Project Presentation On

"Used Car Price Prediction"

Presented By: Wesley sirra

Agenda:

- Overview.
- Problem Statement.
- Problem Understanding.
- What is Used Car Price Prediction?
- Importance of Used Car price prediction.
- Exploratory data analysis.
- Visualizations.
- Analysis.
- Data cleaning steps.
- Model Building.
- Hyper Parameter Tunning.
- Saving the model and predictions from saved best model.
- Conclusion.

Overview:

- ✔ In this particular presentation we will be looking on:
  - How to analyze the dataset of Used Car Price Prediction.
  - What are the EDA steps in cleaning the dataset.
  - Overall analysis on the problem.
  - Model building from cleaned dataset.
  - Predicting Used Car Price for saved best model.

Problem Statement:

- 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. This project contains two phase

- Data Collection Phase: You have to scrape at least 5000 used cars data. You can scrape more data as well, it's up to you. more the data better the model In this section You need to scrape the data of used cars from websites (Olx, cardekho, Cars24 etc.) You need web scraping for this. You have to fetch data for different locations. The number of columns for data doesn't have limit, it's up to you and your creativity. Generally, these columns are Brand, model, variant, manufacturing year, driven kilometers, fuel, number of owners, location and at last target variable Price of the car. Try to include all types of cars in your data for example- SUV, Sedans, Coupe, minivan, Hatchback.

- 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

Problem Understanding:

- There are lots of individuals who are interested in the used car market at some points in their life because they wanted to sell their car or buy a used car. In this process, it's a big corner to pay too much or sell less than it's market value.

- There are one of the biggest target group that can be interested in results of this study. If used car sellers better understand what makes a car desirable, what are the important features for a used car, then they may consider this knowledge and offer a better service.

What is Used Car Price Prediction?

- ✔ The increased prices of new cars and the financial incapability of the customers to buy them, Used Car sales are on a **global increase**. Therefore, there is an urgent need for a Used Car Price Prediction system which effectively determines the worthiness of the car using a variety of features.

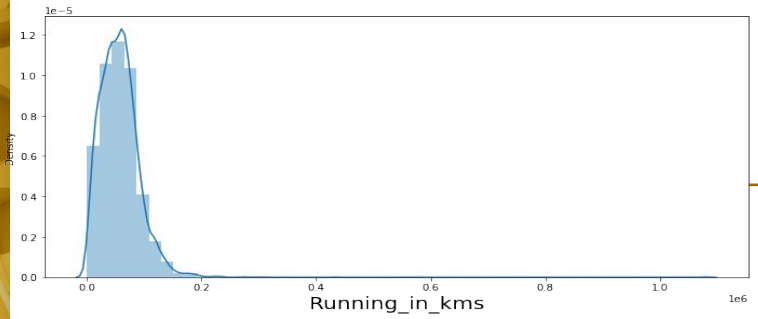
Importance of Used Car Price Prediction.

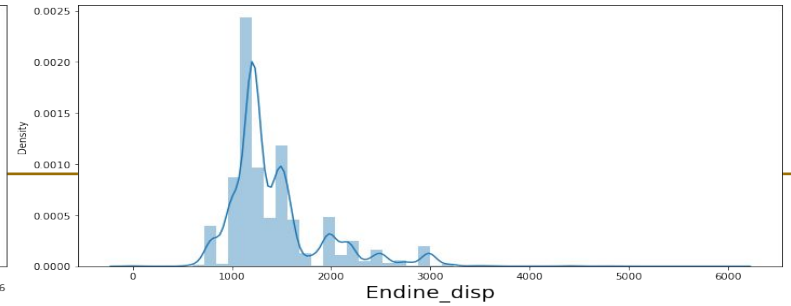
- The prices of new cars in the industry is fixed by the manufacturer with some additional costs incurred by the Government in the form of taxes. So, customers buying a new car can be assured of the money they invest to be worthy. But due to the increased price of new cars and the incapability of customers 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.

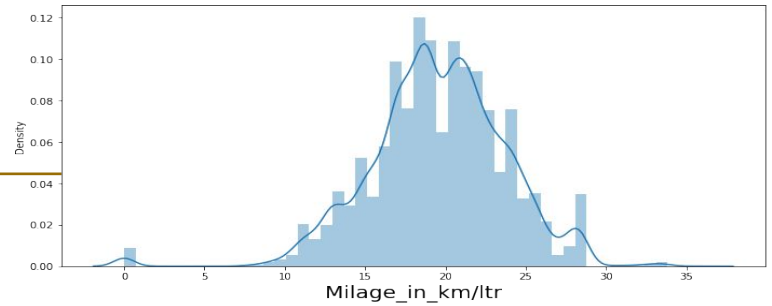
Exploratory Data Analysis:

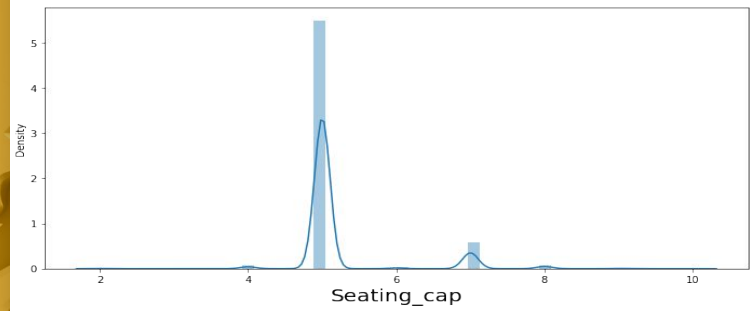
- ✔ As a first step I have scrapped all the required information from cardekho website.
- ✔ I have imported required libraries and I have imported the dataset which was in excel format.
- ✔ Then I did all the statistical analysis like checking shape, nunique, value counts, info etc…..
- ✔ While checking for null values I found null values in the dataset and I replaced them using imputation technique.
- ✔ I have also droped Unnamed:0, cargo_volume and Insp_score column as I found they are useless.
- ✔ Next as a part of feature extraction I converted the data types of all the columns and I have extracted usefull information from the raw dataset. Thinking that this data will help us more than raw data.

