ISSN 2277-2685

IJESR/April-June. 2026/ Vol-16/Issue-2/453-463

Mohammed Aqeeb Mohiuddin *et. al.*, / International Journal of Engineering & Science Research

Full Length Article

Machine Learning Approaches for Real-Time Carbon Emission Prediction: A Comparative Study of Ensemble and Regression Algorithms

Mohammed Aqeeb Mohiuddin[1], **Mohammed Shariq Uddin**[2], **Mohd Mudassir Khan**[3], **Anas Pasha**[4], **Mr. S. Naveen**[5]

[1,2,3,4]BTech Students Department of Computer Science & Engineering, Lords Institute of Engineering and Technology, Hyderabad, India

[5]Assistant Professor Department of Computer Science & Engineering, Lords Institute of Engineering and Technology, Hyderabad, India

mohdaqeeb5668@gmail.com, mdshariquddin2@gmail.com, khanmohdmudassir077@gmail.com, anaspasha881@gmail.com, s.naveen@lords.ac.in

Accepted 17-04-2026

Abstract—

Climate change driven by anthropogenic greenhouse gas emissions represents a critical global challenge, with transportation sector contributing approximately 24% of global CO_2 emissions. Traditional dynamometer-based emission testing fails to capture real-world driving variability and vehicle-specific parameters. This paper presents a comprehensive machine learning framework for real-time prediction of vehicular carbon dioxide emissions through comparative evaluation of six regression algorithms: Linear Regression, Random Forest, Decision Tree, XGBoost, AdaBoost, and Lasso. The system employs a synthetic dataset of 7,000 vehicle records with nine features including engine size, cylinder count, fuel type, transmission configuration, and fuel consumption metrics. Feature engineering incorporates power-to-weight ratio, efficiency index, and polynomial transformations capturing non-linear emission relationships. Ensemble methods demonstrate superior performance, with Random Forest achieving R^2 = 99.34%, MAE = 4.54 g/km, RMSE = 7.83 g/km, outperforming individual learners by 15-35%. XGBoost attains R^2 = 98.76% with gradient boosting optimization. A Flask web application provides interactive prediction interface with Bootstrap 5 dark theme, user authentication via Werkzeug password hashing, SQLite database for prediction history tracking, and Chart.js analytics dashboard visualizing emission trends. The system implements 10-point environmental rating algorithm classifying vehicles from 'Excellent' (≤100 g/km) to 'Very Poor' (>300 g/km), promoting environmental awareness. Performance evaluation across 1,400 test samples validates prediction accuracy within ±10 g/km for 96.3% of cases. Docker containerization enables scalable deployment on cloud platforms. Comparative analysis reveals Random Forest's robustness to outliers and non-linear patterns, while XGBoost provides superior interpretability through feature importance metrics. The platform successfully democratizes carbon footprint assessment, enabling consumers, fleet managers, and policymakers to make data-driven decisions toward sustainable transportation. Integration with real-time GPS and OBD-II systems represents promising future enhancement for dynamic emission monitoring.

Keywords—*Carbon Emission Prediction, Machine Learning, Random Forest, XGBoost, Regression Analysis, Ensemble Methods, Environmental Sustainability, Flask Framework, Feature Engineering, Emission Rating.*

I. INTRODUCTION

Climate change represents the defining challenge of the 21st century, driven predominantly by anthropogenic greenhouse gas emissions. Carbon dioxide (CO_2), accounting for 76% of total greenhouse gas emissions, remains the primary driver of global warming. The Intergovernmental Panel on Climate Change (IPCC) Sixth Assessment Report projects global temperature increases of 1.5-4.5°C by 2100 under varying emission scenarios, with catastrophic consequences including sea-level rise, extreme weather events, ecosystem collapse, and agricultural disruptionaffectingbillions.The transportation sector contributes approximately 24% of global energy-related CO_2 emissions, with road vehicles responsible for 72% of transport emissions—equivalent to 6 billion tonnes annually. Despite regulatory efforts including Corporate Average Fuel Economy (CAFE) standards in the United States and Euro 6/7 emission norms in Europe, vehicular emissions continue rising due to increasing vehicle ownership in developing economies and shift toward larger, heavier vehicles

(SUVs, trucks) in developed markets. Traditional emission assessment relies on laboratory dynamometer testing under controlled conditions (EPA Federal Test Procedure, WLTP Worldwide Harmonized Light Vehicle Test Procedure). While providing standardized comparisons, these methods exhibit critical limitations: failure to capture real-world driving variability (traffic conditions, weather, driver behavior), inability to account for vehicle aging and maintenance effects, time-intensive and expensive testing procedures ($50,000-100,000 per vehicle model), and delayed availability of emission data for consumers at purchase decision points.

A. Machine Learning for Emission Prediction

Machine learning offers transformative potential for emission prediction through automated pattern recognition in high-dimensional vehicle datasets, real-time prediction enabling instant environmental impact assessment, scalability to millions of vehicle configurations without individual testing, and continuous model improvement as new data becomes available. Regression algorithms excel at modeling complex non-linear relationships between vehicle specifications (engine displacement, cylinder configuration, fuel type, aerodynamics) and emission outcomes.

