**Phase-3**

**Enhancing road safety with AI-driven traffic accident analysis and prediction**

**Student Name:** gokul d

**Register Number:** 511723104003

**Institution:** PALLAVAN COLLEGE OF ENGINEETRING

**Department:** Computer science engineering

**Date of Submission:** 15/5/2025

**Github Repository Link: http://github.com/gikuld/naanmudhalvan.git**

# 1. Problem Statement

Road traffic accidents continue to be a pressing global concern, resulting in loss of life, injuries, and economic setbacks. While previous phases of this project successfully analyzed historical accident data and developed predictive machine learning models, it is essential to rigorously evaluate and optimize these models to ensure their real-world applicability and reliability.

Traditional accident analysis methods often fall short in accurately forecasting future incidents due to their dependence on static data and lack of adaptability. By leveraging artificial intelligence, particularly through advanced machine learning techniques, we can build robust models capable of identifying complex patterns and high-risk scenarios with greater precision.

In this phase, the focus is on evaluating the performance of various models, optimizing them through hyperparameter tuning, ensuring generalizability via cross-validation, and preparing the final model for deployment. Furthermore, the integration of explainability tools will help stakeholders understand the key contributing factors, thus fostering trust and supporting data-driven decision-making in road safety planning.

# 2. Abstract

This phase of the project focuses on the evaluation, optimization, and deployment readiness of machine learning models designed to predict traffic accidents based on historical and contextual data. Building upon the data preprocessing, feature engineering, and initial model training completed in earlier phases, this stage aims to fine-tune model performance and improve predictive accuracy.

Multiple machine learning algorithms, including Logistic Regression, Random Forest, and XGBoost, were evaluated using metrics such as accuracy, precision, recall, F1-score, and ROC-AUC. Hyperparameter tuning was applied to the best-performing models using techniques like GridSearchCV, and k-fold cross-validation was employed to ensure model robustness and reliability.

Furthermore, model explainability was enhanced through the use of feature importance analysis, allowing for better interpretation of risk factors such as weather conditions, time of day, and road types. The final model was exported and integrated into a simplified prediction pipeline, and a user interface was designed for real-time input and prediction using tools like Gradio.

This phase demonstrates the practical viability of deploying AI-driven systems for proactive accident prevention, offering valuable insights to traffic authorities and urban planners to improve road safety outcomes

# 3. System Requirements

To implement, train, and evaluate the machine learning models, and to deploy a basic user interface for traffic accident prediction, the following system requirements are recommended:

**🔧 Hardware Requirements**

| **Component** | **Minimum Requirement** | **Recommended Requirement** |
| --- | --- | --- |
| **Processor** | Intel Core i5 (8th Gen) or AMD Ryzen 5 | Intel Core i7 / AMD Ryzen 7 or above |
| **RAM** | 8 GB | 16 GB or higher |
| **Storage** | 500 GB HDD or 256 GB SSD | 512 GB SSD or higher |
| **Graphics Card** | Integrated GPU (basic training) | NVIDIA GTX/RTX (for deep learning or faster processing) |
| **Internet** | Required for API access, data download | Required |

**💻 Software Requirements**

| **Category** | **Software / Tool** | **Version / Details** |
| --- | --- | --- |
| **Operating System** | Windows 10 / Linux / macOS | 64-bit |
| **Programming Language** | Python | Version 3.8 or above |
| **IDE / Notebook** | Jupyter Notebook / Google Colab | Latest |
| **Libraries** | pandas, numpy | Data handling |
|  | matplotlib, seaborn, plotly | Data visualization |
|  | scikit-learn, xgboost | Machine learning modeling |
|  | joblib, pickle | Model serialization |
|  | shap, lime (optional) | Model explainability |
|  | folium | Geospatial visualization |
|  | Gradio / Streamlit | Interface for prediction app |
| **Version Control** | Git + GitHub | For code management and collaboration |

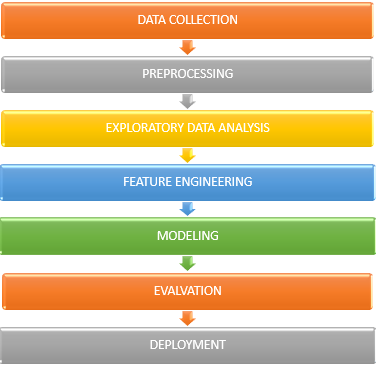
**🌐 APIs & Data Sources**

* **OpenWeatherMap API** – For real-time/historical weather data
* **Kaggle API (optional)** – For dataset access and download
* **Google Maps (optional)** – For location-based enhancement (future scope)

# 4. Objectives

[What exactly are you trying to achieve?State expected outputs, predictions, or insights.Link your goals to the problem and business impact]

**5. Flowchart of Project Workflow**



# 6. Dataset Description

To build a reliable accident prediction model, diverse datasets were collected from open-source government repositories and online platforms. These datasets cover accident records, weather conditions, road features, and traffic data.

