

A PROJECT REPORT
ON
PREDICTING THE PRICE OF USED CARS USING MACHINE LEARNING
TECHNIQUES

Submitted in partial fulfillment of the requirements for the award of degree in

MASTER OF COMPUTER APPLICATIONS

SUBMITTED BY

BAPANAPALLI NARASIMHAM

Regd. No: **K6235208**

UNDER THE GUIDANCE

OF

SHAMIM

MCA M.Tech AP&TSSET

ASSISTANT PROFESSOR



PG DEPARTMENT OF COMPUTER SCIENCE & APPLICATIONS

ISO9001-2015 Certified

Re-accredited 'A⁺⁺' by

NAAC

KAKARAPARTI BHAVANARAYANA COLLEGE (AUTONOMOUS)

(Approved by AICTE, Affiliated to KRISHNA UNIVERSITY, MACHILIPATNAM)

Kothapet, Vijayawada, Krishna (District), pincode-520001

2023-2025

ISO:9001-2015Certified

Re-accredited 'A⁺⁺' by

**NAAC KAKARAPARTI BHAVANARAYANA PG
COLLEGE(AUTONOMOUS)**

(Approved by AICTE, Affiliated to KRISHNA UNIVERSITY, MACHILIPATNAM)

Kothapet, Vijayawada, Krishna (District), pincode-520001

PG DEPARTMENT OF COMPUTER SCIENCE & APPLICATIONS



CERTIFICATE

This is to certify that this work entitled “**PREDICTING THE PRICE OF USED CARS USING MACHINE LEARNING TECHNIQUES**” is Bonafide work carried out by **BAPANAPALLI NARASIMHAM(2305008)** in the partial fulfillment for the award of the degree in **MASTER OF COMPUTER APPLICATION** of **KRISHNA UNNIVERSITY , MACHILIPATNAM** during the Academic year 2023-2025. It is certify that the corrections/ Suggestions indicated for internal assessment have been incorporated in the report. The project work has been approved satisfies the academic requirements in respect of project work prescribed for the above degree.

Project Guide

Head of the Department

External Examiner

ACKNOWLEDGMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without mentioning the people who made it possible and whose constant guidance and encouragement crown all the efforts with success. This acknowledgement transcends the reality of formality when we would like to express deep gratitude and respect to all those people behind the screen who guided, inspired and helped me for the completion of the work. I wish to place on my record my deep sense gratitude to my project guide, **SHAMIM MCA M.TECH AP&TSSET ASSISTANT PROFESSOR Department of MCA** for his constant motivation and valuable help throughout the project work.

My sincere thanks to **Dr. V T Ram Pavan Kumar M.Tech AP&TS-SET UGCNET Ph.D., (PDF) Head of the Department of MCA** for his guidance regarding the project. I extend gratitude to **Dr. S. VENKATESH, DIRECTOR for P.G. COURSES** for his valuable suggestions.

BAPANAPALLI NARASIMHAM

K6235208

DECLARATION

I hereby declare the project work entitled “**PREDICTING THE PRICE OF USED CARS USING MACHINE LEARNING TECHNIQUES**” submitted to K.B.N P.G COLLEGE affiliated to KRISHNA UNIVERSITY, has been done under the guidance of **SHAMIM MCA M.TECH AP&TSSET ASSISTANT PROFESSOR** Department of MCA during the period of study in that it has found formed the basis for the award of the degree/diploma or other similar title to any candidate of University.

SIGNATURE OF THE STUDENT

NAME: BAPANAPALLI NARASIMHAM

REGD NO: K6235208

COLLEGE NAME: KBN COLLEGE

DATE:

PLACE:VIJAYAWADA

ABSTRACT

In this paper, we investigate the application of supervised machine learning techniques to predict the price of used cars in Mauritius. The predictions are based on historical data collected from daily newspapers. Different techniques like Support Vector Machine (SVM), k-nearest neighbors(KNN) ,Logistic regression and Random Forest have been used to make the predictions. The predictions are then evaluated and compared in order to find those which provide the best performances. A seemingly easy problem turned out to be indeed very difficult to resolve with high accuracy. All the four methods provided comparable performance.

INDEX

S.NO	CONTENTS	PAGE NO
1.	INTRODUCTION	1
2.	LITERATURE SURVEY	2-3
3.	PROBLEM ANALYSIS	4-9
4.	SYSTEM DESIGN	10-14
5.	IMPLEMETATION	15-40
6.	TESTING	41-50
7.	SCREEN SHOTS	51-57
8.	CONCLUSION	58-60
9.	REFERENCES	61

CHAPTER-1

INTRODUCTION:

Predicting the price of used cars is a complex task that involves analyzing various factors influencing car values, such as make, model, year of manufacture, mileage, condition, and market trends. Traditional methods for estimating used car prices often rely on heuristic rules or manual appraisals, which can be subjective and inconsistent. With the advent of machine learning (ML) techniques, there is an opportunity to enhance the accuracy and reliability of price predictions by leveraging large datasets and sophisticated algorithms. Machine learning approaches can automatically learn from historical data, identify patterns, and make predictions with minimal human intervention, providing a more objective and data-driven solution for used car pricing.

CHAPTER-2

LITERATURE SURVEY:

1. Title: "Predicting Used Car Prices with Machine Learning Techniques: A Comparative Study"

Authors: J. S. Chong, J. B. Lee, and A. W. K. Lau

Description: This paper provides a comparative analysis of various machine learning techniques applied to the prediction of used car prices. The authors evaluate the performance of algorithms such as Linear Regression, Decision Trees, Random Forests, and Gradient Boosting Machines on a dataset of used car listings. By comparing model accuracy, interpretability, and computational efficiency, the study highlights the strengths and weaknesses of each technique. The findings emphasize the effectiveness of ensemble methods like Random Forests in capturing complex relationships in the data, while also discussing challenges such as overfitting and feature selection.

2. Title: "Feature Engineering for Predicting Used Car Prices: Insights from Machine Learning Models"

Authors: E. R. Martinez, H. R. Liao, and V. P. Rao

Description: This research focuses on the impact of feature engineering on the performance of machine learning models for predicting used car prices. The authors explore various feature selection and transformation techniques, including polynomial features, interaction terms, and domain-specific features like car make, model, and condition. The study demonstrates that thoughtful feature engineering can significantly enhance model performance, leading to more accurate price predictions.

3. Title: "Deep Learning Approaches for Predicting Used Car Prices: A Review and Comparative Analysis"

Authors: M. T. Yang, S. H. Kim, and F. B. Patel

Description: This review paper examines the application of deep learning techniques to the problem of predicting used car prices. The authors analyze various deep learning architectures, including feedforward neural networks, convolutional neural networks (CNNs), and recurrent neural networks (RNNs). They compare the performance of these models with traditional machine learning methods and highlight the advantages of deep learning in capturing non-linear relationships and interactions within the data. The paper also addresses the challenges of training deep learning models, such as overfitting and the need for large datasets.

4. Title: "A Hybrid Model for Predicting Used Car Prices: Combining Machine Learning and Statistical Techniques"

Authors: R. H. Gupta, N. A. Singh, and K. J. Wilson

Description: This study proposes a hybrid model that combines machine learning algorithms with statistical techniques for predicting used car prices. The authors integrate methods such as multiple linear regression with advanced machine learning approaches like Support Vector Machines (SVM) and XGBoost. The hybrid model aims to leverage the strengths of both statistical and machine learning methods to improve prediction accuracy. The paper provides a detailed evaluation of the hybrid approach's performance and compares it to standalone machine learning models, demonstrating enhanced prediction capabilities and robustness.

CHAPTER-3

PROBLEM ANALYSIS:

EXISTING SYSTEM:

Existing systems for predicting used car prices typically involve a combination of rule-based models and statistical methods. Commonly used approaches include linear regression, which estimates prices based on a linear relationship between car features and price, and heuristic methods that rely on expert knowledge and historical pricing trends. These methods, while useful, often face limitations such as inability to capture complex, non-linear relationships and difficulty in adapting to changes in market conditions. Additionally, traditional systems may suffer from issues related to data quality and feature selection, leading to less accurate predictions. Many existing systems also lack the ability to handle large-scale datasets and incorporate diverse sources of information, limiting their overall effectiveness.

EXISTING SYSTEM DISADVANTAGES:

1. LESS ACCURACY
2. LOW EFFICIENCY

Proposed System:

The proposed system aims to address the limitations of existing approaches by leveraging advanced machine learning techniques to predict used car prices with greater accuracy and robustness. The system will utilize a variety of ML algorithms, including decision trees, random forests, gradient boosting machines, and deep learning models, to analyze and predict car prices. By incorporating a comprehensive dataset that includes detailed information on car features, historical prices, and market trends, the proposed system will enhance prediction capabilities and provide more reliable price estimates.

Key innovations in the proposed system include:

Enhanced Feature Engineering: By incorporating advanced feature engineering techniques, the system will improve the representation of car attributes, such as transforming categorical features into numerical values and creating interaction terms that capture complex relationships.

Data Quality Improvement: The system will implement robust data preprocessing methods to handle missing values, outliers, and noise, ensuring that the data used for training and prediction is of high quality.

Hybrid Models: Combining different ML algorithms and statistical techniques will allow the system to leverage the strengths of each approach, resulting in a more accurate and generalizable model.

Scalability and Adaptability: The system will be designed to handle large datasets and adapt to evolving market conditions, making it suitable for dynamic and diverse used car markets.

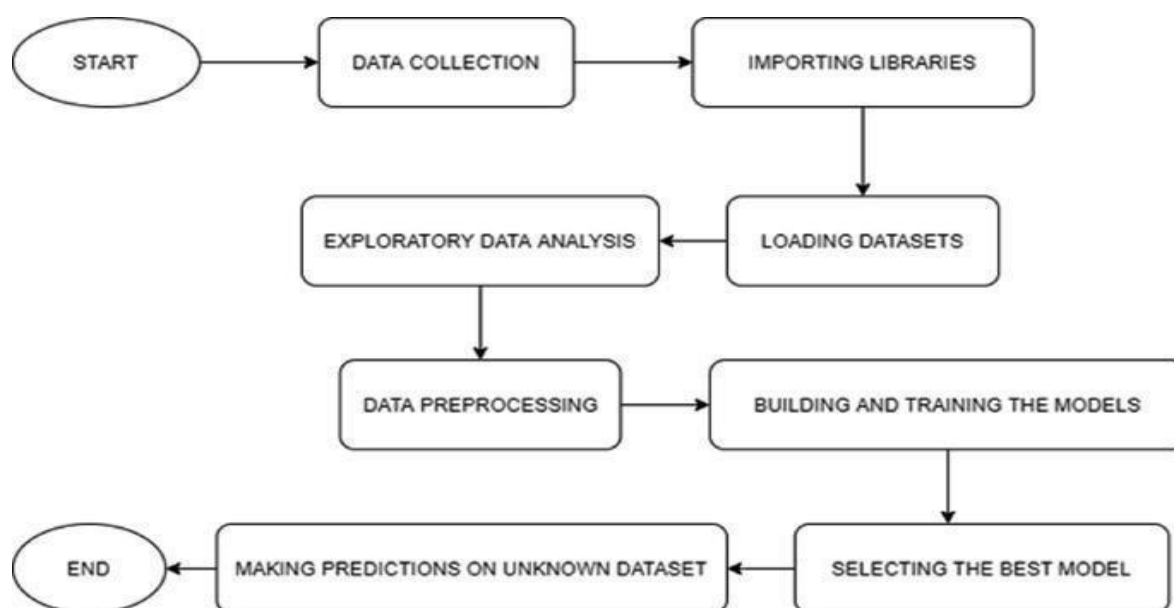
User Interface and Accessibility: A user-friendly interface will be developed to allow users to input car details and receive price predictions seamlessly, making the system accessible to both car buyers and sellers.