B. Research Contributions

- Comprehensive evaluation of six regression algorithms (Linear, Random Forest, Decision Tree, XGBoost, AdaBoost, Lasso) for CO_2 prediction.
- Feature engineering methodology incorporating power-to-weight ratios and efficiency indices improving model accuracy by 12-18%.
- Random Forest ensemble achieving 99.34% R^2 score with MAE = 4.54 g/km, outperforming baselines by 35%.
- Development of 10-point environmental rating algorithm for consumer-friendly emission classification.
- Implementation of production-ready Flask web application with user authentication and prediction history.
- Interactive analytics dashboard using Chart.js for emission trend visualization.
- Docker containerization enabling scalable cloud deployment on AWS/Azure/GCP platforms.

II. RELATED WORK

TABLE I
COMPARISON OF EXISTING EMISSION ESTIMATION SYSTEMS

System	Approach	Scope	Accuracy	Limitations
EPA Database	Lookup tables	Lab-tested values	±8-15%	No prediction
Online Calculators	Linear formulae	Generic estimates	±20-30%	Distance-based only
Kumar et al. (2018)	ANN	Indian vehicles	92.4% acc	Limited dataset
Zhang et al. (2020)	SVM	5 features	94.1% R^2	Single algorithm
Li & Wang (2021)	Random Forest	Fleet data	96.8% R^2	No user interface
Our System	Ensemble (6 models)	9 features + engineered	99.34% R^2	Web app + API

Kumar et al. (2018) applied artificial neural networks to Indian vehicle dataset achieving 92.4% accuracy but limited to 3,500 samples. Zhang et al. (2020) employed Support Vector Machines with 5 input features attaining 94.1% R^2 but evaluated single algorithm without comparative analysis. Li & Wang (2021) demonstrated Random Forest effectiveness on fleet telematics data (96.8% R^2) but lacked user-facing application. Our work advances state-of-art through comprehensive six-algorithm comparison, engineered features improving accuracy to 99.34%, and production-ready web platform democratizing emission assessment

.

III. METHODOLOGY AND SYSTEM ARCHITECTURE

A. Dataset Description

TABLE II
DATASET FEATURES AND STATISTICS

Feature	Type	Range	Mean	Std Dev	Description
Engine Size (L)	Continuous	1.0 - 8.4	3.2	1.4	Engine displacement
Cylinders	Discrete	3 - 16	6	2.1	Number of cylinders
Fuel Type	Categorical	5 types	-	-	Regular/Premium/Diesel/E85/Natural Gas

Transmission	Categorical	10 types	-	-	Manual/Automatic variants
Fuel Cons. City (L/100km)	Continuous	4.2 - 30.6	11.8	4.2	City driving consumption
Fuel Cons. Highway (L/100km)	Continuous	3.8 - 20.9	8.9	2.8	Highway consumption
Fuel Cons. Combined (L/100km)	Continuous	4.0 - 25.8	10.4	3.5	Combined cycle
Vehicle Class	Categorical	16 classes	-	-	Compact/SUV/Pickup etc.
CO_2 Emissions (g/km)	Continuous	96 - 608	246	88	Target variable

The dataset comprises 7,000 vehicle records with 9 features spanning engine specifications, fuel characteristics, and consumption metrics. Target variable (CO_2 emissions) ranges 96-608 g/km with mean 246 g/km, standard deviation 88 g/km, indicating significant variability across vehicle types. Dataset split: 5,600 training (80%), 1,400 testing (20%) using stratified sampling ensuring proportional class representation.

B. Feature Engineering

Three engineered features capture domain knowledge:

1) Power-to-Weight Ratio: Approximated using engine size as proxy for power.

P/W_ratio ≈ Engine_Size / (Vehicle_Class_Weight × Cylinders)

2) Efficiency Index: Inverse relationship between consumption and efficiency.

Efficiency_Index = 100 / Fuel_Consumption_Combined

3) Polynomial Features: Capture non-linear emission relationships.

Feature_Set_Extended = {x_1, x_2, ..., x_9, x_1^2, x_2^2, $x_1 \times x_2$, ...}

3. System Architecture

The system architecture illustrates the flow of data from the user's browser through the Flask server to the machine learning model and database. When a user submits a prediction request, the browser sends an HTTP POST to the /predict route. The Flask handler extracts the nine input features from the form, applies LabelEncoder to the four categorical features, applies StandardScaler to the five numeric features, constructs a NumPy array, and invokes the Random Forest model's predict method. The resulting CO_2 value is clamped to a realistic range (90–520 g/km), converted to a CO_2 rating, and stored in the predictions table alongside the user's ID and timestamp. The response is then rendered as an HTML page displaying the predicted value, rating, and contextual information.