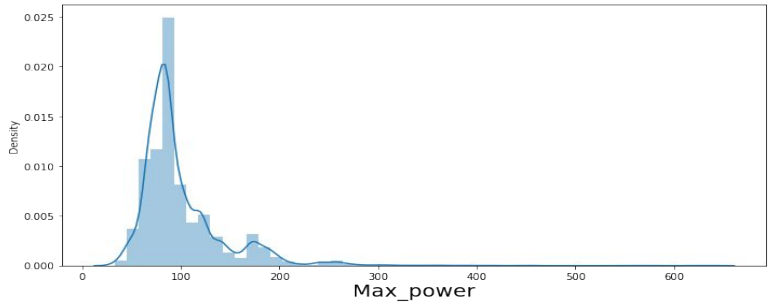
Univariate Visualization of numerical columns:

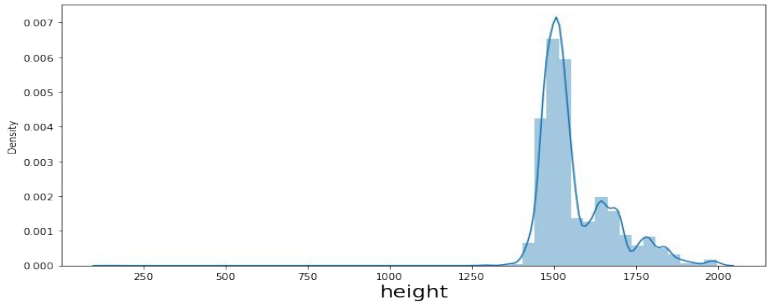


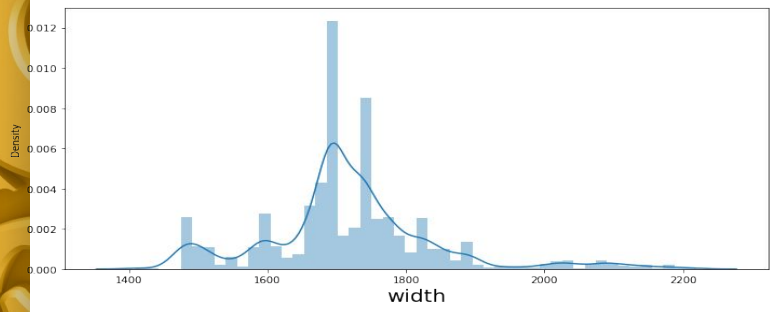


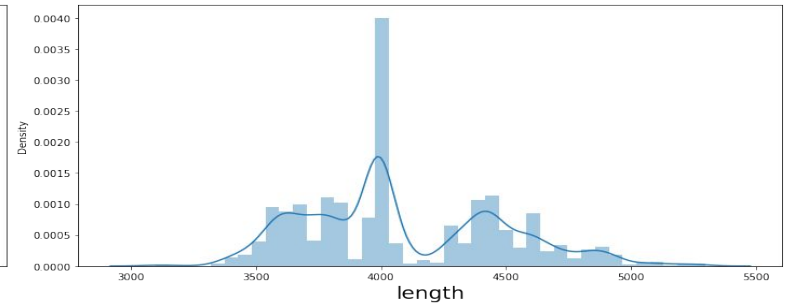


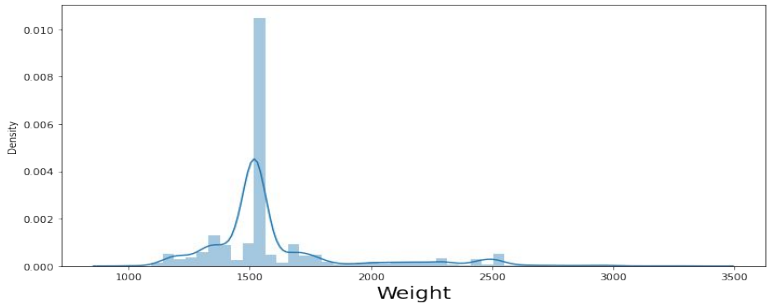


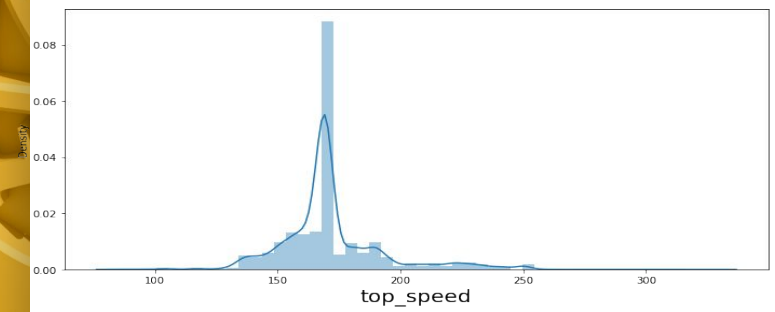


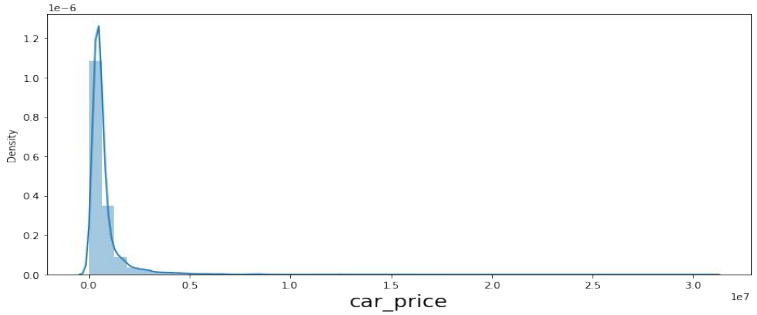