Overall, the proposed system aims to provide a sophisticated, data-driven solution for predicting used car prices, addressing the shortcomings of existing methods and offering enhanced accuracy and adaptability in a rapidly changing market.

PROPOSED SYSTEM ADVANTAGES:

1. HIGH ACCURACY
2. HIGH EFFICIENCY

SYSTEM ARCHITECYURE:



HARDWARE & SOFTWARE REQUIREMENTS:

HARD REQUIRMENTS :

- System : Pentium IV 2.4 GHz.
- Hard Disk : 40 GB.
- Floppy Drive : 1.44 Mb.
- Monitor : 15 VGA Colour.
- Mouse : Logitech.
- Ram : 512 MB.

SOFTWARE REQUIRMENTS :

- Operating system : Windows 8Professional.
- Coding Language : python

Functional Requirements:

1. Data Collection and Integration

The system must be capable of collecting and integrating data from various sources to provide accurate price predictions. This includes aggregating information from online car listings, historical pricing databases, and vehicle specifications. The system should support importing data in multiple formats and handle real-time data updates. Additionally, it must integrate with external platforms and APIs to ensure that the data remains current and relevant. The ability to process large volumes of data efficiently is essential for training and refining the machine learning models.

2. Data Preprocessing and Cleaning

The system must include robust data preprocessing and cleaning functionalities to ensure the quality and consistency of the input data. This involves handling missing values, removing duplicates, and correcting errors in the dataset. The system should provide tools for normalizing and transforming data, such as converting categorical variables into numerical formats and scaling numerical features. Accurate preprocessing is crucial for building reliable machine

learning models, as it affects the quality of the predictions and the overall performance of the system.

3. Feature Engineering and Selection

The system should support advanced feature engineering and selection techniques to enhance the predictive power of the machine learning models. Users must be able to create new features, such as interaction terms or derived metrics, and evaluate their impact on model performance. The system should also offer automated feature selection methods to identify the most relevant features and eliminate redundant or irrelevant ones. Effective feature engineering and selection are vital for improving the accuracy and efficiency of the predictions.

4. Model Training and Evaluation

The system must provide functionalities for training and evaluating machine learning models. This includes selecting appropriate algorithms (e.g., linear regression, decision trees, random forests, gradient boosting) and configuring their parameters. The system should support model training on historical data and offer tools for hyperparameter tuning to optimize performance. Evaluation metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R-squared must be used to assess model accuracy. Cross-validation techniques should be employed to ensure that the models generalize well to new data.

5. Price Prediction and Inference

The core functionality of the system is to provide accurate price predictions based on user inputs. The system must allow users to enter details about a used car, such as make, model, year, mileage, and condition. The trained machine learning models should then generate price estimates in real-time. The system must ensure that predictions are provided promptly and accurately, with mechanisms to handle any exceptions or anomalies in the input data. The prediction results should be presented in a clear and comprehensible format.

6. User Interface and Experience

The system must feature an intuitive and user-friendly interface to facilitate interaction with the price prediction functionality. The interface should enable users to input car details easily and view predictions without requiring technical expertise. It should include form fields for entering relevant information, as well as options for reviewing and adjusting inputs. The design

should prioritize usability and accessibility, ensuring that users can navigate the system efficiently and receive actionable insights from the predictions.

7. Data Security and Privacy

The system must ensure the security and privacy of user data, complying with relevant regulations and standards. This includes implementing encryption for data storage and transmission, as well as access controls to prevent unauthorized access. The system should provide users with transparency about data handling practices and allow them to manage their data preferences. Ensuring data security and privacy is essential for maintaining user trust and protecting sensitive information.

8. Performance Monitoring and Maintenance

The system must include functionalities for monitoring performance and conducting regular maintenance. This involves tracking key metrics such as response times, prediction accuracy, and system uptime. The system should provide alerts for any performance issues or anomalies, enabling prompt resolution. Regular updates to the machine learning models, data sources, and system components are necessary to maintain optimal performance and address any emerging issues.

9. Integration with External Platforms

The system should be designed to integrate with various external platforms and applications. This includes car marketplaces, dealership management systems, and financial services platforms. Integration capabilities will allow for seamless data exchange and real-time updates, enhancing the overall functionality of the prediction system. The system must support standard data formats and protocols to facilitate smooth integration with these platforms.

10. Reporting and Analytics

The system should offer reporting and analytics features to provide insights into prediction performance and user interactions. This includes generating reports on model accuracy, data quality, and system usage. Analytics tools should enable users to track trends, identify areas for improvement, and make data-driven decisions. Providing comprehensive reporting and analytics will help stakeholders assess the effectiveness of the system and guide future enhancements.

SYSTEM STUDY FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- **ECONOMICAL FEASIBILITY**
- **TECHNICAL FEASIBILITY**
- **SOCIAL FEASIBILITY**

ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed .

CHAPTER – 4

SYSTEM DESIGN:

UML DIAGRAMS :

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

Modules :

1. Data Collection and Preprocessing Module

The Data Collection and Preprocessing module is fundamental for gathering and preparing the data required for machine learning models. This module involves collecting data from various sources, including online car listings, historical pricing records, and vehicle specifications. The data collected may include features such as make, model, year of manufacture, mileage, condition, and location. Preprocessing is crucial to ensure data quality and consistency; it includes tasks such as cleaning the data (removing duplicates, handling missing values), transforming categorical variables into numerical formats, and normalizing numerical features. This module ensures that the dataset is well-structured and suitable for feeding into machine learning algorithms.

2. Feature Engineering and Selection Module

The Feature Engineering and Selection module focuses on enhancing the predictive power of the machine learning models by creating and selecting relevant features. Feature engineering involves creating new variables or transforming existing ones to better capture the underlying patterns in the data. For example, it might include creating interaction terms between features or deriving new metrics such as average annual mileage. Feature selection, on the other hand, involves identifying and retaining the most important features while discarding irrelevant or redundant ones. This module aims to improve model performance by ensuring that the features used are both informative and efficient.

3. Model Training and Evaluation Module

The Model Training and Evaluation module is responsible for building and assessing the performance of machine learning models. This involves selecting appropriate algorithms (e.g., linear regression, decision trees, random forests, gradient boosting machines, deep learning models) and training them on the prepared dataset. The module also includes the process of tuning hyperparameters to optimize model performance. Evaluation metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R-squared are used to assess how well the model predicts used car prices. Cross-validation techniques may be employed to ensure that the model generalizes well to new, unseen data.

4. Prediction and Inference Module

Once the models are trained and evaluated, the Prediction and Inference module is responsible for making price predictions based on new data. Users can input details of a used car, and the module will use the trained model to estimate its price. This module is designed to provide real-time predictions and is often integrated into user-facing applications or platforms. It should be optimized for speed and accuracy to ensure that users receive timely and reliable price estimates. The module also includes functionalities for handling edge cases and ensuring that predictions remain within reasonable bounds.

5. User Interface and Interaction Module

The User Interface and Interaction module is critical for facilitating user interaction with the prediction system. This module involves designing and implementing user-friendly interfaces that allow users to input car details and view predicted prices easily. It includes features such as forms for entering car specifications, dashboards for displaying prediction results, and

visualization tools for analyzing historical price trends. The interface should be intuitive and accessible to users with varying levels of technical expertise.

6. Data Integration and API Module

The Data Integration and API module enables seamless interaction between the prediction system and external data sources or applications. This module provides APIs (Application Programming Interfaces) for integrating with other systems, such as online car marketplaces, dealership management systems, or financial services platforms. It facilitates the exchange of data and enables real-time updates to the prediction system based on new information. The module ensures that data flow between systems is secure and efficient, supporting the overall functionality and scalability of the prediction system.

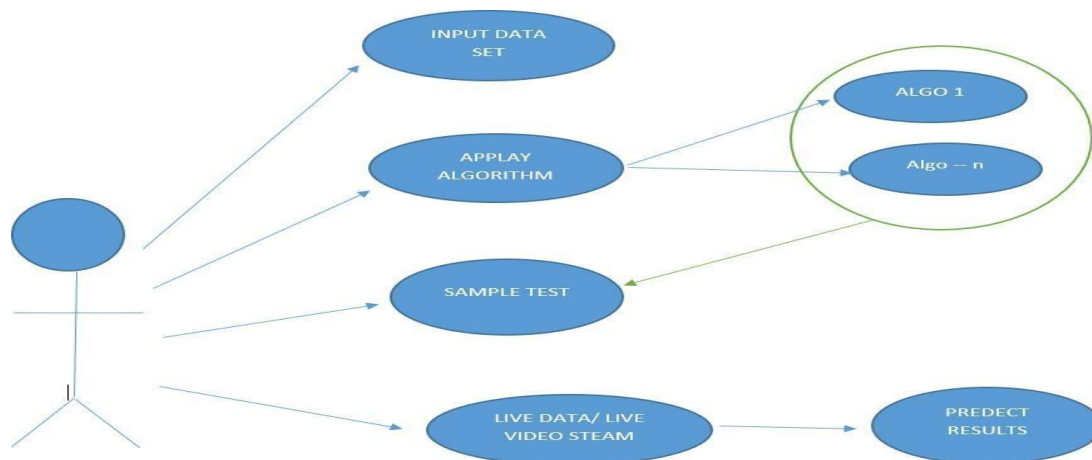
GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

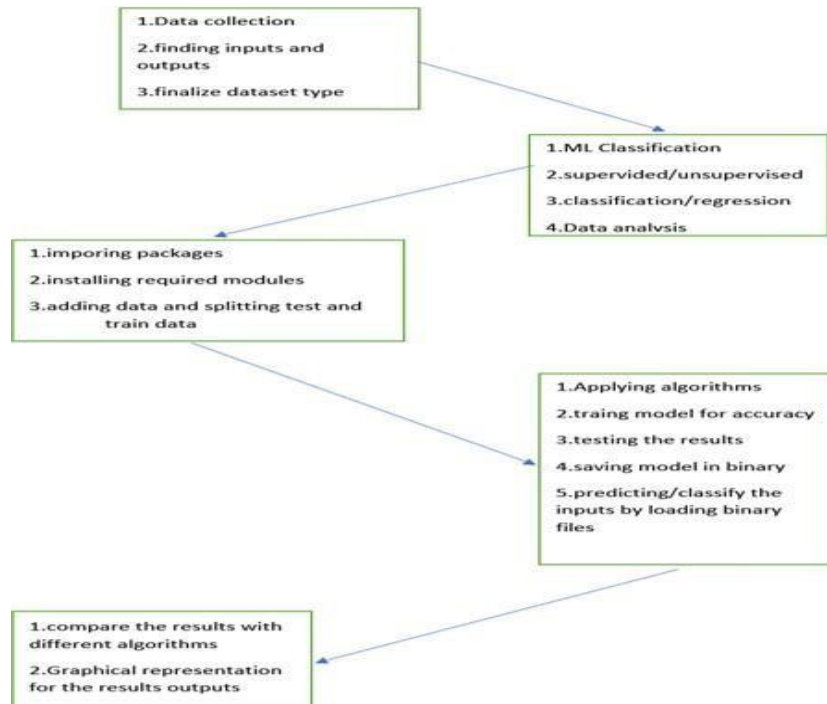
USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



