PROJECT REPORT ON:

"Car Price Prediction Project"

Submitted by:

MEGHA SINGH

ACKNOWLEDGEMENT

The internship opportunity I have with Flip Robo Technologies is a great chance for learning and professional development. I perceive this opportunity as a big milestone in my career development. I will strive to use gained skills acknowledge in the best possible way.

I would like to extend my appreciation and thanks for the mentors from DataTrained and professionals from FlipRoboTechnologies who had extended their help and support.

References: www.scipy.org, Kaggle, Github

INTRODUCTION

With the covid 19 impact in the market, we have seen lot of Changes in the car market. Now some cars are in demand hence making them costly and some are not in demand hence cheaper. One of our clients works with small traders, who sell used cars. With the change in market due to covid 19 impact, our client is facing problems with their previous car price valuation machine learning models. So, they are looking for new machine learning models from new data. We have to make car price valuation model.

Review of Literature:

This project is more about exploration, feature engineering and classification that can be done on this data. Since we scrape huge amount of data that includes more car related features, we can do better data exploration and derive some interesting features using the available columns. The goal of this project is to build an application which can predict the car prices with the help of other features. In the long term, this would allow people to better explain and reviewing their purchase with each other in this increasingly digital world.

Importance of Used Car Price Prediction:

The prices of new cars in the industry is fixed by the manufacturer with some additional costs incurred by the Government in the form of taxes. So, customers buying a new car can be assured of the money they invest to be worthy. But due to the increased price of new cars and the incapability of customers to buy new cars due to the lack of funds, used cars sales are on a global increase. There is a need for a used car price prediction system to effectively determine the worthiness of the car using a variety of features. Even though there are websites that offers this service, their prediction method may not be the best. Besides, different models and systems may contribute on predicting power for a used car’s actual market value. It is important to know their actual market value while both buying and selling.

Model building phase:

After collecting the data, you need to build a machine learning model. Before model building do all data pre-processing steps. Try different models with different hyper parameters and select the best model.

➔Follow the complete life cycle of data science. Include all the steps like:

1. Data Cleaning

2.Exploratory Data Analysis

3. Data Pre-processing

4. Model Building

5. Model Evaluation

6. Saving the best model

1. Data Cleaning:

```
In [5]:  #Loading data
         #Loaded .csv file and converted to dataframe.
         df=pd.read_csv("usedcar_data.csv")
         df
```

Out[5]:

	DESCRIPTION	LOCATION	MANUFACTURER	MODEL	YEAR	FUEL TYPE	KMS DRIVEN	PRICE
0	Maruti suzuki omni van	Bengaluru	Maruti Suzuki	Omni	2010	Petrol	24000.0	210000.0
1	Skoda Rapid 1.5 Tdi Cr Ambition (make Year 201...	Bengaluru	Skoda	Rapid	2012	Diesel	53000.0	530000.0
2	Hyundai Santro GL Plus	Bengaluru	Hyundai	Santro Xing	2013	Petrol	25400.0	315000.0
3	Chevrolet Beat Lt Petrol (make Year 2013) (pet...	Bengaluru	Chevrolet	Beat	2013	Petrol	26000.0	365000.0
4	Skoda Laura Elegance 2.0 Tdi Cr At (make Year ...	Bengaluru	Skoda	Laura	2010	Diesel	89000.0	790000.0
...	...	...	...	...	...	...	...	...
20000	Skoda Rapid Ambition 1.6 Tdi Cr Mt Plus (make ...	Delhi	Skoda	Rapid	2013	Diesel	58000.0	450000.0
20001	Renault Duster 110 Ps Rxz Diesel Plus (make Ye...	Delhi	Renault	Duster	2014	Diesel	50000.0	875000.0
20002	Mahindra Scorpio S10 (make Year 2015) (diesel)	Delhi	Mahindra	Scorpio	2015	Diesel	12000.0	1365000.0
20003	Ford Figo (make Year 2011) (diesel)	Delhi	Ford	Figo	2011	Diesel	36000.0	275000.0
20004	Maruti Suzuki Swift Lxi 1.2 Bs-iv (make Year 2...	Delhi	Maruti Suzuki	Swift	2011	Petr	NaN	NaN

20005 rows × 8 columns

```
In [6]:  # Name of the columns
         df.columns

Out[6]: Index(['DESCRIPTION', 'LOCATION', 'MANUFACTURER', 'MODEL', 'YEAR', 'FUEL TYPE',
               'KMS DRIVEN', 'PRICE'],
              dtype='object')
```

Our Target Variable is the selling price of used cars. Checking the data type of each column and information of the database.

Here we can see that we have 20005 rows and 8 columns in our datasets.