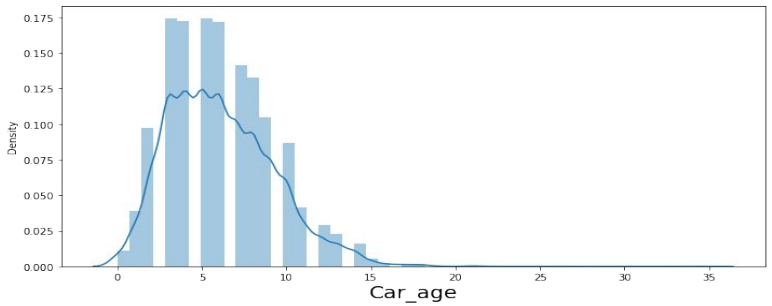




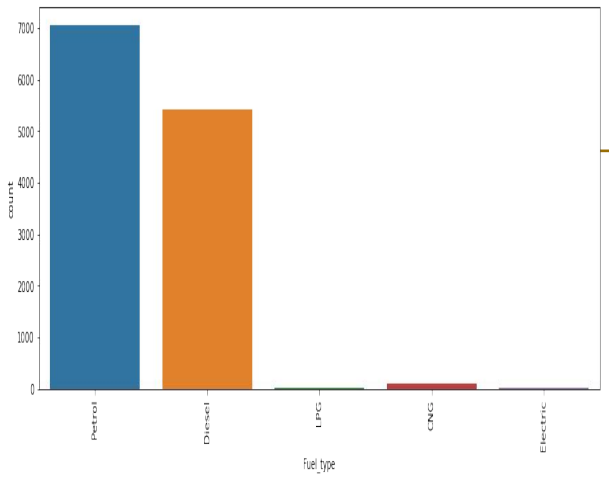


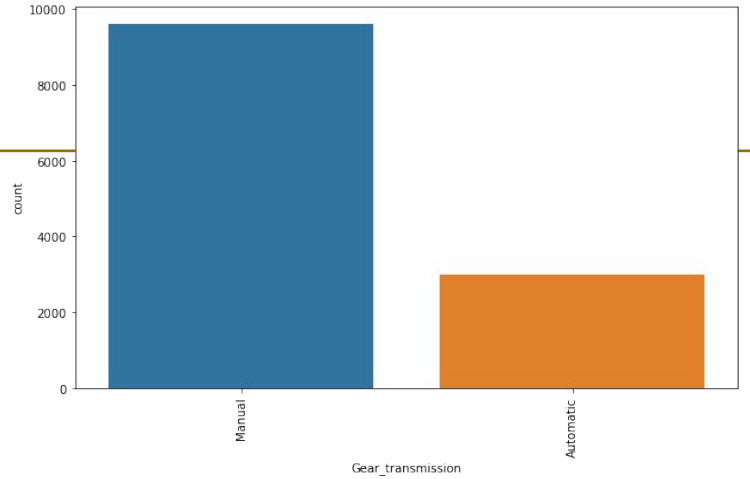


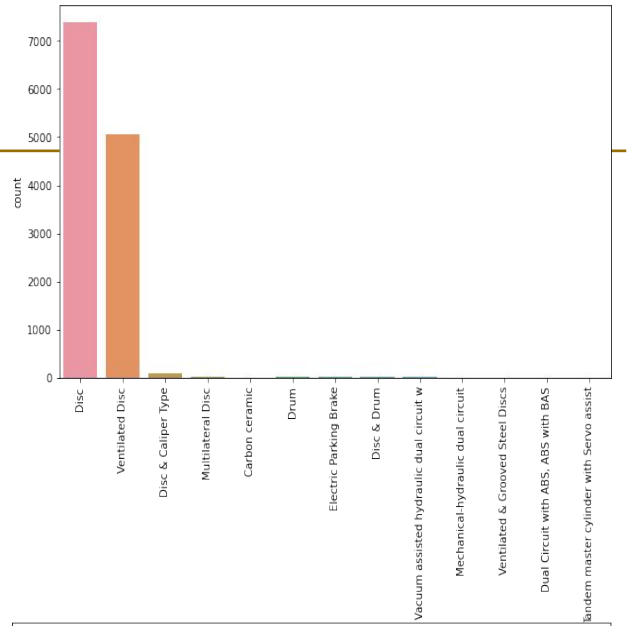


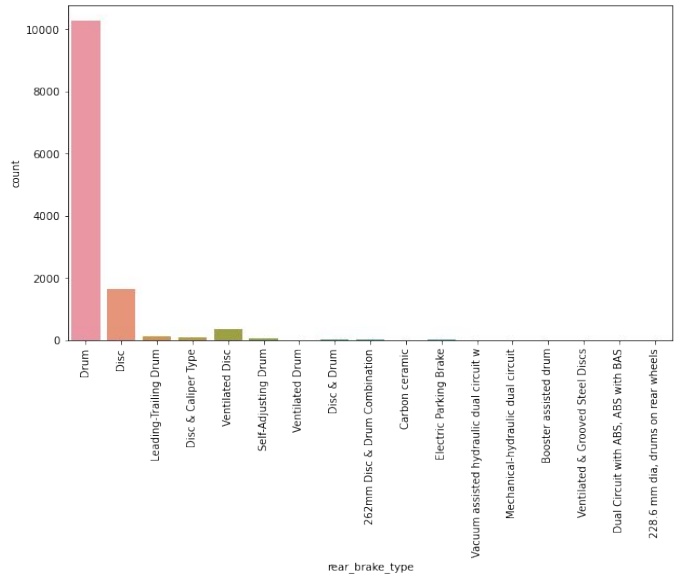


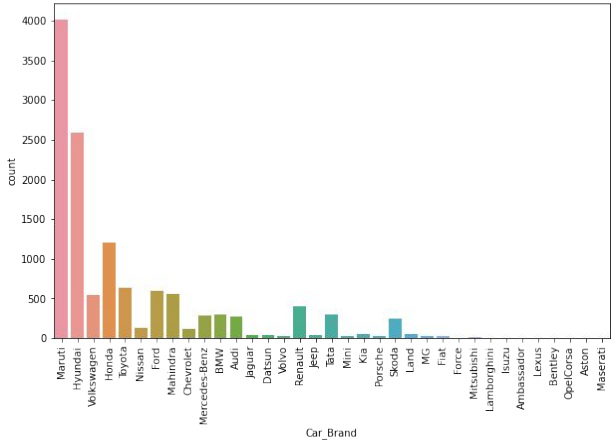
Univariate Vizualization of Categorical columns:

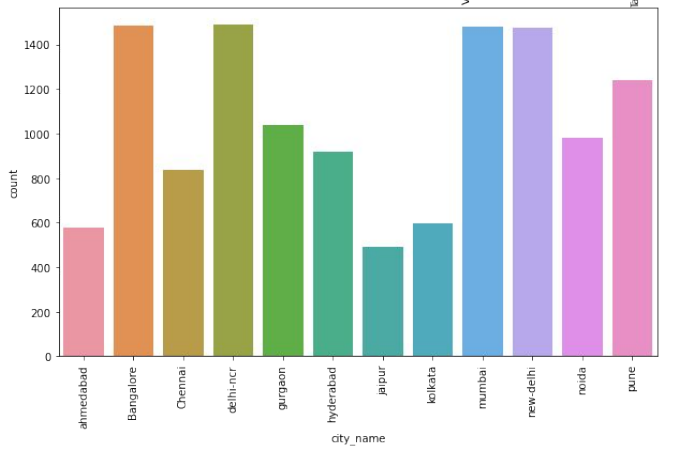












Observations:

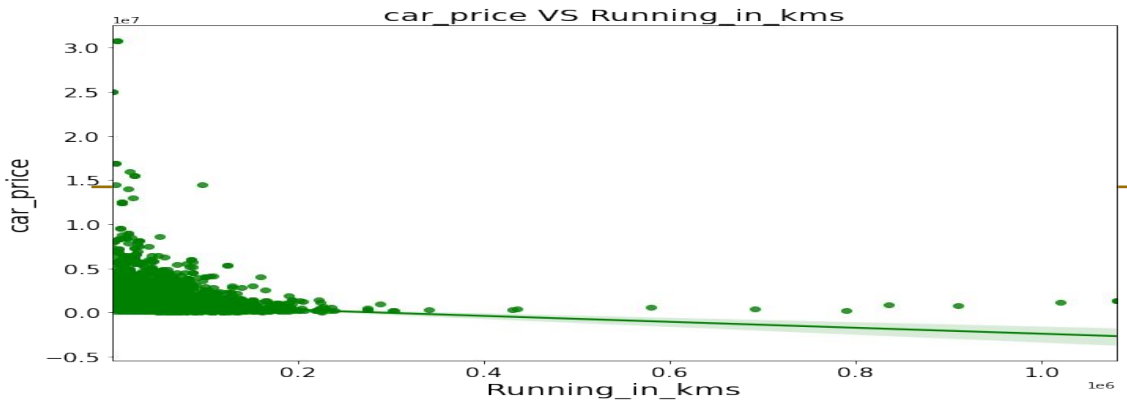
Univariate numerical columns:

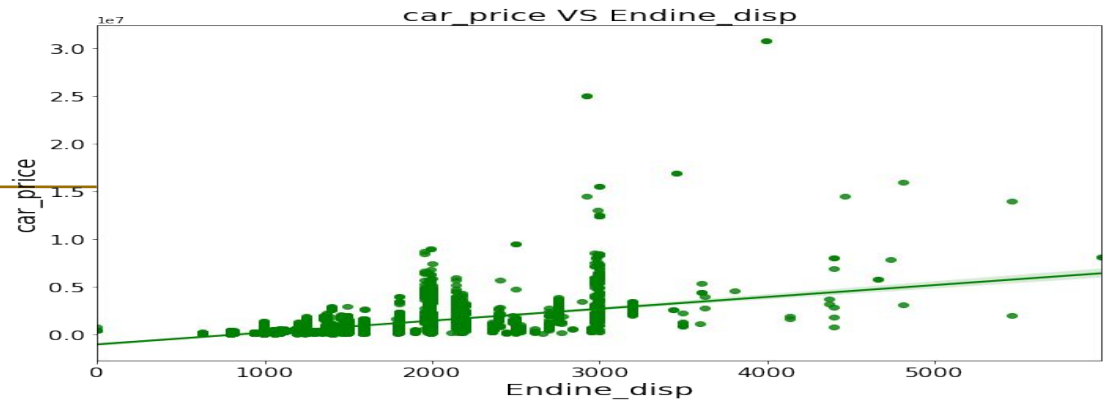
- ✔ We can clearly see that there is skewness in most of the columns so we have to treat them using suitable methods.

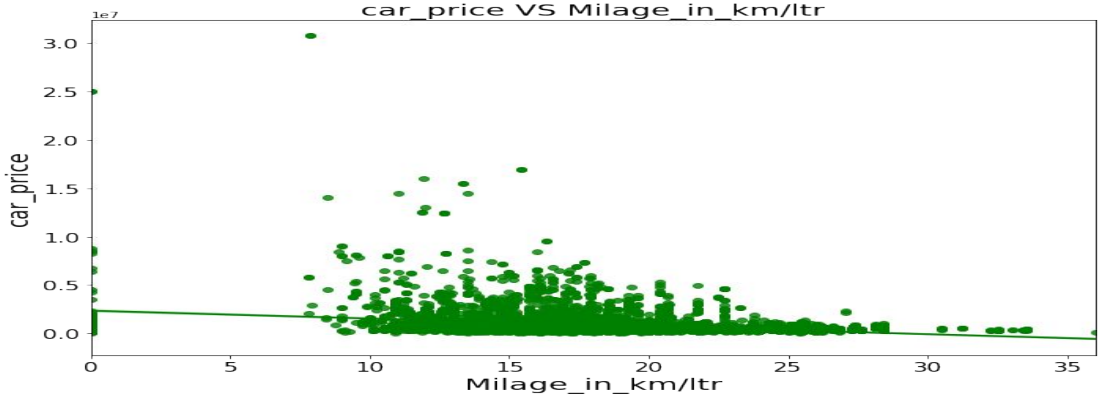
Univariate categorical columns:

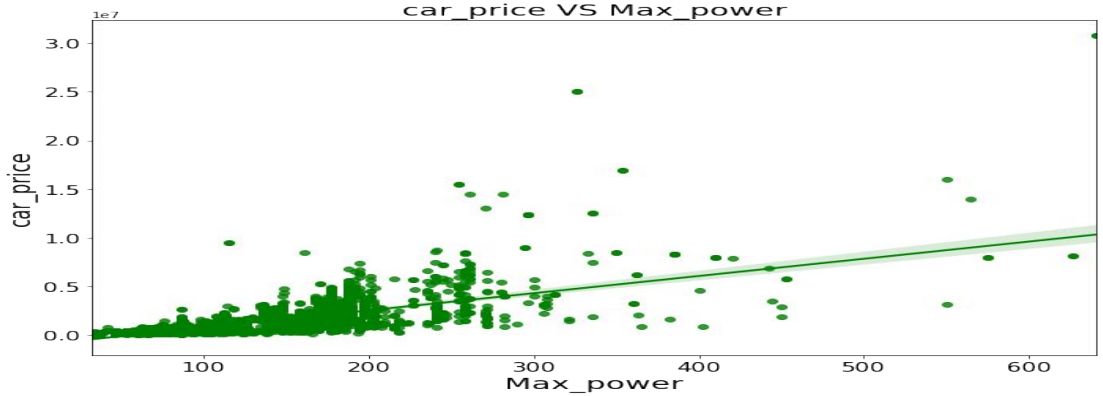
- ✔ Maximum cars are petrol driven and also diesel driven.
- ✔ Maximum cars are with Manual gear transmission.
- ✔ Disc front brake cars are more in number followed by Ventilated Disc.
- ✔ Drum rare break cars are more in number.
- ✔ Maximum cars under sale are Maruti followed by Hyundai.
- ✔ In Bangalore, delhi-ncr, mumbai and new-delhi we can find maximum cars for sale. Since these are most populated places.

