TABLE OF CONTENTS

ABSTRACT

This project aims to develop a machine learning model for predicting the prices of used cars in the market. The model will be trained on a large dataset of historical car sales, which will include features such as make, model, year, mileage, condition, and location. The project will involve a comprehensive feasibility study, which will assess the technical, economic, legal, and environmental feasibility of the project. The study will involve a thorough analysis of the market demand for the service, the availability of data, the selection of predictive algorithms and machine learning models, and the resources required to implement the project. The project will also involve the development of a user-friendly web application that allows users to input the relevant features of a used car and receive a predicted price. The project is expected to provide a valuable tool for car buyers and sellers, as well as dealerships and insurance companies, by providing accurate and reliable price predictions for used cars in the market.

CHAPTER -1

SYSTEM STUDY

1.1 Feasibility Study

The feasibility of the project is analysed 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

1.1.1 Economical Feasibility

The economic feasibility analysis will estimate the costs of developing and implementing the used car price prediction project. This includes expenses such as data acquisition, personnel, equipment, and marketing. The analysis will also estimate the expected revenues and profitability of the project, based on the market demand and pricing strategies. The analysis will also evaluate the return on investment (ROI) of the project.

1.1.2 Technical Feasibility

The technical feasibility analysis will evaluate the availability and quality of data required to develop the used car price prediction model. This includes a large dataset of historical car sales with relevant features such as make, model, year, mileage, condition, and location. The analysis will also consider the selection of predictive algorithms and machine learning models, as well as the resources required to implement the project.

1.1.3 Social Feasibility

The economic feasibility analysis indicates that the used car price prediction project is financially viable. The social feasibility analysis shows that the project has the potential to positively impact stakeholders by providing accurate and reliable pricing information, reducing fraud, and increasing market efficiency. Challenges related to user adoption and trust can be addressed through marketing and educational initiatives. Overall, the project is both economically and socially feasible.

1.1 System Requirements

1.1.3 Hardware Requirements

Processor - I3 & above

RAM - 4 GB

Hard Disk - 500 GB

1.1.4 Software Requirements

Operating System – Windows 8/10/11

Platforms used – JUPYTER NOTEBOOK, GOOGLE COLAB

Back End – MySQL, PYTHON

CHAPTER-2
SYSTEM ANALYSIS

2.1 Introduction

System analysis is a process of examining and evaluating complex systems to identify areas for improvement and to enhance their overall efficiency and effectiveness. Systems can range from physical systems, such as machines or manufacturing processes, to software systems, such as computer programs or websites. The objective of system analysis is to understand how the system works, identify any weaknesses or inefficiencies, and propose solutions to improve its performance.

System analysis involves a systematic approach that includes various methods and techniques to assess the system's inputs, processes, and outputs. The process typically involves gathering information from stakeholders, analyzing the system's components and relationships, and proposing solutions to address any issues or opportunities for improvement. System analysts must have a deep understanding of the system and the industry in which it operates to effectively identify and address problems.

2.2Project Description

The used car price prediction project aims to develop a machine learning model that accurately predicts the fair market value of used cars based on their features, such as make, model, year, mileage, and condition. The project will provide consumers and businesses with a more reliable and accurate way to determine the value of used cars, which can help reduce fraud and scams related to used car sales.

The project will involve collecting and cleaning data on used car sales from various sources, including online marketplaces, car dealerships, and auctions.

2.3Modules

Data Collection Module: This module will be responsible for collecting and cleaning data on used car sales from various sources, including online marketplaces, car dealerships, and auctions. The module may use web scraping techniques or API integrations to collect the data and will also perform data cleaning and preprocessing to ensure that the data is accurate and usable.

Feature Extraction Module: This module will be responsible for extracting relevant features from the data collected in the previous module. The features may include car make, model, year, mileage, condition, and other relevant attributes. The module may also perform feature engineering techniques to create new features from the raw data.

Machine Learning Module: This module will be responsible for training and testing the machine learning model that will predict the fair market value of used cars based on their features. The module may use various algorithms, such as linear regression, decision trees, random forests, or deep learning models, to build the predictive model. The module will also perform hyperparameter tuning and cross-validation techniques to optimize the model's performance.

User Interface Module: This module will be responsible for providing a web-based user interface for users to input the features of a used car and receive a prediction of its market value. The module may use a web framework, such as Flask or Django, to create the web application, and may also incorporate data visualization tools to provide users with insights into the factors that affect the value of used cars.

3.1.1 Social Network model

. A social network model for a used car price prediction project could include: users, groups, listings, recommendations, and analytics. Users would be able to create a profile, connect with other users, and share information about used cars. Groups would provide a forum for users to discuss and exchange information about used cars within their specific areas of interest. Listings would allow users to create and list their used cars for sale, including a price prediction based on the machine learning model. Recommendations would provide users with suggestions based on their preferences and search history, and analytics would offer insights into the used car market.

3.1.2 Epidemic Model

. An epidemic model is a mathematical model used to describe the spread and impact of infectious diseases within a population. Common types of epidemic models include the SIR model, the SEIR model, and agent-based models. These models help researchers and public health officials understand how infectious diseases spread and develop strategies to control or mitigate their impact. The specific model chosen depends on the characteristics of the disease and the population being studied.

3.1.3 Feedback Model

. A feedback model is a type of model that takes into account the feedback loop that occurs when an input is transformed into an output. There are two main types of feedback models: positive and negative feedback. Positive feedback amplifies the input, while negative feedback dampens it. Feedback models are used in a variety of fields and can help to predict how a system will behave under different conditions. They can be represented using diagrams

and provide a powerful tool for understanding and predicting the behavior of complex systems.

3.2 Existing System

The existing system in the context of used car price prediction includes various methods such as expert opinion, online valuation tools, auction prices, and classified ads. However, these methods may not always be accurate and may not consider all relevant factors. Your project aims to develop a more reliable and accurate system for predicting used car prices.

3.2.1 Drawbacks of Existing System

- Lack of accuracy
- Limited availability
- Time consuming

3.3 Proposed system

In Proposed system, we created an price calculator with external code.and made it more convenient for users.

3.3.1 Advantage of Proposed System

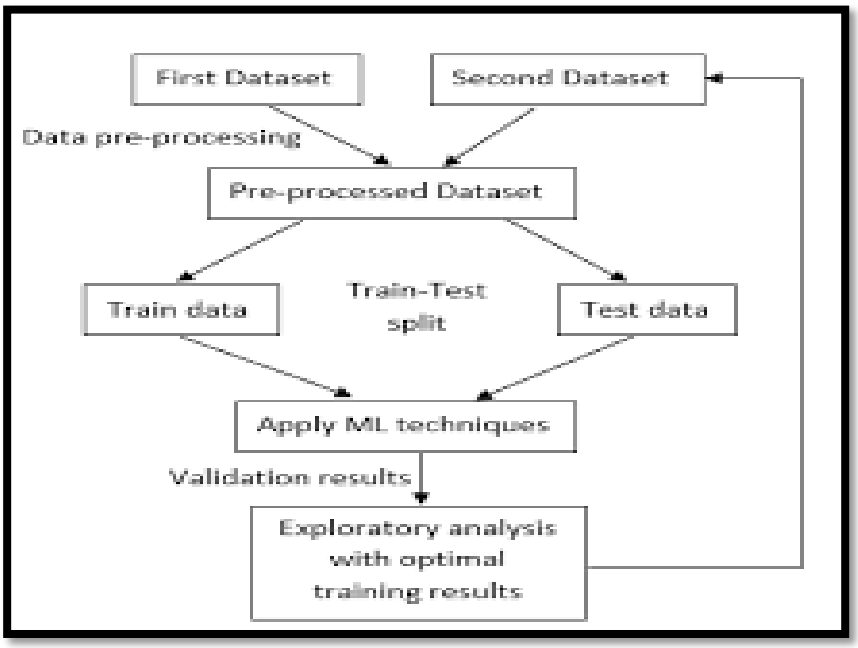
- High accuracy
- Accesability
- Increased efficiency