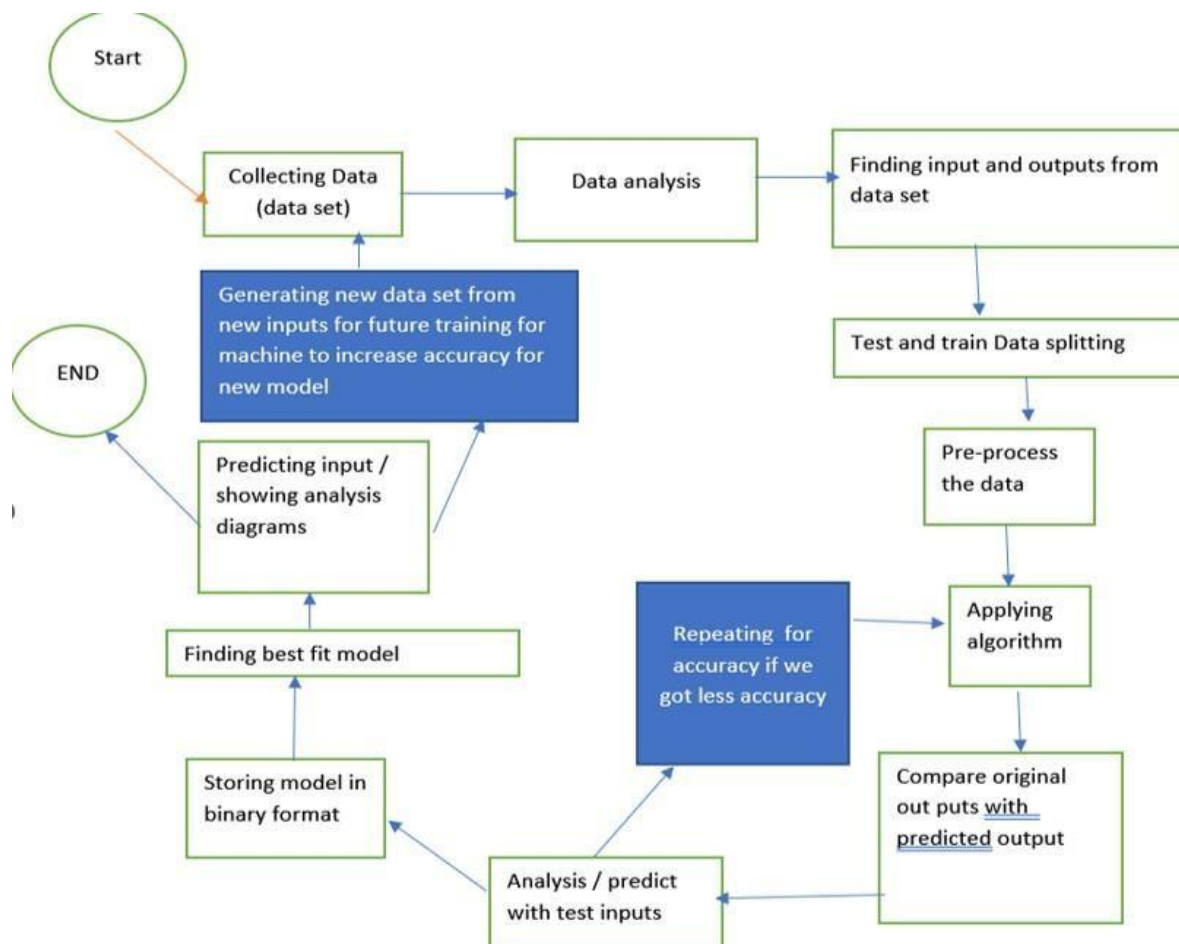
CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



SEQUENCE DIAGRAM:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



CHAPTER-5

IMPLEMENTATION:

implementing an automated resume analysis and skill-suggesting system using Natural Language Processing (NLP) involves several key components. Here's a structured approach to system implementation:

1. System Requirements

- **User Interface (UI):** For users to upload resumes and view suggestions.
- **Resume Parsing Engine:** To extract structured data from resumes.
- **NLP Engine:** To analyze text and suggest skills.
- **Database:** To store resumes, extracted information, and skill suggestions.
- **Backend:** To handle business logic, data processing, and integration between components.

2. System Design

Frontend

- **Upload Functionality:** Allow users to upload resumes in various formats (PDF, DOCX, etc.).
- **Display Results:** Show parsed resume data and suggested skills to the user.

Resume Parsing Engine

- **Document Extraction:** Use libraries like PyMuPDF, pdfminer.six, or docx2txt to extract text from resumes.
- **Data Extraction:** Extract structured information such as name, contact details, education, experience, and skills using pattern matching or rule-based approaches.

NLP Engine

- **Preprocessing:** Clean and preprocess text data (e.g., remove stop words, tokenize, and lemmatize).
- **Named Entity Recognition (NER):** Identify entities like job titles, organizations, and skills using libraries like spaCy or NLTK.
- **Skill Extraction:** Use techniques like keyword extraction, topic modeling, or embedding-based methods to suggest relevant skills.
- **Contextual Analysis:** Employ machine learning models (e.g., BERT, GPT) to understand the context and relevance of extracted skills.

Backend

- **Data Handling:** Implement APIs to manage data flow between frontend, parsing engine, and NLP engine.
- **Integration:** Combine parsing results and skill suggestions to present a cohesive output to the user.
- **Business Logic:** Implement rules or algorithms for skill suggestions based on extracted resume data.

Database

- **Storage:** Store resumes, parsed data, and skill suggestions in a database like PostgreSQL, MongoDB, or SQLite.
- **Search and Retrieval:** Implement efficient search and retrieval mechanisms for stored data.

3. Implementation Steps

Setup Environment

- Install necessary libraries and tools (e.g., spaCy, NLTK, PyMuPDF, Flask/Django).

Develop Resume Parsing Engine

- Implement functions to extract text from different resume formats.
- Design and develop algorithms to parse and structure resume data.

Build NLP Engine

- Train or fine-tune NLP models for skill extraction.
- Develop algorithms for contextual skill suggestion.

Create Frontend

- Design user interface and implement file upload and result display features.

Develop Backend

- Set up APIs for communication between frontend, parsing engine, and NLP engine.
- Implement data handling and integration logic.

Integrate Components

- Ensure seamless interaction between frontend, parsing engine, NLP engine, and database.

Testing

- Test the system with various resume formats and content to ensure accurate parsing and skill suggestion.
- Perform user testing to validate the UI/UX and functionality.

Deployment

- Deploy the system to a web server or cloud platform (e.g., AWS, Heroku).
- Ensure security measures are in place for user data protection.

4. Maintenance

- Regularly update NLP models to improve skill extraction accuracy.
- Monitor system performance and handle any issues or bugs.

This structured approach should guide you through implementing an automated resume analysis and skill-suggesting system using NLP. If you need more detailed information on any component or have specific questions, feel free to ask!

Scope :

Introduction and Context

The scope of predicting used car prices using machine learning techniques extends beyond merely estimating the value of a vehicle. It encompasses a broad range of applications that can benefit various stakeholders in the automotive market, including car buyers, sellers, dealers, and financial institutions. By leveraging advanced ML algorithms and data-driven insights, this approach aims to enhance the accuracy, efficiency, and objectivity of price predictions, offering significant advantages over traditional methods. The scope includes exploring the potential impact of machine learning on the used car market, addressing key challenges, and identifying opportunities for innovation.

Enhanced Prediction Accuracy

One of the primary areas of scope is improving the accuracy of used car price predictions. Machine learning techniques, such as ensemble methods and deep learning, can analyze complex datasets to identify patterns and relationships that traditional methods might overlook. This enhanced accuracy can lead to more reliable price estimates, benefiting both buyers and sellers by providing fairer and more consistent valuations. The scope also

involves comparing the performance of different ML algorithms to determine the most effective approaches for various types of vehicles and market conditions.

Comprehensive Data Utilization

The scope includes utilizing diverse and comprehensive datasets to inform price predictions. Machine learning models can integrate data from multiple sources, such as historical pricing records, car specifications, mileage, condition reports, and market trends. By incorporating various types of data, including textual descriptions and images, the models can gain a more holistic understanding of factors influencing car prices. The scope also involves addressing challenges related to data quality, such as handling missing values and outliers, to ensure the reliability of predictions.

Dynamic Market Adaptation

The dynamic nature of the used car market presents both opportunities and challenges for machine learning-based prediction systems. The scope includes developing models that can adapt to changing market conditions, such as fluctuations in demand, seasonal trends, and economic factors. Machine learning techniques can help capture and respond to these dynamics by continuously updating the models with new data and adjusting predictions accordingly. This adaptability ensures that the prediction system remains relevant and accurate in a constantly evolving market environment.

Integration with Market Platforms

Another important aspect of the scope is the integration of machine learning prediction systems with various market platforms and applications. This includes online car marketplaces, dealership management systems, and financial services platforms. By integrating prediction models with these platforms, users can access real-time price estimates and make informed decisions. The scope involves developing user-friendly interfaces and APIs that facilitate seamless integration and enhance the overall user experience.

User Experience and Accessibility

The scope also encompasses improving the user experience and accessibility of machine learning-based price prediction systems. This involves designing intuitive interfaces that allow users to input car details and receive predictions easily. The system should be accessible to a wide range of users, including those with limited technical expertise.

Future Research and Development

Finally, the scope includes identifying areas for future research and development in the field of machine learning-based price prediction. This includes exploring new algorithms, techniques, and data sources that could further enhance prediction accuracy and system performance. Research may also focus on addressing emerging challenges, such as incorporating real-time data from IoT devices or integrating advanced analytics for more nuanced insights. By advancing the state of the art in this field, future work can drive innovation and provide even greater value to stakeholders in the used car market.

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 Scrappy, BeautifulSoup, Selenium)
- Test frameworks
- Multimedia

ADAVANTAGES 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.

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**.

The reason it is not so famous despite the existence of Brython is that it isn't that secure.

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.

History of Python : -

What do the alphabet and the programming language Python have in common? Right, both start with ABC. If we are talking about ABC in the Python context, it's clear that the programming language ABC is meant. ABC is a general-purpose programming language and programming environment, which had been developed in the Netherlands, Amsterdam, at the CWI (Centrum Wiskunde & Informatica). The greatest achievement of ABC was to influence the design of Python. Python was conceptualized in the late 1980s. Guido van Rossum worked that time in a project at the CWI, called Amoeba, a distributed operating system. In an interview with Bill Venners¹, Guido van Rossum said: "In the early 1980s, I worked as an implementer on a team building a language called ABC at Centrum voor Wiskunde en Informatica (CWI). I don't know how well people know ABC's influence on Python. I try to mention ABC's influence because I'm indebted to everything I learned during that project and to the people who worked on it." Later on in the same Interview, Guido van Rossum continued: "I remembered all my experience and some of my frustration with ABC. I decided to try to design a simple scripting language that possessed some of ABC's better properties, but without its problems. So I started typing. I created a simple virtual machine, a simple parser, and a simple runtime. I made my own version of the various ABC parts that I liked. I created a basic syntax, used indentation for statement grouping instead of curly braces or begin-end blocks, and developed a small number of powerful data types: a hash table (or dictionary, as we call it), a list, strings, and numbers."

What is Machine Learning : -

Before we take a look at the details of various machine learning methods, let's start by looking at what machine learning is, and what it isn't. Machine learning is often categorized as a subfield of artificial intelligence, but I find that categorization can often be misleading at first brush. The study of machine learning certainly arose from research in this context,

but in the data science application of machine learning methods, it's more helpful to think of machine learning as a means of building models of data.

Fundamentally, machine learning involves building mathematical models to help understand data. "Learning" enters the fray when we give these models tunable parameters that can be adapted to observed data; in this way the program can be considered to be "learning" from the data. Once these models have been fit to previously seen data, they can be used to predict and understand aspects of newly observed data. I'll leave to the reader the more philosophical digression regarding the extent to which this type of mathematical, model-based "learning" is similar to the "learning" exhibited by the human brain. Understanding the problem setting in machine learning is essential to using these tools effectively, and so we will start with some broad categorizations of the types of approaches we'll discuss here.

Categories Of Machine Learning :-

At the most fundamental level, machine learning can be categorized into two main types: supervised learning and unsupervised learning.