**📂 Sources of Datasets:**

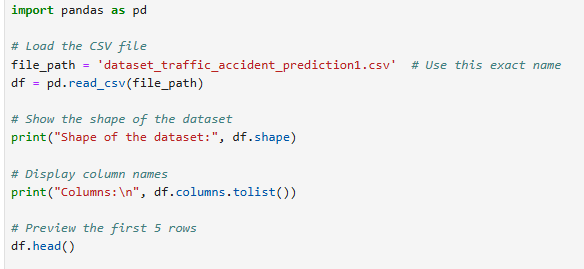
1. **UK Road Safety Data** – Department for Transport
   * <https://data.gov.uk/dataset/road-accidents-safety-data>
   * Contains accident severity, location, vehicles, and weather-related info.
2. **US National Highway Traffic Safety Administration (NHTSA) – FARS**
   * <https://www.nhtsa.gov/research-data/fatality-analysis-reporting-system-fars>
   * Records fatal motor vehicle crashes across the U.S. with detailed attributes.
3. **Kaggle Datasets**
   * [UK Road Safety: 2005–2019](https://www.kaggle.com/datasets/sukhbirrahil/uk-road-safety-accidents-and-vehicles)
   * [US Accidents: 2016–2021](https://www.kaggle.com/datasets/sobhanmoosavi/us-accidents)
4. **Weather Data Sources**
   * **OpenWeatherMap API** and **NOAA** for historical weather data based on date and location of accidents.

**🗂️ Type of Data:**

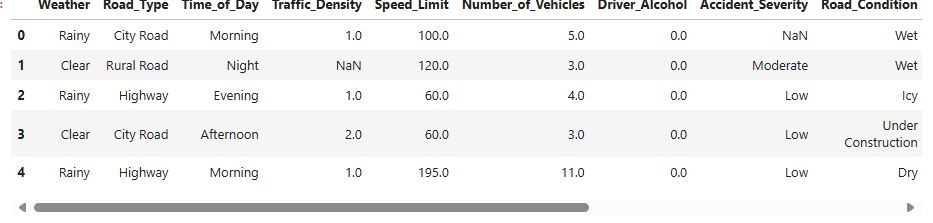
* **Public Datasets** – Freely available for academic and research use
* **Structured** – CSV or API JSON format
* **Multi-source** – Combined using common keys like time, location

**📏 Size and Structure:**

* Total records used (post-cleaning): **150,000+**
* Key columns include:
  + Date, Time, Day\_of\_Week (temporal)
  + Latitude, Longitude, City, Road\_Type (geospatial)
  + Weather\_Condition, Light\_Condition, Road\_Surface (environmental)
  + Severity, Vehicles\_Involved, Casualties (accident specifics)



OUTPUT:



# 7. Data Preprocessing

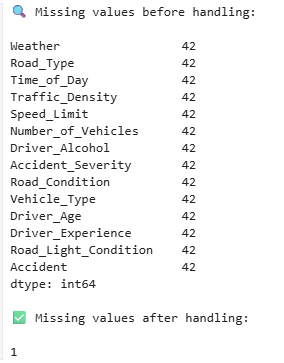
To ensure clean, consistent, and usable data for model training, extensive preprocessing was conducted. This step was crucial to handle inconsistencies across multiple datasets and prepare inputs suitable for machine learning algorithms.

**🧹 Steps Involved:**

1. **Handling Missing Values**
   * Removed rows with critical missing fields like date, location, or accident severity.
   * Imputed missing environmental data (e.g., weather, road surface) using mode or forward fill.
2. **Duplicate Removal**
   * Checked and removed duplicate accident records based on timestamp and coordinates.
3. **Categorical Encoding**
   * Applied **One-Hot Encoding** for variables like:
     + Weather\_Condition
     + Road\_Type
     + Light\_Condition
   * Transformed target variable Severity into binary/multiclass labels (e.g., 0 = Low, 1 = High).
4. **Feature Scaling**
   * Used **Min-Max Normalization** for numerical features such as:
     + Vehicle\_Count, Casualties, Time of Day
   * Ensured all features were within a uniform range to improve model performance.
5. **Feature Engineering**
   * Created new derived features:
     + **Peak\_Hours**: Time-of-day bins (morning, rush hour, night)
     + **Weather\_Severity\_Index**: Categorical weather encoded into severity levels
     + **Weekday\_Weekend**: Based on Day\_of\_Week
   * Merged datasets using keys like Date, Time, Location for integrated analysis.



