NAME OF THE PROJECT

Used Car Price Prediction using Machine Learning techniques

Submitted by:

Prathamesh Nayak

Introduction

Business Problem Framing

The price of a new car 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 prices of new cars and the financial incapability of the customers to buy them, Used Car sales are on a global increase. With the covid 19 impact in the market, we have seen a 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.

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. Existing System includes a process where a seller decides a price randomly and buyer has no idea about the car and its value in the present-day scenario. In fact, the seller also has no idea about the car's existing value or the price he should be selling the car at. To overcome this problem there is a need to develop a model which will be highly effective to predict the actual price of a car rather than the price range of a car.

Conceptual Background of the Domain Problem

Determining whether the listed price of a used car is a challenging task, due to the many factors that drive a used vehicle's price on the market. The focus of this project is developing machine learning models that can accurately predict the price of a used car based on its features, in order to make informed purchases.

Accurate car price prediction involves expert knowledge, because price usually depends on many distinctive features and factors. Typically, the most significant ones are brand and model, age, horsepower and mileage.

The fuel type used in the car as well as fuel consumption per mile highly affect the price of a car due to a frequent change in the price of a fuel. Different features like exterior color, door number, type of transmission, dimensions, safety, air condition, interior, whether it has 5 navigation or not will also influence the car price. In this project, we applied different methods and techniques in order to achieve higher precision of the used car price prediction.

Regression Algorithms are used because they provide us with continuous value as an output and not a categorized value because of which it will be possible to predict the actual price of a car rather than the price range of a car.
The data associated with the investigation was very large because there are thousands of used cars and each car's data comprises values of many features. Both data gathering and analysis are complex. Used cars data scrape from www. cardheko.com which is a well-known online platform for reselling used and new cars in India. Features like car's model, make, seating capacity, color, mileage, engine capacity, brakes, Torque, Transmission type, Maximum power, Gearbox type, power steering, engine type, turbocharger, supercharger and price were included.

Review of Literature

Pudaruth [1] applied various machine learning algorithms, namely: k-nearest neighbors, multiple linear regression analysis, decision trees and naïve bayes for car price prediction in Mauritius. The dataset used to create a prediction model was collected manually from local newspapers in a period less than one month, as time can have a noticeable impact on the price of the car. He studied the following attributes: brand, model, cubic capacity, mileage in kilometers, production year, exterior color, transmission type and price. However, the author found out that Naive Bayes and Decision Tree were unable to predict and classify numeric values. Additionally, a limited number of dataset instances could not give high classification performances, i.e., accuracy is less than 70%.

As per information that was gotten from the Agency for Statistics of BiH, 921.456 vehicles were registered in 2014 from which 84% of them are cars for personal usage [2]. This number has increased by 2.7% since 2013 and it is likely that this trend will continue, and the number of cars 6 will increase in future. This adds additional significance to the problem of the car price prediction.

Nitis Monburinon et al. [3] proposed a prediction of Prices for Used Car by Using Regression Models. In this paper, the authors selected the data from the German ecommerce site. The main goal of this work is to find a suitable predictive model to predict the used cars price. They used different machine learning techniques for comparison and used the mean absolute error (MAE) as the metric. They proposed that their model with gradient boosted regression has a lower error with MAE value 0.28 and this gives the higher performance where linear regression has the MAE value 0.55, random forest with MAE value 0.35.

Enis Gegic et al. [4] proposed Car Price Prediction using Machine Learning Techniques. In this paper, they proposed an ensemble model by collecting different types of machine learning techniques like Support Vector Machine, Random Forest and Artificial neural network. They collected the data from the web portal www.autopijaca.ba and built this model to predict the price of used cars in Herzegovina and Bosnia. The accuracy of their model is 87%.

Kanwal Noor and Sadaqat Jan [5] proposed a Vehicle Price Prediction System using Machine Learning Techniques. In this paper, they proposed a model to predict the price of the cars through multiple linear regression methods. They selected the most influencing feature and removed the rest by performing feature selection technique. The Proposed model achieved the prediction precision of about 98%.

Kuiper [6] used this model to predict the price of 2005 General Motor (GM) cars. The price prediction of cars does not require any special knowledge so the data available online is enough to predict prices like the data available on www.pakwheels.com. Kuiper [6] did the same i.e. car price

prediction and introduced variable selection techniques which helped in finding which International Journal of Computer Applications (0975 – 8887) Volume 167 – No.9, June 2017 28 variables are more relevant for inclusion in model.
Normally, a value greater than 2 for standardized residuals is considered large. Since raw residuals have no capability to detect outliers, 7 therefore, Standardized residuals are considered a better measure for such detection [7].

