IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

CAR PRICE PREDICTION USING MACHINE LEARNING

Submitted By:

BIDHAN CHANDRA SEN

UNIVERSITY ROLL NO: 12011222002

GUIDED BY:

Prof. Biswajit Mondal

Dr.B.C.Roy ENGINEERINGCOLLEGE

DEPARTMENT OF COMPUTERSCIENCE AND ENGINEERING

ABSTRACT

The goal of this work is to develop a model that can forecast the fair market value of used automobiles based on a number of factors, such as kms_driven, year of purchase, fuel type, present price of the car, transmission type, seller type, and Owner. In the used car industry, this model can help makers, buyers, and sellers of automobiles. When it's done, using the data that people enter, it may produce a somewhat accurate price estimate. Data science and machine learning are included in the model development process. The used dataset was taken from used vehicle listings through scraping. To attain the best accuracy, a variety of regression techniques were used in the study, including gradient boosting, random forest, support vector, XGB regression, and linear regression.

Keywords: By predicting the car price we can help the customersbuy at a reasonable price.

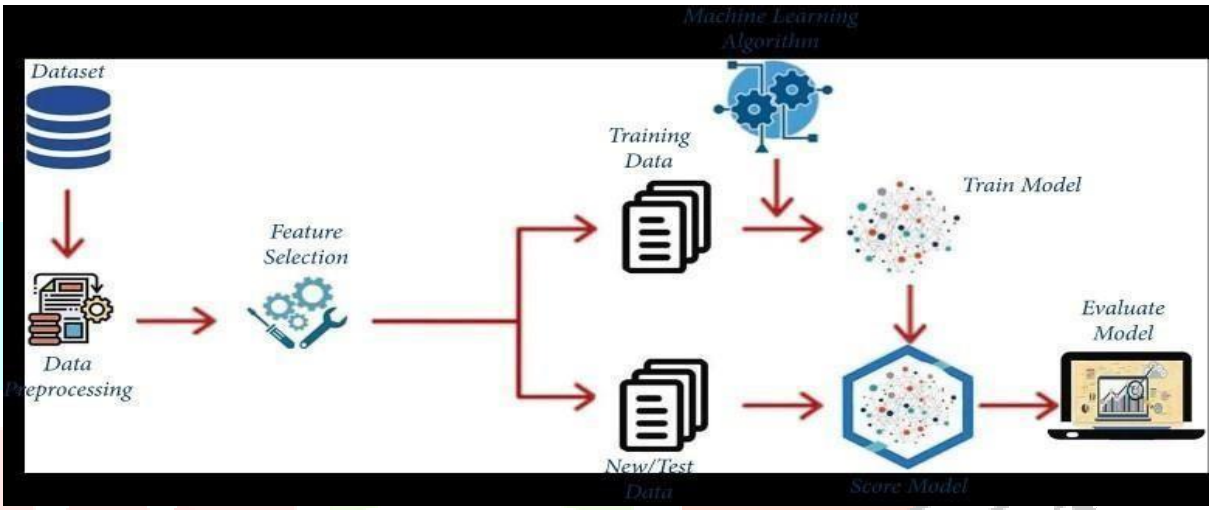
INTRODUCTION

Because there are so many variables that influence a used car's price on the market, it can be difficult to determine whether the quoted price is reasonable. In order to make educated decisions, the goal of this project is to create machine learning models that can precisely forecast the price of a used car based on its properties. We employ and assess diverse learning techniques on a dataset comprising the selling prices of distinct brands and models. We will evaluate the effectiveness of several machine learning methods, including XGB Regressor, Random Forest Regression, Gradient Boosting Regression, and Linear Regression, and select the top performer. We will calculate the car's price based on a number of factors.

Project Objectives

In this project I have a shortened ‘ prediction’ Dataset from Kaggle. We are predicting car costs, Year will be our target feature, which is a continuous attribute.
Our objective in this project is to study the given dataset of ‘Medical Insurance Cost Prediction’. We might need to pre-process the given dataset if we need to. Then, we would train 4 models viz. ‘Linear Regression classifier model’, ’XGB Regressor model’, ’Random Forest Regressor model’, and ‘Gradient Boosting Regression model’.
Our methodology for solving the problems in the given project is described below:

- Load the required dataset.
- Study the dataset.
- Describe the dataset.
- Visualise the dataset.
- Find out if the dataset needs to be pre-processed. o It will be determined on the basis of whether the dataset has null values or outliers or any such discrepancy that might affect the output of the models to be trained.



- If the dataset is required to be pre-processed, take the necessary stepsto pre-process the data.
- Find out the principal attributes for training
- Calculate the accuracy of the 4 models and find out the classification reports.
- Plot the necessary graphs.
- Use each trained model to predict the outcomes of the given test dataset.
- Choose the best model among the 4 trained models bases on the accuracy and classification reports.

Project Scope

The broad scope of ‘Car Price Prediction’ project is given below:

- The given dataset has attributes of the car.
- It is a useful project as the Classifier models can be used to quickly determine the which model is appropriate for Car Price Prediction of large datasets.
- Various banking institution can use these models and modify them according to their needs to use in their insurance. This will reduce the manual labour and time.
- Beneficiary who intend to buy car can use these trained models to check whether the beneficiary get more benefits .
- The dataset given to us is a shortened form of the original dataset from Kaggle. So, the results might have some mismatch with the real-world applications. But that can be avoided if the models are trained

accordingly.

Data Description

Source of the data: Kaggle. The given dataset is a shortened version of the originaldataset in Kaggle.
Data Description: The given train dataset has 301 rows and 9 columns.

Columns	Attribute Name	Type	Description	Target Attribute
Car_Name	Car_Name	Non-Categorical	Name of the Car	No
Year	Year	Non-Categorical	Year of Purchase	Yes
Selling_Price	Selling_Price	Non-Categorical	Selling Price of theCar	No
Present_Price	Present_Price	Non-Categ orical	Present Price of the Car	No
Kms_Driven	Kms_Driven	Non-Categorical	No.of kms driven by the Car	No
Fuel_Type	Fuel_Type	Categorical	Fuel_Type of the Car Petrol,Diesel Or CNG	No
Seller_Type	Seller_Type	Categorical	Seller_Type of the Car Dealer Or Individual	No
Transmission	Transmission	Categorical	Transmission type of the Car Manual Or Automatic	No
Owner	Owner	Categorical	Owner type of the Car First Owner ,Second Owner Or Third Owner	No

The following table shows the 5 number statistics of the given dataset:

	Year	Selling_P rice	Present_Price	Kms_Driven
Count	301.000000	301.000000	301.0000000	301.000000
mean	2013.627907	4.661296	7.628472	36947.205980
std	2.891554	5.082812	8.644115	38886.883882
min	2003.000000	0.100000	0.320000	500.000000
25%	2012.000000	0.900000	1.200000	15000.000000
50%	2014.000000	3.600000	6.400000	32000.000000
75%	2016.000000	6.000000	9.900000	48767.000000
max	2018.000000	35.000000	92.600000	500000.000000

Data Pre-processing

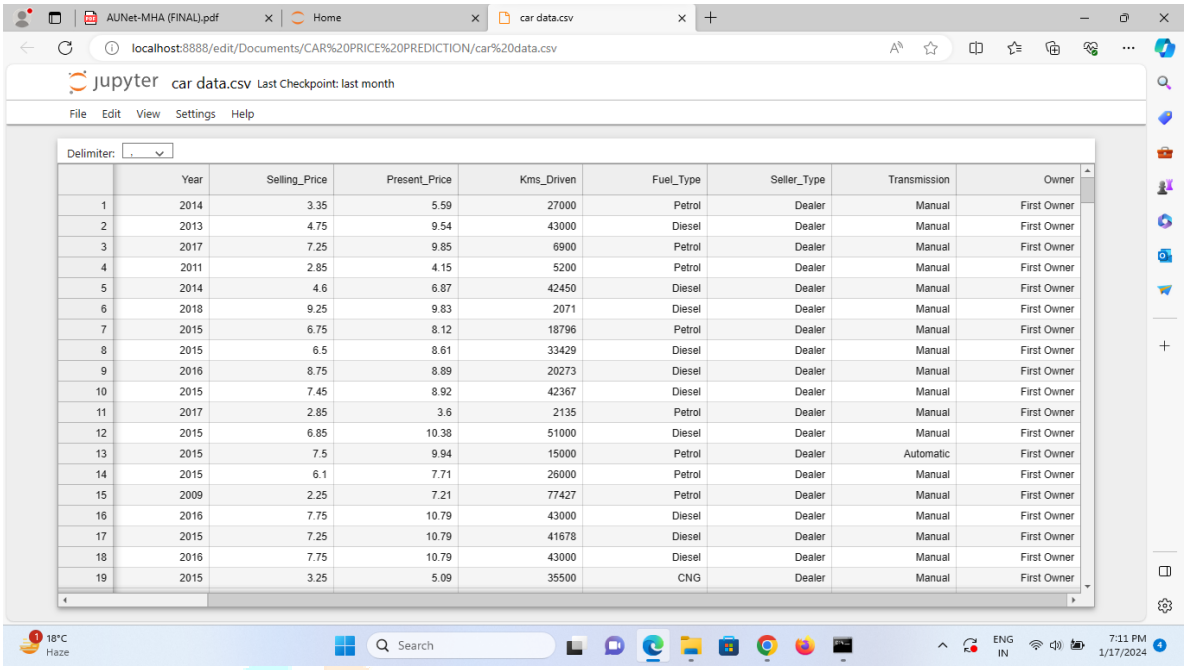
As the given dataset had Categorical and Non-categorical data mixed, we converted the categorical data into non-categorical data accordingly. We converted the binary categories into 0,1 and 2. We converted the other categorical attributes into suitable numerical values. The following table shows the conversion record:

Non-numaric to numeric change table		
Column	Initial Value	Replaced value
Transmission_Type	Manual	0
	Automatic	1
Seller_Type	Dealer	0
	Individual	1
Fuel_Type	Petrol	0
	Diesel	1
	CNG	2
Owner_Type	First_Owner	0
	Second_Owner	1
	Third_Owner	2

We searched for null values in our dataset and formed the following table:

Column name	**Count of null value**
Car_Name	0
Year	0
Selling_Price	0
Present_Price	0
Kms_Driven	0
Fuel_Type	0
Seller_Type	0
Transmission	0
Owner	0

DATASET

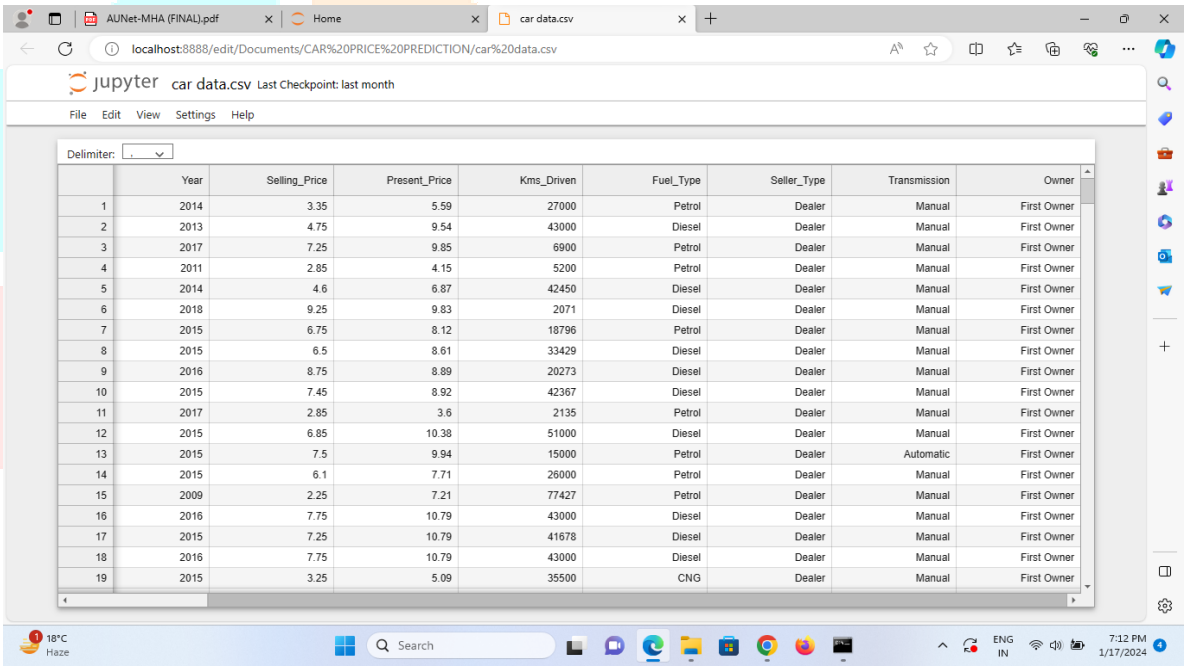


jupyter car data.csv Last Checkpoint: last month

File Edit View Settings Help

Delimiter: ,

	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner
1	2014	3.35	5.59	27000	Petrol	Dealer	Manual	First Owner
2	2013	4.75	9.54	43000	Diesel	Dealer	Manual	First Owner
3	2017	7.25	9.85	6900	Petrol	Dealer	Manual	First Owner
4	2011	2.85	4.15	5200	Petrol	Dealer	Manual	First Owner
5	2014	4.6	6.87	42450	Diesel	Dealer	Manual	First Owner
6	2018	9.25	9.83	2071	Diesel	Dealer	Manual	First Owner
7	2015	6.75	8.12	18796	Petrol	Dealer	Manual	First Owner
8	2015	6.5	8.61	33429	Diesel	Dealer	Manual	First Owner
9	2016	8.75	8.89	20273	Diesel	Dealer	Manual	First Owner
10	2015	7.45	8.92	42367	Diesel	Dealer	Manual	First Owner
11	2017	2.85	3.6	2135	Petrol	Dealer	Manual	First Owner
12	2015	6.85	10.38	51000	Diesel	Dealer	Manual	First Owner
13	2015	7.5	9.94	15000	Petrol	Dealer	Automatic	First Owner
14	2015	6.1	7.71	26000	Petrol	Dealer	Manual	First Owner
15	2009	2.25	7.21	77427	Petrol	Dealer	Manual	First Owner
16	2016	7.75	10.79	43000	Diesel	Dealer	Manual	First Owner
17	2015	7.25	10.79	41678	Diesel	Dealer	Manual	First Owner
18	2016	7.75	10.79	43000	Diesel	Dealer	Manual	First Owner
19	2015	3.25	5.09	35500	CNG	Dealer	Manual	First Owner



jupyter car data.csv Last Checkpoint: last month

File Edit View Settings Help

Delimiter: ,

	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner
1	2014	3.35	5.59	27000	Petrol	Dealer	Manual	First Owner
2	2013	4.75	9.54	43000	Diesel	Dealer	Manual	First Owner
3	2017	7.25	9.85	6900	Petrol	Dealer	Manual	First Owner
4	2011	2.85	4.15	5200	Petrol	Dealer	Manual	First Owner
5	2014	4.6	6.87	42450	Diesel	Dealer	Manual	First Owner
6	2018	9.25	9.83	2071	Diesel	Dealer	Manual	First Owner
7	2015	6.75	8.12	18796	Petrol	Dealer	Manual	First Owner
8	2015	6.5	8.61	33429	Diesel	Dealer	Manual	First Owner
9	2016	8.75	8.89	20273	Diesel	Dealer	Manual	First Owner
10	2015	7.45	8.92	42367	Diesel	Dealer	Manual	First Owner
11	2017	2.85	3.6	2135	Petrol	Dealer	Manual	First Owner
12	2015	6.85	10.38	51000	Diesel	Dealer	Manual	First Owner
13	2015	7.5	9.94	15000	Petrol	Dealer	Automatic	First Owner
14	2015	6.1	7.71	26000	Petrol	Dealer	Manual	First Owner
15	2009	2.25	7.21	77427	Petrol	Dealer	Manual	First Owner
16	2016	7.75	10.79	43000	Diesel	Dealer	Manual	First Owner
17	2015	7.25	10.79	41678	Diesel	Dealer	Manual	First Owner
18	2016	7.75	10.79	43000	Diesel	Dealer	Manual	First Owner
19	2015	3.25	5.09	35500	CNG	Dealer	Manual	First Owner

(Methodology)

Splitting data for training and testing purpose

We split the given train dataset into two parts for training and testing purpose. The split ratio we used is 0.80 which indicates we used 80% data for training purpose and 20% data for testing purpose. We will be using the same split ratio for all the modelstrained.

