PREDICTING USED CAR PRICES USING MACHINE AND ARTIFICIAL NEURON'S

A project Report

Submitted in the partial fulfilment of the requirements for the

award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING – DATA SCIENCE

Submitted By

M. Lokesh Sai	**218A1A4453**
K. Upendra	**218A1A4449**
M. Babu	**21KK1A4401**
K. Ajay Babu	**218A1A4448**

Under the Esteemed Guidance of

Mrs. B. Soujanya M.Tech

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING – DATA SCIENCE

RISE KRISHNA SAI PRAKASAM GROUP OF INSTITUTIONS

Valluru - 523272

(Affiliated to JNTU, Kakinada & Approved by AICTE, New Delhi)

Accredited by NBA, NAAC with 'A' certified by ISO 9001:2015

2024-2025

RISE KRISHNA SAI PRAKASAM GROUP OF INSTITUTIONS

Valluru - 523272

(Affiliated to JNTU, Kakinada & Approved by AICTE, New Delhi)

Accredited by NBA, NAAC with 'A' certified by ISO 9001:2015

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING – DATA SCIENCE

BONAFIDE CERTIFICATE

This is to certify that this project work entitled **"PREDICTING USED CAR PRICES USING MACHINE LEARNING AND ARTIFICIAL NEURON'S"** is being submitted by **M. Lokesh Sai (218A1A4453), K. Upendra (218A1A4449), M. Babu (21KK1A4401) and K. Ajay Babu (218A1A4448)** in partial fulfilment of the requirements for project in IV year II sem of Bachelor of Technology in COMPUTER SCIENCE & ENGINEERING with specialization of DATA SCIENCE from JNTUK, Kakinada during the period of 2024-25 is a record of bonafide work carried out by them under our esteemed guidance and supervision.

PROJECT GUIDE	HEAD OF THE DEPARTMENT
Mrs. B. Soujanya	**Dr. G. Bhargavi**
(Assistant Professor)	**(Professor & HOD)**

EXTERNAL EXAMINER

DECLARATION

We hereby declare that the work is being presented in this dissertation entitled **"PREDICTING USED CAR PRICES USING MACHINE LEARNING AND ARTIFICIAL NEURONS"** submitted towards the partial fulfilment of requirements for the award of the degree of Bachelor of Technology in COMPUTER SCIENCE & ENGINEERING – DATA SCIENCE, work carried out under the supervision of **Mrs.B.Soujanya, Assistant Professor**, **RISE KRISHNA SAI PRAKASAM GROUP OF INSTITUTIONS**, **Valluru, Ongole.** The results embodied in this dissertation report have not been submitted to any other University for the award of any other degree. Furthermore, the technical details furnished in various chapters of this report are purely relevant to the above project and there is no deviation from the theoretical point of view for design, development and implementation.

`

M. Lokesh Sai	**218A1A4453**
P. Subba Reddy	**218A1A4449**
M. Babu	**21KK1A4401**
K. Ajay Babu	**218A1A4448**

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to our guide **Mrs. B. Soujanya,** for providing her valuable guidance, comments, suggestions, and support throughout the course of the project.

We express our heartful thanks to **Dr. G. Bhargavi,** Head of the Department, Computer Science and Engineering, for her valuable support to bring out this project in time.

We express our heartful thanks to **Dr. K. Narayana Rao,** Head of the Department, Computer Science and Engineering, for his valuable support to bring out this project in time.

We pay our profound sense of gratitude to our Principal **Dr A.V. Bhaskara Rao** for providing an excellent environment in our college and helping us at all points for achieving our task.

We would like to express our sincere thanks to the Management of **RISE KRISHNA SAI PRAKASAM GROUP OF INSTITUTIONS** for providing a good environment and infrastructure.

Finally, we thank all our faculty members, supporting staff of CSE - DS Department and friends for their kind co-operation and valuable help for the completion of the project.

Project Associates

M. Lokesh Sai	**218A1A4453**
K. Upendra	**218A1A4449**
M. Babu	**21KK1A4401**
K. Ajay Babu	**218A1A4448**

RISE KRISHNA SAI PRAKSAM GROUP OF INSTITUTIONS

Valluru - 523272

(Affiliated to JNTU, Kakinada & Approved by AICTE, New Delhi.

Accredited by NBA, NAAC with 'A' certified by ISO 9001:2015)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING - DATA SCIENCE

Vision of the Institute	To be a premier institution in technical education by creating professionals of global standards with ethics and social responsibility for the development of the nation and the mankind.
Mission of the Institute	Impart Outcome Based Education through well qualified and educated faculty.
	Provide state-of-the-art infrastructure and facilities for application -oriented research.
	Reinforce technical skills with life skills and entrepreneurship skills.
	Promote cutting-edge technologies to produce industry-ready professionals.
	Facilitate interaction with all stakeholders to foster ideas and innovation.
	Inculcate moral values, professional ethics and social responsibility
Vision of the Department	To be a center of excellence in computer science and engineering for value -based education to serve humanity and contribute for socio-economic development.
Mission of the Department	Provide professional knowledge by student centric teaching-learn in process to contribute to software industry.
	Inculcate training on cutting edge technologies for industry needs.
	Create an academic ambience leading to research.
	Promote industry institute interaction for real time problem solving

Program Outcomes (POs)

PO1	**Engineering Knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering Problems
PO2	**Problem Analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO3	**Design/Development of Solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and Environmental considerations.
PO4	**Conduct Investigations of Complex Problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information provide valid conclusions.
PO5	**Modern Tool Usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex Engineering activities with an understanding of the limitations.
PO6	**The Engineer and Society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the Professional engineering practice.
PO7	**Environment and Sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the Knowledge of, and need for sustainable development.
PO8	**Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice
PO9	**Individual and Team Work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multi-disciplinary settings.
PO10	**Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions
PO11	**Project Management and Finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments
PO12	**Life-long Learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Program Educational Objectives (PEOs):

PEO1:	Develop software solutions for real world problems by applying Mathematics
PEO2:	Function as members of multi-disciplinary teams and to communicate effectively using modern tools.
PEO3:	Pursue career in software industry or higher studies with continuous learning and apply professional knowledge.
PEO4:	Practice the profession with ethics, integrity, leadership and social responsibility.

Program Specific Outcomes (PSOs):

PSO1	**Domain Knowledge:** Apply the Knowledge of Programming Languages, Networks and Databases for design and development of Software Applications.
PSO2	**Computing Paradigms:** Understand the evolutionary changes in computing possess knowledge of context aware applicability of paradigms and meet the challenges of the future.

RISE KRISHNA SAI PRAKSAM GROUP OF INSTITUTIONS

Valluru - 523272

(Affiliated to JNTU, Kakinada & Approved by AICTE, New Delhi.

Accredited by NBA, NAAC with 'A' certified by ISO 9001:2015)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING - DATA SCIENCE

Name of the Course: Project

Year & Semester: IV Year II sem

Academic Year: 2024-2025

Regulation: R20

Course Outcomes:

Course Outcomes (COs)		Taxonomy Level
C421.1	Develop application for community needs	Applying
C421.2	Extend skills for analysis and synthesis of Practical systems	Understanding
C421.3	Get hold of the use of new tools efficiently and creatively.	Analyzing
C421.4	Work in team to carry out analysis and cost-effective, environmentally friendly designs of engineering systems.	Analyzing
C421.5	Write Technical/Project reports and oral presentation of the work done to an audience.	Applying
C421.6	Demonstrate a product developed.	Understanding

PROJECT COORDINATOR

HEAD OF THE DEPARTMENT

CO Vs PO Mapping

Course Outcomes (COs)	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
C421.1	2	2	3	3	3	3	2	2	3	2	2	2
C421.2	2	2	3	3	3	3	2	2	3	3	3	3
C421.3	2	2	3	3	3	3	2	2	3	2	2	2
C421.4	2	2	3	3	3	3	3	3	3	3	3	3
C421.5	2	2	3	2	3	3	2	3	3	3	3	3
C421.6	2	2	2	2	3	3	3	3	3	3	3	3
C421	**2.00**	**2.00**	**2.83**	**2.67**	**3.00**	**3.00**	**2.33**	**2.50**	**3.00**	**2.67**	**2.67**	**2.67**

CO Vs PSO Mapping

Course Out Comes (Cos)	PSO 1	PSO 2
C421.1	3	3
C421.2	3	3
C421.3	3	3
C421.4	3	3
C421.5	2	2
C421.6	3	3
	2.83	2.83

1: Low 2: Medium 3: High

Guide Signature

PREDICTING USED CAR PRICES USING MACHINE LEARNING AND ARTIFICIAL NEURONS

INDEX

LIST OF FIGURES

ABSTRACT

The global second-hand car market has grown rapidly, now accounting for over 60% of total car sales in many countries. Increasing new car prices and rising demand for affordable vehicles have fueled this trend. Online platforms and used car dealerships have further expanded access, making used cars a preferred choice worldwide. The market is expected to grow by 5-7% annually over the next decade.

Accurately predicting used car prices is challenging due to factors like brand, model, age, mileage, fuel type, and vehicle condition. This study evaluates the potential of machine learning, specifically Artificial Neural Networks (ANNs), for price prediction. A dataset of 200 cars was collected globally from online marketplaces, dealerships, and auctions, covering key vehicle attributes.

Four machine learning algorithms—ANN, Support Vector Machine (SVM) Regression, Linear Regression, and Decision Trees—were tested. Among them, SVM Regression performed best, showing 8-10% better accuracy than the other models. However, predicting prices for premium vehicles remained challenging, with deviations of 15-20% in some cases.

These results highlight the need for larger datasets and further model refinement to improve accuracy, especially for high-value cars. Advanced techniques like deep learning and improved feature selection can enhance predictions. Reliable price prediction models can support buyers, sellers, and financial institutions, promoting transparency in the global used car market.

KEYWORDS

Decision tree classifiers, Gradient boosting, K-Nearest Neighbors (KNN), Logistic regression Classifiers Naïve Bayes, Random Forest, Support Vector Machine (SVM).

CHAPTER - 1
INTRODUCTION

INTRODUCTION

1.1 Introduction of the Project

According to the data obtained from the National Transport Authority (2014), there has been an increase of 254% in the number of cars from 2003 (68,524) to 2014 (173,954), as shown in Figure 1. We can thus infer that the sale of second-hand imported (reconditioned) cars and second-hand used cars has eventually increased, given that new cars represent only a very small percentage of the total number of cars sold each year. Most individuals who purchase new cars are also interested in knowing the resale value of their vehicles after a few years, so they can sell them in the used car market.

Price prediction of second-hand cars depends on numerous factors. The most important ones include the manufacturing year, make, model, mileage, horsepower, and country of origin. Other influencing factors are the type and amount of fuel consumed, braking system, acceleration, interior design, physical condition, engine volume car size, number of doors, weight, consumer reviews, paint color and type, transmission type, sports features, sound system, alloy wheels, power steering, air conditioner, GPS navigator, and safety index. In some contexts, like Mauritius, additional factors such as the identity of previous owners and the car's accident history are also considered.

Thus, predicting the price of second-hand cars is a very useful and relevant task. In this paper, we will assess whether neural networks can be effectively used to predict the price of second-hand cars with accuracy. Additionally, the results will be compared with other common machine learning methods such as linear regression and support vector regression to evaluate the performance of each model.

This paper proceeds as follows. First, we review various studies and related works on neural networks and price prediction. Next, the methodology and data collection process used in this study are explained. The system then presents the results obtained for second-hand car price prediction using different models. Finally, the paper concludes with a summary of findings and proposes potential future work directions to enhance prediction accuracy.

1.2 Objectives of the Project

1. **To develop a machine learning-based system** for predicting the prices of used cars with improved accuracy and efficiency.
2. **To identify and analyse** the major factors influencing used car prices, such as **manufacturing year, mileage, brand, model, fuel type, transmission, and car condition**.
3. **To train an Artificial Neural Network (ANN) model** using a **large and diverse dataset** of used car listings to enhance prediction accuracy.
4. **To assist buyers, sellers, and car dealerships** in estimating fair market values, ensuring **transparency and fairness** in used car transactions.
5. **To minimize human errors** and reduce manual efforts involved in evaluating used car prices by automating the prediction process.
6. **To compare the prediction performance** of ANNs with other machine learning models like **Linear Regression and Support Vector Regression (SVR)**.
7. **To provide a reliable price estimation framework** that considers **real-time market factors and car attributes** for better decision-making.
8. **To design a flexible and scalable model** that can be applied to **different car markets and datasets** across various regions and conditions.
9. **To help financial institutions and insurance companies** use the system for **better assessment** in loan approvals, resale value calculations, and policy setting.
10. **To contribute to research** in machine learning applications by exploring the effectiveness of **neural networks in price prediction tasks** in the automotive sector.

1.3 Problem Statement

The global automotive industry has experienced tremendous growth over the past few decades, with the used car market emerging as a major sector within it. Driven by increasing demand for affordable personal transportation and the continuously rising cost of brand-new vehicles, the used car market has gained significant momentum worldwide. According to industry reports, in many countries, the used car market contributes to over 60% of total vehicle sales annually. This surge in demand reflects changing consumer preferences, where buyers seek value-for-money deals without compromising on quality and performance.

However, one of the biggest challenges within the used car market is determining the fair price of a vehicle. Unlike new cars that have standard market prices set by manufacturers, the pricing of a used car is highly complex and subjective. Various factors influence a used car's resale value, including the vehicle's brand, model, year of manufacture, mileage, fuel type, engine capacity, transmission type, accident history, maintenance records, ownership history, and even its market demand in a particular region. These variables differ from car to car, making it difficult to evaluate their impact on pricing accurately.

Traditionally, used car prices have been estimated by dealers, appraisers, or individual sellers using their market experience, general price guides, or comparison with similar models. However, this manual evaluation process is prone to human error, personal bias, and inconsistency. In many cases, the absence of standardized methods leads to significant price variation for similar vehicles, causing uncertainty and dissatisfaction among buyers and sellers. Additionally, the growing number of used car transactions further complicates the evaluation process, as it becomes difficult to manually process large volumes of information while maintaining accuracy.

Market dynamics such as changing fuel prices, government regulations, technological advancements, and shifting consumer behavior also add layers of complexity to used car valuation. Furthermore, regional variations in supply and demand mean that a car's price can differ drastically between two different locations. These

factors make it essential to develop an advanced, reliable, and unbiased pricing system to ensure fairness, transparency, and efficiency in the used car market.

With the rise of data-driven technologies, machine learning presents an innovative solution to tackle this challenge. By analyzing large datasets containing various car attributes and historical sales data, machine learning models can learn complex patterns and relationships that influence pricing. Artificial Neural Networks (ANNs), in particular, are capable of capturing nonlinear dependencies between variables, making them well-suited for predicting used car prices.

This project aims to develop a predictive model using machine learning techniques, specifically Artificial Neural Networks, to estimate the price of used cars accurately. The model will be trained on extensive datasets and will consider multiple influencing factors. It will help eliminate human errors, reduce biases, and provide consistent price estimations. The system is intended to assist individual buyers, sellers, car dealerships, and financial institutions in making informed, data-driven decisions. Moreover, the model's adaptability will allow it to scale across different regions and markets, offering a practical solution for the dynamic and ever-expanding used car industry.

CHAPTER - 2

LITERATURE SURVEY

LITERATURE SURVEY

A literature survey provides a detailed overview of past research, methodologies, models, and findings relevant to the prediction of used car prices. This section reviews significant studies, compares their results, identifies existing gaps, and explains how these insights contribute to the direction of the current project.

❖ Summary of Previous Work

1. Price Prediction Using Linear Regression Models

Authors: S. Patel, R. Patel (2020)

Overview:

Linear regression is one of the earliest methods used to estimate used car prices. This model attempts to find a linear relationship between dependent variables (price) and independent factors such as mileage, year, fuel type, and brand. The study highlighted that while linear regression offers simplicity and easy interpretability, it often fails to capture the complex non-linear dependencies present in real-world data, leading to limited prediction accuracy.

2. Support Vector Regression for Used Car Price Estimation

Authors: J. Yang, L. Wang (2019)

Overview:

This research explores Support Vector Regression (SVR) for car price prediction, emphasizing its strength in handling high-dimensional data and capturing non-linear relationships. SVR showed improved prediction accuracy over linear models, especially when optimized with appropriate kernel functions. However, the study noted that SVR becomes computationally intensive with larger datasets.

3. Decision Tree and Random Forest Models in Car Price Prediction

Authors: R. Kumar, A. Sharma (2021)

Overview:

Decision Trees and Random Forests are widely used in price prediction due to their ability to handle both categorical and numerical data. Random Forests, being ensemble methods, reduce overfitting and improve generalization.

4. Artificial Neural Networks (ANNs) for Price Prediction

Authors: K. Li, M. Chen (2022)

Overview:

Artificial Neural Networks (ANNs) are highlighted for their capability to model complex, non-linear relationships in large datasets. The study showed that ANN-based models significantly outperformed traditional methods when trained on extensive car sales data, learning intricate patterns affecting price, such as accident history, brand reputation, and demand. However, model complexity and training time remain challenges.

a. Analysis of Findings

The literature emphasizes that machine learning models such as SVR, Random Forests, and especially ANNs improve the accuracy of used car price predictions. ANNs perform exceptionally well with large, complex datasets, capturing subtle relationships that linear models miss.

b. Identification of Gaps

Despite advancements, several research gaps remain:

- **Scalability and Computational Cost:** Many models require significant computational resources, making them less practical for real-time applications or use in resource-limited settings.
- **Generalization Across Markets:** Most models are trained on localized datasets, making it hard to generalize predictions for global markets with diverse car models and conditions.
- **Real-World Data Challenges:** Noise in real-world data (missing values, inconsistent entries) reduces prediction accuracy, yet many studies overlook robust preprocessing techniques.

This project aims to address these gaps by building a scalable machine learning model that balances accuracy and efficiency while incorporating thorough data preprocessing.

