Car Price Prediction Project

NAME OF THE PROJECT

Submitted by:

Salman

Pradhan

ACKNOWLEDGMENT

I wish to thank Flip Robo Technologies for assigning this project to me and thanks to Mr.Shubham Yadav sir for his valuable technical support on this project.

INTRODUCTION

- Business Problem Framing

  First We scrape the data of used cars from websites(Olx and cardekho).

  Then We have to build a model using Machine Learning in order to predict the Price of used cars.

- Conceptual Background of the Domain Problem

  Created model can be used for predicting Price of used car.

- Review of Literature

  Now a days if people buy anything,then they are searching for the price from online websites.By using this model,we predict the price of car .

- Motivation for the Problem Undertaken

  DataScience help us to make predictions at areas like health sectors,

  education, media etc. For our project we decided to implement a model which Predict price of used cars which will help customers who wants to sell their cars so that the owner gets his original price of car without broker.We can also make an app like cardekho ,olx etc.

Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem

  The prediction will base on the dataset which is scrapped from Olx and cardekho websites.

  Scrapping data from websites contains features like Brand,Model,Variant,registration year,Fuel,Transmission,Km driven,No of owners,Location and Price.

- Data Sources and their formats

  To collect data first we need to scrape Brand,Model,Variant,registration year,Fuel,Transmission,Km driven,No of owners,Location and Price.

  We scrape data by using selenium.

```
:
import pandas as pd
from selenium import webdriver
import time
from selenium.common.exceptions import NoSuchElementException
```

```
:
driver=webdriver.Chrome(r"chromedriver.exe")
time.sleep(2)
url="https://www.olx.in/"
driver.get(url)#enter to olx.in through chrome browser
```

```
driver.get("https://www.cardekho.com/")#enter into cardekho.com for scrapping
```

```
# car search for ahmedabad
driver.get(driver.find_element_by_xpath("//nav[@class='gsc_col-sm-12 gsc_col-md-12']/ul/li[2]/ul/li/ul/li/a").get_attribute("href
```

After scrapping data we named it as car.csv . Now we can build model by using this csv file. Then we are ready to