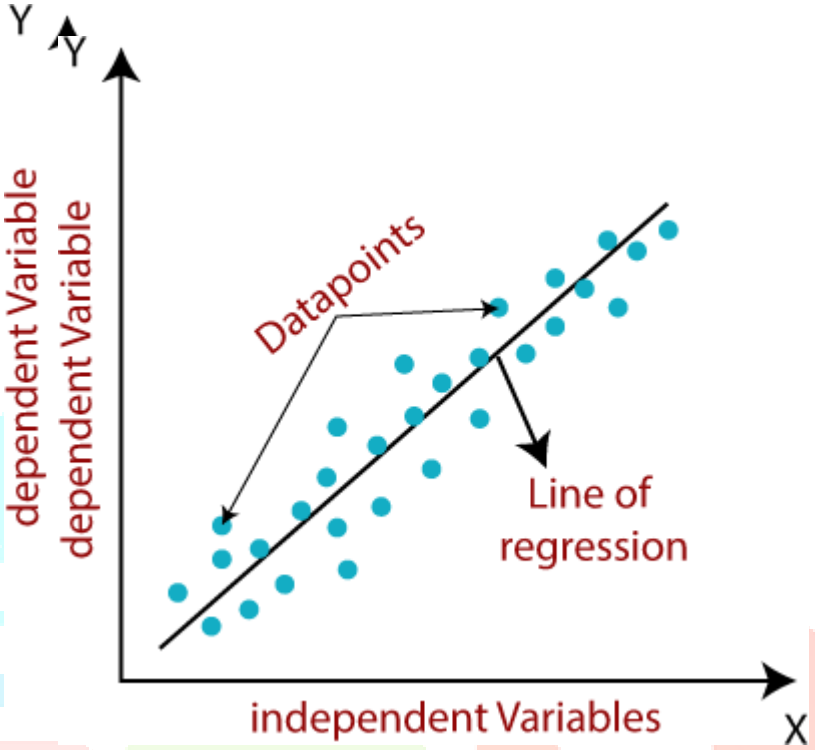
Now we will be training our required models. Our project goal requires us to train specific 4 classifier models viz.

1.Linear Regression Classifier
2.XGB Regression
3.Random Forest Classifier
4.Gradient Boosting Regression

Linear Regression Classifier

What is linear regression?
Linear regression analysis is used to predict the value of a variable based on the value of another variable. The variable you want to predict is called the dependent variable. The variable you are using to predict the other variable's value is called the independent variable.

# Loading the Linear Regression Model LR =LinearRegression() [34]:
LR.fit(X_train,Y_train) LinearRegression()



Making Prediction with Linear Regression Model:

Given the representation is a linear equation, making predictions is as simple as solving the equation for a specific set of inputs.

Let’s make this concrete with an example. Imagine we are predicting weight (y) from height (x). Our linear regression model representation forthis problem would be:

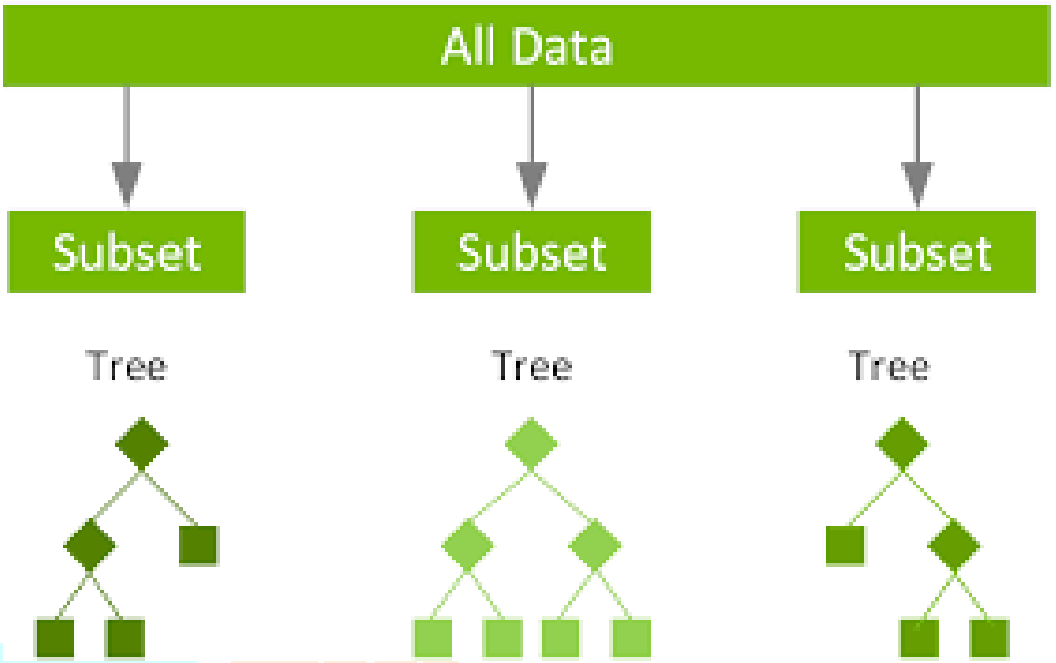
y = B0 + B1 * x1or

weight =B0 +B1 * height

Where B0 is the bias coefficient and B1 is the coefficient for the height column. We use a learning technique to find a good set of coefficient values. Once found, we can plug in different height values to predict the weight.

XGB Regression

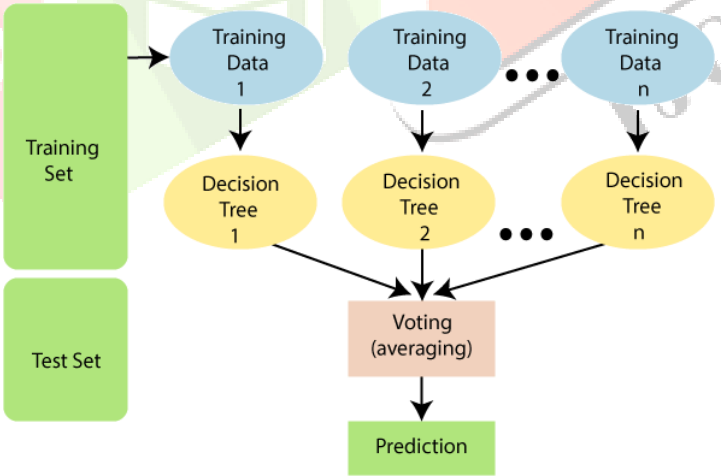
XGBoost (eXtreme Gradient Boosting) is a well-known and robust machine learning algorithm often used for supervised learning tasks such as classification, regression, and ranking. It is based on gradient-boosting architecture and has gained popularity because of its high accuracy and scalability.



Random Forest Classifier

Random Forest Regression is a supervised learning algorithm that uses ensemble learning method for regression. Ensemble learning method is a technique that combines predictions from multiple machine learning algorithms to make a more accurate prediction than a single model.

Loading the Random forest Regressor Modelrf = RandomForestRegressor() rf.fit(X_train,Y_train)



Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase.

The Working process can be explained in the below steps and diagram:

Step-1: Select random K data points from the training set.

Step-2: Build the decision trees associated with the selected data points (Subsets).

Step-3: Choose the number N for decision trees that you want to build.

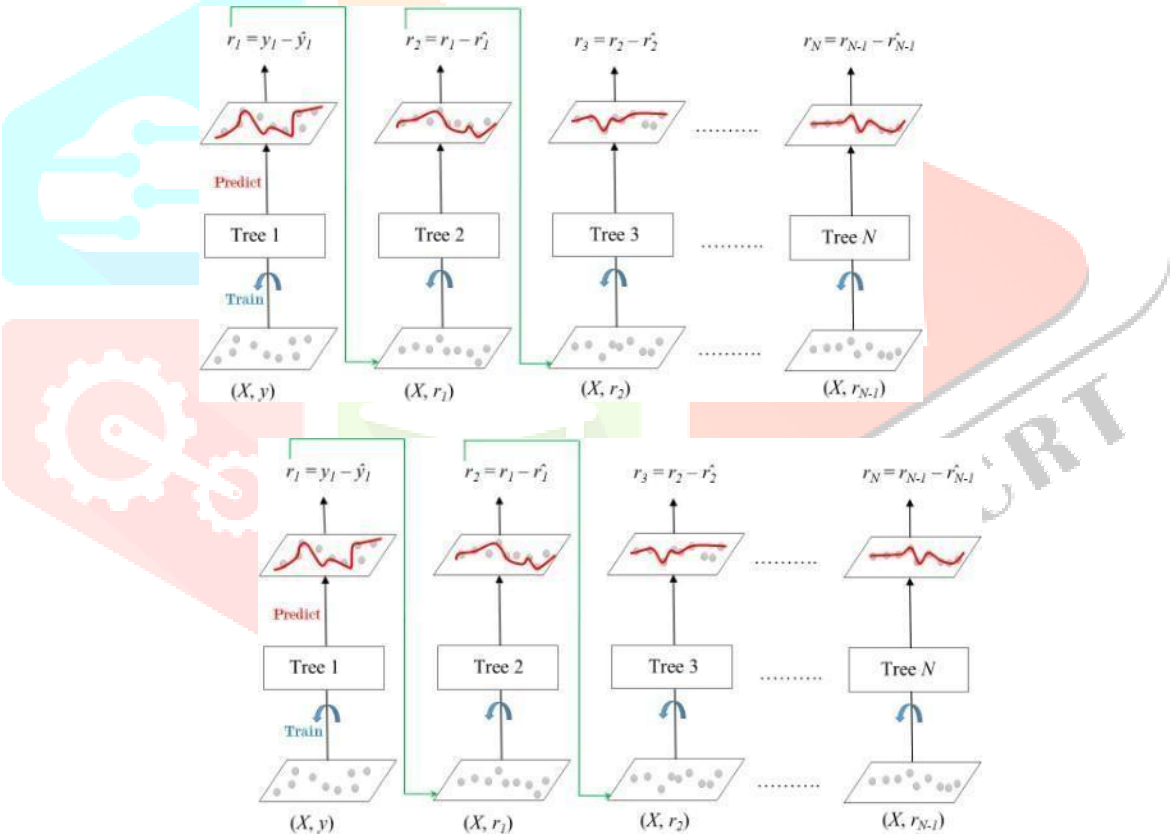
Step-4: Repeat Step 1 & 2.

Step-5: For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

Gradient Boosting Regression

Gradient boosting constructs additive regression models by sequentially fitting a simple parameterized function (base learner) to current "pseudo"-residuals by least squares at each iteration. The pseudo-residuals are the gradient of the loss functional being minimized, with respect to the model values at each training data point evaluated at the current step. It is shown that both the approximation accuracy and execution speed of gradient boosting can be substantially improved by incorporating randomization into the procedure. Specifically, at each iteration a subsample of the training data is drawn at random (without replacement) from the full training data set. This randomly selected subsample is then used in place of the full sample to fit the base learner and compute the model update for the current iteration. This randomized approach also increases robustness against overcapacity of the base learner.

# Loading the Gradient Boosting Regression Modelgr = GradientBoostingRegressor()
gr.fit(X_train, Y_train)
[37] : GradientBoostingRegressor()



How it works:

Let's say the output model y when fit to only 1 decision tree, is given by:y=A1+(B1∗X)+e1

where, e_1 is the residual from this decision tree.

In gradient boosting, we fit the consecutive decision trees on the residual from the last one. So when gradient boosting is applied to this model, the consecutive decision trees will be mathematically represented as:
e1=A2+(B2∗X)+e2

and e2=A3+(B3∗X)+e3

Note that here we stop at 3 decision trees, but in an actual gradient boosting model, the number of learners or decision trees is much more. Combining all three equations, the final model of the decision tree will be given

by:

$y=A1+A2+A3+(B1*x)+(B2*x)+(B3*x)+e3$

R Square value calculation:

- R-squared is a statistical method that determines the goodness of fit.
- It measures the strength of the relationship between the dependent and independent variables on a scale of 0–100%.
- The high value of R-square determines the less difference between the predicted values and actual values and hence represents a good model.
- It is also called a **coefficient of determination,** or **coefficient of multipledetermination** for multiple regression.
- It can be calculated from the below formula:

$$R2 = \frac{N\sum xy - \sum x\sum y}{\sqrt{[N\sum x^2 - (\sum x)^2][N\sum y^2 - (\sum y)^2]}}$$

- R = The Correlation coefficient
- n = number in the given dataset
- x = first variable in the context
- y = second variable

Mean Absolute Error or MAE

We know that an error basically is the absolute difference between the actual or true values and the values that are predicted. Absolute difference means that if the resulthas a negative sign, it is ignored.