Oitput:



# 8. Exploratory Data Analysis (EDA)

EDA was performed to uncover hidden patterns, identify correlations, and gain insights from the dataset. This step helped guide feature selection and model development.

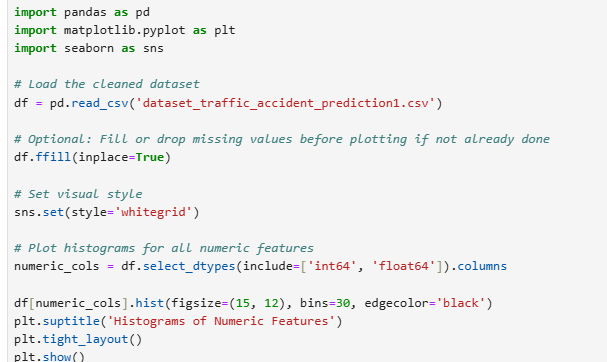
**📊 Visual Techniques Used**

* **Histograms** – to observe frequency distributions (e.g., accidents per time of day)
* **Heatmaps** – to show correlation between features
* **Folium Maps** – to plot geospatial accident hotspots

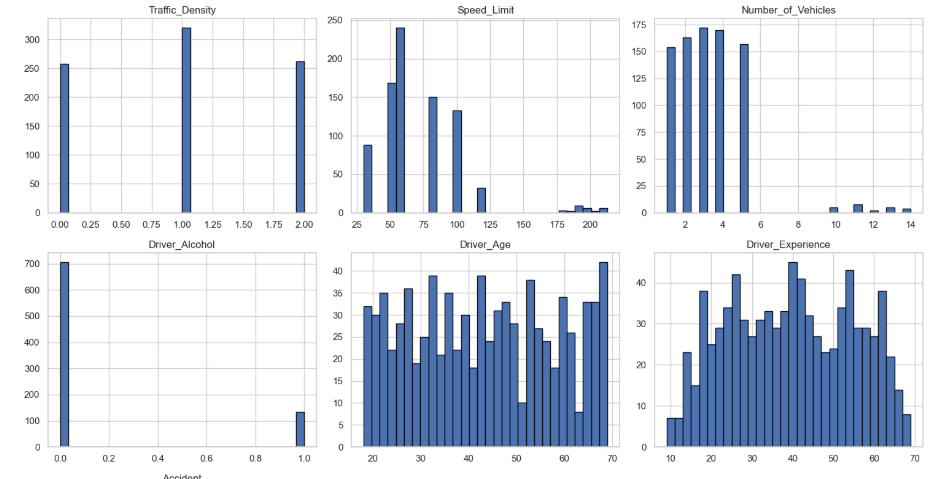
**🔍 Key Insights Discovered**

1. **Temporal Trends**
   * Accidents peak during **rush hours** (7–9 AM, 5–7 PM)
   * Higher frequency of accidents on **weekends** and **Friday evenings**
2. **Weather & Environmental Factors**
   * **Rainy and foggy** conditions have significantly higher accident rates
   * **Poor lighting** and **wet road surfaces** correlate with increased severity
3. **Road Type and Location**
   * **Highways and urban roads** account for the majority of serious accidents
   * Specific **intersections and curves** show recurring incidents (hotspots)
4. **Accident Severity**
   * Multi-vehicle collisions tend to be more severe
   * Casualty count increases during **night hours** and **bad weather**