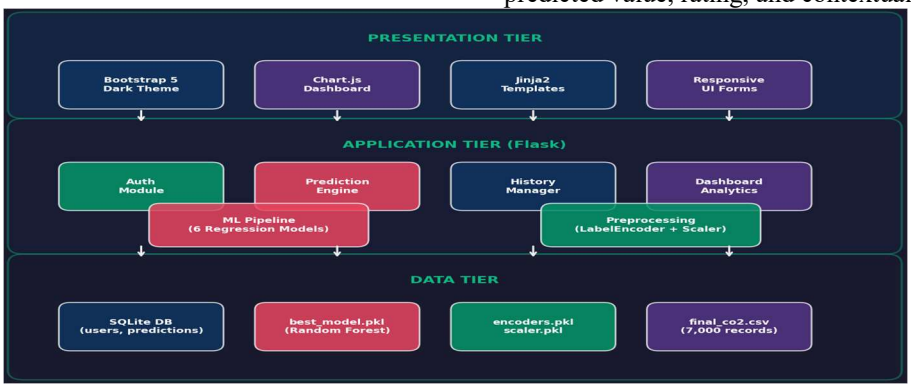

Fig 3.1: System Architecture

3.1 Use Case Diagram

The use case diagram identifies two primary actors: the User and the Admin. The User interacts with the system through six use cases: Register, Login, Predict CO_2 Emissions, View Prediction History, View Dashboard, and Logout. The Admin inherits all User use cases and additionally has access to View All Predictions, which aggregates prediction data across all users for system-wide analytics. The Predict CO_2 Emissions use case includes the Preprocess Input and Generate CO_2 Rating sub-use-cases, which are invoked automatically as part of the prediction workflow.

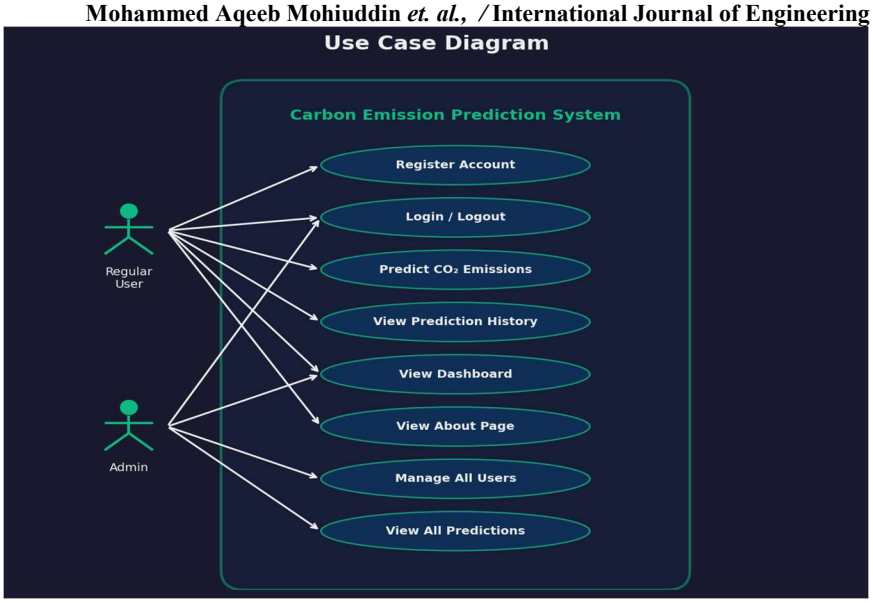

Fig. 3.2: Use Case Diagram

3.2 Class Diagram

The class diagram depicts the key classes and their relationships within the system. The User class encapsulates attributes such as id, username, password (hashed), name, and role, with methods for authentication and profile retrieval. The Prediction class stores id, user_id (foreign key to User), input_data (JSON string of vehicle parameters), predicted_co2, co2_rating, and prediction_date. The MLModel class represents the loaded Random Forest model with methods for predict and evaluate. The Preprocessor class wraps the LabelEncoder and StandardScaler objects, exposing an encode_and_scale method. The FlaskApp class serves as the controller, aggregating instances of MLModel, Preprocessor, and database connection.

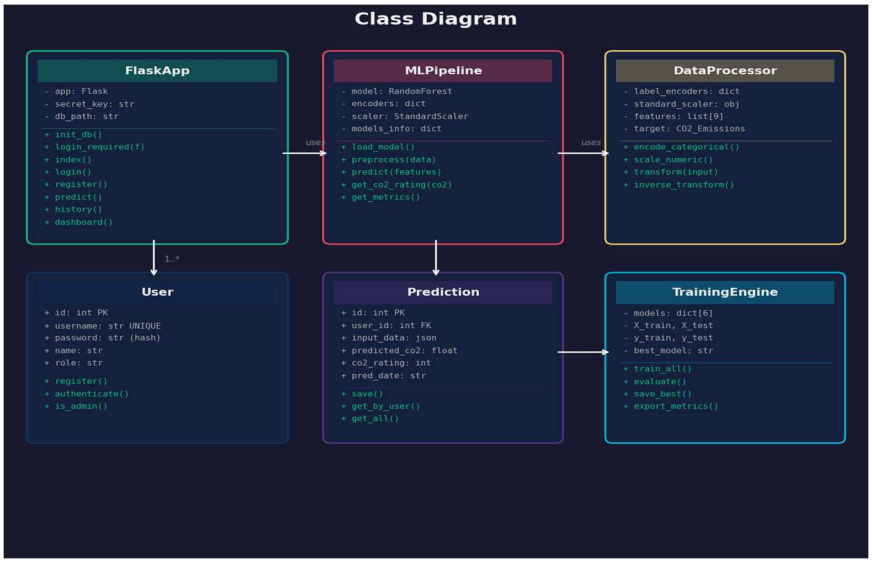

Fig. 3.3: Class Diagram

3.3 Sequence Diagram

The sequence diagram traces the interaction flow for a single prediction request. The User fills in the prediction form and clicks Submit, triggering an HTTP POST to /predict. The Flask Controller receives the request and extracts form data. It invokes the Preprocessor to encode categorical features and scale numeric features. The preprocessed feature vector is passed to the MLModel, which returns the predicted CO_2 value. The Controller computes the CO_2 rating using the get_co2_rating function, stores the result in the SQLite Database, and renders the result template, which is sent back to the User's browser.