Hence, **MAE = True values – Predicted values**

MAE takes the **average** of this error from every sample in a dataset and gives theoutput.

$$MAE = \frac{1}{N}\sum_i^N |y_{i,pred} - y_{i,true}|$$

(RESULT)

We trained 4 models using the 4 algorithms viz.

1. Linear Regression Classifier
2. XGB Regressor
3. Random Forest Classifier
4. Gradient Boosting Regressor Comparison against R squared value:

Models which gives maximum R squared value is best model . Here XGB Regressorgives max value.

The 4 models had different accuracy. The comparison of the accuracies of the models aregiven below.

models	R squared value
Linear Regression Classifier	0.687529
XGB Regressor	0.888747
Random Forest Classifier	0.760966
Gradient Boosting Regressor	0.885538

Therefore, from the above evaluation we conclude that XGB Regression model is best for the insurance datasets.

TestDataset

We were given a test dataset for this Car Price Prediction problem. We pre-process the given test dataset in similar way we pre-processed ourtrain dataset. The methodology followed is given below:

- Change the categorical values to numeric values using same process we used during changing values in our train dataset.
- Checking for null values.
- If null values are present, we will fill them or drop the rowcontaining the null value based on the dataset.

GUI Interface for Joblib

Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is the most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter is the fastest and easiest way to create the GUI applications. Creating a GUI using tkinteris an easy task.

To create a tkinter app:

1.Importing the module – tkinter
2.Create the main window (container)
3.Add any number of widgets to the main window
4.Apply the event Trigger on the widgets.

Importing tkinter is same as importing any other module in the Python code. Note that the name of the module in Python 2.x is 'Tkinter' and in Python 3.x it is 'tkinter'.

There are two main methods used which the user needs to remember while creating the Python application with GUI.

1. **Tk(screenName=None, baseName=None, className='Tk', useTk= 1):** The basic code used to create the main window of the application is:m=tkinter.Tk() where m is the name of the main window object

2. **mainloop():** There is a method known by the name mainloop() is used when your application is ready to run. mainloop() is an infinite loop used to run the application, wait for an event to occur and process the event as long as the window is not closed.

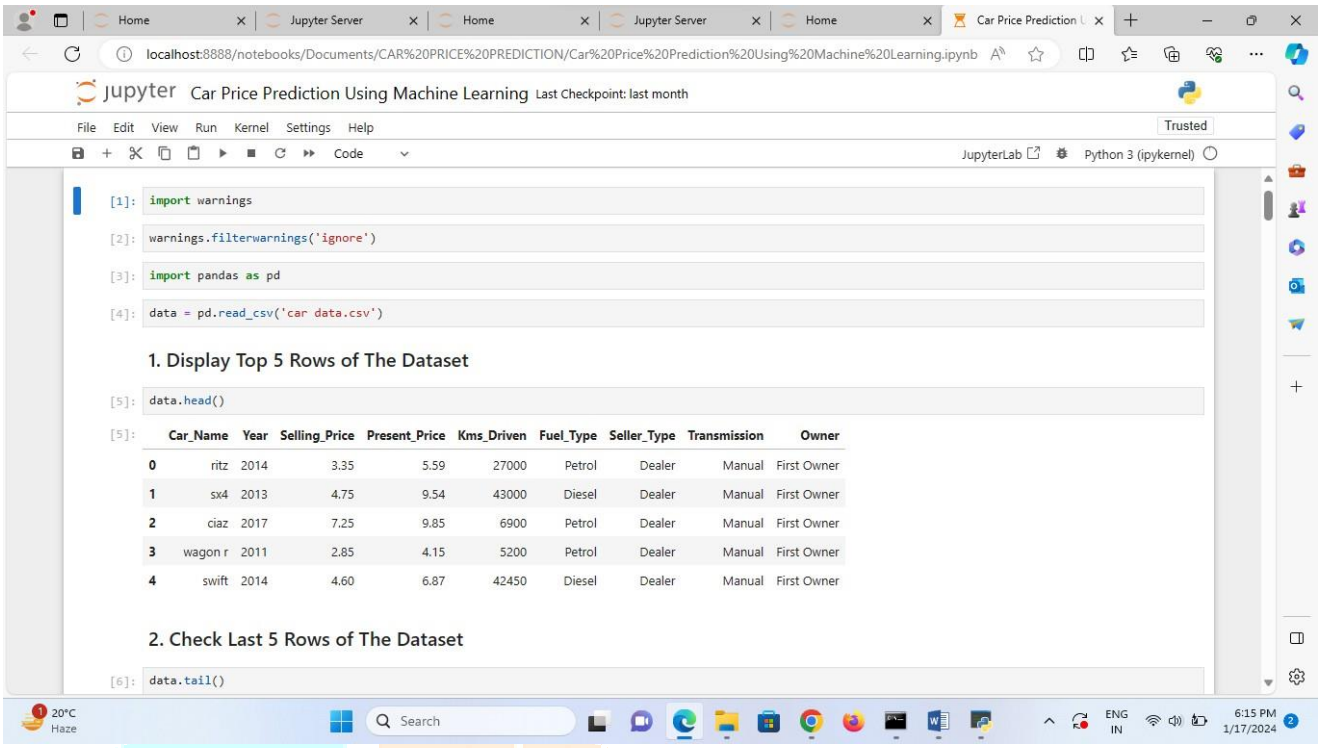
m.mainloop()

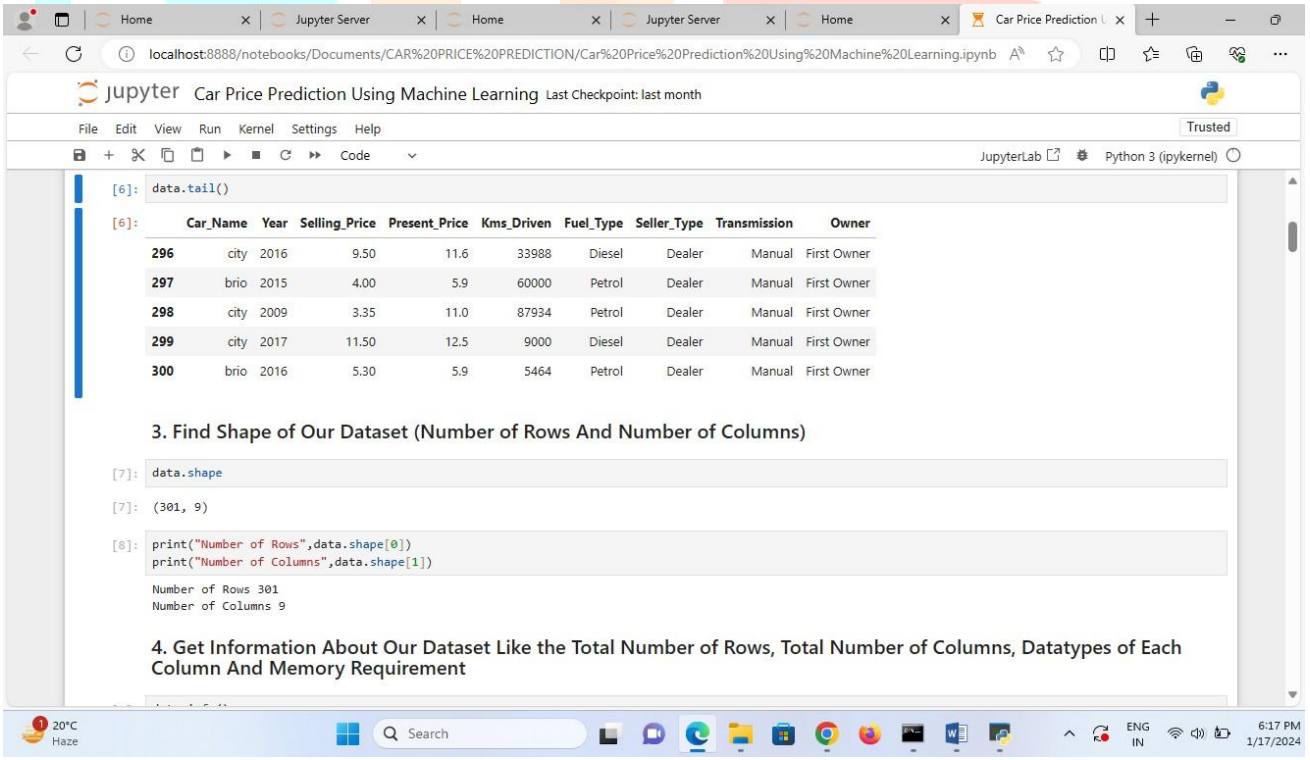
There are a number of widgets which you can put in your tkinter application. Some of the major widgets are explained below:

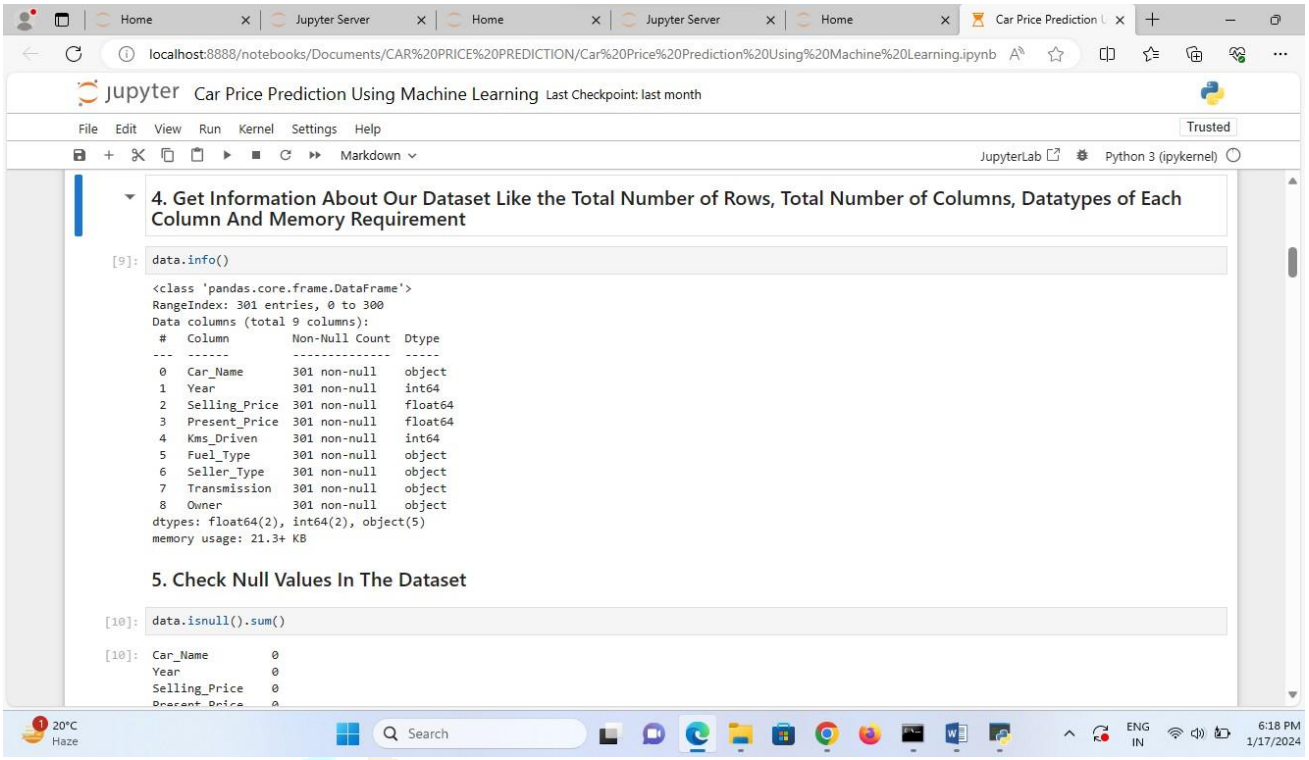
Button:To add a button in your application, this widget is used. Thegeneral syntax is:

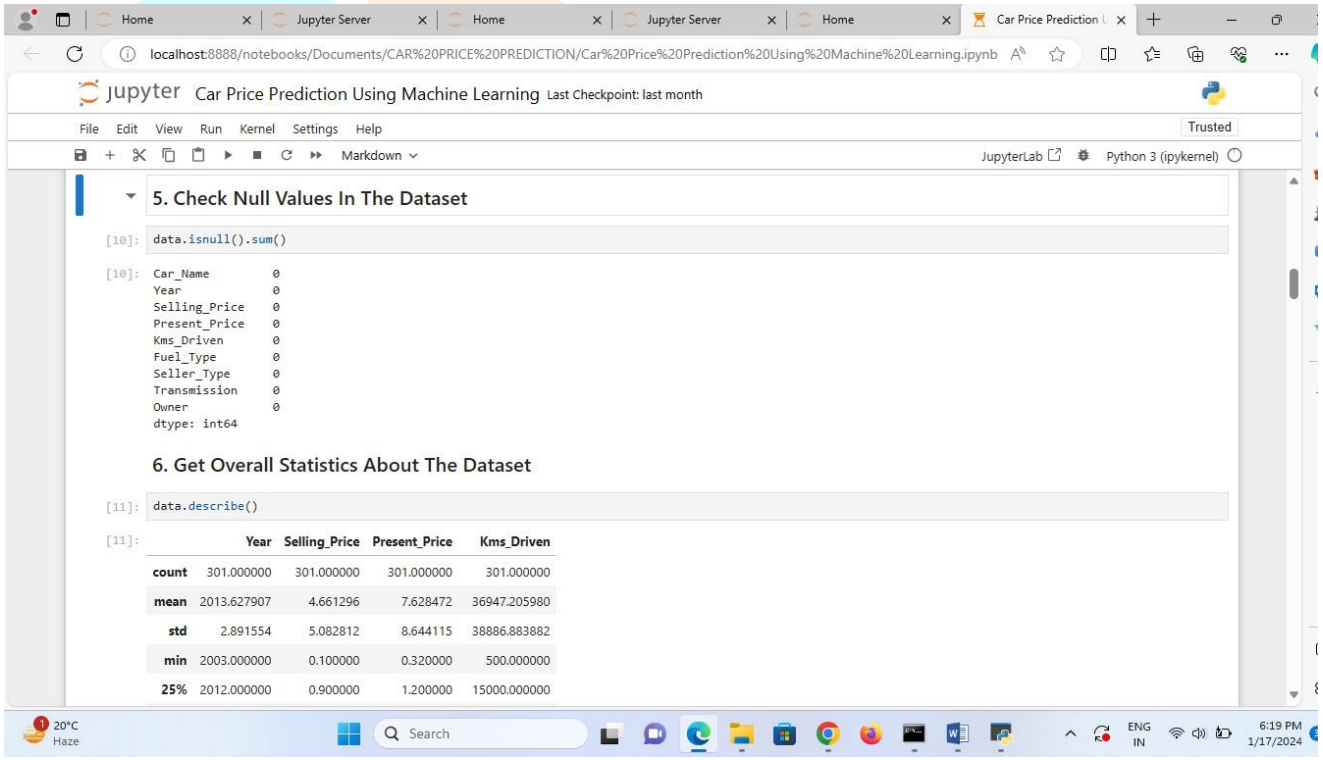
w=Button(master, option=value)