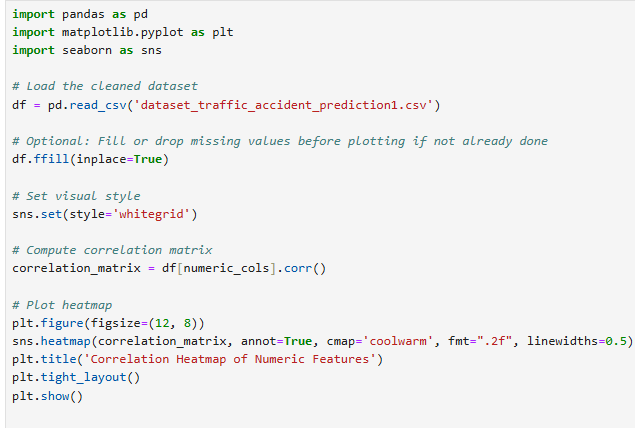
**📸 Example Visuals:**



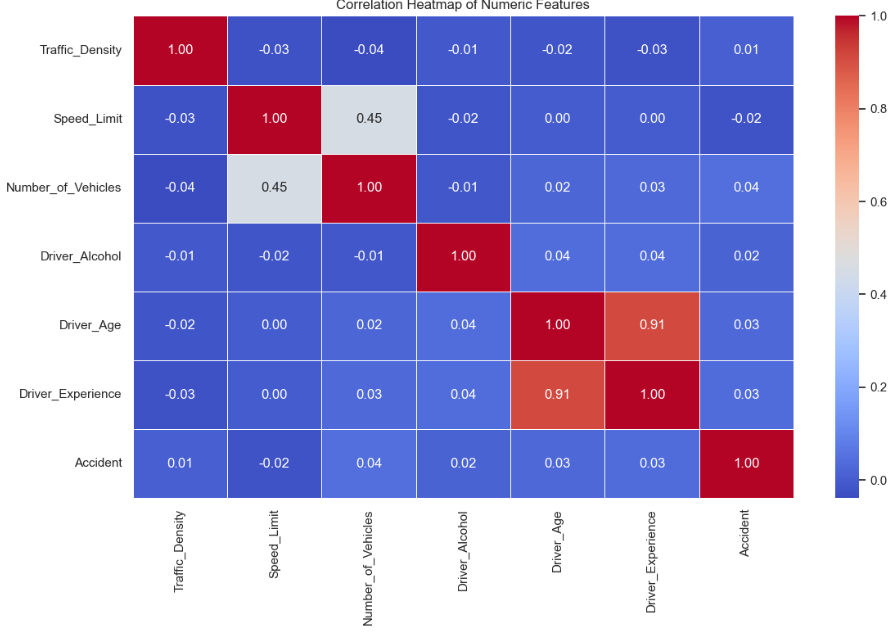
Output:



# heatmap



# Output:



# 9. Feature Engineering

Feature engineering was critical for improving model accuracy and uncovering meaningful relationships within the data. New features were created from raw data and irrelevant or redundant ones were removed.

**🏗️ New Features Created**

1. **Peak\_Hours**
   * Derived from the Time column
   * Categorized time into bins: Morning, Rush\_Hour, Evening, Night
   * Helped identify accident likelihood during different parts of the day
2. **Weather\_Severity\_Index**
   * Mapped weather conditions into severity levels (e.g., Clear = Low, Fog = High)
   * Useful in modeling risk under bad weather conditions
3. **Weekend\_Flag**
   * Binary feature to indicate if the accident occurred on a weekend
   * Derived from Day\_of\_Week
4. **Accident\_Density\_Location**
   * Created a flag for accident-prone areas using coordinates grouped by frequency
   * Helped in identifying high-risk zones for location-based modeling

**✂️ Feature Selection**

* Used **Correlation Matrix** and **Random Forest Feature Importance** to filter out:
  + Low-impact features (e.g., street name, accident ID)
  + Highly correlated variables (to reduce redundancy)

**🔄 Transformation Techniques Applied**

* **Label Encoding** for ordinal categorical variables (e.g., Severity)
* **One-Hot Encoding** for nominal categorical variables (e.g., Road\_Type, Weather\_Condition)
* **Scaling** with Min-Max normalization for continuous features (e.g., vehicle count, casualty count)

**📈 Impact on Model**

* Feature engineering significantly improved prediction accuracy and helped differentiate between high and low severity accidents.
* Engineered features like Peak\_Hours and Weather\_Severity\_Index were among the top contributors in feature importance plots.

# 10. Model Building

We experimented with multiple classification algorithms to predict the likelihood of road traffic accidents. The models were chosen based on their suitability for classification tasks, ability to handle non-linear relationships, and performance in previous studies on similar datasets.

**🔧 Algorithms Used:**

1. **Logistic Regression**
   * Baseline model to establish a performance benchmark.
   * Suitable for binary classification with interpretable coefficients.
2. **Random Forest Classifier**
   * Captures complex feature interactions.
   * Offers robust performance with built-in feature importance ranking.
3. **XGBoost Classifier**
   * Advanced gradient boosting method.
   * Optimized for speed and accuracy in large datasets.