Also we can see the columns name in our data set.

```
In [8]:  #checking the datatype of each column
         print(df.dtypes)

DESCRIPTION      object
LOCATION         object
MANUFACTURER     object
MODEL            object
YEAR              int64
FUEL TYPE        object
KMS DRIVEN      float64
PRICE           float64
dtype: object
```

```
In [9]:  #Information of the database
         df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20005 entries, 0 to 20004
Data columns (total 8 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   DESCRIPTION   20005 non-null  object
 1   LOCATION      20005 non-null  object
 2   MANUFACTURER  20005 non-null  object
 3   MODEL         20004 non-null  object
 4   YEAR          20005 non-null  int64
 5   FUEL TYPE     20005 non-null  object
 6   KMS DRIVEN    20004 non-null  float64
```

- ✓ Data has been scrapped from cardekho website so we have to clean it for our convenience.
- ✓ In my datasets I found null values, outliers and also skewness.
- ✓ I have used imputation method to replace null values. To remove outliers I have used Z-score method. And to remove skewness I have used yeo-johnson method.
- ✓ To encode the categorical columns I have use Label Encoding.
- ✓ Use of Pearson’s correlation coefficient to check the correlation between dependent and independent features.
- ✓ Also I have used standardization. Then followed by model building with all regression algorithms.

2. Exploratory Data Analysis:

EDA (Exploratory Data Analysis)

In [11]:
```
#finding null values in the database
df.isnull().sum()
```

Out[11]:
```
DESCRIPTION      0
LOCATION         0
MANUFACTURER     0
MODEL            1
YEAR             0
FUEL TYPE        0
KMS DRIVEN       1
PRICE            1
dtype: int64
```

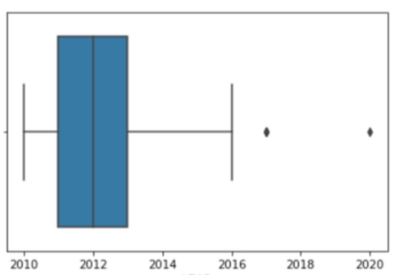
We can see only 3 null values in the entire dataset, Which is negligible.

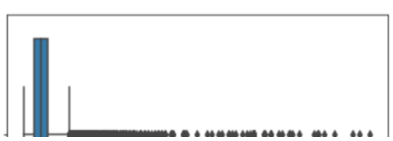
In [12]:
```
#Making Heatmap of null values
sns.heatmap(df.isnull())
```

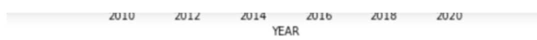
Out[12]: <AxesSubplot:>

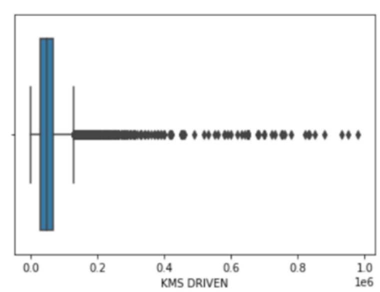
CHECKING FOR OUTLIERS

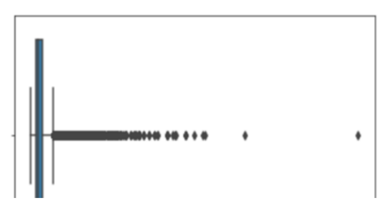
In [21]:
```
# Cheacking whether the columns has outliers or not
for i in df.describe().columns:
    sns.boxplot(df[i])
    plt.show()
```









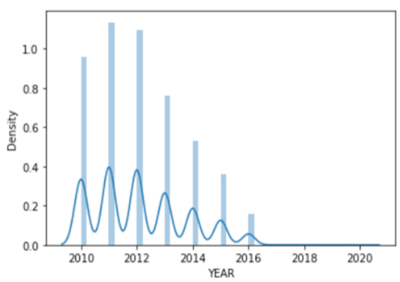


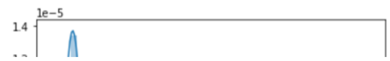
From below observation we can say, fewer outliers are present in the dataset.

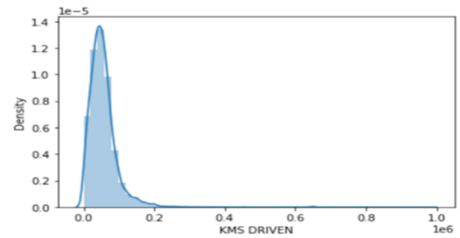
CHECKING FOR SKEWNESS

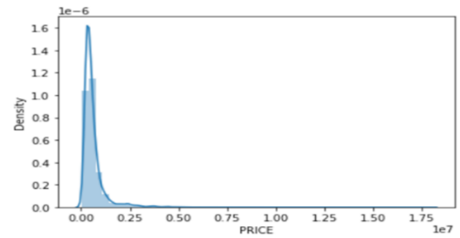
[22]:
```
#checking wheather the columns are normally distributed or not

for i in df.describe().columns:
    sns.distplot(df[i])
    plt.show()
```