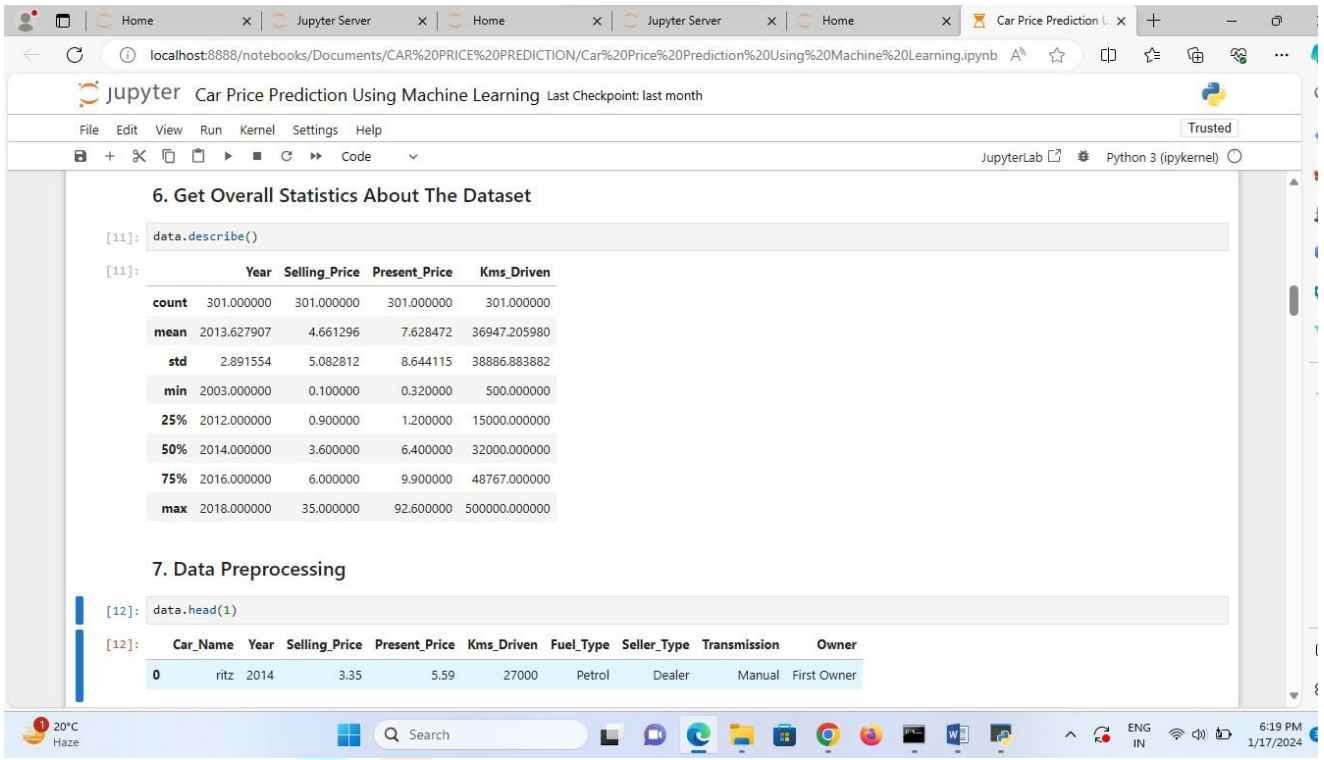
Codes


Car Price Prediction Using Machine Learning
Last Checkpoint: last month
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
[1]: import warnings
[2]: warnings.filterwarnings('ignore')
[3]: import pandas as pd
[4]: data = pd.read_csv('car data.csv')
1. Display Top 5 Rows of The Dataset
[5]: data.head()
[5]:
Car_Name Year Selling_Price Present_Price Kms_Driven Fuel_Type Seller_Type Transmission Owner
0 ritz 2014 3.35 5.59 27000 Petrol Dealer Manual First Owner
1 sx4 2013 4.75 9.54 43000 Diesel Dealer Manual First Owner
2 ciaz 2017 7.25 9.85 6900 Petrol Dealer Manual First Owner
3 wagon r 2011 2.85 4.15 5200 Petrol Dealer Manual First Owner
4 swift 2014 4.60 6.87 42450 Diesel Dealer Manual First Owner
2. Check Last 5 Rows of The Dataset
[6]: data.tail()


Car Price Prediction Using Machine Learning
Last Checkpoint: last month
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
[6]: data.tail()
[6]:
Car_Name Year Selling_Price Present_Price Kms_Driven Fuel_Type Seller_Type Transmission Owner
296 city 2016 9.50 11.6 33988 Diesel Dealer Manual First Owner
297 brio 2015 4.00 5.9 60000 Petrol Dealer Manual First Owner
298 city 2009 3.35 11.0 87934 Petrol Dealer Manual First Owner
299 city 2017 11.50 12.5 9000 Diesel Dealer Manual First Owner
300 brio 2016 5.30 5.9 5464 Petrol Dealer Manual First Owner
3. Find Shape of Our Dataset (Number of Rows And Number of Columns)
[7]: data.shape
[7]: (301, 9)
[8]: print("Number of Rows",data.shape[0])
print("Number of Columns",data.shape[1])
Number of Rows 301
Number of Columns 9
4. Get Information About Our Dataset Like the Total Number of Rows, Total Number of Columns, Datatypes of Each Column And Memory Requirement


Home
Jupyter Server
Home
Jupyter Server
Home
Car Price Prediction
localhost:8888/notebooks/Documents/CAR%20PRICE%20PREDICTION/Car%20Price%20Prediction%20Using%20Machine%20Learning.ipynb
jupyter Car Price Prediction Using Machine Learning Last Checkpoint: last month
File Edit View Run Kernel Settings Help
Trusted
Markdown
JupyterLab
Python 3 (ipykernel)
4. Get Information About Our Dataset Like the Total Number of Rows, Total Number of Columns, Datatypes of Each Column And Memory Requirement
[9]: data.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 301 entries, 0 to 300
Data columns (total 9 columns):
Column Non-Null Count Dtype
--- ------ -------------- -----
0 Car_Name 301 non-null object
1 Year 301 non-null int64
2 Selling_Price 301 non-null float64
3 Present_Price 301 non-null float64
4 Kms_Driven 301 non-null int64
5 Fuel_Type 301 non-null object
6 Seller_Type 301 non-null object
7 Transmission 301 non-null object
8 Owner 301 non-null object
dtypes: float64(2), int64(2), object(5)
memory usage: 21.3+ KB
5. Check Null Values In The Dataset
[10]: data.isnull().sum()
[10]: Car_Name 0
Year 0
Selling_Price 0
20°C Haze
Search
ENG IN
6:18 PM 1/17/2024


Home
Jupyter Server
Home
Jupyter Server
Home
Car Price Prediction
localhost:8888/notebooks/Documents/CAR%20PRICE%20PREDICTION/Car%20Price%20Prediction%20Using%20Machine%20Learning.ipynb
jupyter Car Price Prediction Using Machine Learning Last Checkpoint: last month
File Edit View Run Kernel Settings Help
Trusted
Markdown
JupyterLab
Python 3 (ipykernel)
5. Check Null Values In The Dataset
[10]: data.isnull().sum()
[10]: Car_Name 0
Year 0
Selling_Price 0
Present_Price 0
Kms_Driven 0
Fuel_Type 0
Seller_Type 0
Transmission 0
Owner 0
dtype: int64
6. Get Overall Statistics About The Dataset
[11]: data.describe()
[11]:
Year Selling_Price Present_Price Kms_Driven
count 301.000000 301.000000 301.000000 301.000000
mean 2013.627907 4.661296 7.628472 36947.205980
std 2.891554 5.082812 8.644115 38886.883882
min 2003.000000 0.100000 0.320000 500.000000
25% 2012.000000 0.900000 1.200000 15000.000000
20°C Haze
Search
ENG IN
6:19 PM 1/17/2024



Jupyter Car Price Prediction Using Machine Learning Last Checkpoint: last month

6. Get Overall Statistics About The Dataset

```
[11]: data.describe()
```

[11]:

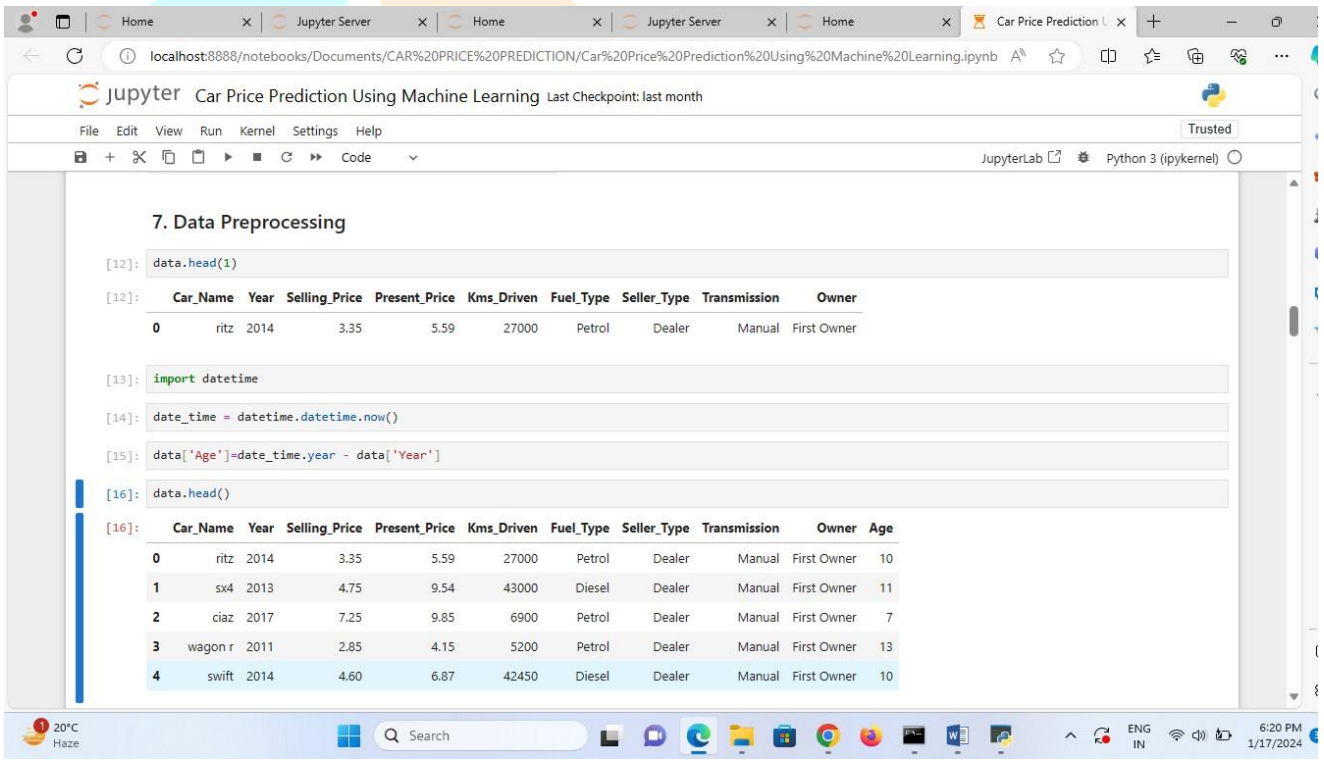
	Year	Selling_Price	Present_Price	Kms_Driven
count	301.000000	301.000000	301.000000	301.000000
mean	2013.627907	4.661296	7.628472	36947.205980
std	2.891554	5.082812	8.644115	38886.883882
min	2003.000000	0.100000	0.320000	500.000000
25%	2012.000000	0.900000	1.200000	15000.000000
50%	2014.000000	3.600000	6.400000	32000.000000
75%	2016.000000	6.000000	9.900000	48767.000000
max	2018.000000	35.000000	92.600000	500000.000000

7. Data Preprocessing

```
[12]: data.head(1)
```

[12]:

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner
0	ritz	2014	3.35	5.59	27000	Petrol	Dealer	Manual	First Owner



Jupyter Car Price Prediction Using Machine Learning Last Checkpoint: last month

7. Data Preprocessing

```
[12]: data.head(1)
```

[12]:

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner
0	ritz	2014	3.35	5.59	27000	Petrol	Dealer	Manual	First Owner

```
[13]: import datetime
```

```
[14]: date_time = datetime.datetime.now()
```

```
[15]: data['Age']=date_time.year - data['Year']
```

```
[16]: data.head()
```

[16]:

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner	Age
0	ritz	2014	3.35	5.59	27000	Petrol	Dealer	Manual	First Owner	10
1	sx4	2013	4.75	9.54	43000	Diesel	Dealer	Manual	First Owner	11
2	ciaz	2017	7.25	9.85	6900	Petrol	Dealer	Manual	First Owner	7
3	wagon r	2011	2.85	4.15	5200	Petrol	Dealer	Manual	First Owner	13
4	swift	2014	4.60	6.87	42450	Diesel	Dealer	Manual	First Owner	10

Home | Jupyter Server | Home | Jupyter Server | Home | Car Price Prediction

localhost:8888/notebooks/Documents/CAR%20PRICE%20PREDICTION/Car%20Price%20Prediction%20Using%20Machine%20Learning.ipynb

jupyter Car Price Prediction Using Machine Learning Last Checkpoint: last month

File Edit View Run Kernel Settings Help

Trusted

Code

JupyterLab Python 3 (ipykernel)

[16]:
```
data.head()
```

[16]:

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner	Age
0	ritz	2014	3.35	5.59	27000	Petrol	Dealer	Manual	First Owner	10
1	sx4	2013	4.75	9.54	43000	Diesel	Dealer	Manual	First Owner	11
2	ciaz	2017	7.25	9.85	6900	Petrol	Dealer	Manual	First Owner	7
3	wagon r	2011	2.85	4.15	5200	Petrol	Dealer	Manual	First Owner	13
4	swift	2014	4.60	6.87	42450	Diesel	Dealer	Manual	First Owner	10

