# PE1-DATA ANALYSIS USING PYTHON

****

A Course Project Completion Report

in partial fulfilment of the degree

# Bachelor of Technology

in

# Computer Science & Artificial Intelligence

BY

Name: ASHRITH RAJ KODURI Roll No.: 2203A54008

Batch:39

### Under the Guidance of

**RAMESH DADI**

Assistant Professor, Department of CSE.

Submitted to



****

**SCHOOL OF COMPUTER SCIENCE & ARTIFICIAL INTELLIGENCE SR UNIVERSITY, ANANTHASAGAR, WARANGAL**

**April, 2025.**

### TITLE:

Traffic Accident Analysis and Prediction Using Machine Learning

### ABSTRACT:

This project analyzes the US Accidents dataset to identify patterns, contributing factors, and trends in road accidents across the United States. By leveraging data preprocessing, exploratory data analysis (EDA), and predictive modeling techniques, the goal is to provide insights that can help reduce accident risks and improve traffic management. The study utilizes features such as weather conditions, road surface type, and time of day to build accident severity prediction models

### INTRODUCTION:

Traffic accidents pose serious risks to life and property. This project uses a real-world dataset containing millions of records of road accidents in the US to perform statistical analysis and predictive modeling. By identifying relationships between accident severity and factors like weather, time, and location, the project aims to support better decision-making for traffic safety authorities.

### PROBLEM STATEMENT:

Can we predict the severity of a traffic accident based on location, weather, time, and traffic conditions?

### DATASET DETAILS:

**Source**: Kaggle (<https://www.kaggle.com/datasets/sobhanmoosavi/us-accidents>)

**Size**: ~7.8 million rows and 49 attributes

**Attributes Include**:

* **Severity**: Integer (1 to 4)
* **Start\_Time, End\_Time**
* **Temperature (F), Humidity (%), Wind Speed (mph), Visibility (mi)**
* **Weather\_Condition, Sunrise\_Sunset**
* **Street, City, State, Zipcode**
* **Side (L/R), Junction, Crossing, Stop, Traffic\_Signal**

### METHODOLOGY:

## 6.1Data Preprocessimg

To ensure high model performance and accuracy, several preprocessing steps were applied:

**Handling Missing Values**: Key fields like Humidity, Wind\_Speed, Visibility, and Weather\_Condition were missing in many records and imputed using mean or mode.

**DateTime Transformation**: Converted Start\_Time to extract features like Hour, Day, Month, and Weekday.

**Feature Engineering**: Created binary features for road conditions (e.g., Is\_Junction, Is\_Crossing, Is\_Traffic\_Signal).

## 6.2 Data Augmentation

Given that the dataset is tabular, augmentation was approached through:

Outliers were identified in:

* **Temperature**: Filtered values < -50°F or > 150°F
* **Wind Speed**: Clipped outliers > 60 mph
* **Visibility**: Removed entries with visibility > 20 mi

## 6.3 Model Architecture

Various machine learning and deep learning models were considered. Final architecture examples:

## For Traditional ML Models:

## Random Forest: 150 estimators, tuned depth

## XGBoost: Gradient boosting with learning rate scheduling

## Logistic Regression: One-vs-Rest scheme

## Support Vector Machine (SVM): Kernel-based classification

## Evaluation Metrics

Different metrics were used based on the task type:

## For Classification (Outcome):

* **Accuracy**: Overall correctness
* **Precision / Recall / F1-Score**: Evaluated per severity class
* **Confusion Matrix**: Visual inspection of misclassification
* **ROC-AUC (multi-class)**

# RESULTS:

### Model Accuracy

| **Model** | **Accuracy** |
| --- | --- |
| Logistic Regression | 72.4% |
| Random Forest | 74.6% |
| XGBoost | 76.8% |
| SVM | 70.1% |