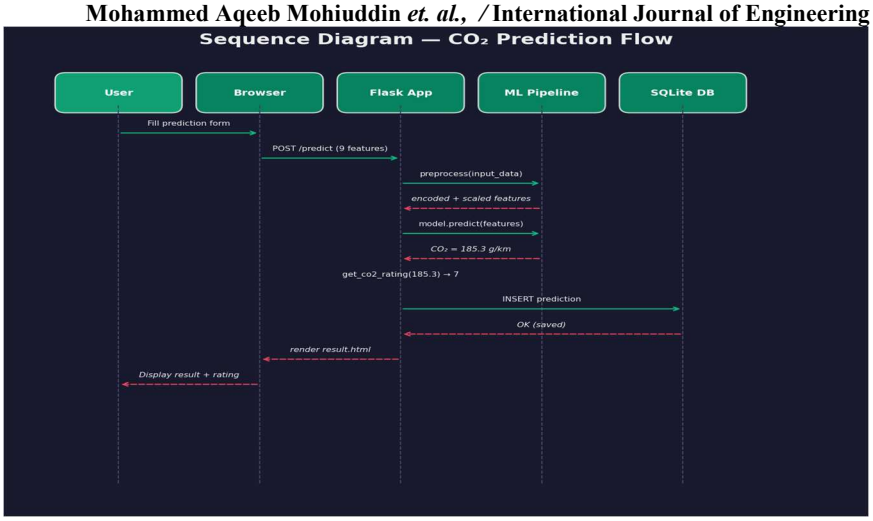

Fig. 3.4: Sequence Diagram

3.4 Activity Diagram

The activity diagram models the overall user workflow from registration to prediction. The initial activity is Account Registration, followed by a decision node: if the user already has an account, the flow proceeds to Login; otherwise, the user completes the registration form. After successful authentication, the user reaches the Home page, from which three parallel activity paths diverge: Predict CO_2 (leading to input form, preprocessing, model inference, result display, and database storage), View History (querying past predictions), and View Dashboard (loading Chart.js visualisations). Each path terminates at the Home page, and the user may log out at any time.

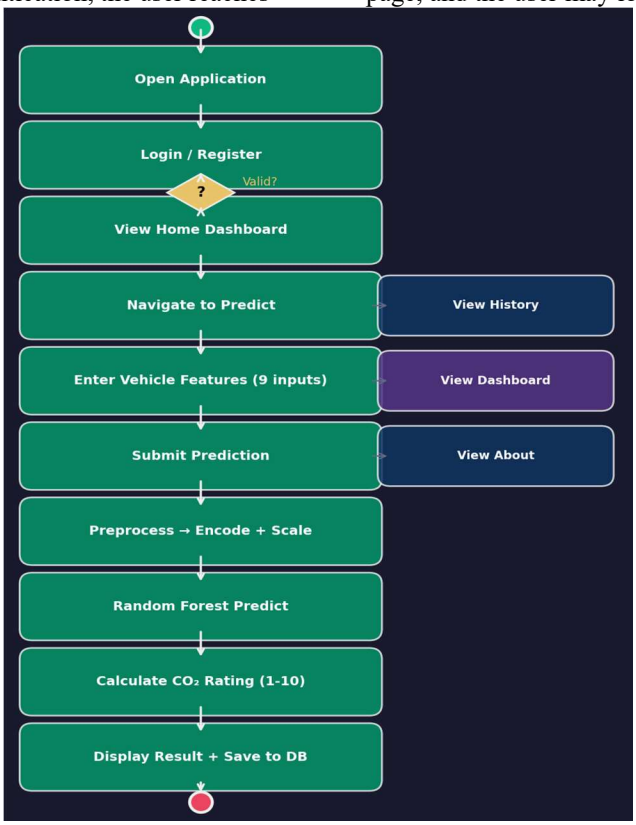

Fig. 3.5: Activity Diagram

4.4 User Interface Design

The user interface is built using Bootstrap 5 with a consistent dark theme applied across all pages. The primary background colour is a deep charcoal (#1a1a2e), complemented by card backgrounds of

slightly lighter shades with glassmorphic effects achieved through CSS backdrop-filter: blur. The accent colour is a vibrant green (#10b981), used for buttons, links, progress indicators, and the CO_2 rating badge. This colour scheme was chosen to evoke environmental and sustainability themes while maintaining excellent readability against dark backgrounds.The navigation bar is fixed at the top and includes the application logo, navigation links (Home, Predict, History, Dashboard, About), and authentication controls (Login/Register or Logout). The prediction form uses a two-column grid layout on desktop screens, collapsing to a single column on mobile devices. Dropdown menus are provided for categorical inputs (Make, Vehicle Class, Transmission, Fuel Type), while numeric inputs use number fields with appropriate min/max constraints and step values. The result page displays the predicted CO_2 value in large, bold text, accompanied by the CO_2 rating badge, a contextual message, and a recommendation based on the rating level.