[17]:
```
data.drop('Year',axis=1,inplace=True)
```

[18]:
```
data.head()
```

[18]:

	Car_Name	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner	Age
0	ritz	3.35	5.59	27000	Petrol	Dealer	Manual	First Owner	10
1	sx4	4.75	9.54	43000	Diesel	Dealer	Manual	First Owner	11
2	ciaz	7.25	9.85	6900	Petrol	Dealer	Manual	First Owner	7
3	wagon r	2.85	4.15	5200	Petrol	Dealer	Manual	First Owner	13
4	swift	4.60	6.87	42450	Diesel	Dealer	Manual	First Owner	10

Outlier Removal

18°C Air: Very poor | Search | ENG IN | 6:21 PM 1/17/2024

Home | Jupyter Server | Home | Jupyter Server | Home | Car Price Prediction

localhost:8888/notebooks/Documents/CAR%20PRICE%20PREDICTION/Car%20Price%20Prediction%20Using%20Machine%20Learning.ipynb

jupyter Car Price Prediction Using Machine Learning Last Checkpoint: last month

File Edit View Run Kernel Settings Help

Trusted

Code

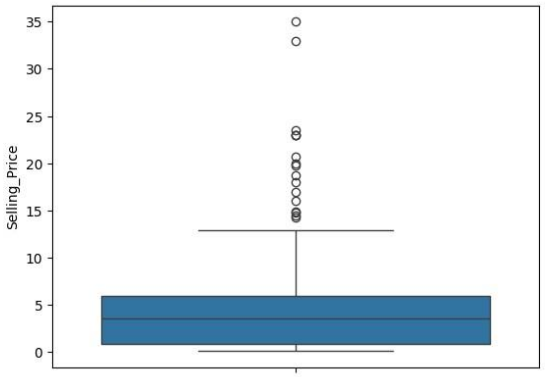
JupyterLab Python 3 (ipykernel)

Outlier Removal

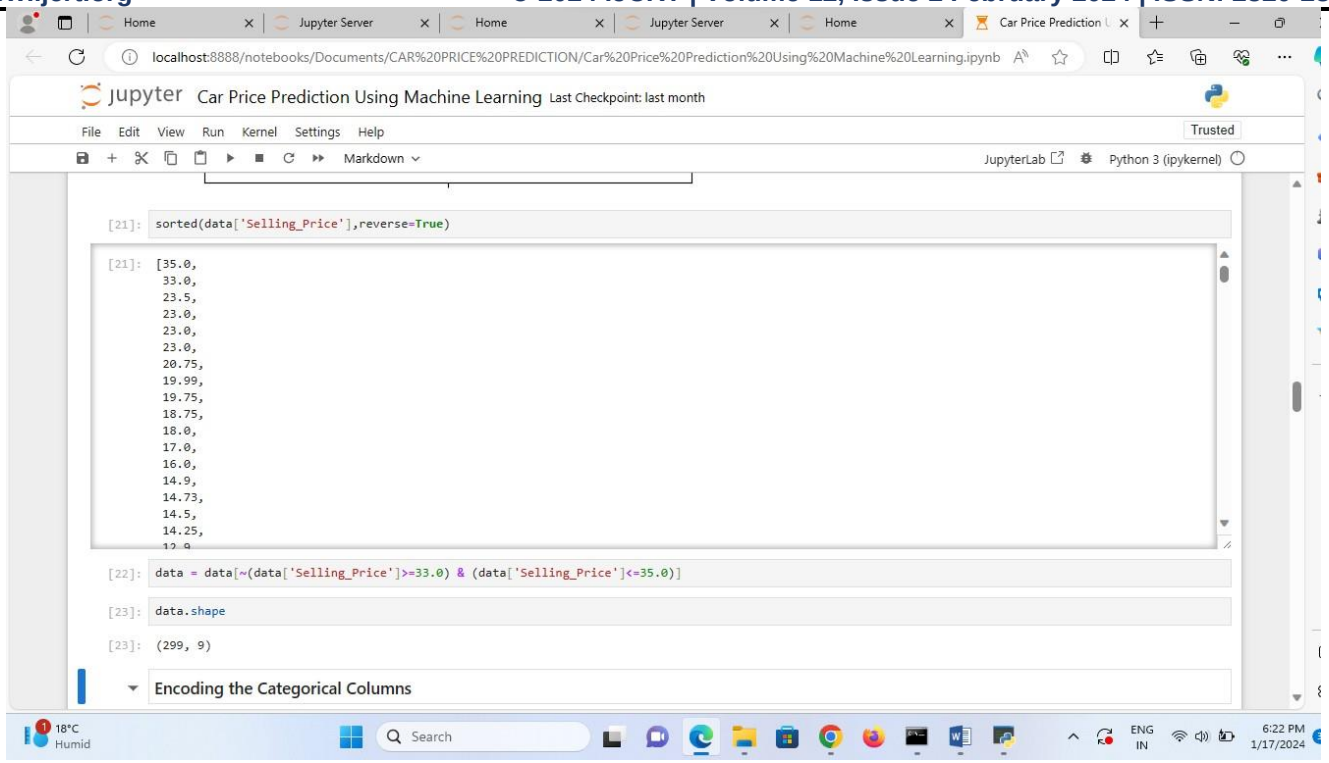
[19]:
```
import seaborn as sns
```

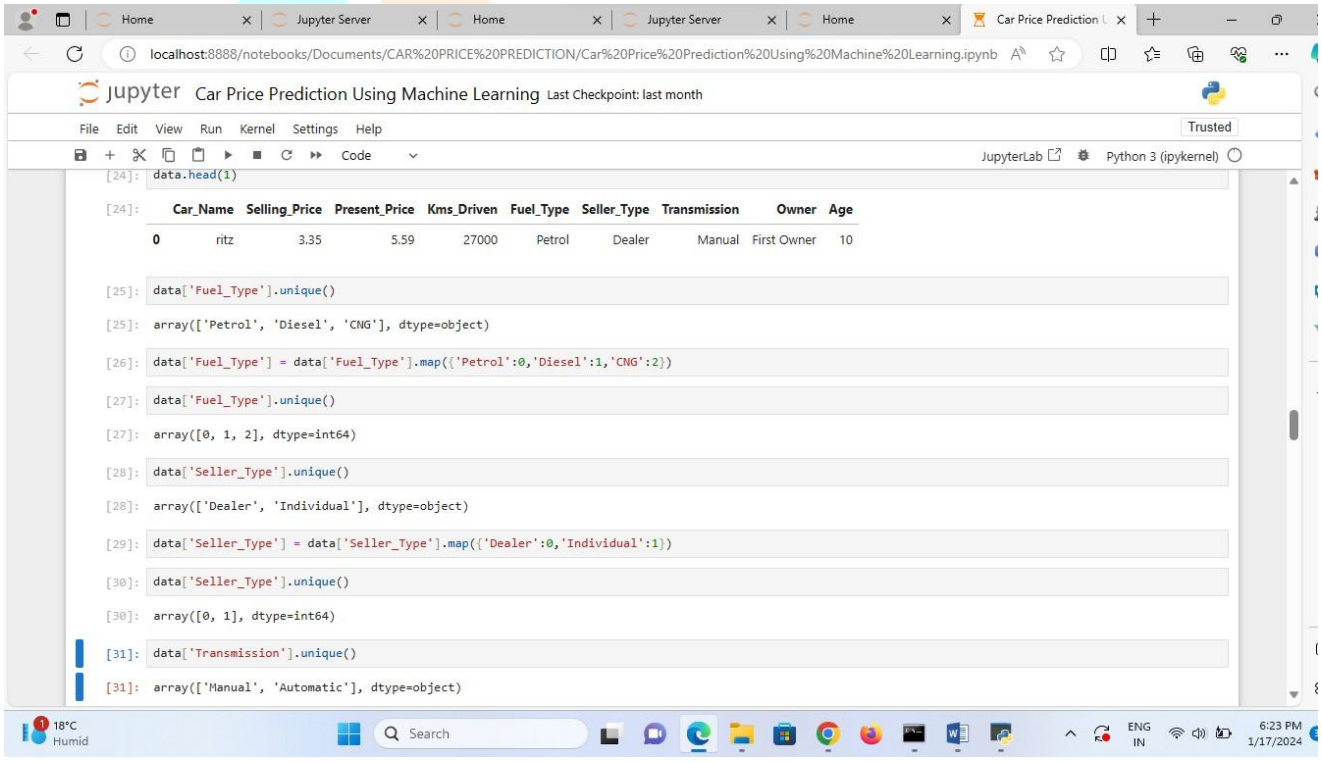
[20]:
```
sns.boxplot(data['Selling_Price'])
```

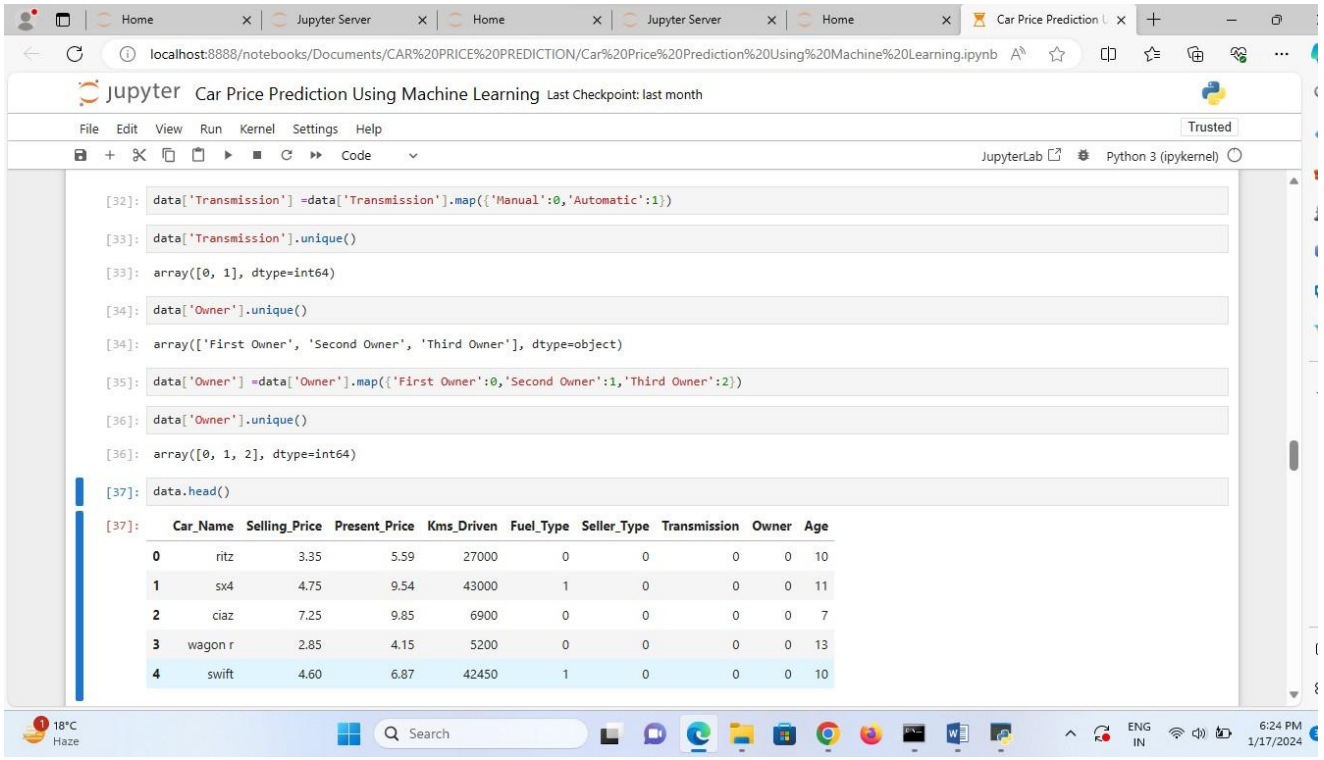
[20]:
```
<Axes: ylabel='Selling_Price'>
```



18°C Air: Very poor | Search | ENG IN | 6:22 PM 1/17/2024


Car Price Prediction Using Machine Learning
Last Checkpoint: last month
File Edit View Run Kernel Settings Help
Trusted
Markdown
JupyterLab
Python 3 (ipykernel)
[21]: sorted(data['Selling_Price'],reverse=True)
[21]: [35.0, 33.0, 23.5, 23.0, 23.0, 23.0, 20.75, 19.99, 19.75, 18.75, 18.0, 17.0, 16.0, 14.9, 14.73, 14.5, 14.25,
[22]: data = data[~(data['Selling_Price']>=33.0) & (data['Selling_Price']<=35.0)]
[23]: data.shape
[23]: (299, 9)
Encoding the Categorical Columns


Car Price Prediction Using Machine Learning
Last Checkpoint: last month
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
[24]: data.head(1)
Car_Name Selling_Price Present_Price Kms_Driven Fuel_Type Seller_Type Transmission Owner Age
0 ritz 3.35 5.59 27000 Petrol Dealer Manual First Owner 10
[25]: data['Fuel_Type'].unique()
[25]: array(['Petrol', 'Diesel', 'CNG'], dtype=object)
[26]: data['Fuel_Type'] = data['Fuel_Type'].map({'Petrol':0,'Diesel':1,'CNG':2})
[27]: data['Fuel_Type'].unique()
[27]: array([0, 1, 2], dtype=int64)
[28]: data['Seller_Type'].unique()
[28]: array(['Dealer', 'Individual'], dtype=object)
[29]: data['Seller_Type'] = data['Seller_Type'].map({'Dealer':0,'Individual':1})
[30]: data['Seller_Type'].unique()
[30]: array([0, 1], dtype=int64)
[31]: data['Transmission'].unique()
[31]: array(['Manual', 'Automatic'], dtype=object)

