A Report On Used Car Price Prediction

Submitted By:
Rohit Kachhal

ACKNOWLEDGMENT

I would like to express my special gratitude to “Flip Robo” team, who has given me this opportunity to deal with a beautiful dataset and it has helped me to improve my analyzation skills.

A huge thanks to my academic team “Data trained” who are the reason behind what I am doing today.

INTRODUCTION

- Business Problem Framing
  - ❖ With the covid 19 impact in the market, lot of changes are observed in the car market. Now some cars are in demand hence making them costly and some are not in demand hence cheaper. Small traders, who sell used cars, with the change in market due to covid 19 impact, are facing problems with their previous car price valuation machine learning models. So, need to build a new machine learning models from new data. I have to make car price valuation model.
- Conceptual Background of the Domain Problem
  - ❖ When it comes to shopping for a used car, budget is one of the main constraints. Prices can differ for a variety of factors, and the customer experience is quite different than buying a new vehicle. The search process is the first step, and the buyer has access to a wealth of information and reviews on the internet. With all that assumption, there is virtually no way to know if an offer is fair or overpriced. In practice, prior experience and extensive search can help.
- Review of Literature
  - ❖ The second-hand car market has continued to expand even as the reduction in the market of new cars. According to the recent report on India’s pre-owned car market by Indian Blue Book, nearly 4 million used cars were purchased and sold in 2018-19. The second-hand car market has created the business for both buyers and sellers. Most of the people prefer to buy the used cars because of the affordable price and they can resell that again after some years of usage which may get some profit. The price of used cars depends on many factors like fuel type, colour, model, mileage, transmission, engine, number of seats etc., The used cars price in the market will keep on changing. Thus, the evaluation model to predict the price of the used cars is required.

- Motivation for the Problem Undertaken
  - ❖ Based on data, Since the automobile services are providing a way to the transportation, the problem being looked at is to predict a price value of the used cars. Since the covid pandemic may impact the business. Hence traders are expecting to find a way to increase the business. So, machine learning is one of solution to help them to increase their business by build a model based on the current situation to predict the car price nearer to the exact value. The main objective behind this initiation is to help these traders to increase their business so that they can get profit out of selling price and as well as buyers can crack the best deal.

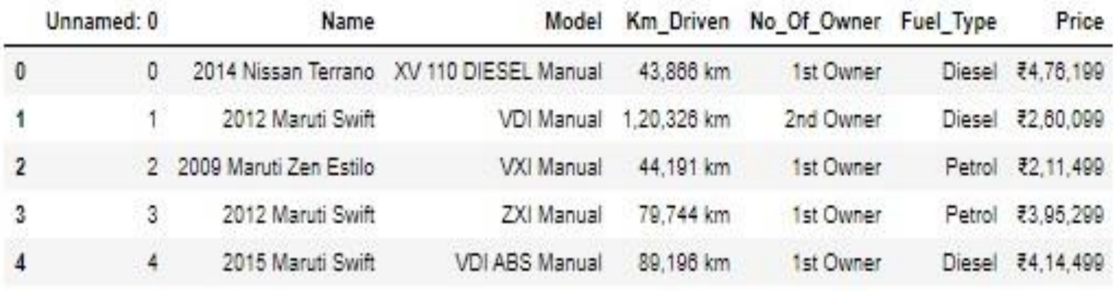
Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem

  In this given dataset, the target variable is 'Price' which contains continuous type of values. So, this problem is regression problem. For this problem, I have worked on 8 different regression model of machine learning. This dataset does not contain any null value. It is also observed that from some features like 'Name', 'Model' I need to extract different important values into other columns like 'Manufacturing Year', 'Brand Name', 'Model Name', 'Variant', 'Transmission Type'. Features 'Transmission_Type' have 89 values for which 'Transmission_Type' is no defined so replace these values by 'not defined'. Some features like 'Km_Driven', 'No_Of_Owner', 'Price' contains extra information which are not useful so need to replace it. I used different visualization plot to see the trend of data like distplot to see the distribution and skewness in dataset, boxplot to see the outliers, line plot & scatterplot to see the relationship between target and features, heatmap to see the multicollinearity. From these plots, I can observe skewness, outliers which I treated accordingly. I have also done hyperparameter tuning on finalize model to improve performance of model but not always hyperparameter tuning improve the performance of model. After that, I saved the model and compared the performance of model.

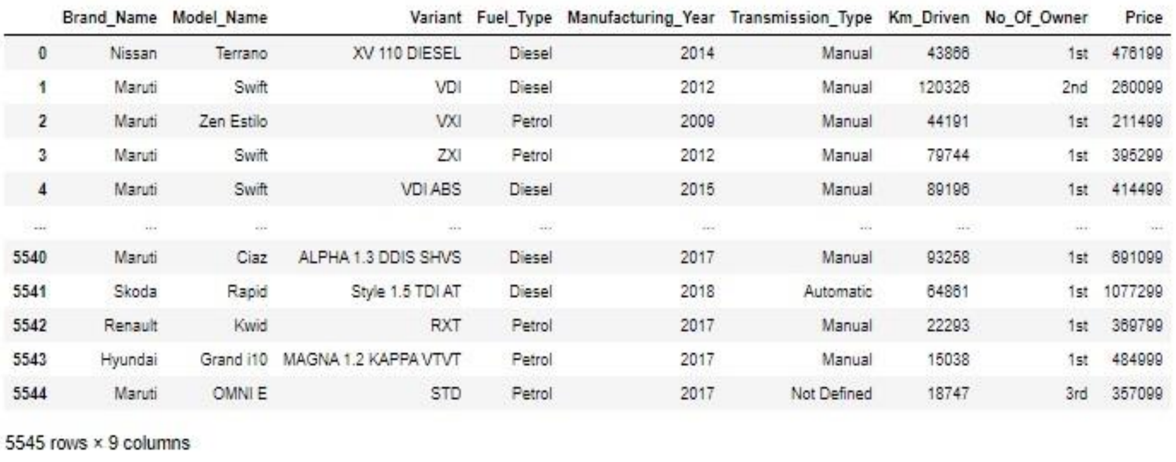
- Data Sources and their formats

  ❖ **DataFrame**



	Unnamed: 0	Name	Model	Km_Driven	No_Of_Owner	Fuel_Type	Price
0	0	2014 Nissan Terrano	XV 110 DIESEL Manual	43,866 km	1st Owner	Diesel	₹4,76,199
1	1	2012 Maruti Swift	VDI Manual	1,20,326 km	2nd Owner	Diesel	₹2,60,099
2	2	2009 Maruti Zen Estilo	VXI Manual	44,191 km	1st Owner	Petrol	₹2,11,499
3	3	2012 Maruti Swift	ZXI Manual	79,744 km	1st Owner	Petrol	₹3,95,299
4	4	2015 Maruti Swift	VDI ABS Manual	89,196 km	1st Owner	Diesel	₹4,14,499

❖ After data cleaning & pre-processing data frame



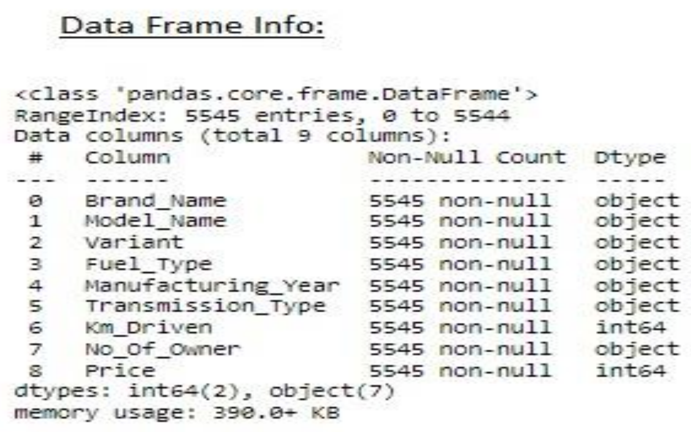
	Brand_Name	Model_Name	Variant	Fuel_Type	Manufacturing_Year	Transmission_Type	Km_Driven	No_Of_Owner	Price
0	Nissan	Terrano	XV 110 DIESEL	Diesel	2014	Manual	43866	1st	476199
1	Maruti	Swift	VDI	Diesel	2012	Manual	120326	2nd	260099
2	Maruti	Zen Estilo	VXI	Petrol	2009	Manual	44191	1st	211499
3	Maruti	Swift	ZXI	Petrol	2012	Manual	79744	1st	395299
4	Maruti	Swift	VDI ABS	Diesel	2015	Manual	89196	1st	414499
...	...	...	...	...	...	...	...	...	...
5540	Maruti	Ciaz	ALPHA 1.3 DDIS SHVS	Diesel	2017	Manual	93258	1st	691099
5541	Skoda	Rapid	Style 1.5 TDI AT	Diesel	2018	Automatic	64861	1st	1077299
5542	Renault	Kwid	RXT	Petrol	2017	Manual	22293	1st	369799
5543	Hyundai	Grand i10	MAGNA 1.2 KAPPA VTVT	Petrol	2017	Manual	15038	1st	484999
5544	Maruti	OMNI E	STD	Petrol	2017	Not Defined	18747	3rd	357099

5545 rows × 9 columns

Outcome:

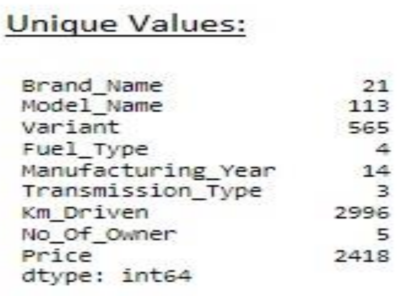
- ✓ This dataframe contains 5545 rows and 9 columns.
- ✓ There are no null values in the dataset.
- ✓ As data is important so, loss of data should be less than 5%.

