visibility, and wind direction. While changes in weather continue outdoors, the road conditions also change, setting the pace for monitoring each and every change.

c) TomTom geocoding, routing, and traffic flow APIs

Geocoding: It connects the address to map coordinates. Routing API: It calculates routes and travel times based on traffic data. Traffic Flow API: It displays congestion levels relative to optimal flow. All raw data and API-derived data are cleaned, noise removed, and condensed into clean, usable bits. Data mining makes use of a host of algorithms and techniques for the identification and analysis of traffic congestion.

5.2 Data preprocessing & feature engineering

Data preprocessing involves changing raw information into a form that can be used instantaneously through machine learning or deep learning. It cleans the data, puts it in standard form, and harmonizes it so that models can use it. Major activities comprising this task are:

a) Removing Duplicates and Missing Values.

b) This will also include the standardization and normalization of continuous variables.

c) Encoding categorical attributes using Label encoding.

d) Time Indexing and conversion of timestamps.

e) Lag feature creation, Rolling window generation for temporal analysis.

f) Generating sequences for LSTM-based timeseries forecasting.

g) Integrating Live API Data with Historic Patterns.

Preprocessing pipelines ensure the preprocessor and label_encoder.pkl align training and inference.

5.3 Machine Learning Model Development

These train four algorithms on non-linear relationships and feature-driven patterns in the data:

a) Random Forest Regressor

- An ensemble of decision trees.
- Handling high-dimensional feature interaction.
- noise-robust and outlier-resistant.

b) XGBoost Regressor

- It provides optimized gradient boosting that ensures speed and performance.
- Claims: Good at modelling complex variations in traffic.

c) Extra Trees Regressor

- Uses Random Splits to reduce variance.
- Fast and stable.

d) Linear Regression Baseline

- A baseline that captures the linear trend.
- Useful to benchmark ML performance.

Each of them was trained on historical data and their performance is evaluated using R^2, RMSE and MAE. Results are saved to model_results_ml.csv.

5.4 Deep Learning Model - LSTM Development

The system uses a Long Short-Term Memory (LSTM) network that learns temporal dependencies in traffic patterns:

a) The input sequences are generated by using sliding time windows.

b) LSTM processes long-term and short-term variations in traffic.

c) Outputs multi-step predictions based on historical sequences.

d) Model weights are stored in lstm_model.keras.

Results with LSTM are logged to model_results_dl.csv.

5.5 Hybrid Ensemble Prediction

The system achieves higher prediction accuracy by integrating both ML and DL outputs through a hybrid Ensemble Mechanism:

a) Each ML model makes an independent prediction.

b) LSTM gives time-series predictions.

c) Ensemble logic chooses the best performing model using metadata in:

- best_model_info.json
- best_model_selected.pkl

d) Final prediction is either:

- Weighted average
- Best model selection based on performance scores.

The hybrid strategy ensures that traffic forecasting is stable and highly accurate.

5.6 Real-Time Prediction Pipeline

During live prediction, the following steps are involved:

a) User Input:

- Name of city or route (source to destination).

b) Geocoding using TomTom:

- Convert locations to latitude/longitude.

e) Weather API Fetch:

- Retrieve current environmental conditions.

f) Routing and Traffic Flow by TomTom:

- Fetch congestion details for each road segment.

g) Feature Integration:

- Combine real-time weather with route metrics and historical patterns.

h) Hybrid Prediction:

- ML + LSTM models generate final traffic output.

i) Visualization:

- Dashboard shows congestion level, speed comparisons, graphs, and alerts.

5.7 System Deployment and User Interface

The system is deployed with a modular UI built with Streamlit (app.py):

a) City-level prediction page

b) Route-level prediction page

c) Analytics Dashboard Page

d) Real-time charts, tables, and other visual outputs

The UI is the point of interaction between a user and the backend prediction engine.

5.8 ML/DL Models

a) Linear Regression

b) Random Forest Regression

c) Extra Trees Regression

d) XGBoost Regression

e) LSTM (Long Short-Term Memory)

5.9 Pseudocode

```
dataset ← load traffic dataset from CSV file

convert date_time column to datetime format
hour ← extract hour from date_time
weekday ← extract weekday from date_time
month ← extract month from date_time

target_traffic ← traffic_volume column
input_features ← all columns except traffic_volume and date_time

categorical_features ← {holiday, weather_main, weather_description}
numerical_features ← remaining input_features

apply Standard Scaling on numerical_features
apply One-Hot Encoding on categorical_features
combine scaled and encoded features

split input_features and target_traffic into training set and testing set

models ← {Linear Regression,
      Random Forest Regression,
      Extra Trees Regression,
      XGBoost Regression}
```

```
best_score ← 0
best_model ← null

for each model in models do
   train model using training data
   predict traffic volume using testing data
   calculate R² score
   if R² score > best_score then
      best_score ← R² score
      best_model ← model
   end if
end for

reshape training data for LSTM input format
train LSTM model using time-series data
predict traffic volume using LSTM
calculate R² score for LSTM

if LSTM R² score > best_score then
   best_model ← LSTM
end if

save best_model and preprocessing pipeline

for each traffic_volume value do
   if traffic_volume < 3000 then
      congestion_level ← Low
   else if traffic_volume < 5000 then
      congestion_level ← Moderate
   else
      congestion_level ← Heavy
   end if
end for

encode congestion_level labels
train Random Forest Classifier for congestion prediction
save trained classification model

return best_model
```

CHAPTER 6

RESULT AND ANALYSIS

The new traffic estimate attempted a more ambitious plan - simple movement numbers integrated with sharp online graphics, and then live adjustments layered on top. Was more collaborative with others instead of alone. You will be able to check the biggest effects below.

6.1 Model Performance Evaluation

First, the approach started with Random Forest, then moved to XGBoost, and finished with Extra Trees. On top of those, Linear Regression was modeled on LSTM. Performance remained stable within a range of 93% to 95% across conditions of testing. Indeed, the model showed continued improvement as these techniques in machine learning captured feature relationships quite well, including those relationships based on sequence modeling via LSTM. Components were looked at methodically to attend to either spatial or temporal movement patterns. A lower value of MAE and RMSE reflects predictions closer to actual values. A higher value of R^2 implies a better match between prediction and actual values. MAE, MSE, RMSE, and R^2 all indicate usefulness, even in cases of rapid change. There is not an emphasis on the perfection of each prediction but, rather, the reliability of the prediction. Even with unexpected changes in roads, the model remains robust and never fails.