Gonggie [8] proposed a model that is built using ANN (Artificial Neural Networks) for the price prediction of a used car. He considered several attributes: miles passed, estimated car life and brand. The proposed model was built so it could deal with nonlinear relations in data which was not the case with previous models that were utilizing the simple linear regression techniques. The non-linear model was able to predict prices of cars with better precision than other linear models.

Problem Undertaken

The project was the first provided to me by Flip Robo Technologies as a part of the internship programme. The exposure to real world data and the opportunity to deploy my skills in solving a real time problem has been the primary motivation

Data needed for this project is required to scrap from the internet and work over it. Deciding whether a used car is worth the posted price when you see listings online can be difficult. The focus of this project is developing machine learning models that can accurately predict the price of a used car based on its features, in order to make informed purchases. The model developed in this study may help online web services that tell a used car’s market value.

Analytical Problem Framing

Mathematical / Analytical Modeling of the Problem

Whenever we employ any ML algorithm, statistical models or feature pre-processing in the background a lot of mathematical framework work. In this project we have done a lot of data pre-processing & ML model building. Different ML algorithms used in this project have their own mathematical background, for which you can refer to Scikit documentation [1].

Data Sources and their formats

```
# Importing dataset excel file using pandas.
df= pd.read_excel('Oldcars.xlsx')
```

```
print('No. of Rows :',df.shape[0])
print('No. of Columns :',df.shape[1])
pd.set_option('display.max_columns',None) # # This will enable us to see truncated columns
df.head()
```

```
No. of Rows : 10000
No. of Columns : 24
```

Data Scrap for 10000 cars and 24 features scrap form website. So this dataset contains 10544 rows with 24 columns.

```
df.columns.to_series().groupby(df.dtypes).groups
```

```
{float64: ['Make Year', 'No of Cylinder'], object: ['Car Model', 'Fuel Type', 'KMs driven', 'Engine Displacement(CC)', 'Transmi
ssion', 'Milage(kmpl)', 'Max Power(bhp)', 'Torque(Nm)', 'Seating Capacity', 'Color', 'Gear Box', 'Steering Type', 'Front Brake
Type', 'Rear Brake Type', 'Tyre Volume', 'Engine Type', 'Turbo Charger', 'Super Charger', 'Length(mm)', 'Width(mm)', 'Height(m
m)', 'Price(Rs)']}
```

Price is our Target variable which is to be predicted. We can see that some numerical variables like torque, mileage are by default with object data types.

Data Pre-processing

The dataset is large and it may contain some data error. In order to reach clean, error free data some preprocessing is done on data. First task is 9 to find errors in data or data entry correction. Second task is to convert data types into appropriate data types. Third one is to perform feature engineering on 'make year' to extract car age. Different data error found in data as followed:

▪ KMs Driven involved numeric value with object data types and Value are in format like 2,36,000 KMs. Here comma (' , ') is removed from data and converted into numeric data types

```
: df['KMs driven'] = df['KMs driven'].map(lambda x : x.split(' ')[0])

: df['KMs driven'] = df['KMs driven'].map(lambda x : x.replace('Kms',''))

: df['KMs driven'] = df['KMs driven'].map(lambda x : x.replace(',',''))

: df['KMs driven']=pd.to_numeric(df['KMs driven'])

: df['KMs driven'].dtypes

: dtype('int64')
```

▪ Some of the data in 'Engine Displacement (CC)' comes with CC notation attached to value, for example, 1497 CC. So, CC is stripped away and subsequently converted into numeric data types.

```
df['Engine Displacement(CC)'] = df['Engine Displacement(CC)'].astype(str)

df['Engine Displacement(CC)']= df['Engine Displacement(CC)'].map(lambda x : x.replace('cc',''))

df['Engine Displacement(CC)'] = pd.to_numeric(df['Engine Displacement(CC)'])

df['Engine Displacement(CC)'].dtypes

dtype('int64')
```

▪ Feature Mileage contains dash (-) and some values in units of km/kg or km/hr which need to be converted into standard unit kmpl. This conversion in the proper unit is done and finally mileage data types are converted into numeric data types.

```
: df['Milage(kmpl)'] = df['Milage(kmpl)'].astype(str)
```

```
: df['Milage(kmpl)'] = df['Milage(kmpl)'].map(lambda x : x.replace('km/kg',''))
```

```
: df['Milage(kmpl)'] = df['Milage(kmpl)'].map(lambda x : x.replace('-',''))
```

```
: df['Milage(kmpl)'] = df['Milage(kmpl)'].map(lambda x : x.replace('km/hr',''))
```

```
: df['Milage(kmpl)'] = pd.to_numeric(df['Milage(kmpl)'])
```

```
: df['Milage(kmpl)'].dtypes
```

```
: dtype('float64')
```

▪ Max Power contains several errors like presence of null, dash (-), some values with units like PS. These errors are resolved and converted into numeric data types.