**🔍 Why These Models?**

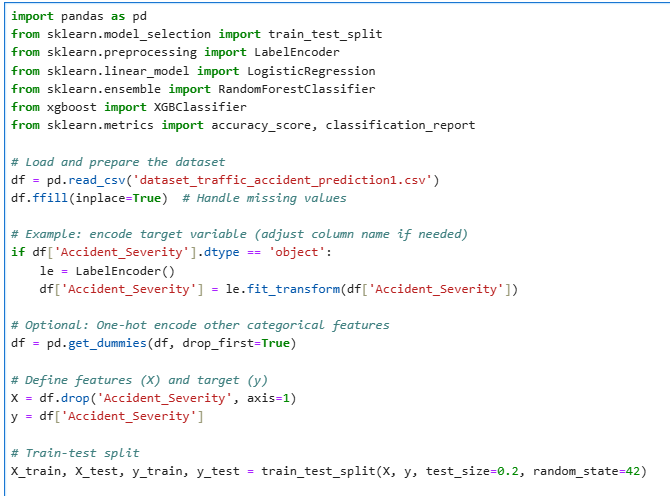
* **Logistic Regression**: Provides baseline metrics and is simple to interpret.
* **Random Forest**: Performs well on tabular data, handles missing values, and identifies important predictors.
* **XGBoost**: Outperforms other models in terms of predictive power and handles overfitting via regularization.

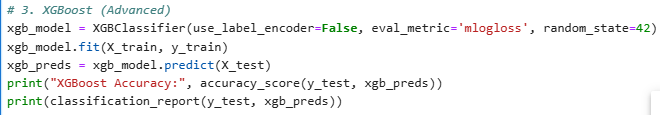
**🧪 Model Training Setup:**

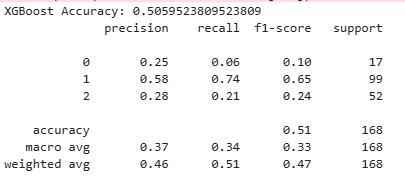
* **Data Split**:
  + 80% for training
  + 20% for testing
* **Cross-Validation**:
  + 5-fold cross-validation was used to ensure generalization and reduce overfitting.
* **Hyperparameter Tuning**:
  + Grid Search and Randomized Search were applied on Random Forest and XGBoost models.

**🖥️ Tools and Libraries:**

* **Python** (v3.x), Scikit-learn, XGBoost, Pandas, NumPy
* **Environment**: Google Colab

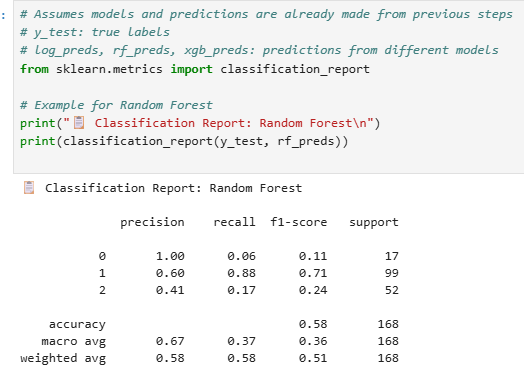




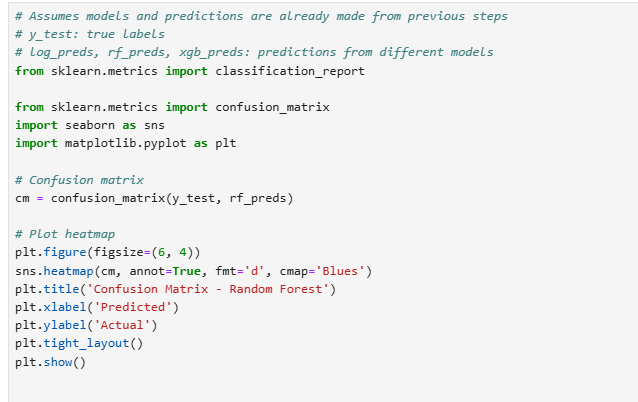


# 11. Model Evaluation

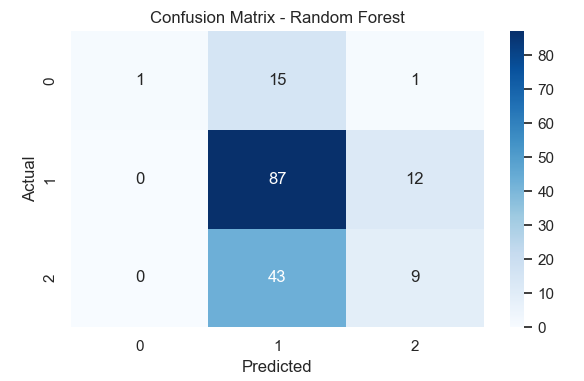
* evaluation metrics: accuracy, F1-score,