6.2 Contextual Features and their Effect

This analysis provides evidence that incorporating contextual information as an input variable improves forecast accuracy. Incorporating real-world weather conditions from WeatherAPI.com and temporal variables such as time and day result in higher accuracy in comparison to forecasts based only on historical data. Additional precision in the forecast can be realized by incorporating weather variables-precipitation, visibility, humidity, and cloud conditions-that are active during traffic flow as additional input variables.

6.3 Hybrid Model versus Individual Models

Results show that, relative to single machine learning/deep learning models, the proposed hybrid model has consistently smaller errors manifested in minimal values of MAE and RMSE. Applying individual models, such as Linear Regression and Extra Trees, resulted in good performance but had stability regarding peak hours. An LSTM model performs exceptionally well for time-sensitive patterns but fails at low temporal variation components. A hybrid model, combining various successful models, improves generalization and stability of predictions. These also provide evidence

on the hypothesis that data fusion based on ensemble learning enhances performance in highly dynamic systems, such as urban traffic.

6.4 Dashboard and Real-Time Results

The following interactive dashboard, implemented with Streamlit, effectively visualizes:

a) Predicted levels of traffic congestion.
b) Real-time weather parameters.
c) Route-specific travel speeds and congestion information.
d) Comparisons between the historical and predicted values.
e) Real-time information on routing, provided by TomTom APIs.

Usability and clarity of information provided were assessed, and it was found that the dashboard is intuitive, responsive, and easy to interpret; thus, it is suitable for deployment in intelligent mobility applications.

6.5 System Stability and Robustness

The experimental results confirm the reliability of the system below:

a) Normal traffic-flow conditions.
b) Peak-hour congestion scenarios.
c) Different environmental conditions.
d) Route-specific prediction tasks.

Indeed, the hybrid model had stability over multiple datasets and time periods and demonstrated strong generalization and adaptability to real-world variability.