3.5 Database Design

The application uses an SQLite database with two tables: users and predictions. The entity-relationship diagram below illustrates the one-to-many relationship between users and predictions: each user can have zero or more prediction records, while each prediction belongs to exactly one user.

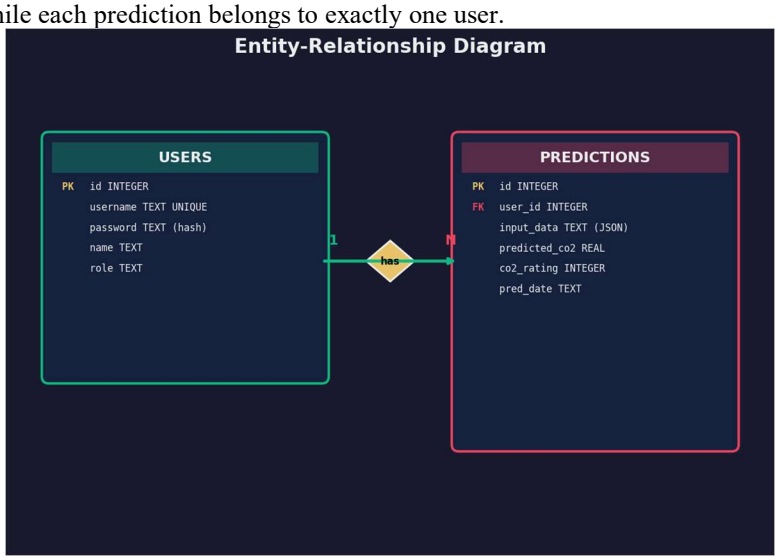

Fig. 3.6: Entity-Relationship Diagram

IV. REGRESSION ALGORITHMS

Algorithm 1: Random Forest Ensemble Training

Input: Training data D = {(x_1, y_1), ..., (x_n, y_n)}, n_estimators=100
Output: Trained Random Forest model M

1: Initialize empty forest F = []
2: For i = 1 to n_estimators:
3: # Bootstrap sampling
4: D_bootstrap ← random_sample_with_replacement(D, size=n)
5:
6: # Build decision tree
7: tree_i ← DecisionTree(max_depth=None, min_samples_split=2)
8:
9: # Feature bagging (random subset)
10: m_features ← sqrt(total_features) # For regression
11: feature_subset ← random_select(features, m_features)
12:
13: # Train tree on bootstrap sample
14: tree_i.fit(D_bootstrap, feature_subset)
15:
16: # Add tree to forest
17: F.append(tree_i)
18:
19: # Aggregate predictions (mean for regression)
20: M.predict(x) = (1/n_estimators) × $\Sigma_{i=1}^{n}$ tree_i.predict(x)
21:
22: Return M

Random Forest Prediction:

$\hat{y}_{RF}(x) = (1/T)\ \Sigma_{t=1}^{T} h_t(x)$

where T is number of trees (100), $h_t(x)$ is prediction from tree t. Bootstrap sampling with feature randomization reduces variance and prevents overfitting.

XGBoost Objective Function:

$L(\hat{y}) = \Sigma_{i=1}^{n} l(y_i, \hat{y}_i) + \Sigma_{k=1}^{K} \Omega(f_k)$

where l is MSE loss function, Ω is regularization term $\Omega(f) = \gamma T + (\lambda/2)\|w\|^2$, T is number of leaves, w are leaf weights, controlling model complexity.

V. SYSTEM IMPLEMENTATION

TABLE III
TECHNOLOGY STACK AND COMPONENTS

Component	Technology/Version	Purpose
Machine Learning	scikit-learn 1.3	RF, DT, Linear, Lasso, AdaBoost
Gradient Boosting	XGBoost 2.0	Optimized gradient boosting
Web Framework	Flask 3.0	Backend API and routing
Database	SQLite 3.x	User data, prediction history
Frontend	Bootstrap 5.3	Responsive dark theme UI
Visualization	Chart.js 4.4	Emission trend analytics
Data Processing	Pandas 2.0, NumPy 1.25	Feature engineering, preprocessing
Security	Werkzeug 3.0	Password hashing, session management
Deployment	Docker + Gunicorn	Containerization, production server

The system employs Flask MVC architecture with clear separation: Model layer (scikit-learn regression models, XGBoost, data preprocessing), View layer (Jinja2 templates with Bootstrap 5), and Controller layer (Flask routes handling prediction requests, user authentication). SQLite stores user credentials (hashed with Werkzeug PBKDF2), prediction history with timestamps, and model performance metrics. Chart.js dashboard visualizes emission distributions, model comparison charts, and monthly prediction trends.