▪ Some torque values came with units like kg m or nm. This entry is converted appropriately in the NM unit. In addition, it also contains some (-), which is resolved by replacing it with Null value.

▪ The features like length, width and height come with a comma in between values (for example 4,5820 mm) along with (-) at some places. These features come with default object data type which was converted into numeric datatype after resolving mentioned error.

```
df['Length(mm)'] = df['Length(mm)'].str.replace(',','')
```

```
df['Length(mm)'] = df['Length(mm)'].str.replace('-','')
```

```
df['Length(mm)'] = pd.to_numeric(df['Length(mm)'])
```

```
df['Length(mm)'].dtypes
```

```
dtype('float64')
```

Similar code is applied for remaining feature width and height.

▪ In price column some entry comes in term of lakh and Cr along with numeric value (for example, 32.15 lakh*) resulting in an object datatype. The conversion into appropriate numeric value is required for such entries which are resolved using the following piece of code.

```
df['Price(Rs)'] = df['Price(Rs)'].str.replace('Lakh*','100000')
df['Price(Rs)'] = df['Price(Rs)'].str.replace('Cr*','100000')
df['Price(Rs)'] = df['Price(Rs)'].str.replace(',','')
```

```
df['Price(Rs)'] = df['Price(Rs)'].str.replace('*','')
```

```
df['Price(Rs)'] = df['Price(Rs)'].str.replace('100000ore','100000')
```

```
df[['a','b']] = df['Price(Rs)'].str.split(expand=True)
df['a'] = df['a'].astype("float")
df['b'] = df['b'].astype("float")
```

```
df['b'] = df['b'].fillna(value = 1)
df["Price (Rs.)"] = df['a'] * df['b']
```

```
df.drop(columns=['Price(Rs)','a','b'], inplace = True)
```

▪ Small feature engineering performs on make year to extract how old a car is since first sold out by manufacturer?

```
df['Car_Age'] = 2022 - df['Make Year']
```

```
df.drop(columns=['Make Year'], inplace = True)
```

▪ The car model name we extracted from website contain both car

```
df['Car_Brand'] = df["Car Model"].str.split(' ').str[:2]
df['Car_Brand'] = df['Car_Brand'].apply(lambda x: ','.join(map(str, x)))
df['Car_Brand'] = df['Car_Brand'].str.replace(',',' ')
df['Car_Model'] = df["Car Model"].str.split(' ').str[2:]
df['Car_Model'] = df['Car_Model'].apply(lambda x: ','.join(map(str, x)))
df['Car_Model'] = df['Car_Model'].str.replace(',',' ')
```

brand and its variant. Some feature engineering performs to extract these additional features out of the original car model feature using the following piece of code.

In the next phase of data pre-processing, we will make the correction in the name of subcategories of categorical features as multiple names for the same label found in value_counts ().

```
df['Super Charger'].value_counts()
```

```
No     8497
Yes      12
NO        4
Name: Super Charger, dtype: int64
```

```
df['Super Charger'].replace('NO','No',inplace= True)
```