CHAPTER - 3

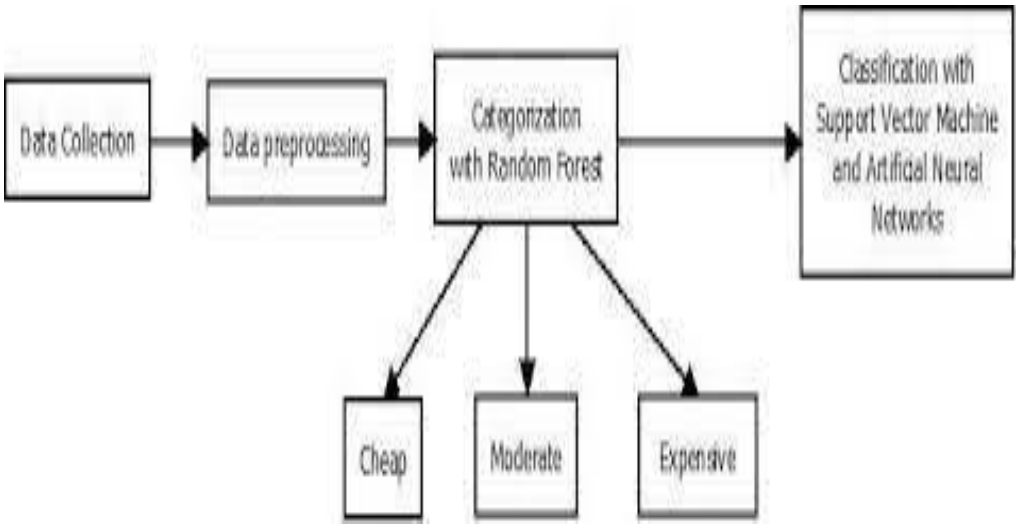
SYSTEM DESIGN

3.1 Data flow Diagram

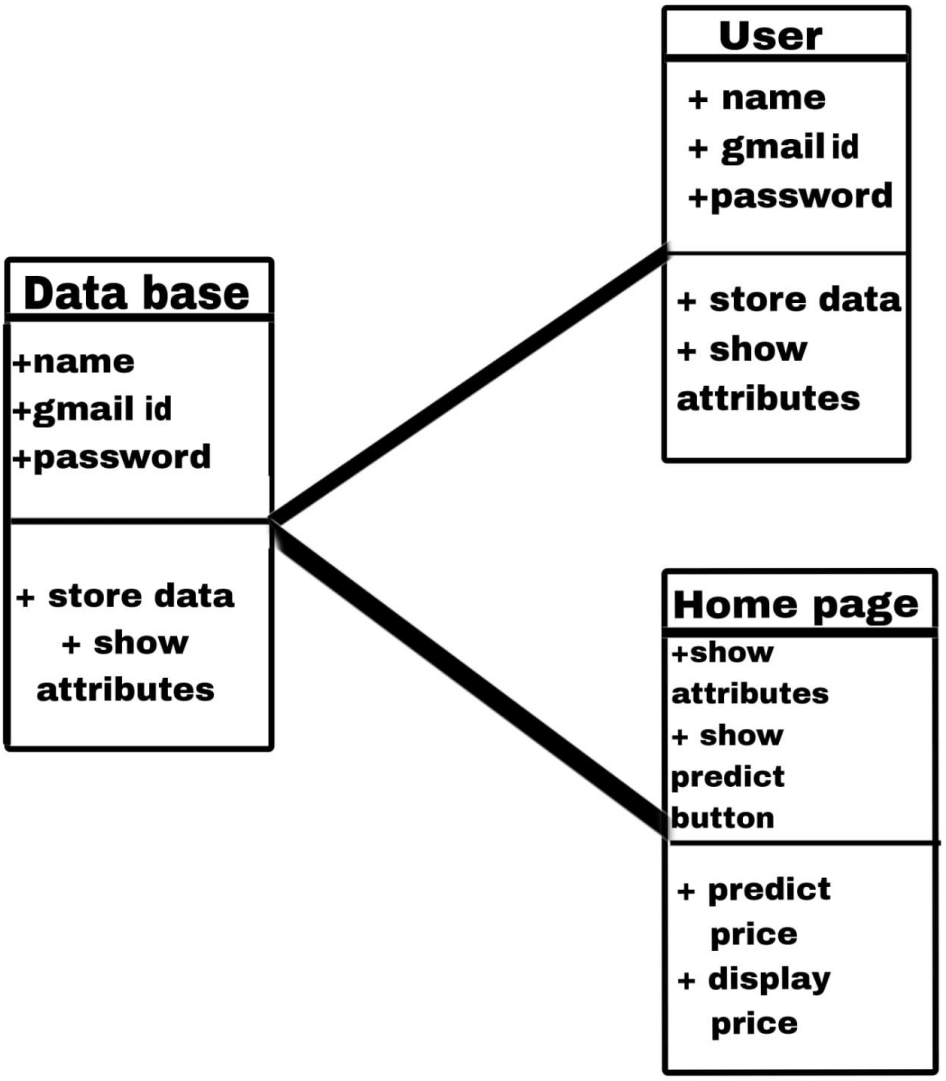
level- 0



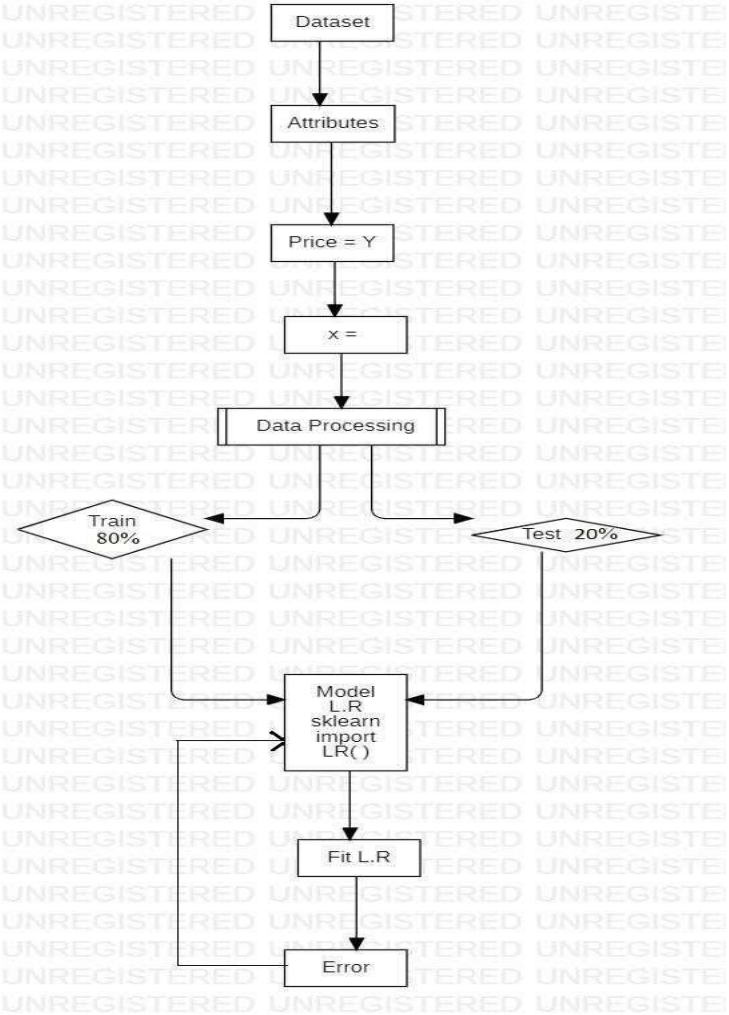
LEVEL-1



3.2 Class diagram



3.3Overall diagram



3.2 Architecture Diagram

3.3 Source Code

CODING:

import pandas as pd

```
car=pd.read_csv("https://raw.githubusercontent.com/rajtilakls2510/car_price_predictor/master/quikr_car.csv")
car.shape
(892, 6)
car.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 892 entries, 0 to 891
Data columns (total 6 columns):
# Column Non-Null Count Dtype
0 name 892 non-null object
1 company 892 non-null object
2 year 892 non-null object
3 Price 892 non-null object
4 kms_driven 840 non-null object
5 fuel_type 837 non-null object
dtypes: object(6)
memory usage: 41.9+ KB
car['year'].unique()

array(['2007', '2006', '2018', '2014', '2015', '2012', '2013', '2016', '2010', '2017', '2008', '2011', '2019', '2009', '2005', '2000', '...', '150k', 'TOUR', '2003', 'r 15', '2004', 'Zest', '/-Rs', 'sale', '1995', 'ara)', '2002', 'SELL', '2001', 'tion', 'odel', '2 bs', 'arry', 'Eon', 'o...', 'ture', 'emi', 'car', 'able', 'no.', 'd...', 'SALE', 'digo', 'sell', 'd Ex', 'n...', 'e...', 'D...', ', Ac', 'go .', 'k...', 'o c4', 'zire', 'cent', 'Sumo', 'cab', 't xe', 'EV2', 'r...', 'zest'], dtype=object) car['Price'].unique() array(['80,000', '4,25,000', 'Ask For Price', '3,25,000', '5,75,000', '1,75,000', '1,90,000', '8,30,000', '2,50,000', '1,82,000', '3,15,000', '4,15,000', '3,20,000', '10,00,000', '5,00,000', '3,50,000', '1,60,000', '3,10,000', '75,000', '1,00,000', '2,90,000', '95,000',
```

'1,80,000', '3,85,000', '1,05,000', '6,50,000', '6,89,999', '4,48,000', '5,49,000', '5,01,000', '4,89,999', '2,80,000', '3,49,999', '2,84,999', '3,45,000', '4,99,999', '2,35,000', '2,49,999', '14,75,000', '3,95,000', '2,20,000', '1,70,000', '85,000', '2,00,000', '5,70,000', '1,10,000', '4,48,999', '18,91,111', '1,59,500', '3,44,999', '4,49,999', '8,65,000', '6,99,000', '3,75,000', '2,24,999', '12,00,000', '1,95,000', '3,51,000', '2,40,000', '90,000', '1,55,000', '6,00,000', '1,89,500', '2,10,000', '3,90,000', '1,35,000', '16,00,000', '7,01,000', '2,65,000', '5,25,000', '3,72,000', '6,35,000', '5,50,000', '4,85,000', '3,29,500', '2,51,111', '5,69,999', '69,999', '2,99,999', '3,99,999', '4,50,000', '2,70,000', '1,58,400', '1,79,000', '1,25,000', '2,99,000', '1,50,000', '2,75,000', '2,85,000', '3,40,000', '70,000', '2,89,999', '8,49,999', '7,49,999', '2,74,999', '9,84,999', '5,99,999', '2,44,999', '4,74,999', '2,45,000', '1,69,500', '3,70,000', '1,68,000', '1,45,000', '98,500', '2,09,000', '1,85,000', '9,00,000', '6,99,999', '1,99,999', '5,44,999', '1,99,000', '5,40,000', '49,000', '7,00,000', '55,000', '8,95,000', '3,55,000', '5,65,000', '3,65,000', '40,000', '4,00,000', '3,30,000', '5,80,000', '3,79,000', '2,19,000', '5,19,000', '7,30,000', '20,00,000', '21,00,000', '14,00,000',