❖ More about dataset



Data Frame Info:

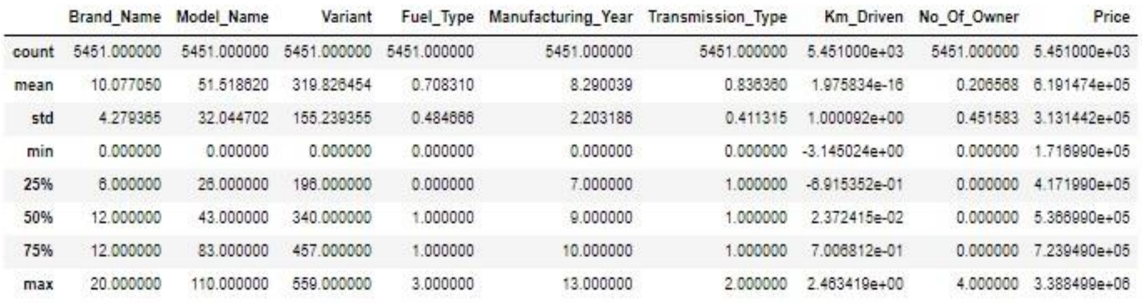
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5545 entries, 0 to 5544
Data columns (total 9 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Brand_Name          5545 non-null   object
 1   Model_Name          5545 non-null   object
 2   Variant             5545 non-null   object
 3   Fuel_Type           5545 non-null   object
 4   Manufacturing_Year  5545 non-null   object
 5   Transmission_Type   5545 non-null   object
 6   Km_Driven           5545 non-null   int64
 7   No_Of_Owner         5545 non-null   object
 8   Price               5545 non-null   int64
dtypes: int64(2), object(7)
memory usage: 390.0+ KB
```



Unique Values:

```
Brand_Name              21
Model_Name             113
Variant                565
Fuel_Type                4
Manufacturing_Year      14
Transmission_Type        3
Km_Driven             2996
No_Of_Owner              5
Price                 2418
dtype: int64
```

❖ Descriptive Statistical Analysis



	Brand_Name	Model_Name	Variant	Fuel_Type	Manufacturing_Year	Transmission_Type	Km_Driven	No_Of_Owner	Price
count	5451.000000	5451.000000	5451.000000	5451.000000	5451.000000	5451.000000	5.451000e+03	5451.000000	5.451000e+03
mean	10.077050	51.518620	319.826454	0.708310	8.290039	0.836360	1.975834e-16	0.206568	6.191474e+05
std	4.279365	32.044702	155.239355	0.484666	2.203186	0.411315	1.000092e+00	0.451583	3.131442e+05
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	-3.145024e+00	0.000000	1.716990e+05
25%	6.000000	26.000000	196.000000	0.000000	7.000000	1.000000	-6.915352e-01	0.000000	4.171990e+05
50%	12.000000	43.000000	340.000000	1.000000	9.000000	1.000000	2.372415e-02	0.000000	5.366990e+05
75%	12.000000	83.000000	457.000000	1.000000	10.000000	1.000000	7.006812e-01	0.000000	7.239490e+05
max	20.000000	110.000000	559.000000	3.000000	13.000000	2.000000	2.463419e+00	4.000000	3.388499e+06

- Outcome:
  - ✓ Analysis clearly describes the misleading values, Outliers and skewness present in the dataset.

- Data Preprocessing Done
  - ❖ From some features like ‘Name’, ‘Model’ I have to extract different important values into other columns like ‘Manufacturing Year’, ‘Brand Name’, ‘Model Name’, ‘Variant’, ‘Transmission Type’.
  - ❖ Features ‘Transmission_Type’ have 89 values for which ‘Transmission_Type’ is no defined. So, replace these values by ‘not defined’.
  - ❖ Some features like ‘Km_Driven’, ‘No_Of_Owner’, ‘Price’ contain extra information which are not useful. So, need to replace it.
  - ❖ Changed the data type of ‘Km_Driven’ & ‘Price’ from object to int.
  - ❖ Drop unwanted columns.
  - ❖ Dealing with outliers using z-score method.
  - ❖ Dealing with skewness using ‘yeo-johnson’ method of Power Transformer.

- Data Inputs- Logic- Output Relationships
  - ❖ I used line plot and scatter plot to visualize the impact of feature on target.
  - ❖ All data now are of either int or float.
  - ❖ I observed, ‘Mercedes’ is most expensive brand.
  - ❖ ‘Benz GLC Class’ of ‘Mercedes’ is most expensive model.

- Hardware and Software Requirements and Tools Used
  - ❖ Hardware:
    - ▪ Processor: I3 or above
    - ▪ RAM: 4 GB or above
    - ▪ Storage: 250 GB or above

- Software:
  - Anaconda
  - Jupyter notebook etc
- Libraries:
  - numpy
  - pandas
  - matplotlib
  - seaborn
  - model_selection: train_test_split, cross_val_score, GridSearchCV
  - datetime
  - preprocessing : PowerTransformer, StandardScaler,
  - score: r2 score, MSE score, RMSE score

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

  First of all, I cleaned the data like extract important information from column ‘Name’ , ‘Model’ into other columns like ‘Manufacturing Year’, ‘Brand Name’, ‘Model Name’, ‘Variant’, Transmission Type’. After that, drop unwanted and extra columns from data frame by using **drop()**. After that, rearrange the columns. After that, replace extra information from columns ‘Km_Driven’, ‘No_Of_Owner’ and ‘Price’ with blank by using **replace()**. Then changed the data type of ‘Km_Driven’ and ‘Price’ into **int**. After that, replace blank ‘Transmission_Type’ with ‘not defined’ by using **replace()**. Finally dealing with outliers using **z-score method**, skewness using **‘yeo-johnson’ method of Power Transformer** and scaling the independent variable by using **Standard Scaler**.

- Testing of Identified Approaches (Algorithms)

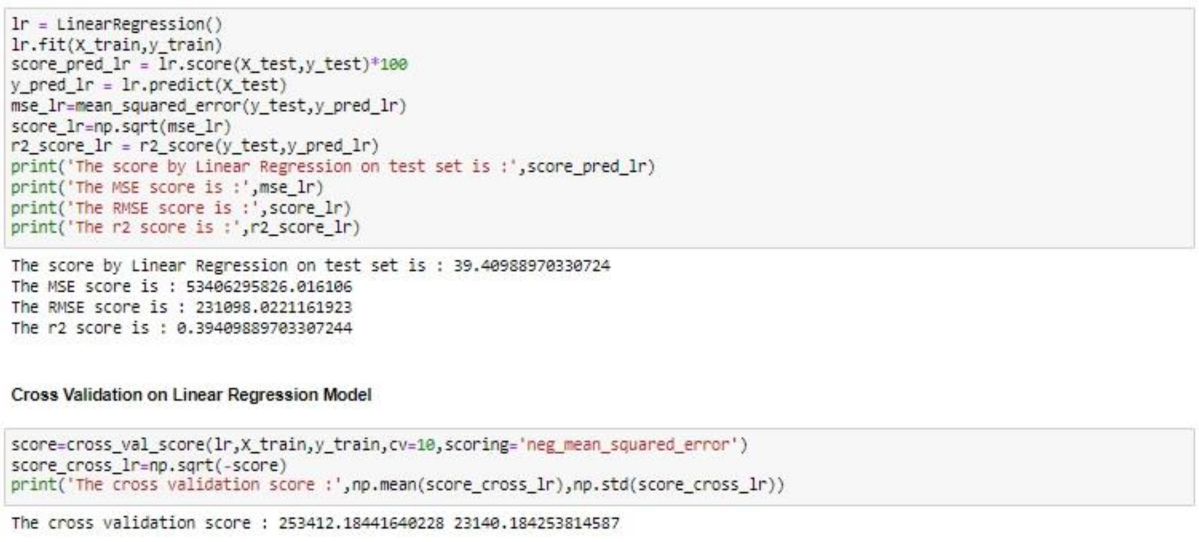
  As the problem is regression problem. In machine learning there are several algorithms for regression problem. I used total 8 algorithm to build best model on this dataset. The algorithms are:

  - ❖ Linear Regression
  - ❖ K-Neighbors Regressor
  - ❖ Decision Tree Regressor
  - ❖ Random Forest Regressor
  - ❖ Bagging Regressor
  - ❖ Gradient Boosting Regressor
  - ❖ AdaBoost Regressor
  - ❖ XGB Regressor

- ## Run and Evaluate selected models

  Here are the coding and insights of all models:

  - ### Linear Regression



```
lr = LinearRegression()
lr.fit(X_train,y_train)
score_pred_lr = lr.score(X_test,y_test)*100
y_pred_lr = lr.predict(X_test)
mse_lr=mean_squared_error(y_test,y_pred_lr)
score_lr=np.sqrt(mse_lr)
r2_score_lr = r2_score(y_test,y_pred_lr)
print('The score by Linear Regression on test set is :',score_pred_lr)
print('The MSE score is :',mse_lr)
print('The RMSE score is :',score_lr)
print('The r2 score is :',r2_score_lr)
```

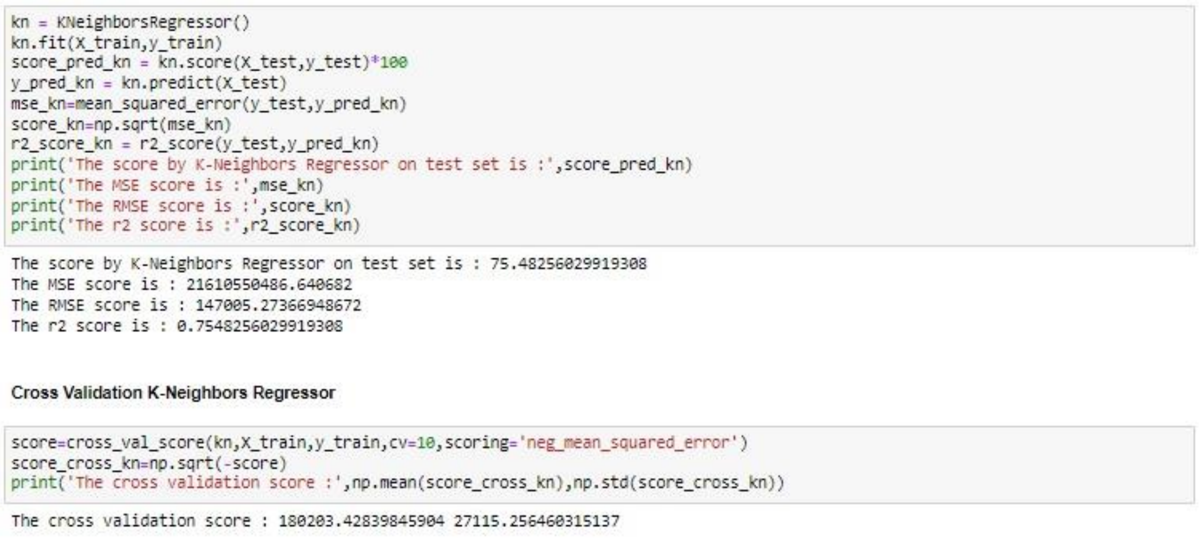
```
The score by Linear Regression on test set is : 39.40988970330724
The MSE score is : 53406295826.016106
The RMSE score is : 231098.0221161923
The r2 score is : 0.39409889703307244
```

Cross Validation on Linear Regression Model