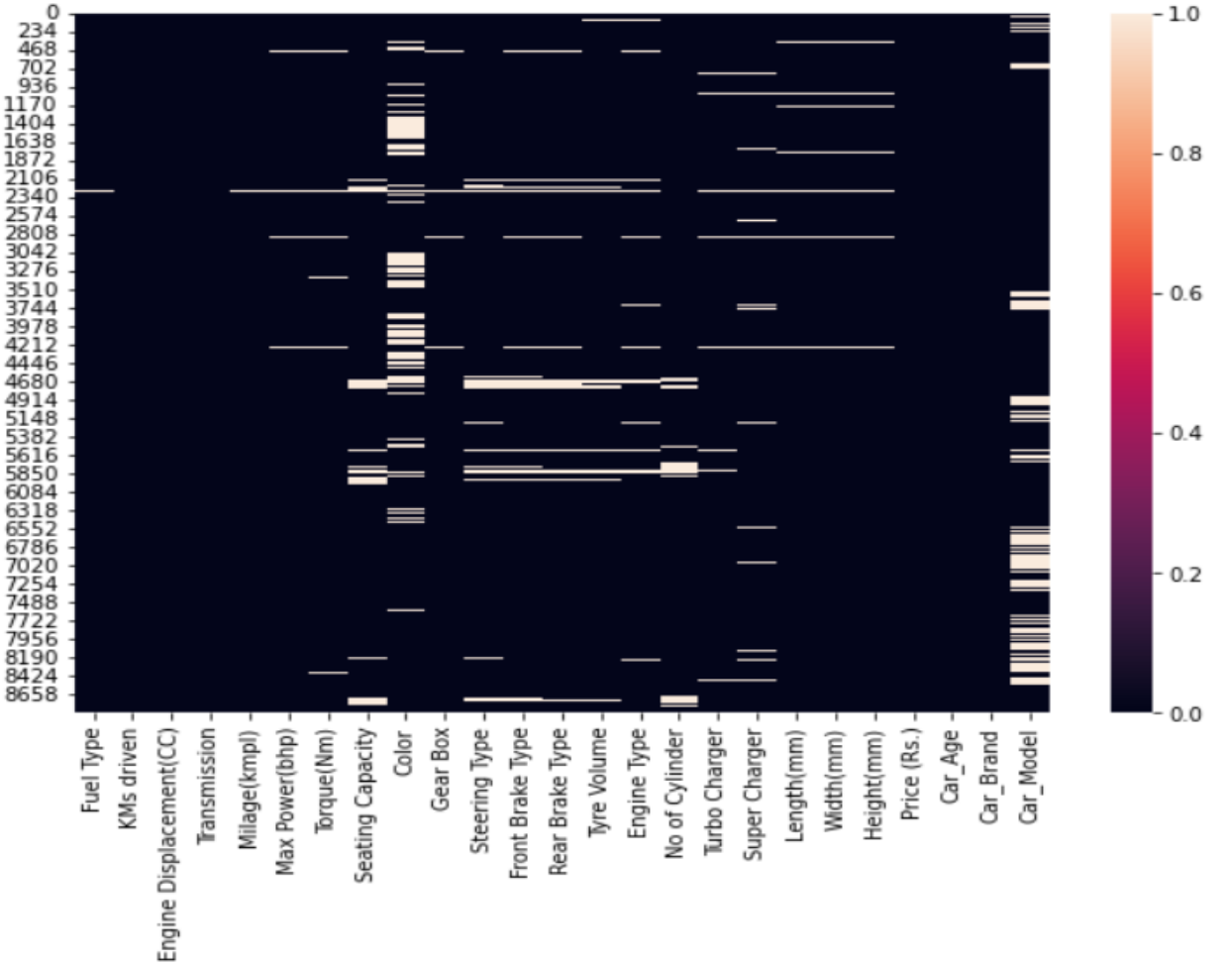
```
df['Super Charger'].value_counts()
```

```
No     8501
Yes      12
Name: Super Charger, dtype: int64
```

We can see subcategory ‘No’ comes in two different names in value_counts of Supercharger and we also see intended correction is made in subcategory values with replace command. Similar kind of correction perform on following feature:

➢ Turbo Charger
➢ Front Brake Type
➢ Rear Brake Type
➢ Steering Type 13
➢ Gearbox
➢ Tyre Volume

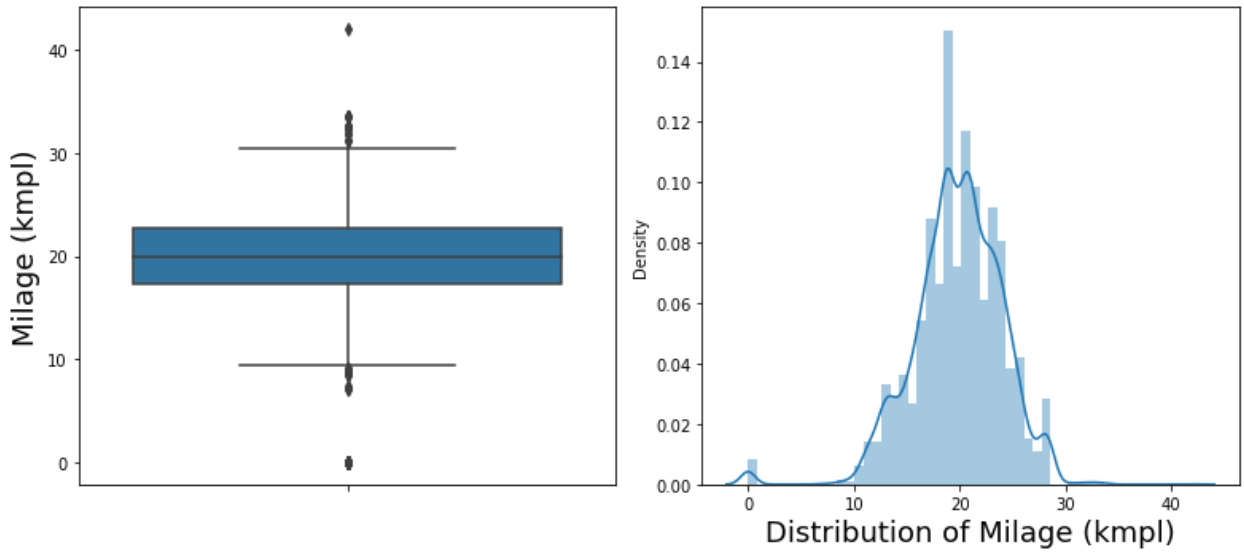
Next phase of data pre-processing is handling missing values.