Importing Libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
df=pd.read_csv("car.csv")#load dataset
df.head()
```

	Unnamed: 0	Unnamed: 0.1	Brand	model	variant	year	fuel	transmission	km driven	No of owners	location	price
0	0	0	BMW	5 Series	523i	2010	Petrol	Automatic	69,000 km	2nd	Thane West, Thane, Maharashtra	₹ 10,25,000
1	1	1	Hyundai	Elite i20	Asta 1.2	2016	Petrol	Manual	27,884 km	-	Baner Road, Pune, Maharashtra	₹ 6,40,000
2	2	2	Renault	KWID	RXL	2016	Petrol	Manual	21,000 km	1st	Thillai Nagar, Tiruchirappalli, Tamil Nadu	₹ 3,00,000
3	3	3	Maruti Suzuki	Wagon R 1.0	1.0 LXi	2014	Petrol	Manual	8,350 km	1st	Govind Nagar, Nashik, Maharashtra	₹ 3,80,000
4	4	4	Mercedes-Benz	B Class	B180 Sport	2014	Petrol	Automatic	35,000 km	2nd	Adajan, Surat, Gujarat	₹ 12,50,000

go for model building.

1. The dataset has 10 columns .our target variable Price other columns are our independent features
2. All the columns are object type.
3. There is no null values present in the dataset.

- Data Preprocessing Done

1.Some rows have '- ' present .so we have to remove these rows.

2.All most all the columns like Brand,Model, Variant etc. have so many noise .we have to remove noise and make these columns value as unique.

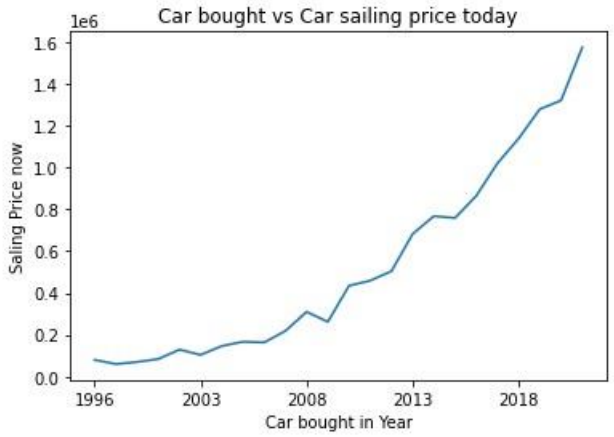
3.In year and Location column some values are repetative ,so have to make them one value.

4.Columns like Location,Brand,Model,Variant have lots of unique values.so we take first top 10 most repetative unique values and make other values as other.

- Data Inputs- Logic- Output Relationships

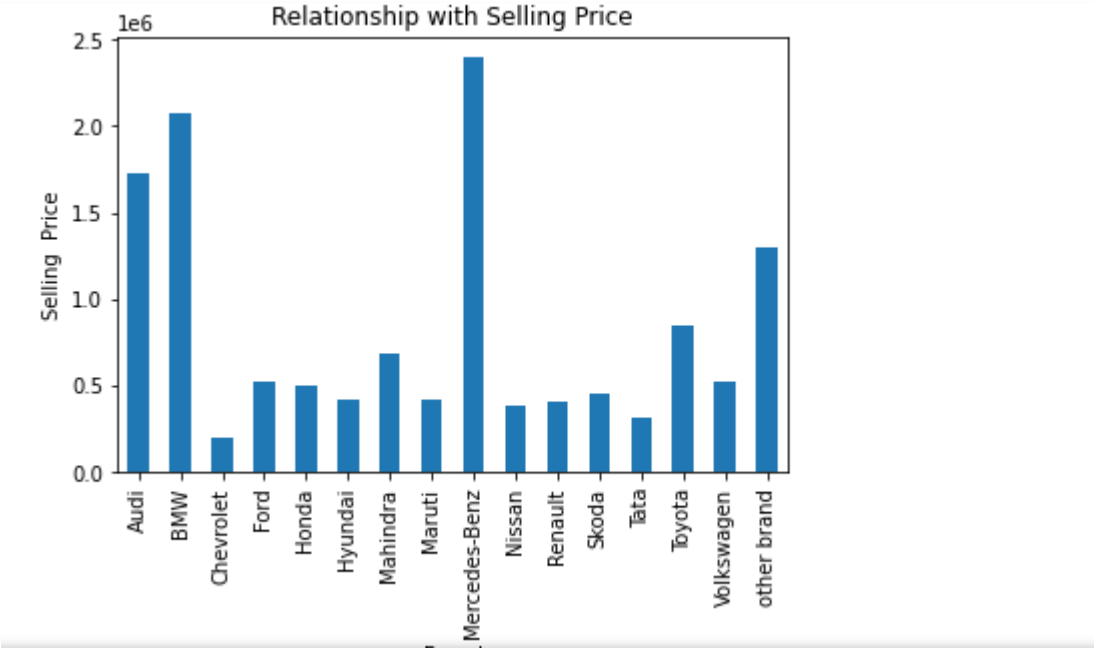
```
# We will check whether there is a relation between  registration year of car and the price
df.groupby('year')['Price'].mean().plot()#taking median due to outliers may be present
plt.xlabel('Car bought in Year')
plt.ylabel('Saling Price now')
plt.title("Car bought vs Car sailing price today")
```

```
Text(0.5, 1.0, 'Car bought vs Car sailing price today')
```



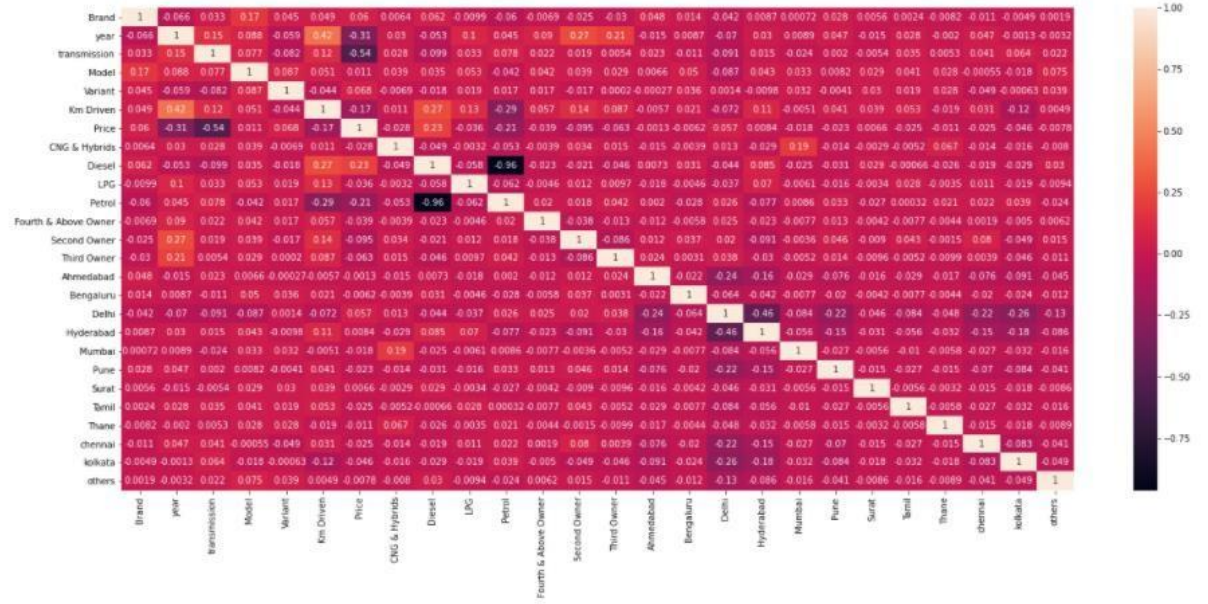