```
score=cross_val_score(lr,X_train,y_train,cv=10,scoring='neg_mean_squared_error')
score_cross_lr=np.sqrt(-score)
print('The cross validation score :',np.mean(score_cross_lr),np.std(score_cross_lr))
```

```
The cross validation score : 253412.18441640228 23140.184253814587
```

  - ### K-Neighbors Regressor



```
kn = KNeighborsRegressor()
kn.fit(X_train,y_train)
score_pred_kn = kn.score(X_test,y_test)*100
y_pred_kn = kn.predict(X_test)
mse_kn=mean_squared_error(y_test,y_pred_kn)
score_kn=np.sqrt(mse_kn)
r2_score_kn = r2_score(y_test,y_pred_kn)
print('The score by K-Neighbors Regressor on test set is :',score_pred_kn)
print('The MSE score is :',mse_kn)
print('The RMSE score is :',score_kn)
print('The r2 score is :',r2_score_kn)
```

```
The score by K-Neighbors Regressor on test set is : 75.48256029919308
The MSE score is : 21610550486.640682
The RMSE score is : 147005.27366948672
The r2 score is : 0.7548256029919308
```

Cross Validation K-Neighbors Regressor

```
score=cross_val_score(kn,X_train,y_train,cv=10,scoring='neg_mean_squared_error')
score_cross_kn=np.sqrt(-score)
print('The cross validation score :',np.mean(score_cross_kn),np.std(score_cross_kn))
```

```
The cross validation score : 180203.42839845904 27115.256460315137
```

❖ Decision Tree Regressor

```
dt = DecisionTreeRegressor()
dt.fit(X_train,y_train)
score_pred_dt = dt.score(X_test,y_test)*100
y_pred_dt = dt.predict(X_test)
mse_dt=mean_squared_error(y_test,y_pred_dt)
score_dt=np.sqrt(mse_dt)
r2_score_dt = r2_score(y_test,y_pred_dt)
print('The score by  Decision TreeRegressor on test set is :',score_pred_dt)
print('The MSE score is :',mse_dt)
print('The RMSE score is :',score_dt)
print('The r2 score is :',r2_score_dt)
```

```
The score by  Decision TreeRegressor on test set is : 94.7055893209844
The MSE score is : 4666683416.870416
The RMSE score is : 68313.12770522527
The r2 score is : 0.9470558932098441
```

Cross Validation Decision Tree Regressor

```
score=cross_val_score(dt,X_train,y_train,cv=10,scoring='neg_mean_squared_error')
score_cross_dt=np.sqrt(-score)
print('The cross validation score :',np.mean(score_cross_dt),np.std(score_cross_dt))
```

```
The cross validation score : 98767.70914795814 29153.25027203568
```

❖ Random Forest Regressor

```
rf = RandomForestRegressor()
rf.fit(X_train,y_train)
score_pred_rf = rf.score(X_test,y_test)*100
y_pred_rf = rf.predict(X_test)
mse_rf=mean_squared_error(y_test,y_pred_rf)
score_rf=np.sqrt(mse_rf)
r2_score_rf = r2_score(y_test,y_pred_rf)
print('The score by  Random Forest Regressor on test set is :',score_pred_rf)
print('The MSE score is :',mse_rf)
print('The RMSE score is :',score_rf)
print('The r2 score is :',r2_score_rf)
```

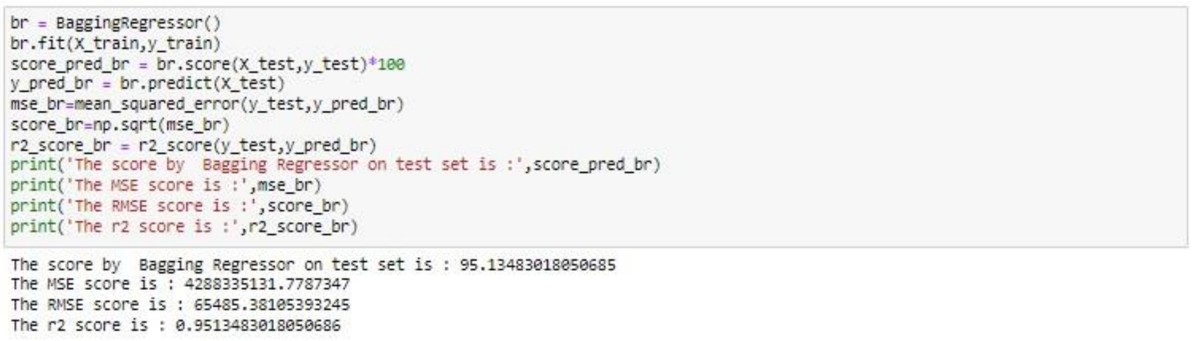
```
The score by  Random Forest Regressor on test set is : 95.42285609870767
The MSE score is : 4034458759.584178
The RMSE score is : 63517.389426708796
The r2 score is : 0.9542285609870768
```

Cross Validation Random Forest Regressor

```
score=cross_val_score(rf,X_train,y_train,cv=10,scoring='neg_mean_squared_error')
score_cross_rf=np.sqrt(-score)
print('The cross validation score :',np.mean(score_cross_rf),np.std(score_cross_rf))
```

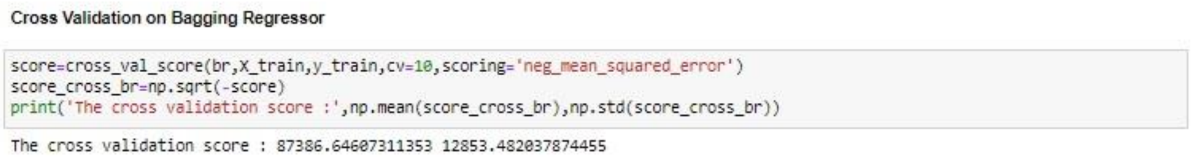
```
The cross validation score : 76352.37941556584 10866.290844139448
```

❖ Bagging Regressor



```
br = BaggingRegressor()
br.fit(X_train,y_train)
score_pred_br = br.score(X_test,y_test)*100
y_pred_br = br.predict(X_test)
mse_br=mean_squared_error(y_test,y_pred_br)
score_br=np.sqrt(mse_br)
r2_score_br = r2_score(y_test,y_pred_br)
print('The score by  Bagging Regressor on test set is :',score_pred_br)
print('The MSE score is :',mse_br)
print('The RMSE score is :',score_br)
print('The r2 score is :',r2_score_br)
```

```
The score by  Bagging Regressor on test set is : 95.13483018050685
The MSE score is : 4288335131.7787347
The RMSE score is : 65485.38105393245
The r2 score is : 0.9513483018050686
```

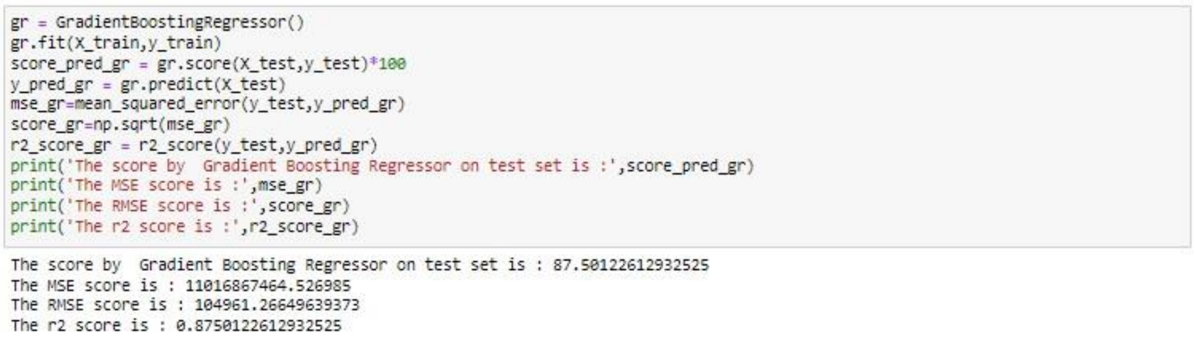


Cross Validation on Bagging Regressor

```
score=cross_val_score(br,X_train,y_train,cv=10,scoring='neg_mean_squared_error')
score_cross_br=np.sqrt(-score)
print('The cross validation score :',np.mean(score_cross_br),np.std(score_cross_br))
```

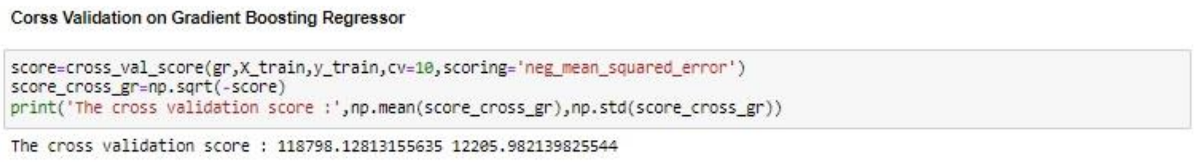
```
The cross validation score : 87386.64607311353 12853.482037874455
```

❖ Gradient Boosting Regressor