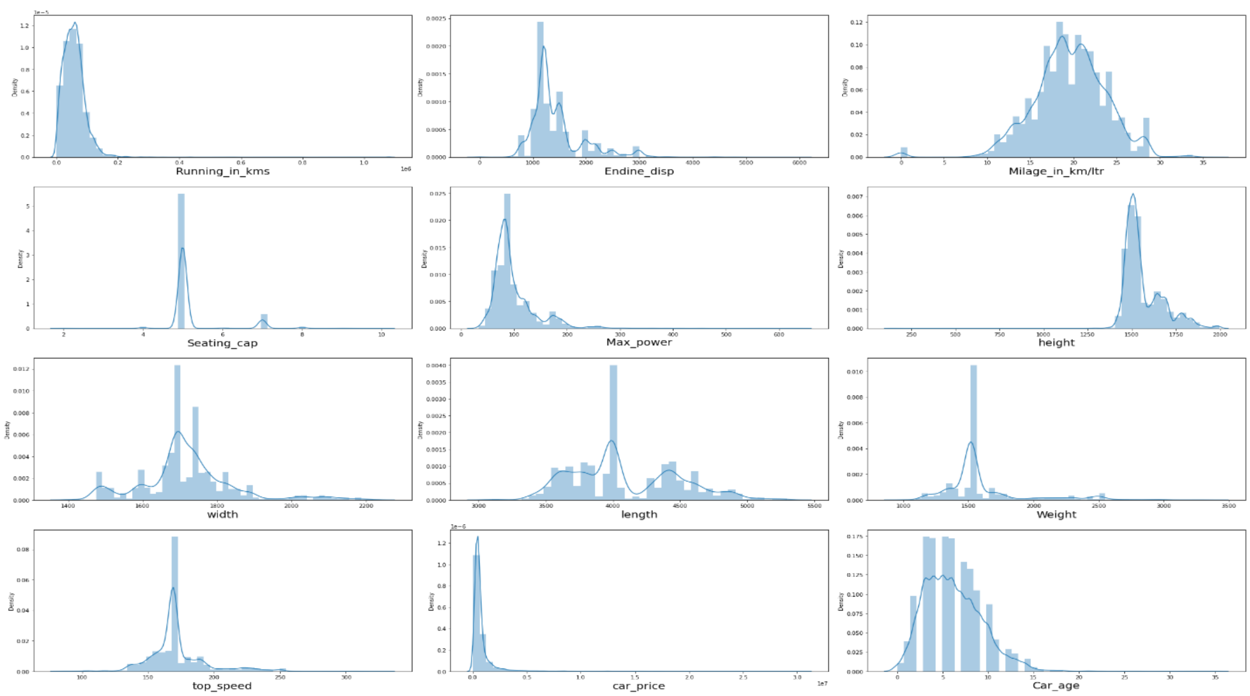




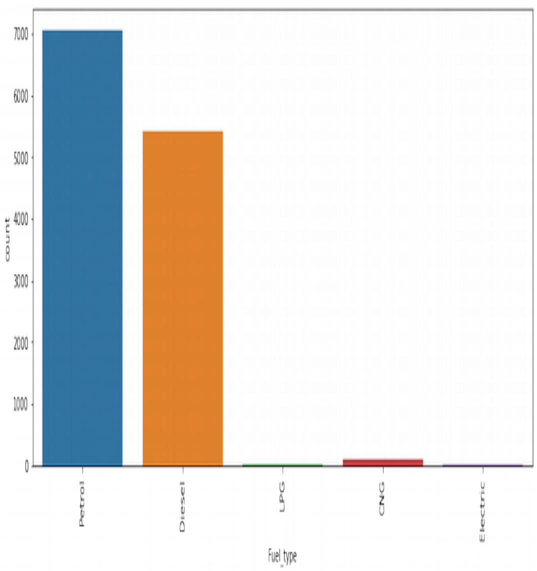


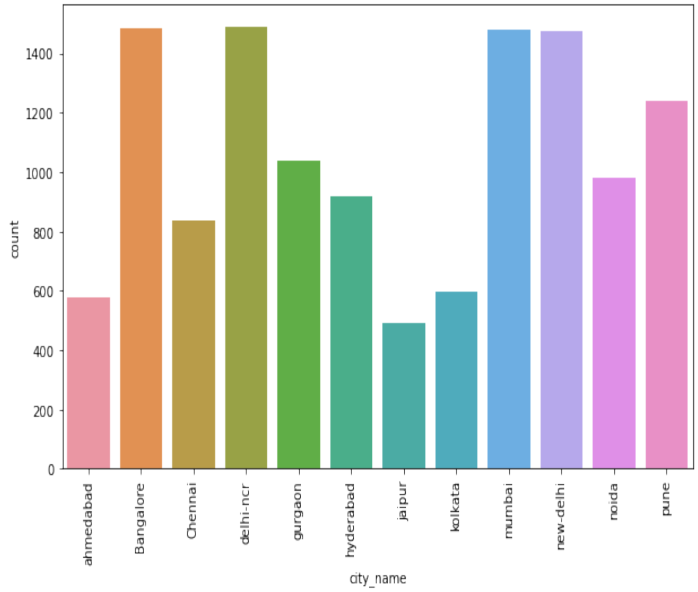


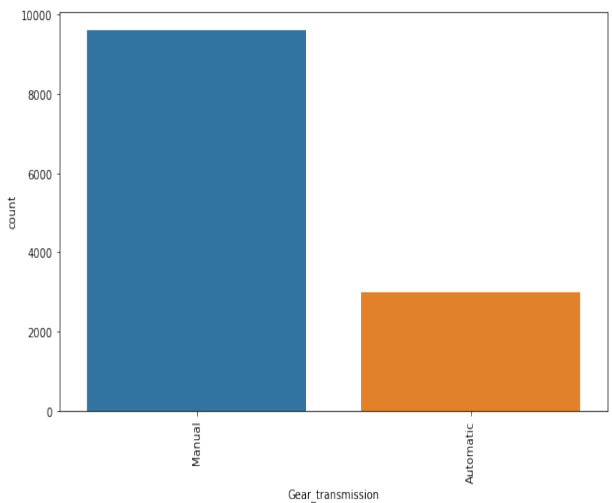
Year has positive correlation with Price. Kms Driven has negative correlation over Price.

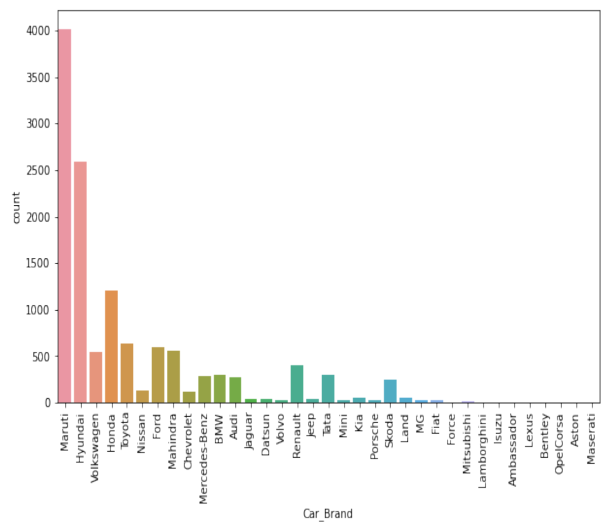


Univariate Visualization of Categorical columns:

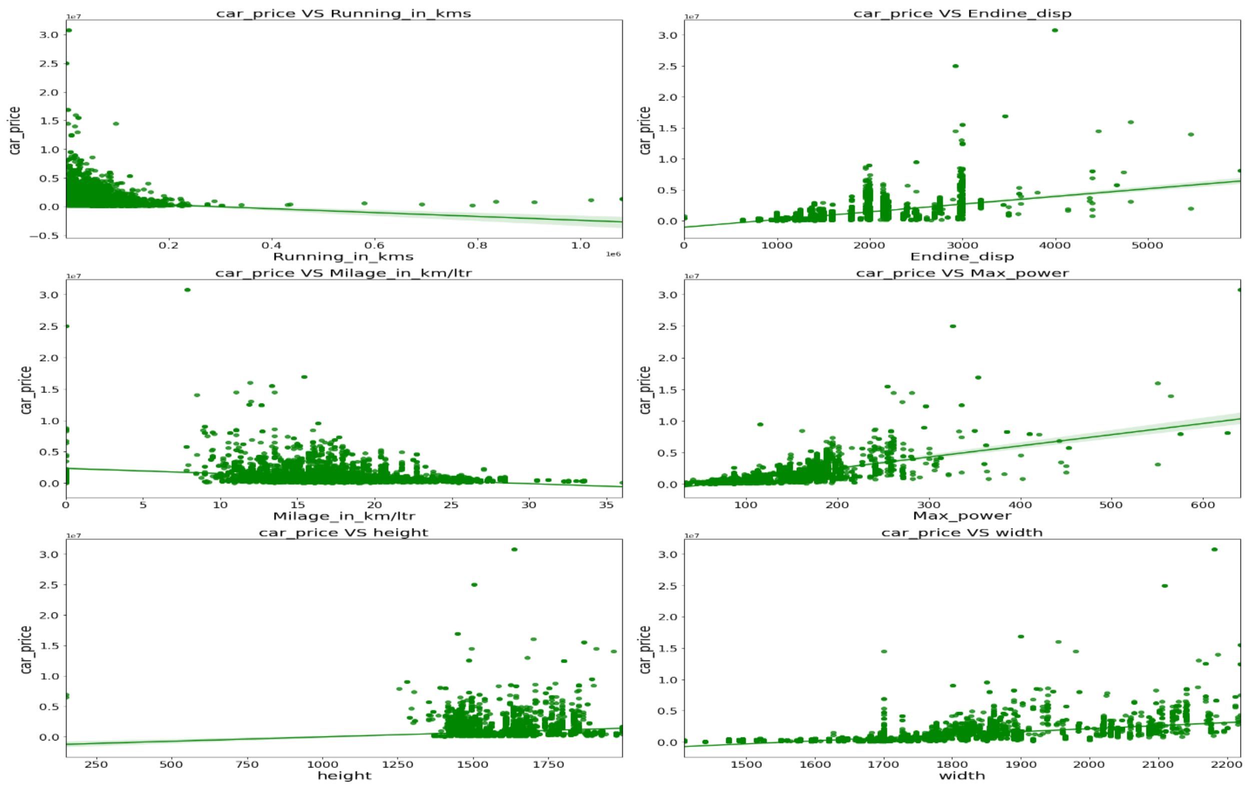




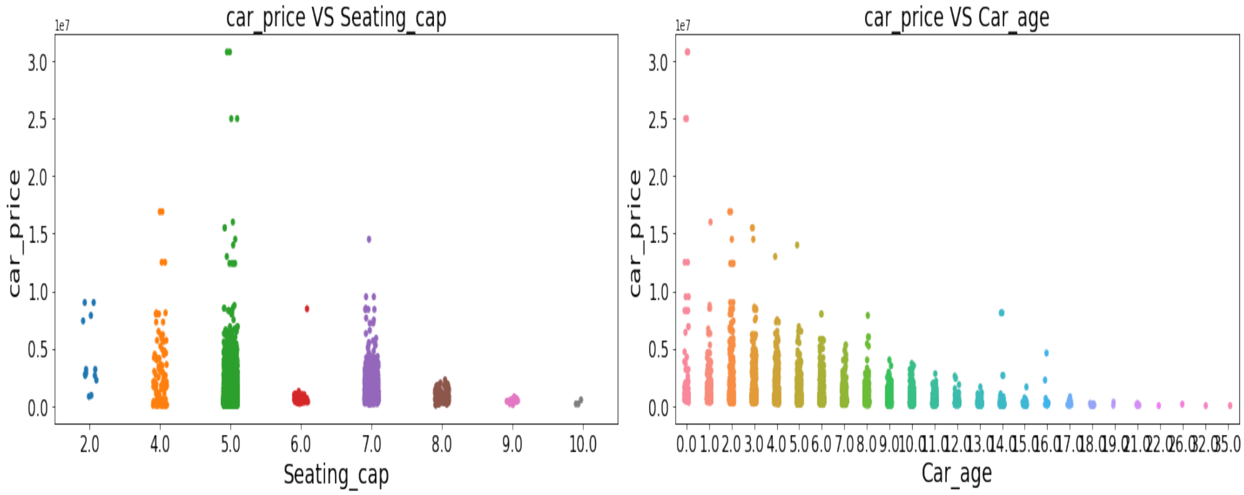


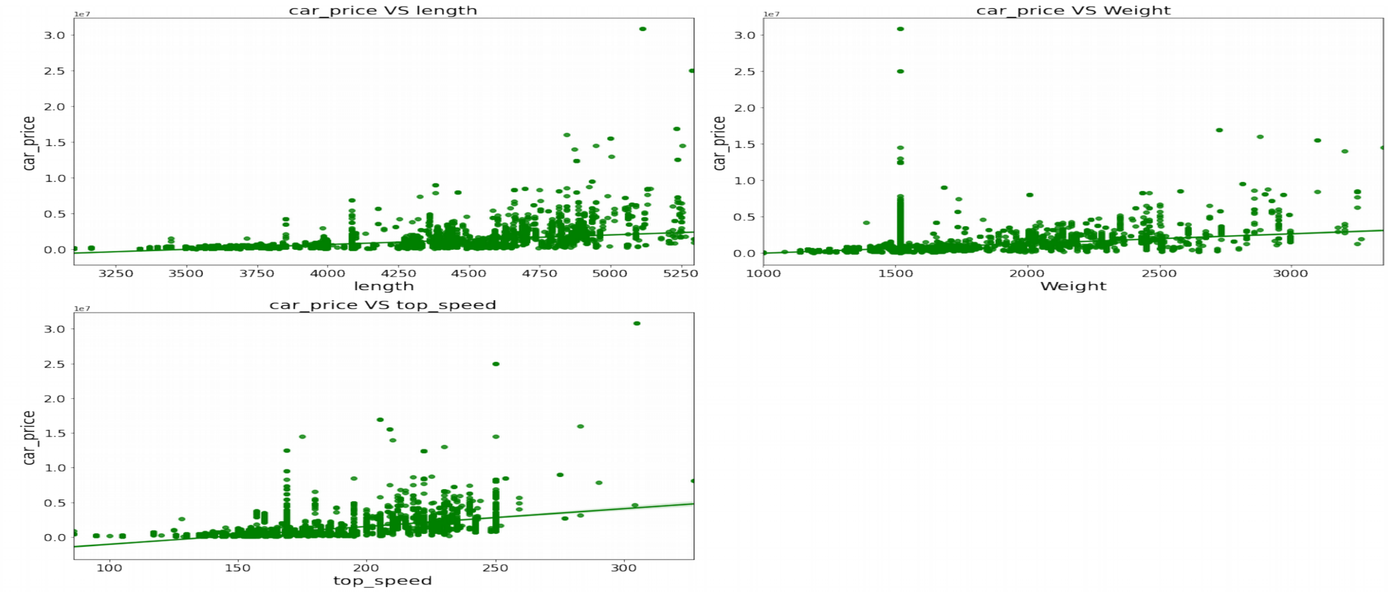


Bivariate Visualization of numerical columns:

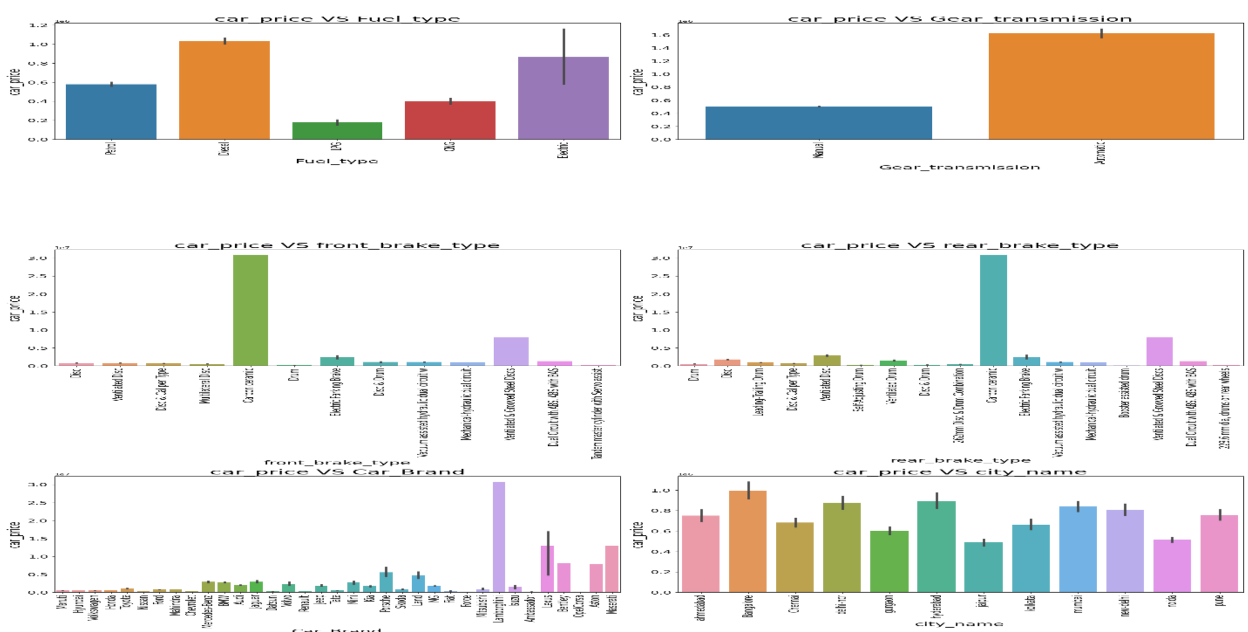


Bivariate Visualization of numerical columns:





Bivariate Vizualization of categorical columns:



Hardware and Software Requirements and Tools Used:

For doing this project, the hardware used is a laptop with high end specification and a stable internet connection. While coming to software part, I had used anaconda navigator and in that I have used Jupyter notebook to do my python programming and analysis. For using an CSV file, Microsoft excel is needed. In Jupyter notebook, I had used lots of python libraries to carry out this project and I have mentioned below with proper justification.

Model Building:

- ✓ Since Car Price was my target and it was a continuous column so this perticular problem was regression problem. And I have used all regression algorithms to build my model. By looking into the difference of r2 score and cross validation score I found DecisionTreeRegressor as a best model with least difference. Also to get the best model we have to run through multiple models and to avoid the confusion of overfitting we have go through cross validation. Below are the list of regression algorithms I have used in my project.