```
#As the registration year increases,Price of used car also increases
```

Car price of First owner is high and more the no. of owners lesser is the price of car.



Brands like Mercedes-Benz,Audi and Bmw have higher price of car.

```
plt.figure(figsize=(24,10))
sns.heatmap(df.corr(),annot=True)
plt.show()
```



Transmission,year,Petrol,km driven is negatively corelated with price and Diesel is positively corelated with price.

- State the set of assumptions (if any) related to the problem under consideration

1.We have taken year column as this year(2021) minus registration year of car before model building.

2.Car Price has taken between 20000 and 9000000.

- Hardware and Software Requirements and Tools Used

NumPy: Base n-dimensional array package **Matplotlib**: Comprehensive 2D/3D plotting **Seaborn**: For plotting graph
Pandas: Data structures and analysis
Scikit-learrn: provides a range of algorithm
Selenium: For scrpping data from websites

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

1.Understand business problem
2.Scrapping data and save it in a csv file
3.Data analysis
4.Data Cleaning
5.Visualization with Price
6.Data preprocessing
7.Feature scaling
8.Model bulding

- Testing of Identified Approaches (Algorithms)

1.LinearRegression

2.Lasso
3.Ridge
4.ElasticNet
5.RandomForestRegressor
6.XGBRegressor

- Run and Evaluate selected models

  After dividing dataset in independent and dependent variable,we have to choose best random_state for our model.

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score,mean_squared_error,mean_absolute_error
```

```
#choosing best random_state
maxacc=0
maxrs=0
lr=LinearRegression()
for i  in range(1,100):
    x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.1,random_state=i)
    lr.fit(x_train,y_train)
    y_pred=lr.predict(x_test)
    acc=r2_score(y_test,y_pred)
    if acc>maxacc:
        maxacc=acc
        maxrs=i
print('best accuracy score is',maxacc,'on random_state',maxrs)
```

```
best accuracy score is 0.6848273968137861 on random_state 74
```

Then we have to use random_state 74 on other algorithms to see in which model my accuracy is good.

```
model=[lr,ls,rd,els,dt]
```

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.05,random_state=74)
for i in model:
    i.fit(x_train,y_train)
    y_pred=i.predict(x_test)
    print(i)
    print(r2_score(y_test,y_pred))
    print(mean_squared_error(y_test,y_pred))
    print(mean_absolute_error(y_test,y_pred))
```

```
LinearRegression()
0.6927190129451797
206096822405.37582
274785.1183121079
Lasso()
0.6927302043417598
206089316209.514
274778.8497463478
Ridge()
0.6943501582255216
205002794875.77786
274019.7857552726
ElasticNet()
0.41479256811187526
392505222389.9413
360307.47621716984
DecisionTreeRegressor()
0.8132341646333757
125265951440.33333
137777.77366255145
```

```
from sklearn.ensemble import RandomForestRegressor
rf=RandomForestRegressor()
rf.fit(x_train,y_train)
y_Pred=rf.predict(x_test)
acc=r2_score(y_test,y_pred)
acc
```