```
gr = GradientBoostingRegressor()
gr.fit(X_train,y_train)
score_pred_gr = gr.score(X_test,y_test)*100
y_pred_gr = gr.predict(X_test)
mse_gr=mean_squared_error(y_test,y_pred_gr)
score_gr=np.sqrt(mse_gr)
r2_score_gr = r2_score(y_test,y_pred_gr)
print('The score by  Gradient Boosting Regressor on test set is :',score_pred_gr)
print('The MSE score is :',mse_gr)
print('The RMSE score is :',score_gr)
print('The r2 score is :',r2_score_gr)
```

```
The score by  Gradient Boosting Regressor on test set is : 87.50122612932525
The MSE score is : 11016867464.526985
The RMSE score is : 104961.26649639373
The r2 score is : 0.8750122612932525
```

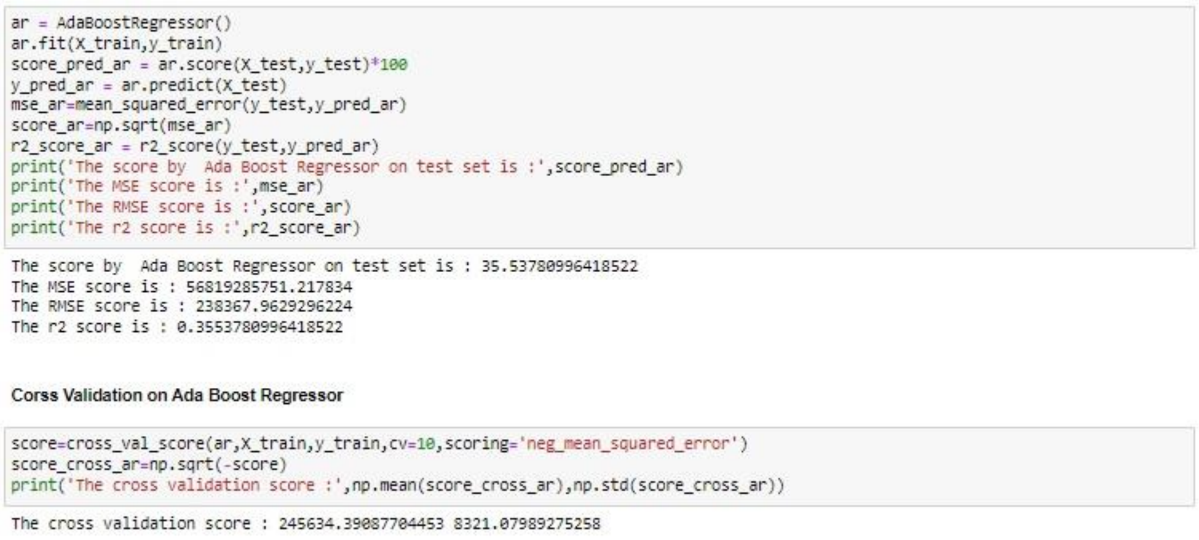


Corss Validation on Gradient Boosting Regressor

```
score=cross_val_score(gr,X_train,y_train,cv=10,scoring='neg_mean_squared_error')
score_cross_gr=np.sqrt(-score)
print('The cross validation score :',np.mean(score_cross_gr),np.std(score_cross_gr))
```

```
The cross validation score : 118798.12813155635 12205.982139825544
```

❖ AdaBoost Regressor



```
ar = AdaBoostRegressor()
ar.fit(X_train,y_train)
score_pred_ar = ar.score(X_test,y_test)*100
y_pred_ar = ar.predict(X_test)
mse_ar=mean_squared_error(y_test,y_pred_ar)
score_ar=np.sqrt(mse_ar)
r2_score_ar = r2_score(y_test,y_pred_ar)
print('The score by  Ada Boost Regressor on test set is :',score_pred_ar)
print('The MSE score is :',mse_ar)
print('The RMSE score is :',score_ar)
print('The r2 score is :',r2_score_ar)
```

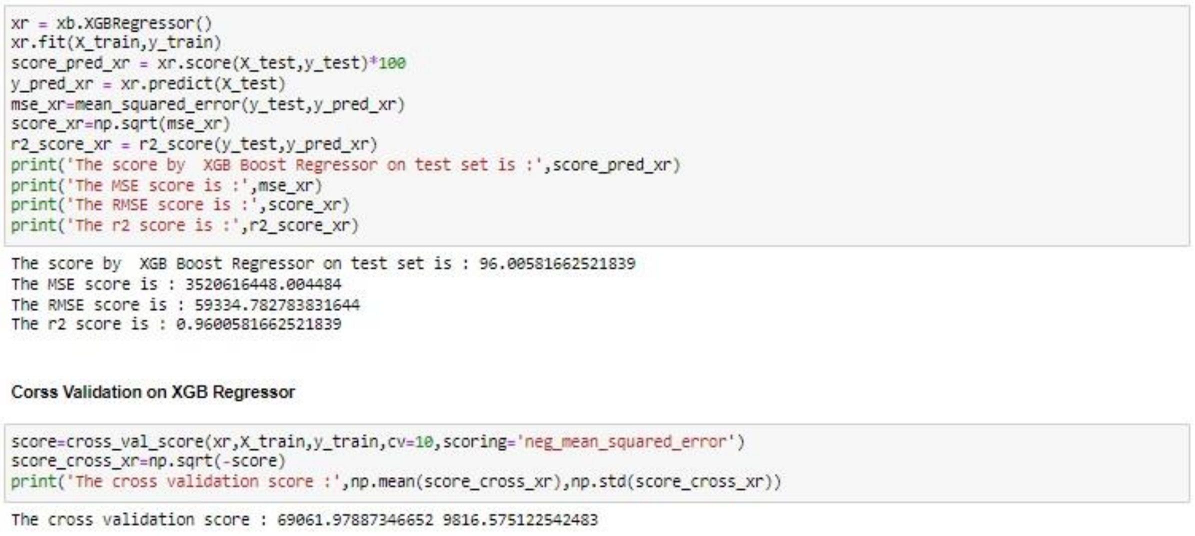
```
The score by  Ada Boost Regressor on test set is : 35.53780996418522
The MSE score is : 56819285751.217834
The RMSE score is : 238367.9629296224
The r2 score is : 0.3553780996418522
```

Corss Validation on Ada Boost Regressor

```
score=cross_val_score(ar,X_train,y_train,cv=10,scoring='neg_mean_squared_error')
score_cross_ar=np.sqrt(-score)
print('The cross validation score :',np.mean(score_cross_ar),np.std(score_cross_ar))
```

```
The cross validation score : 245634.39087704453 8321.07989275258
```

❖ XGB Regressor



```
xr = xb.XGBRegressor()
xr.fit(X_train,y_train)
score_pred_xr = xr.score(X_test,y_test)*100
y_pred_xr = xr.predict(X_test)
mse_xr=mean_squared_error(y_test,y_pred_xr)
score_xr=np.sqrt(mse_xr)
r2_score_xr = r2_score(y_test,y_pred_xr)
print('The score by  XGB Boost Regressor on test set is :',score_pred_xr)
print('The MSE score is :',mse_xr)
print('The RMSE score is :',score_xr)
print('The r2 score is :',r2_score_xr)
```

```
The score by  XGB Boost Regressor on test set is : 96.00581662521839
The MSE score is : 3520616448.004484
The RMSE score is : 59334.782783831644
The r2 score is : 0.9600581662521839
```

Corss Validation on XGB Regressor