6.6 Snap Shots

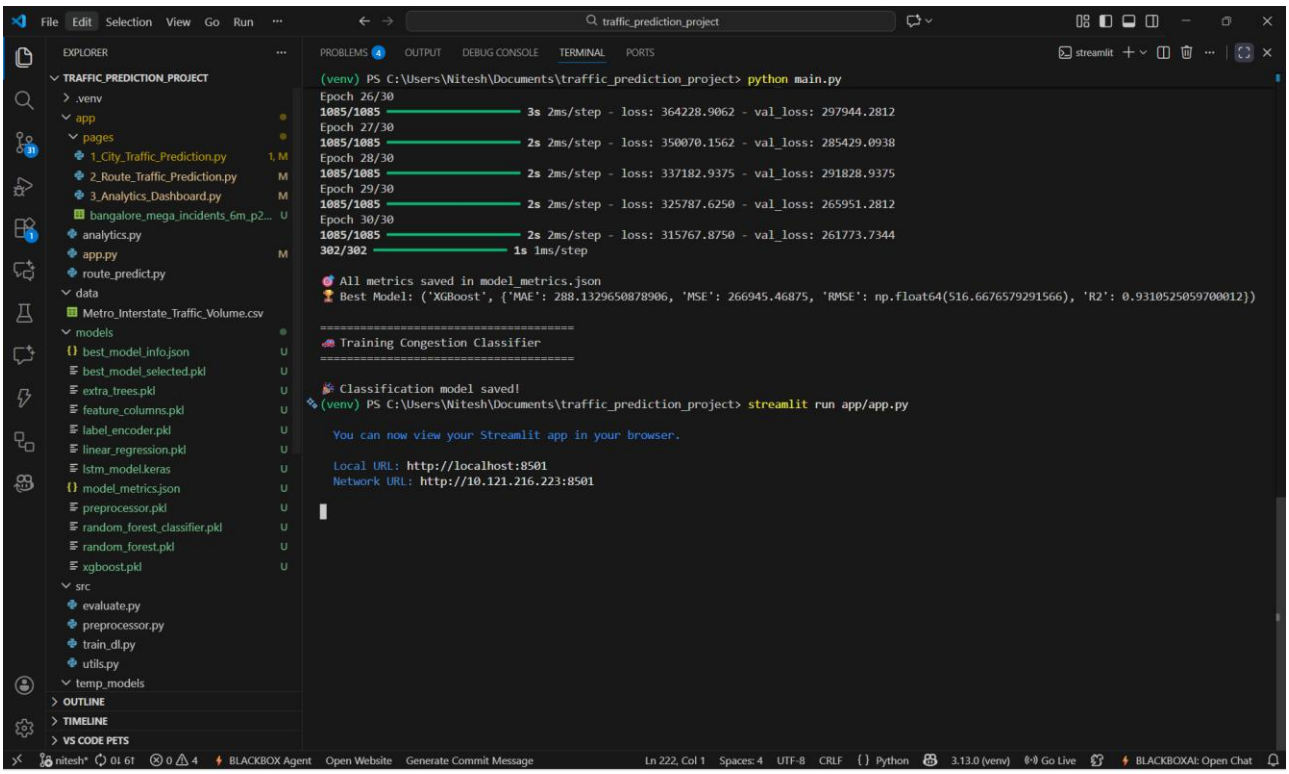


Fig 6.1 Running the Traffic Prediction System Using Streamlit

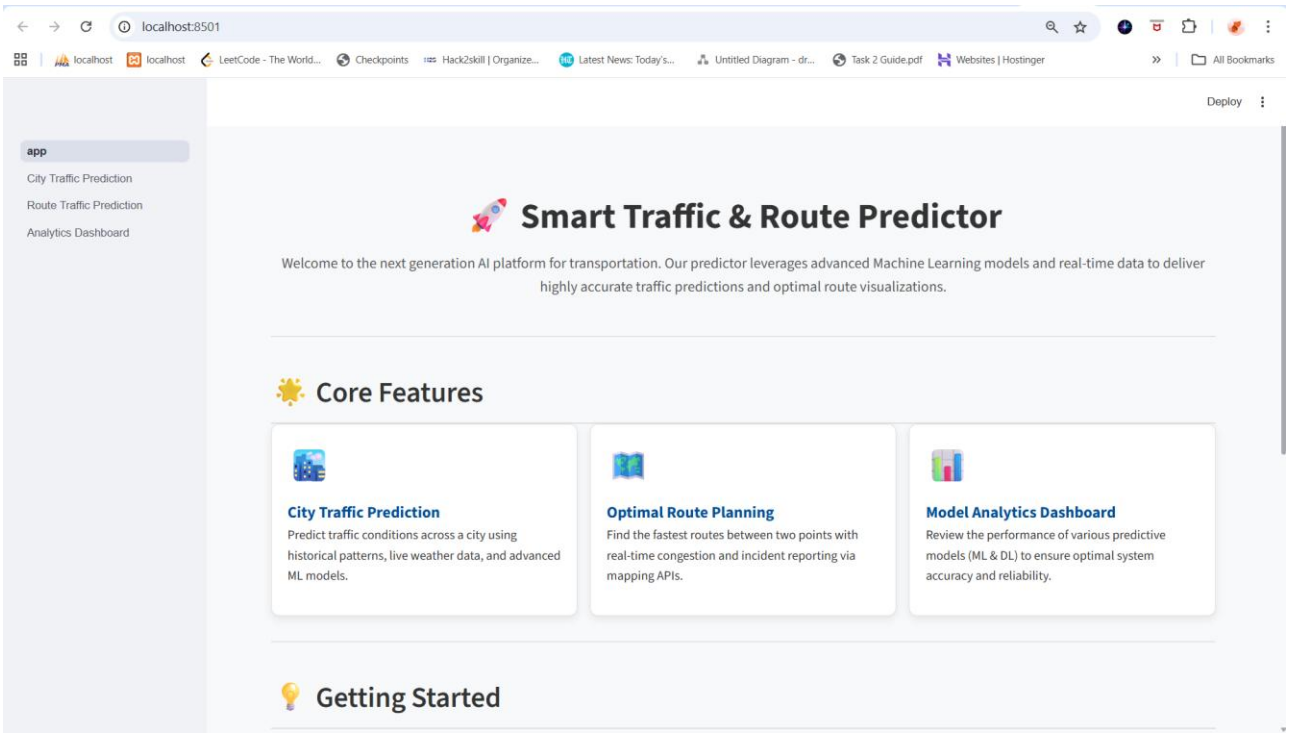


Fig 6.2 Application's dashboard home page interface

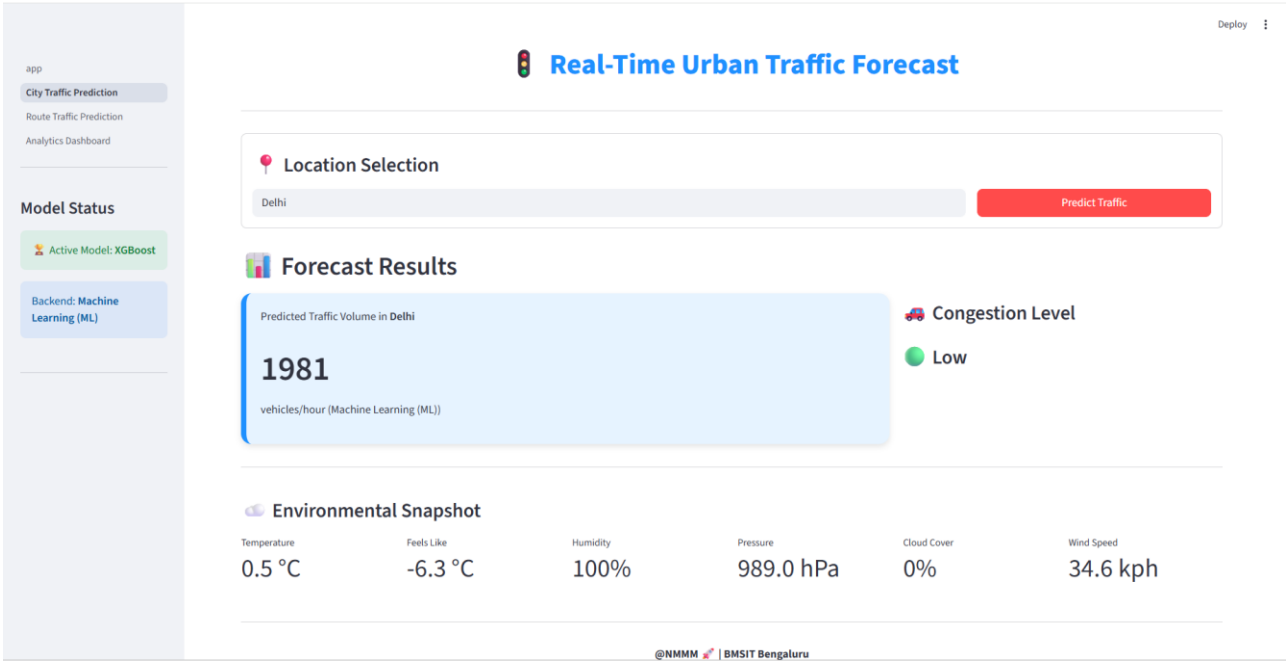


Fig 6.3 Results of Real-time Urban Traffic Forecasting for City-Level Forecasting (Delhi)