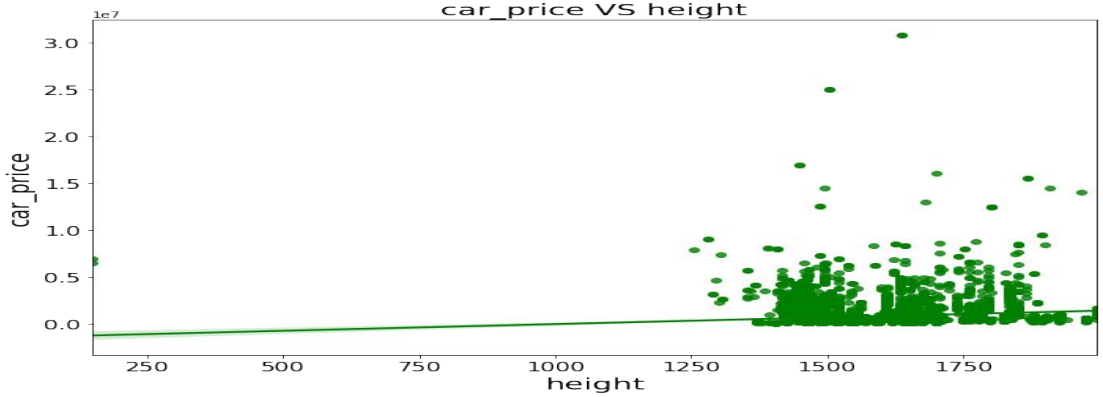
Bivariate Vizualization of numerical columns:

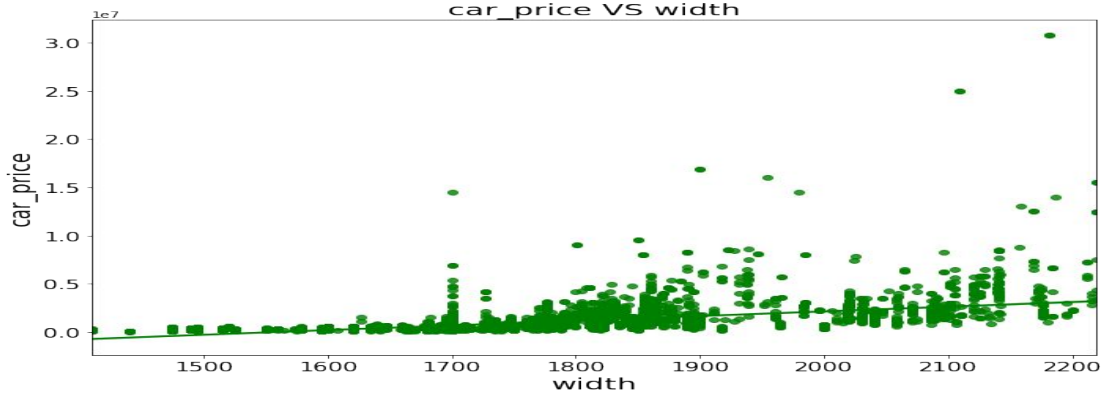




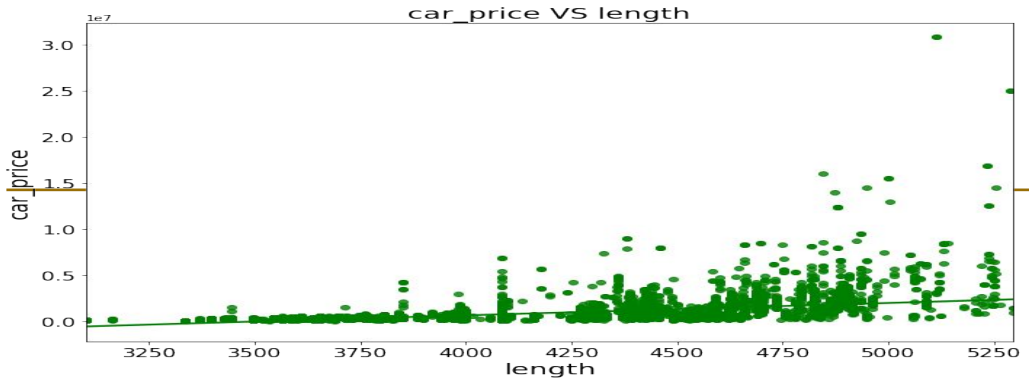


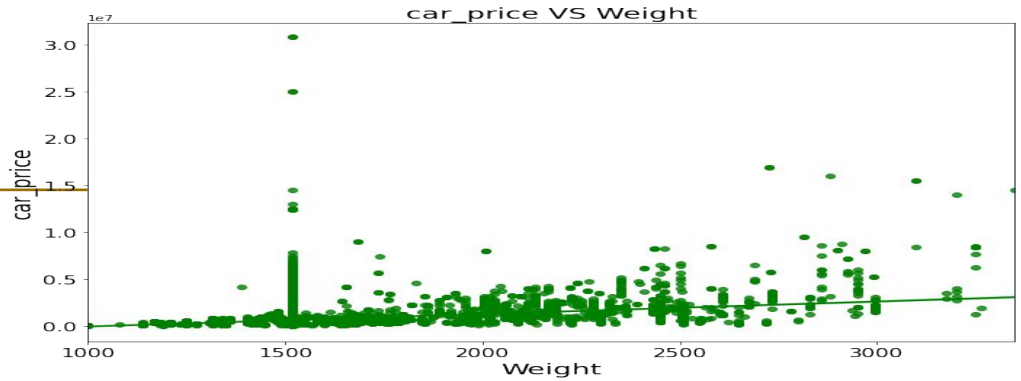


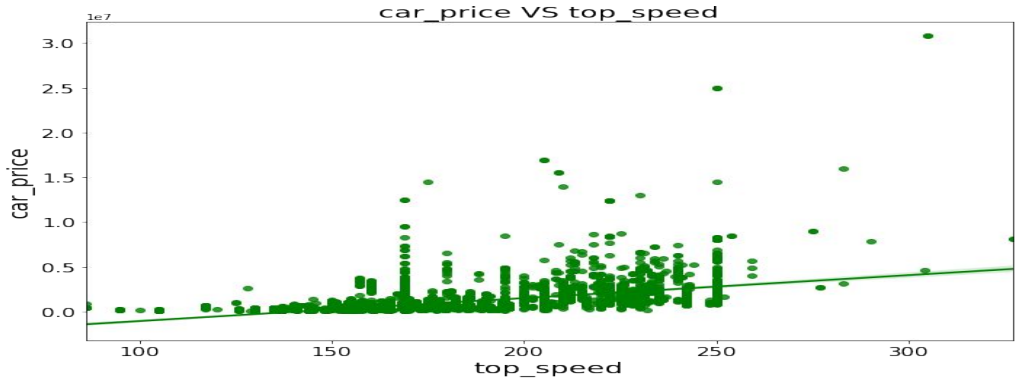


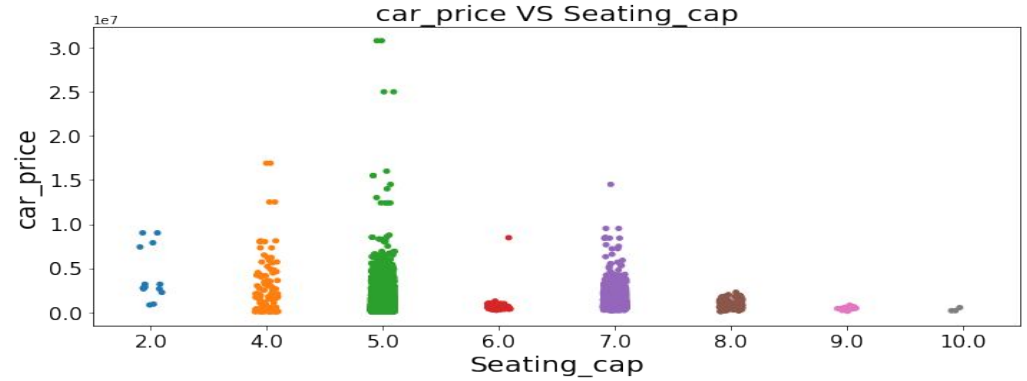


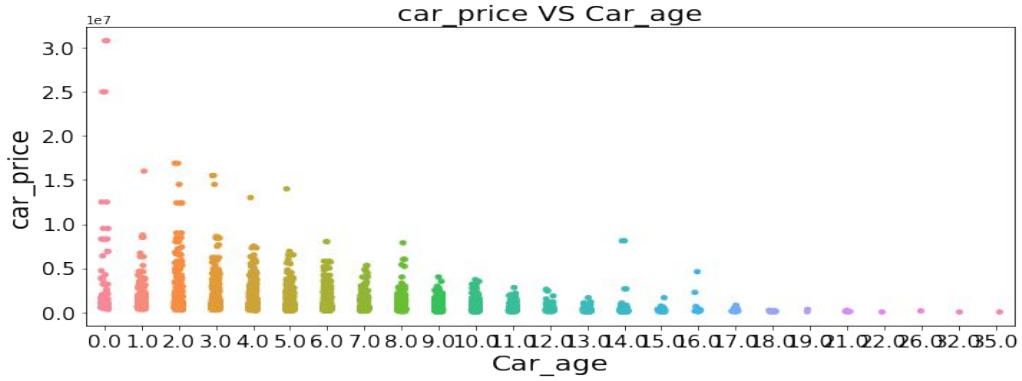
Bivariate Vizualization of numerical columns:







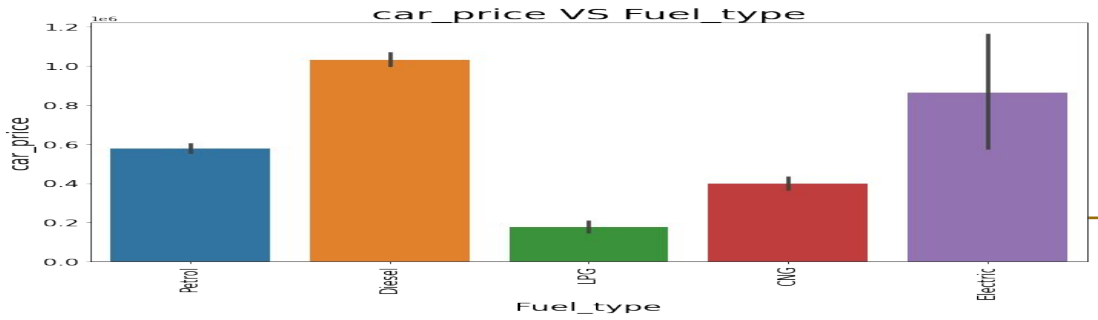


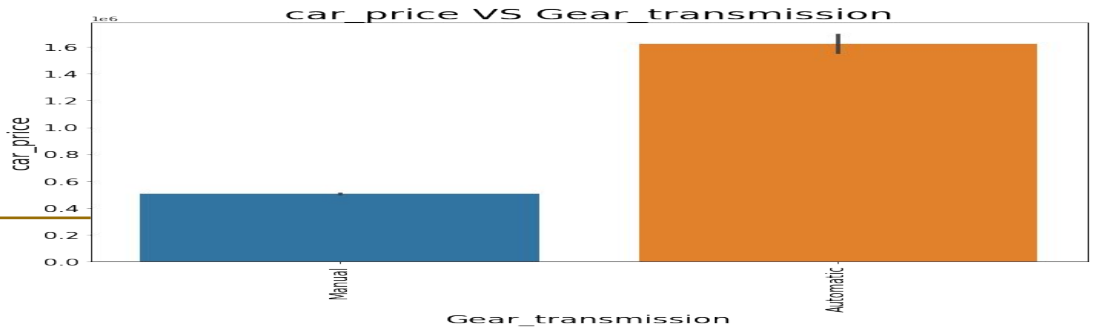


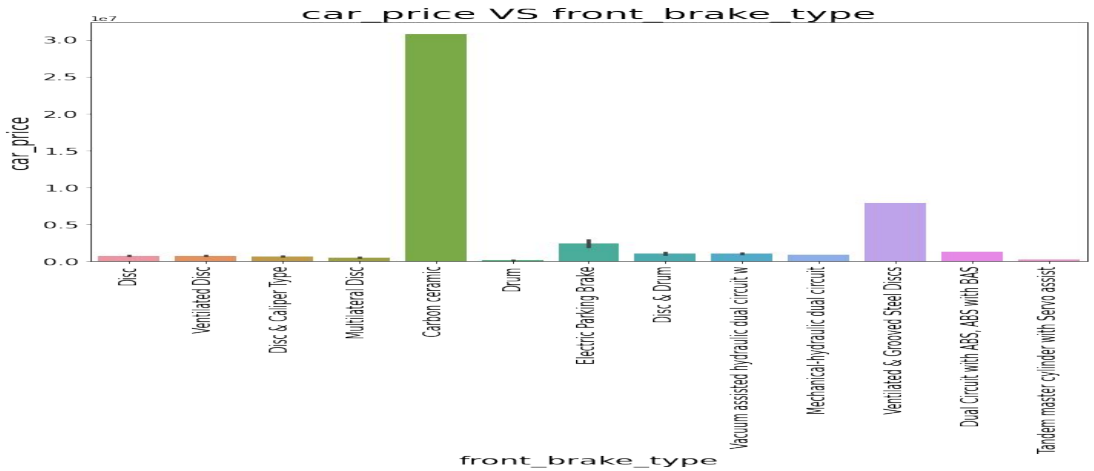
Observations:

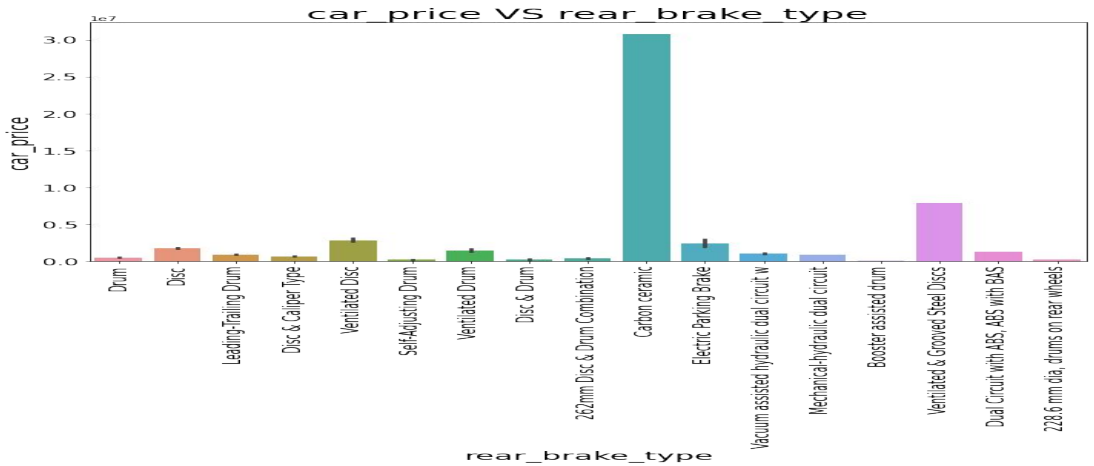
- ✔ Maximum cars are having below 20k driven kms. And car price is high for less driven cars.
- ✔ Maximum cars are having 1000-3000 Endine_disp. And car price is high for 3000 Endine_disp.
- ✔ Maximum cars are having milage of 10-25kms. And ,milage has no proper relation with car price.
- ✔ As Max_power is increasing car price is also increasing.
- ✔ Car_price has no proper relation with height.
- ✔ As the width is increasing car price is also increasing.
- ✔ As length is increasing car price is also increasing.
- ✔ Weight also has linear relationship with car price.
- ✔ As top_speed is increasing car price is also increasing.
- ✔ Cars with 5 and 4 seats are having highest price.
- ✔ As the age of the car increases the car price decreases.

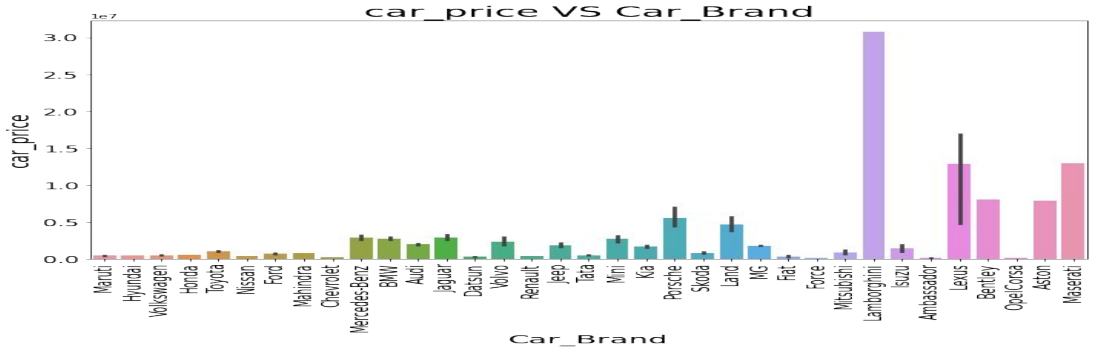
Bivariate Vizualization of categorical columns:











Observations:

- ✔ For Diesel and Electric cars the price is high compared to Petrol, LPG and CNG.
- ✔ Cars with automatic gear are costlier than manual gear cars.
- ✔ Cars with Carbon Ceramic front break are costlier compared to other cars.
- ✔ Cars with carbon Ceramic rear break are costlier compared to other cars.
- ✔ Lamborghini brand cars are having highest sale price.
- ✔ In Bangalore, Hyderabad and delhi-ncr the car prices are high as they are highly populated cities.