```
score=cross_val_score(xr,X_train,y_train,cv=10,scoring='neg_mean_squared_error')
score_cross_xr=np.sqrt(-score)
print('The cross validation score :',np.mean(score_cross_xr),np.std(score_cross_xr))
```

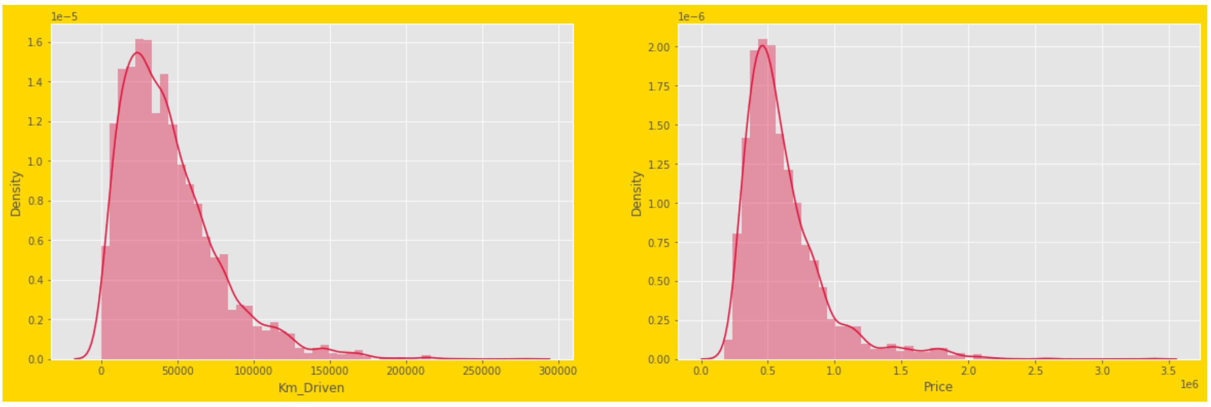
```
The cross validation score : 69061.97887346652 9816.575122542483
```

- Key Metrics for success in solving problem under consideration

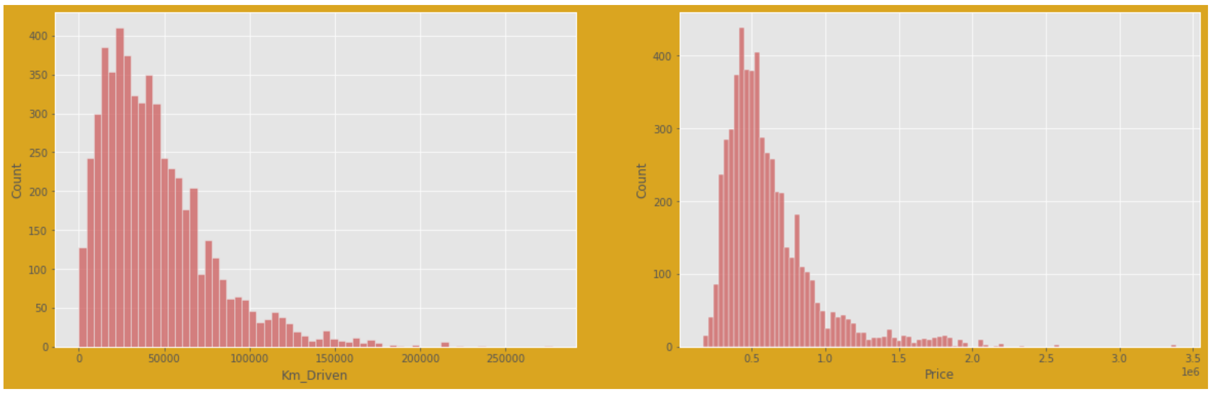
  Following are the metrics to observe good model:

  - ❖ Mean absolute error: MAE is a very simple metric which calculates the absolute difference between actual and predicted values. It should be minimum.
  - ❖ Mean squared error : MSE is a very simple metric with a little bit of change in mean absolute error. Mean squared error states that finding the squared difference between actual and predicted value.
  - ❖ Root mean absolute error: It is a simple square root of mean squared error.
  - ❖ R2 score: R2 score is a metric that tells the performance of your model, not the loss in an absolute sense that how many wells did your model perform.
  - ❖ Cross_val_score: Cross-validation is a resampling method that uses different portions of the data to test and train a model on different iterations. It is mainly used in settings where the goal is prediction, and one wants to estimate how accurately a predictive model will perform in practice.

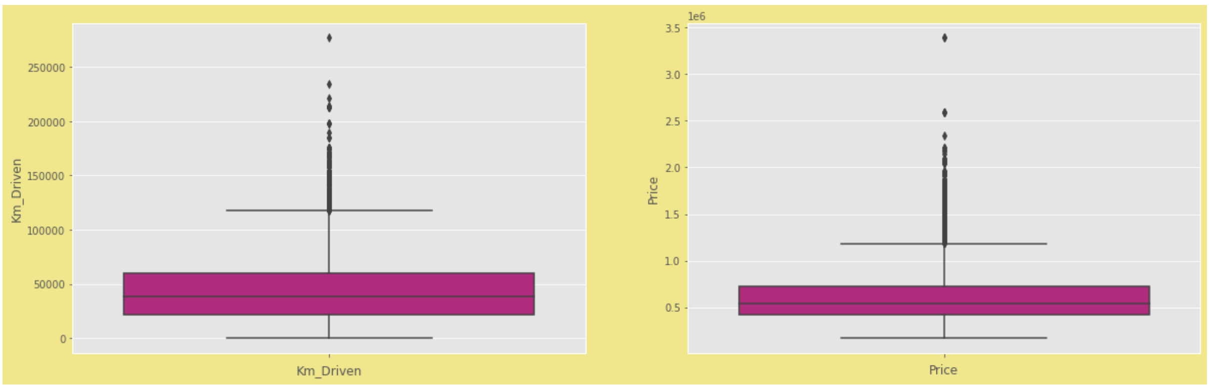
- Visualizations
  - ❖ Univariate