- ➢ RandomForestRegressor
- ➢ XGBRegressor
- ➢ GradientBoostingRegressor
- ➢ DecisionTreeRegressor
- ➢ BaggingRegressor

1) RandomForestRegressor:

```
In [104]: RFR=RandomForestRegressor()
          RFR.fit(X_train,y_train)
          pred=RFR.predict(X_test)
          R2_score = r2_score(y_test,pred)*100
          print('R2_score:',R2_score)
          print('mean_squared_error:',metrics.mean_squared_error(y_test,pred))
          print('mean_absolute_error:',metrics.mean_absolute_error(y_test,pred))
          print('root_mean_squared_error:',np.sqrt(metrics.mean_squared_error(y_test,pred)))

          #cross validation score
          scores = cross_val_score(RFR, X, y, cv = 10).mean()*100
          print("\nCross validation score :", scores)

          #difference of accuracy and cv score
          diff = R2_score - scores
          print("\nR2_Score - Cross Validation Score :", diff)
```

```
R2_score: 96.46493782044703
mean_squared_error: 9226345022.16905
mean_absolute_error: 51153.49883627064
root_mean_squared_error: 96053.86521202076

Cross validation score : 93.03122692981853

R2_Score - Cross Validation Score : 3.433710890628504
```

- RandomForestRegressor has given me 96.46% r2_score and the difference between r2_score and cross validation score is 3.43%, but still we have to look into multiple models.

2) GradientBoostingRegressor:

```
In [107]: GBR=GradientBoostingRegressor()
GBR.fit(X_train,y_train)
pred=GBR.predict(X_test)
R2_score = r2_score(y_test,pred)*100
print('R2_score:',R2_score)
print('mean_squared_error:',metrics.mean_squared_error(y_test,pred))
print('mean_absolute_error:',metrics.mean_absolute_error(y_test,pred))
print('root_mean_squared_error:',np.sqrt(metrics.mean_squared_error(y_test,pred)))

#cross validation score
scores = cross_val_score(GBR, X, y, cv = 10).mean()*100
print("\nCross validation score :", scores)

#difference of accuracy and cv score
diff = R2_score - scores
print("\nR2_Score - Cross Validation Score :", diff)
```

```
R2_score: 94.9328623763045
mean_squared_error: 13224989439.06563
mean_absolute_error: 71240.04884627696
root_mean_squared_error: 114999.95408288487

Cross validation score : 90.1937305617025

R2_Score - Cross Validation Score : 4.739131814602004
```

- GradientBoosting Regressor is giving me 90.193% r2_score and the difference between r2_score and cross validation score is 4.74%.

3) XGBRegressor:

```
In [105]: XGB=XGBRegressor()
XGB.fit(X_train,y_train)
pred=XGB.predict(X_test)
R2_score = r2_score(y_test,pred)*100
print('R2_score:',R2_score)
print('mean_squared_error:',metrics.mean_squared_error(y_test,pred))
print('mean_absolute_error:',metrics.mean_absolute_error(y_test,pred))
print('root_mean_squared_error:',np.sqrt(metrics.mean_squared_error(y_test,pred)))

#cross validation score
scores = cross_val_score(XGB, X, y, cv = 10).mean()*100
print("\nCross validation score :", scores)

#difference of accuracy and cv score
diff = R2_score - scores
print("\nR2_Score - Cross Validation Score :", diff)
```

```
R2_score: 96.8578288022264
mean_squared_error: 8200918149.917081
mean_absolute_error: 50118.93210268505
root_mean_squared_error: 90558.92087429643

Cross validation score : 93.2469040953667

R2_Score - Cross Validation Score : 3.6109247068596915
```

✓ XGBRegressor is giving me 93.24% r2_score and the difference between r2_score and cross validation score is 3.61%.

4) **DecisionTreeRegressor**:

```
In [108]: DTR=DecisionTreeRegressor()
DTR.fit(X_train,y_train)
pred=DTR.predict(X_test)
R2_score = r2_score(y_test,pred)*100
print('R2_score:',R2_score)
print('mean_squared_error:',metrics.mean_squared_error(y_test,pred))
print('mean_absolute_error:',metrics.mean_absolute_error(y_test,pred))
print('root_mean_squared_error:',np.sqrt(metrics.mean_squared_error(y_test,pred)))

#cross validation score
scores = cross_val_score(DTR, X, y, cv = 10).mean()*100
print("\nCross validation score :", scores)

#difference of accuracy and cv score
diff = R2_score - scores
print("\nR2_Score - Cross Validation Score :", diff)
```

```
R2_score: 91.79408304824462
mean_squared_error: 21417054969.521046
mean_absolute_error: 64770.82728592162
root_mean_squared_error: 146345.6694594037

Cross validation score : 88.83907795864332

R2_Score - Cross Validation Score : 2.9550050896013005
```

- DecisionTreeRegressor is giving me 88.83% r2_score and the difference between r2_score and cross validation score is 2.96%.