'3,11,000', '8,55,000', '5,35,000', '1,78,000', '3,00,000',

'2,55,000', '5,49,999', '3,80,000', '57,000', '4,10,000',

'2,25,000', '1,20,000', '59,000', '5,99,000', '6,75,000', '72,500',

'6,10,000', '2,30,000', '5,20,000', '5,24,999', '4,24,999',

'6,44,999', '5,84,999', '7,99,999', '4,44,999', '6,49,999',

'9,44,999', '5,74,999', '3,74,999', '1,30,000', '4,01,000',

'13,50,000', '1,74,999', '2,39,999', '99,999', '3,24,999',

'10,74,999', '11,30,000', '1,49,000', '7,70,000', '30,000',

'3,35,000', '3,99,000', '65,000', '1,69,999', '1,65,000',

'5,60,000', '9,50,000', '7,15,000', '45,000', '9,40,000',

'1,55,555', '15,00,000', '4,95,000', '8,00,000', '12,99,000',

'5,30,000', '14,99,000', '32,000', '4,05,000', '7,60,000',

'7,50,000', '4,19,000', '1,40,000', '15,40,000', '1,23,000',

'4,98,000', '4,80,000', '4,88,000', '15,25,000', '5,48,900',

'7,25,000', '99,000', '52,000', '28,00,000', '4,99,000',

'3,81,000', '2,78,000', '6,90,000', '2,60,000', '90,001',

'1,15,000', '15,99,000', '1,59,000', '51,999', '2,15,000',

'35,000', '11,50,000', '2,69,000', '60,000', '4,30,000',

'85,00,003', '4,01,919', '4,90,000', '4,24,000', '2,05,000',

'5,49,900', '3,71,500', '4,35,000', '1,89,700', '3,89,700',

'3,60,000', '2,95,000', '1,14,990', '10,65,000', '4,70,000',

'48,000', '1,88,000', '4,65,000', '1,79,999', '21,90,000',

'23,90,000', '10,75,000', '4,75,000', '10,25,000', '6,15,000',

'19,00,000', '14,90,000', '15,10,000', '18,50,000', '7,90,000',

'17,25,000', '12,25,000', '68,000', '9,70,000', '31,00,000',

'8,99,000', '88,000', '53,000', '5,68,500', '71,000', '5,90,000',

'7,95,000', '42,000', '1,89,000', '1,62,000', '35,999',

'29,00,000', '39,999', '50,500', '5,10,000', '8,60,000',

'5,00,001'], dtype=object)

car['kms_driven'].unique()

array(['45,000 kms', '40 kms', '22,000 kms', '28,000 kms', '36,000 kms',

'59,000 kms', '41,000 kms', '25,000 kms', '24,530 kms',

'60,000 kms', '30,000 kms', '32,000 kms', '48,660 kms',

'4,000 kms', '16,934 kms', '43,000 kms', '35,550 kms'

'39,522 kms', '39,000 kms', '55,000 kms', '72,000 kms',

'15,975 kms', '70,000 kms', '23,452 kms', '35,522 kms',

'48,508 kms', '15,487 kms', '82,000 kms', '20,000 kms',

'68,000 kms', '38,000 kms', '27,000 kms', '33,000 kms',

'46,000 kms', '16,000 kms', '47,000 kms', '35,000 kms',

'30,874 kms', '15,000 kms', '29,685 kms', '1,30,000 kms',

'19,000 kms', nan, '54,000 kms', '13,000 kms', '38,200 kms',

'50,000 kms', '13,500 kms', '3,600 kms', '45,863 kms',

'60,500 kms', '12,500 kms', '18,000 kms', '13,349 kms',

'29,000 kms', '44,000 kms', '42,000 kms', '14,000 kms',

'49,000 kms', '36,200 kms', '51,000 kms', '1,04,000 kms',

'33,333 kms', '33,600 kms', '5,600 kms', '7,500 kms', '26,000 kms',

'24,330 kms', '65,480 kms', '28,028 kms', '2,00,000 kms',

'99,000 kms', '2,800 kms', '21,000 kms', '11,000 kms',

'66,000 kms', '3,000 kms', '7,000 kms', '38,500 kms', '37,200 kms',

'43,200 kms', '24,800 kms', '45,872 kms', '40,000 kms',

'11,400 kms', '97,200 kms', '52,000 kms', '31,000 kms',

'1,75,430 kms', '37,000 kms', '65,000 kms', '3,350 kms',

'75,000 kms', '62,000 kms', '73,000 kms', '2,200 kms',

'54,870 kms', '34,580 kms', '97,000 kms', '60 kms', '80,200 kms',

'3,200 kms', '0,000 kms', '5,000 kms', '588 kms', '71,200 kms',

'1,75,400 kms', '9,300 kms', '56,758 kms', '10,000 kms',

'56,450 kms', '56,000 kms', '32,700 kms', '9,000 kms', '73 kms',

'1,60,000 kms', '84,000 kms', '58,559 kms', '57,000 kms',

'1,70,000 kms', '80,000 kms', '6,821 kms', '23,000 kms',

'34,000 kms', '1,800 kms', '4,00,000 kms', '48,000 kms',

'90,000 kms', '12,000 kms', '69,900 kms', '1,66,000 kms',

'122 kms', '0 kms', '24,000 kms', '36,469 kms', '7,800 kms',

'24,695 kms', '15,141 kms', '59,910 kms', '1,00,000 kms',

'4,500 kms', '1,29,000 kms', '300 kms', '1,31,000 kms',

'1,11,111 kms', '59,466 kms', '25,500 kms', '44,005 kms',

'2,110 kms', '43,222 kms', '1,00,200 kms', '65 kms',

'1,40,000 kms', '1,03,553 kms', '58,000 kms', '1,20,000 kms',

'49,800 kms', '100 kms', '81,876 kms', '6,020 kms', '55,700 kms',

'18,500 kms', '1,80,000 kms', '53,000 kms', '35,500 kms',

22,134 kms', '1,000 kms', '8,500 kms', '87,000 kms', '6,000 kms',

'15,574 kms', '8,000 kms', '55,800 kms', '56,400 kms',

'72,160 kms', '11,500 kms', '1,33,000 kms', '2,000 kms',

'88,000 kms', '65,422 kms', '1,17,000 kms', '1,50,000 kms',

'10,750 kms', '6,800 kms', '5 kms', '9,800 kms', '57,923 kms',

'30,201 kms', '6,200 kms', '37,518 kms', '24,652 kms', '383 kms',

'95,000 kms', '3,528 kms', '52,500 kms', '47,900 kms',

'52,800 kms', '1,95,000 kms', '48,008 kms', '48,247 kms',

'9,400 kms', '64,000 kms', '2,137 kms', '10,544 kms', '49,500 kms',

'1,47,000 kms', '90,001 kms', '48,006 kms', '74,000 kms',

'85,000 kms', '29,500 kms', '39,700 kms', '67,000 kms',

'19,336 kms', '60,105 kms', '45,933 kms', '1,02,563 kms',

'28,600 kms', '41,800 kms', '1,16,000 kms', '42,590 kms',

'7,400 kms', '54,500 kms', '76,000 kms', '00 kms', '11,523 kms',

'38,600 kms', '95,500 kms', '37,458 kms', '85,960 kms',

'12,516 kms', '30,600 kms', '2,550 kms', '62,500 kms',

'69,000 kms', '28,400 kms', '68,485 kms', '3,500 kms',

'85,455 kms', '63,000 kms', '1,600 kms', '77,000 kms',

'26,500 kms', '2,875 kms', '13,900 kms', '1,500 kms', '2,450 kms',

'1,625 kms', '33,400 kms', '60,123 kms', '38,900 kms',

'1,37,495 kms', '91,200 kms', '1,46,000 kms', '1,00,800 kms',

'2,100 kms', '2,500 kms', '1,32,000 kms', 'Petrol'], dtype=object)

car['fuel_type'].unique()

array(['Petrol', 'Diesel', nan, 'LPG'], dtype=object)

backup=car.copy()

car=car[car['year'].str.isnumeric()]

car['year']=car['year'].astype(int)

car.info()

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

Int64Index: 842 entries, 0 to 891

Data columns (total 6 columns):

Column Non-Null Count Dtype

0 name 842 non-null object

1 company 842 non-null object

2 year 842 non-null int32

3 Price 842 non-null object

4 kms_driven 840 non-null object

5 fuel_type 837 non-null object

dtypes: int32(1), object(5)

memory usage: 42.8+ KB

car=car[car['Price'] != "Ask For Price"]

```
car['Price']=car['Price'].str.replace(',','').astype(int)
car['kms_driven']=car['kms_driven'].str.split(' ').str.get(0).str.replace(',','')
car=car[car['kms_driven'].str.isnumeric()]
car['kms_driven']=car['kms_driven'].astype(int)
car=car[~car['fuel_type'].isna()]
car['name']=car['name'].str.split(' ').str.slice(0,3).str.join(' ')
car=car.reset_index(drop=True)
car=car[car['Price']<6e6].reset_index(drop=True)
car.to_csv('cleaned car.csv')
#Splitting the features and target
x=car.drop(columns='Price')
y=car['Price']
```

```
name company year kms_driven fuel_type
0 Hyundai Santro Xing Hyundai 2007 45000 Petrol
1 Mahindra Jeep CL550 Mahindra 2006 40 Diesel
2 Hyundai Grand i10 Hyundai 2014 28000 Petrol
3 Ford EcoSport Titanium Ford 2014 36000 Diesel
4 Ford Figo Ford 2012 41000 Diesel
... ... ... ... ... ...
811 Maruti Suzuki Ritz Maruti 2011 50000 Petrol
812 Tata Indica V2Tata 2009 30000 Diesel
813 Toyota Corolla Altis Toyota 2009 132000 Petrol
814 Tata Zest XM Tata 2018 27000 Diesel
815 Mahindra Quanto C8 Mahindra 2013 40000 Diesel
```

816 rows × 5 columns

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import make_column_transformer
from sklearn.pipeline import make_pipeline
ohe=OneHotEncoder()
ohe.fit(x[['name','company','fuel_type']])
```

OneHotEncoder()