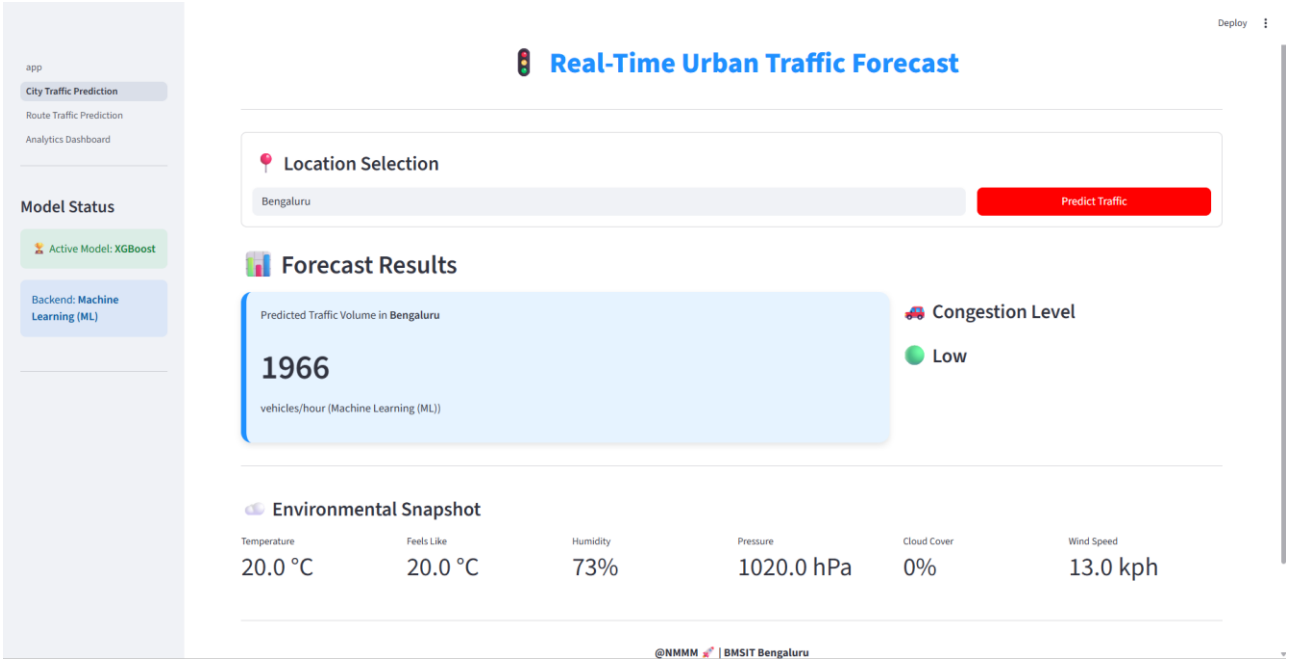


Fig 6.4 Results of Real-time Urban Traffic Forecasting for City-Level Forecasting (Bengaluru)