Lots of features contain missing values. Categorical value can be imputed with mode of categories.

```
df['Front Brake Type'].replace('Ventilated Discs','Ventilated Disc', inplace = True)
df['Front Brake Type'].replace('Ventilated DIsc','Ventilated Disc', inplace = True)
df['Front Brake Type'].replace('Disk','Disc', inplace = True)
df['Front Brake Type'].replace('Ventilated Disk','Ventilated Disc', inplace = True)
df['Front Brake Type'].replace('Ventilated discs','Ventilated Disc', inplace = True)
df['Front Brake Type'].replace('Disc, 236 mm','Disc (236 mm)', inplace = True)
df['Front Brake Type'].replace('Vantilated Disc','Ventilated Disc', inplace = True)
df['Front Brake Type'].replace('Ventlated Disc','Ventilated Disc', inplace = True)
df['Front Brake Type'].replace('Ventillated Disc','Ventilated Disc', inplace = True)
df['Front Brake Type'].replace('Ventillated Discs','Ventilated Disc', inplace = True)
df['Front Brake Type'].replace('264mm Ventilated discs','Ventilated Disc (264mm)', inplace = True)
df['Front Brake Type'].replace('disc','Disc', inplace = True)
df['Front Brake Type'].replace('260mm discs','Disc (260mm)', inplace = True)
df['Front Brake Type'].replace('Disc brakes','Disc', inplace = True)
df['Front Brake Type'].replace('Discs','Disc', inplace = True)
df['Front Brake Type'].replace('Disc,internally ventilated','Ventilated Disc', inplace = True)
df['Front Brake Type'].replace('Ventilated disc','Ventilated Disc', inplace = True)
```

Numerical value can be imputed with mean or median depending on sensitivity to outliers. For example, in this project imputation of missing 14 values can be done after examining boxplot for outliers & distribution using distplot.



Milage (kmpl) is almost symmetrical in nature. Outliers are also spread to both lower & upper bound. So, we will be imputing Milage (kmpl) with mean.

```
print("Mean of Milage(kmpl):",df['Milage(kmpl)'].mean(),'kmpl')
print("Median of Milage(kmpl):",df['Milage(kmpl)'].median(),'kmpl')
```

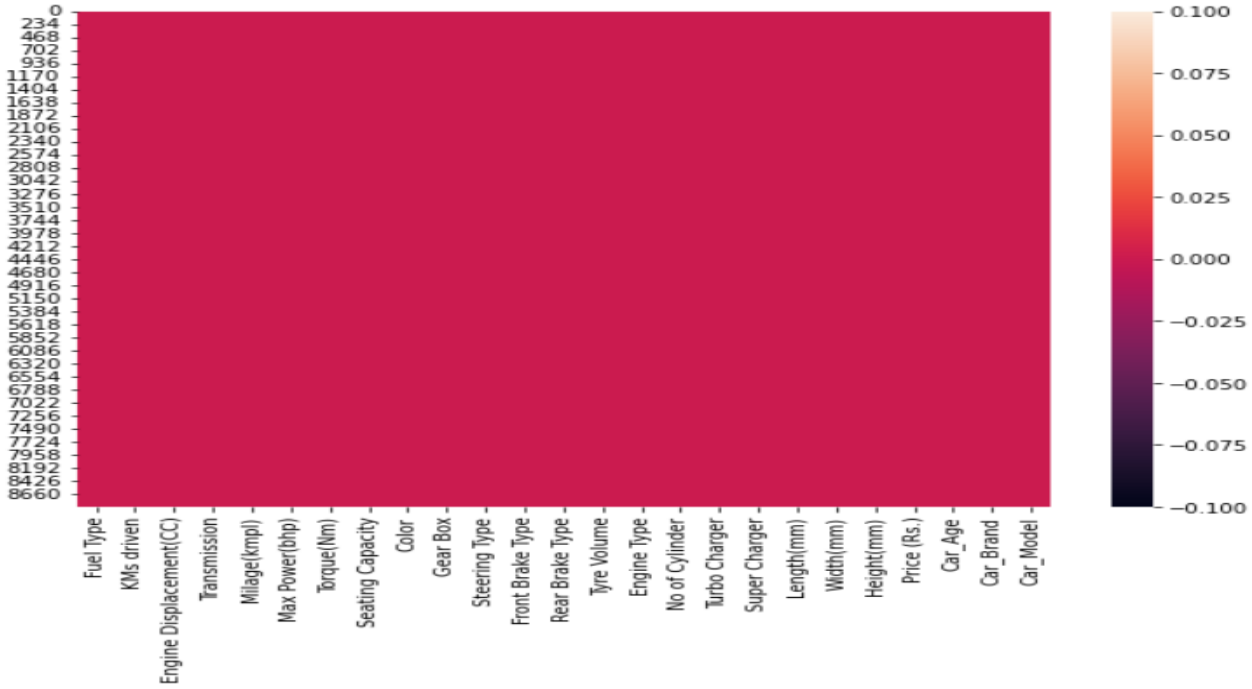
```
Mean of Milage(kmpl): 19.82472965345972 kmpl
Median of Milage(kmpl): 20.0 kmpl
```

```
df['Milage(kmpl)'].fillna(df['Milage(kmpl)'].mean(), inplace =True)
```