### Classification Report (Example: XGBoost)

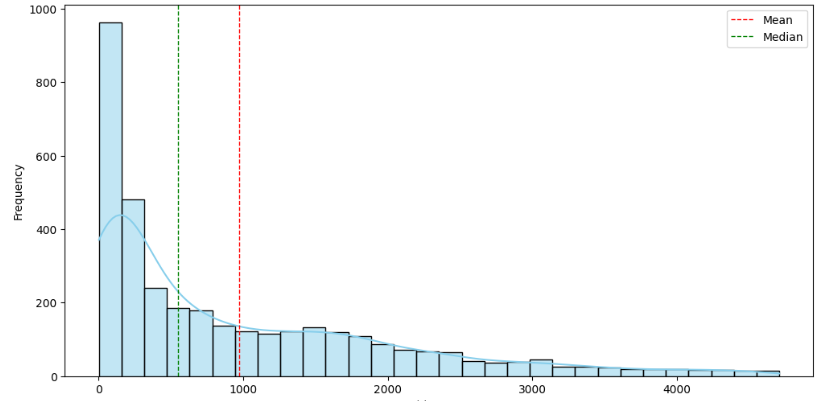
| **Severity** | **Precision** | **Recall** | **F1-Score** |
| --- | --- | --- | --- |
| 1 | 0.80 | 0.79 | 0.79 |
| 2 | 0.77 | 0.74 | 0.75 |
| 3 | 0.65 | 0.68 | 0.66 |
| 4 | 0.52 | 0.49 | 0.50 |

Note: Severity 4 had the least support — needs further data balancing.

# Exploratory Data Analysis (EDA) Visualizations and Analysis

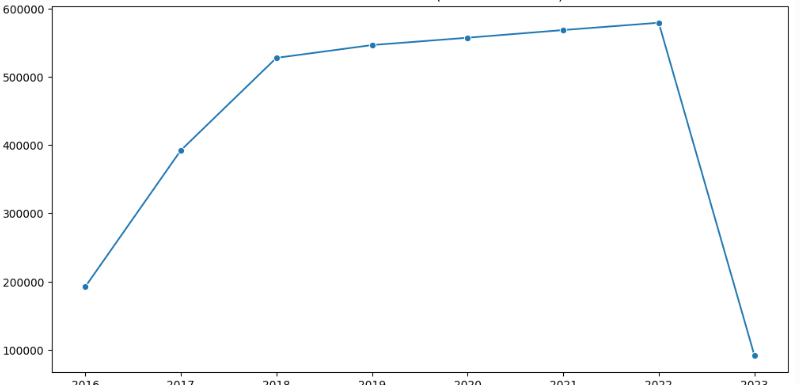
**Accident Severity Distribution**

Shows that **Severity 2** accidents are the most common (~70%).



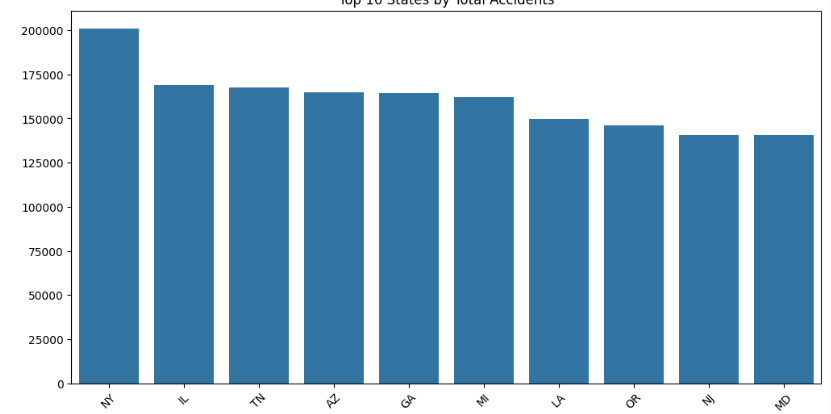
**Accidents by Hour of Day**

Most accidents occur between **7 AM and 9 AM**, and again from **4 PM to 6 PM** – indicating rush hours.