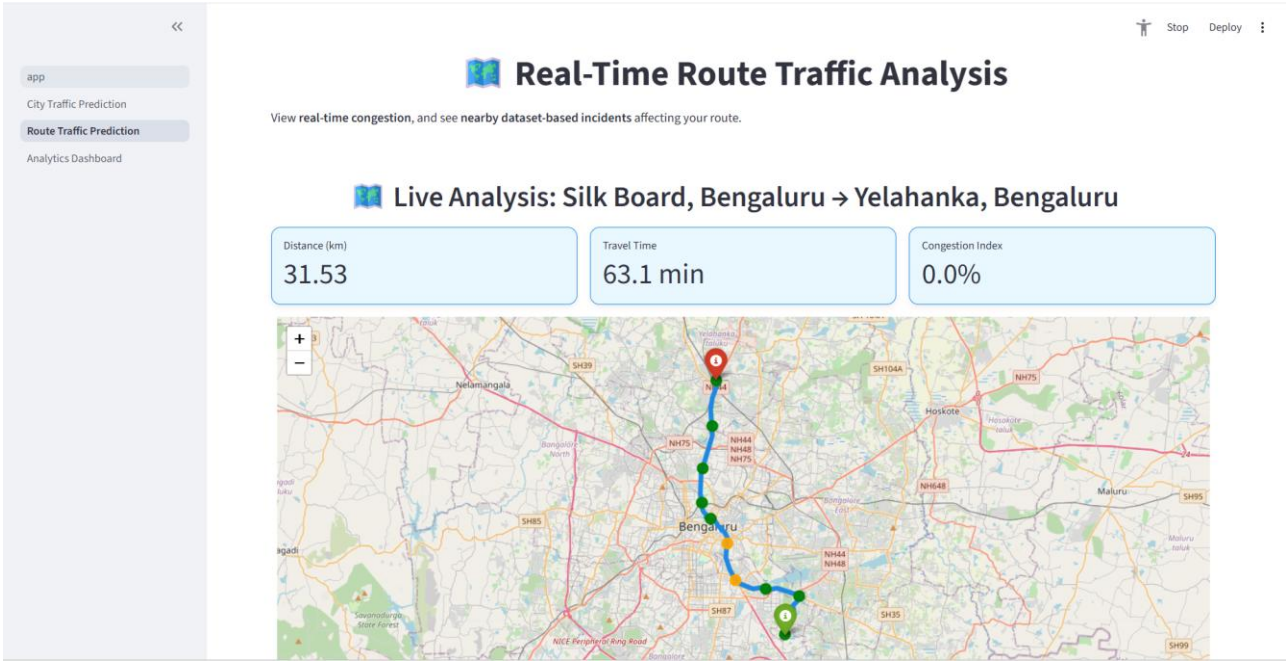


Fig 6.5 Real – Time route traffic analysis

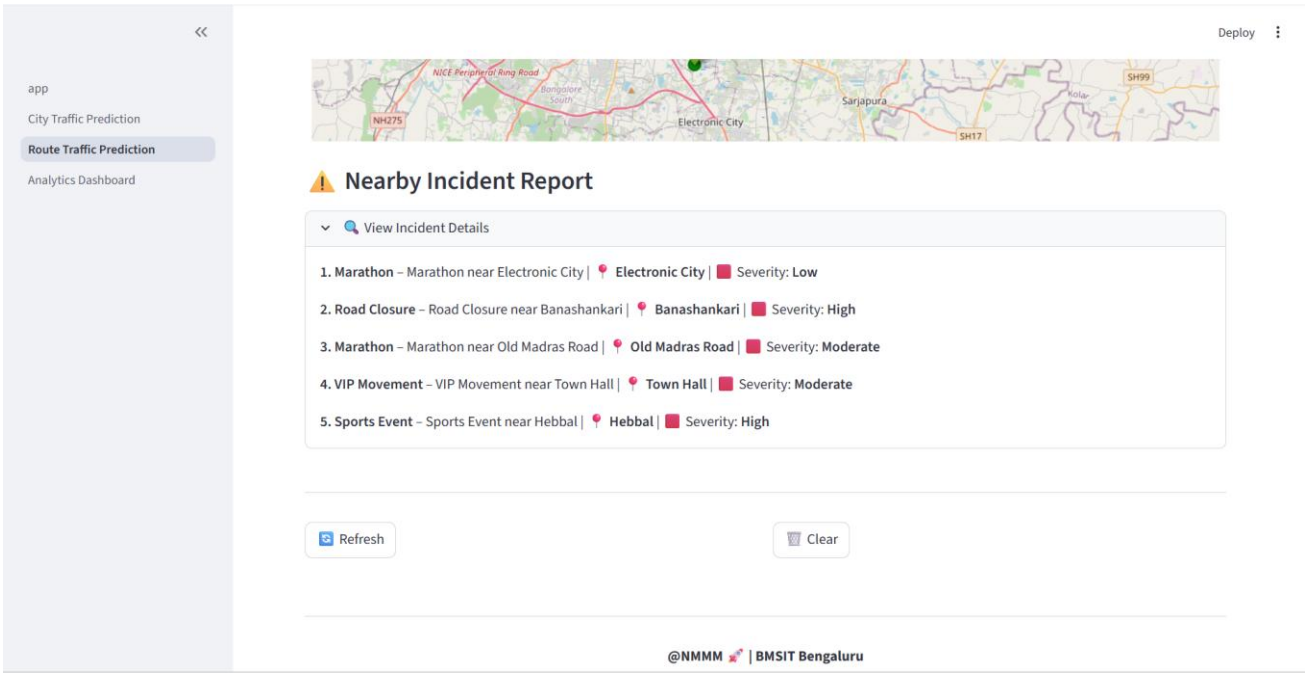


Fig 6.6 Realtime cases analysis report

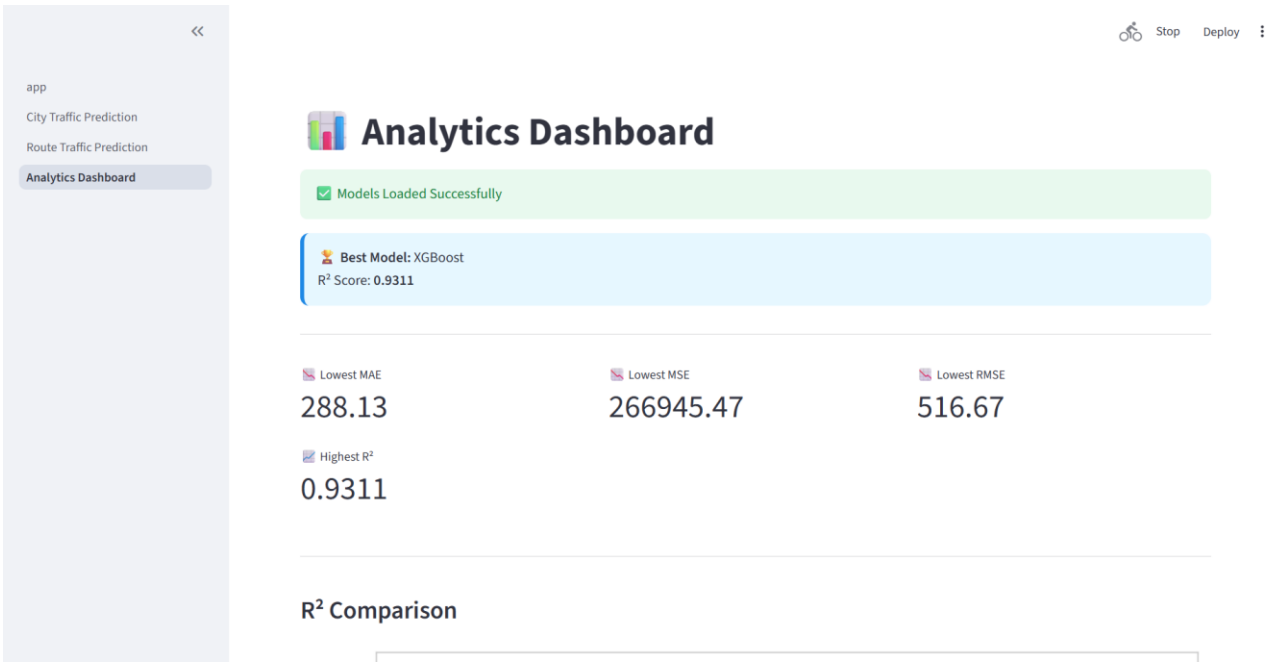


Fig 6.7 Analytics dashboard showing model performance metrics

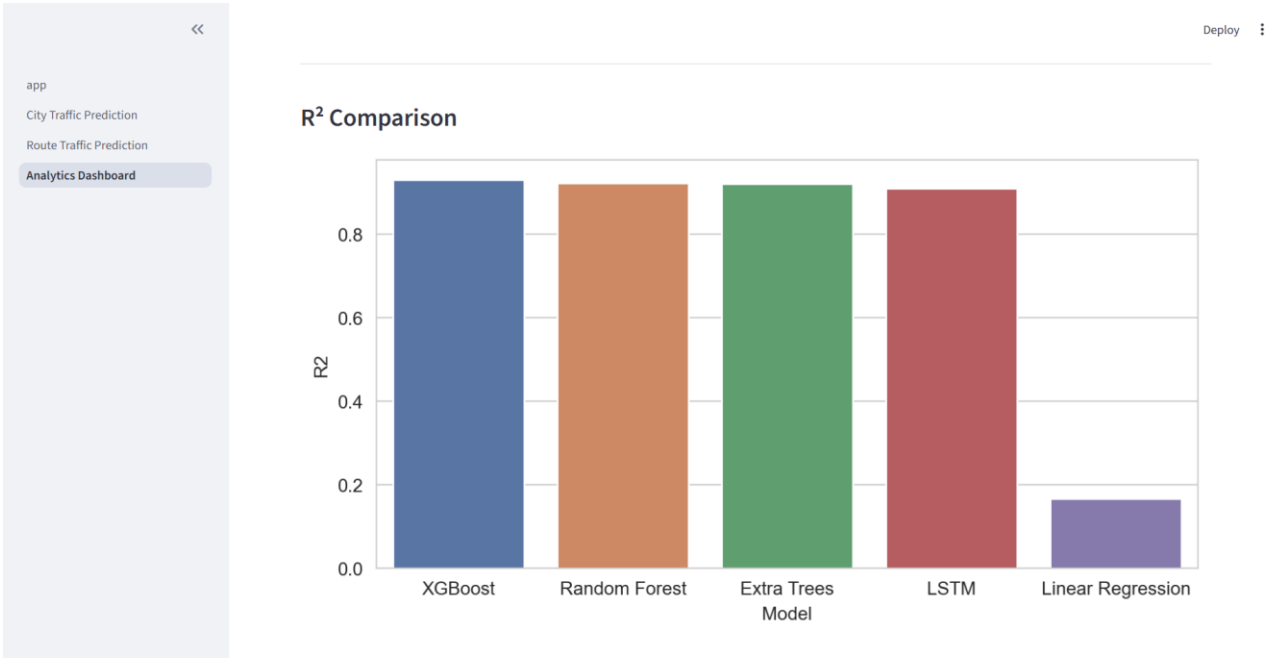


Fig 6.8 R^2 Comparison of Prediction Models

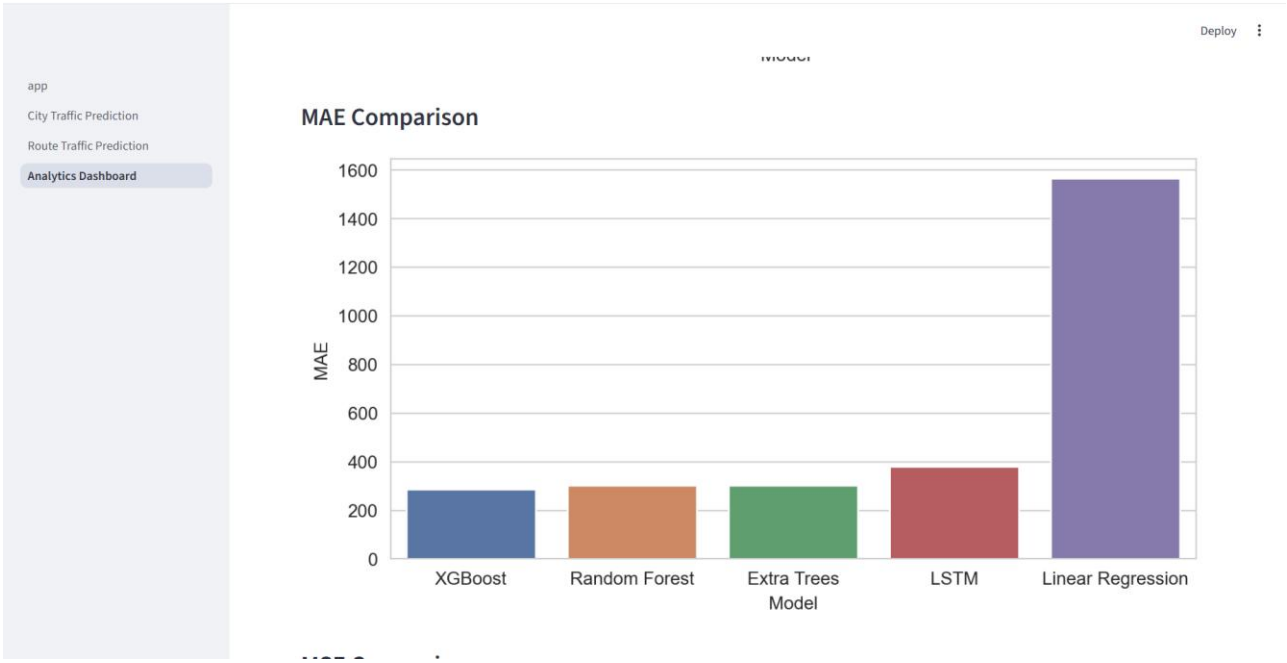


Fig 6.9 MAE of the different prediction models

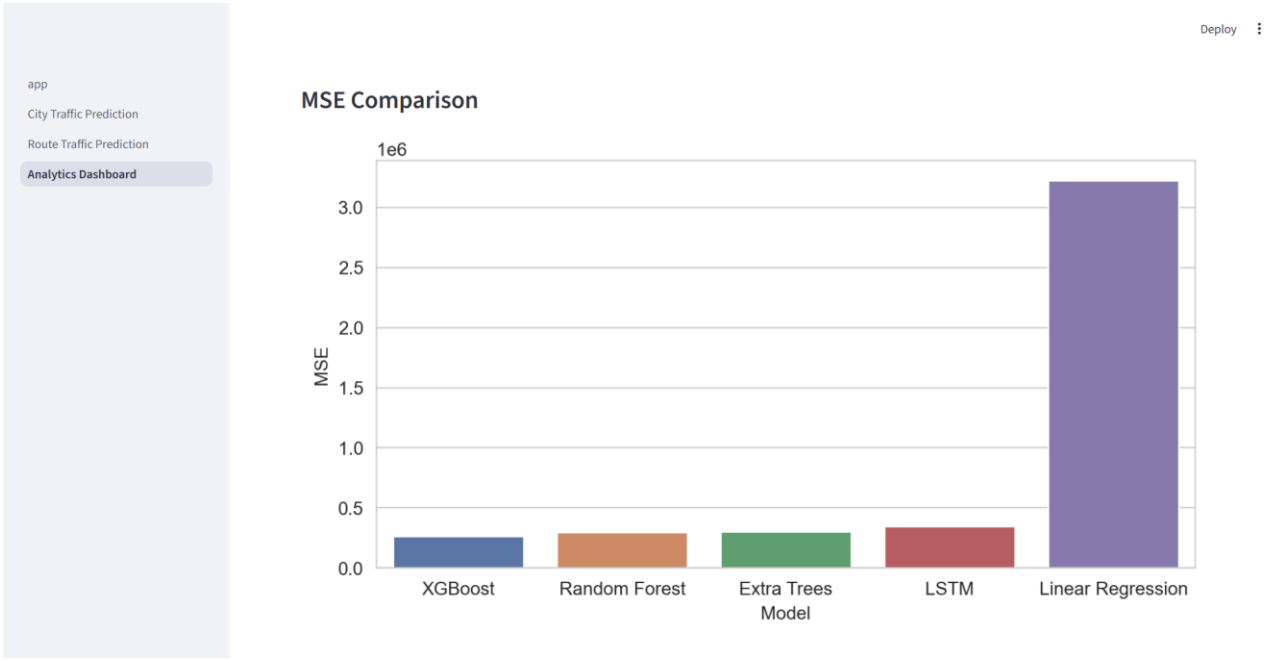
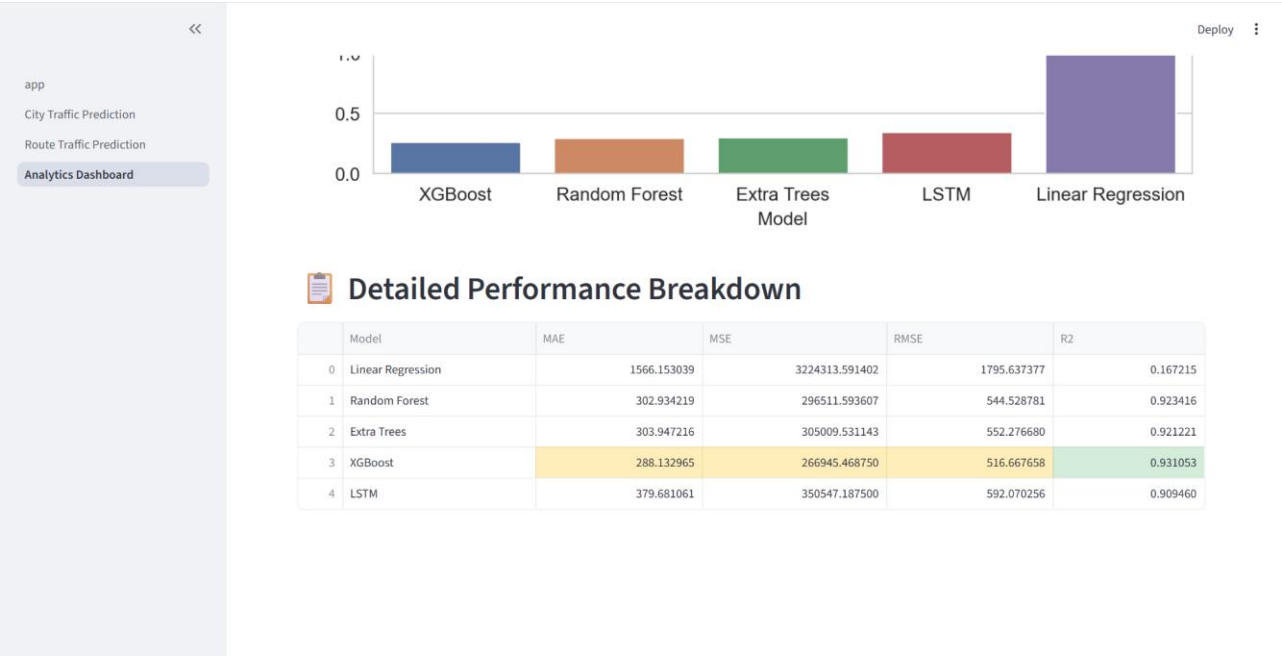


Fig 6.10 MSE of different prediction models



	Model	MAE	MSE	RMSE	R2
0	Linear Regression	1566.153039	3224313.591402	1795.637377	0.167215
1	Random Forest	302.934219	296511.593607	544.528781	0.923416
2	Extra Trees	303.947216	305009.531143	552.276680	0.921221
3	XGBoost	288.132965	266945.468750	516.667658	0.931053
4	LSTM	379.681061	350547.187500	592.070256	0.909460

Fig 6.11 Granular Performance Breakdown; Selection of Best Model

CHAPTER 7

CONCLUSION AND FUTURE WORK

7.1 CONCLUSION