Similar kind of imputation is done for remaining feature after examining boxplot & distribution as follow:

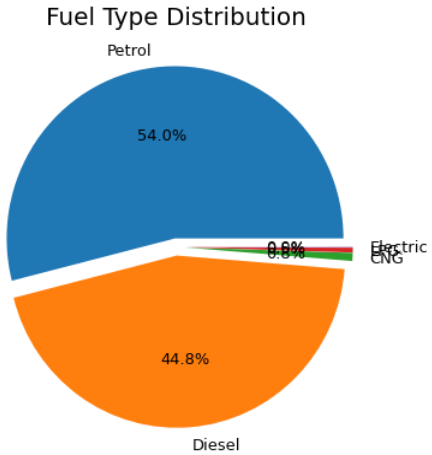
➢ Imputing Max Power with Median as median is less sensitive to outliers.
➢ Due presence of outliers missing values impute with median.
➢ Imputing length with mean as almost no outliers present.
➢ Width & Height are imputed with median of respective feature.

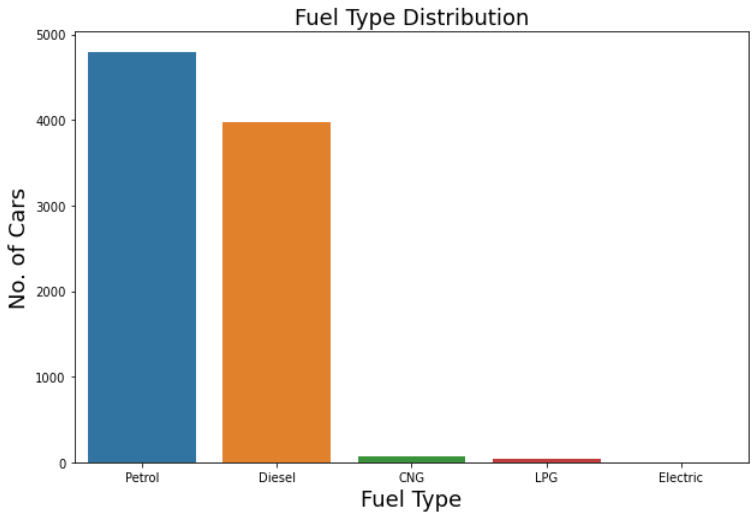
Heatmap of missing value after imputation shown below