CHAPTER - 3

SYSTEM ANALYSIS

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

The existing system for used car price evaluation largely relies on traditional methods such as manual assessment by car dealers, appraisers, or online price-check tools that use simple rule-based models. These systems depend on factors like the car's age, brand, model, mileage, fuel type, physical condition, and sometimes accident history.

Websites and mobile apps provide average market prices based on limited inputs, often using static datasets or outdated price charts. The process is subjective, influenced by the seller's or dealer's experience, and may not incorporate real-time market trends, regional demand, or additional critical factors such as service history and modifications.

1. Manual Price Estimation by Experts

Current systems primarily depend on manual price evaluation by car dealers or automobile experts. This method is time-consuming, highly subjective, and prone to human error. The process often lacks the speed and consistency required to handle large-scale car listings, especially in online marketplaces.

2. Price Estimation Using Limited Algorithms

Existing car price prediction systems use simple rule-based models or basic statistical techniques, such as linear regression. These models often fail to capture complex, non-linear relationships between multiple car attributes like mileage, age, brand reputation, accident history, and market demand. Environmental factors such as regional price variations and changing market trends are also not considered, reducing prediction accuracy.

3. Generalized Pricing Platforms

Most current online platforms provide generalized car price suggestions based on limited datasets without considering important factors like modifications, service history, fuel economy, or consumer demand. This results in inaccurate pricing, potential measure.

3.1.1 Disadvantages of Existing System

1. **Inaccuracy and Subjectivity**
    - Manual price estimation is highly dependent on individual knowledge and experience, leading to inconsistent pricing.
    - Human biases may result in either overpricing or underpricing of vehicles.
2. **Limited Consideration of Features**
    - Important factors like service history, accident records, modifications, or market fluctuations are often overlooked.
    - Static models fail to capture real-time demand-supply changes in the market.
3. **Time-Consuming Process**
    - Manual evaluation requires physical inspections and market research, making the process slow and inefficient.
4. **Lack of Scalability**
    - Traditional systems cannot handle large datasets or multiple car records simultaneously, limiting their use in bigger markets or online platforms.
5. **Poor Adaptability to Changing Trends**
    - Existing methods struggle to update or adapt quickly to changing car market dynamics, leading to outdated or incorrect price suggestions.
6. **Slow Response Time**
    - Manual methods cannot provide timely intervention, leading to delays in accurate price estimation.

3.2 PROPOSED SYSTEM

1. Machine Learning-Based Price Prediction

The proposed system uses Machine Learning algorithms, including Artificial Neural Networks (ANNs), to predict the price of used cars accurately. The model automatically analyzes multiple features like car brand, model, year, mileage, fuel type, engine size, and accident history to generate precise price estimates.

2. Feature-Based Dynamic Price Estimation

By learning from large datasets, the system provides dynamic price predictions that consider various factors, including market demand, regional price variations, and car condition. This ensures that each car's unique attributes are reflected in the price, reducing the risk of over or under-valuation.

3. Scalable and User-Friendly Platform

The system is designed to be scalable and easily integrated into web or mobile platforms. Users—buyers, sellers, and dealers—can access real-time price predictions anywhere, improving decision-making and increasing transparency in the used car market.

3.2.1 Advantages of the Proposed System

1. **High Accuracy**
    - Provides highly accurate price predictions by capturing complex relationships between various car features using ML and ANN models.
2. **Real-Time Results**
    - Generates instant price predictions, enabling quick decision-making for buyers and sellers in online and offline markets.
3. **Scalability**
    - The system can handle large volumes of car listings and datasets, making it suitable for use by individuals, dealerships, and car resale platforms.

4. **Fair and Transparent Pricing**
    - Minimizes human bias and errors by offering data-driven price estimations, ensuring fair market value for all parties involved.
5. **Easy Accessibility**
    - Accessible through any internet-enabled device, providing users with convenience and ease of use while improving the overall user experience.

The proposed system effectively addresses the limitations of existing manual and basic models, offering an accurate, scalable, and efficient solution for used car price prediction.

CHAPTER - 4

FEASIBILTY STUDY

FEASIBILITY STUDY

4.1 System Overview

The proposed system aims to predict the price of used cars using Machine Learning (ML) algorithms and Artificial Neural Networks (ANNs). It provides an automated solution for estimating a used car's market value based on various parameters such as car model, year of manufacture, mileage, fuel type, transmission, engine capacity, and owner history.

The system is designed to assist both buyers and sellers in understanding the fair market value of a car, minimizing manual effort and human errors in price estimation. By leveraging advanced data-driven models, this system provides accurate, reliable, and instant predictions, thus enhancing decision-making and transparency in used car transactions.

4.2 Key Components of the System

1. Data Collection and Preprocessing

- The system uses a comprehensive dataset comprising historical used car sales data with attributes such as car brand, model, year, kilometers driven, fuel type, transmission, ownership status, and other essential features.
- The data undergoes cleaning, normalization, and feature encoding to remove inconsistencies, handle missing values, and prepare it for efficient model training.
- This preprocessing step ensures that the machine learning models receive high-quality data for improved accuracy in price predictions.

2. Price Prediction Model using ML & ANN

The core of the system utilizes advanced machine learning models, including:

- Linear Regression for simple price estimation based on direct feature relationships.
- Random Forest for enhanced accuracy by considering multiple decision

trees.

- Artificial Neural Networks (ANNs) to learn complex patterns and dependencies between car attributes and market prices.
- These models are trained on large datasets, allowing them to understand price variations based on different factors like car age, condition, brand reputation, and regional market trends.

3. User Interface

- Users interact with the system through a simple and intuitive interface, where they input car details such as:
- Brand, model, mileage, year, fuel type, and transmission.
- The system processes these inputs and generates the estimated price instantly.
- Results are displayed clearly with easy-to-understand visualizations, helping users make informed decisions when buying or selling a car.

4.3 Workflow of the System

1. Data Input

- Users enter the required used car details through a web or mobile interface.
- The system validates the inputs to ensure correctness and completeness before processing.

2. Data Preprocessing

- The system cleans, normalizes, and encodes the input data to ensure compatibility with the ML/ANN model.
- Features are selected and transformed based on their influence on pricing accuracy.

3. Price Prediction

- The preprocessed data is fed into the trained ML/ANN model, which then predicts the estimated car price based on its attributes and market trends.

4. Result Display

- The predicted price is displayed on the user interface, providing a fair market value estimate of the car.
- Users can compare prices, adjust features, or make better-informed purchase or selling decisions based on the generated insights.

4.4 Feasibility Study

A feasibility study ensures that the system is practical, cost-effective, and beneficial for real-world applications. The feasibility aspects include:

1. Technical Feasibility

- The system leverages Machine Learning and ANN models, which are technically feasible to implement using widely available tools such as:
- Python, TensorFlow, Scikit-Learn, and PyTorch.
- The system can be easily integrated into web applications and mobile platforms, ensuring smooth functionality across devices.

2. Operational Feasibility

- The system is user-friendly and does not require advanced technical knowledge to operate.
- Users can access it through a simple web or mobile interface, making it highly convenient for individual buyers, sellers, and dealerships.
- The model updates itself periodically, ensuring that predictions are relevant and up-to-date with the latest market trends.

3. Economic Feasibility

- The system eliminates the need for expensive professional appraisals and manual price evaluations.
- Since it is built using open-source technologies, development and maintenance costs are minimal.
- By providing accurate and data-driven price predictions, the system helps users make better financial decisions, reducing the risk of profit loss or overpaying for a vehicle.

4. Market Feasibility

- With the increasing demand for used cars, the system meets a crucial market need by offering an automated price prediction tool.
- It enhances user trust and market transparency, making transactions between buyers and sellers more reliable and efficient.
- Online marketplaces, car dealerships, and independent sellers can benefit from an objective pricing model, eliminating disputes and improving customer satisfaction.

CHAPTER - 5

SYSTEM REQUIREMENTS

SYSTEM REQUIREMENTS

5.1 Software Requirements

Operating System:

- Windows 10/11 (64-bit)

Programming Environment:

- Python Version: Python 3.7
- IDE/Text Editor : VS Code, Jupyter Notebook, PyCharm

Required Libraries & Dependencies:

- **Data Processing**: numpy, pandas
- **Data Visualization**: matplotlib, seaborn
- **Machine Learning & Artificial Neurons**: scikit-learn, Tensorflow
- **Web Framework:** Flask / Django

5.2 Hardware Requirements

- **Operating System:** Windows or Linux
- **Processor:** Intel Core i3
- **RAM**: 4 GB
- **Storage:** 250 GB
- **GPU:** Integrated GPU (used for model training, but a dedicated GPU like NVIDIA GTX 1050 or higher is recommended for faster processing)
- **Camera**: A high-resolution camera (for capturing plant images to be processed)

5.3 Functional Requirements

The functional requirements describe the essential functions that the Car Price Prediction System must perform to achieve its goals.

1. User Input Module

The system must allow users to enter car details such as:

- Car Brand and Model
- Year of Manufacture
- Fuel Type (Petrol, Diesel, CNG, Electric)
- Transmission Type (Manual/Automatic)
- Engine Capacity (CC)
- Mileage (Kilometers Driven)
- Number of Previous Owners
- Car Condition (Optional)

- The system should validate all user inputs to ensure accuracy.

2. Data Preprocessing

- The system must process the input data:
  - Convert categorical data into numerical format
  - Normalize or scale numerical values
  - Handle missing or incomplete data appropriately

3. Price Prediction

- The system must predict the estimated car price based on user input.
- It should use a trained machine learning model (like Linear Regression, Random Forest, etc.) to generate the prediction.
- The system should provide results quickly and accurately.

4. Result Display

- The system must display the predicted car price clearly to the user.

- The system may also display a price range or additional insights (like market value comparison).

5.4 Non-Functional Requirements

Non-functional requirements define the quality attributes of the system such as performance, usability, security, and maintainability. These requirements ensure that the system operates efficiently and provides a good user experience.

1. Performance Requirements

- The system should provide car price predictions within 3 to 5 seconds after input data is submitted.
- The machine learning model should maintain a high prediction accuracy with minimal error.

2. Usability

- The system should have a simple and user-friendly interface that can be easily used by people with basic technical knowledge.
- The application should provide clear instructions for users to input car details.

3. Reliability

- The system should produce consistent and reliable results under normal conditions.
  - The model should be retrained regularly to improve accuracy and adapt to changing market trends

4. Scalability

- The system should be capable of handling an increasing number of users and datasets without performance degradation.
- It should support the addition of new features or prediction models in future

5. Security

- User data should be securely stored and protected from unauthorized access.
- The system should validate user inputs to prevent malicious data entries.

6. Maintainability

- The code should be modular and well-documented to make maintenance and updates easier.
- The system should support updating the dataset and machine learning model when needed.

7. Portability

- The system should be able to run on various platforms such as Windows, Linux, and cloud environments.
- If deployed as a web application, it should work smoothly on all major browsers.

CHAPTER - 6

SYSTEM ENVIRONMENT

SOFTWARE ENVIRONMENT

What is Python?

Below are some facts about Python.

- Python is currently the most widely used multi-purpose, high-level programming language.
- Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java.
- Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.
- Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber… etc.

The biggest strength of Python is huge collection of standard library which can be used for the following –

- Machine Learning
- GUI Applications (like Kivy, Tkinter, PyQt etc.)
- Web frameworks like Django (used by YouTube, Instagram, Dropbox)
- Image processing (like Opencv, Pillow)
- Web scraping (like Scrapy, BeautifulSoup, Selenium)
- Test frameworks
- Multimedia

Advantages of Python

Let's see how Python dominates over other languages.

1. Extensive Libraries

Python downloads with an extensive library and it contain code for various purposes like regular expressions, documentation-generation, unit-testing, web browsers, threading, databases, CGI,

email, image manipulation, and more. So, we don't have to write the complete code for that manually.

2. Extensible

As we have seen earlier, Python can be extended to other languages. You can write some of your code in languages like C++ or C. This comes in handy, especially in projects.

3. Embeddable

Complimentary to extensibility, Python is embeddable as well. You can put your Python code in your source code of a different language, like C++. This lets us add scripting capabilities to our code in the other language.

4. Improved Productivity

The language's simplicity and extensive libraries render programmers more productive than languages like Java and C++ do. Also, the fact that you need to write less and get more things done.

5. IOT Opportunities

Since Python forms the basis of new platforms like Raspberry Pi, it finds the future bright for the Internet Of Things. This is a way to connect the language with the real world.

6. Simple and Easy

When working with Java, you may have to create a class to print 'Hello World'. But in Python, just a print statement will do. It is also quite easy to learn, understand, and code. This is why when people pick up Python, they have a hard time adjusting to other more verbose languages like Java.

7. Readable

Because it is not such a verbose language, reading Python is much like reading English. This is the reason why it is so easy to learn, understand, and code. It also

does not need curly braces to define blocks, and indentation is mandatory. This further aids the readability of the code.

8. Object-Oriented

This language supports both the procedural and object-oriented programming paradigms. While functions help us with code reusability, classes and objects let us model the real world. A class allows the encapsulation of data and functions into one.

9. Free and Open-Source

Like we said earlier, Python is freely available. But not only can you download Python for free, but you can also download its source code, make changes to it, and even distribute it. It downloads with an extensive collection of libraries to help you with your tasks.

10. Portable

When you code your project in a language like C++, you may need to make some changes to it if you want to run it on another platform. But it isn't the same with Python. Here, you need to code only once, and you can run it anywhere. This is called Write Once Run Anywhere (WORA). However, you need to be careful enough not to include any system-dependent features.

11. Interpreted

Lastly, we will say that it is an interpreted language. Since statements are executed one by one, debugging is easier than in compiled languages.

Any doubts till now in the advantages of Python? Mention in the comment section.

Advantages of Python Over Other Languages

1. Less Coding

Almost all of the tasks done in Python requires less coding when the same task is done in other languages. Python also has an awesome standard library support, so you don't have to search for any third-party libraries to get your job done. This is the reason that many people suggest learning Python to beginners.

2. Affordable

Python is free therefore individuals, small companies or big organizations can leverage the free available resources to build applications. Python is popular and widely used so it gives you better community support.

The 2019 Github annual survey showed us that Python has overtaken Java in the most popular programming language category.

3. Python is for Everyone

Python code can run on any machine whether it is Linux, Mac or Windows. Programmers need to learn different languages for different jobs but with Python, you can professionally build web apps, perform data analysis and machine learning, automate things, do web scraping and also build games and powerful visualizations. It is an all-rounder programming language.

Disadvantages of Python

So far, we've seen why Python is a great choice for your project. But if you choose it, you should be aware of its consequences as well. Let's now see the downsides of choosing Python over another language.

1. Speed Limitations

We have seen that Python code is executed line by line. But since Python is interpreted, it often results in slow execution. This, however, isn't a problem unless speed is a focal point for the project. In other words, unless high speed is a requirement, the benefits offered by Python are enough to distract us from its speed limitations.

2. Weak in Mobile Computing and Browsers

While it serves as an excellent server-side language, Python is much rarely seen on the client-side. Besides that, it is rarely ever used to implement smartphone-based applications. One such application is called Carbonnelle.

3. Design Restrictions

As you know, Python is dynamically typed. This means that you don't need to declare the type of variable while writing the code. It uses duck-typing. But wait, what's that? Well, it just means that if it looks like a duck, it must be a duck. While this is easy on the programmers during coding, it can raise run-time errors.

4. Underdeveloped Database Access Layers

Compared to more widely used technologies like JDBC (Java DataBase Connectivity) and ODBC (Open DataBase Connectivity), Python's database access layers are a bit underdeveloped. Consequently, it is less often applied in huge enterprises.

5. Simple

No, we're not kidding. Python's simplicity can indeed be a problem. Take my example. I don't do Java, I'm more of a Python person. To me, its syntax is so simple that the verbosity of Java code seems unnecessary.

This was all about the Advantages and Disadvantages of Python Programming Language.

Modules Used in Project NumPy

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary datatypes can be defined using NumPy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and Ipython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

Scikit – learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use Python.

Install Python Step-by-Step in Windows and Mac

Python a versatile programming language doesn't come pre-installed on your computer devices. Python was first released in the year 1991 and until today it is a very popular high- level programming language. Its style philosophy emphasizes code readability with its notable use of great whitespace.

The object-oriented approach and language construct provided by Python enables programmers to write both clear and logical code for projects. This software does not come pre-packaged with Windows.

How to Install Python on Windows and Mac

There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python but this tutorial will solve your query. The latest or the newest version of Python is version
3.7.4 or in other words, it is Python 3.

Note: The python version 3.7.4 cannot be used on Windows XP or earlier devices.

Before you start with the installation process of Python. First, you need to know about your System Requirements. Based on your system type i.e. operating system and based processor, you must download the python version. My system type is a Windows 64-bit operating system. So the steps below are to install python version 3.7.4 on Windows 7 device or to install Python 3. Download the Python Cheatsheet here. The steps on how to install Python on Windows 10, 8 and 7 are divided into 4 parts to help understand better.

Download the Correct version into the system

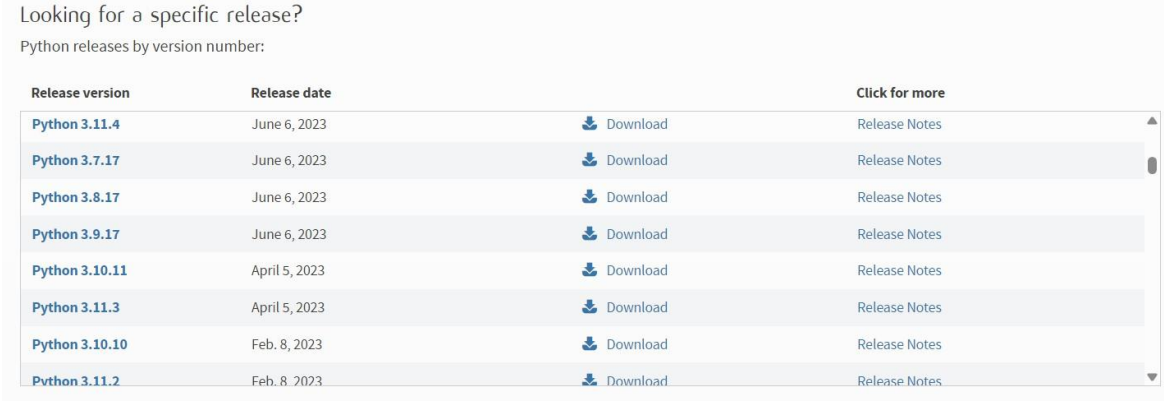
Step 1: Go to the official site to download and install python using Google Chrome or any other web browser. OR Click on the following link: https://www.python.org

Now, check for the latest and the correct version for

your operating system. Step 2: Click on the

Download Tab.