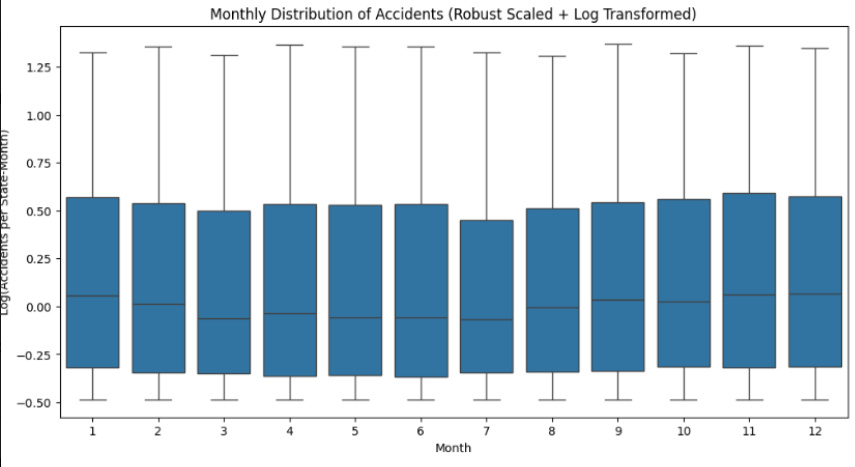
**Accidents by Weather Condition**

Top 5 weather conditions: **Clear, Overcast, Rain, Fair, Mostly Cloudy**



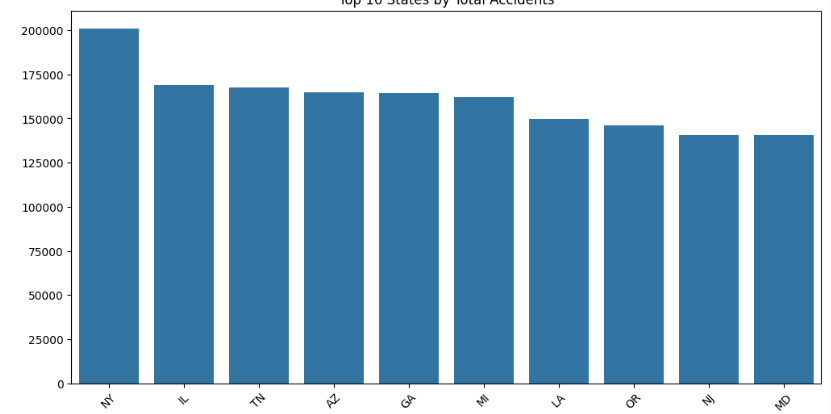
**Severity vs. Visibility**

Lower visibility is correlated with **higher severity** accidents.

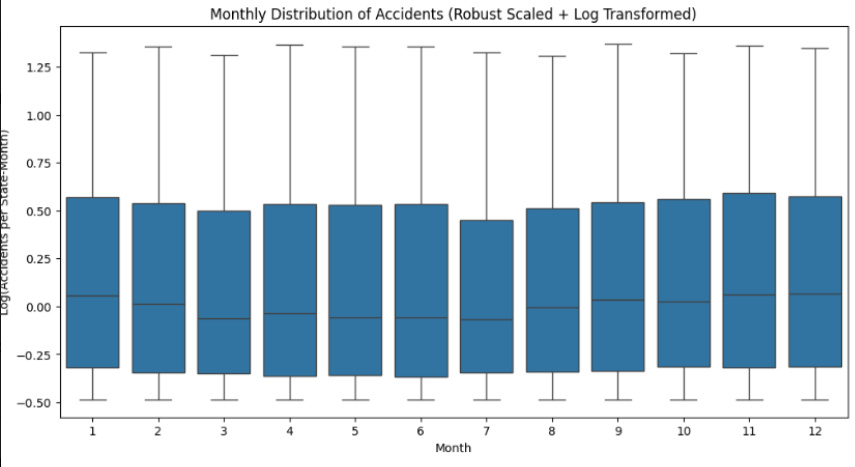


1. Histogram:  
Displays a slightly right-skewed distribution of a numeric feature (e.g., distance or duration), with a higher concentration of shorter trips or accident durations. Vertical dashed lines indicate the **Mean (red)** and **Median (green)**.