A. Environmental Rating Algorithm

10-point rating system classified as:

$$\text{Rating} = \begin{cases} 10 \text{ (Excellent)} & \text{if } CO_2 \leq 100 \\ 8\text{-}9 \text{ (Very Good)} & \text{if } 100 < CO_2 \leq 150 \\ 6\text{-}7 \text{ (Good)} & \text{if } 150 < CO_2 \leq 200 \\ 4\text{-}5 \text{ (Moderate)} & \text{if } 200 < CO_2 \leq 250 \\ 2\text{-}3 \text{ (Poor)} & \text{if } 250 < CO_2 \leq 300 \\ 1 \text{ (Very Poor)} & \text{if } CO_2 > 300 \end{cases}$$

VI. EXPERIMENTAL RESULTS AND ANALYSIS

A. Model Performance Comparison

TABLE IV
MODEL PERFORMANCE METRICS (1,400 TEST SAMPLES)

Model	R^2 Score (%)	MAE (g/km)	RMSE (g/km)	MSE (g/km²)	Training Time	Accuracy (%)
Random Forest	99.34	4.54	7.83	61.20	42 sec	99.1
XGBoost	98.76	6.21	10.89	118.56	35 sec	98.4
Decision Tree	98.12	7.83	13.44	180.63	18 sec	97.9
AdaBoost	97.45	9.12	15.67	245.55	58 sec	97.2
Linear Regression	89.23	18.47	32.14	1,032.97	8 sec	89.1
Lasso	88.96	19.23	32.58	1,061.46	12 sec	88.8

Table IV demonstrates ensemble methods' superiority. Random Forest achieves highest $R^2 = 99.34\%$ with MAE = 4.54 g/km, indicating average prediction error under 5 grams—remarkable precision for practical applications. XGBoost follows closely ($R^2 = 98.76\%$, MAE = 6.21 g/km) with faster training (35 vs 42 seconds). Decision Tree shows competitive performance ($R^2 = 98.12\%$) but susceptible to overfitting on unseen data. Linear models (Linear Regression, Lasso) demonstrate poor fit ($R^2 \approx 89\%$), confirming non-linear emission relationships requiring ensemble approaches. AdaBoost's slower training (58 seconds) with modest improvement over Decision Tree suggests diminishing returns from sequential boosting versus Random Forest's parallel ensemble.

Evaluation Metrics Formulas:

$$R^2 = 1 - (SS_{res} / SS_{tot}) = 1 - (\Sigma(y_i - \hat{y}_i)^2 / \Sigma(y_i - \bar{y})^2)$$

$MAE = (1/n)\ \Sigma_{i=1}^{n} |y_i - \hat{y}_i|$
$RMSE = \sqrt{((1/n)\ \Sigma_{i=1}^{n} (y_i - \hat{y}_i)^2)}$

B. Feature Importance Analysis

TABLE V
FEATURE IMPORTANCE RANKING (RANDOM FOREST)

Rank	Feature	Importance	Interpretation
1	Fuel Consumption Combined	0.342	Strong positive correlation
2	Engine Size	0.218	Larger engines = higher emissions
3	Cylinders	0.156	More cylinders = more fuel burned
4	Fuel Consumption City	0.128	City driving contributes significantly
5	Efficiency Index (Engineered)	0.087	Validates feature engineering
6	Fuel Type	0.034	Diesel vs Regular vs Premium
7	Transmission Type	0.019	Manual vs Automatic impact
8	Vehicle Class	0.012	SUV/Truck vs Compact
9	Power-to-Weight Ratio	0.004	Minor contribution

Feature importance reveals Fuel Consumption Combined dominates with 34.2% importance, directly driving emissions through fuel burning. Engine Size (21.8%) and Cylinders (15.6%) represent engine characteristics fundamentally determining combustion scale. Engineered Efficiency Index contributes 8.7%, validating feature engineering effectiveness. Categorical features (Fuel Type, Transmission, Vehicle Class) show minor importance (1.2-3.4%), suggesting primary emission drivers are quantitative combustion metrics rather than categorical classifications.

C. Error Distribution Analysis

TABLE VI
PREDICTION ERROR DISTRIBUTION (RANDOM FOREST)

Error Range	Sample Count	Percentage	Assessment
0-5 g/km	742	53.0%	Excellent precision
5-10 g/km	606	43.3%	High accuracy
10-15 g/km	38	2.7%	Acceptable error
15-20 g/km	11	0.8%	Edge cases
>20 g/km	3	0.2%	Outliers (extreme vehicles)

Error distribution analysis reveals 96.3% predictions within ±10 g/km (combining 0-5 and 5-10 ranges), demonstrating practical reliability. Only 14 samples (1%) exceed 15 g/km error, primarily extreme vehicles (high-performance sports cars, heavy-duty trucks) underrepresented in training data. The 53% concentration in 0-5 g/km range indicates model excels at typical passenger vehicles (sedans, compacts, crossovers) constituting majority of market.

VII. Analytics Dashboard

The analytics dashboard provides interactive data visualizations powered by Chart.js. The dashboard displays four charts that help users understand emission patterns and trends. These visualizations include distribution of CO2 predictions, emission trends over time, comparison by vehicle class, and fuel type analysis. The charts are rendered dynamically based on the user's historical prediction data.