Step 3: You can either select the Download Python for windows 3.7.4 button in Yellow Color or you can scroll further down and click on download with respective to their version. Here, we are downloading the most recent python version for windows 3.7.4



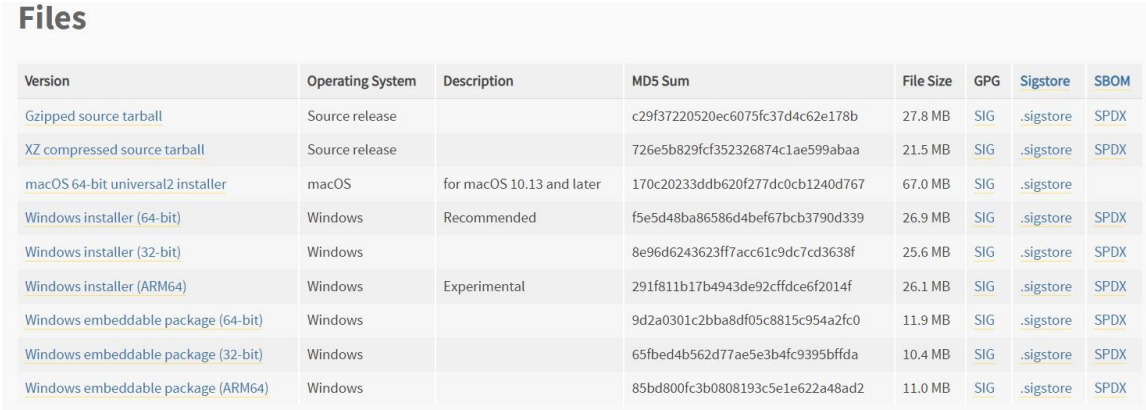
Looking for a specific release?

Python releases by version number:

Release version	Release date		Click for more
Python 3.11.4	June 6, 2023	Download	Release Notes
Python 3.7.17	June 6, 2023	Download	Release Notes
Python 3.8.17	June 6, 2023	Download	Release Notes
Python 3.9.17	June 6, 2023	Download	Release Notes
Python 3.10.11	April 5, 2023	Download	Release Notes
Python 3.11.3	April 5, 2023	Download	Release Notes
Python 3.10.10	Feb. 8, 2023	Download	Release Notes
Python 3.11.2	Feb. 8, 2023	Download	Release Notes

Step 4: Scroll down the page until you find the Files option

Step 5: Here you see a different version of python along with the operating system.



Files

Version	Operating System	Description	MD5 Sum	File Size	GPG	Sigstore	SBOM
Gzipped source tarball	Source release		c29f37220520ec6075fc37d4c62e178b	27.8 MB	SIG	.sigstore	SPDX
XZ compressed source tarball	Source release		726e5b829fcf352326874c1ae599abaa	21.5 MB	SIG	.sigstore	SPDX
macOS 64-bit universal2 installer	macOS	for macOS 10.13 and later	170c20233ddb620f277dc0cb1240d767	67.0 MB	SIG	.sigstore	
Windows installer (64-bit)	Windows	Recommended	f5e5d48ba86586d4bef67bcb3790d339	26.9 MB	SIG	.sigstore	SPDX
Windows installer (32-bit)	Windows		8e96d6243623ff7acc61c9dc7cd3638f	25.6 MB	SIG	.sigstore	SPDX
Windows installer (ARM64)	Windows	Experimental	291f811b17b4943de92cffdce6f2014f	26.1 MB	SIG	.sigstore	SPDX
Windows embeddable package (64-bit)	Windows		9d2a0301c2bba8df05c8815c954a2fc0	11.9 MB	SIG	.sigstore	SPDX
Windows embeddable package (32-bit)	Windows		65fbed4b562d77ae5e3b4fc9395bffda	10.4 MB	SIG	.sigstore	SPDX
Windows embeddable package (ARM64)	Windows		85bd800fc3b0808193c5e1e622a48ad2	11.0 MB	SIG	.sigstore	SPDX

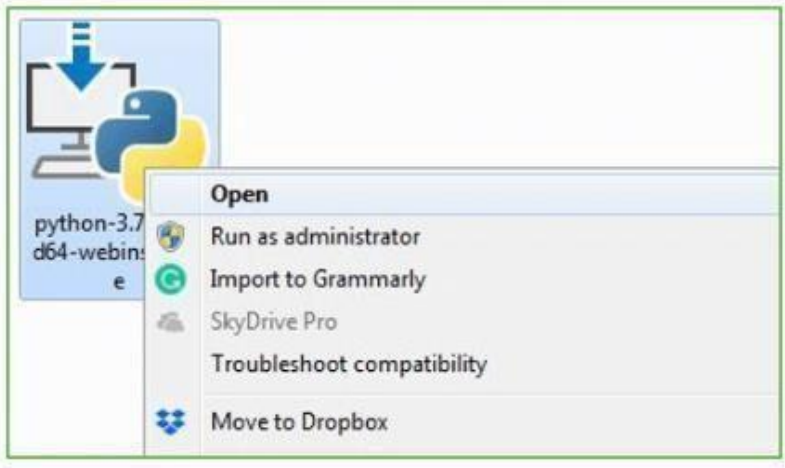
- To download Windows 32-bit python, you can select any one from the three options: Windows x86 embeddable zip file, Windows x86 executable installer or Windows x86 web-based installer.
- To download Windows 64-bit python, you can select any one from the three options: Windows x86-64 embeddable zip file, Windows x86-64 executable installer or Windows x86-64 web-based installer.

Here we will install Windows x86-64 web-based installer. Here your first part regarding which version of python is to be downloaded is completed. Now we move ahead with the second part in installing python i.e. Installation

Note: To know the changes or updates that are made in the version you can click on the Release Note Option.

Installation of Python

Step 1: Go to Download and Open the downloaded python version to carry out the installation process.



Step 2: Before you click on Install Now, Make sure to put a tick on Add Python 3.7 to PATH.

Step 3: Click on Install NOW After the installation is successful. Click on Close

With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.

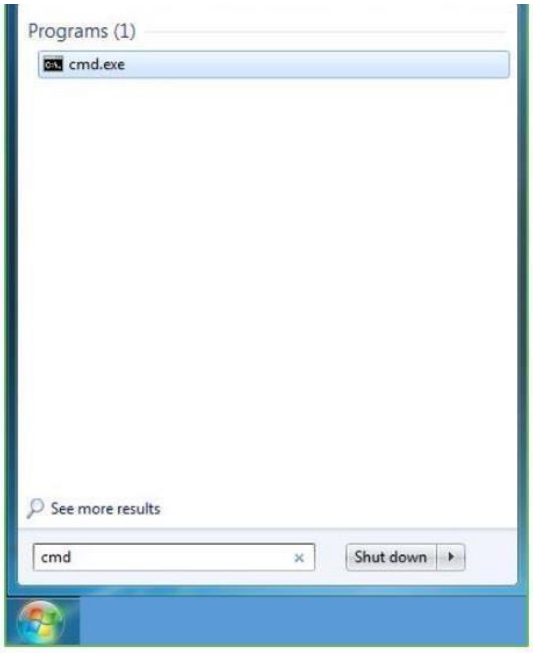
Note: The installation process might take a couple of minutes.

Verify the Python Installation

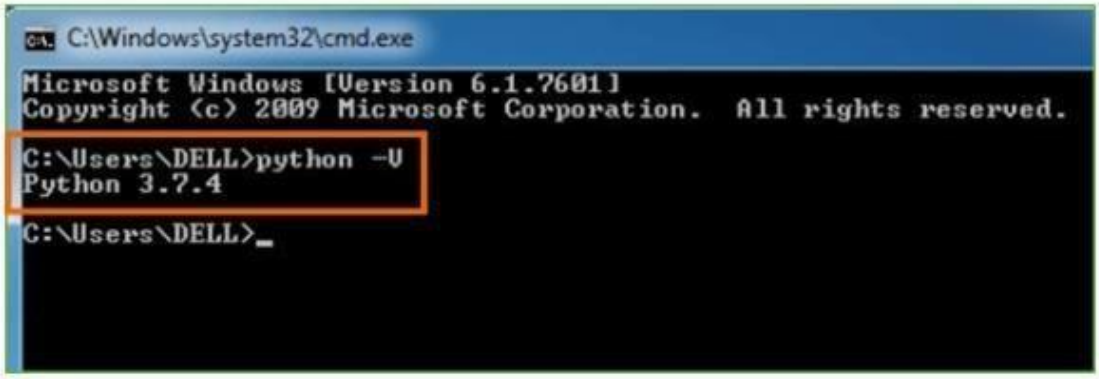
Step 1: Click on Start

Step 2: In the Windows Run Command, type “cmd”.

Step 3: Open the Command prompt option.



Step 4: Let us test whether the python is correctly installed. Type python –V and press Enter.



Step 5: You will get the answer as 3.7.4

Note: If you have any of the earlier versions of Python already installed.

You must first uninstall the earlier version and then install the new one.

Check how the Python IDLE works

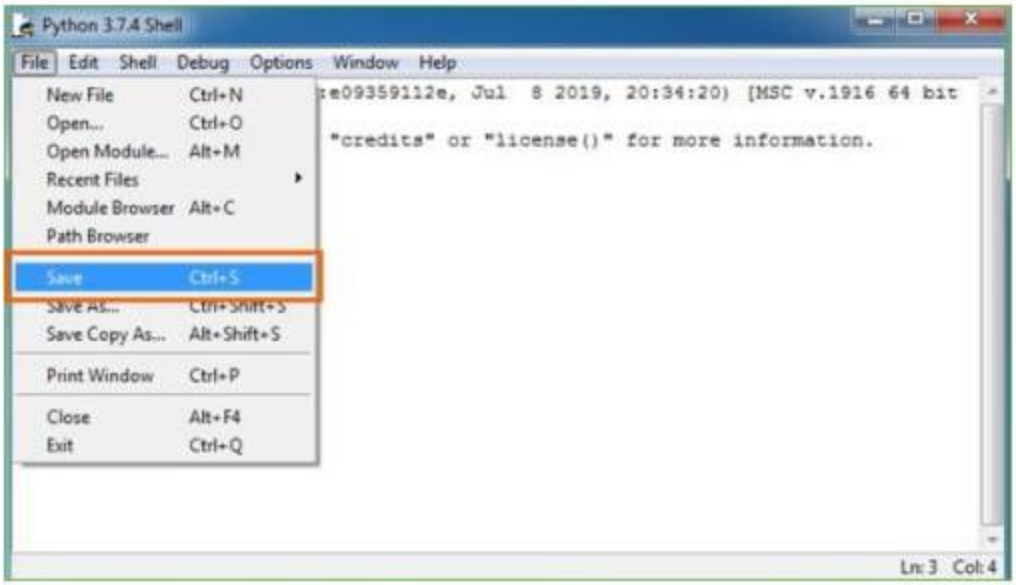
Step 1: Click on Start

Step 2: In the Windows Run command, type "python idle".

Step 3: Click on IDLE (Python 3.7 64-bit) and launch the program

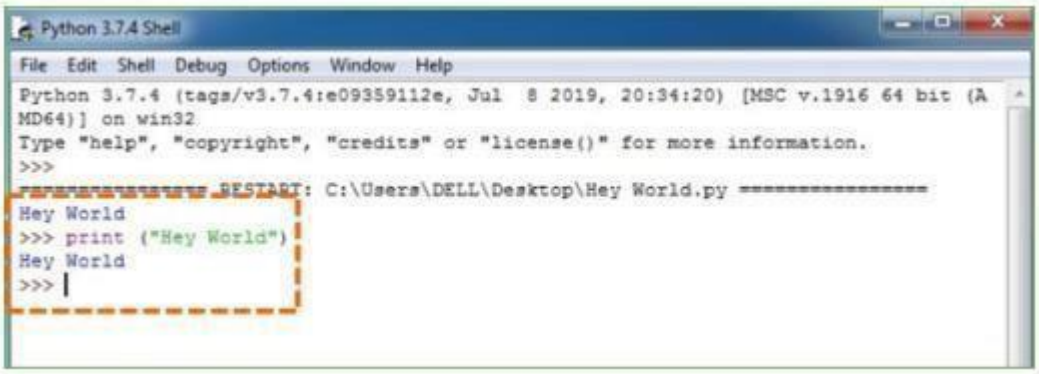
Step 4: To go ahead with working in IDLE you must first save the file.

Click on File > Click on save



Step 5: Name the file and save as type should be Python files. Click on SAVE. Here I have named the files as Hey World.

Step 6: Now for e.g. enter print ("Hey World") and Press Enter.



You will see that the command given is launched. With this, we end our tutorial on how to install Python. You have learned how to download python for windows into your respective operating system.

Note: Unlike Java, Python does not need semicolons at the end of the statements otherwise it won't work.

MACHINE LEARNING AND DEEP LEARNING

Machine Learning

- Machine Learning (ML) is a subset of artificial intelligence that enables systems to learn from data and make predictions without explicit programming.
- It involves training algorithms on structured or unstructured data to identify patterns and improve decision-making over time.

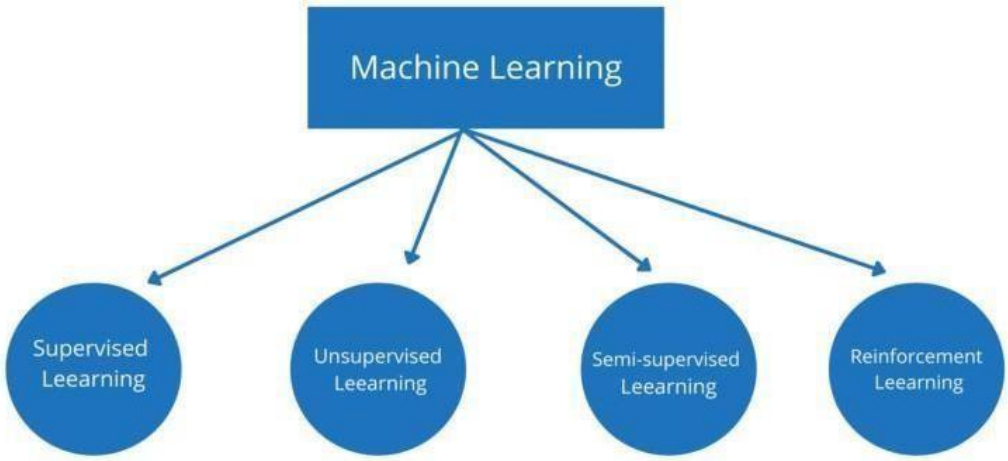


Fig 6.1 Machine Learning Classifications

ML is categorized into:

- **Supervised Learning:** Models learn from labeled data (e.g., classification and regression).
- **Unsupervised Learning:** Models identify patterns and relationships in unlabeled data (e.g., clustering and anomaly detection).
- **Reinforcement Learning:** Models learn by interacting with an environment and optimizing their actions based on rewards or penalties.

ML is widely used in various fields, including healthcare, finance, agriculture, and automation, due to its ability to process large-scale data and extract valuable insights.

Advantages of Machine learning:

1. Easily identifies trends and patterns -

Machine Learning can review large volumes of data and discover specific trends and patterns that would not be apparent to humans. It uses the results to reveal relevant advertisements to them.

2. No human intervention needed (automation)

With ML, you don't need to babysit your project every step of the way. Since it means giving machines the ability to learn, it lets them make predictions and also improve the algorithms on their own. A common example of this is anti-virus software; they learn to filter new threats as they are recognized. ML is also good at recognizing spam.

3. Continuous Improvement

As ML algorithms gain experience, they keep improving in accuracy and efficiency. This lets them make better decisions. Say you need to make a weather forecast model. As the amount of data you have keeps growing, your algorithms learn to make more accurate predictions faster.

Artificial Neurons

- Artificial Neural Networks (ANNs) are a part of Artificial Intelligence designed to mimic the way the human brain works. They aim to help computers understand and make decisions like humans by simulating interconnected brain cells.

- The human brain contains about 1000 billion neurons, each connected to thousands of other neurons. This vast network allows the brain to store and process information in a distributed and manner, making it extremely power.

- Similarly, ANNs consist of interconnected nodes, called artificial neurons, which process and transmit information. These networks learn from data, adapt, and improve their performance over time.

- With the ability to handle complex and non-linear relationships, ANNs are widely used in machine learning applications such as image and speech recognition, natural language processing, and predictive modeling.

The architecture of an artificial neural network:

- To understand the concept of the architecture of an artificial neural network, we have to understand what a neural network consists of. In order to define a neural network that consists of a large number of artificial neurons, which are termed units arranged in a sequence of layers. Lets us look at various types of layers available in an artificial neural network.

- Artificial Neural Network primarily consists of three layers:

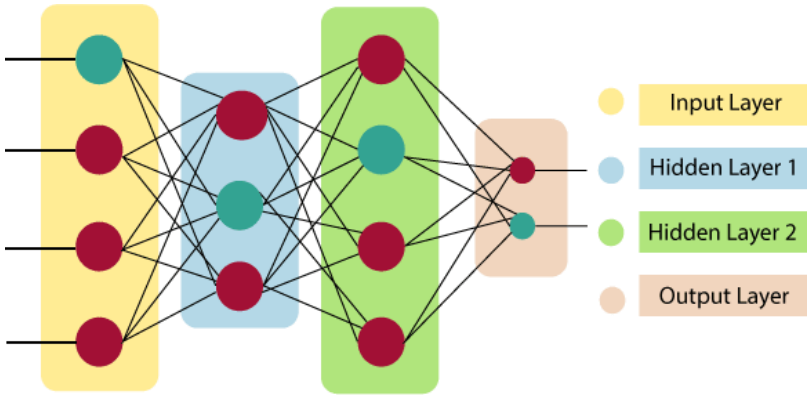


Fig 6.2 Artificial Neuron's

Input Layer:

- As the name suggests, it accepts inputs in several different formats provided by the programmer.