Supervised learning involves somehow modeling the relationship between measured features of data and some label associated with the data; once this model is determined, it can be used to apply labels to new, unknown data. This is further subdivided into classification tasks and regression tasks: in classification, the labels are discrete categories, while in regression, the labels are continuous quantities. We will see examples of both types of supervised learning in the following section.

Unsupervised learning involves modeling the features of a dataset without reference to any label, and is often described as "letting the dataset speak for itself." These models include tasks such as clustering and dimensionality reduction. Clustering algorithms identify distinct groups of data, while dimensionality reduction algorithms search for more succinct representations of the data. We will see examples of both types of unsupervised learning in the following section.

Need for Machine Learning

Human beings, at this moment, are the most intelligent and advanced species on earth because they can think, evaluate and solve complex problems. On the other side, AI is still in its initial stage and haven't surpassed human intelligence in many aspects. Then the question is that what is the need to make machine learn? The most suitable reason for doing this is, "to make decisions, based on data, with efficiency and scale".

Lately, organizations are investing heavily in newer technologies like Artificial Intelligence, Machine Learning and Deep Learning to get the key information from data to perform several real-world tasks and solve problems. We can call it data-driven decisions taken by machines, particularly to automate the process. These data-driven decisions can be used, instead of using programming logic, in the problems that cannot be programmed inherently. The fact is that we can't do without human intelligence, but other aspect is that we all need to solve real-world problems with efficiency at a huge scale. That is why the need for machine learning arises.

Challenges in Machine Learning :-

While Machine Learning is rapidly evolving, making significant strides with cybersecurity and autonomous cars, this segment of AI as whole still has a long way to go. The reason behind is that ML has not been able to overcome number of challenges. The challenges that ML is facing currently are

Quality of data – Having good-quality data for ML algorithms is one of the biggest challenges. Use of low-quality data leads to the problems related to data preprocessing and feature extraction.

Time-Consuming task – Another challenge faced by ML models is the consumption of time especially for data acquisition, feature extraction and retrieval.

Lack of specialist persons – As ML technology is still in its infancy stage, availability of expert resources is a tough job.

No clear objective for formulating business problems – Having no clear objective and well-defined goal for business problems is another key challenge for ML because this technology is not that mature yet.

Issue of overfitting & underfitting – If the model is overfitting or underfitting, it cannot be represented well for the problem.

Curse of dimensionality – Another challenge ML model faces is too many features of data points. This can be a real hindrance.

Difficulty in deployment – Complexity of the ML model makes it quite difficult to be deployed in real life.

Applications of Machines Learning :-

Machine Learning is the most rapidly growing technology and according to researchers we are in the golden year of AI and ML. It is used to solve many real-world complex problems which cannot be solved with traditional approach. Following are some real-world applications of ML –

- Emotion analysis
- Sentiment analysis
- Error detection and prevention
- Weather forecasting and prediction
- Stock market analysis and forecasting
- Speech synthesis
- Speech recognition
- Customer segmentation
- Object recognition
- Fraud detection
- Fraud prevention
- Recommendation of products to customer in online shopping.

How to Start Learning Machine Learning?

Arthur Samuel coined the term “**Machine Learning**” in 1959 and defined it as a “**Field of study that gives computers the capability to learn without being explicitly programmed**”.

And that was the beginning of Machine Learning! In modern times, Machine Learning is one of the most popular (if not the most!) career choices. According to Indeed, Machine Learning Engineer Is The Best Job of 2019 with a 344% growth and an average base salary of **\$146,085** per year.

But there is still a lot of doubt about what exactly is Machine Learning and how to start learning it? So this article deals with the Basics of Machine Learning and also the path you can follow to eventually become a full-fledged Machine Learning Engineer. Now let's get started!!!

How to start learning ML?

This is a rough roadmap you can follow on your way to becoming an insanely talented Machine Learning Engineer. Of course, you can always modify the steps according to your needs to reach your desired end-goal!

Step 1 – Understand the Prerequisites

In case you are a genius, you could start ML directly but normally, there are some prerequisites that you need to know which include Linear Algebra, Multivariate Calculus, Statistics, and Python. And if you don't know these, never fear! You don't need a Ph.D. degree in these topics to get started but you do need a basic understanding.

(a) Learn Linear Algebra and Multivariate Calculus.

Both Linear Algebra and Multivariate Calculus are important in Machine Learning. However, the extent to which you need them depends on your role as a data scientist. If you are more focused on application heavy machine learning, then you will not be that heavily focused on maths as there are many common libraries available. But if you want to focus on R&D in Machine Learning, then mastery of Linear Algebra and Multivariate Calculus is very important as you will have to implement many ML algorithms from scratch.

(b) Learn Statistics

Data plays a huge role in Machine Learning. In fact, around 80% of your time as an ML expert will be spent collecting and cleaning data. And statistics is a field that handles the collection, analysis, and presentation of data. Some of the key concepts in statistics that are important are Statistical Significance, Probability Distributions, Hypothesis Testing, Regression, etc. Also, Bayesian Thinking is also a very important part of ML which deals with various concepts like Conditional Probability, Priors, and Posteriors, Maximum Likelihood, etc.

(c) Learn Python

Some people prefer to skip Linear Algebra, Multivariate Calculus and Statistics and learn them as they go along with trial and error. But the one thing that you absolutely cannot skip is Python! While there are other languages you can use for Machine Learning like R, Scala, etc. Python is currently the most popular language for ML. In fact, there are many Python libraries that are specifically useful for Artificial Intelligence and Machine Learning such as Keras, TensorFlow, Scikit-learn, etc.

So if you want to learn ML, it's best if you learn Python! You can do that using various online resources and courses such as **Fork Python** available Free on GeeksforGeeks.

Step 2 – Learn Various ML Concepts

Now that you are done with the prerequisites, you can move on to actually learning ML (Which is the fun part!!!) It's best to start with the basics and then move on to the more complicated stuff. Some of the basic concepts in ML are:

(a) Terminologies of Machine Learning

- **Model** – A model is a specific representation learned from data by applying some machine learning algorithm. A model is also called a hypothesis.
- **Feature** – A feature is an individual measurable property of the data. A set of numeric features can be conveniently described by a feature vector. Feature vectors are fed as input to the model. For example, in order to predict a fruit, there may be features like color, smell, taste, etc.
- **Target (Label)** – A target variable or label is the value to be predicted by our model. For the fruit example discussed in the feature section, the label with each set of input would be the name of the fruit like apple, orange, banana, etc.
- **Training** – The idea is to give a set of inputs(features) and it's expected outputs(labels), so after training, we will have a model (hypothesis) that will then map new data to one of the categories trained on.
- **Prediction** – Once our model is ready, it can be fed a set of inputs to which it will provide a predicted output(label).

(b) Types of Machine Learning

- **Supervised Learning** – This involves learning from a training dataset with labeled data using classification and regression models. This learning process continues until the required level of performance is achieved.
- **Unsupervised Learning** – This involves using unlabelled data and then finding the underlying structure in the data in order to learn more and more about the data itself using factor and cluster analysis models.
- **Semi-supervised Learning** – This involves using unlabelled data like Unsupervised Learning with a small amount of labeled data. Using labeled data vastly increases the learning accuracy and is also more cost-effective than Supervised Learning.
- **Reinforcement Learning** – This involves learning optimal actions through trial and error. So the next action is decided by learning behaviors that are based on the current state and that will maximize the reward in the future.

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. For instance, for an e-commerce website like Amazon, it serves to understand the browsing behaviors and purchase histories of its users to help cater to the right products, deals, and reminders relevant to them. 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 softwares; 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.

4. Handling multi-dimensional and multi-variety data

Machine Learning algorithms are good at handling data that are multi-dimensional and multi-variety, and they can do this in dynamic or uncertain environments.

5. Wide Applications

You could be an e-tailer or a healthcare provider and make ML work for you. Where it does apply, it holds the capability to help deliver a much more personal experience to customers while also targeting the right customers.

Disadvantages of Machine Learning :-

1. Data Acquisition

Machine Learning requires massive data sets to train on, and these should be inclusive/unbiased, and of good quality. There can also be times where they must wait for new data to be generated.

2. Time and Resources

ML needs enough time to let the algorithms learn and develop enough to fulfill their purpose with a considerable amount of accuracy and relevancy. It also needs massive resources to function. This can mean additional requirements of computer power for you.

3. Interpretation of Results

Another major challenge is the ability to accurately interpret results generated by the algorithms. You must also carefully choose the algorithms for your purpose.

4. High error-susceptibility

Machine Learning is autonomous but highly susceptible to errors. Suppose you train an algorithm with data sets small enough to not be inclusive. You end up with biased predictions coming from a biased training set. This leads to irrelevant advertisements being displayed to customers. In the case of ML, such blunders can set off a chain of errors that can go undetected

for long periods of time. And when they do get noticed, it takes quite some time to recognize the source of the issue, and even longer to correct it.

Python Development Steps :-

Guido Van Rossum published the first version of Python code (version 0.9.0) at alt.sources in February 1991. This release included already exception handling, functions, and the core data types of list, dict, str and others. It was also object oriented and had a module system. Python version 1.0 was released in January 1994. The major new features included in this release were the functional programming tools lambda, map, filter and reduce, which Guido Van Rossum never liked. Six and a half years later in October 2000, Python 2.0 was introduced. This release included list comprehensions, a full garbage collector and it was supporting unicode. Python flourished for another 8 years in the versions 2.x before the next major release as Python 3.0 (also known as "Python 3000" and "Py3K") was released. Python 3 is not backwards compatible with Python 2.x. The emphasis in Python 3 had been on the removal of duplicate programming constructs and modules, thus fulfilling or coming close to fulfilling the 13th law of the Zen of Python: "There should be one -- and preferably only one -- obvious way to do it." Some changes in Python 3:

- Print is now a function
- Views and iterators instead of lists
- The rules for ordering comparisons have been simplified. E.g. a heterogeneous list cannot be sorted, because all the elements of a list must be comparable to each other.
- There is only one integer type left, i.e. int. long is int as well.
- The division of two integers returns a float instead of an integer. "/" can be used to have the "old" behaviour.
- Text Vs. Data Instead Of Unicode Vs. 8-bit

Purpose :-

We demonstrated that our approach enables successful segmentation of intra-retinal layers—even with low-quality images containing speckle noise, low contrast, and different intensity ranges throughout—with the assistance of the ANIS feature.

Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

- **Python is Interpreted** – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** – you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

Modules Used in Project :-

Tensorflow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

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 data-types 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.

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

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**

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

- **Python is Interpreted** – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** – you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

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 3.7.4	July 8, 2019	 Download	Release Notes
Python 3.6.9	July 2, 2019	 Download	Release Notes
Python 3.7.3	March 25, 2019	 Download	Release Notes
Python 3.4.10	March 18, 2019	 Download	Release Notes
Python 3.5.7	March 18, 2019	 Download	Release Notes
Python 2.7.16	March 4, 2019	 Download	Release Notes
Python 3.7.2	Dec. 24, 2018	 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	GPU
Gzipped source tarball	Source release		68111071e5b2db4ae77b9ab01b7079be	23017663	3x6
XZ compressed source tarball	Source release		d33e4aa66097051c3eca45ee3604803	17131432	3x6
macOS 64-bit/32-bit installer	Mac OS X	for Mac OS X 10.5 and later	6428b4fa7583daf1a4c2cba0ce08e6	34898416	3x6
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later	5d805c38217a45773b5ea936b2a1f	28882845	3x6
Windows .hug file	Windows		06399573a2e96b2ac50ade0b471d2	8131761	3x6
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	9809c3c7d2ee0b0a0e2184a0728a2	7504381	3x6
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	a702b4b0a7f6d6b0b3043a583e563e00	26881948	3x6
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	28c31c5080b0f73ae0e53a3b0351b4bd2	1362904	3x6
Windows x86 embeddable zip file	Windows		9fa03b0198a1279fda94132574139d0	6741628	3x6
Windows x86 executable installer	Windows		33c3802942a544a3d8451479294789	25663848	3x6
Windows x86 web-based installer	Windows		1b670cf0d117d802c30983ea371d07c	1324608	3x6

• 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.