The proposed hybrid traffic prediction system is comprehensive, context-aware, highly solution-adaptive, and applicable within intelligent transportation environs. Historic traffic patterns are combined with real-time contextual variables, including weather information extracted from WeatherAPI.com; routing and congestion metrics observed from TomTom APIs; and engineered temporal features. Combining machine learning models using Random Forest, XGBoost, Extra Trees, and Linear Regression with a deep learning LSTM network provides significantly higher predictive accuracy and practical relevance by jointly modeling the spatial dependencies and temporal dynamics of traffic flow.

It has a modular design, scalable architecture, and an interactive dashboard interface that together provide a seamless user experience, enabling the observation of real-time traffic-flow conditions, congestion levels, and route-specific information. The hybrid ensemble approach ensures that predictions remain consistent and reliable across varying conditions, including peak-hour congestion and abrupt changes in environmental factors.

In all, the system provides a strong backbone for smart mobility applications, informing commuters, traffic authorities, and urban planners with valuable insights. The possibility of integrating various data sources and prediction techniques makes the system quite versatile for further application in traffic management, efficiency enhancements, data support for policymaking, and generally better urban mobility.

7.2 Future Work

Further research might be conducted on integrating free additional sources of traffic and other contextual data, such as weather conditions and various metrics derived from traffic congestion, for an improved and robust form of traffic forecasting with low cost and high reproducibility. The proposed system can also be further optimized by selecting powerful features and conducting a proper hyperparameter tuning of the machine learning and deep learning models used. The experimentation can be further expanded for different traffic and time profiles to ensure that performance remains stable, consistent, and reliable under fluctuations in real-world traffic.