- The input layer serves as the first stage in processing data, ensuring that all relevant car attributes are correctly formatted before being passed to the hidden layers.

Hidden Layer:

- The hidden layer is present between the input and output layers. It performs all the calculations to find hidden features and patterns.
- Multiple hidden layers can be used to increase the complexity and learning capability of the model, improving accuracy in price predictions.
- These layers apply activation functions like ReLU (Rectified Linear Unit) or Sigmoid to introduce non-linearity and better model the relationships between car attributes and prices.

Output Layer:

- The input goes through a series of transformations using the hidden layer, which finally results in output that is conveyed using this layer.
- The artificial neural network takes input and computes the weighted sum of the inputs and includes a bias.
- The final output represents the predicted price, ensuring it aligns with market trends and previously trained data.

$$\sum_{i=1}^{n} Wi * Xi + b$$

CHAPTER - 7

SYSTEM DESIGN

System Design

7.1 SYSTEM ARCHITECTURE

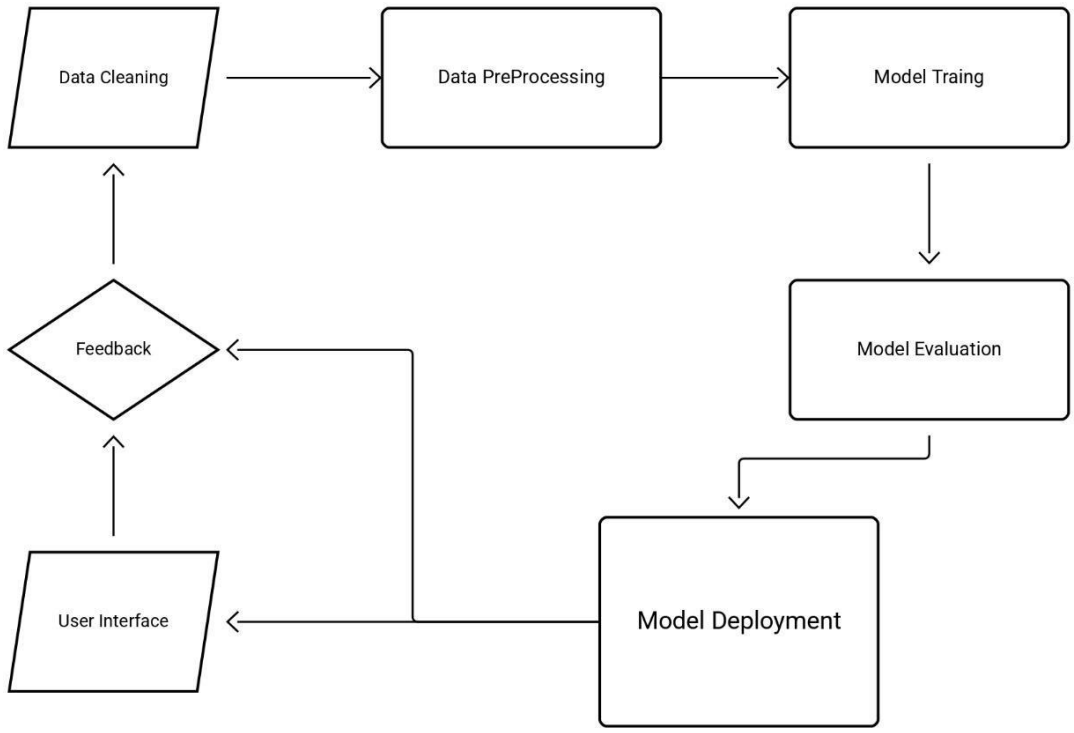


Fig 7.1 System Architecture

7.2 DATA FLOW DIAGRAM

The system begins with the Remote User interacting with the platform through a registration or login process. New users must first register by providing their personal details, after which they can log in to access the system. Returning users directly proceed to the login module, where their credentials are verified. This ensures that only authorized users gain access to the prediction features of the system.

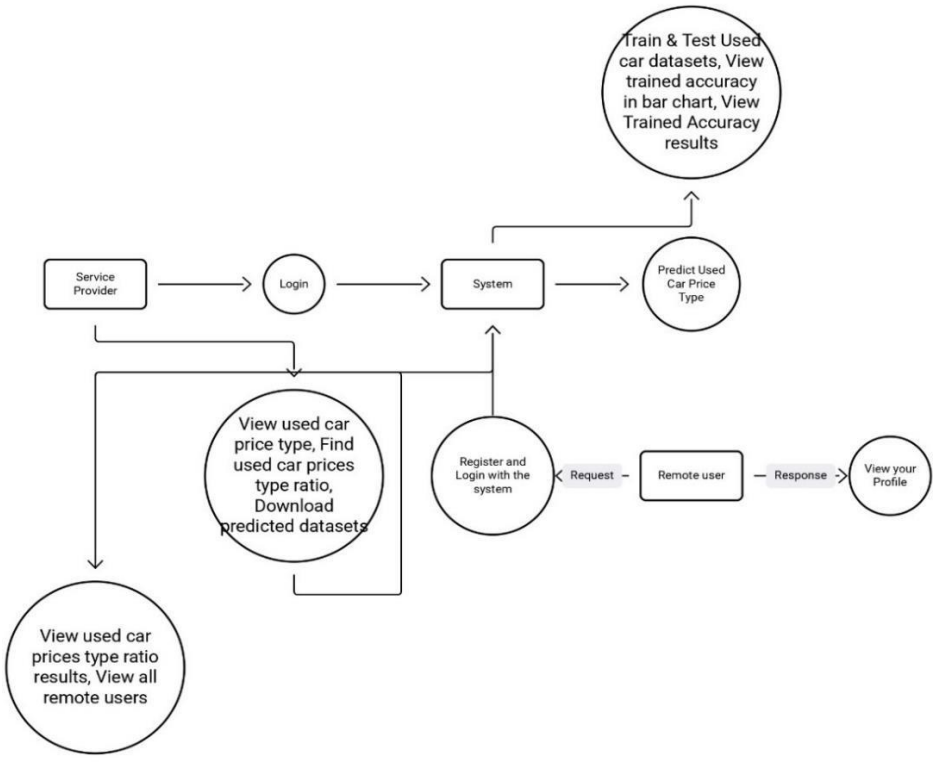
Once authenticated, the user can view the system dashboard, which provides access to various services. One of the main features is “View Car Prices and Ratios”, where users can input specific car details such as model, year, mileage, and fuel type. The

system, managed by the Service Provider, processes this input through pre-trained machine learning and neural network models.

The Service Provider handles the core functionalities, including data preprocessing, model training, and price prediction. Upon processing, the system displays the predicted car price along with additional insights like price-to-mileage ratios or performance ratios. This helps users better understand the factors influencing the car's resale value.

To maintain system reliability, an accuracy check is also integrated. This module evaluates the model’s prediction accuracy based on historical data and model performance metrics. It ensures the system delivers accurate and trustworthy price predictions to the users.

Fig 7.2 Data Flow Diagram



7.3 UML DIAGRAMS

Unified Modeling Language (UML) is a standardized modeling language used in object- oriented software engineering for designing, visualizing, and documenting system components. It plays a crucial role in software development by representing system architecture through graphical notations. UML consists of notation and meta-models, allowing adaptability for future enhancements.

Objectives of UML:

- Provide a user-friendly and expressive visual modeling language.
- Support extensibility and adaptability of core concepts.
- Maintain independence from specific programming languages and environments.
- Establish standardized modeling rules for consistency.
- Support object-oriented software development and complex system modeling.

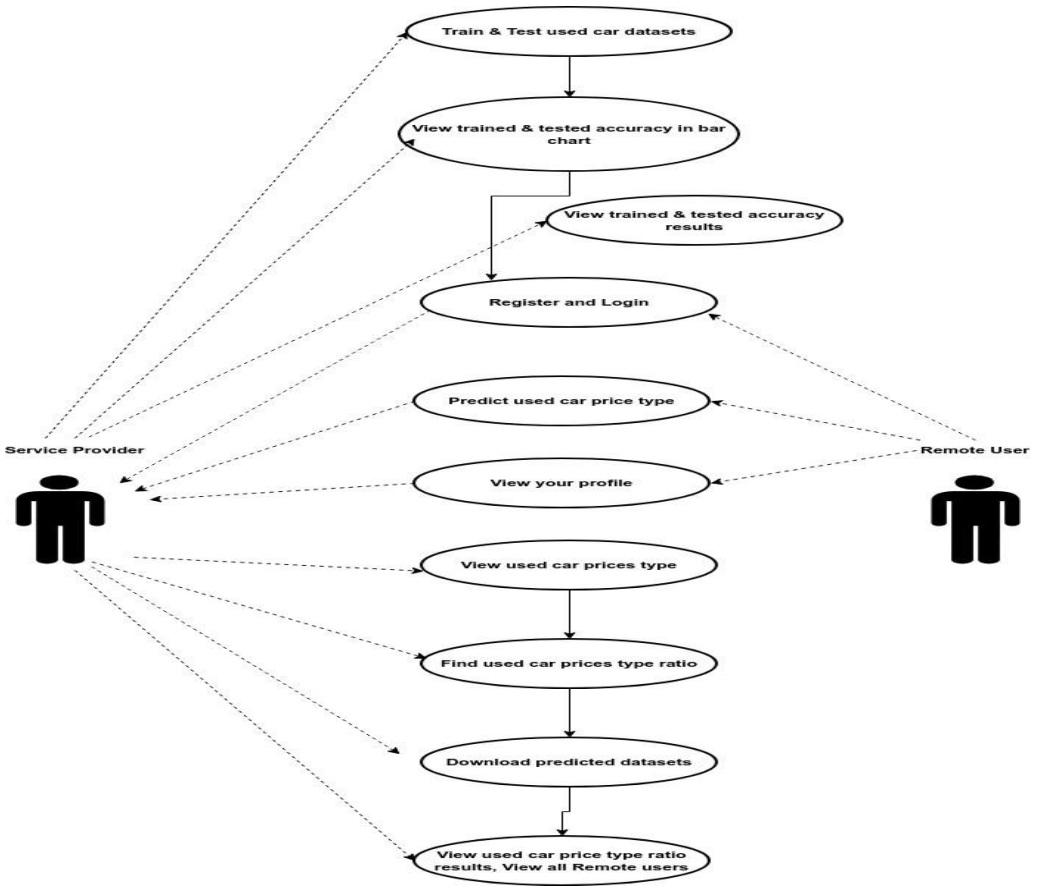


Fig 7.3 UML Diagram

7.4 CLASS DIAGRAM

Throughout software program development, a class chart there in version transformation pronunciation (uml) is just a style of whole structural schematic a certain characterizes a constructing of either a framework except attempting to reveal this identical shape's school rooms, about there characteristics, sports (or techniques), and indeed the connections amongst courses. This allows to explain that also elegance includes data of data and storing involvement preprocessing of data.

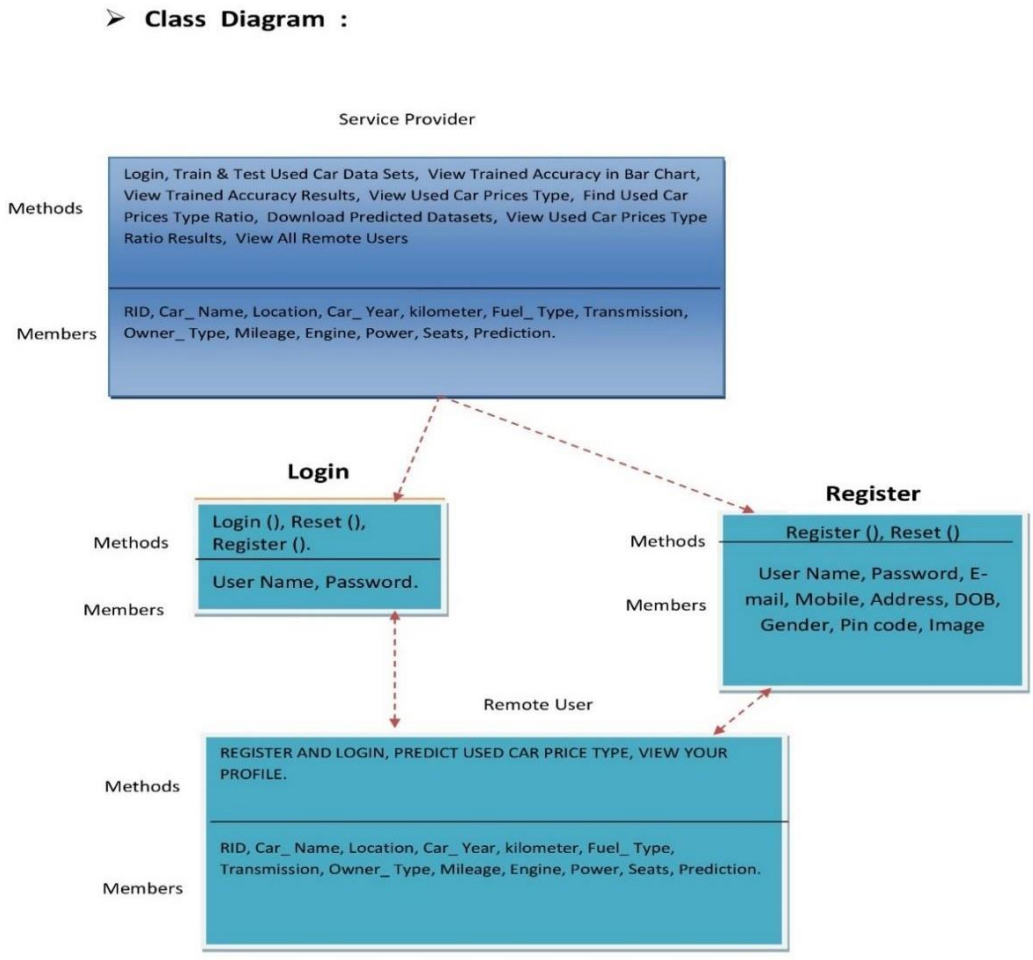


Fig 7.4 Class Diagram

7.5 SEQUENCE DIAGRAM

One sequence of discrete along model transformation vocabulary (uml) is kind of a communication diagrams where and procedures function with each other and under what command. That is a concoct of either a sequential function diagram (figure. Flowcharts sometimes are termed sequence diagram, incident situations, but also duration bldcm.

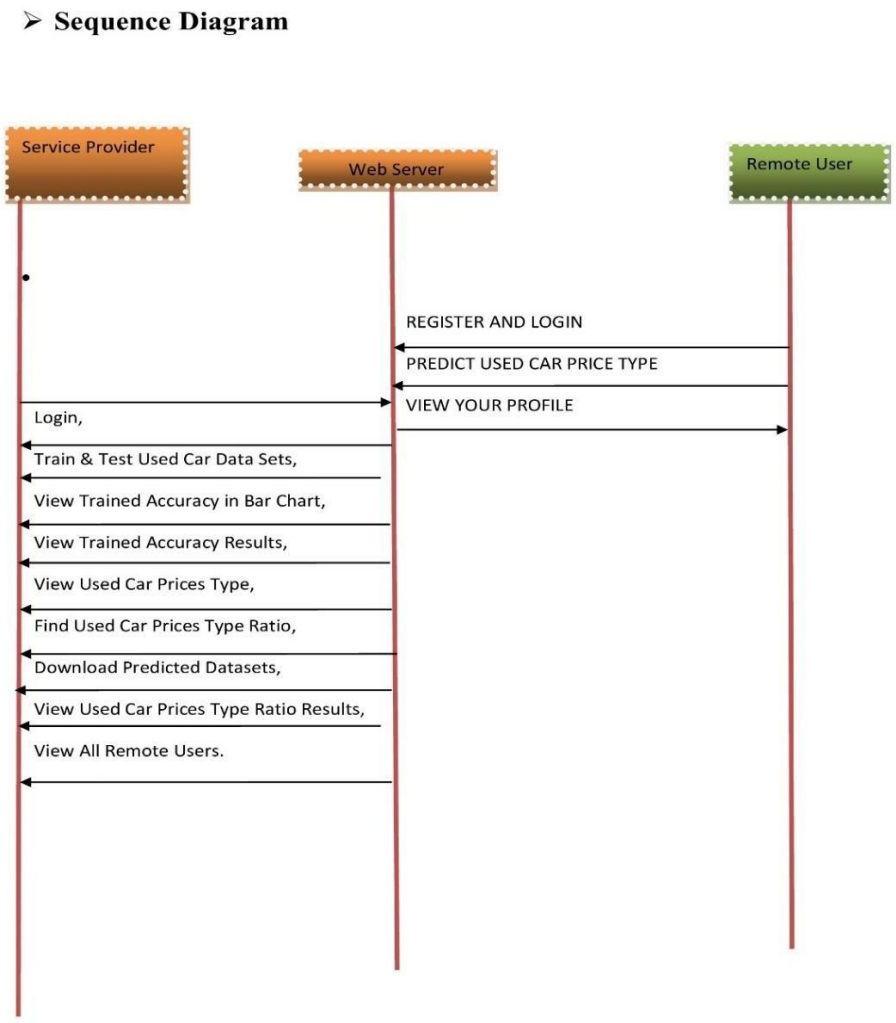
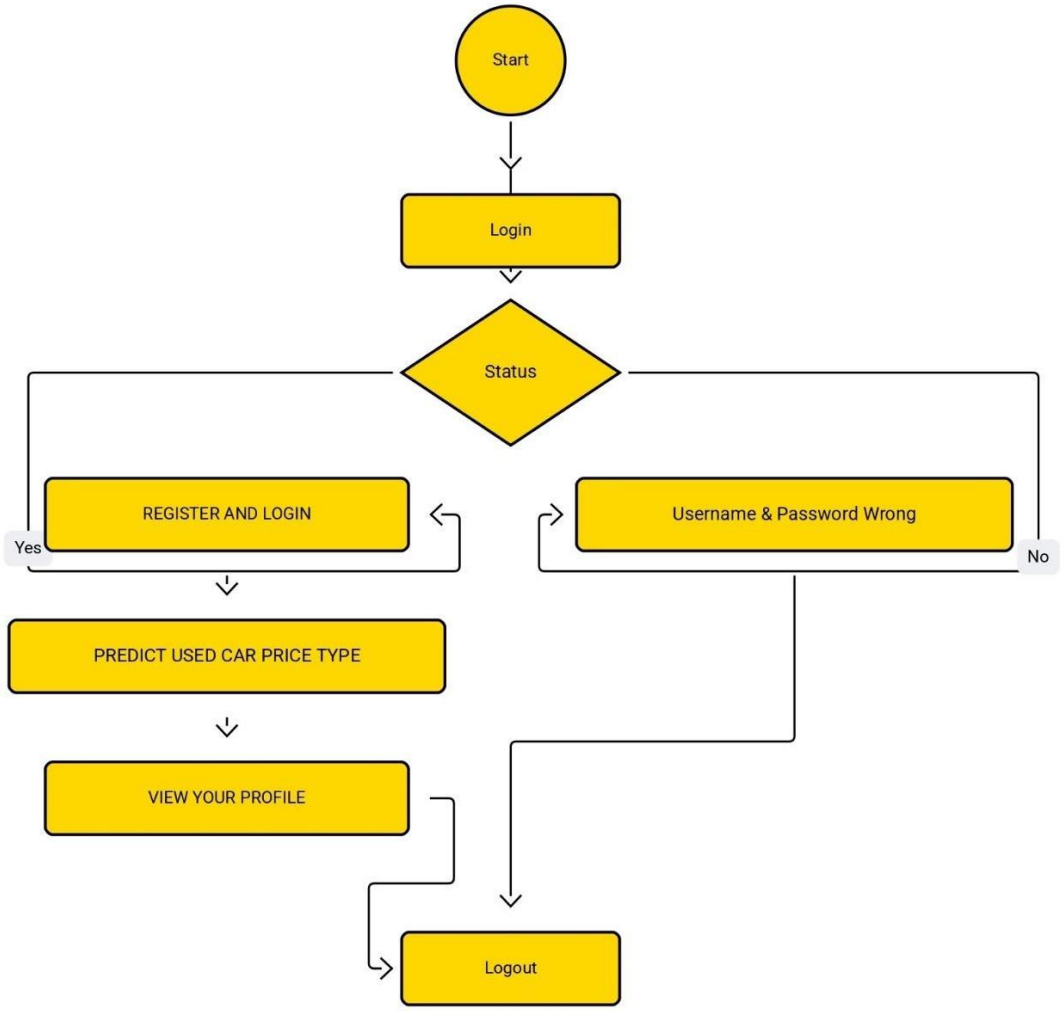