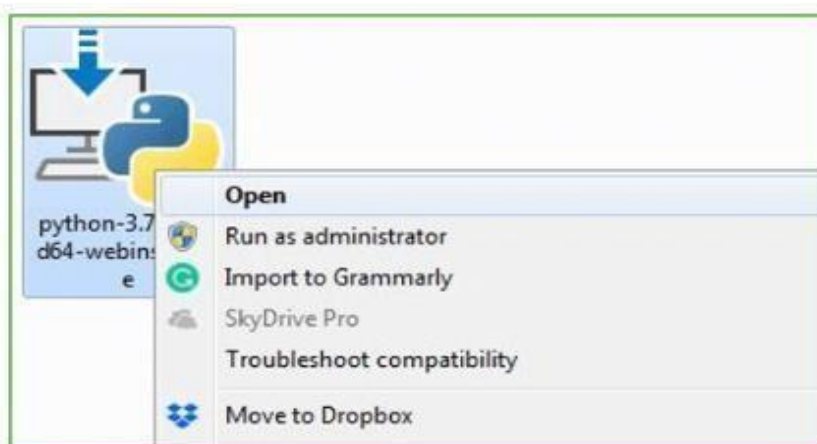
• To download Windows 64-bit python, you can select any one from the three options: Windows x86-64 embeddable zip file, Windows x86-64 executable installer or Windows x86-64 web-based installer.

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.



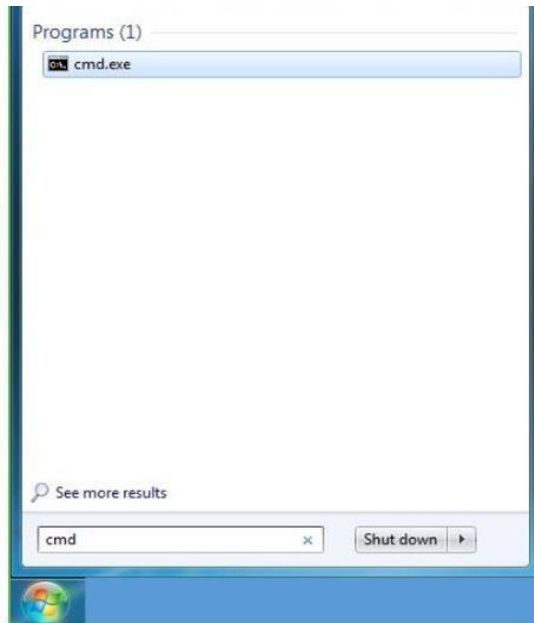
With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.

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.



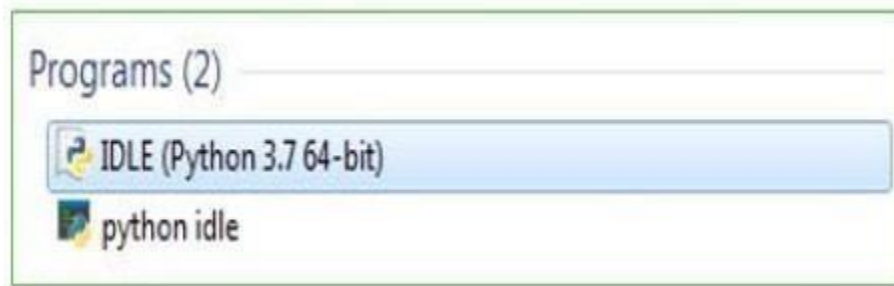
Step 5: You will get the answer as 3.7.4

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

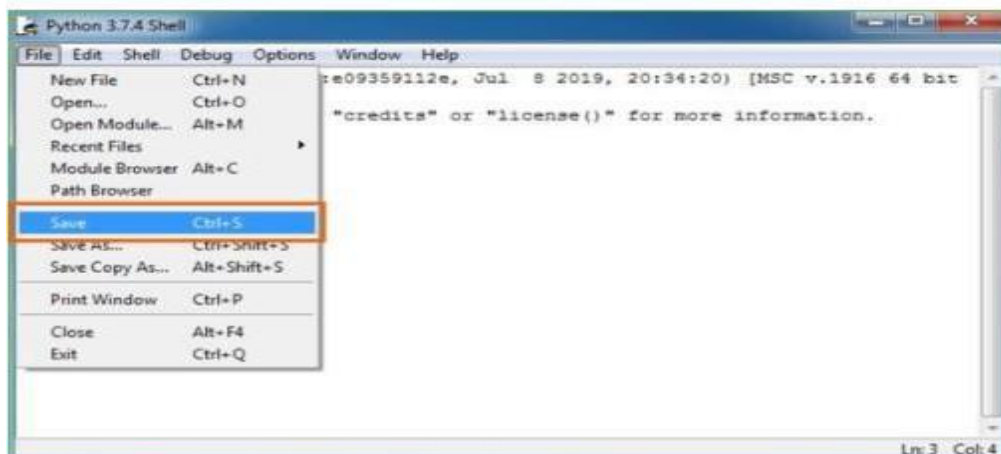
Step 1: Click on Start

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.

CHAPTER-6

TESTING:

Machine Learning Testing

First of all, what are we trying to achieve when performing ML testing, as well as any software testing whatsoever?

- Quality assurance is required to make sure that the software system works according to the requirements. Were all the features implemented as agreed? Does the program behave as expected? All the parameters that you test the program against should be stated in the technical specification document.
- Moreover, software testing has the power to point out all the defects and flaws during development. You don't want your clients to encounter bugs after the software is released and come to you waving their fists. Different kinds of testing allow us to catch bugs that are visible only during runtime.

However, in machine learning, a programmer usually inputs the data and the desired behaviour, and the logic is elaborated by the machine. This is especially true for deep learning. Therefore, the purpose of machine learning testing is, first of all, to ensure that this learned logic will remain consistent, no matter how many times we call the program.

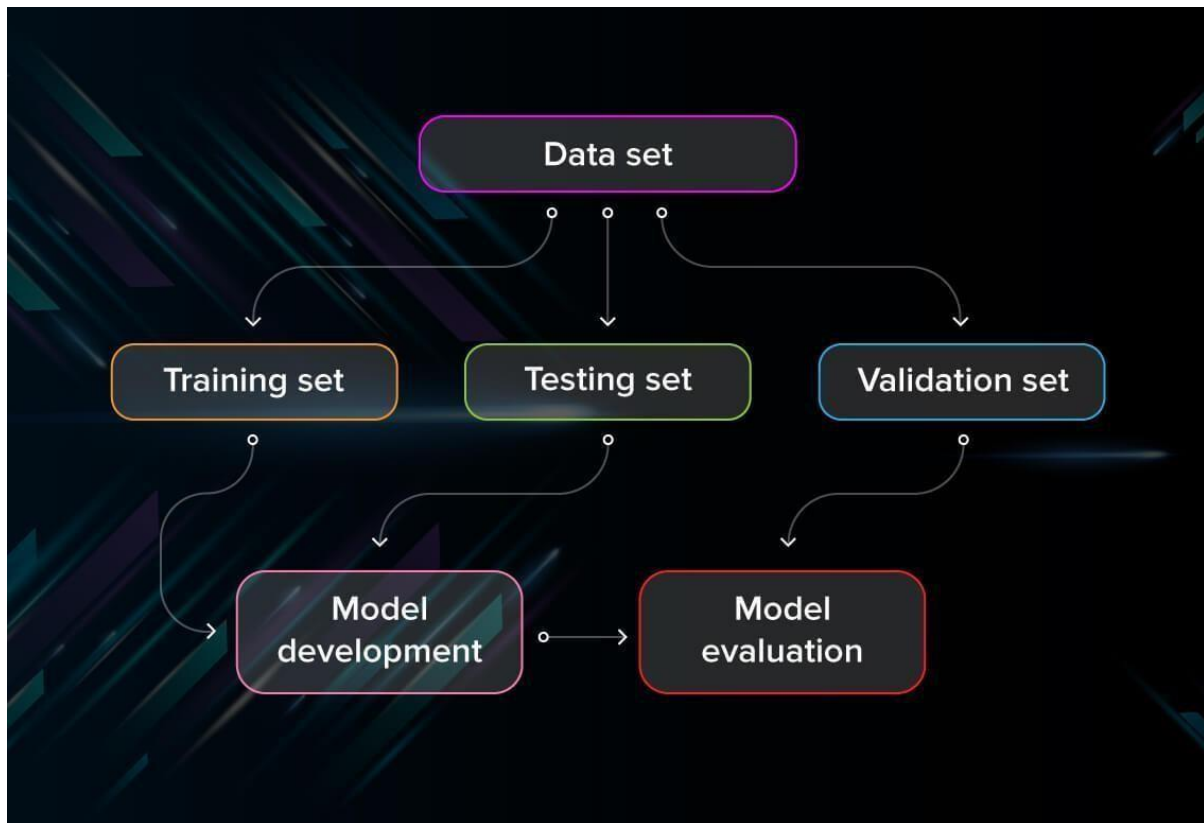
Model evaluation in machine learning testing

Usually, software testing includes:

- **Unit tests:** The program is broken down into blocks, and each element (unit) is tested separately.
- **Regression tests:** They cover already tested software to see if it doesn't suddenly break.
- **Integration tests:** This type of testing observes how multiple components of the program work together.

Moreover, there are certain rules that people follow: don't merge the code before it passes all the tests, always test newly introduced blocks of code, when fixing bugs, write a test that captures the bug.

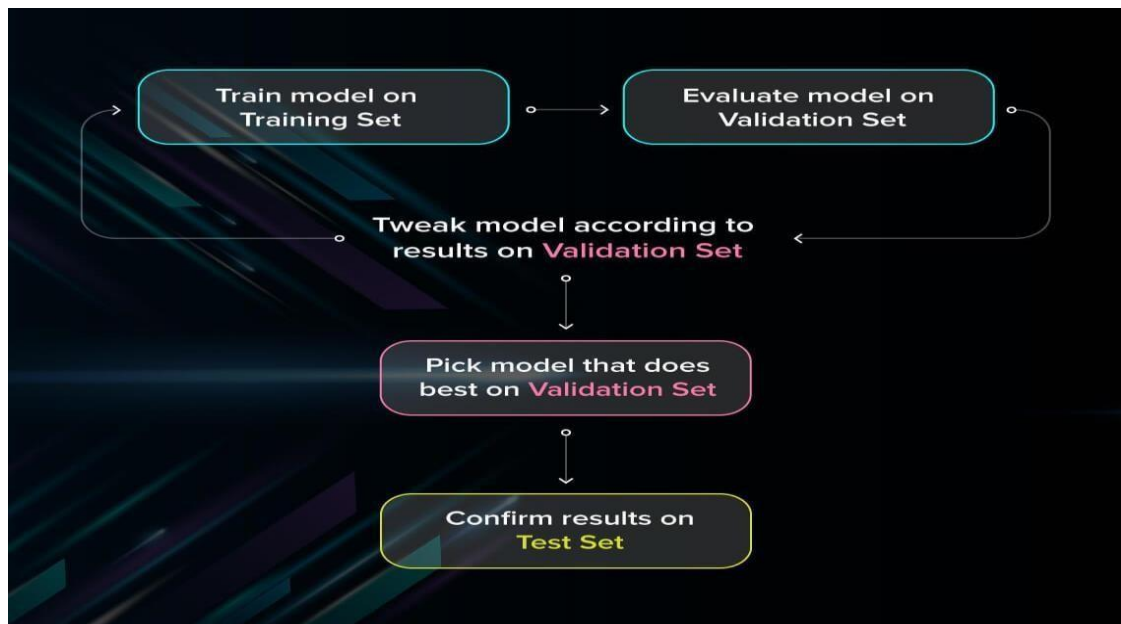
Machine learning adds up more actions to your to-do list. You still need to follow [ML's best practices](#). Moreover, every ML model needs not only to be tested but **evaluated**. Your model should generalize well. This is not what we usually understand by testing, but evaluation is needed to make sure that the performance is satisfactory.