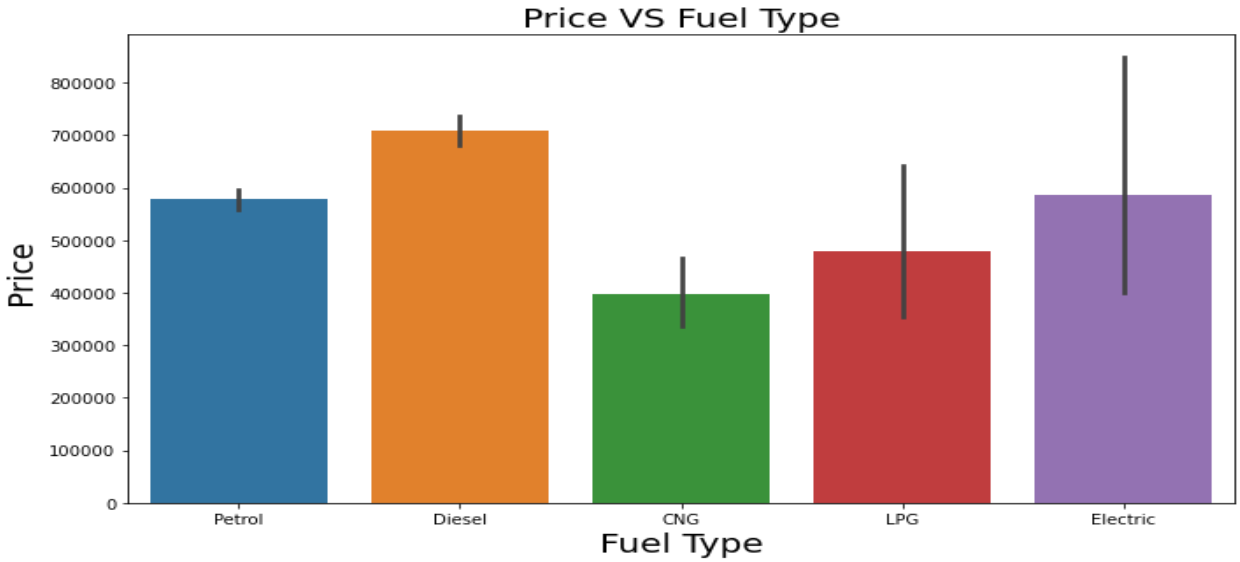


Exploratory Data Analysis

Let see key result from EDA, start with fuel type

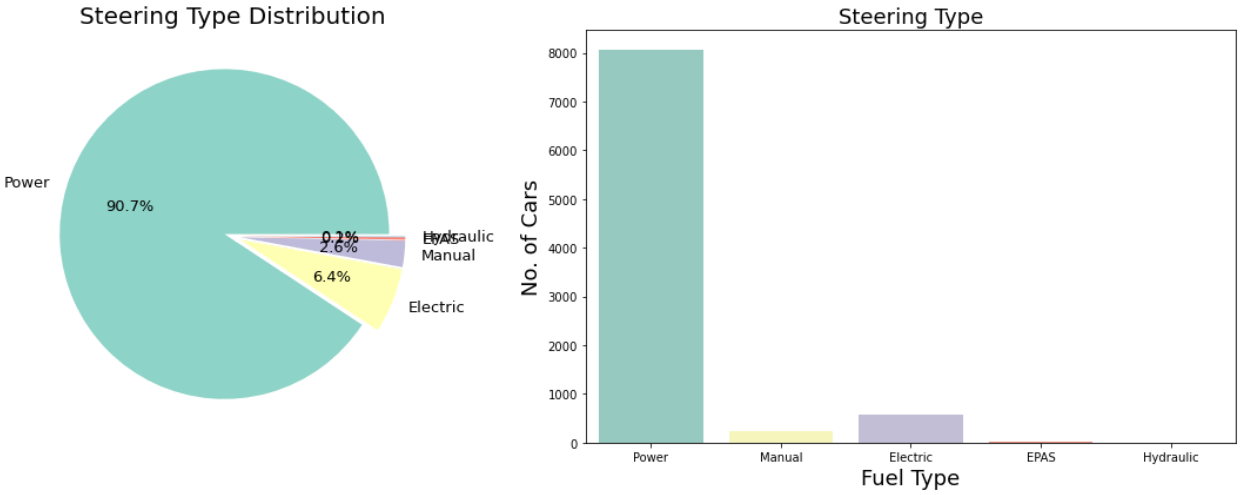




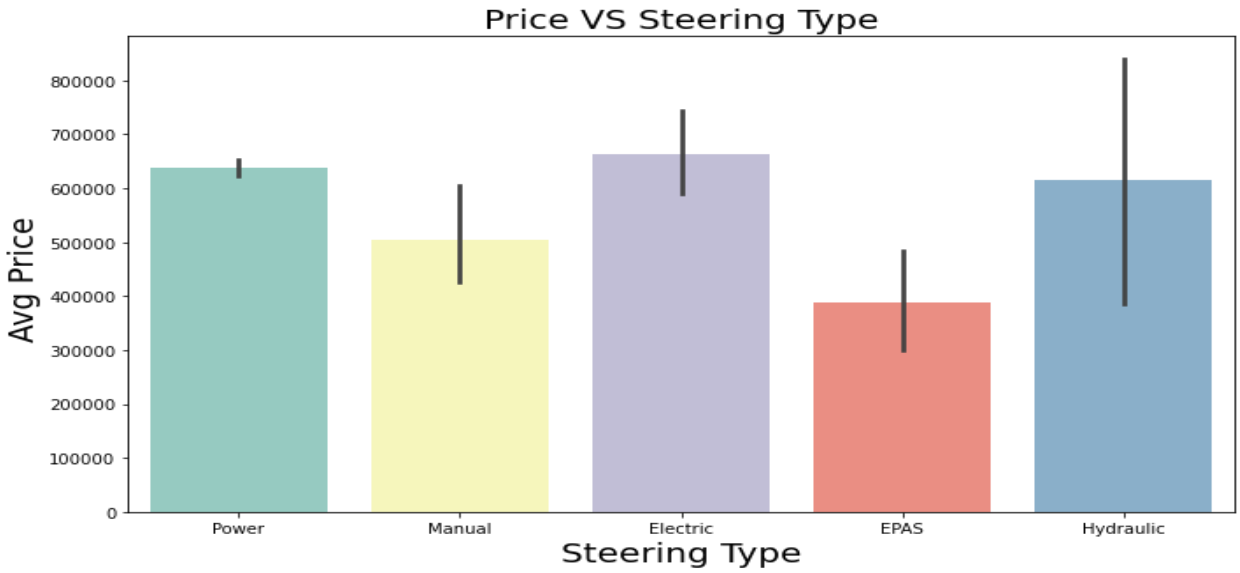


• Most of the cars are Petrol operated followed by Diesel. This may be due to low prices of Petrol car compared to diesel car.