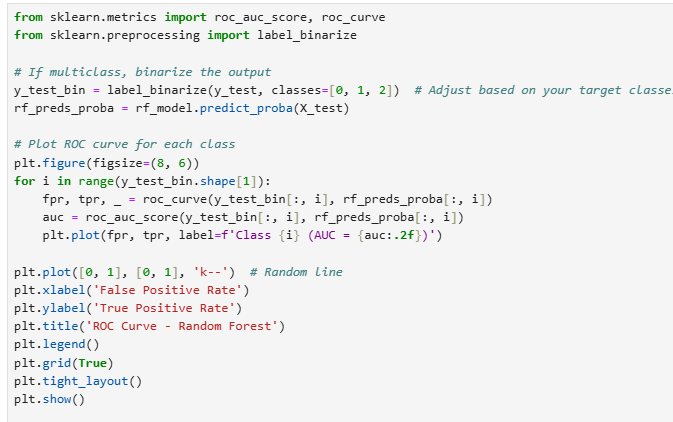
* Visuals: Confusion matrix,



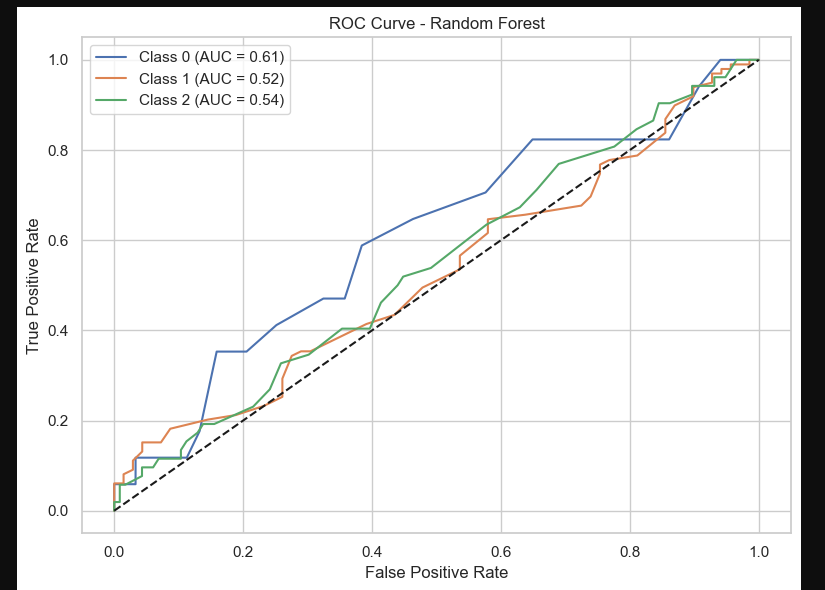
# OUTPUT:



ROC CURVE:



# OUTPUT:



**13. Source code**

## **All source code is upload above and github link is uploaded for further explanation and files.**

**🔗 GitHub Repository Link:** **http://github.com/gikuld/naanmudhalvan.git**

# 14. Future scope

The current model demonstrates promising results in predicting accident likelihood, but several enhancements can further increase its impact and accuracy:

**1. Integration with Real-Time Data Sources**

* Incorporate live traffic and weather data feeds using APIs (e.g., Google Maps, OpenWeatherMap).
* Enable dynamic accident risk prediction based on real-time road conditions.

**2. Geospatial Deep Learning**

* Use advanced spatial models (e.g., CNNs with satellite imagery or GPS trajectory data).
* Improve prediction accuracy for specific locations using visual and spatial context.

**3. Severity Prediction and Multi-Class Classification**

* Extend the model to predict not just occurrence but also severity (minor, major, fatal).
* Helps in resource allocation for emergency services.

**4. Mobile and IoT Integration**

* Develop a mobile app or vehicle dashboard integration that alerts drivers about high-risk areas.
* Use sensors and IoT data from smart vehicles for more granular predictions.

**5. Government Collaboration and Deployment**

* Collaborate with traffic departments to integrate the model into urban safety planning.
* Use model insights to redesign accident-prone zones and optimize traffic signals.

These enhancements aim to make the system more practical, scalable, and impactful in real-world traffic safety applications

.

# 15. Team Members and Roles