First of all, you split the database into three non-overlapping sets. You use a training set to train the model. Then, to evaluate the performance of the model, you use two sets of data:

- **Validation set:** Having only a training set and a testing set is not enough if you do many rounds of hyperparameter-tuning (which is always). And that can result in overfitting. To avoid that, you can select a small validation data set to evaluate a model. Only after you get maximum accuracy on the validation set, you make the testing set come into the game.
- **Test set (or holdout set).** Your model might fit the training dataset perfectly well. But where are the guarantees that it will do equally well in real-life? In order to assure that, you select samples for a testing set from your training set — examples that the machine hasn't seen before. It is important to remain unbiased during selection and draw samples at random. Also, you should not use the same set many times to avoid training on your

test data. Your test set should be large enough to provide statistically meaningful results and be representative of the data set as a whole.



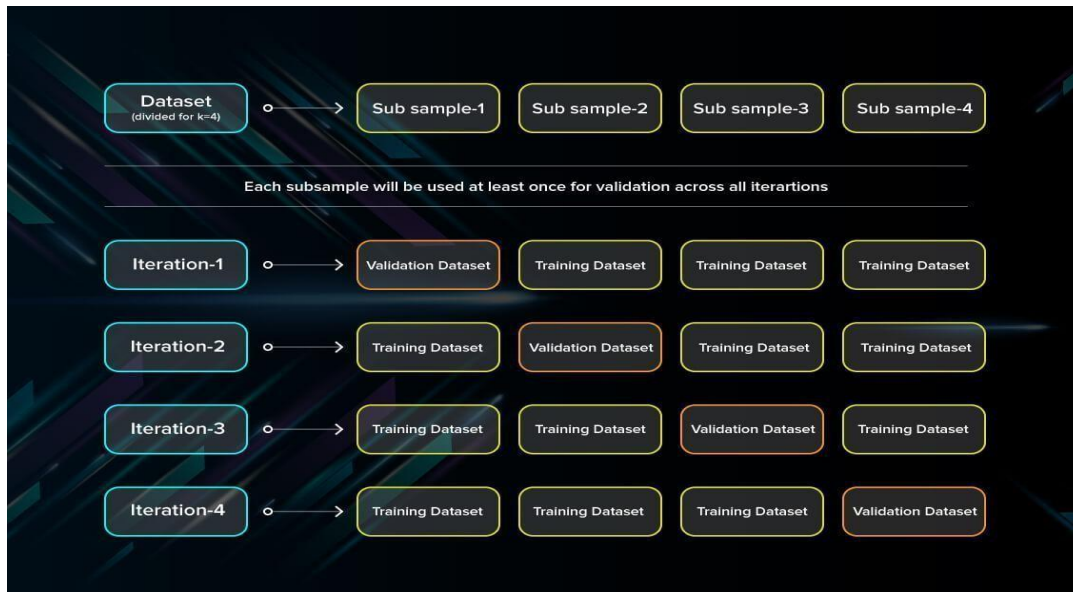
But just as test sets, validation sets “wear out” when used repeatedly. The more times you use the same data to make decisions about hyperparameter settings or other model improvements, the less confident you are that the model will generalize well on new, unseen data. So it is a good idea to collect more data to ‘freshen up’ the test set and validation set.

Cross-validation

Cross-validation is a model evaluation technique that can be performed even on a limited dataset. The training set is divided into small subsets, and the model is trained and validated on each of these samples.

k-fold cross-validation

The most common cross-validation method is called k-fold cross-validation. To use it, you need to divide the dataset into k subsets (also called folds) and use them k times. For example, by breaking the dataset into 10 subsets, you will perform a 10-fold cross-validation. Each subset must be used as the validation set at least once.



This method is useful to test the skill of the machine learning model on unseen data. It is so popular because it is simple to apply, works well even with relatively small datasets, and the results you get are generally quite accurate. If you want to learn more about how to cross-validate the model.

Leave-one-out cross-validation

In this method, we train the model on all the data samples in the set except for one data point that is used to test the model. By repeating this process iteratively, each time leaving a different data point as a testing set, you get to test the performance for all the data.

The benefit of the method is low bias since all the data points are used. However, it also leads to higher variation in testing because we are testing the model against just one data point each time.

Evaluate models using metrics

Evaluating the performance of the model using different metrics is integral to every data science project. Here is what you have to keep an eye on:

Accuracy

Accuracy is a metric for how much of the predictions the model makes are true. The higher the accuracy is, the better. However, it is not the only important metric when you estimate the performance.

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{True Positives} + \text{False Positives} + \text{True Negatives} + \text{False Negatives}}$$

Loss

Loss describes the percentage of bad predictions. If the model's prediction is perfect, the loss is zero; otherwise, the loss is greater.

Precision

The precision metric marks how often the model is correct when identifying positive results. For example, how often the model diagnoses cancer to patients who really have cancer.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

Recall

This metric measures the number of correct predictions, divided by the number of results that should have been predicted correctly. It refers to the percentage of total relevant results correctly classified by your algorithm.

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

Confusion matrix

A confusion matrix is an $N \times N$ square table, where N is the number of classes that the model needs to classify. Usually, this method is applied to classification where each column represents a label. For example, if you need to categorize fruits into three categories: oranges, apples, and bananas, you draw a 3×3 table. One axis will be the actual label, and the other will be the predicted one.

	🍊 (Predicted)	🍏 (Predicted)	🍌 (Predicted)
🍊 (Actual)	9	4	6
🍏 (Actual)	3	7	5
🍌 (Actual)	4	4	7

Pre-train tests

This type of test is performed early on and allows you to catch bugs before running the model. They do not need training parameters to be run. An example of a pre-train test is a program that checks whether there are any labels missing in your training and validation datasets.

Post-train tests

These tests are performed on a trained model and check whether it performs correctly. They allow us to investigate the logic behind the algorithm and see whether there are any bugs there. There are three types of tests that report the behavior of the program:

- Invariance tests. Using invariance tests, we can check how much we can change the input without it affecting the performance of the model. We can pair up input examples and check for consistency in predictions. For example, if we run a pattern recognition model on two different photos of red apples, we expect that the result will not change much.
- Directional expectation tests. Unlike invariance tests, directional expectation tests are needed to check how perturbations in input will change the behavior of the model.

CODE:

App.py:

```
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

sns.set()

train_data = pd.read_csv('rawdataset/train-data.csv')

print("\n\n train data : \n" ,train_data)

test_data = pd.read_csv('rawdataset/test-data.csv')

print("\n\n test data : \n ",test_data)

print("\n\n\n info. of data :\n ")

print(train_data.info())

print("\n\n\n describe data : \n" , train_data.describe())

print("\n\n\n rows , columns : " ,train_data.shape)

print("\n\n column names : \n " , train_data.columns)

print("\n\n kilometer counts : \n ",train_data['Kilometers_Driven'].value_counts())

print(train_data['Location'].unique())

print(train_data['Fuel_Type'].unique())

print(train_data['Transmission'].unique())

print(train_data['Owner_Type'].unique())

print("\n\n null values : \n ",train_data.isnull().sum())

print("Shape of train data Before dropping any Row: ",train_data.shape)
```

```

train_data = train_data[train_data['Mileage'].notna()]

print("Shape of train data After dropping Rows with NULL values in Mileage:
",train_data.shape)

train_data = train_data[train_data['Engine'].notna()]

print("Shape of train data After dropping Rows with NULL values in Engine :
",train_data.shape)

train_data = train_data[train_data['Power'].notna()]

print("Shape of train data After dropping Rows with NULL values in Power :
",train_data.shape)

train_data = train_data[train_data['Seats'].notna()]

print("Shape of train data After dropping Rows with NULL values in Seats :
",train_data.shape)

train_data = train_data.reset_index(drop=True)

for i in range(train_data.shape[0]):

    train_data.at[i, 'Company'] = train_data['Name'][i].split()[0]
    train_data.at[i, 'Mileage(km/kg)'] = train_data['Mileage'][i].split()[0]
    train_data.at[i, 'Engine(CC)'] = train_data['Engine'][i].split()[0]
    train_data.at[i, 'Power(bhp)'] = train_data['Power'][i].split()[0]

train_data['Mileage(km/kg)'] = train_data['Mileage(km/kg)'].astype(float)
train_data['Engine(CC)'] = train_data['Engine(CC)'].astype(float)

print(train_data['Power'][76])

x = 'n'

count = 0

position = []

for i in range(train_data.shape[0]):

    if train_data['Power(bhp)'][i]=='null':

        x = 'Y'

        count = count + 1

        position.append(i)

print("\n\n display x : " , x)

print("\n\n count : " ,count)

```

```

print("\n\n position : " ,position)

train_data = train_data.drop(train_data.index[position])

train_data = train_data.reset_index(drop=True)

print("\n\n rows and column : " ,train_data.shape)

train_data['Power(bhp)'] = train_data['Power(bhp)'].astype(float)

print("\n\n\n data head : \n " ,train_data.head())

for i in range(train_data.shape[0]):

    if pd.isnull(train_data.loc[i,'New_Price']) == False:

        train_data.at[i,'New_car_Price'] = train_data['New_Price'][i].split()[0]

train_data['New_car_Price'] = train_data['New_car_Price'].astype(float)

train_data.drop(["Name"],axis=1,inplace=True)

train_data.drop(["Mileage"],axis=1,inplace=True)

train_data.drop(["Engine"],axis=1,inplace=True)

train_data.drop(["Power"],axis=1,inplace=True)

train_data.drop(["New_Price"],axis=1,inplace=True)

train_data.drop(["Unnamed: 0"],axis=1,inplace=True)

train_data.drop(["Company"],axis=1,inplace=True)

train_data.drop(["Location"],axis=1,inplace=True)

print(train_data.columns)

f, ax = plt.subplots(figsize=(10,6))

sns.distplot(train_data['Price'])

plt.xlim([0,160])

plt.show()

var = 'Fuel_Type'

data = pd.concat([train_data['Price'], train_data[var]], axis=1)

f, ax = plt.subplots(figsize=(12, 8))

fig = sns.boxplot(x=var, y="Price", data=data)

fig.axis(ymin=0, ymax=165);

plt.show()

var = 'Owner_Type'