Analysis:

- ✔ I have used dist plot to check the skewness in numerical columns.
- ✔ I have used bar plot for each of categorical feature that shows the relation with the median car price for all the sub categories in each categorical feature.
- ✔ And also for continuous numerical variables I have used reg plot and strip to show the relationship between continuous numerical variable and target variable.
- ✔ I found that there is a linear relationship between continuous numerical variable and Car Price.

Data Cleaning Steps:

- ✔ 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.

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

i) 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.

ii) 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 96.86% r2_score and the difference between r2_score and cross validation score is 3.61%.

iii) 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
```

- GradientBoostingRegressor is giving me 94.93% r2_score and the difference between r2_score and cross validation score is 4.74%.

iv) 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 : 2.955005089601300 5
```

- DecisionTreeRegressor is giving me 91.79% r2_score and the difference between r2_score and cross validation score is 2.96%.

v) BaggingRegressor:

In [109]:
```
BR=BaggingRegressor()
BR.fit(X_train,y_train)
pred=BR.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(BR, 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: 95.67561886178403
mean_squared_error: 11286430156.537165
mean_absolute_error: 55118.9760619255
root_mean_squared_error: 106237.61177914894

Cross validation score : 92.60347410600774

R2_Score - Cross Validation Score : 3.072144755776293
```

- BaggingRegressor is giving me 95.68% r2_score and the difference between r2_score and cross validation score is 3.07%.
- **By looking into the difference of r2_score and cross validation score i found DecisionTreeRegressor as the best model with 91.79% 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']})
```

Hyper Parameter Tunning:

```
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:

- ✔ I have saved my best model using .pkl as follows.
- ✔ Now after saving the best model, loading my saved model and predicting the price values.

```
In [117]: # Loading the saved model
          model=joblib.load("Car_Price.pkl")

          #Prediction
          prediction = model.predict(X_test)
          prediction

Out[117]: array([ 379000.        , 1650000.        ,  355000.        , ...,
                  556333.33333333,  199923.07692308,  364800.        ])
```

```
In [118]: pd.DataFrame([model.predict(X_test)[:],y_test[:]],index=["Predicted","Actual"])
```

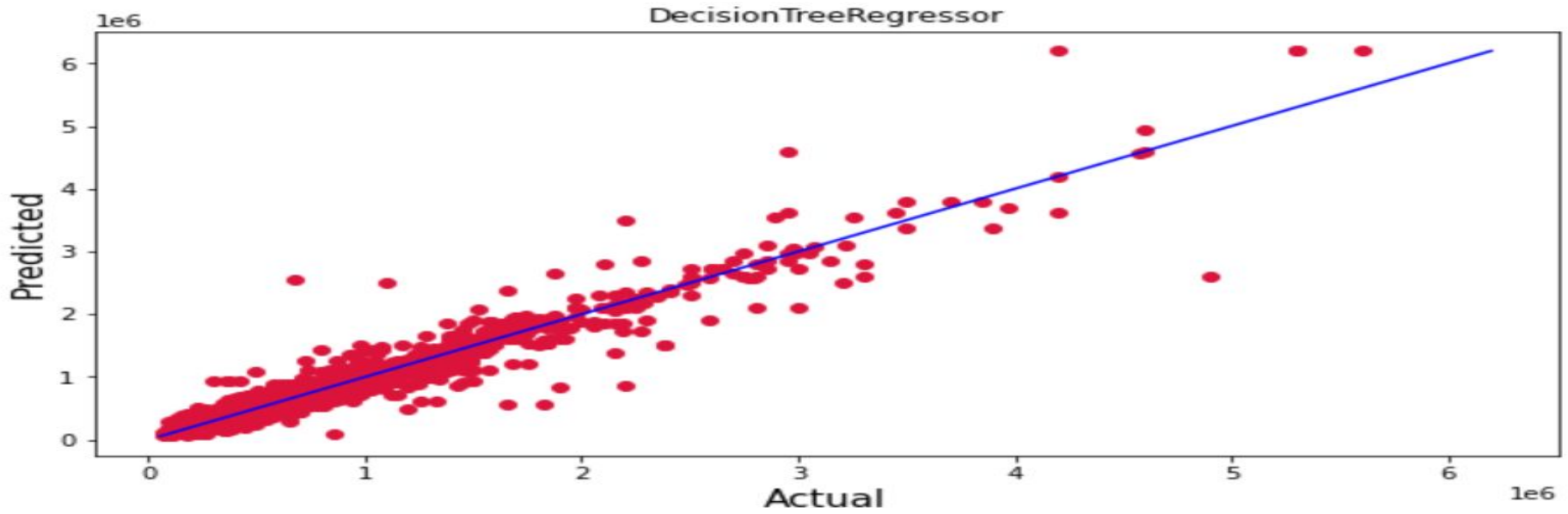
Out[118]:

	0	1	2	3	4	5	6	7	8	9	10
Predicted	379000.0	1650000.0	355000.0	332666.666667	490333.333333	470382.352941	590000.0	590000.0	1.541333e+06	687000.0	7.133333e+05	412017.24137
Actual	379000.0	1650000.0	465000.0	435000.000000	550000.000000	450000.000000	550000.0	550000.0	1.500000e+06	643000.0	1.125000e+06	385000.00000

- ✔ **I have predicted the Car Price using saved model, and the predictions look good. The Predicted values are almost same as actual values.**

Ploting 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()
```



- **Plotting Actual vs Predicted, To get better insight. Bule line is the actual line and red dots are the predicted values.**

Conclusion:

- ✔ In this project report, we have used machine learning algorithms to predict the used car price. We have mentioned the step by step procedure to analyze the dataset and finding the correlation between the features.
- ✔ Thus we can select the features which are correlated to each other and are independent in nature. The power of visualization has helped us in understanding the data by graphical representation it has made me to understand what data is trying to say.
- ✔ Data cleaning is one of the most important steps to remove unrealistic values and unnecessary values.
- ✔ These feature set were then given as an input to five algorithms and a hyper parameter tunning was done to the best model and the accuracy has been improved. Hence we calculated the performance of each model using different performance metrics and compared them based on these metrics.
- ✔ Then we have also saved the best model and predicted the car price. It was good that the predicted and actual values were almost same.
- ✔ 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.


THANK
YOU!