```
0.8132341646333757
```

From using above models we see that RandomForestRegressor and DecisionTreeRegressor are giving good accuracy compared to all regression algorithms and we check cross val score of both model and found that RandomForestRegressor is the best algorithm for the dataset.So we have to do Hyperparameter Tuning in RandomForestRegressor

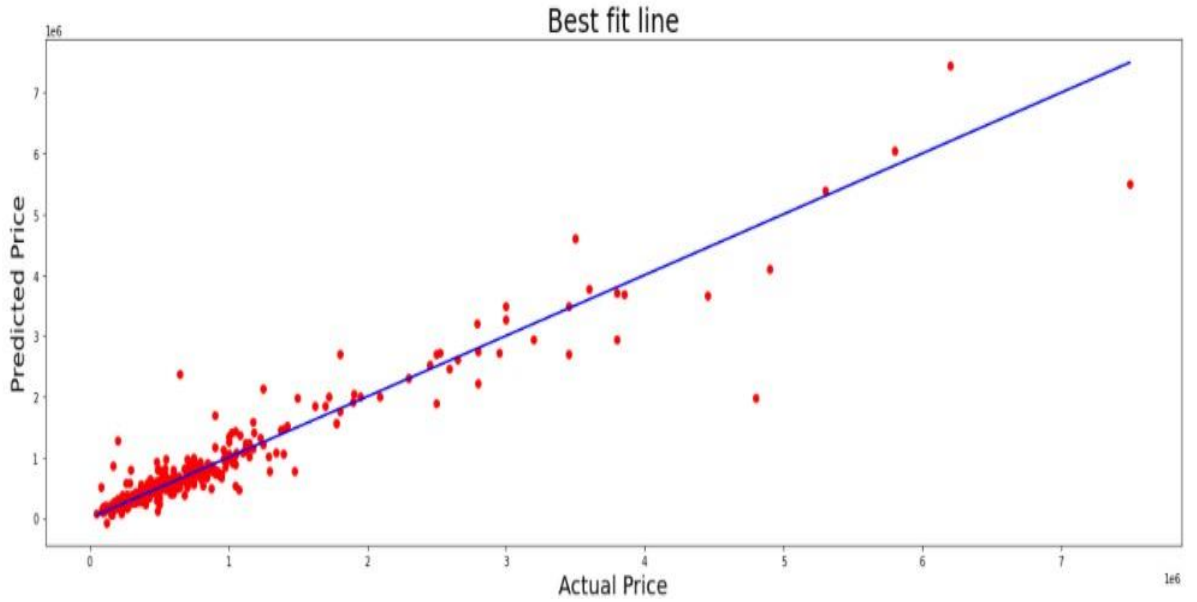
- Key Metrics for success in solving problem under consideration

I have used r2_score (to see accuracy of model)and mean_squared_error to check how close regression line is to set of points)to see which model is best for our dataset.I conclude that XGBRegressor is the best model for
the dataset.

- Visualizations

  Below is our best fit line for this use case.

```
plt.figure(figsize=(22,7))
plt.scatter(x=y_test,y=y_pred,color='r')
plt.plot(y_test,y_test,color='b')
plt.xlabel('Actual Price',fontsize=20)
plt.ylabel('Predicted Price',fontsize=20)
plt.title('Best fit line',fontsize=25)
plt.show()
```



- Interpretation of the Results

  Our model gives us accuracy score of 81.32% and after hypertuning my model ,we get accuracy of 94.45%.

CONCLUSION

- Key Findings and Conclusions of the Study
  1. Km driven is negatively related to Car price.
  2. As the registration year increases,Price of used car also increases
  3. Brands like Mercedes-Benz,Audi and Bmw have higher price of car.
  4. Car price of First owner is high and more the no. of owners lesser is the price of car.
  5. Diesel car have more price than others.
  6. Cars where transmission is Automatic have generally higher price than manual.
- Learning Outcomes of the Study in respect of Data Science

➢ At first, we scrape data from olx and cardekho.Then we keep data in a dataframe and save for model building.

➢ Then we analyse dataset and clean the unnecessary and noise data from the dataset.After cleaning we do some visualization to extract some information from the dataset.

➢ Then we preprocess data and build our model by using scikit learn algorithms.

➢ After getting our our best model we hypertune its parameter for better accuracy and save the model for future prediction purpose.

- Limitations of this work and Scope for Future Work

  As our model gives us approximately 94% of accuracy,It is used for predicting car price.

  As data is increasing every day,if we scrape more data from websites ,our model extracts more information from the data and hence our accuracy also increases.