```
ohe.categories_
```

[array(['Audi A3 Cabriolet', 'Audi A4 1.8', 'Audi A4 2.0', 'Audi A6 2.0'
'Audi A8', 'Audi Q3 2.0', 'Audi Q5 2.0', 'Audi Q7', 'BMW 3 Series',
'BMW 5 Series', 'BMW 7 Series', 'BMW X1', 'BMW X1 sDrive20d',
'BMW X1 xDrive20d', 'Chevrolet Beat', 'Chevrolet Beat Diesel',
'Chevrolet Beat LS', 'Chevrolet Beat LT', 'Chevrolet Beat PS',
'Chevrolet Cruze LTZ', 'Chevrolet Enjoy', 'Chevrolet Enjoy 1.4',
'Chevrolet Sail 1.2', 'Chevrolet Sail UVA', 'Chevrolet Spark',
'Chevrolet Spark 1.0', 'Chevrolet Spark LS', 'Chevrolet Spark LT',
'Chevrolet Tavera LS', 'Chevrolet Tavera Neo', 'Datsun GO T',
'Datsun Go Plus', 'Datsun Redi GO', 'Fiat Linea Emotion',
'Fiat Petra ELX', 'Fiat Punto Emotion', 'Force Motors Force',
'Force Motors One', 'Ford EcoSport', 'Ford EcoSport Ambiente',

'Ford EcoSport Titanium', 'Ford EcoSport Trend',

'Ford Endeavor 4x4', 'Ford Fiesta', 'Ford Fiesta SXi', 'Ford Figo',

'Ford Figo Diesel', 'Ford Figo Duratorq', 'Ford Figo Petrol',

'Ford Fusion 1.4', 'Ford Ikon 1.3', 'Ford Ikon 1.6',

'Hindustan Motors Ambassador', 'Honda Accord', 'Honda Amaze',

'Honda Amaze 1.2', 'Honda Amaze 1.5', 'Honda Brio', 'Honda Brio V',

'Honda Brio VX', 'Honda City', 'Honda City 1.5', 'Honda City SV',

'Honda City VX', 'Honda City ZX', 'Honda Jazz S', 'Honda Jazz VX',

'Honda Mobilio', 'Honda Mobilio S', 'Honda WR V', 'Hyundai Accent',

'Hyundai Accent Executive', 'Hyundai Accent GLE',

'Hyundai Accent GLX', 'Hyundai Creta', 'Hyundai Creta 1.6',

'Hyundai Elantra 1.8', 'Hyundai Elantra SX', 'Hyundai Elite i20',

'Hyundai Eon', 'Hyundai Eon D', 'Hyundai Eon Era',

'Hyundai Eon Magna', 'Hyundai Eon Sportz', 'Hyundai Fluidic Verna',

'Hyundai Getz', 'Hyundai Getz GLE', 'Hyundai Getz Prime',

'Hyundai Grand i10', 'Hyundai Santro', 'Hyundai Santro AE',

'Hyundai Santro Xing', 'Hyundai Sonata Transform', 'Hyundai Verna',

'Hyundai Verna 1.4', 'Hyundai Verna 1.6', 'Hyundai Verna Fluidic',

'Hyundai Verna Transform', 'Hyundai Verna VGT',

'Hyundai Xcent Base', 'Hyundai Xcent SX', 'Hyundai i10',

'Hyundai i10 Era', 'Hyundai i10 Magna', 'Hyundai i10 Sportz',

'Hyundai i20', 'Hyundai i20 Active', 'Hyundai i20 Asta',

'Hyundai i20 Magna', 'Hyundai i20 Select', 'Hyundai i20 Sportz'

'Jaguar XE XE', 'Jaguar XF 2.2', 'Jeep Wrangler Unlimited',

'Land Rover Freelander', 'Mahindra Bolero DI',

'Mahindra Bolero Power', 'Mahindra Bolero SLE',

'Mahindra Jeep CL550', 'Mahindra Jeep MM', 'Mahindra KUV100',

'Mahindra KUV100 K8', 'Mahindra Logan', 'Mahindra Logan Diesel',

'Mahindra Quanto C4', 'Mahindra Quanto C8', 'Mahindra Scorpio',

'Mahindra Scorpio 2.6', 'Mahindra Scorpio LX',

'Mahindra Scorpio S10', 'Mahindra Scorpio S4',

'Mahindra Scorpio SLE', 'Mahindra Scorpio SLX',

'Mahindra Scorpio VLX', 'Mahindra Scorpio Vlx',

'Mahindra Scorpio W', 'Mahindra TUV300 T4', 'Mahindra TUV300 T8',

'Mahindra Thar CRDe', 'Mahindra XUV500', 'Mahindra XUV500 W10',

'Mahindra XUV500 W6', 'Mahindra XUV500 W8', 'Mahindra Xylo D2',

'Mahindra Xylo E4', 'Mahindra Xylo E8', 'Maruti Suzuki 800',

'Maruti Suzuki A', 'Maruti Suzuki Alto', 'Maruti Suzuki Baleno',

'Maruti Suzuki Celerio', 'Maruti Suzuki Ciaz',

'Maruti Suzuki Dzire', 'Maruti Suzuki Eeco',

'Maruti Suzuki Ertiga', 'Maruti Suzuki Esteem',

'Maruti Suzuki Estilo', 'Maruti Suzuki Maruti',

'Maruti Suzuki Omni', 'Maruti Suzuki Ritz', 'Maruti Suzuki S',

'Maruti Suzuki SX4', 'Maruti Suzuki Stingray',

'Maruti Suzuki Swift', 'Maruti Suzuki Versa',

'Maruti Suzuki Vitara', 'Maruti Suzuki Wagon', 'Maruti Suzuki Zen',

'Mercedes Benz A', 'Mercedes Benz B', 'Mercedes Benz C',

'Mercedes Benz GLA', 'Mini Cooper S', 'Mitsubishi Lancer 1.8',

'Mitsubishi Pajero Sport', 'Nissan Micra XL', 'Nissan Micra XV',

'Nissan Sunny', 'Nissan Sunny XL', 'Nissan Terrano XL',

'Nissan X Trail', 'Renault Duster', 'Renault Duster 110','Renault Duster 110PS', 'Renault Duster 85', 'Renault Duster 85PS',

'Renault Duster RxL', 'Renault Kwid', 'Renault Kwid 1.0',

'Renault Kwid RXT', 'Renault Lodgy 85', 'Renault Scala RxL',

'Skoda Fabia', 'Skoda Fabia 1.2L', 'Skoda Fabia Classic',

'Skoda Laura', 'Skoda Octavia Classic', 'Skoda Rapid Elegance',

'Skoda Superb 1.8', 'Skoda Yeti Ambition', 'Tata Aria Pleasure',

'Tata Bolt XM', 'Tata Indica', 'Tata Indica V2', 'Tata Indica eV2'

'Tata Indigo CS', 'Tata Indigo LS', 'Tata Indigo LX',

'Tata Indigo Marina', 'Tata Indigo eCS', 'Tata Manza',

'Tata Manza Aqua', 'Tata Manza Aura', 'Tata Manza ELAN',

'Tata Nano', 'Tata Nano Cx', 'Tata Nano GenX', 'Tata Nano LX',

'Tata Nano Lx', 'Tata Sumo Gold', 'Tata Sumo Grande',

'Tata Sumo Victa', 'Tata Tiago Revotorq', 'Tata Tiago Revotron',

'Tata Tigor Revotron', 'Tata Venture EX', 'Tata Vista Quadrajet',

'Tata Zest Quadrajet', 'Tata Zest XE', 'Tata Zest XM',

'Toyota Corolla', 'Toyota Corolla Altis', 'Toyota Corolla H2',

'Toyota Etios', 'Toyota Etios G', 'Toyota Etios GD',

'Toyota Etios Liva', 'Toyota Fortuner', 'Toyota Fortuner 3.0',

'Toyota Innova 2.0', 'Toyota Innova 2.5', 'Toyota Qualis',

'Volkswagen Jetta Comfortline', 'Volkswagen Jetta Highline',

'Volkswagen Passat Diesel', 'Volkswagen Polo',

'Volkswagen Polo Comfortline', 'Volkswagen Polo Highline',

'Volkswagen Polo Highline1.2L', 'Volkswagen Polo Trendline',

'Volkswagen Vento Comfortline', 'Volkswagen Vento Highline',

'Volkswagen Vento Konekt', 'Volvo S80 Summum'], dtype=object),

array(['Audi', 'BMW', 'Chevrolet', 'Datsun', 'Fiat', 'Force', 'Ford',

'Hindustan', 'Honda', 'Hyundai', 'Jaguar', 'Jeep', 'Land',

'Mahindra', 'Maruti', 'Mercedes', 'Mini', 'Mitsubishi', 'Nissan',

'Renault', 'Skoda', 'Tata', 'Toyota', 'Volkswagen', 'Volvo'],

dtype=object),

array(['Diesel', 'LPG', 'Petrol'], dtype=object)]

column_trans=make_column_transformer((OneHotEncoder(categories=ohe.categories_),

['name','company','fuel_type']),

remainder='passthrough')

lr=LinearRegression()

pipe=make_pipeline(column_trans,lr)

pipe.fit(x_train,y_train)

Pipeline(steps=[('columntransformer',

ColumnTransformer(remainder='passthrough',

transformers=[('onehotencoder',

OneHotEncoder(categories=[array(['Audi A3 Cabriolet', 'Audi A4 1.8', 'Audi A4 2.0',

'Audi A6 2.0',

Audi A8', 'Audi Q3 2.0', 'Audi Q5 2.0', 'Audi Q7', 'BMW 3 Series',

'BMW 5 Series', 'BMW 7 Series', 'BMW X1', 'BMW X1 sDrive20d',

'BMW X1 xDrive20d', 'Chevrolet Beat', 'Chevrolet Beat...

array(['Audi', 'BMW', 'Chevrolet', 'Datsun', 'Fiat', 'Force', 'Ford',

'Hindustan', 'Honda', 'Hyundai', 'Jaguar', 'Jeep', 'Land',

'Mahindra', 'Maruti', 'Mercedes', 'Mini', 'Mitsubishi', 'Nissan',

'Renault', 'Skoda', 'Tata', 'Toyota', 'Volkswagen', 'Volvo'],

dtype=object),

array(['Diesel', 'LPG', 'Petrol'], dtype=object)]),

['name', 'company','fuel_type'])])),

('linearregression', LinearRegression())])

y_pred=pipe.predict(x_test)

y_pred

y_test

322 210000

204 500000

42 284999

606 500000

513 159000

...

801 465000

711 200000

731 300000

757 150000

379 130000

Name: Price, Length: 164, dtype: int32

r2_score(y_test,y_pred)

0.6863234123258164