Hyper Parameter Tunning:

```
In [110]: #importing necessary libraries
from sklearn.model_selection import GridSearchCV
```

```
In [111]: parameter = {'criterion':['squared_error', 'friedman_mse', 'absolute_error', 'poisson'],
              'splitter':['best','random'],
              'max_features':['auto','sqrt','log2'],
              'min_samples_split':[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15],
              'max_depth':[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15]}
```

Giving DecisionTreeRegressor parameters.

```
In [112]: GCV=GridSearchCV(DecisionTreeRegressor(),parameter,cv=10)
```

Running grid search CV for ExtraTreesRegressor.

```
In [113]: DecisionTreeRegressorGCV.fit(X_train,y_train)
```

```
Out[113]: GridSearchCV(cv=10, estimator=DecisionTreeRegressor(),
             param_grid={'criterion': ['squared_error', 'friedman_mse',
                                      'absolute_error', 'poisson'],
                         'max_depth': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,
                                      13, 14, 15],
                         'max_features': ['auto', 'sqrt', 'log2'],
                         'min_samples_split': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
                                              11, 12, 13, 14, 15],
                         'splitter': ['best', 'random']})
```

```
In [114]: GCV.best_params_

Out[114]: {'criterion': 'friedman_mse',
 'max_depth': 13,
 'max_features': 'auto',
 'min_samples_split': 4,
 'splitter': 'random'}
```

Got the best parameters for DecisionTreeRegressor.

```
In [115]: Best_mod=DecisionTreeRegressor(criterion='friedman_mse',max_depth=15,max_features='auto',min_samples_split=4,splitter='random')
Best_mod.fit(X_train,y_train)
pred=Best_mod.predict(X_test)
print('R2_Score:',r2_score(y_test,pred)*100)
print('mean_squared_error:',metrics.mean_squared_error(y_test,pred))
print('mean_absolute_error:',metrics.mean_absolute_error(y_test,pred))
print("RMSE value:",np.sqrt(metrics.mean_squared_error(y_test, pred)))

R2_Score: 92.28906442588051
mean_squared_error: 20125176994.634956
mean_absolute_error: 70467.88619495885
RMSE value: 141863.2334138587
```

- I have choosed all parameters of DecisionTreeRegressor, after tunning the model with best parameters I have incresed my model accuracy from 91.79% to 92.29%.

Saving the model and predictions using saved model:

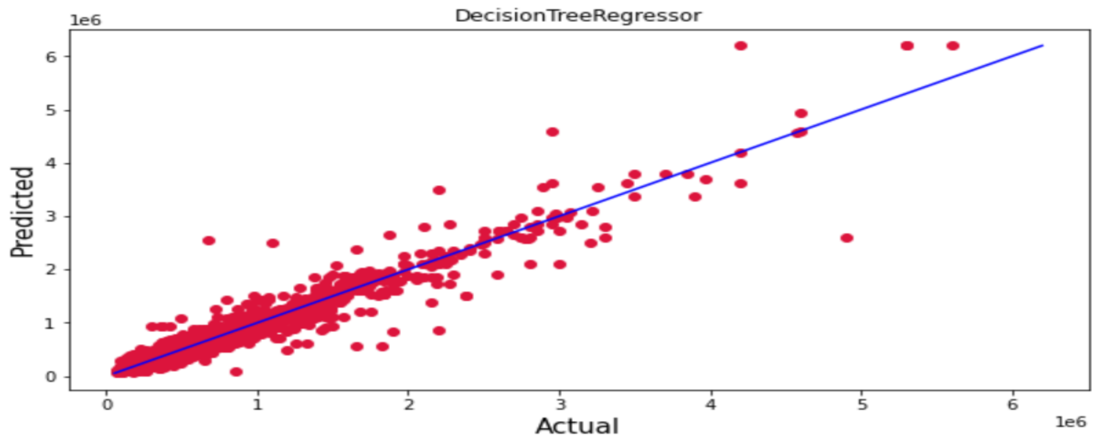
Saving The Best Model

```
[102]: import pickle
# save the model to disk
filename = 'finalized_model_ls.pkl'
pickle.dump(ls,open(filename,'wb'))
#Load the model from disk
loaded_model= pickle.load(open(filename,'rb'))
loaded_model.predict(x_test)

Out[102]: array([666.45951927, 675.96851202, 659.89825027, ..., 653.67304576,
       508.09058153, 688.08974107])
```

Plotting the predicted values v/s actual values:

In [119]:
```
plt.figure(figsize=(10,5))
plt.scatter(y_test, prediction, c='crimson')
p1 = max(max(prediction), max(y_test))
p2 = min(min(prediction), min(y_test))
plt.plot([p1, p2], [p1, p2], 'b-')
plt.xlabel('Actual', fontsize=15)
plt.ylabel('Predicted', fontsize=15)
plt.title("DecisionTreeRegressor")
plt.show()
```



Conclusion:

- To conclude, the application of machine learning in car price prediction is still at an early stage. We hope this study has moved a small step ahead in providing some methodological and empirical contributions to 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.

Limitations of this work and Scope for Future Work:

First drawback is scrapping the data as it is fluctuating process.

✓ Followed by more number of outliers and skewness these two will reduce our model accuracy.

✓ Also, we have tried best to deal with outliers, skewness and null values. So it looks quite good that we have achieved a accuracy of 92.29% even after dealing all these drawbacks.