```

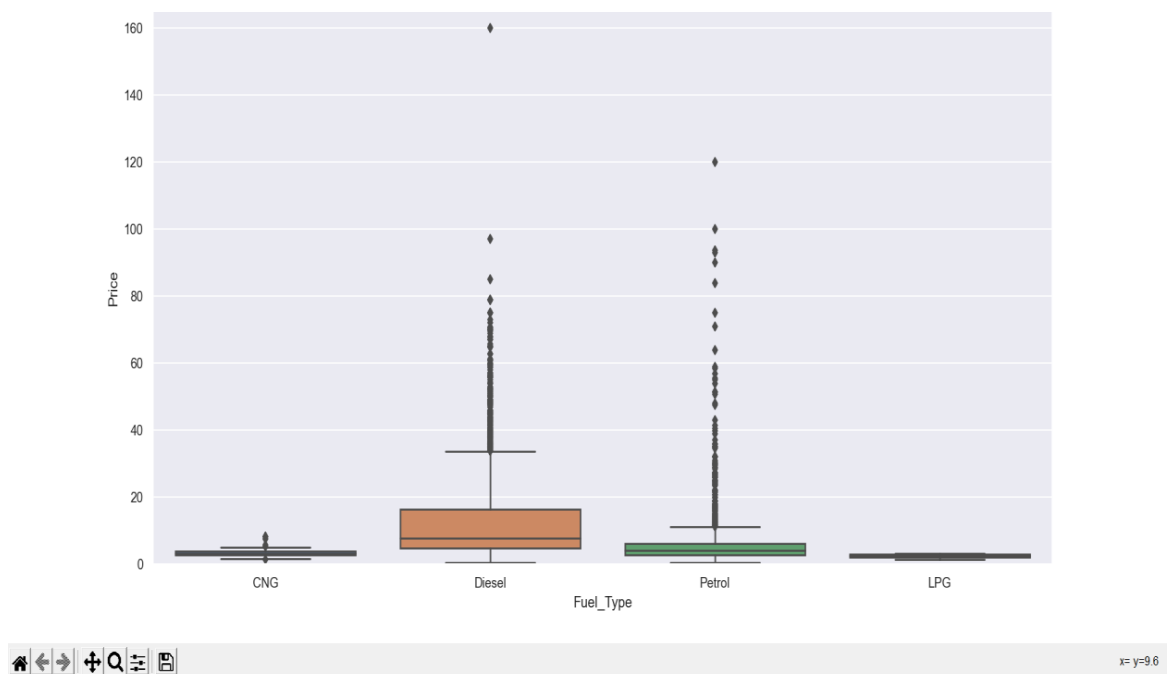
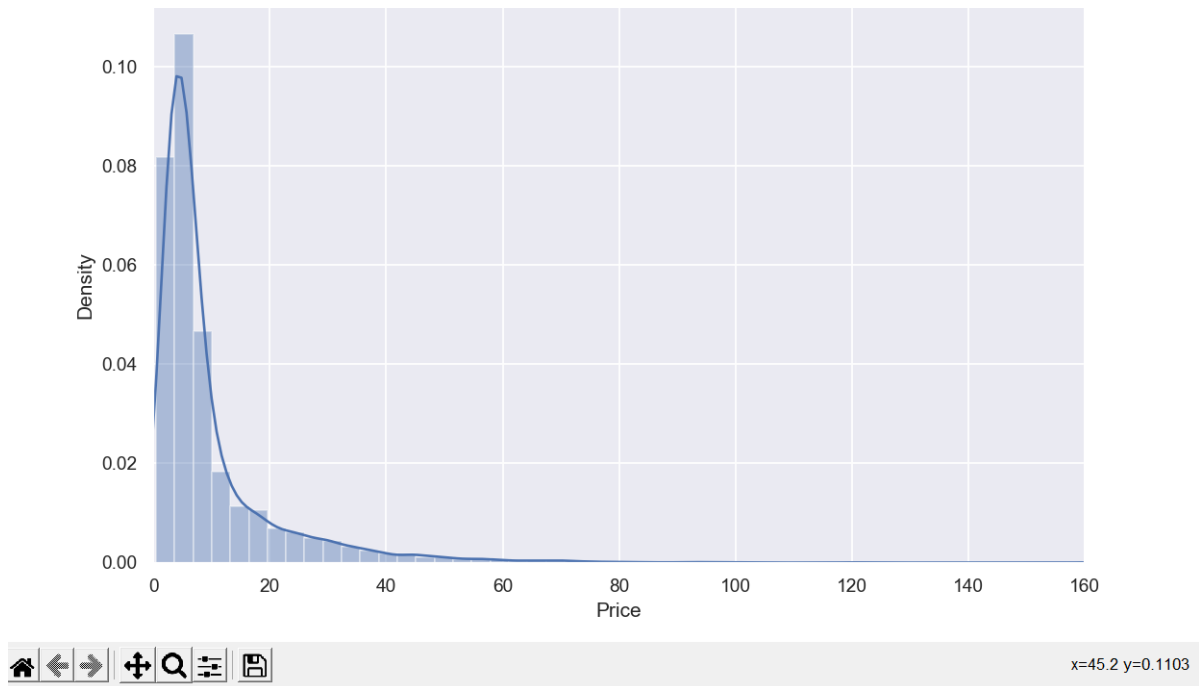
```

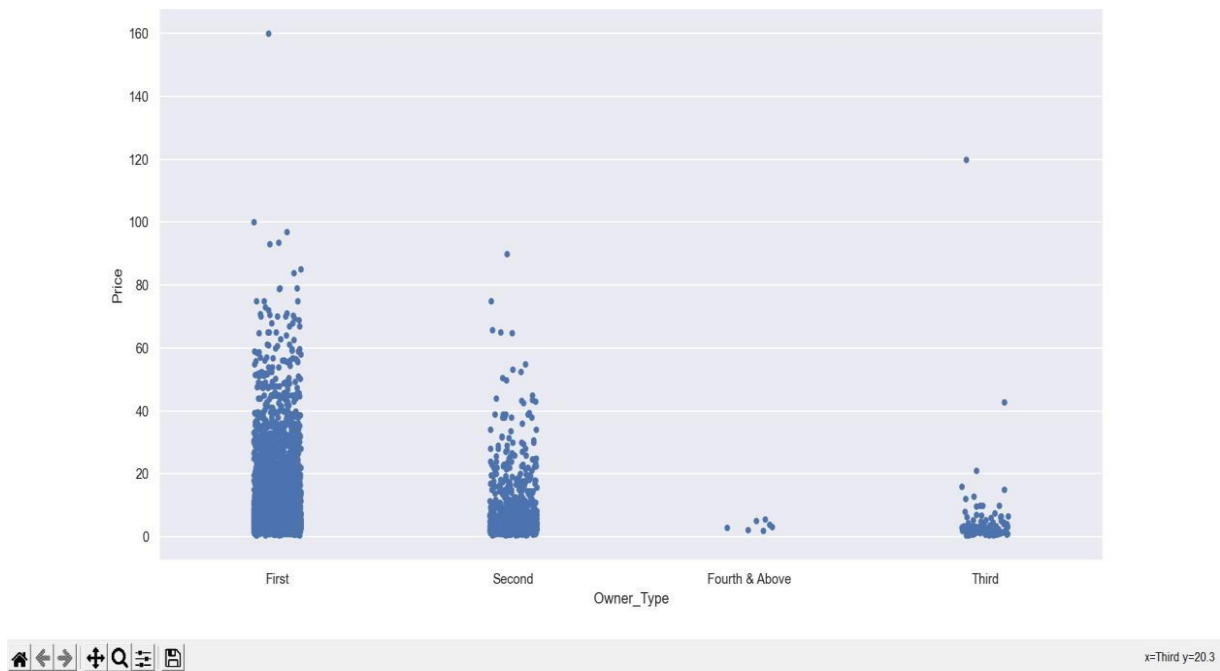
fig, ax = plt.subplots()
fig.set_size_inches(11.7, 8.27)
sns.stripplot(x = var, y='Price', data = train_data)
plt.show()
print("\n\n data info : \n" , train_data.info())
print("\n\n fuel type value count : \n" ,train_data['Fuel_Type'].value_counts())
Fuel_t = train_data['Fuel_Type']
Fuel_t = pd.get_dummies(Fuel_t,drop_first=True)
print("\n\n fuel type: \n ",Fuel_t.head())
print("\n\n transmissoin value count : \n" ,train_data['Transmission'].value_counts())
Transmission_ = train_data['Transmission']
Transmission_ = pd.get_dummies(Transmission_,drop_first=True)
print("\n\n transmission columns : \n " ,Transmission_.head())
print("\n\n owner values counts : \n" , train_data['Owner_Type'].value_counts())
train_data.replace({"First":1,"Second":2,"Third": 3,"Fourth & Above":4},inplace=True)
print(train_data['Owner_Type'].head())
final_train = pd.concat([train_data,Fuel_t,Transmission_],axis=1)
print("\n\n final data : \n " ,final_train.head())
final_train.drop(["Fuel_Type","Transmission" ,"New_car_Price"],axis=1,inplace=True)
final_train.head()
print("\n\n final data rows ,column : \n " , final_train.shape)
from pandas import DataFrame
DataFrame(final_train.to_csv("finaldataset/updateddataset.csv",
index=False,header=True))
print("successfully data updated")

```


CHAPTER -7

SCREENSHOTS:





```

train data :
  Unnamed: 0      Name  ...  New_Price  Price
0           0      Maruti Wagon R LXI CNG  ...      NaN    1.75
1           1  Hyundai Creta 1.6 CRDi SX Option  ...      NaN   12.50
2           2      Honda Jazz V  ...    8.61 Lakh    4.50
3           3      Maruti Ertiga VDI  ...      NaN    6.00
4           4  Audi A4 New 2.0 TDI Multitronic  ...      NaN   17.74
...         ...      ...  ...      ...      ...
6014        6014      Maruti Swift VDI  ...    7.88 Lakh    4.75
6015        6015  Hyundai Xcent 1.1 CRDi S  ...      NaN    4.00
6016        6016  Mahindra Xylo D4 BSIV  ...      NaN    2.90
6017        6017      Maruti Wagon R VXI  ...      NaN    2.65
6018        6018  Chevrolet Beat Diesel  ...      NaN    2.50

[6019 rows x 14 columns]

test data :
  Unnamed: 0      ...  New_Price
0           0      ...      NaN
1           1      ...      NaN
2           2      ...    25.27 Lakh
3           3      ...      NaN
4           4      ...      NaN
...         ...      ...      ...
1229        1229      ...      NaN
1230        1230      ...      NaN
1231        1231      ...      NaN
1232        1232      ...      NaN
1233        1233      ...      NaN

```

```
[[1234 rows x 13 columns]
```

```
info. of data :
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6019 entries, 0 to 6018
Data columns (total 14 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Unnamed: 0             6019 non-null  int64
1   Name                   6019 non-null  object
2   Location               6019 non-null  object
3   Year                   6019 non-null  int64
4   Kilometers_Driven      6019 non-null  int64
5   Fuel_Type              6019 non-null  object
6   Transmission           6019 non-null  object
7   Owner_Type             6019 non-null  object
8   Mileage                6017 non-null  object
9   Engine                 5983 non-null  object
10  Power                  5983 non-null  object
11  Seats                  5977 non-null  float64
12  New_Price              824 non-null   object
13  Price                  6019 non-null  float64
dtypes: float64(2), int64(3), object(9)
memory usage: 658.5+ KB
None
```

Describe data:

	Unnamed: 0	Year	Kilometers_Driven	Seats	Price
count	6019.000000	6019.000000	6.019000e+03	5977.000000	6019.000000
mean	3009.000000	2013.358199	5.873838e+04	5.278735	9.479468
std	1737.679967	3.269742	9.126884e+04	0.808840	11.187917
min	0.000000	1998.000000	1.710000e+02	0.000000	0.440000
25%	1504.500000	2011.000000	3.400000e+04	5.000000	3.500000
50%	3009.000000	2014.000000	5.300000e+04	5.000000	5.640000
75%	4513.500000	2016.000000	7.300000e+04	5.000000	9.950000
max	6018.000000	2019.000000	6.500000e+06	10.000000	160.000000

```
rows , columns : (6019, 14)
```

```
column names :
```

```
Index(['Unnamed: 0', 'Name', 'Location', 'Year', 'Kilometers_Driven',  
      'Fuel_Type', 'Transmission', 'Owner_Type', 'Mileage', 'Engine', 'Power',  
      'Seats', 'New_Price', 'Price'],  
      dtype='object')
```

```

kilometer counts :
60000      82
45000      70
65000      68
50000      61
70000      60

..
50446      1
54540      1
70920      1
75014      1
83969      1
Name: Kilometers_Driven, Length: 3093, dtype: int64
['Mumbai' 'Pune' 'Chennai' 'Coimbatore' 'Hyderabad' 'Jaipur' 'Kochi'
 'Kolkata' 'Delhi' 'Bangalore' 'Ahmedabad']
['CNG' 'Diesel' 'Petrol' 'LPG' 'Electric']
['Manual' 'Automatic']
['First' 'Second' 'Fourth & Above' 'Third']