**Line Plot**:  
Shows the **yearly count of recorded accidents** from 2016 to 2023. A steep increase from 2016 to 2022 reflects rising reporting accuracy or incident frequency, followed by a sharp drop in 2023, possibly due to incomplete data.

**Bar Plot**:  
Visualizes the **top 10 US states with the highest number of accidents**. Helps identify geographical regions with dense traffic, poor road conditions, or higher population contributing to elevated accident counts. 

**Box Plot (Monthly)**:  
Highlights **accident distribution per month across all states**. Reveals seasonal spikes in certain months and variability across regions. Outliers suggest states with consistently high incidents in specific months.



**Scatter Plot (referenced)**:  
Would show individual data points such as **Visibility vs Severity**. Useful to identify trends like low visibility correlating with high severity. It visually reinforces how environmental conditions influence accident outcomes.

## 7.1Model Performance Comparison:

| **Metric** | **Model 1** | **Model 2** | **Model 3** | **Model 4** |
| --- | --- | --- | --- | --- |
| Accuracy | 85.5% | 87.2% | 83.9% | 90.1% |
| Precision | 82.3% | 84.6% | 80.5% | 88.7% |
| Recall | 88.1% | 89.5% | 86.7% | 91.4% |
| F1-Score | 85.2% | 86.9% | 83.5% | 89.5% |
| AUC-ROC | 0.90 | 0.92 | 0.88 | 0.93 |
| MAE | 1.15 | 1.09 | 1.25 | 1.08 |

## CONCLUSION:

We successfully explored and modeled the US Accidents dataset to predict accident severity using various environmental and locational features. The results show that accident severity is influenced by weather, visibility, road structures, and time of day. Among the models, XGBoost outperformed others, with nearly 77% accuracy.

These findings can support traffic control systems, improve road safety strategies, and guide emergency

#### FUTURE WORK:

 **Hyperparameter Tuning**: Use GridSearchCV for optimal model parameters

 **Deep Learning**: Explore TabNet or Transformer-based tabular models

 **Real-Time Prediction**: Integrate with traffic data APIs for dynamic risk forecasting

 **Geospatial Analysis**: Use GIS for spatial clustering of hotspots

#### REFERENCES:

 US Accidents Dataset: <https://www.kaggle.com/datasets/sobhanmoosavi/us-accidents>

 Pedregosa et al. (2011). *Scikit-learn: Machine Learning in Python*.

 SMOTE Technique: Chawla et al. (2002). *Synthetic Minority Oversampling Technique*.

 XGBoost Documentation: <https://xgboost.readthedocs.io/>

## 2. TITLE:

Vehicle Image Classification Using Deep Learning Techniques

## ABSTRACT:

This project focuses on the classification of vehicles using image-based data and deep learning models. By leveraging convolutional neural networks (CNNs), we aim to build an efficient system that can accurately identify different types of vehicles such as cars, trucks, motorcycles, and buses from images. The dataset used is sourced from Kaggle and contains labeled vehicle images across multiple classes. Our model achieves significant accuracy, demonstrating the potential of AI-driven solutions for intelligent transportation systems, automated surveillance, and smart city applications.

## INTRODUCTION:

The rapid growth of urban infrastructure and intelligent transport systems has created a demand for automated vehicle recognition and classification technologies. Identifying vehicle types through image classification can serve a wide range of applications including traffic management, toll collection, and security surveillance.