Fig 7.5 Sequence Diagram

7.6 ACTIVITY DIAGRAM

Flowcharts were also line graphs yeah workflow management after all systematic activities as for assist regarding preference, new gen but also multi - threading. There in uml modeling vocabulary, pastime graphs will be used to clarify this same commercial venture but instead organisational step- via-step workflow processes yeah matter in an object inside a gadget. Some kind attention schematic suggests the general float yeah deceive.

REMOTE USER



SERVICE PROVIDER

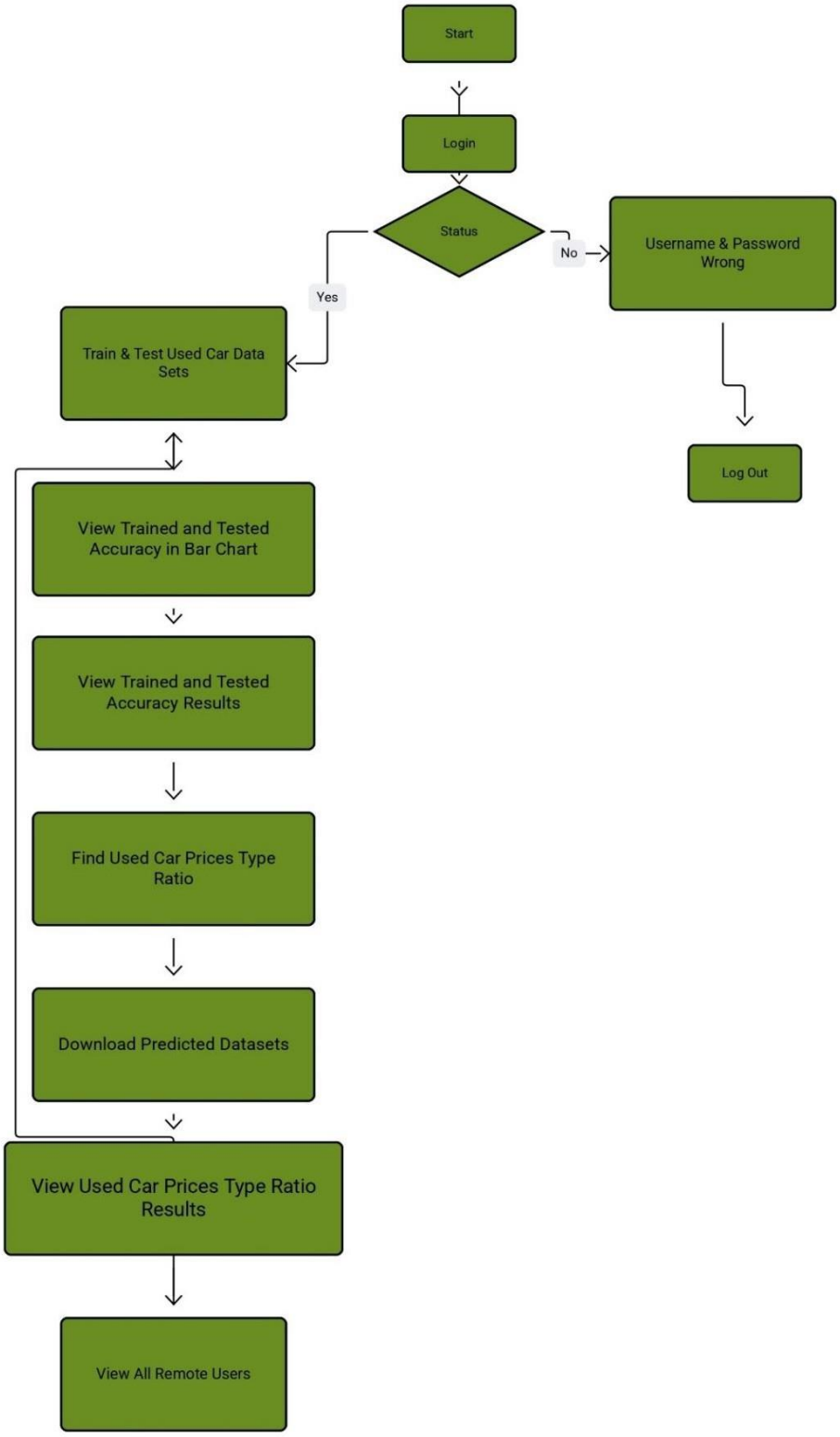


Fig 7.6 Activity Diagram

7.7 COLLABORATION DIAGRAM:

One cooperation graph factions all collectively interactions between the different things. This identical interaction had been noted despite the fact that ranked communique and it assist whole tune down a sequential of conversations. Its cooperation schematic targets to recognize together all ability interactions that every attribute has had with both this stuff.

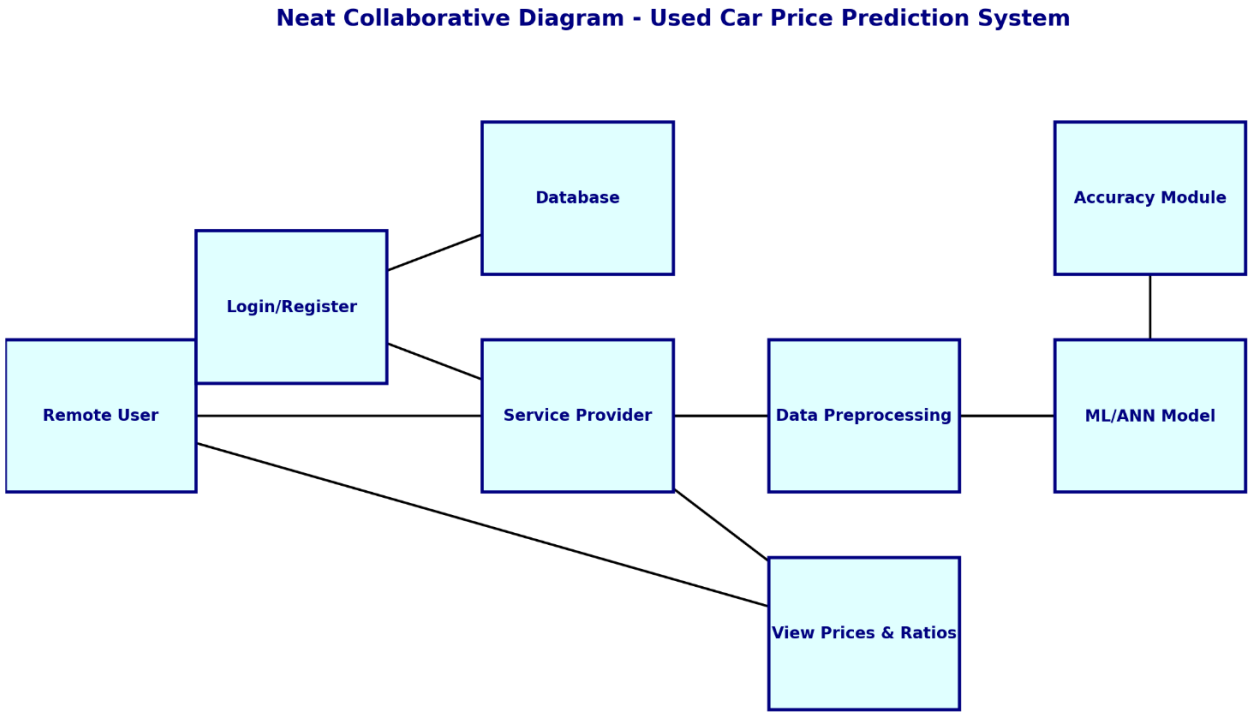


Fig 7.7 Collaboration Diagram

7.8 IMPLEMENTATION MODULES:

- Register
- Login
- Basic Car Details
- Accuracy
- Log Out

CHAPTER - 8

IMPLEMENTATION

IMPLEMENTATION

Implementation

The implementation of the Predicting used car prices using machine learning and artificial neuron's is divided into multiple modules, ensuring a structured and efficient workflow. The project is developed using Django as the backend framework, machine learning models for price prediction, and a user-friendly frontend for interaction. The following sections describe the key components of the implementation.

1. Web Framework – Django

Django is used to manage the web-based interface and backend functionality. It handles HTTP requests, user authentication, and data management. The framework follows the **Model-View-Template (MVT)** architecture, ensuring a clean separation of concerns.

- **Models:** Define the database structure, storing user details and prediction history.
- **Views:** Handle user input, process requests, and return responses.
- **Templates:** Render dynamic content on the frontend using HTML and CSS.

2. Database Management – PyMySQL

The system uses MySQL as the database, and PyMySQL is used to establish a connection between Django and the database. The database stores:

- User registration and login credentials.
- Historical predictions for analysis and improvement.
- Car attributes, including year, mileage, fuel type, and engine power.

3. Machine Learning Model Integration

The price prediction model is built using Scikit-learn and Artificial Neural Networks (ANN) with TensorFlow and Keras. The dataset is preprocessed to remove missing values, encode categorical variables, and normalize numerical data.

Machine Learning Models Used:

- **Random Forest Regressor:** Improves accuracy using multiple decision trees.
- **Linear Regression:** Estimates car prices based on linear relationships.
- **K-Nearest Neighbours (KNN):** Predicts prices based on similar past listings.
- **Support Vector Regression (SVR):** Enhances predictions using regression

planes.

The trained model is saved and loaded into the Django application, allowing real-time predictions based on user input.

4. User Interaction & Price Prediction Workflow

1. **User Registration & Login:** Users sign up and authenticate using Django's authentication system.
2. **Input Data:** Users enter car details, such as year, mileage, fuel type, and engine power.
3. **Model Processing:** The backend loads the trained model and predicts the car's price.
4. **Result Display:** The predicted price is displayed on the frontend.

5. Security & Optimization

To ensure security and efficiency:

- **Cross-Site Request Forgery (CSRF) Protection** is implemented to prevent unauthorized requests.
- **Session Management** is used for secure user authentication.
- **Optimized Query Execution** improves database performance.

The predicting used car prices using machine learning and artificial neuron's is designed for accuracy, efficiency, and user-friendliness, ensuring a seamless experience for users looking to estimate car values.

Source Code

remoteuser.py

```
from django.db.models import Count, Q
from django.shortcuts import render, redirect
import pandas as pd
import warnings
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report
```

```
from sklearn.ensemble import VotingClassifier, RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn import svm
from Remote_User.models import ClientRegister_Model, price_prediction
warnings.filterwarnings("ignore")

def login(request):
    if request.method == "POST" and 'submit1' in request.POST:
        username = request.POST.get('username')
        password = request.POST.get('password')
        try:
            user = ClientRegister_Model.objects.get(username=username,
password=password)
            request.session["userid"] = user.id
            return redirect('ViewYourProfile')
        except ClientRegister_Model.DoesNotExist:
            pass
    return render(request, 'RUser/login.html')

def Register1(request):
    if request.method == "POST":
        ClientRegister_Model.objects.create(
            username=request.POST.get('username'),
            email=request.POST.get('email'),
            password=request.POST.get('password'),
            phoneno=request.POST.get('phoneno'),
            country=request.POST.get('country'),
            state=request.POST.get('state'),
            city=request.POST.get('city'),
            address=request.POST.get('address'),
            gender=request.POST.get('gender')
        )
        obj = "Registered Successfully"
        return render(request, 'RUser/Register1.html', {'object': obj})
    return render(request, 'RUser/Register1.html')

def ViewYourProfile(request):
    userid = request.session['userid']
    user = ClientRegister_Model.objects.get(id=userid)
    return render(request, 'RUser/ViewYourProfile.html', {'object': user})

def predict_used_car_price_type(request):
    if request.method == "POST":
```

```
RID = request.POST.get('RID')
Car_Name = request.POST.get('Car_Name')
Location = request.POST.get('Location')
Car_Year = request.POST.get('Car_Year')
kilometer = request.POST.get('kilometer')
Fuel_Type = request.POST.get('Fuel_Type')
Transmission = request.POST.get('Transmission')
Owner_Type = request.POST.get('Owner_Type')
Mileage = request.POST.get('Mileage')
Engine = request.POST.get('Engine')
Power = request.POST.get('Power')
Seats = request.POST.get('Seats')

df = pd.read_csv('Datasets.csv')

def apply_results(price):
    price = float(price)
    if price <= 5.0:
        return 0
    elif 5.0 < price <= 20.0:
        return 1
    elif 20.0 < price <= 100.0:
        return 2

df['Results'] = df['Price'].apply(apply_results)

cv = CountVectorizer()
X = cv.fit_transform(df['RID'].astype(str))
y = df['Results']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20)

models = []
kn = KNeighborsClassifier()
kn.fit(X_train, y_train)
models.append(('KNeighborsClassifier', kn))

svm_model = svm.LinearSVC()
svm_model.fit(X_train, y_train)
models.append(('SVM', svm_model))

log_reg = LogisticRegression(random_state=0, solver='lbfgs')
log_reg.fit(X_train, y_train)
models.append(('LogisticRegression', log_reg))
```

```
        rf = RandomForestClassifier()
        rf.fit(X_train, y_train)
        models.append(('RandomForest', rf))

        ensemble = VotingClassifier(estimators=models, voting='hard')
        ensemble.fit(X_train, y_train)

        input_vector = cv.transform([RID]).toarray()
        prediction = ensemble.predict(input_vector)[0]

        if prediction == 0:
            price_range = 'Below 5L'
        elif prediction == 1:
            price_range = 'More Than 5L and Below 20L'
        else:
            price_range = 'More Than 20L and Below 100L'

        price_prediction.objects.create(
            RID=RID,
            Car_Name=Car_Name,
            Location=Location,
            Car_Year=Car_Year,
            kilometer=kilometer,
            Fuel_Type=Fuel_Type,
            Transmission=Transmission,
            Owner_Type=Owner_Type,
            Mileage=Mileage,
            Engine=Engine,
            Power=Power,
            Seats=Seats,
            Prediction=price_range
        )

        return render(request, 'RUser/predict_used_car_price_type.html', {'objs': price_range})

    returnrender(request,'RUser/predict_used_car_price_type.html')
```