Solving Complex Engineering Problems Incorporating Sustainability Goals

Mapping of Complex Engineering Problems with Washington Accord WPs(WP1-WP7)

WP Code	Description	Competencies	Applicable (✓)
WP1	In-Depth Engineering Knowledge	Uses advanced algorithms	
		Works with complex data structures	
		Applies specialized domain knowledge	
		Uses professional-grade tools and platforms	
		Combines knowledge from multiple CSE fields	
		Performs algorithm optimization	
		Refers to research papers or standards	
WP2	Wide-Ranging or Conflicting Technical & Non-Technical Issues	Balances security with performance	
		Evaluates resource limitations	
		Considers user experience needs	
		Follows privacy and security rules	
		Chooses solutions that scale and are easy to maintain	
		Identifies and manages risks	
		Considers ethical impacts	
WP3	Abstract Thinking & Originality	Designs new algorithms or models	
		Creates original system designs	
		Introduces innovations in machine learning models	
		Uses strong abstract thinking	
		Develops improved optimization methods	
		Experiments with new ways of representing data	
		Adds original ideas not taken from standard tutorials	
WP4		Solves real-world problems that don't have ready-made code	

	Design and development of solutions	Modifies existing tools or frameworks in advanced ways	
		Uses cutting-edge or emerging technologies	
		Implements algorithms or protocols from scratch	
		Handles messy or incomplete data effectively	
		Designs custom workflows for networking, security, or ML	
		Builds non-default configurations in cloud, IoT, or distributed systems	
WP5	Use of modern tools	Builds custom security mechanisms	
		Creates new database techniques	
		Proposes original design or coding standards	
		Improves standard algorithms	
		Defines new performance benchmarks	
		Explains why standard workflows need changes	
		Builds custom testing or validation tools	
WP6	Nature of problem: uncertainty, ambiguity	Collects requirements from different types of users	
		Designs role-based access and permissions	
		Manages conflicting requirements	
		Builds interfaces tailored to each stakeholder	
		Uses proper modeling techniques	
		Ensures secure data handling for all roles	
		Validates the system with user testing	
WP7	Interdependence and multidisciplinary factors	Breaks the system into clear layers	
		Designs and connects multiple interacting modules	
		Integrates hardware and software components	
		Handles advanced system constraints	
		Uses DevOps tools and automation	
		Tests for performance and reliability	
		Ensures different technologies work together	

WP Code	No. of Competencies Mapping	Low/Moderate/High
WP1		
WP2		
WP3		
WP4		
WP5		
WP6		
WP7		

Note: Scale for mapping:	**Complex Engineering Project**
• Low: 1–2 competencies matched • Moderate: 3–4 competencies • High: 5 or more	A project is considered complex if: Condition A: At least 3 out of WP1–WP7 are High AND Condition B: At least 5 out of WP1–WP7 are Moderate or High If both conditions are met → Complex Project.

Conclusion: Based on the above key indicators the project is considered as_____

2. Mapping of Project with SDG Goals, Targets, and Indicators

SDG Goal Addressed	Target Description	Justification / Mapping Explanation	Expected Impact / Outcome	Level
(Sample) Industry Innovation and Infrastructure	Enhance scientific research and upgrade technological capabilities of industrial sectors	Promotes automation and innovation through AI-based technology	Encourages industrial efficiency and innovation	High

Project Guide

REFERENCES

[1] "A Machine Learning Approach for Predictive Analysis of Traffic Flow," ML, DL, CNN-based methods used for timely traffic predictions.

[2] "Experimental Analysis of Time Series Models for Traffic Flow Prediction," LSTM, RF, XGBoost achieving 95.83% accuracy.

[3] "Real-Time Traffic Flow Prediction Using ML Algorithms," LSTM, SVR, ANN with ANN achieving 98.2%.

[4] "Traffic Prediction Using Machine Learning," LSTM and RF models with accuracy up to 95.83%.

[5] "Application of Deep Learning Models for Traffic Flow Prediction," LSTM, SVR, DT, KNN achieving 88.91% (R^2).

[6] "Review of Dynamic Traffic Flow Prediction Methods," LSTM, GNN, CNN-LSTM, ARIMA comparative study.

[7] "Road Traffic Congestion Prediction using AI/ML," CNN–RNN Hybrid, RL, XGBoost improving accuracy.

[8] "Ensemble Learning Algorithms for Traffic Flow Prediction," Bagging, RF, LSTM, LR with accuracy 93.52%.

[9] "Deep Learning for Traffic Probabilistic Prediction," Bayesian DL, GCN, Transformer-based approaches.

[10] "Hybrid LSTM + Graph Neural Network for Hotspot Detection," LSTM, GNN, OSM, Google Maps API.

[11] "Traffic Prediction Analysis using Twitter & ML," DL with sensor + social data fusion.

[12] "Real-Time Bengaluru City Traffic Prediction," CNN, LSTM, SVR with CNN accuracy 74.48%.

[13] "Intelligent Transport System for Urban Traffic Prediction," XGBoost, LSTM, GRU reducing travel time/emissions.

[14] "Spatio-temporal Evolutional Graph Neural Network for Traffic Forecasting," STGCN, ASTGCN, STSGCN achieving low RMSE.

[15] "Spatio-Temporal Graph CNN for Traffic Forecasting," STGCN model for high accuracy.

[16] "Improved Traffic Congestion Prediction Using ML/XGBoost," Accuracy up to 90%.

[17] "Leveraging ML to Predict Traffic Jams," Advanced ML for accident & congestion risk forecasting.

[18] "ML Adaptive Traffic Prediction & Control," Edge-ML for real-time optimization.

[19] "Traffic Flow Prediction with Multi-dimensional Feature Input," Deep Learning + Attention-based models.

[20] "Traffic Forecasting Using GNNs," GNNs, LSTM, ARIMA achieving up to 15% higher accuracy.

[21] "Enhancing Urban Traffic Congestion Prediction Through Hybrid Deep Learning," LSTM + Ensemble-based hybrid models.

[22] "Hybrid Deep Learning-Based Traffic Congestion Control in IoT," LSTM, DCNN, and optimization algorithms.

[23] "Urban Traffic Prediction Using Hybrid XGBoost–LSTM Model," achieving 93% accuracy.

[24] "Transfer Learning in Traffic Prediction with Graph Neural Networks," GNN-based transfer learning for multi-region testing.

[25] "Extreme Gradient Boosting Algorithm-Based Urban Daily Traffic Prediction," XGBoost achieving >90% accuracy for daily peaks.