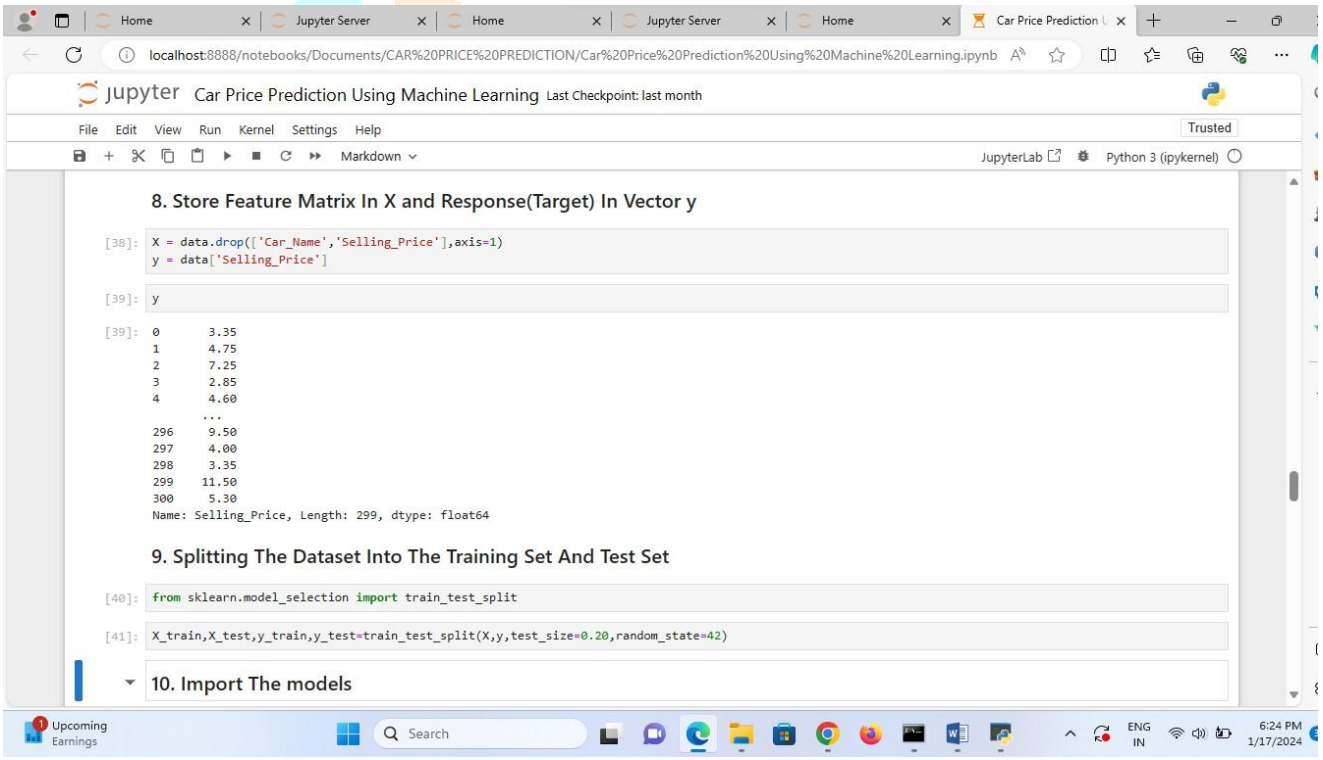


Car Price Prediction Using Machine Learning Last Checkpoint: last month

```
[32]: data['Transmission'] =data['Transmission'].map({'Manual':0,'Automatic':1})
[33]: data['Transmission'].unique()
[33]: array([0, 1], dtype=int64)
[34]: data['Owner'].unique()
[34]: array(['First Owner', 'Second Owner', 'Third Owner'], dtype=object)
[35]: data['Owner'] =data['Owner'].map({'First Owner':0,'Second Owner':1,'Third Owner':2})
[36]: data['Owner'].unique()
[36]: array([0, 1, 2], dtype=int64)
[37]: data.head()
```

[37]:

	Car_Name	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner	Age
0	ritz	3.35	5.59	27000	0	0	0	0	10
1	sx4	4.75	9.54	43000	1	0	0	0	11
2	ciaz	7.25	9.85	6900	0	0	0	0	7
3	wagon r	2.85	4.15	5200	0	0	0	0	13
4	swift	4.60	6.87	42450	1	0	0	0	10



Car Price Prediction Using Machine Learning Last Checkpoint: last month

8. Store Feature Matrix In X and Response(Target) In Vector y

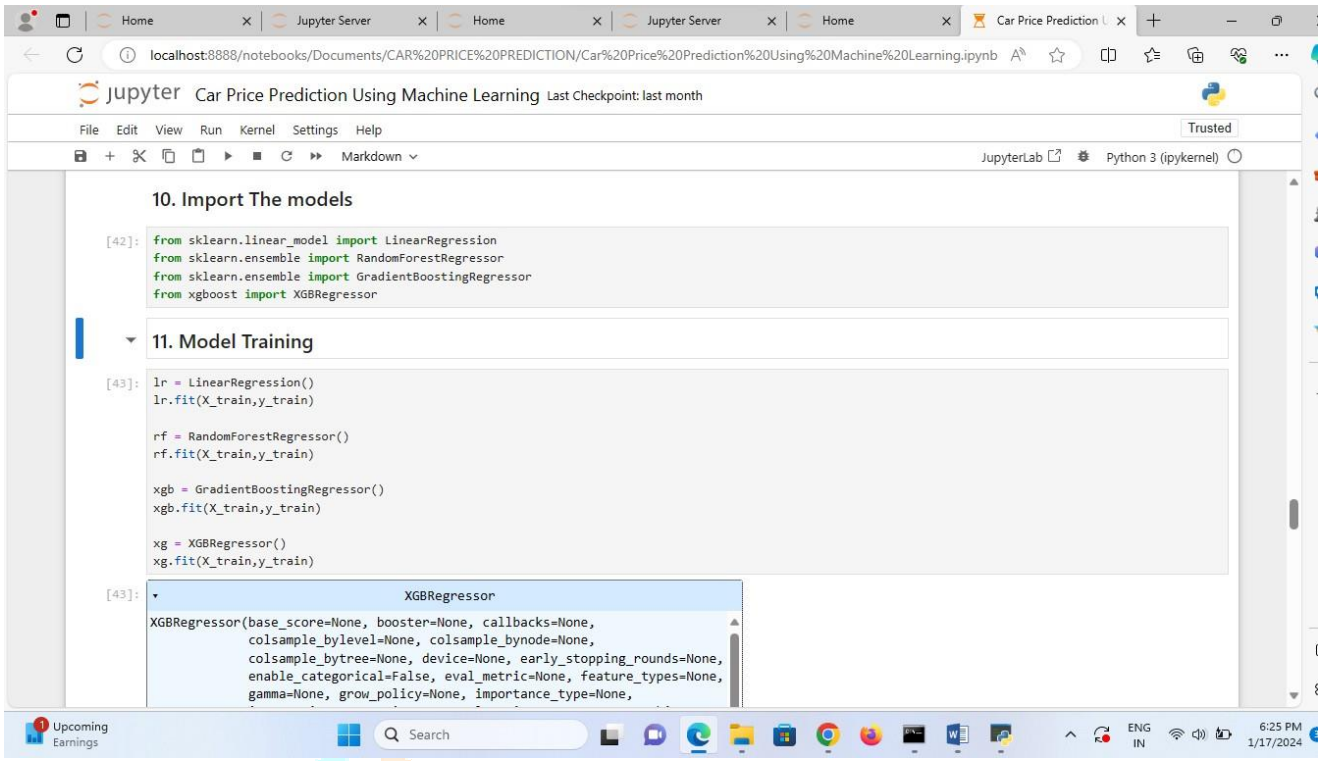
```
[38]: X = data.drop(['Car_Name','Selling_Price'],axis=1)
      y = data['Selling_Price']
[39]: y
```

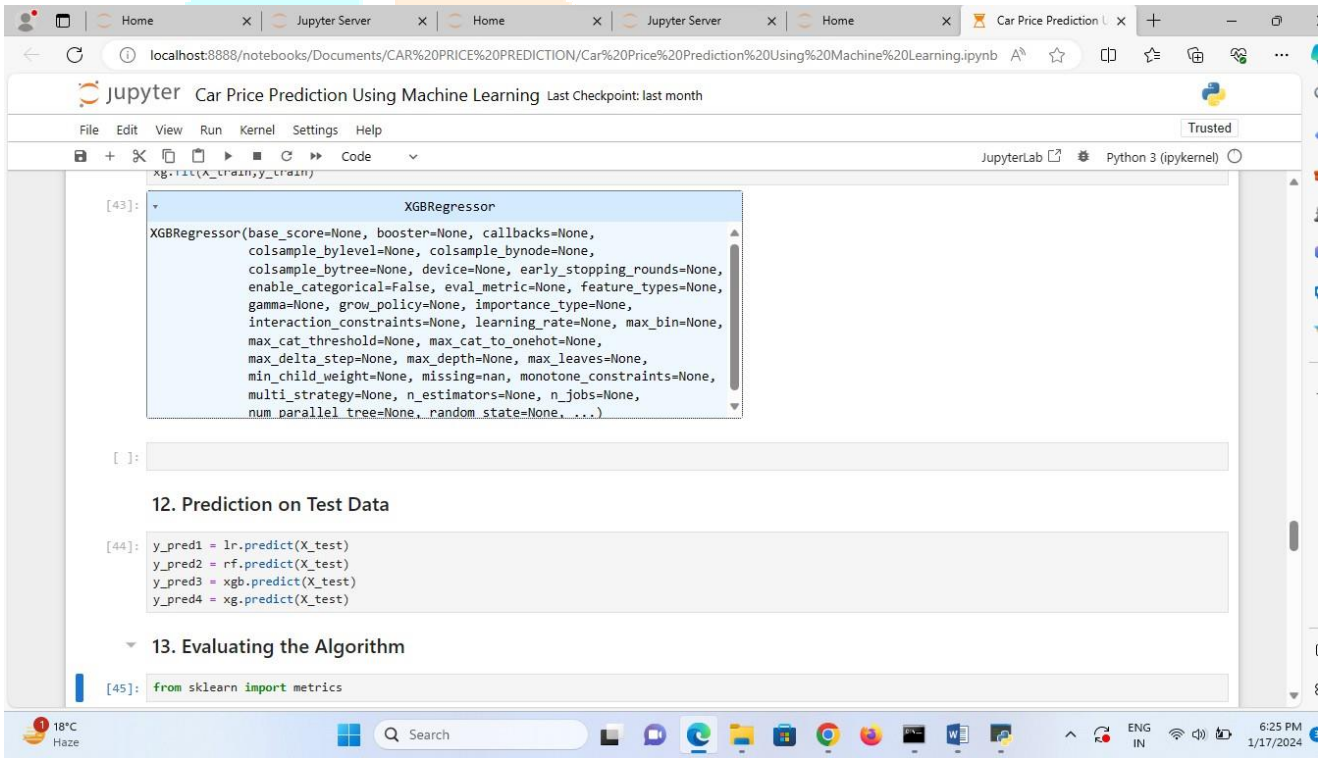
```
[39]: 0      3.35
      1      4.75
      2      7.25
      3      2.85
      4      4.60
            ...
      296    9.50
      297    4.00
      298    3.35
      299   11.50
      300    5.30
      Name: Selling_Price, Length: 299, dtype: float64
```

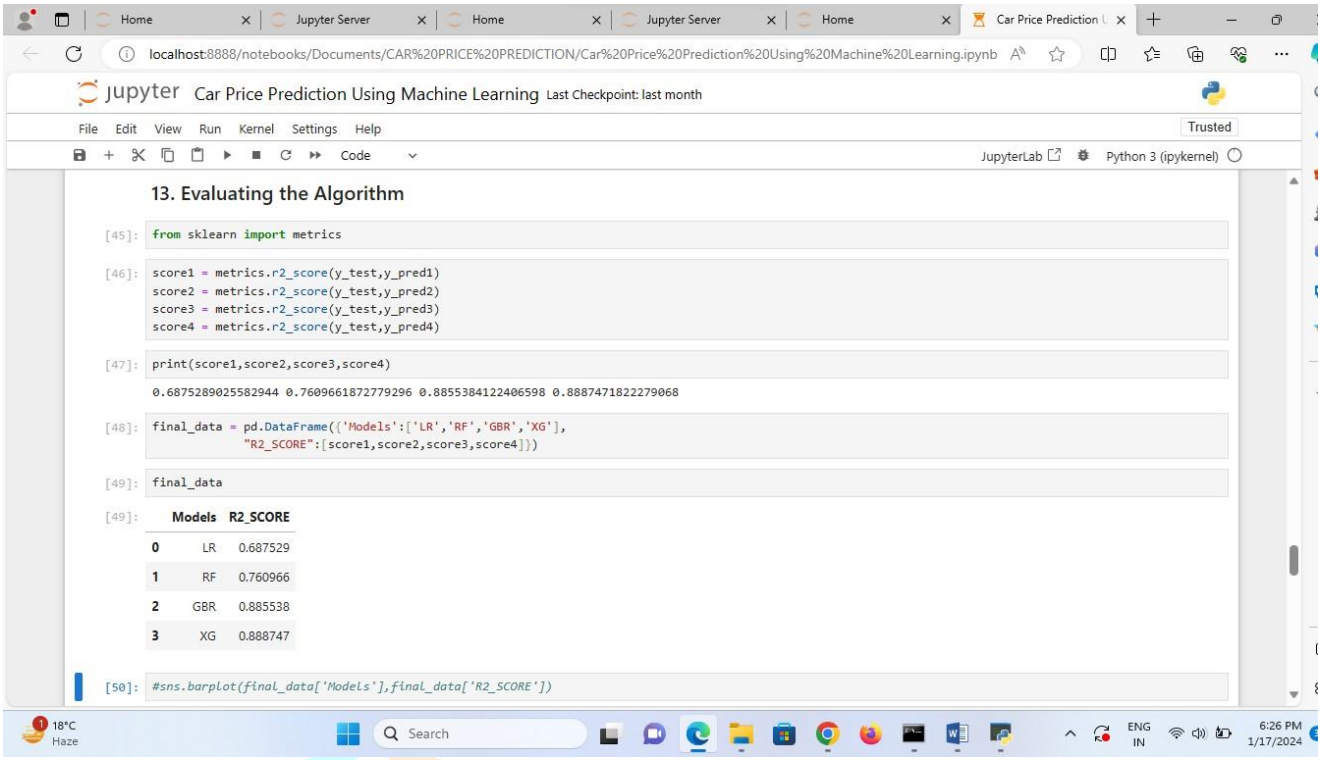
9. Splitting The Dataset Into The Training Set And Test Set