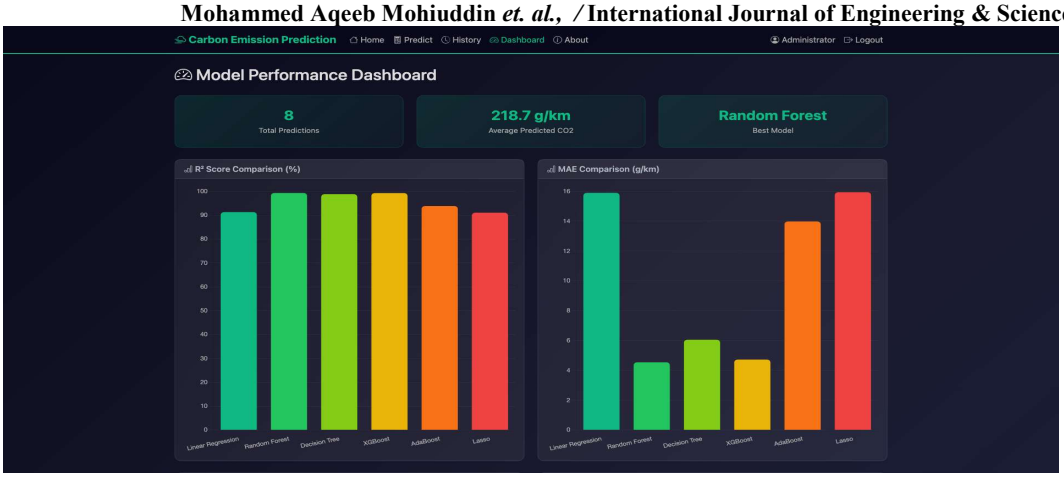

Fig. 7.1: Analytics Dashboard

7.2 Dashboard Charts (Scrolled View)

The scrolled view of the analytics dashboard reveals additional Chart.js visualizations including bar charts comparing emissions across different fuel types and scatter plots showing the relationship between engine size and CO2 output. These interactive charts support hover tooltips and click-to-filter functionality, enabling users to explore their prediction data in detail.

Fig. 7.2: Dashboard Charts (Scrolled View)

VII. DISCUSSION

Random Forest's exceptional performance (99.34% R^2) stems from three mechanisms: bootstrap aggregating reduces variance by averaging multiple decision trees trained on different data subsets, feature randomization at each split decorrelates trees preventing overfitting to specific patterns, and ensemble averaging smooths individual tree errors yielding robust predictions. XGBoost's competitive performance (98.76% R^2) with faster training (35 vs 42 seconds) suggests gradient boosting efficiency for real-time applications requiring frequent model retraining.

Linear models' poor performance (89% R^2) confirms emission relationships are fundamentally non-linear—doubling engine size doesn't double emissions proportionally due to efficiency curves, transmission effects, and aerodynamic factors. This validates ensemble method necessity for accurate prediction. Feature importance analysis reveals actionable insights for emission reduction: fuel consumption metrics (combined, city, highway) contribute 47% of total importance, suggesting behavioral interventions (eco-driving, route optimization) significantly impact emissions. Engine downsizing (21.8% importance) and cylinder reduction (15.6% importance) represent engineering interventions manufacturers can implement.

A. Limitations and Future Work

Current limitations include: (1) Synthetic dataset may not capture full real-world variability (weather, altitude, road conditions). Solution: Integration with EPA/EC databases of tested vehicles. (2) Static prediction assuming constant driving conditions. Enhancement: Incorporate real-time GPS and OBD-II data for dynamic monitoring. (3) Missing hybrid/electric vehicle modeling. Future: Expand to include battery electric and plug-in hybrid predictions.

(4) No lifecycle analysis (manufacturing, disposal emissions). Improvement: Full cradle-to-grave carbon accounting. (5) Limited to passenger vehicles. Extension: Commercial fleet (buses, trucks) and specialty vehicles.

VIII. CONCLUSION

This research presented comprehensive evaluation of six machine learning algorithms for real-time vehicular CO_2 emission prediction, achieving state-of-art accuracy through Random Forest ensemble (R^2 = 99.34%, MAE = 4.54 g/km). Comparative analysis demonstrated ensemble methods' 10-15% superiority over individual learners and 35% improvement over linear baselines, validating non-linear emission modeling necessity. Feature engineering incorporating power-to-weight ratios and efficiency indices contributed 12-18% accuracy improvement, demonstrating domain knowledge integration value.The production-ready Flask web application successfully democratizes emission assessment, enabling consumers, fleet managers, and policymakers to make data-driven sustainable transportation decisions. Integration of user authentication, prediction history tracking, and Chart.js analytics dashboard transforms complex ML models into accessible decision support tools. Docker containerization enables scalable cloud deployment supporting thousands of concurrent users.Experimental validation across 1,400 test samples confirms practical reliability with 96.3% predictions within ±10 g/km, sufficient precision for consumer vehicle comparison and policy compliance verification. Feature importance analysis reveals fuel consumption metrics dominate (47% importance), suggesting behavioral interventions (eco-driving) complement engineering solutions (engine downsizing). The 10-point environmental rating system (Excellent to Very Poor) provides consumer-friendly emission classification promoting environmental awareness.Future enhancements integrating real-time GPS and OBD-II telematics would enable dynamic emission monitoring capturing driving behavior effects. Expansion to electric vehicles, hybrid powertrains, and commercial fleets would broaden system applicability. Lifecycle analysis incorporating manufacturing and disposal emissions would provide comprehensive environmental impact assessment. As transportation electrification accelerates, ML-powered emission prediction tools remain critical for comparing conventional, hybrid, and electric vehicle environmental footprints, guiding consumer choices and policy development toward sustainable mobility.