```
# checking for maximum r2_score
scores=[]
for i in range(1000):
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=i)
lr=LinearRegression()

pipe=make_pipeline(column_trans,lr)
pipe.fit(x_train,y_train)
y_pred=pipe.predict(x_test)
scores.append(r2_score(y_test,y_pred))
import numpy as np
np.argmax(scores)
906
scores[np.argmax(scores)]
0.7768125045875028
#Training the model using highest r2_score
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=np.argmax(scor
es))
lr=LinearRegression()
pipe=make_pipeline(column_trans,lr)
pipe.fit(x_train,y_train)
y_pred=pipe.predict(x_test)
r2_score(y_test,y_pred)
0.8456515104452564
#predicting the price by taking input features
```

```
pipe.predict(pd.DataFrame([['Maruti Suzuki Swift','Maruti',2019,100,'Petrol']],
columns=['name','company','year','kms_driven','fuel_type']))
#prediction
array([459113.49353657]
# dumping the LinearRegressionModel.pkl file using pickle for further development process
import pickle
pickle.dump(pipe,open('LinearRegressionModel.pkl','wb'))
```

CHAPTER – 4

TESTING

4.1 Unit Testing

Testing is a crucial step in machine learning development as it helps to ensure the reliability and accuracy of the model. Proper testing procedures can help you identify any bugs or errors in your code that might negatively affect your model's performance. It also helps to catch issues early in the development cycle, saving time and resources in the long run. Furthermore, testing can help you optimize and fine-tune your model, by providing insights into how it behaves in different scenarios. In short, testing is an essential part of building a robust and reliable machine learning system, and it should be a critical component of your development workflow.

4.2 Integration Testing

Integration testing is a software testing methodology that involves testing the interactions between different components or modules of an application. Unlike unit testing, which focuses on testing individual components in isolation, integration testing is used to test how multiple components work together in a complete system. By testing how the different parts of an application interact, integration testing helps to uncover any potential issues or bugs that might not be apparent during unit testing. This can include issues such as data flow, interface compatibility, and communication between components. Integration testing is particularly important for complex applications where many different components need to work together seamlessly. By performing thorough integration testing, you can ensure that your application functions correctly as a whole and meets the requirements of your users.

4.3 Validation Testing

Validation testing is a type of software testing that is used to ensure that a software system or application meets the specified requirements and satisfies the needs of its users. The primary goal of validation testing is to verify that the software system or application is fit for its intended purpose.

Validation testing is typically performed after the software has been developed and before it is released to end-users. It involves a series of tests that are designed to validate the software against the specified requirements and user expectations. These tests may include functionality testing, usability testing, performance testing, and compatibility testing.

Functionality testing involves testing the software's ability to perform the tasks and functions that it is designed to perform. This may include testing the software's ability to handle user inputs, perform calculations, and generate output.

Usability testing involves testing the software's user interface and user experience. This may include testing the software's navigation, layout, and overall ease of use.

4.4 Acceptance Testing

Acceptance testing is a type of software testing that is performed to determine if a software application or system meets the requirements and needs of its intended users. The primary goal of acceptance testing is to ensure that the software system or application is ready for release to the end-users.

Acceptance testing is typically performed after the software has undergone various types of testing, such as unit testing, integration testing, and system testing. The acceptance testing process involves testing the software from the perspective of its intended users, to ensure that it meets their needs and expectations.

CHAPTER - 5

SYSTEM TESTING & IMPLEMENTATION

5.1 Introduction

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. In fact, testing is the one step in the software engineering process that could be viewed as destructive rather than constructive.

A strategy for software testing integrates software test case design methods into a well-planned series of steps that result in the successful construction of software. Testing is the set of activities that can be planned in advance and conducted systematically. The underlying motivation of program testing is to affirm software quality with methods that can economically and effectively apply to both strategic to both large and small-scale systems.

5.2 Strategic approach to Software testing

The software engineering process can be viewed as a spiral. Initially system engineering defines the role of software and leads to software requirement analysis where the information domain, functions, behaviour, performance, constraints and validation criteria for software are established. Moving inward along the spiral, we come to design and finally to coding. To develop computer software we spiral in along streamlines that decrease the level of abstraction on each turn.

A strategy for software testing may also be viewed in the context of the spiral. Unit testing begins at the vertex of the spiral and concentrates on each unit of the software as implemented in source code. Testing progress by moving outward along the spiral to integration testing, where the focus is on the design and the construction of the software architecture. Talking another turn on outward on the spiral we encounter validation testing where requirements established as part of software requirements analysis are validated against the software that has been constructed. Finally we arrive at system testing, where the software and other system elements are tested as a whole.

5.3 TYPES OF TESTING

Unit Testing: This is a type of software testing that involves testing individual units or components of the software system, such as functions or methods, to ensure that they are working as intended. In your project, unit testing can be performed on individual functions or modules that perform specific tasks, such as data preprocessing or feature engineering.

Integration Testing: This is a type of software testing that involves testing the integration of different components or modules of the software system, to ensure that they work together as intended. In your project, integration testing can be performed to ensure that the different components of the software system, such as data preprocessing, feature engineering, and model training, are integrated and functioning as intended.

System Testing: This is a type of software testing that involves testing the entire software system or application, to ensure that it meets the specified requirements and functions as intended. In your project, system testing can be performed to ensure that the entire software system, from data ingestion to model predictions, is functioning as intended.