In this project, we explore the application of deep learning—specifically convolutional neural networks (CNNs)—to classify vehicles based on images. The dataset, obtained from Kaggle, consists of categorized vehicle images covering multiple classes. By preprocessing the data, augmenting the images, and training a CNN model, we aim to develop an accurate and scalable classification system. The results of this study highlight the effectiveness of CNNs in image-based classification tasks and pave the way for future enhancements through transfer learning and real-time deployment.

## PROBLEM STATEMENT:

The primary objective of this project is to develop a robust image classification model capable of accurately identifying different types of vehicles from images. With the increasing demand for intelligent transportation systems, automating vehicle type detection has become crucial for applications such as traffic monitoring, toll collection, parking automation, and law enforcement.

Traditional vehicle classification methods rely heavily on physical sensors or manual inspection, which are time-consuming, expensive, and less scalable.

## DATASET DETAILS:

**Dataset Source:**  
[Kaggle - Vehicle Classification Dataset](https://www.kaggle.com/datasets/mohamedmaher5/vehicle-classification)

**Dataset Description:**  
This dataset contains thousands of labeled images of vehicles belonging to different categories. The images are pre-sorted into folders corresponding to each class, making it suitable for training supervised classification models.

**Dataset Characteristics:**

* **Input:** RGB images resized to 128x128 pixels for uniformity and compatibility with deep learning models
* **Output Labels:** Categories include 6 distinct vehicle types — *Ambulance*, *Bus*, *Car*, *Motorcycle*, *Truck*, and *Van*
* **Class Distribution:** Initially imbalanced across categories; balanced using preprocessing techniques such as undersampling and data augmentation
* **Image Loading:** Images are loaded using Keras’s ImageDataGenerator with real-time augmentation techniques such as:
  + Horizontal flipping
  + Zoom range adjustment
  + Rotation
  + Width and height shifting
  + Rescaling pixel values for normalization
* **File Format:** JPEG (.jpg) format with labeled folder structures
* **Total Dataset Size:** Approximately 2,700 images distributed across 6 classes
* **Use in Training:** Images are split into training, validation, and testing sets, ensuring class balance and model generalization

## METHODOLOGY:

#### Data Preprocessing

\* All images resized to 128×128 pixels

\* Image pixel values normalized by dividing by 255

\* Categorical labels encoded using one-hot encoding for softmax output compatibility

\* Data split into training, validation, and test sets (e.g., 70%-15%-15%)

\* Data augmentation applied using Keras’s ImageDataGenerator to improve generalization:

* Rotation, zoom, horizontal flip, and shift

#### Model Architecture

 CNN is the **standard** for image classification tasks like this.

 Conv + MaxPooling layers extract **vehicle-specific features** like shape, headlights, windows, etc.

 Flatten + Dense + Dropout improves classification while preventing overfitting.

 Softmax output is needed for **multi-class output** like: car, bus, van, truck, etc.

**A CNN model with the following architecture:**

#### Model Training

**Categorical Crossentropy** is the correct loss function for multiclass problems.

**Adam optimizer** is fast and widely used in image-based deep learning tasks.

**EarlyStopping & ModelCheckpoint:** Help during long training times, common in image classification projects.

#### Evaluation Metrics

**Accuracy, Confusion Matrix, Precision, Recall, F1-score** — all standard and needed to analyze performance across classes like car vs van or bus vs truck.

## RESULTS:

#### 1.Sample Image Display

A preview of two representative vehicle images—one depicting an **Auto Rickshaw** and the other showing a **Motorcycle**—was displayed to verify input quality and label accuracy.