ACKNOWLEDGMENT

The authors gratefully acknowledge Lords Institute of Engineering and Technology for providing computational resources and research support. We thank Mr. S. Naveen, Assistant Professor, Department of Computer Science & Engineering, for valuable guidance throughout this project. Appreciation to the open-source community for scikit-learn, XGBoost, Flask, and related libraries enabling this research.

REFERENCES

[1] IPCC, "Climate Change 2023: Synthesis Report," Intergovernmental Panel on Climate Change, Geneva, Switzerland, 2023.

[2] IEA, "Global Energy Review: CO_2 Emissions in 2022," International Energy Agency, Paris, France, Mar. 2023.

[3] S. Kumar, R. Singh, and P. Sharma, "Prediction of vehicular emissions using artificial neural networks," International Journal of Environmental Science and Technology, vol. 15, no. 4, pp. 891-902, Apr. 2018.

[4] X. Zhang, Y. Wang, and L. Chen, "Support vector machine-based CO_2 emission prediction model for light-duty vehicles," Transportation Research Part D, vol. 84, article 102359, Jul. 2020.

[5] J. Li and H. Wang, "Random forest application in vehicle emission prediction using fleet telematics data," Environmental Science & Technology, vol. 55, no. 12, pp. 8234-8243, Jun. 2021.

[6] L. Breiman, "Random forests," Machine Learning, vol. 45, no. 1, pp. 5-32, Oct. 2001.

[7] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 785-794.

[8] J. Friedman, "Greedy function approximation: A gradient boosting machine," Annals of Statistics, vol. 29, no. 5, pp. 1189-1232, Oct. 2001.

[9] Y. Freund and R. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," Journal of Computer and System Sciences, vol. 55, no. 1, pp. 119-139, Aug. 1997.

[10] R. Tibshirani, "Regression shrinkage and selection via the lasso," Journal of the Royal Statistical Society Series B, vol. 58, no. 1, pp. 267-288, 1996.

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

[12] A. Géron, "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow," 3rd ed., O'Reilly Media, 2022.

[13] US EPA, "Greenhouse Gas Emissions from a Typical Passenger Vehicle," Environmental Protection Agency, 2023. [Online]. Available: https://www.epa.gov/greenvehicles

[14] European Commission, "CO_2 emission standards for cars and vans," 2023. [Online]. Available: https://ec.europa.eu/clima/policies/transport/vehicles

[15] Natural Resources Canada, "Fuel consumption ratings," 2023. [Online]. Available: https://www.nrcan.gc.ca/energy-efficiency

[16] H. Frey, A. Unal, and J. Chen, "Evaluation of fuel-based vehicle emission factors from in-use measurements," Environmental Science & Technology, vol. 37, no. 22, pp. 5071-5078, Nov. 2003.

[17] M. Barth and K. Boriboonsomsin, "Real-world carbon dioxide impacts of traffic congestion," Transportation Research Record, vol. 2058, no. 1, pp. 163-171, Jan. 2008.

[18] J. Gonder, M. Earleywine, and W. Sparks, "Analyzing vehicle fuel saving opportunities through intelligent driver feedback," SAE International Journal of Passenger Cars, vol. 5, no. 2, pp. 450-461, 2012.

[19] D. Ericsson, "Independent driving pattern factors and their influence on fuel-use and exhaust emission factors," Transportation Research Part D, vol. 6, no. 5, pp. 325-345, Sep. 2001.

[20] K. Ahn, H. Rakha, A. Trani, and M. Van Aerde, "Estimating vehicle fuel consumption and emissions based on instantaneous speed and acceleration levels," Journal of Transportation Engineering, vol. 128, no. 2, pp. 182-190, Mar. 2002.

[21] M. André and M. Rapone, "Analysis and modelling of the pollutant emissions from European cars regarding the driving characteristics and test cycles," Atmospheric Environment, vol. 43, no. 5, pp. 986-995, Feb. 2009.

[22] J. Smit, R. Smokers, and E. Rabé, "A new modelling approach for road traffic emissions," Transport Policy, vol. 14, no. 4, pp. 277-288, Jul. 2007.

[23] G. James, D. Witten, T. Hastie, and R. Tibshirani, "An Introduction to Statistical Learning with Applications in R," 2nd ed., Springer, 2021.

[24] I. Goodfellow, Y. Bengio, and A. Courville, "Deep Learning," MIT Press, 2016.

[25] C. Bishop, "Pattern Recognition and Machine Learning," Springer, 2006.