    - ▪ Using distplot checking distribution



    - ▪ Using histplot checking distribution



    - ▪ Using boxplot checking outliers

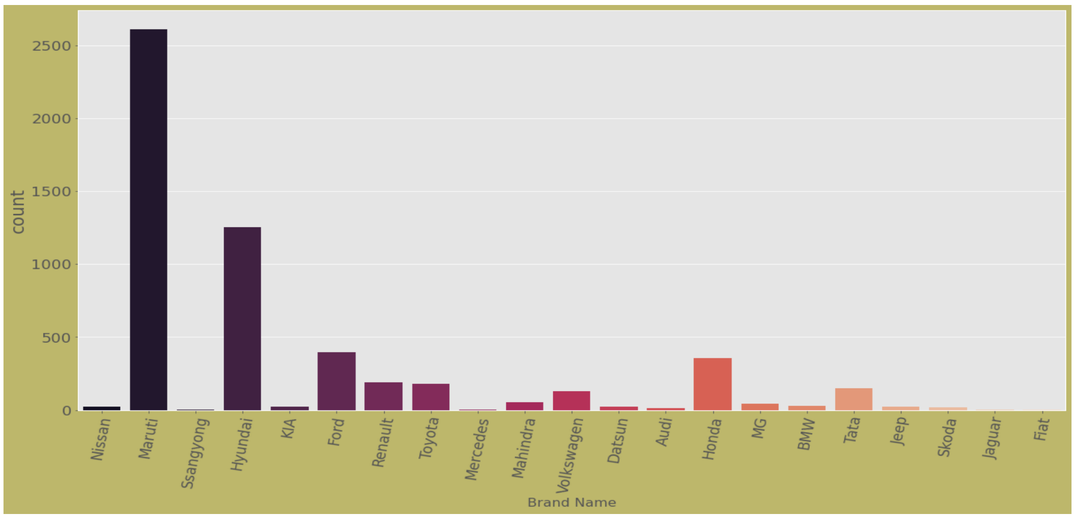


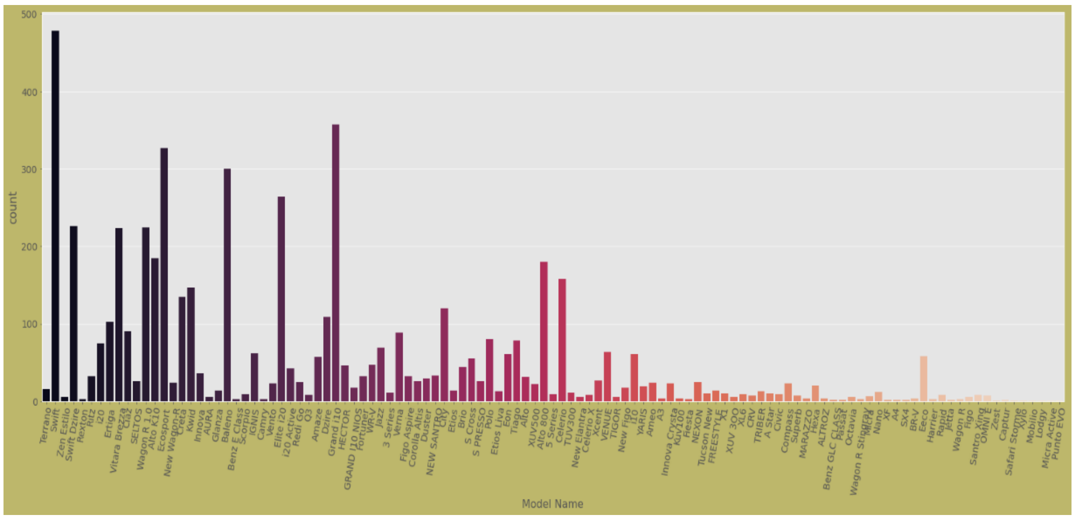
Outcome:

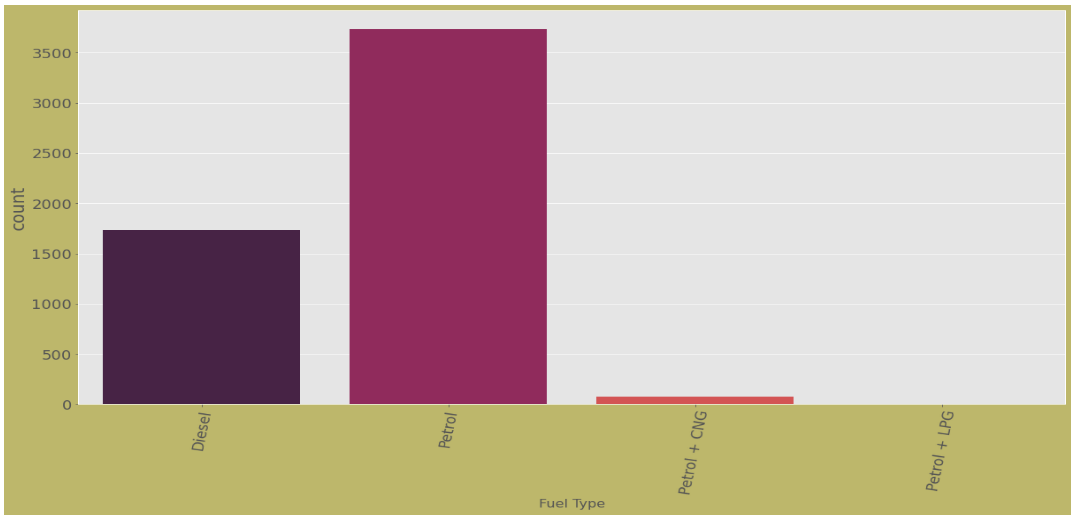
✓ Skewness is there

✓ Outliers is there

- Visualization of object column

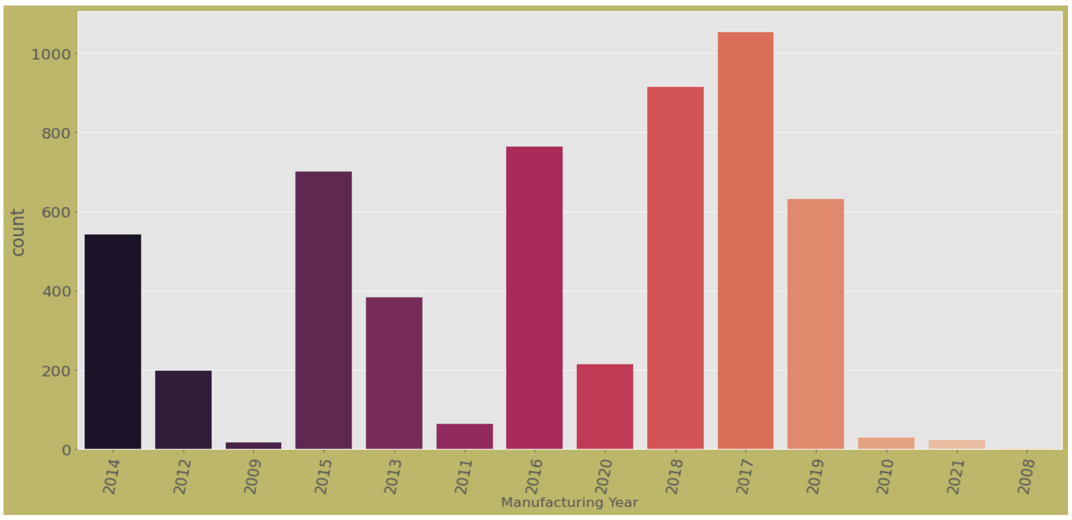


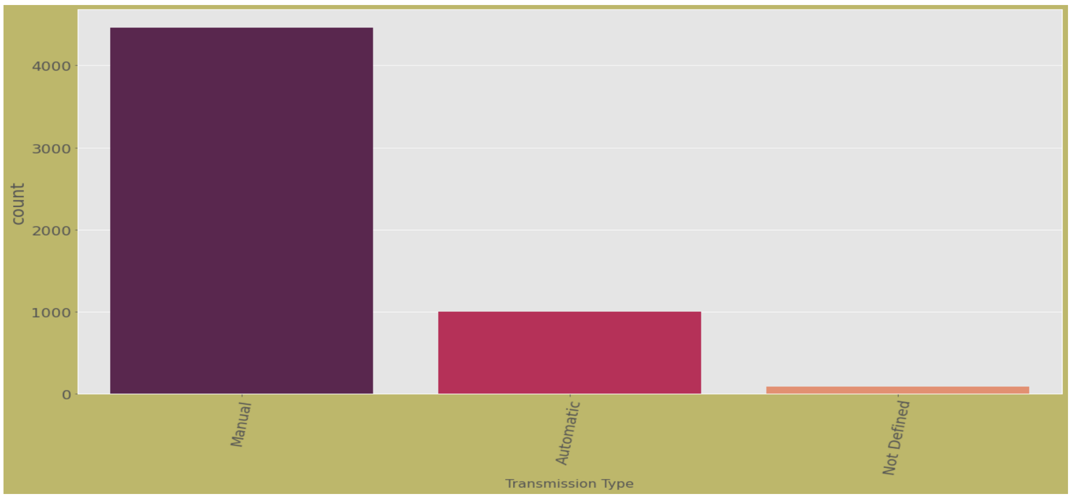


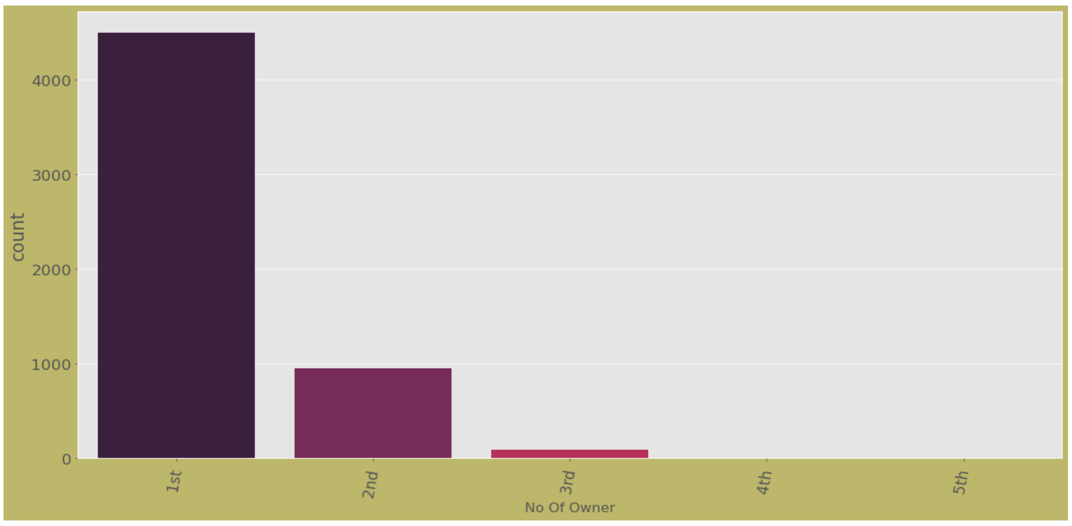


Outcome

- ✓ Most available cars are of ‘Maruti’, followed by ‘Hyundai’.
- ✓ Most available model is ‘Swift’ of ‘Maruti’, followed by ‘Grand i10’ of ‘Hyundai’ and so on.
- ✓ Mostly have fuel type as ‘Petrol’.



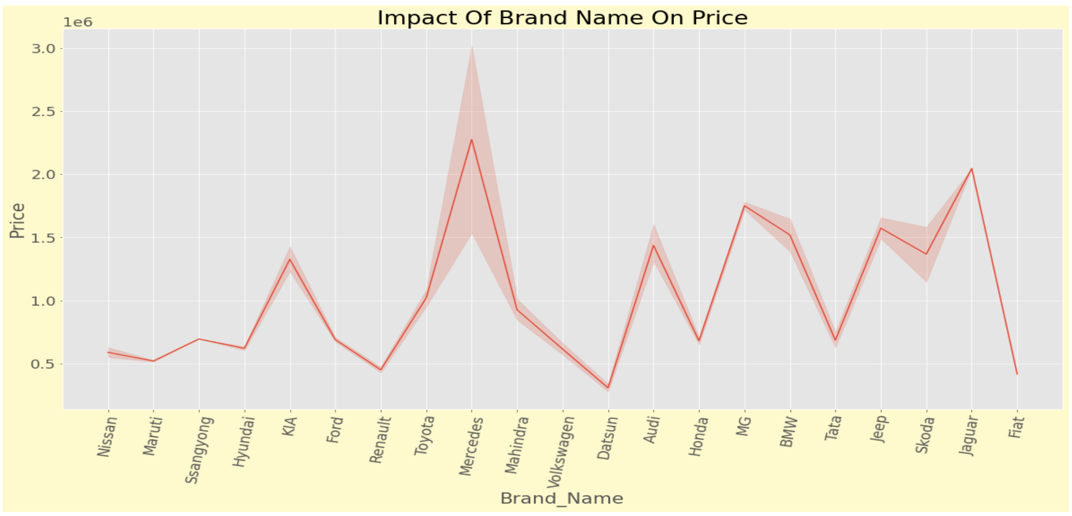


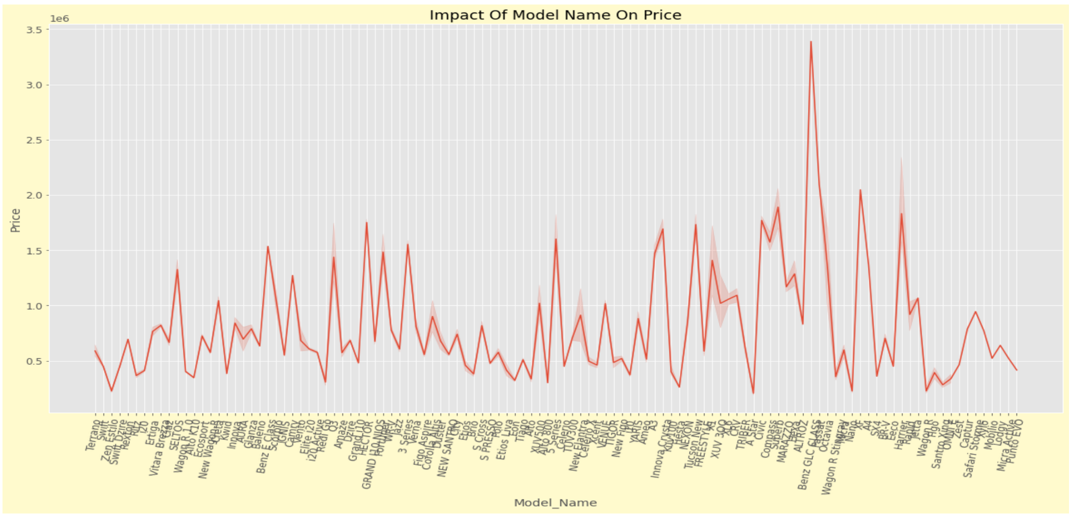


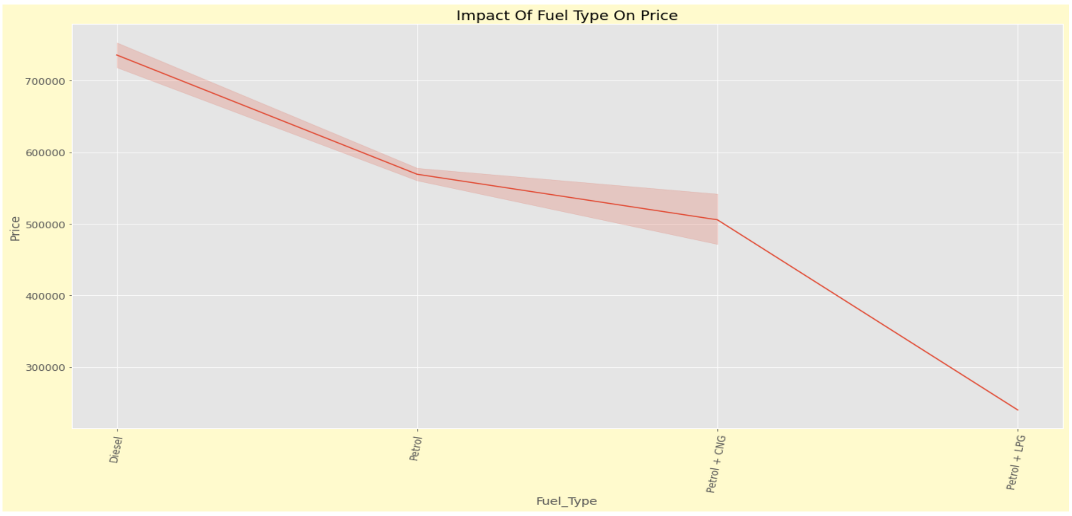
Outcome

- ✓ The cars that have 'Manufacturing Year' as 2017 are most in counting, followed by 2018.
- ✓ 'Transmission Type' as 'Manual' is most.
- ✓ Mostly available cars have '1st owner'.

❖ Bivariate

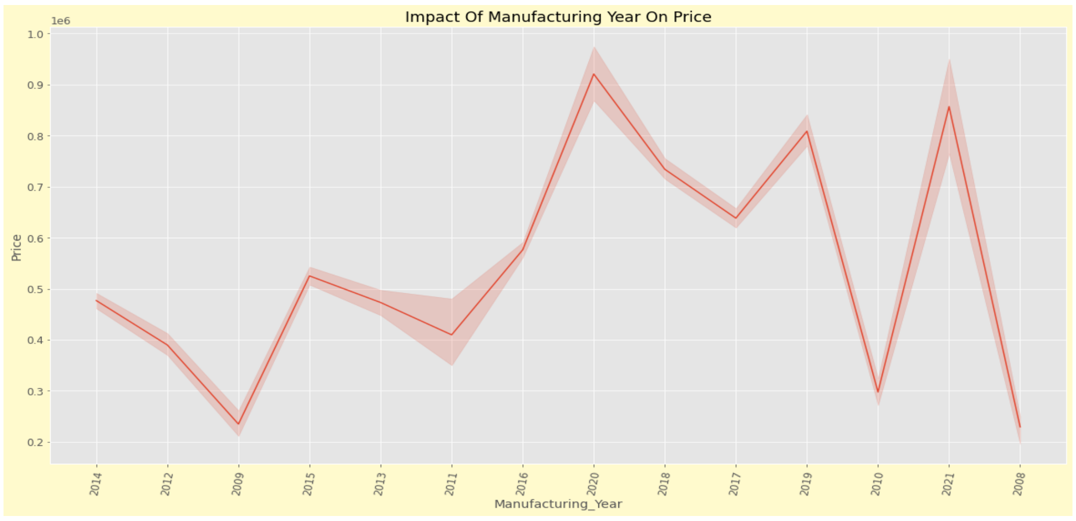


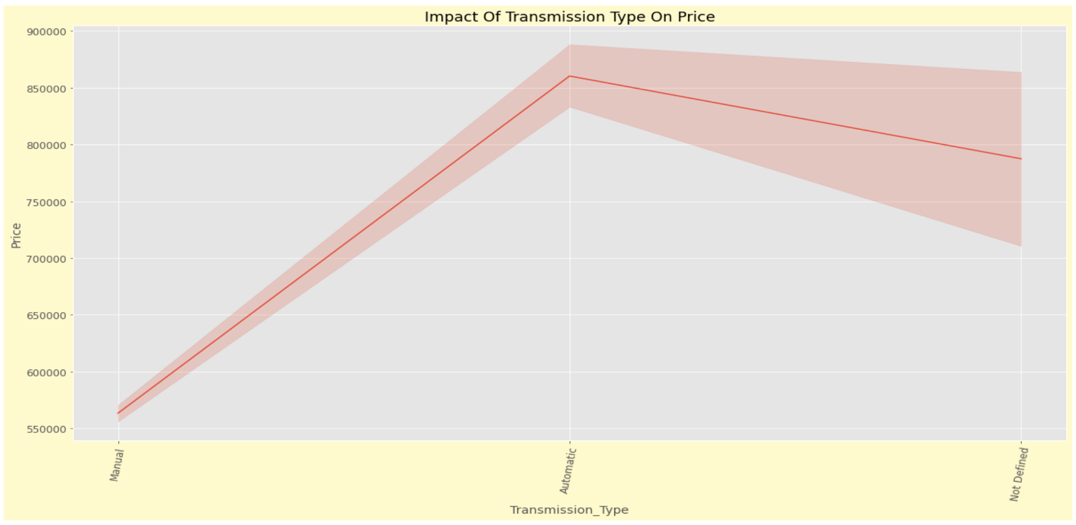


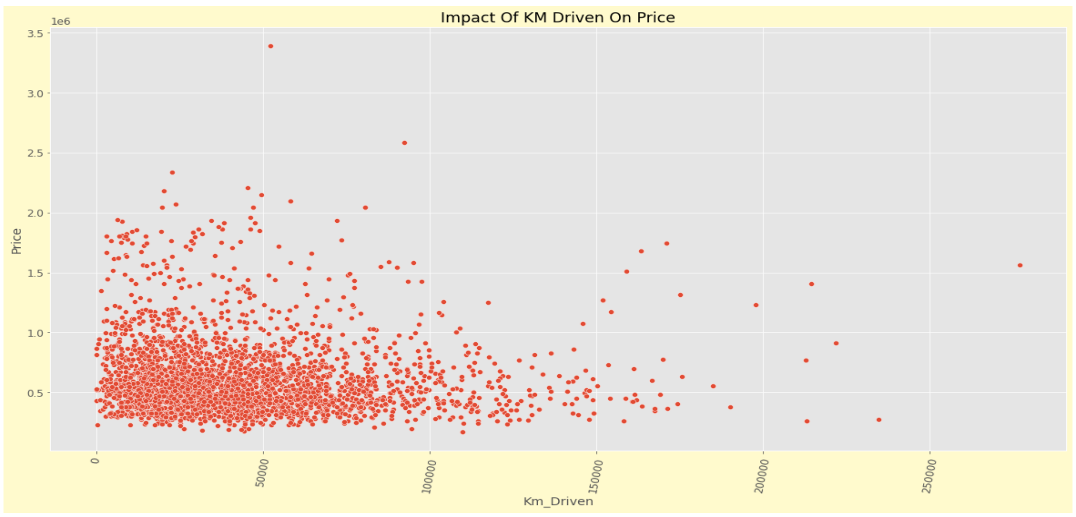


Outcome

- ✓ 'Mercedes' cars are most expensive, followed by 'Jaguar'.
- ✓ 'Benz GLC Class' of 'Mercedes' are the most expensive car model in the dataset.
- ✓ 'Diesel' as 'fuel_Type' cars are costly.

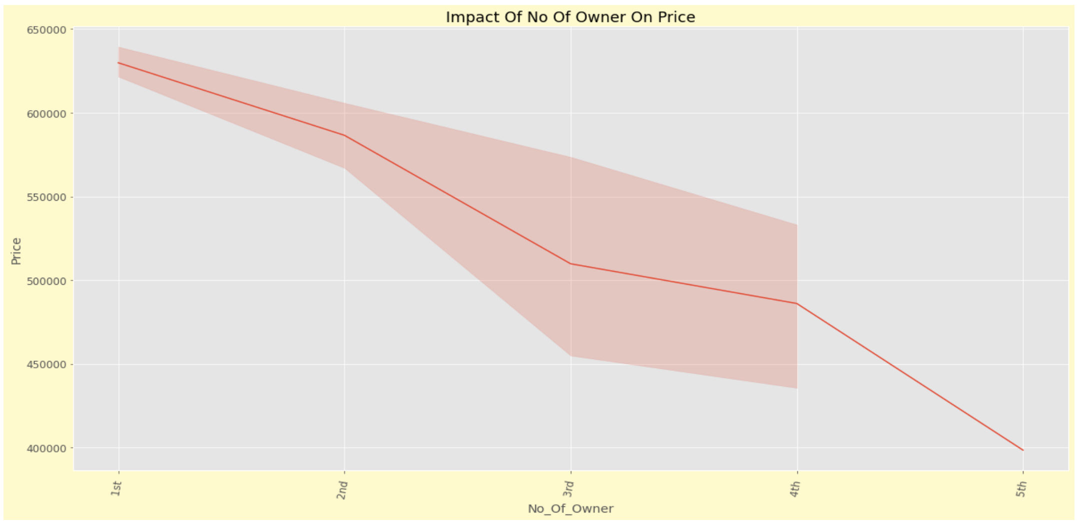






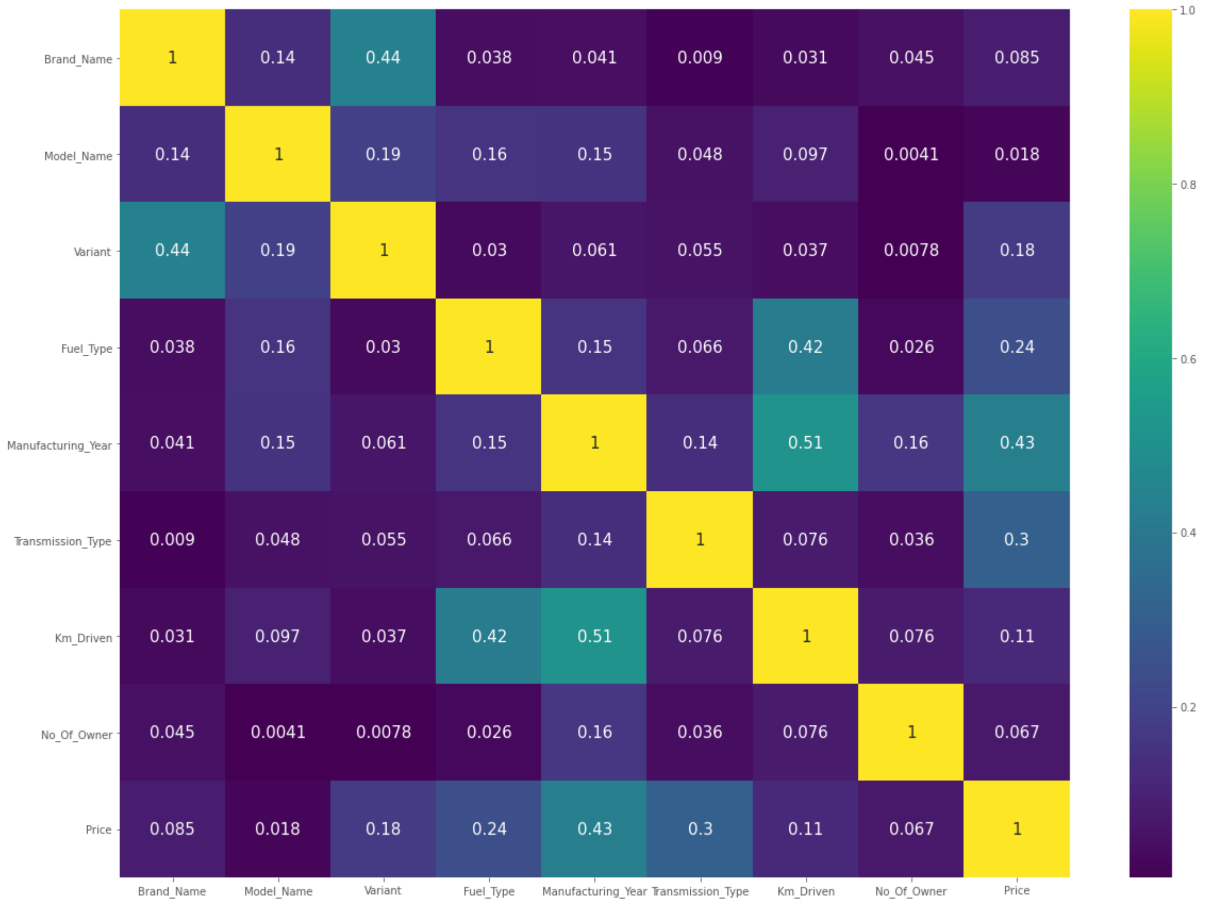
Outcome

- ✓ 'Price' is high when 'Manufacturing Year' is 2020, followed by 2021.