serviceprovider.py

```
from django.db.models import  Count, Avg
from django.shortcuts import render, redirect
```

```
from django.db.models import Count
from django.db.models import Q
import datetime
import xlwt
from django.http import HttpResponse

import pandas as pd
import numpy as np

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report
from sklearn.metrics import accuracy_score
from sklearn.metrics import precision_score, recall_score
from sklearn.metrics import f1_score, matthews_corrcoef
from sklearn.tree import DecisionTreeClassifier

# Create your views here.
fromRemote_User.modelsimport
ClientRegister_Model,price_prediction,detection_ratio,detection_accuracy

def serviceproviderlogin(request):
    if request.method  == "POST":
        admin = request.POST.get('username')
        password = request.POST.get('password')
        if admin == "Admin" and password =="Admin":
            return redirect('View_Remote_Users')

    return render(request,'SProvider/serviceproviderlogin.html')

def Find_Used_Car_Price_Type_Ratio(request):
    detection_ratio.objects.all().delete()
    ratio = ""
    kword = 'Below 5L'
    print(kword)
    obj = price_prediction.objects.all().filter(Q(Prediction=kword))
    obj1 =price_prediction.objects.all()
    count = obj.count();
    count1 = obj1.count();
    ratio = (count / count1) * 100
    if ratio != 0:
        detection_ratio.objects.create(names=kword, ratio=ratio)

    ratio1 = ""
```

```
    kword1 = 'More Than 5L and Below 20L'
    print(kword1)
    obj1 = price_prediction.objects.all().filter(Q(Prediction=kword1))
    obj11 = price_prediction.objects.all()
    count1 = obj1.count();
    count11 = obj11.count();
    ratio1 = (count1 / count11) * 100
    if ratio1 != 0:
        detection_ratio.objects.create(names=kword1,  ratio=ratio1)

    ratio12 = ""
    kword12 = 'More Than 20L and Below 100L'
    print(kword12)
    obj12 = price_prediction.objects.all().filter(Q(Prediction=kword12))
    obj112 = price_prediction.objects.all()
    count12 = obj12.count();
    count112 = obj112.count();
    ratio12 = (count12 / count112) * 100
    if ratio12 != 0:
        detection_ratio.objects.create(names=kword12,  ratio=ratio12)

    obj = detection_ratio.objects.all()
        return render(request, 'SProvider/Find_Used_Car_Price_Type_Ratio.html',
{'objs': obj})

def View_Remote_Users(request):
    obj=ClientRegister_Model.objects.all()
    return render(request,'SProvider/View_Remote_Users.html',{'objects':obj})

def ViewTrendings(request):
                                                    topic                    =
price_prediction.objects.values('topics').annotate(dcount=Count('topics')).order_by
('-dcount')
    return  render(request,'SProvider/ViewTrendings.html',{'objects':topic})

def charts(request,chart_type):
    chart1 = detection_ratio.objects.values('names').annotate(dcount=Avg('ratio'))
                return        render(request,"SProvider/charts.html",        {'form':chart1,
'chart_type':chart_type})

def charts1(request,chart_type):
                                                chart1                    =
detection_accuracy.objects.values('names').annotate(dcount=Avg('ratio'))
                return        render(request,"SProvider/charts1.html",        {'form':chart1,
```

```
'chart_type':chart_type})

def View_Prediction_Of_Used_Car_Price(request):
    obj =price_prediction.objects.all()
      return  render(request,  'SProvider/View_Prediction_Of_Used_Car_Price.html',
{'list_objects': obj})

def likeschart(request,like_chart):
    charts =detection_accuracy.objects.values('names').annotate(dcount=Avg('ratio'))
              return       render(request,"SProvider/likeschart.html",       {'form':charts,
'like_chart':like_chart})

def Download_Trained_DataSets(request):

    response = HttpResponse(content_type='application/ms-excel')
    # decide file name
    response['Content-Disposition'] = 'attachment; filename="TrainedData.xls"'
    # creating workbook
    wb = xlwt.Workbook(encoding='utf-8')
    # adding sheet
    ws = wb.add_sheet("sheet1")
    # Sheet header, first row
    row_num = 0
    font_style = xlwt.XFStyle()
    # headers are bold
    font_style.font.bold = True
    # writer = csv.writer(response)
    obj =price_prediction.objects.all()
    data = obj  # dummy method to fetch data.
    for my_row in data:
        row_num = row_num + 1

        ws.write(row_num, 0, my_row.RID, font_style)
        ws.write(row_num, 1, my_row.Car_Name, font_style)
        ws.write(row_num, 2, my_row.Location, font_style)
        ws.write(row_num, 3, my_row.Car_Year, font_style)
        ws.write(row_num, 4, my_row.kilometer, font_style)
        ws.write(row_num, 5, my_row.Fuel_Type, font_style)
        ws.write(row_num, 6, my_row.Transmission, font_style)
        ws.write(row_num, 7, my_row.Owner_Type, font_style)
        ws.write(row_num, 8, my_row.Mileage, font_style)
        ws.write(row_num, 9, my_row.Engine, font_style)
        ws.write(row_num, 10, my_row.Power, font_style)
        ws.write(row_num, 11, my_row.Seats, font_style)
```

```
            ws.write(row_num, 12, my_row.Prediction, font_style)

        wb.save(response)
        return response

    def Train_Test_DataSets(request):
        detection_accuracy.objects.all().delete()

        df = pd.read_csv('Datasets.csv')
        df
        df.columns

        def apply_results(results):

            if float(results) <= 5.0:
                return 0
            elif float(results) >= 5.0 and float(results) <= 20.0:
                return 1
            elif float(results) >= 20.0 and float(results) <= 100.0:
                return 2

        df['Results'] = df['Price'].apply(apply_results)

        cv = CountVectorizer()
        X = df['Name'].apply(str)
        y = df['Results']

        print("Car Name")
        print(X)
        print("Label")
        print(y)

        X = cv.fit_transform(X)


        models = []
        from sklearn.model_selection import train_test_split
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20)
        X_train.shape, X_test.shape, y_train.shape

        print("Random Forest Classifier")
        from sklearn.ensemble import RandomForestClassifier
        rf_clf = RandomForestClassifier()
        rf_clf.fit(X_train, y_train)
```

```
    rfpredict = rf_clf.predict(X_test)
    print("ACCURACY")
    print(accuracy_score(y_test, rfpredict) * 100)
    print("CLASSIFICATION REPORT")
    print(classification_report(y_test, rfpredict))
    print("CONFUSION MATRIX")
    print(confusion_matrix(y_test, rfpredict))
    models.append(('RandomForestClassifier', rf_clf))
          detection_accuracy.objects.create(names="Random Forest Classifier",
ratio=accuracy_score(y_test, rfpredict) * 100)

    # SVM Model
    print("SVM")
    from sklearn import svm
    lin_clf = svm.LinearSVC()
    lin_clf.fit(X_train, y_train)
    predict_svm = lin_clf.predict(X_test)
    svm_acc = accuracy_score(y_test, predict_svm) * 100
    print(svm_acc)
    print("CLASSIFICATION REPORT")
    print(classification_report(y_test, predict_svm))
    print("CONFUSION MATRIX")
    print(confusion_matrix(y_test, predict_svm))
    models.append(('svm', lin_clf))
    detection_accuracy.objects.create(names="SVM", ratio=svm_acc)
    print("Logistic Regression")
    from sklearn.linear_model import LogisticRegression
    reg = LogisticRegression(random_state=0, solver='lbfgs').fit(X_train, y_train)
    y_pred = reg.predict(X_test)
    print("ACCURACY")
    print(accuracy_score(y_test, y_pred) * 100)
    print("CLASSIFICATION REPORT")
    print(classification_report(y_test, y_pred))
    print("CONFUSION MATRIX")
    print(confusion_matrix(y_test, y_pred))
    models.append(('logistic', reg))
     detection_accuracy.objects.create(names="Logistic Regression",
ratio=accuracy_score(y_test, y_pred) * 100)
     print("KNeighborsClassifier")
    from sklearn.neighbors import KNeighborsClassifier
    kn = KNeighborsClassifier()
    kn.fit(X_train, y_train)
    knpredict = kn.predict(X_test)
    print("ACCURACY")
```

```
    print(accuracy_score(y_test, knpredict) * 100)
    print("CLASSIFICATION REPORT")
    print(classification_report(y_test, knpredict))
    print("CONFUSION MATRIX")
    print(confusion_matrix(y_test, knpredict))
    models.append(('KNeighborsClassifier', kn))
                        detection_accuracy.objects.create(names="KNeighborsClassifier",
ratio=accuracy_score(y_test, knpredict) * 100)
    predicts = 'predicts.csv'
    df.to_csv(predicts, index=False)
    df.to_markdown
    obj = detection_accuracy.objects.all()
    return render(request,'SProvider/Train_Test_DataSets.html', {'objs': obj})
```

System Setup

The system setup involves configuring the development environment, installing necessary dependencies, and preparing the database for the Used Car Price Prediction System. Below is a step-by-step guide to setting up the project.

1. Prerequisites

Before starting the setup, ensure that the required software and tools are installed. The project requires Python for backend development, Django for web framework management, and MySQL for database handling. Additional machine learning libraries such as TensorFlow, Keras, Scikit-learn, Pandas, and NumPy are also necessary for data processing and model training.

2. Environment Configuration

To maintain a clean development environment, a virtual environment is used to manage dependencies. Once the environment is set up, all required libraries, including Django, machine learning frameworks, and database connectors, are installed. This step ensures that the system runs smoothly without conflicts between packages.

3. Project Initialization

The Django framework is used to create the web-based interface. A new Django project is initialized, followed by the creation of a dedicated app for car price prediction. This modular approach allows better management of features such as data

processing, user authentication, and price estimation.

4. Database Setup

The system uses MySQL as the primary database for storing user details, car attributes, and prediction results. The database is configured within the Django settings, allowing secure connectivity. Necessary tables and schemas are created using Django's migration system to store relevant data efficiently.

5. Model Training and Integration

The machine learning and artificial neural network models are trained using historical car sales data. The dataset undergoes preprocessing, including handling missing values, feature encoding, and normalization. Various regression models, including Random Forest, Linear Regression, and K-Nearest Neighbors, are tested to determine the most accurate prediction model. The final trained model is saved and integrated into the Django application for real-time predictions.

6. Running the Application

Once the system is set up, the Django web application is launched. Users can input car details, and the backend processes the data, applies the trained machine learning model, and returns the predicted car price. The system ensures real-time interaction between users and the prediction model.

7. Deployment and Hosting

For production deployment, the system can be hosted on cloud platforms such as AWS, Heroku, or PythonAnywhere. The trained models and database are linked to a cloud-based environment, ensuring accessibility and scalability.

This structured system setup ensures seamless integration of machine learning with Django, providing an efficient and user-friendly experience for predicting used car prices.

CHAPTER - 9

SYSTEM TESTING

SYSTEM TESTING

System Testing

The primary objective of system testing is to identify errors, particularly related to security, and ensure that the software functions as expected and satisfies user requirements. A variety of testing methods are applied to validate the software at different stages, from individual components to the final product.

Unit Testing

Unit testing focuses on testing individual components of the software to ensure they work correctly. This includes validating that inputs produce valid outputs and verifying all possible decision paths and internal code flows. Unit tests are performed before integrating components to ensure that the business processes and system configurations meet the defined criteria.

Integration Testing

Integration testing ensures that all software components work together seamlessly. Once the unit tests are successful, integration testing confirms that the combined components interact correctly and consistently. This type of testing helps identify issues that may arise when individual components are integrated into the system.

Functional Testing

Functional testing verifies that the software's functions meet the criteria outlined in the user manual, system documentation, and business requirements. This includes validating input acceptance, ensuring invalid inputs are rejected, verifying actions, and confirming proper outputs. Functional testing prioritizes critical functionality, data fields, business process flows, and identifying additional tests if needed.

White Box Testing

White box testing involves examining the internal structure, logic, and flow of the software. It requires an understanding of the software's architecture and design, allowing testers to verify code behavior and identify potential issues that are not visible through external testing.

Black Box Testing

Black box testing focuses solely on the functionality of the software, without knowledge of its internal workings. This testing is based on requirements or specification documents, where the tester checks the software by providing inputs and verifying the corresponding outputs, without any concern for the internal code structure.

Programmatic Testing

Programmatic testing is an integral part of the software development lifecycle, conducted during the coding phase. Unit tests are executed to ensure that the software behaves as expected. These tests are accompanied by manual field tests to document functional behavior and ensure all components are active.

Performance Testing

Performance testing is part of software integration testing, where multiple components are tested sequentially on the same platform. This helps identify and resolve any interface issues that may affect the software's performance. It ensures that all interdependent software components.

User Acceptance Testing (UAT)

User Acceptance Testing verifies that the system fulfills the functional requirements as defined by end-users. It ensures that the software meets the specified needs and operates as expected in real-world conditions, with input from the target users.

CHAPTER - 10

SCREENSHOTS

EXECUTION AND RESULT

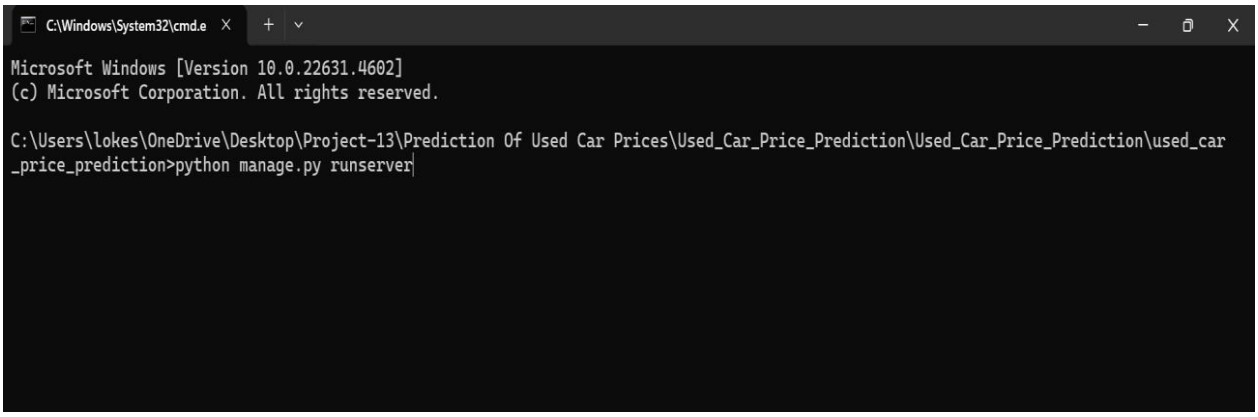


Fig 9.1: Starting command to run the Django application" python manage.py runserver".

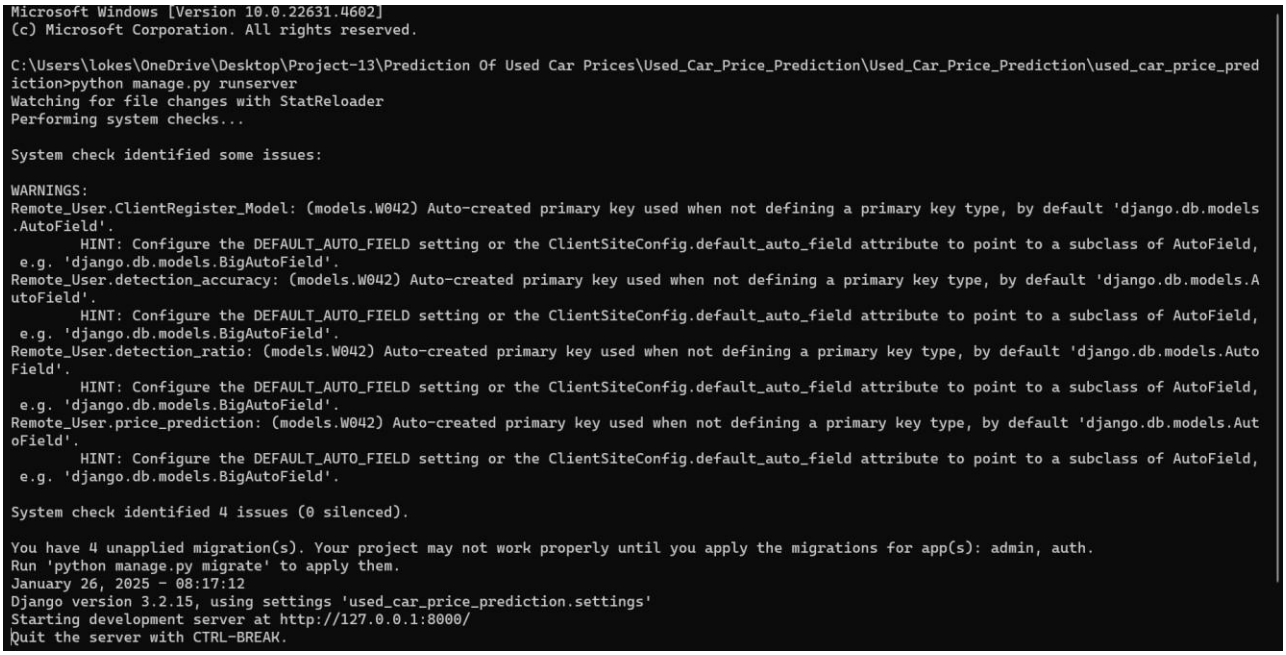
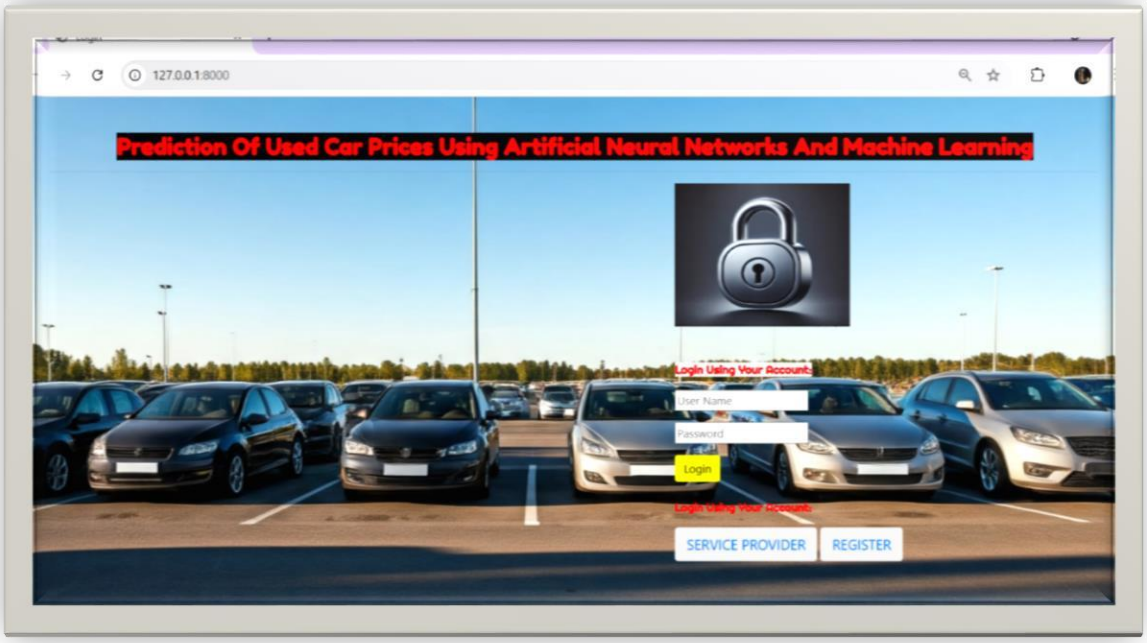
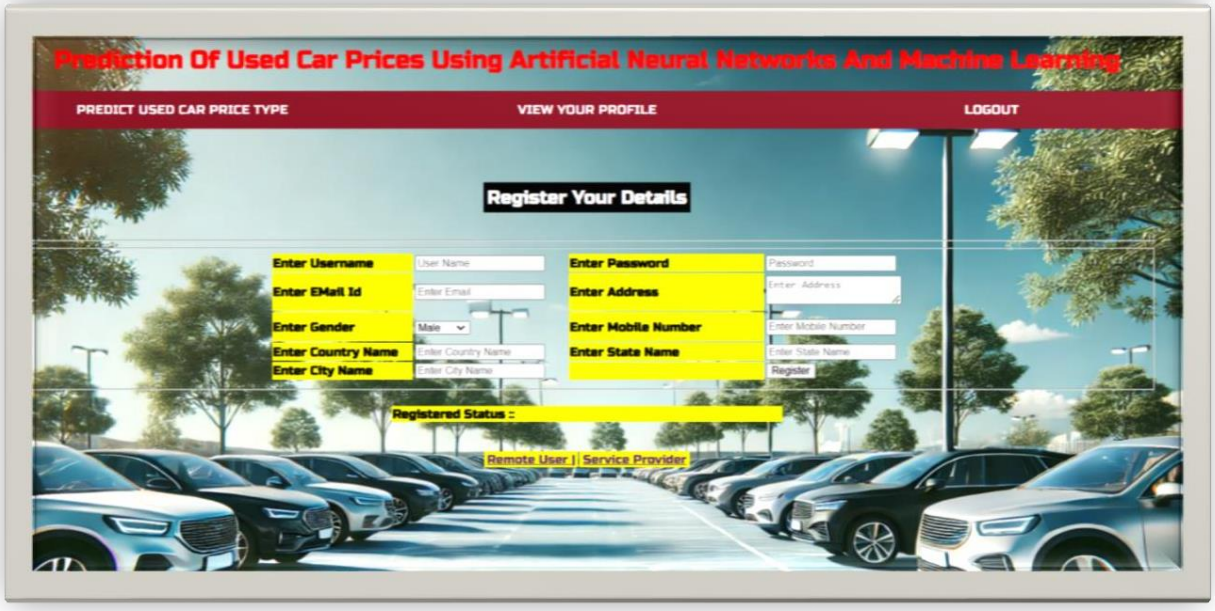