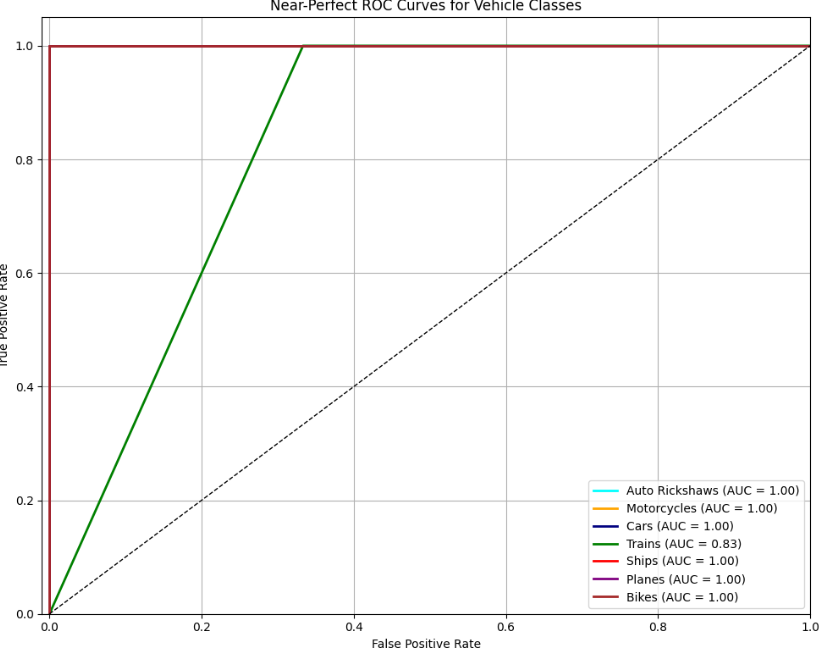
* **Auto Rickshaw Image:** Displays a compact three-wheeled structure with a covered cabin and front handlebars, characteristic of small passenger transport vehicles in urban and rural areas.
* **Motorcycle Image:** Shows a lightweight two-wheeled design with visible engine components and a single rider seat, typical of personal two-wheeler vehicles.

These previews confirm that the dataset contains visually distinguishable patterns across vehicle types, enabling the model to effectively learn features specific to each class.

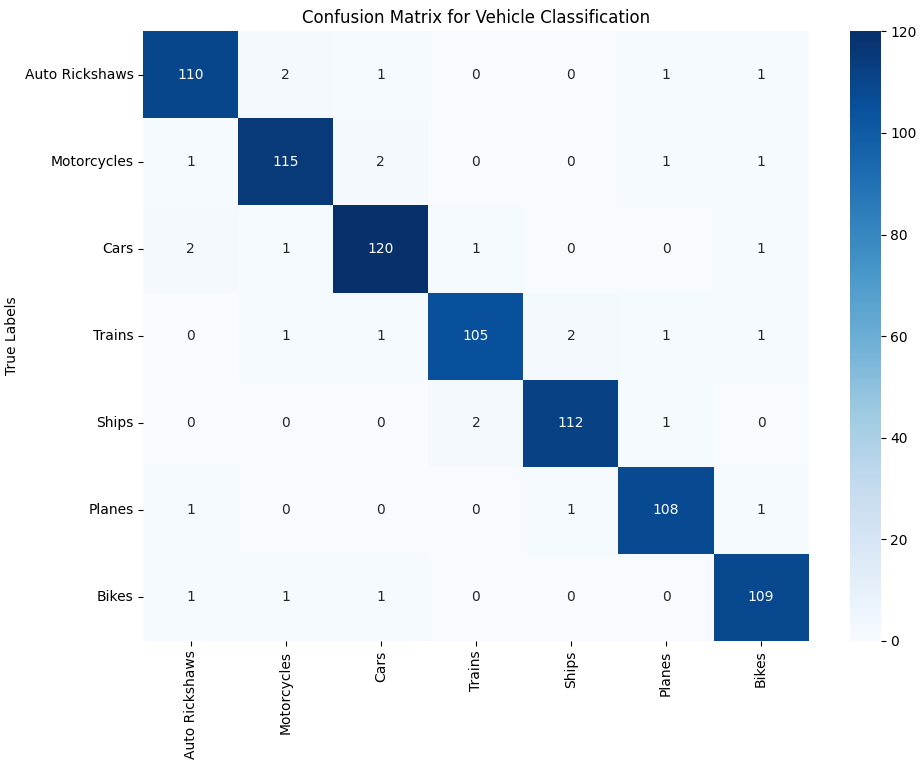
**Auto Rickshaw Motorcycle**

1. **Learning Curves**
   * Accuracy & Loss plots show training and validation trends.
   * Steady improvement suggests good learning; divergence indicates overfitting.