- ✓ 'Automatic' as 'Transmission_Type are expensive.
- ✓ Mostly cars have 'Km Driven' between 0 to 10000 KM.



- Outcome
  - ✓ Cars ‘Price’ are high when the car has ownership of ‘1st owner’.

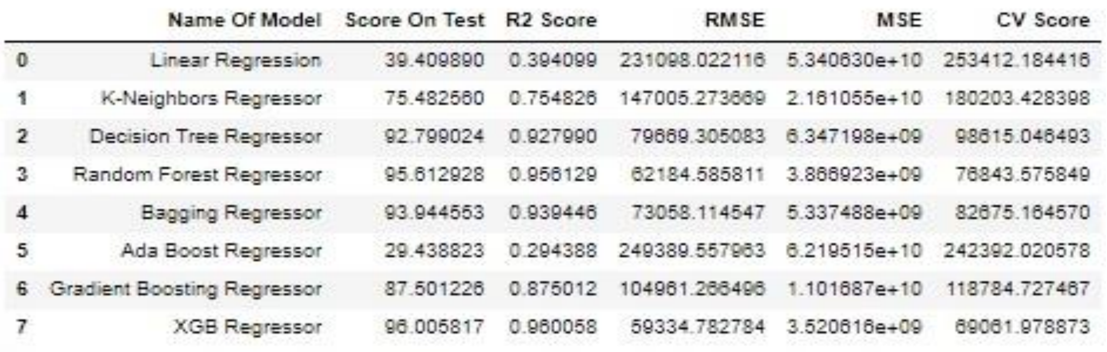
❖ Correlation matrix



- Outcome:
  - ✓ All features are vrey less correlated with target.

- Interpretation of the Results

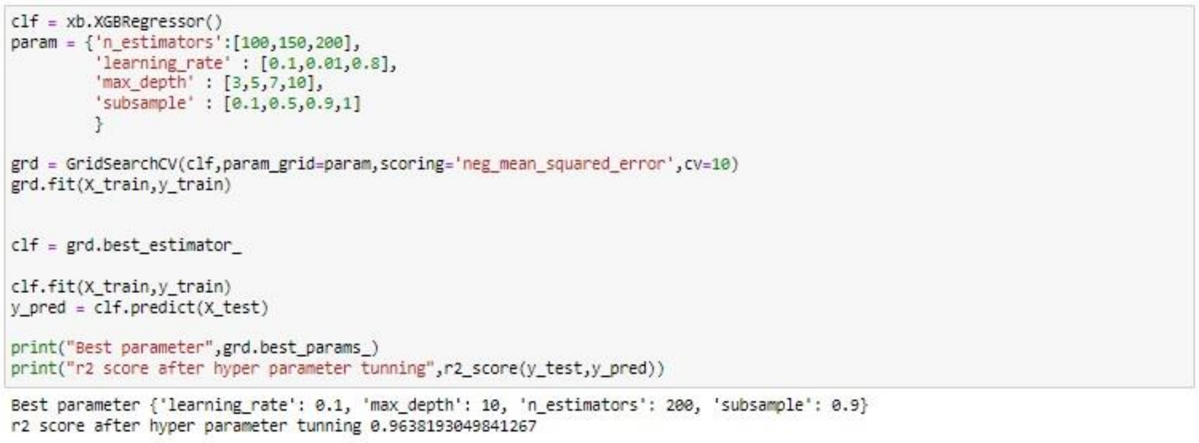
  ❖ Score of all model



	Name Of Model	Score On Test	R2 Score	RMSE	MSE	CV Score
0	Linear Regression	39.409890	0.394099	231098.022116	5.340630e+10	253412.184416
1	K-Neighbors Regressor	75.482560	0.754826	147005.273669	2.161055e+10	180203.428398
2	Decision Tree Regressor	92.799024	0.927990	79669.305083	6.347198e+09	98615.046493
3	Random Forest Regressor	95.612928	0.956129	62184.585811	3.866923e+09	76843.575849
4	Bagging Regressor	93.944553	0.939446	73058.114547	5.337488e+09	82675.164570
5	Ada Boost Regressor	29.438823	0.294388	249389.557963	6.219515e+10	242392.020578
6	Gradient Boosting Regressor	87.501226	0.875012	104961.266496	1.101687e+10	118784.727467
7	XGB Regressor	96.005817	0.960058	59334.782784	3.520616e+09	69061.978873

  - Outcome:
    - ✓ R2 score of Random Forest Regressor and XGB Regressor are also good. The RMSE score of XGB Regressor is less. Cross validation score is less for XGB Regressor as compared with Random Forest Regressor. So, XGB Regressor is final model on this dataset.

  ❖ Hyperparameter Tuning