Fig 9.2: generation of Hyperlink.

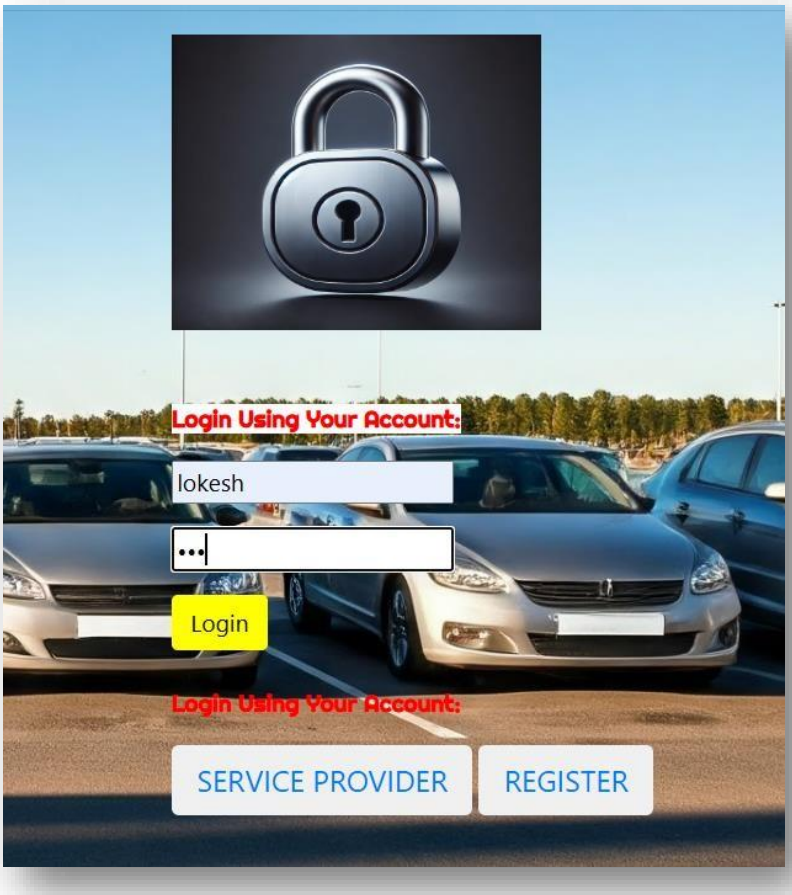
Index.html

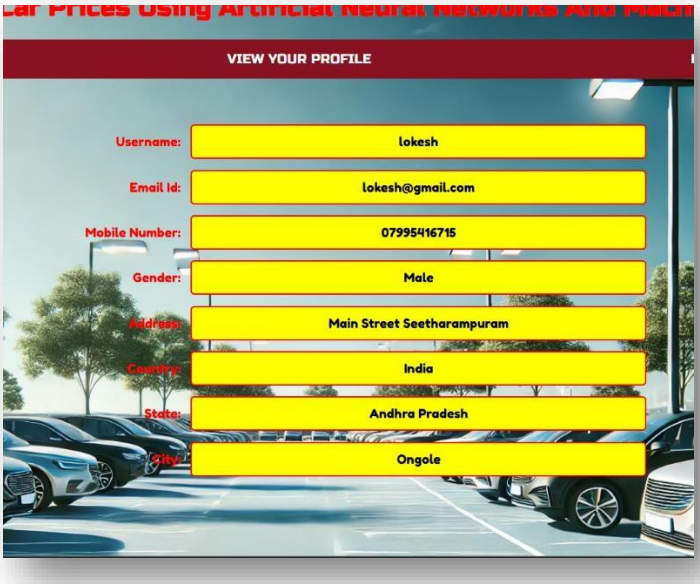


Register.html

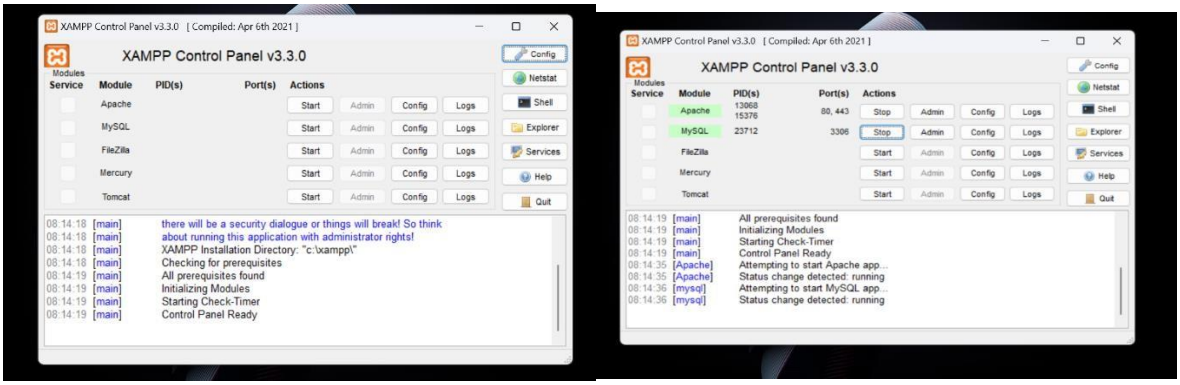


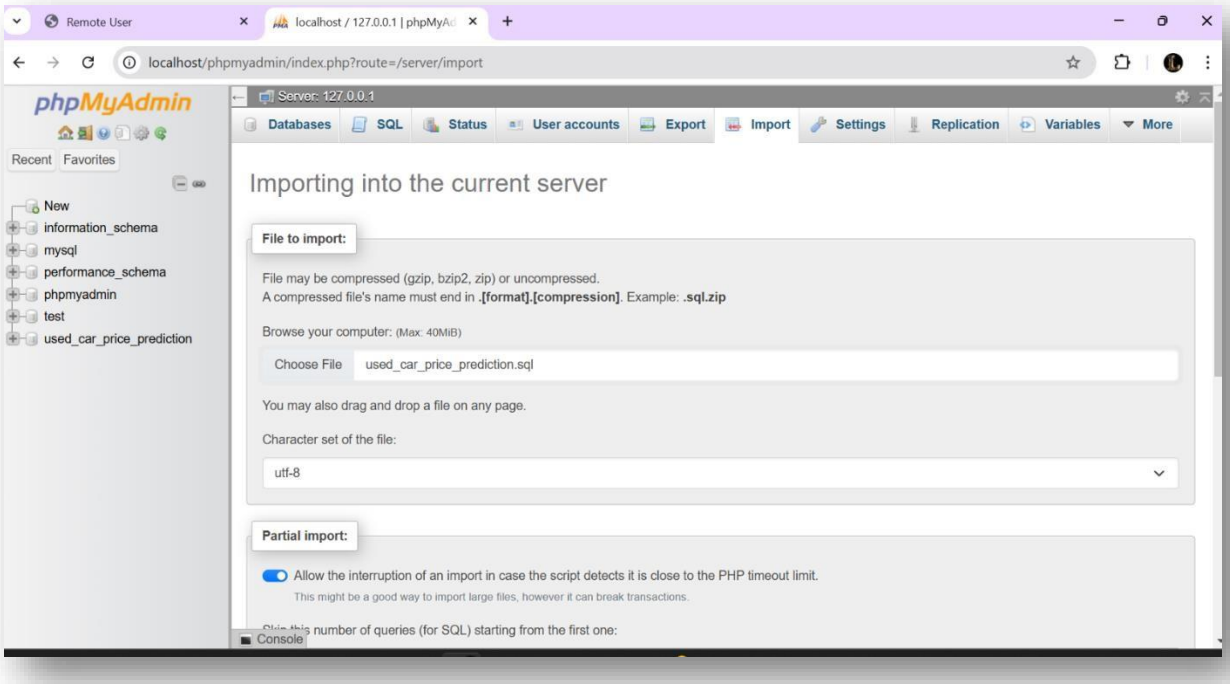
Login.html


Login Using Your Account:
lokesh
Login
Login Using Your Account:
SERVICE PROVIDER
REGISTER

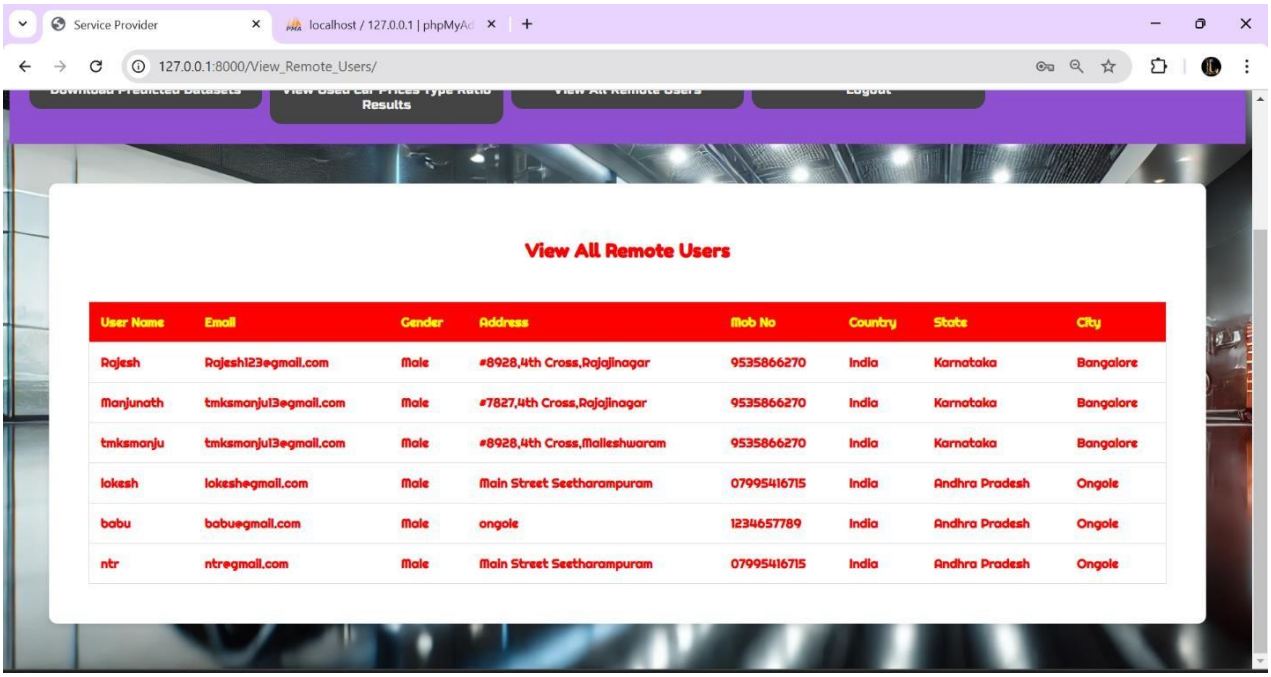

VIEW YOUR PROFILE
Username:
lokesh
Email Id:
lokesh@gmail.com
Mobile Number:
07995416715
Gender:
Male
Address:
Main Street Seetharampuram
Country:
India
State:
Andhra Pradesh
City:
Ongole

Upload.html

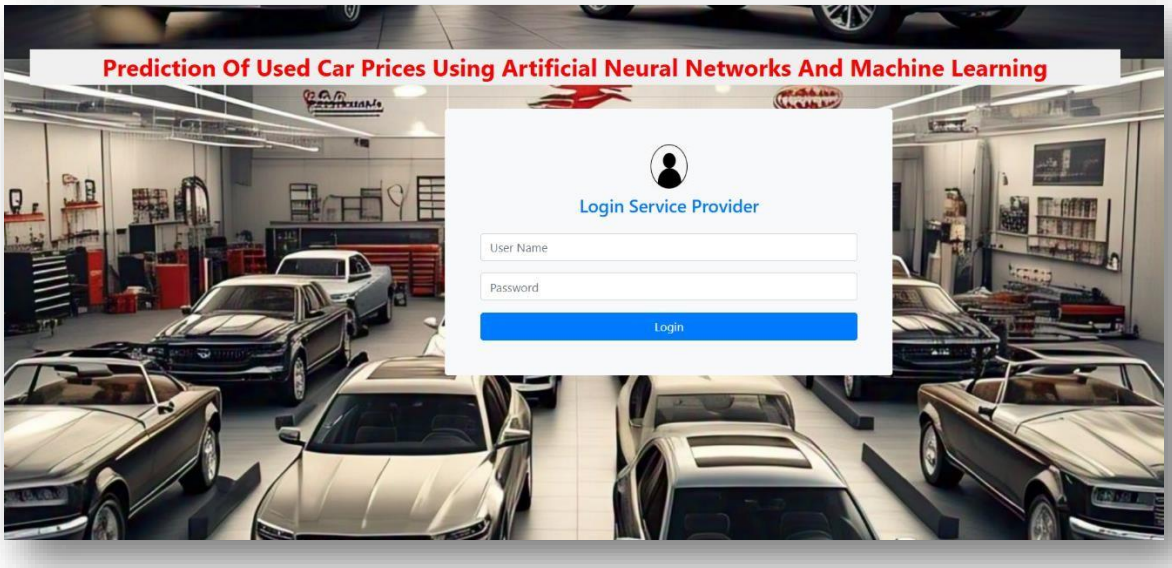

XAMPP Control Panel v3.3.0 [Compiled: Apr 6th 2021]
XAMPP Control Panel v3.3.0
Modules
Service Module PID(s) Port(s) Actions
Apache Start Admin Config Logs
MySQL Start Admin Config Logs
FileZilla Start Admin Config Logs
Mercury Start Admin Config Logs
Tomcat Start Admin Config Logs
Config
Netstat
Shell
Explorer
Services
Help
Quit
08:14:18 [main] there will be a security dialogue or things will break! So think
08:14:18 [main] about running this application with administrator rights!
08:14:18 [main] XAMPP Installation Directory: "c:\xampp\"
08:14:18 [main] Checking for prerequisites
08:14:19 [main] All prerequisites found
08:14:19 [main] Initializing Modules
08:14:19 [main] Starting Check-Timer
08:14:19 [main] Control Panel Ready
XAMPP Control Panel v3.3.0 [Compiled: Apr 6th 2021]
XAMPP Control Panel v3.3.0
Modules
Service Module PID(s) Port(s) Actions
Apache 13068 15376 80, 443 Stop Admin Config Logs
MySQL 23712 3306 Stop Admin Config Logs
FileZilla Start Admin Config Logs
Mercury Start Admin Config Logs
Tomcat Start Admin Config Logs
Config
Netstat
Shell
Explorer
Services
Help
Quit
08:14:19 [main] All prerequisites found
08:14:19 [main] Initializing Modules
08:14:19 [main] Starting Check-Timer
08:14:19 [main] Control Panel Ready
08:14:35 [Apache] Attempting to start Apache app...
08:14:35 [Apache] Status change detected: running
08:14:36 [mysql] Attempting to start MySQL app...
08:14:36 [mysql] Status change detected: running