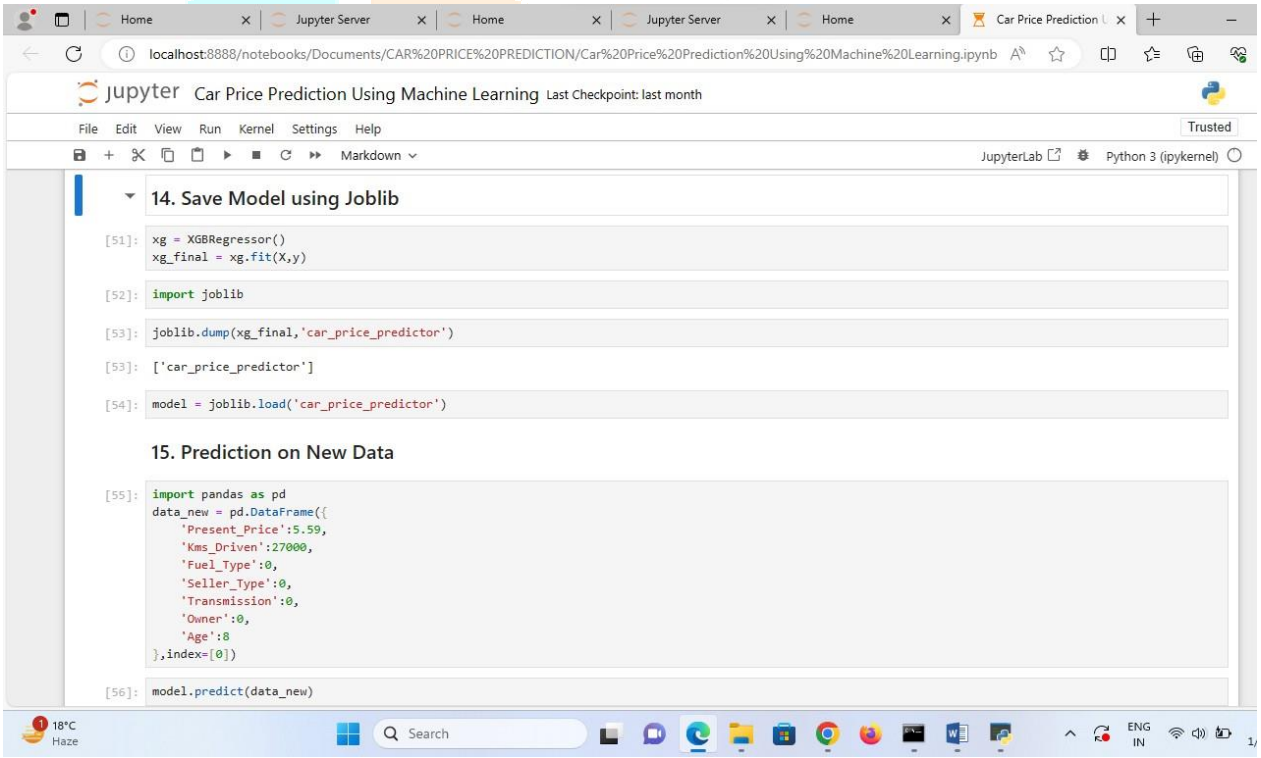
```
[40]: from sklearn.model_selection import train_test_split
[41]: X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.20,random_state=42)
```

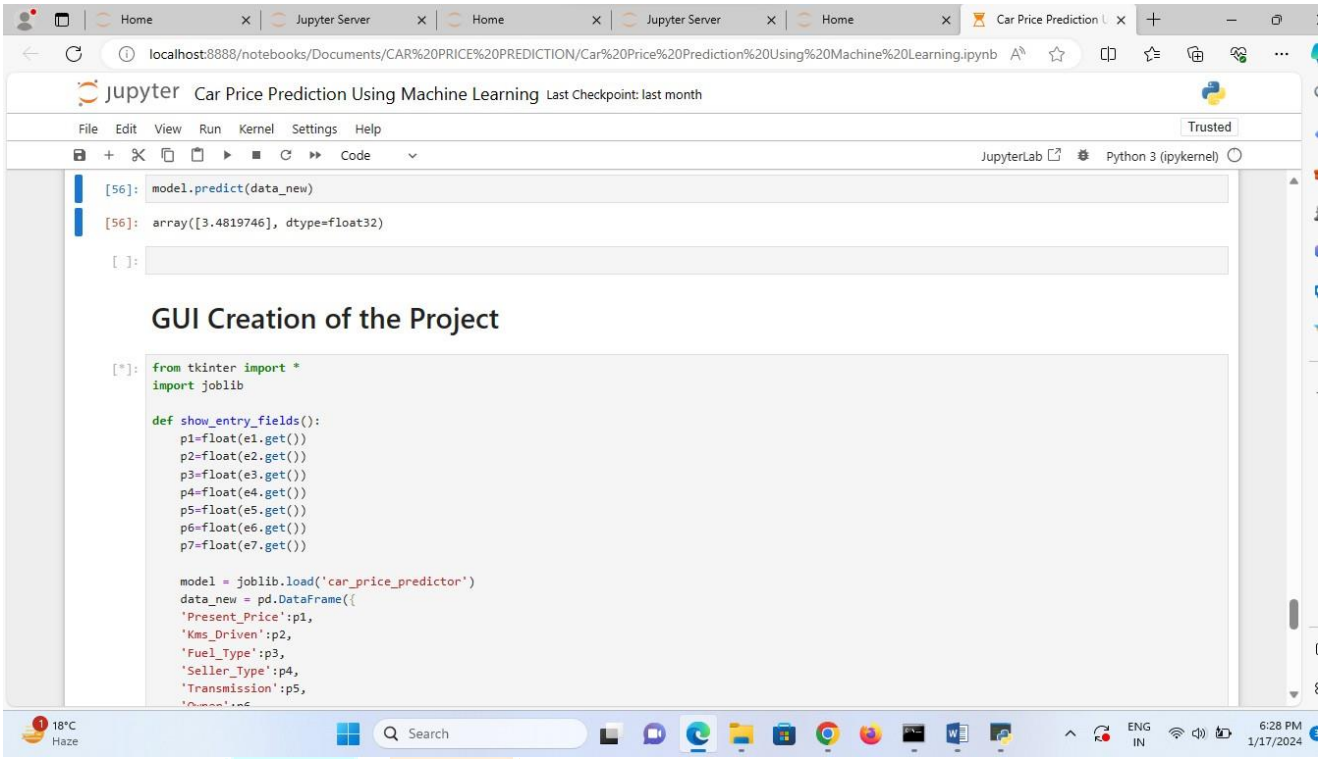
10. Import The models

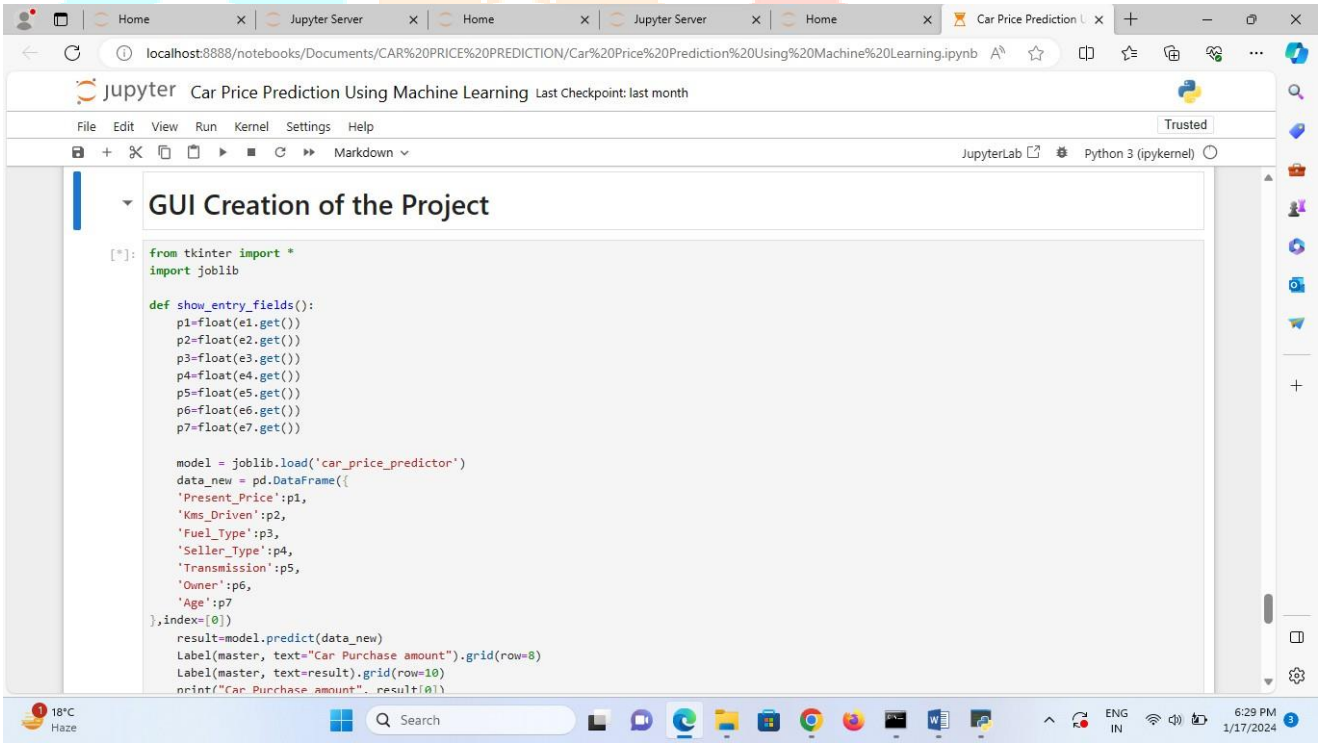

Home
Jupyter Server
Home
Jupyter Server
Home
Car Price Prediction
localhost:8888/notebooks/Documents/CAR%20PRICE%20PREDICTION/Car%20Price%20Prediction%20Using%20Machine%20Learning.ipynb
jupyter Car Price Prediction Using Machine Learning Last Checkpoint: last month
File Edit View Run Kernel Settings Help
Trusted
Markdown
JupyterLab
Python 3 (ipykernel)
10. Import The models
[42]:
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import GradientBoostingRegressor
from xgboost import XGBRegressor
11. Model Training
[43]:
lr = LinearRegression()
lr.fit(X_train,y_train)
rf = RandomForestRegressor()
rf.fit(X_train,y_train)
xgb = GradientBoostingRegressor()
xgb.fit(X_train,y_train)
xg = XGBRegressor()
xg.fit(X_train,y_train)
[43]:
XGBRegressor
XGBRegressor(base_score=None, booster=None, callbacks=None,
colsample_bylevel=None, colsample_bynode=None,
colsample_bytree=None, device=None, early_stopping_rounds=None,
enable_categorical=False, eval_metric=None, feature_types=None,
gamma=None, grow_policy=None, importance_type=None,
Upcoming Earnings
Search
ENG IN
6:25 PM 1/17/2024

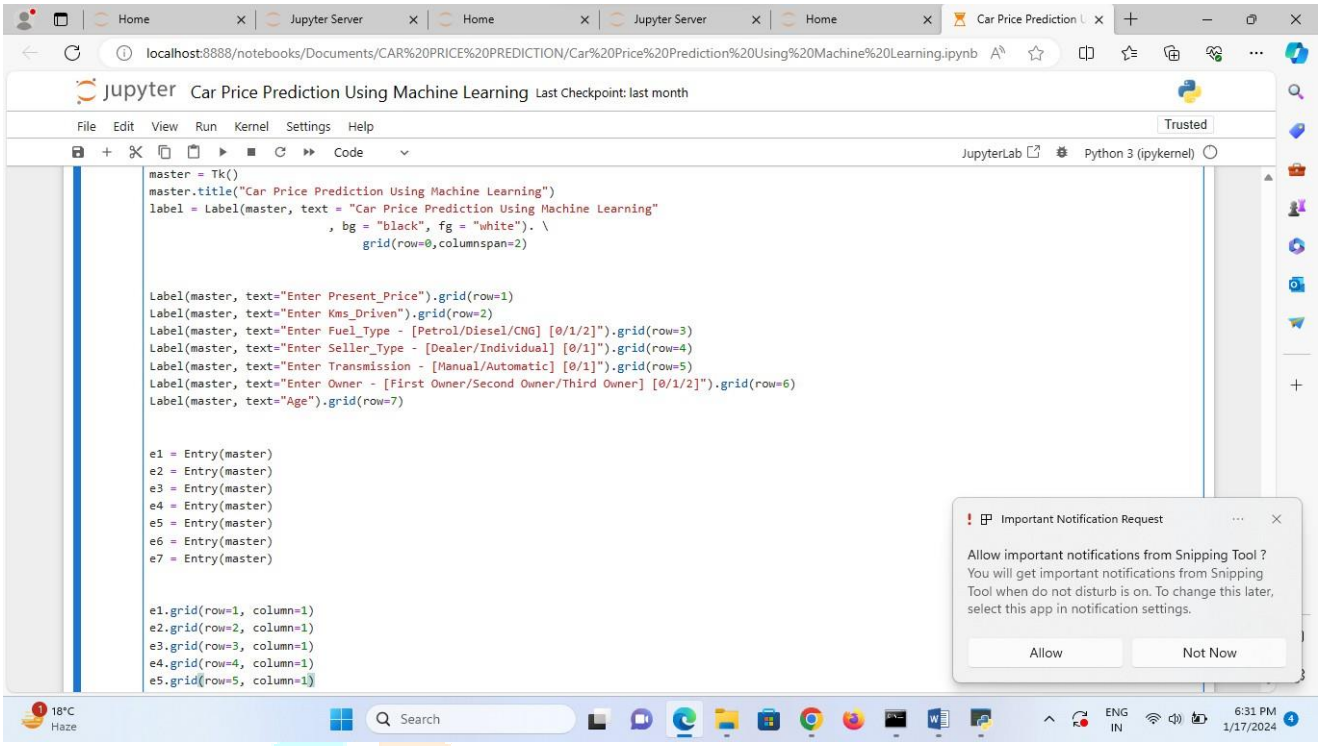

Home
Jupyter Server
Home
Jupyter Server
Home
Car Price Prediction
localhost:8888/notebooks/Documents/CAR%20PRICE%20PREDICTION/Car%20Price%20Prediction%20Using%20Machine%20Learning.ipynb
jupyter Car Price Prediction Using Machine Learning Last Checkpoint: last month
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
[43]:
XGBRegressor
XGBRegressor(base_score=None, booster=None, callbacks=None,
colsample_bylevel=None, colsample_bynode=None,
colsample_bytree=None, device=None, early_stopping_rounds=None,
enable_categorical=False, eval_metric=None, feature_types=None,
gamma=None, grow_policy=None, importance_type=None,
interaction_constraints=None, learning_rate=None, max_bin=None,
max_cat_threshold=None, max_cat_to_onehot=None,
max_delta_step=None, max_depth=None, max_leaves=None,
min_child_weight=None, missing=nan, monotone_constraints=None,
multi_strategy=None, n_estimators=None, n_jobs=None,
num_parallel_tree=None, random_state=None, ...)
[ ]:
12. Prediction on Test Data
[44]:
y_pred1 = lr.predict(X_test)
y_pred2 = rf.predict(X_test)
y_pred3 = xgb.predict(X_test)
y_pred4 = xg.predict(X_test)
13. Evaluating the Algorithm
[45]:
from sklearn import metrics
18°C Haze
Search
ENG IN
6:25 PM 1/17/2024