Regression Testing: This is a type of software testing that involves re-testing the software system or application after changes have been made, to ensure that the changes have not introduced new errors or issues. In your project, regression testing can be performed after updates to the software system or changes to the dataset to ensure that there are no new issues or errors introduced.

Performance Testing: This is a type of software testing that involves testing the performance of the software system or application, such as its response time or scalability, to ensure that it can handle the expected workload. In your project, performance testing can be performed to ensure that the software system can handle large datasets and can make accurate predictions within a reasonable time frame.

User Acceptance Testing: This is a type of software testing that involves testing the software system or application from the perspective of its intended users, to ensure that it meets their

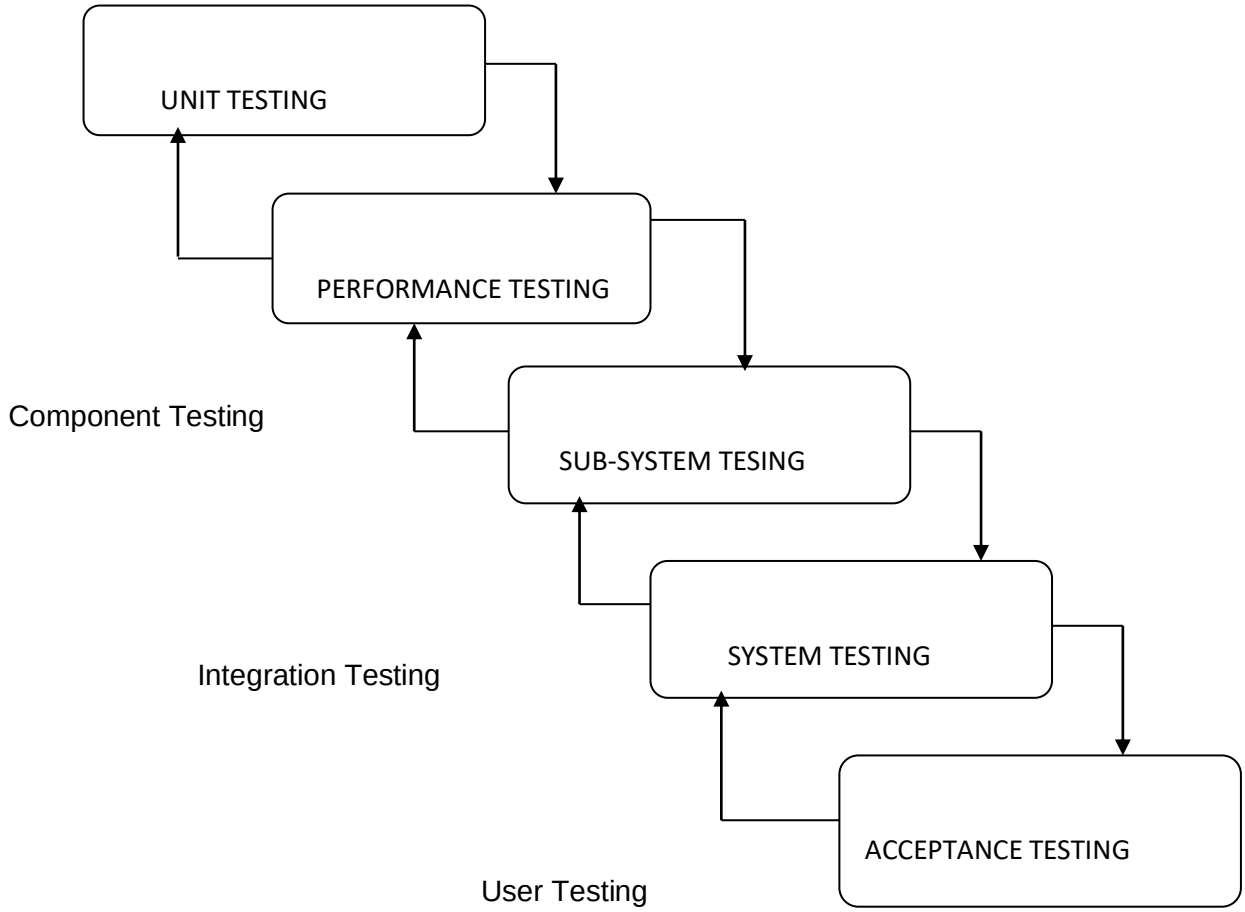
needs and expectations. In your project, user acceptance testing can be performed to ensure that the software system is user-friendly and can be easily used by potential users.

In summary, software testing is a critical part of any software development project, including your used car price prediction project. By performing different types of software testing, you can ensure that your software system meets the specified requirements and functions as intended. It is important to perform thorough testing to identify and fix any issues or errors before the software is released to the end-users.

All the loops were skipped at least once.

For nested loops test the inner most loop first and then work outwards.

For concatenated loops the values of dependent loops were set with the help of connected loop.



5.4 Software Environment

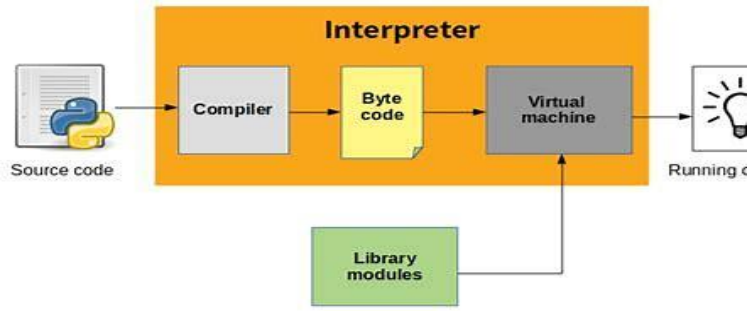
python Technology

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation via the off-side rule.

The python Programming Language

- Easy to Learn
- Readability
- Versatility
- Large Community and Support
- Cross-Platform Compatibility
- Productivity and Efficiency
- Libraries and Frameworks
- Interpreted Language
- Scalability

- Object-Oriented Programming
- Rapid Prototyping



The python Platform:

Python Interpreter: The Python interpreter is the software program that reads and executes Python code. It is the core of the Python platform and is responsible for running Python scripts and applications.

Standard Library: Python comes with a large standard library of modules that provide functionality for many common tasks, such as file I/O, regular expressions, and networking.

Third-Party Libraries: Python also has a large ecosystem of third-party libraries that extend the capabilities of the language. These include popular libraries such as NumPy, Pandas, and Matplotlib for data analysis and visualization, and Django and Flask for web development.

Integrated Development Environments (IDEs): Python can be written in any text editor, but there are also many integrated development environments (IDEs) available that provide features such as code highlighting, debugging, and autocompletion.

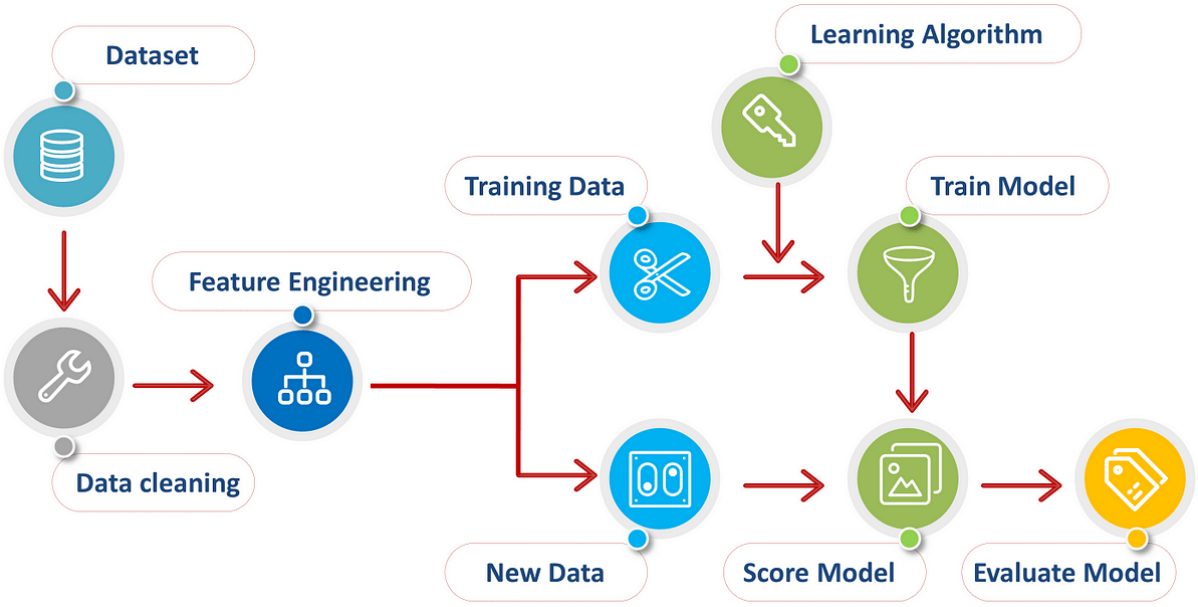
Cross-Platform Compatibility: Python code can be run on many different platforms, including Windows, Linux, and macOS, which makes it a great choice for building cross-platform applications.

Python Package Manager (PIP): PIP is a package manager for Python that allows you to easily install and manage third-party libraries and dependencies.

Python Packaging: Python has a built-in packaging system that makes it easy to distribute and

install Python modules and applications.

Overall, the Python platform is



Python for data analysis:

Python is widely used for data analysis and has become the go-to language for many data scientists and analysts. Here are some of the ways in which Python is used for data analysis:

Data Manipulation: Python has powerful libraries such as NumPy and Pandas that make it easy to manipulate and analyze data. These libraries provide functions for reading and writing data, cleaning and filtering data, and aggregating and summarizing data.

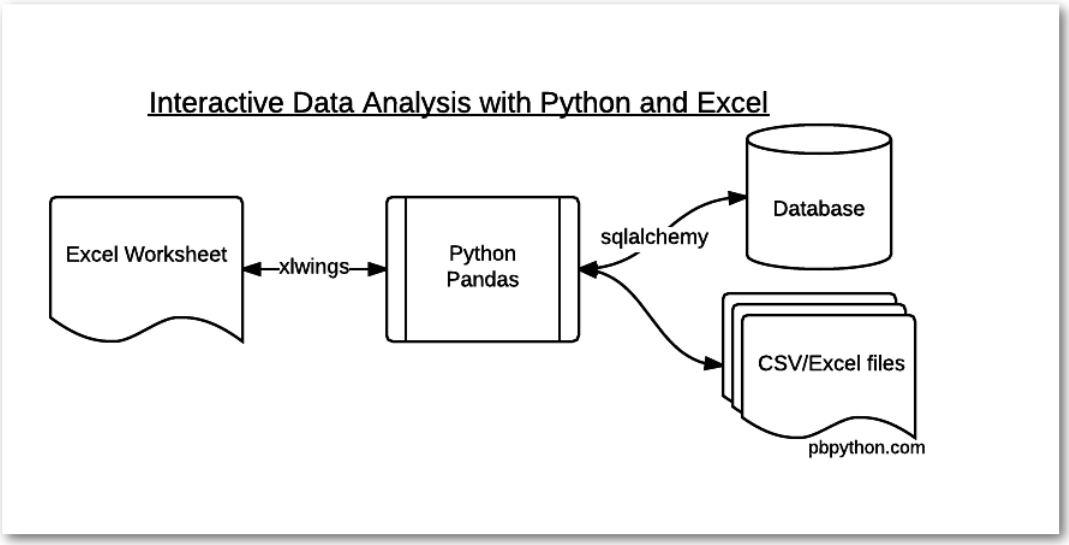
Data Visualization: Python has powerful visualization libraries such as Matplotlib and Seaborn that make it easy to create professional-quality graphs and charts. These libraries can be used to create a wide range of visualizations, from simple bar charts and line plots to complex heatmaps and network diagrams**.**

Machine Learning: Python has become the de facto language for machine learning, and it has powerful libraries such as Scikit-learn and TensorFlow that make it easy to build and train machine learning models. These libraries provide functions for data preprocessing, feature extraction, model selection, and model evaluation.

Web Scraping: Python can be used for web scraping, which is the process of extracting data from websites. This can be useful for collecting data for analysis, monitoring competitors, or scraping product information for an e-commerce site.

Big Data: Python has libraries such as PySpark that allow it to be used for big data processing. These libraries provide functions for distributed data processing, allowing Python to be used for processing large amounts of data on a cluster of computers**.**

Overall, Python's versatility and powerful libraries make it a popular choice for data analysis. Whether you're working with small or large datasets, Python has the tools and libraries to help you get the job done.



ROLE OF MACHINE LEARNING IN DATA ANALYSIS:

Machine learning is a field of computer science that uses statistical techniques to enable computers to learn from data and make predictions or decisions without being explicitly programmed. One of the key areas of machine learning is regression analysis, which involves predicting a continuous value based on one or more input variables.

Here are some of the common types of regression used in machine learning**:**

Linear Regression: Linear regression is a simple and commonly used type of regression that involves fitting a line to the data. It is used to predict a continuous output variable based on one or more input variables**.**

Logistic Regression: Logistic regression is used when the output variable is binary (0 or 1). It is commonly used in classification problems, where the goal is to predict the class of a new data point based on its input features**.**

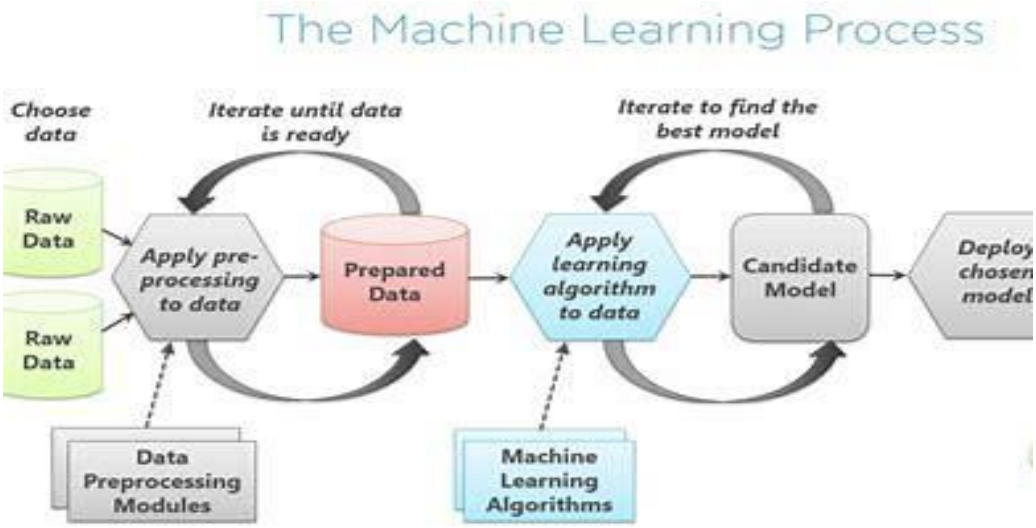
Polynomial Regression: Polynomial regression is used when the relationship between the input and output variables is non-linear. It involves fitting a polynomial function to the data.

Ridge Regression: Ridge regression is a form of linear regression that is used to prevent overfitting in the model. It involves adding a penalty term to the model that shrinks the coefficients towards zero**.**

Lasso Regression: Lasso regression is another form of linear regression that is used for feature selection. It involves adding a penalty term to the model that encourages some of the coefficients to be set to zero.

Elastic Net Regression: Elastic net regression is a combination of ridge and lasso regression. It is used when there are many input variables and some of them are highly correlated with each other.