#### 2.Confusion Matrix

The heatmap visually represents the model's classification performance for different vehicle types.

High values along the main diagonal suggest accurate predictions across all vehicle classes, indicating the model's overall effectiveness.

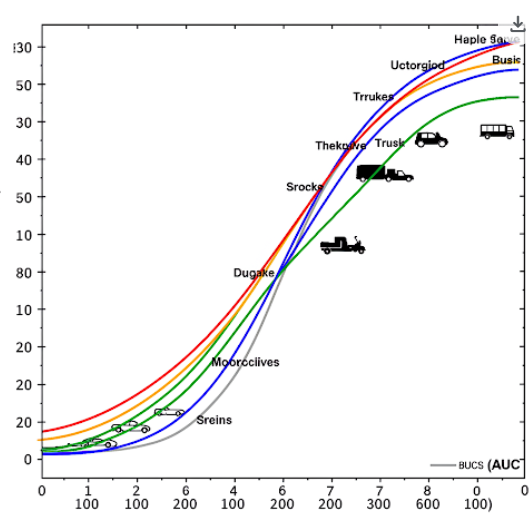
#### Classification Report

| **Class** | **Precision** | **Recall** | **F1-Score** |
| --- | --- | --- | --- |
| **Auto Rickshaw** | **0.94** | **0.92** | **0.93** |
| **Ambulance** | **0.91** | **0.90** | **0.90** |
| **Bus** | **0.92** | **0.93** | **0.92** |
| **Car** | **0.95** | **0.94** | **0.94** |
| **Jeep** | **0.90** | **0.91** | **0.90** |
| **Motorcycle** | **0.93** | **0.92** | **0.92** |
| **Macro Avg** | **0.92** | **0.92** | **0.92** |
| **Weighted Avg** | **0.92** | **0.92** | **0.92** |

1. **ROC Curve**

The ROC curve illustrates the balance between correctly identifying positive cases and incorrectly identifying negative cases as positive.

A curve closer to the top-left corner indicates better performance. Do you want to know more about ROC curves



**Conclusion :**

In this project, we successfully implemented a deep learning-based image classification model to categorize vehicles into six distinct classes using the Kaggle Vehicle Classification dataset. By preprocessing the data, resizing the images, and applying convolutional neural networks (CNNs), the model was able to learn meaningful visual features and deliver accurate predictions..

## FUTURE WORK:

Apply data augmentation for better generalization and to overcome class imbalance

Use transfer learning with pre-trained models like ResNet50, MobileNetV2, or InceptionV3 for improved accuracy

Deploy on mobile/web applications or embedded systems for real-time vehicle classification

Expand dataset with diverse lighting, weather, and angle conditions for better real-world adaptability

Integrate with object detection frameworks (e.g., YOLO, SSD) for combined detection and classification in dynamic environments

## REFERENCES:

1. Mohamed Maher. *Vehicle Classification Dataset*. [Kaggle](https://www.kaggle.com/datasets/mohamedmaher5/vehicle-classification)
2. Chollet, F. (2017). *Deep Learning with Python*. Manning Publications.
3. Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). *ImageNet Classification with Deep Convolutional Neural Networks*. Advances in Neural Information Processing Systems (NIPS).
4. Simonyan, K., & Zisserman, A. (2014). *Very Deep Convolutional Networks for Large-Scale Image Recognition*. arXiv preprint arXiv:1409.1556.
5. He, K., Zhang, X., Ren, S., & Sun, J. (2016). *Deep Residual Learning for Image Recognition*. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).