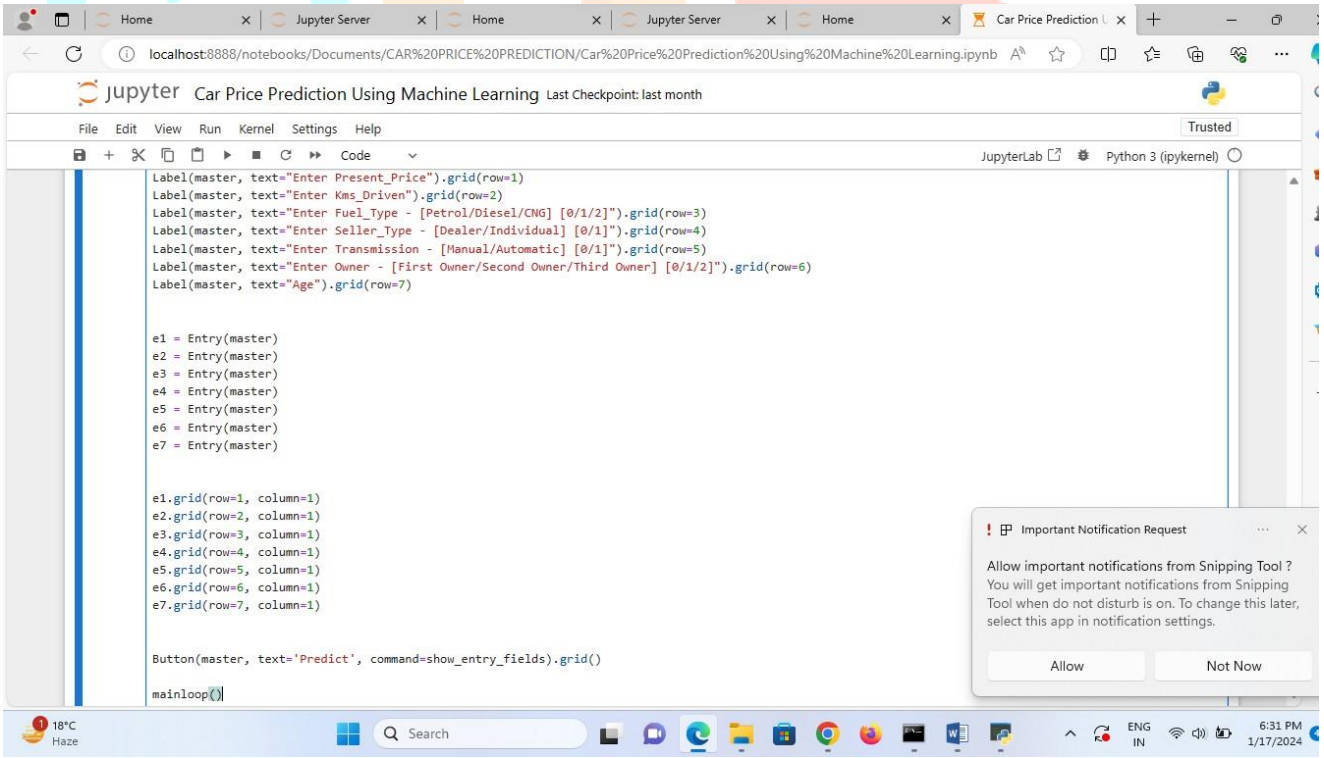

Car Price Prediction Using Machine Learning Last Checkpoint: last month
13. Evaluating the Algorithm
from sklearn import metrics
score1 = metrics.r2_score(y_test,y_pred1)
score2 = metrics.r2_score(y_test,y_pred2)
score3 = metrics.r2_score(y_test,y_pred3)
score4 = metrics.r2_score(y_test,y_pred4)
print(score1,score2,score3,score4)
0.6875289025582944 0.7609661872779296 0.8855384122406598 0.8887471822279068
final_data = pd.DataFrame({'Models':['LR','RF','GBR','XG'],
"R2_SCORE":[score1,score2,score3,score4]})
final_data
Models R2_SCORE
0 LR 0.687529
1 RF 0.760966
2 GBR 0.885538
3 XG 0.888747
#sns.barplot(final_data['Models'],final_data['R2_SCORE'])


Car Price Prediction Using Machine Learning Last Checkpoint: last month
14. Save Model using Joblib
xg = XGBRegressor()
xg_final = xg.fit(X,y)
import joblib
joblib.dump(xg_final,'car_price_predictor')
['car_price_predictor']
model = joblib.load('car_price_predictor')
15. Prediction on New Data
import pandas as pd
data_new = pd.DataFrame({
'Present_Price':5.59,
'Kms_Driven':27000,
'Fuel_Type':0,
'Seller_Type':0,
'Transmission':0,
'Owner':0,
'Age':8
},index=[0])
model.predict(data_new)

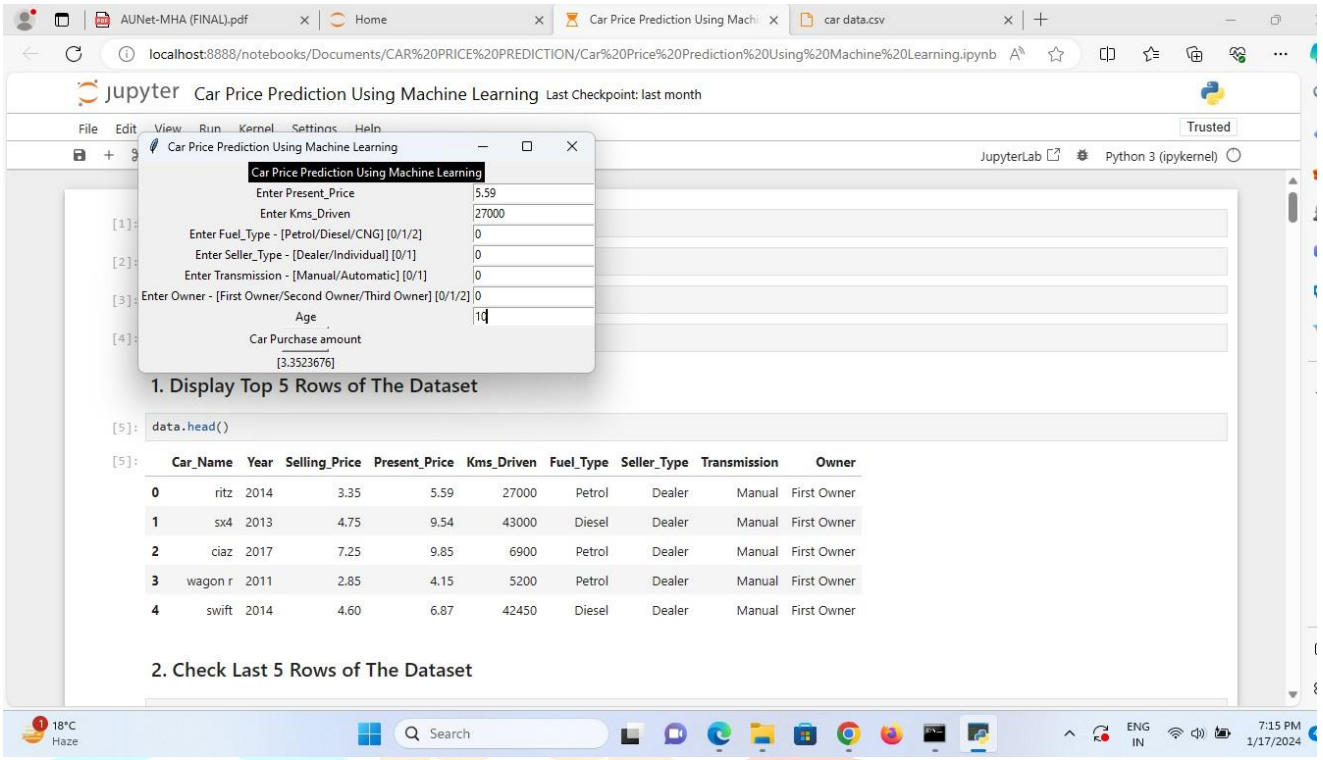

Car Price Prediction Using Machine Learning
Last Checkpoint: last month
Trusted
Python 3 (ipykernel)
[56]: model.predict(data_new)
[56]: array([3.4819746], dtype=float32)
GUI Creation of the Project
from tkinter import *
import joblib
def show_entry_fields():
p1=float(e1.get())
p2=float(e2.get())
p3=float(e3.get())
p4=float(e4.get())
p5=float(e5.get())
p6=float(e6.get())
p7=float(e7.get())
model = joblib.load('car_price_predictor')
data_new = pd.DataFrame({
'Present_Price':p1,
'Kms_Driven':p2,
'Fuel_Type':p3,
'Seller_Type':p4,
'Transmission':p5,


Car Price Prediction Using Machine Learning
Last Checkpoint: last month
Trusted
Python 3 (ipykernel)
GUI Creation of the Project
from tkinter import *
import joblib
def show_entry_fields():
p1=float(e1.get())
p2=float(e2.get())
p3=float(e3.get())
p4=float(e4.get())
p5=float(e5.get())
p6=float(e6.get())
p7=float(e7.get())
model = joblib.load('car_price_predictor')
data_new = pd.DataFrame({
'Present_Price':p1,
'Kms_Driven':p2,
'Fuel_Type':p3,
'Seller_Type':p4,
'Transmission':p5,
'Owner':p6,
'Age':p7
},index=[0])
result=model.predict(data_new)
Label(master, text="Car Purchase amount").grid(row=8)
Label(master, text=result).grid(row=10)


Car Price Prediction Using Machine Learning
Last Checkpoint: last month
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
master = Tk()
master.title("Car Price Prediction Using Machine Learning")
label = Label(master, text = "Car Price Prediction Using Machine Learning"
, bg = "black", fg = "white"). \
grid(row=0,columnspan=2)
Label(master, text="Enter Present_Price").grid(row=1)
Label(master, text="Enter Kms_Driven").grid(row=2)
Label(master, text="Enter Fuel_Type - [Petrol/Diesel/CNG] [0/1/2]").grid(row=3)
Label(master, text="Enter Seller_Type - [Dealer/Individual] [0/1]").grid(row=4)
Label(master, text="Enter Transmission - [Manual/Automatic] [0/1]").grid(row=5)
Label(master, text="Enter Owner - [First Owner/Second Owner/Third Owner] [0/1/2]").grid(row=6)
Label(master, text="Age").grid(row=7)
e1 = Entry(master)
e2 = Entry(master)
e3 = Entry(master)
e4 = Entry(master)
e5 = Entry(master)
e6 = Entry(master)
e7 = Entry(master)
e1.grid(row=1, column=1)
e2.grid(row=2, column=1)
e3.grid(row=3, column=1)
e4.grid(row=4, column=1)
e5.grid(row=5, column=1)
Important Notification Request
Allow important notifications from Snipping Tool ?
You will get important notifications from Snipping Tool when do not disturb is on. To change this later, select this app in notification settings.
Allow
Not Now
18°C Haze
Search
ENG IN
6:31 PM 1/17/2024


Car Price Prediction Using Machine Learning
Last Checkpoint: last month
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
Label(master, text="Enter Present_Price").grid(row=1)
Label(master, text="Enter Kms_Driven").grid(row=2)
Label(master, text="Enter Fuel_Type - [Petrol/Diesel/CNG] [0/1/2]").grid(row=3)
Label(master, text="Enter Seller_Type - [Dealer/Individual] [0/1]").grid(row=4)
Label(master, text="Enter Transmission - [Manual/Automatic] [0/1]").grid(row=5)
Label(master, text="Enter Owner - [First Owner/Second Owner/Third Owner] [0/1/2]").grid(row=6)
Label(master, text="Age").grid(row=7)
e1 = Entry(master)
e2 = Entry(master)
e3 = Entry(master)
e4 = Entry(master)
e5 = Entry(master)
e6 = Entry(master)
e7 = Entry(master)
e1.grid(row=1, column=1)
e2.grid(row=2, column=1)
e3.grid(row=3, column=1)
e4.grid(row=4, column=1)
e5.grid(row=5, column=1)
e6.grid(row=6, column=1)
e7.grid(row=7, column=1)
Button(master, text='Predict', command=show_entry_fields).grid()
mainloop()
Important Notification Request
Allow important notifications from Snipping Tool ?
You will get important notifications from Snipping Tool when do not disturb is on. To change this later, select this app in notification settings.
Allow
Not Now
18°C Haze
Search
ENG IN
6:31 PM 1/17/2024

OUTPUT



	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner
0	ritz	2014	3.35	5.59	27000	Petrol	Dealer	Manual	First Owner
1	sx4	2013	4.75	9.54	43000	Diesel	Dealer	Manual	First Owner
2	ciaz	2017	7.25	9.85	6900	Petrol	Dealer	Manual	First Owner
3	wagon r	2011	2.85	4.15	5200	Petrol	Dealer	Manual	First Owner
4	swift	2014	4.60	6.87	42450	Diesel	Dealer	Manual	First Owner

CONCLUSION

Sales of used automobiles are rising globally due to rising new car prices and consumers' inability to afford them. Consequently, there is a pressing need for a Used Car Price Prediction system that uses a range of features to efficiently assess the car's worthiness. The suggested method will assist in estimating used automobile prices with greater accuracy. In this work, four distinct machine learning algorithms—linear regression, random forests,Gradient Boosting andXGB Regression—are compared.

Acknowledgement

I take this opportunity to express my profound gratitude and deep regards to my faculty, supervisor Prof.Biswajit Mondal for his exemplary guidance, monitoring and constant encouragement throughout the course of this project. The blessing, help and guidance given by him time to time shall carry me a long way in the journey of life on which I am about to embark.

I am obliged to my project mentor for the valuable information provided by him in his respective fields. I am grateful for his cooperation during the period of my assignment.

Bidhan Chandra Sen

Future Scope For Improvement

- Various car company can use these models and modify themaccording to their need.
- Beneficiary who intend to buy a car can use these trained models to check whether they get best benefit. The trained models would be required to be implemented in a platform or interface easily accessible as well as with an easy GUI.

BIBLIOGRAPHY

https://ieeexplore.ieee.org/document/9696839