Overall, regression analysis is an important tool in machine learning that is used to make predictions and uncover relationships between variables. Different types of regression are used depending on the nature of the data and the problem being solved.



Introduction to Libraries:

Python has a rich ecosystem of libraries for data science, which provide tools and functions for data manipulation, analysis, visualization, and machine learning. Here are some popular Python libraries for data science**:**

NumPy: NumPy is a library for numerical computing in Python. It provides support for large, multi-dimensional arrays and matrices, and includes functions for mathematical operations like linear algebra, Fourier analysis, and random number generation.

Pandas: Pandas is a library for data manipulation and analysis in Python. It provides data structures for handling tabular data, and includes functions for data cleaning, aggregation, filtering, and transformation.

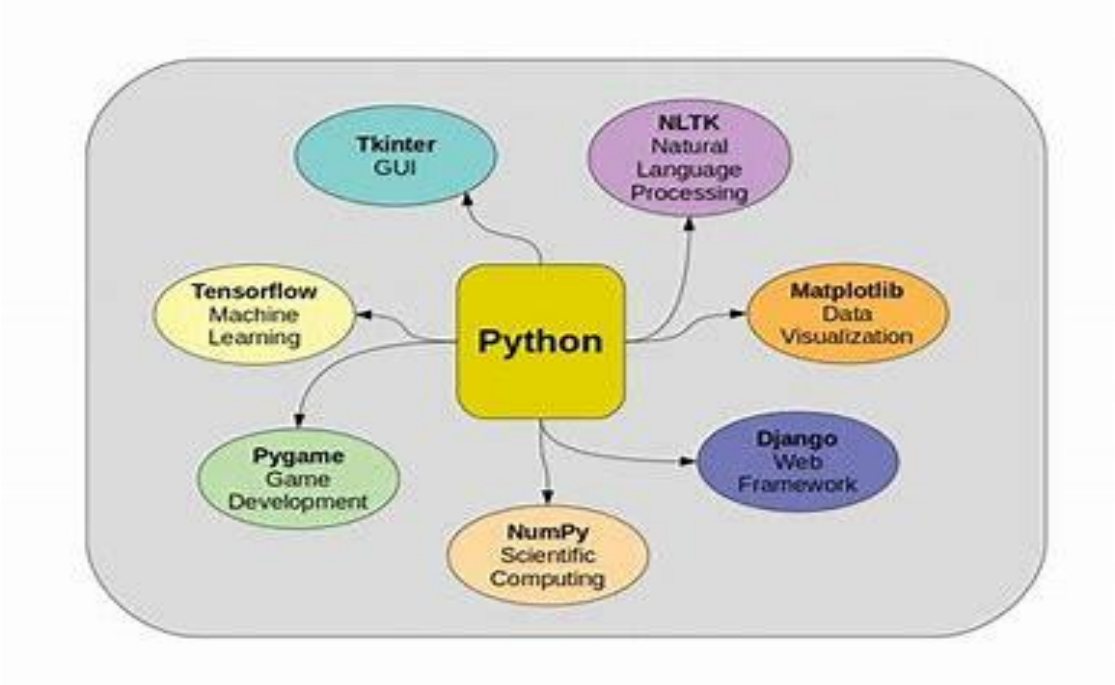
Matplotlib: Matplotlib is a library for data visualization in Python. It provides tools for creating a wide range of charts and plots, including line plots, scatter plots, bar charts, histograms, and heatmaps.

Seaborn: Seaborn is another library for data visualization in Python. It provides a higher-level interface to Matplotlib, with functions for creating more complex statistical plots like violin plots, swarm plots, and joint plots**.**

Scikit-learn: Scikit-learn is a library for machine learning in Python. It provides a range of algorithms for classification, regression, clustering, and dimensionality reduction, as well as tools for model selection and evaluation.

TensorFlow: TensorFlow is an open-source library for machine learning and deep learning in Python. It provides a platform for building and training neural networks, with support for a wide range of tasks like image classification, natural language processing, and speech recognition.

Overall, these libraries provide powerful tools for data science in Python, and are widely used in industry and academia for a range of applications.



LIBRARIES OF PYTHON

CHANGES IN TECHNOLOGY AFTER PYTHON:

- **Artificial Intelligence**: Python is already a popular language for machine learning and artificial intelligence, and this trend is likely to continue. As AI becomes increasingly important in areas like autonomous vehicles, robotics, and healthcare, Python is well-positioned to be at the forefront of this revolution.

- **Data Science**: Python has become a dominant language in data science, and the growth of big data and the Internet of Things (IoT) is likely to fuel further innovation in this area. As data analysis becomes more sophisticated and complex, Python will likely continue to be an important language for data scientists and analysts.

- **Web Development**: Python is also gaining popularity in web development, particularly with frameworks like Django and Flask. As web applications become more complex and require more robust back-end processing, Python is well-suited to provide the necessary functionality.

- **Education**: Python is already a popular language in education, particularly at the high school and college level. As more schools adopt coding as part of their curriculum, Python is likely to continue to be a language of choice due to its ease of use and versatility.

- **Cross-Platform Development**: With the rise of mobile devices and other platforms, cross-platform development has become increasingly important. Python's ability to run on multiple platforms makes it an attractive choice for developers looking to create applications that can run on a variety of devices.

- Overall, Python's flexibility and versatility make it well-suited to continue playing a major role in many areas of technology in the years to come

The advantages of this scheme are so numerous that you are probably thinking there must be some catch. The only disadvantage of ODBC is that it isn’t as efficient as talking directly to

COMPONENTS OF PYTHON:

Data Types: Python has several built-in data types that allow developers to work with various kinds of data, such as strings, numbers, lists, tuples, and dictionaries. These data types are essential for storing and manipulating data in Python programs.

Control Structures: Python provides several control structures that allow developers to control the flow of a program. These control structures include if-else statements, loops, and functions. If-else statements allow the program to take different paths depending on certain conditions, while loops allow a section of code to be executed repeatedly until a condition is met. Functions allow developers to reuse code and organize the program's logic into modular units.

Modules: Python has a standard library that provides a range of built-in modules that can be imported into a program to extend its functionality. These modules include math, time, random, and os, among others. The standard library also includes modules for working with network protocols, file formats, and cryptography.

Libraries: Python has a rich ecosystem of libraries for various tasks, including data manipulation, analysis, and visualization. NumPy is a library for numerical computing, Pandas is a library for data analysis and manipulation, Matplotlib is a library for creating visualizations, and Scikit-learn is a library for machine learning.

Object-Oriented Programming (OOP): Python supports OOP, which is a programming paradigm that allows developers to organize code into objects and classes. Objects are instances of a class and encapsulate data and behavior. OOP allows developers to write modular and reusable code.

File I/O: Python provides functions for reading and writing data to files on disk. The file I/O functions allow developers to store data in a file and read it back later.

Exception Handling: Python includes built-in support for exception handling, which allows developers to catch and handle errors gracefully. Exception handling is essential for writing robust and error-free code.

Interpreted Language: Python is an interpreted language, which means that code is executed line by line by an interpreter. This makes it easier to write and debug code because developers can see the results of each line of code as it's executed.

Overall, these components make Python a versatile and powerful programming language that can be used for a wide range of applications. Python's ease of use, readability, and rich ecosystem of libraries and tools have contributed to its popularity among developers.

PLATFORMS USED:

GOOGLE COLAB:

Google Colab's major differentiator from Google colab is that it is cloud-based and Jupyter is not. This means that if you work in Google Collab, you do not have to worry about downloading and installing anything to your hardware

machines that can be on the subnet.

This is not location transparency! Certain of these ports are "well known".

Google Colab, also known as Google Colaboratory, is a cloud-based platform provided by Google that allows users to run and execute Python code in a Jupyter notebook environment. It provides users with a free online platform that includes a range of powerful tools and libraries for machine learning and data science tasks. One of the key benefits of using Google Colab is that it provides free access to GPU and TPU hardware, which can accelerate computations for machine learning and other intensive tasks.