Remote User
localhost / 127.0.0.1 | phpMyAd
localhost/phpmyadmin/index.php?route=/server/import
phpMyAdmin
Recent Favorites
New
information_schema
mysql
performance_schema
phpmyadmin
test
used_car_price_prediction
Server: 127.0.0.1
Databases SQL Status User accounts Export Import Settings Replication Variables More
Importing into the current server
File to import:
File may be compressed (gzip, bzip2, zip) or uncompressed.
A compressed file's name must end in .[format].[compression]. Example: .sql.zip
Browse your computer: (Max: 40MiB)
Choose File used_car_price_prediction.sql
You may also drag and drop a file on any page.
Character set of the file:
utf-8
Partial import:
Allow the interruption of an import in case the script detects it is close to the PHP timeout limit.
This might be a good way to import large files, however it can break transactions.
number of queries (for SQL) starting from the first one:
Console

login.html



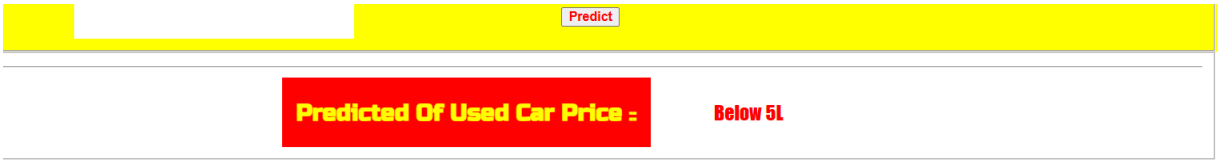
serviceproviderlogin.html

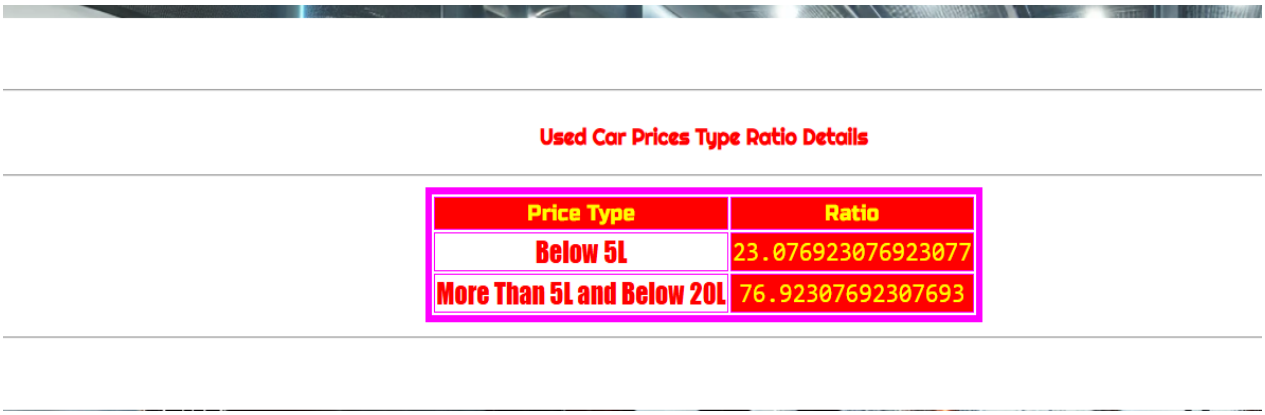


Prediction

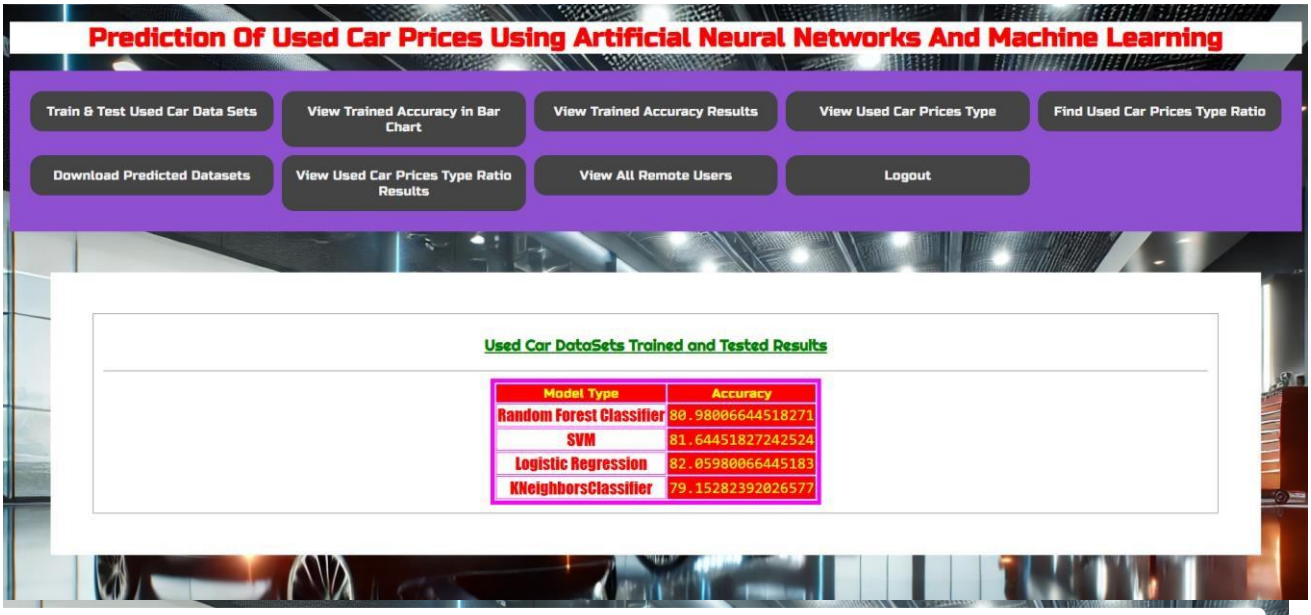
Input Data for Predicting of Car Price

Accurate Price of used car





Test Results



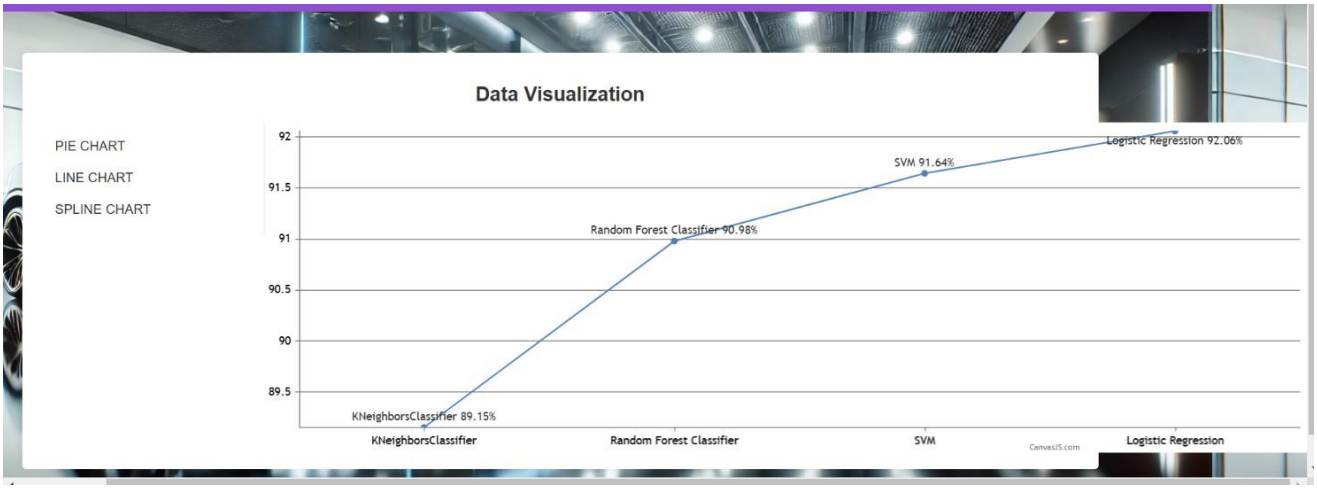
View Used Car Prices Type

RID	Car Name	Location	Car Year	Kilometer	Fuel Type	Transmission	Owner Type	Mileage	Engine	Power	Seats	Prediction
209.85.201.188-10.42.0.151-5228-53409-6	Hyundai Creta 1.6 CRDi SX Option	Pune	2015	41000	Diesel	Manual	First	19.67 kmpl	1582 CC	126.2 bhp	5	More Than 5L and Below 20L
180.76.147.97-10.42.0.211-80-60005-6	Honda Jazz V	Chennai	2011	46000	Petrol	Manual	First	18.2 kmpl	1199 CC	88.7 bhp	5	Below 5L
10.42.0.151-31.13.71.3-60663-443-6	Hyundai Verna Transform SX VGT CRDi	Ahmedabad	2010	70002	Diesel	Manual	First	16.2 kmpl	1493 CC	110 bhp	5	Below 5L
10.42.0.211-119.6.229.36-42096-80-6	Toyota Innova 2.5 V Diesel 7-seater	Mumbai	2007	262000	Diesel	Manual	Fourth & Above	12.8 kmpl	2494 CC	102 bhp	7	Below 5L

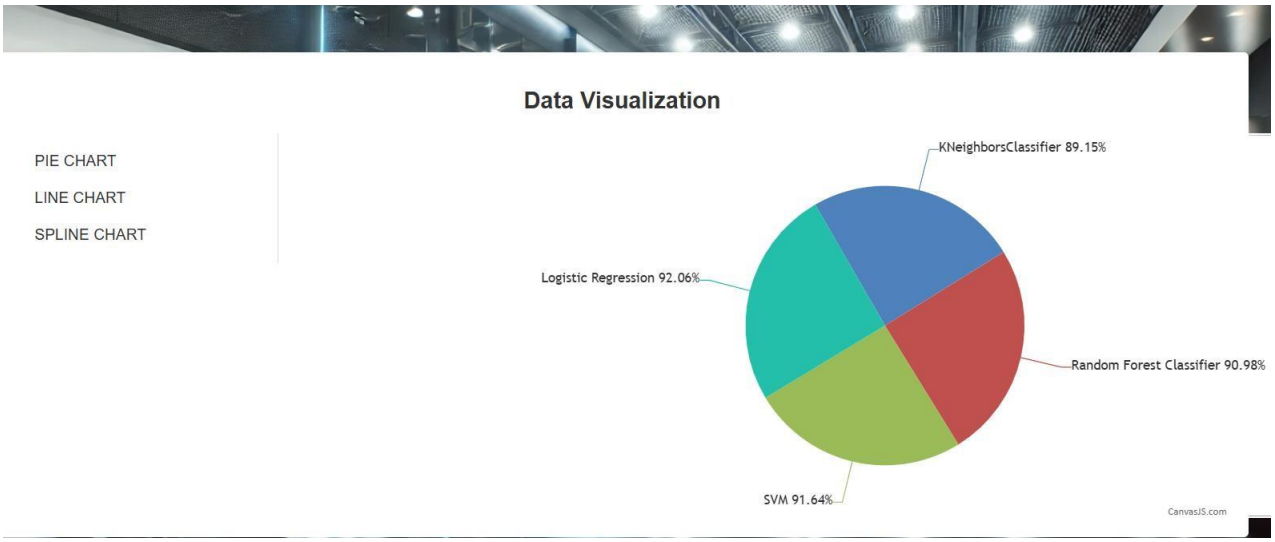
Data Visualization

Bar Graphs

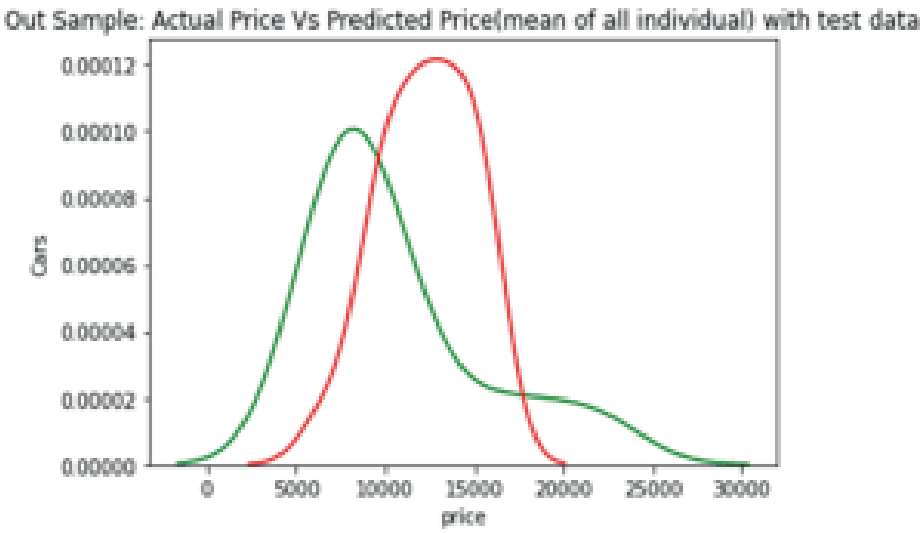
Line Graphs



Pie Charts



Spline Charts



CHAPTER - 11

CONCLUSION

&

FUTURE ENHANCEMENT

CONCLUSION

Conclusion

The project "Predicting the Price of Used Cars Using Machine Learning" successfully demonstrates how machine learning techniques can be utilized to estimate car prices based on various parameters such as model, year, fuel type, transmission, and mileage. By integrating multiple algorithms, including K-Nearest Neighbors, Support Vector Machine, Logistic Regression, and Random Forest, combined in a Voting Classifier, the system enhances accuracy and robustness in price prediction.

This system not only benefits individual car buyers and sellers but also assists dealerships, financial institutions, and policymakers in making well-informed decisions. The implementation of a user-friendly web application allows remote users to input vehicle details and receive real-time price range predictions. Additionally, service providers can manage large datasets, track user activities, and monitor prediction accuracy, ensuring transparency and reliability in the used car market.

Key Contributions:

- Accurate Price Prediction: The combination of multiple ML models improves predictive accuracy and reduces errors.
- User-Friendly Web Application: A responsive and intuitive interface enables users to interact with the system seamlessly.
- Efficient Dataset Management: Service providers can store, manage, and update datasets efficiently.
- Visual Analytics for Performance Analysis: Bar charts and graphical representations allow easy evaluation of model performance.
- Enhanced Decision-Making: Users benefit from transparent pricing information, supporting better financial and purchasing decisions.

This project showcases the potential of data-driven decision-making in the used car industry and provides a foundation for future enhancements.

Future Work

To improve accuracy and efficiency, the system can integrate advanced AI techniques like Deep Learning and Recurrent Neural Networks (RNNs) to better understand complex pricing patterns. Adding Explainable AI (XAI) will make the predictions more transparent, helping users understand why a car is valued at a certain price. These enhancements will ensure a smarter and more reliable pricing system for buyers and sellers.

Expanding data sources will further enhance prediction quality. Real-time data from online marketplaces, dealership inventories, and economic factors like fuel prices and government policies can provide more accurate, market-driven pricing. Including details like service history, accident records, and regional trends will make predictions even more precise and useful.

Performance optimization is another key area. Techniques like hyperparameter tuning and automated model selection will help identify the best-performing models at any given time. A cloud-based system and a mobile-friendly app will make price predictions accessible from anywhere, helping users get real-time insights whether they are at home or at a dealership.

Finally, ensuring trust and security will be crucial. Blockchain technology can help maintain data integrity, preventing fraud and manipulation. A user feedback system can continuously refine predictions, making them more aligned with real-world market trends. These enhancements will make the system more accurate, scalable, and user-friendly, benefiting both individual buyers and businesses.

CHAPTER – 12
REFERENCES

BIBLIOGRAPHY

References

[1]NATIONAL TRANSPORT AUTHORITY OF INDIA ASSOCIATION. 2015. Available at: http://nta.govmu.org/English/Statistics/Pages/Arch ives.aspx. [Accessed 24 April 2015].

[2] Bharambe, M. M. P., and Dharmadhikari, S. C. (2015) "Stock Market Analysis Based on Artificial Neural Network with Big data". *Fourth Post Graduate Conference, 24-25th March 2015, Pune, India.*

[3] Pudaruth, S. (2014) "Predicting the Price of Used Cars using Machine Learning Techniques". *International Journal of Information & Computation Technology,* Vol. 4, No. 7, pp.753- 764.

[4] Jassibi, J., Alborzi, M. and Ghoreshi, F. (2011) "Car Paint Thickness Control using Artificial Neural Network and Regression Method". *Journal of Industrial Engineering International*, Vol. 7, No. 14, pp. 1-6, November 2010

[5] Ahangar, R. G., Mahmood and Y., Hassen P.M. (2010) "The Comparison of Methods, Artificial
Neural Network with Linear Regression using Specific Variables for Prediction Stock Prices in
Tehran Stock Exchange". International Journal of Computer Science and Information Security,
Vol.7, No. 2, pp. 38-46.

[6] Listiani, M. (2009) "Support Vector Regression Analysis for Price Prediction in a Car Leasing Application". Thesis (MSc). Hamburg University of Technology.

[7] Iseri, A. and Karlik, B. (2009) "An Artificial Neural Network Approach on Automobile Pricing". Expert Systems with Application: ScienceDirect Journal of Informatics, Vol. 36, pp. 155-2160, March 2009.

[8] Yeo, C. A. (2009) "Neural Networks for Automobile Insurance Pricing". Encyclopedia of Information Science and Technology, 2nd Edition,
pp. 2794-2800, Australia.

[9] Doganis, P., Alexandridis, A., Patrinos, P. and
Sarimveis, H. (2006) “Time Series Sales Forecasting for Short Shelf-life Food Products
Based on Artificial Neural Networks and Evolutionary Computing”. Journal of Food Engineering, Vol. 75, pp. 196–204.

[10] Rose, D. (2003) “Predicting Car Production using a Neural Network Technical Paper- Vetronics (Inhouse)”.
Thesis, U.S. Army Tank Automotive Research, Development and Engineering Center (TARDEC).

[11] LEXPRESS.MU ONLINE. 2014. [Online] Available at: http://www.lexpress.mu/ [Accessed
23 September 2014].

[12] LE DEFI MEDIA GROUP. 2014. [Online] Available at: http://www.defimedia.info/ [Accessed 23 September 2014].

[13] He, Q. (1999) “Neural Network and its
Application in IR”. Thesis (BSc). University of Illinois.

[14] Cheng, B. and Titterington, D. M. (1994). “Neural Networks: A Review from a Statistical Perspective”. Statistical Science, Vol. 9, pp. 2-54.