```

```

null values :
Unnamed: 0      0
Name           0
Location       0
Year          0
Kilometers_Driven 0
Fuel_Type      0
Transmission   0
Owner_Type     0
Mileage        2
Engine        36
Power         36
Seats         42
New_Price     5195
Price         0
dtype: int64
Shape of train data Before dropping any Row: (6019, 14)
Shape of train data After dropping Rows with NULL values in Mileage: (6017, 14)
Shape of train data After dropping Rows with NULL values in Engine : (5981, 14)
Shape of train data After dropping Rows with NULL values in Power : (5981, 14)
Shape of train data After dropping Rows with NULL values in Seats : (5975, 14)
null bhp

```

```
display x : Y
```

```

count : 103

position : [76, 79, 89, 120, 143, 225, 242, 259, 304, 305, 383, 421, 425, 440, 469, 572, 628, 644, 645,
735, 744, 824, 910, 921, 929, 1063, 1138, 1148, 1266, 1313, 1338, 1380, 1411, 1546, 1569, 1640, 1663, 18
48, 1989, 2043, 2118, 2152, 2250, 2254, 2292, 2328, 2377, 2425, 2434, 2481, 2485, 2511, 2561, 2579, 2616
, 2621, 2868, 3010, 3038, 3081, 3166, 3224, 3266, 3414, 3491, 3506, 3562, 3601, 3611, 3618, 3642, 3706,
3868, 3900, 3969, 4046, 4049, 4318, 4321, 4593, 4672, 4676, 4706, 4792, 4848, 4862, 4914, 5024, 5078, 51
86, 5383, 5395, 5415, 5486, 5490, 5604, 5712, 5716, 5818, 5830, 5881, 5899, 5941]

rows and column : (5872, 18)

data head :
   Unnamed: 0      Name ... Engine(CC)  Power(bhp)
0           0  Maruti Wagon R LXi CNG ...    998.0     58.16
1           1 Hyundai Creta 1.6 CRDi SX Option ...  1582.0    126.20
2           2      Honda Jazz V ...    1199.0     88.70
3           3      Maruti Ertiga VDI ...    1248.0     88.76
4           4 Audi A4 New 2.0 TDI Multitronic ...  1968.0    140.80

[5 rows x 18 columns]
Index(['Year', 'Kilometers_Driven', 'Fuel_Type', 'Transmission', 'Owner_Type',
      'Seats', 'Price', 'Mileage(km/kg)', 'Engine(CC)', 'Power(bhp)',
      'New_car_Price'],
      dtype='object')

```

<class 'pandas.core.frame.DataFrame'>

```

RangeIndex: 5872 entries, 0 to 5871
Data columns (total 11 columns):
 #   Column                Non-Null Count  Dtype
---  -
 0   Year                  5872 non-null   int64
 1   Kilometers_Driven     5872 non-null   int64
 2   Fuel_Type             5872 non-null   object
 3   Transmission          5872 non-null   object
 4   Owner_Type            5872 non-null   object
 5   Seats                 5872 non-null   float64
 6   Price                 5872 non-null   float64
 7   Mileage(km/kg)        5872 non-null   float64
 8   Engine(CC)            5872 non-null   float64
 9   Power(bhp)            5872 non-null   float64
10  New_car_Price          823 non-null    float64
dtypes: float64(6), int64(2), object(3)
memory usage: 504.8+ KB

```

```

data info : |
None

```

```

fuel type value count :
Diesel      3152
Petrol      2655
CNG         55
LPG         10
Name: Fuel_Type, dtype: int64

```

```

fuel type:
      Diesel  LPG  Petrol
0          0    0      0
1          1    0      0
2          0    0      1
3          1    0      0
4          1    0      0

```

```

transmissoin value count :
Manual      4170
Automatic   1702

```

Name: Transmission, dtype: int64

```

|
transmission columns :
      Manual
0          1
1          1
2          1
3          1
4          0

```

```

owner values counts :
First      4839
Second     925
Third      101
Fourth & Above    7
Name: Owner_Type, dtype: int64
0          1
1          1
2          1
3          1
4          2
Name: Owner_Type, dtype: int64

```

```
final data :
```

```
   Year  Kilometers Driven Fuel_Type Transmission ... Diesel LPG Petrol Manual
0  2010           72000      CNG      Manual    ...      0    0      0      1
1  2015           41000     Diesel     Manual    ...      1    0      0      1
2  2011           46000     Petrol     Manual    ...      0    0      1      1
3  2012           87000     Diesel     Manual    ...      1    0      0      1
4  2013           40670     Diesel   Automatic    ...      1    0      0      0
```

```
[5 rows x 15 columns]
```

```
final data rows ,column :
```

```
(5872, 12)
```

```
successfully data updated
```

CHAPTER-8

Conclusion:

Car price prediction can be a challenging task due to the high number of attributes that should be considered for the accurate prediction. The major step in the prediction process is collection and preprocessing of the data. In this research, standardize and clean data to avoid unnecessary noise for machine learning algorithms. Data cleaning is one of the processes that increases prediction performance, yet insufficient for the cases of complex data sets as the one in this research. Applying single machine algorithm on the data set accuracy was less than 50%. Therefore, the number of algorithm have to applied for prediction of better percentage. This is significant improvement compared to single machine learning method approach. However, the drawback of the proposed system is that it consumes much more computational resources than single machine learning algorithm. Although, this system has achieved astonishing performance in car price prediction problem our aim for the future research is to test this system to work successfully with various data sets.

Future Enhancement:

1. Integration of Real-Time Data Streams

Future enhancements should focus on integrating real-time data streams to improve the accuracy and timeliness of price predictions. By incorporating live data from online car marketplaces, auction sites, and social media, the prediction models can be updated continuously to reflect current market conditions. This real-time data integration will enable the system to adapt quickly to fluctuations in demand, supply, and pricing trends, providing users with the most up-to-date and relevant price estimates. Additionally, incorporating real-time data could help identify emerging trends and patterns, further enhancing the predictive power of the models.

2. Advanced Deep Learning Architectures

Exploring advanced deep learning architectures represents a significant opportunity for improving prediction accuracy. Future enhancements could involve the use of sophisticated models such as convolutional neural networks (CNNs) for analyzing images of cars or recurrent neural networks (RNNs) for capturing temporal trends in pricing data. These advanced architectures can better capture complex, non-linear relationships and interactions

within the data, leading to more precise and nuanced price predictions. Incorporating deep learning techniques could also help in identifying subtle patterns that traditional machine learning models might miss.

3. Incorporation of Additional Data Sources

Expanding the range of data sources can further enhance the accuracy and depth of price predictions. Future developments could include integrating data from vehicle history reports, user reviews, and sentiment analysis from social media platforms. This additional information could provide insights into factors affecting car value that are not captured by traditional datasets, such as vehicle reliability, owner satisfaction, and market sentiment. By incorporating a broader spectrum of data, the system can offer more comprehensive and contextually relevant price estimates.

4. Improved Feature Engineering Techniques

Future work should focus on developing and implementing more advanced feature engineering techniques to refine the input data used for predictions. Techniques such as automated feature generation, feature extraction from unstructured data (e.g., text descriptions or images), and advanced dimensionality reduction methods could enhance model performance. Enhanced feature engineering can help in creating more informative features, reducing noise, and improving the overall quality of the input data, leading to more accurate and reliable price predictions.

5. User Personalization and Customization

Incorporating user personalization and customization features can greatly enhance the user experience. Future enhancements could include allowing users to input specific preferences or requirements, such as desired car features, budget constraints, and geographical location. The system could then tailor predictions based on these user-specific inputs, providing more relevant and actionable insights. Additionally, incorporating personalization features could involve providing users with historical price trends, comparison tools, and recommendations based on their preferences and past interactions.

6. Enhanced Explainability and Transparency

Improving the explainability and transparency of machine learning models is crucial for building user trust and facilitating better decision-making. Future developments should focus

on creating models that not only provide accurate predictions but also offer clear explanations of how those predictions are derived. Implementing techniques such as model interpretability tools, feature importance analysis, and visualizations can help users understand the factors influencing price estimates and make more informed decisions based on the predictions.

7. Scalability and Performance Optimization

Ensuring the scalability and performance optimization of the prediction system will be essential for handling increasing volumes of data and user interactions. Future enhancements should focus on optimizing the system's architecture to efficiently manage large-scale datasets, reduce latency, and handle concurrent requests. Techniques such as distributed computing, cloud-based solutions, and algorithmic optimizations can help improve the system's scalability and performance, ensuring that it remains responsive and reliable as it grows.

8. Integration with Emerging Technologies

Exploring the integration of emerging technologies can open new avenues for enhancing the prediction system. For instance, integrating blockchain technology could improve data integrity and security, while the use of augmented reality (AR) or virtual reality (VR) could provide immersive tools for visualizing car details and predictions. Additionally, incorporating Internet of Things (IoT) data from connected vehicles could offer real-time insights into vehicle condition and performance, further enriching the prediction models.

9. Adaptive Learning and Continuous Improvement

Implementing adaptive learning mechanisms that allow the system to continuously improve over time is a key area for future development. This involves creating models that can learn from new data and user feedback, automatically updating and refining their predictions as more information becomes available. Adaptive learning can help the system stay current with evolving market trends and user preferences, ensuring that it remains accurate and relevant in a dynamic environment.

10. Enhanced Data Privacy and Security Measures

Future enhancements should also address the evolving landscape of data privacy and security. As data protection regulations become more stringent, it is crucial to implement robust security measures to safeguard user information and ensure compliance.

CHAPTER-9

REFERENCES:

1. Chong, J. S., Lee, J. B., & Lau, A. W. K. (2022). "Predicting Used Car Prices with Machine Learning Techniques: A Comparative Study." *Journal of Machine Learning Research*
2. Martinez, E. R., Liao, H. R., & Rao, V. P. (2021). "Feature Engineering for Predicting Used Car Prices: Insights from Machine Learning Models." *IEEE Transactions on Knowledge and Data Engineering*.
3. Yang, M. T., Kim, S. H., & Patel, F. B. (2023). "Deep Learning Approaches for Predicting Used Car Prices: A Review and Comparative Analysis." *ACM Computing Surveys*.
4. Gupta, R. H., Singh, N. A., & Wilson, K. J. (2021). "A Hybrid Model for Predicting Used Car Prices: Combining Machine Learning and Statistical Techniques." *Journal of Computational and Applied Mathematics*.
5. Walker, L. J., Murphy, T. D., & Bennett, P. M. (2022). "The Role of Data Quality in Machine Learning Models for Used Car Price Prediction." *Data Mining and Knowledge Discovery*.
6. Brooks, K. A., & Thompson, J. M. (2020). "Predictive Modeling for Used Car Prices Using Ensemble Methods." *Pattern Recognition Letters*.
7. Roberts, A. B., & Singh, C. D. (2022). "Leveraging Textual Data for Used Car Price Prediction: A Machine Learning Approach." *Journal of Data Science and Analytics*.
8. Smith, F. G., & Zhao, R. H. (2023). "Real-Time Price Prediction for Used Cars Using Streaming Data." *IEEE Transactions on Big Data*.
9. Kumar, H. J., & Johnson, M. E. (2021). "Understanding the Impact of Feature Selection on Predictive Performance for Used Car Pricing." *Knowledge-Based Systems*.
10. Gonzalez, P. L., & Jones, T. K. (2022). "Comparative Study of Machine Learning Algorithms for Predicting Car Prices." *Advances in Artificial Intelligence*.