```
clf = xb.XGBRegressor()
param = {'n_estimators':[100,150,200],
         'learning_rate' : [0.1,0.01,0.8],
         'max_depth' : [3,5,7,10],
         'subsample' : [0.1,0.5,0.9,1]
         }

grd = GridSearchCV(clf,param_grid=param,scoring='neg_mean_squared_error',cv=10)
grd.fit(X_train,y_train)


clf = grd.best_estimator_

clf.fit(X_train,y_train)
y_pred = clf.predict(X_test)

print("Best parameter",grd.best_params_)
print("r2 score after hyper parameter tunning",r2_score(y_test,y_pred))
```

```
Best parameter {'learning_rate': 0.1, 'max_depth': 10, 'n_estimators': 200, 'subsample': 0.9}
r2 score after hyper parameter tunning 0.9638193049841267
```

  - Outcome:
    - ✓ After hyperparameter tuning, I am able to improve performance. So, these parameters are the best parameters.

CONCLUSION

- Key Findings and Conclusions of the Study
  - ❖ Mostly the customers with intension of buying the car having best specification and low price.
  - ❖ There are certain cases, in which the car has high price with all the specifications. May be a color, variant, transmission type, engine can be key indicator to set price. Price may vary accordingly.
  - ❖ With this model built, we can certainly determine car price of a specific brand with high performance.

- Learning Outcomes of the Study in respect of Data Science
  - ❖ The outcome of the study includes good factor because dataset was quite interesting to handle as it contains all types of data in it. Also, dataset self-scrapped from cars24 website using selenium.
  - ❖ Improvement in computing technology which made it possible to examine social information that could not previously captured, processed and analysed.
  - ❖ New analytical techniques of machine learning can be used in used car price research. The power of visualization has helped us in understanding the data by graphical representation. From this representation I can easily understand the trend of data.
  - ❖ Data cleaning is one of the most important steps to remove unrealistic values and null values. This study is an exploratory attempt to use eight machine learning algorithms in estimating used car price prediction, and then compare their results.
  - ❖ With the hope that this study has moved a small step ahead in providing some methodological and empirical contributions for online platforms, and presenting an alternative approach to the valuation of used car price.

- Future direction of research may consider incorporating additional used car data from a larger economical background with more features.
- In this project ensemble techniques are also used like XGB Regressor, Gradient Boosting Regressor.
- After observing all model I can conclude that XGB Regressor performed best on dataset.

Limitations of this work and Scope for Future Work

- First draw back is scrapping the data as it is fluctuating process.
- Followed by more number of outliers and skewness these will reduce model performance.
- Also, we have to tried best method to deal with outliers, skewness and null values.
- Also, this study will not cover all Regression algorithms instead, it is focused on the chosen algorithm, starting from the basic ensemble techniques to the advanced ones.