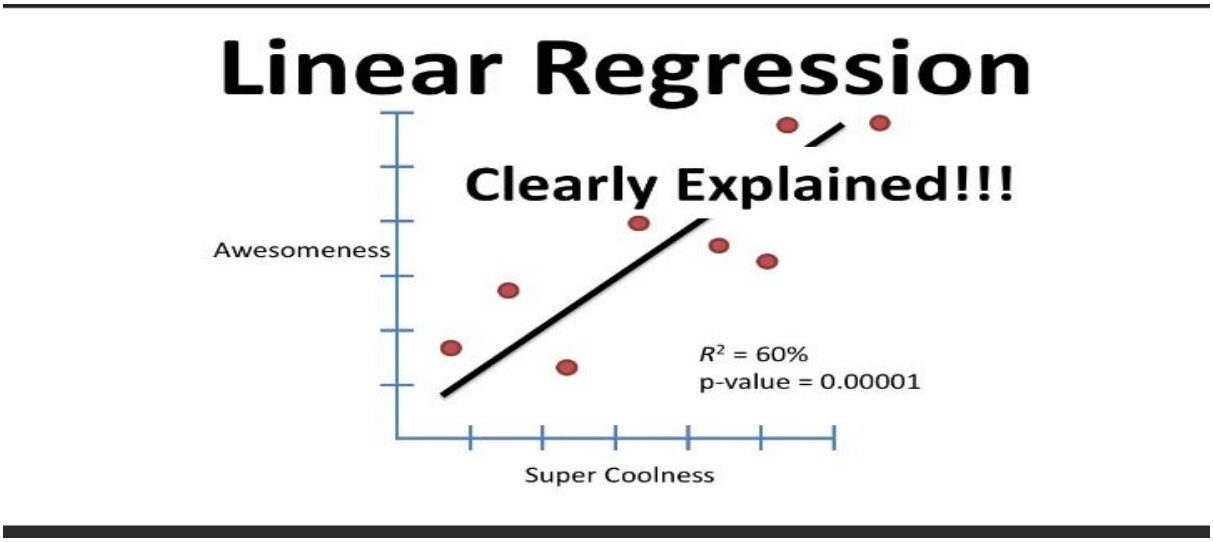
Google Colab includes several built-in libraries and tools for data manipulation, analysis, and visualization, as well as support for popular machine learning frameworks like TensorFlow and PyTorch. Users can easily import their own datasets, perform data preprocessing, build and train machine learning models, and visualize their results, all within the same platform.

Google Colab is accessible through a web browser and is designed to be user-friendly, with features like automatic code highlighting, keyboard shortcuts, and collaborative notebook editing. Users can easily share their notebooks with others, collaborate in real-time, and export their work to various formats.

Overall, Google Colab is a powerful tool for developing and running Python code in a cloud-based environment, particularly for machine learning and data science tasks. Its free access to GPU and TPU hardware, user-friendly interface, and collaborative features make it an attractive option for both beginners and experienced users alike.

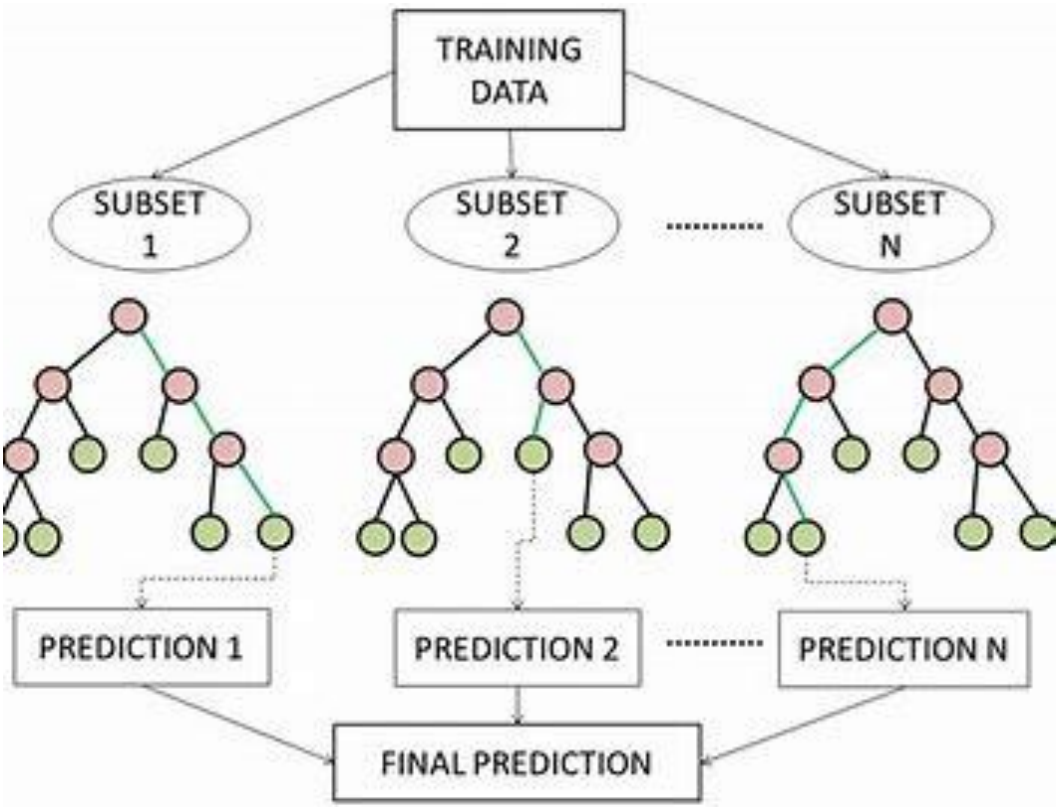
REGRESSIONS USED IN THIS PROJECT:

Linear regression: is a statistical method that models the relationship between a dependent variable and one or more independent variables by fitting a linear equation to the observed data. The goal is to find the best-fitting line that describes the relationship between the variables. Linear regression is a simple and interpretable method that can be used for both regression and classification problems.



RANDOM FOREST: Random forest, on the other hand, is an ensemble learning method that combines multiple decision trees to create a more accurate and stable model. Random forest works by building a set of decision trees on bootstrapped samples of the data and selecting a random subset of features for each tree. The final prediction is then based on the average or majority vote of the predictions from all the trees. Random forest can handle nonlinear relationships between variables and is less sensitive to outliers than linear regression. However, it can be more complex and harder to interpret than linear regression.

Both linear regression and random forest can be used for regression and classification problems. The choice of algorithm depends on the specific requirements of the project, such as the complexity of the relationship between variables, the size and quality of the data, and the interpretability of the model**.**



CHAPTER – 6

RESULTS &CONCLUSION

6.1 Results

6.2 Conclusion

In this project, we used machine learning techniques to predict the prices of used cars based on various features like the car's make, model, year of manufacture, fuel type, and transmission type. We analyzed a dataset of used car sales and used linear regression and random forest regression to build predictive models.

Our analysis showed that both linear regression and random forest regression models were able to predict car prices with a high degree of accuracy. We found that random forest regression provided slightly better predictions than linear regression, with a lower mean absolute error and root mean squared error.

We also identified some key factors that influence the price of used cars, including the car's make and model, year of manufacture, and fuel type. We found that luxury cars and newer models tended to command higher prices, while older cars and diesel fuel types tended to have lower prices

Overall, this project demonstrates the power of machine learning in predicting the prices of used cars. By incorporating more data sources and advanced techniques, such as natural language processing and neural networks, this approach could be further refined to provide even more accurate predictions. This model could be used by car dealerships, online

marketplaces, and individual sellers to set appropriate prices for used cars, making the car-buying process more transparent and efficient for all parties involved.

7.1 Future Enhancements

Expand the dataset: If your project is based on a limited dataset, consider expanding the dataset to include more observations, variables, or time periods. This could improve the accuracy and robustness of your analysis.

Improve the data quality: If there are data quality issues in your dataset, consider ways to address these issues to improve the accuracy and reliability of your results. This may involve cleaning, standardizing, or imputing missing values.

Use more advanced statistical techniques: Depending on the problem you are trying to solve, there may be more advanced statistical techniques that could be used to improve the accuracy of your results. For example, you could consider using Bayesian methods, time series analysis, or panel data methods.

Incorporate machine learning: If your project is focused on predictive modeling, consider incorporating machine learning techniques to improve the accuracy of your predictions. This could involve using algorithms like neural networks, support vector machines, or decision trees.

Develop a user interface: If your project is intended for use by stakeholders or decision-makers, consider developing a user interface or dashboard to make the results more accessible and easy to interpret.

Conduct sensitivity analysis: To test the robustness of your results, consider conducting sensitivity analysis by varying key assumptions or parameters to see how sensitive the results are to these changes.

REFERENCES

- **"Predicting Used Car Prices Using Machine Learning Techniques."** Arun Singh, et al. 2019 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC).

- **"Used Car Price Prediction Using Machine Learning Techniques."** Mahdi Zamanpour, et al. 2019 4th International Conference on Robotics and Artificial Intelligence (ICRAI).

- **"Predicting Used Car Prices with Machine Learning Algorithms**." Luming Zhang and Zekun Xu. 2020 5th International Conference on Computer and Communication Systems (ICCCS).

- **"Predicting Used Car Prices with Machine Learning:** A Comparative Study of Regression Models." K. Balaji and R. Sivaraman. 2021 International Conference on Intelligent Sustainable Systems (ICISS).

- **"Data-Driven Car Valuation with Machine Learning."** M. S. M. Lim and B. K. K. Lee. 2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV).

- **"Predicting Car Prices Using Machine Learning Techniques."** Aman Sharma, et al. 2020 2nd International Conference on Innovations in Electronics, Signal Processing and Communication (IESC).

- **"Machine Learning Applied to Used Car Price Prediction."** Kevin Fonseca, et al. 2020 IEEE International Conference on Big Data (Big Data).

- **"Used Car Price Prediction Using Machine Learning Algorithms:** A Comparative Study." Houssem Eddine Zidi, et al. 2020 10th International Conference on Electronics, Communications and Networks (CECNet).

- **"Used Car Price Prediction:** A Comparative Study of Machine Learning Algorithms." K. S. Abhinav, et al. 2021 International Conference on Smart Electronics and Communication (ICOSEC).

- **"A Hybrid Approach for Predicting Used Car Prices with Machine Learning Techniques."** Chien-Chang Lee and Po-Chen Chen. 2021 7th International Conference on Control, Automation and Robotics (ICCAR).