• Very small segment of electric car and also the price is quite high compared to petrol based.
• CNG based cars are Cheapest compared to others.

➢ Price Vs Steering Type

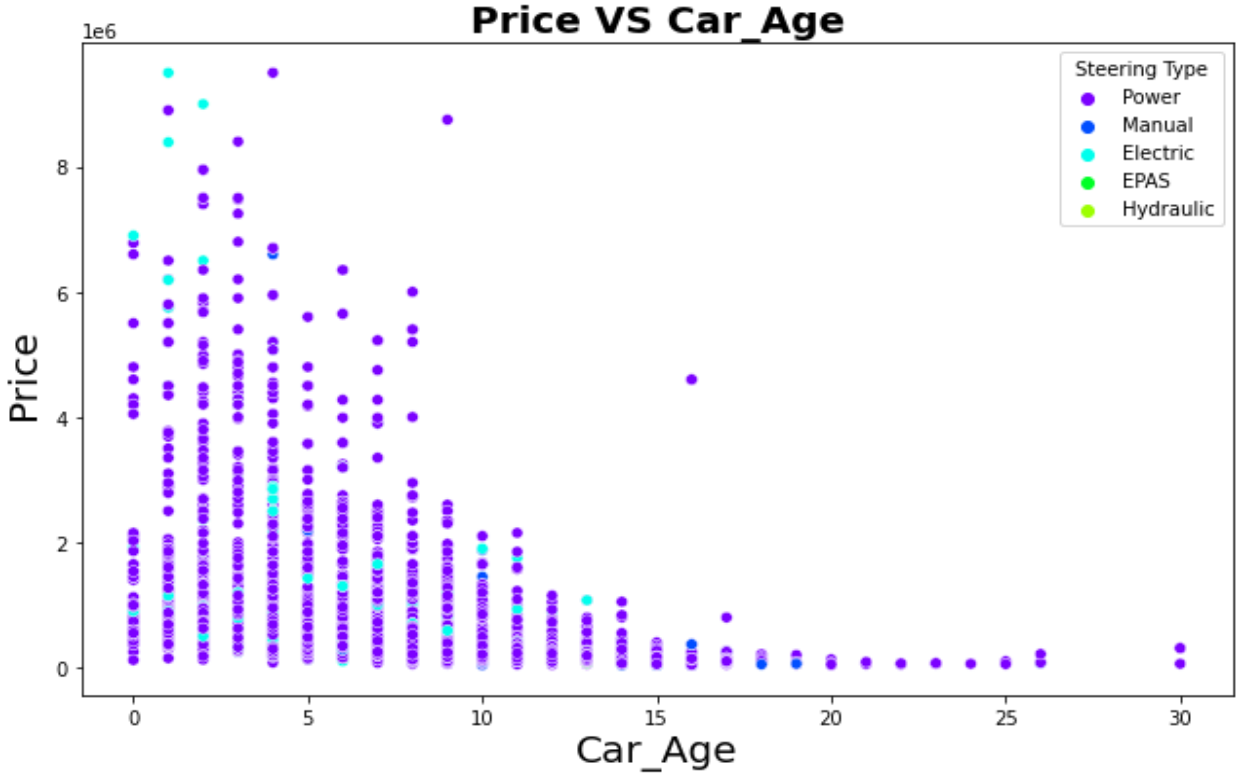


More than 90 % of car users prefer Power steering compares to others.

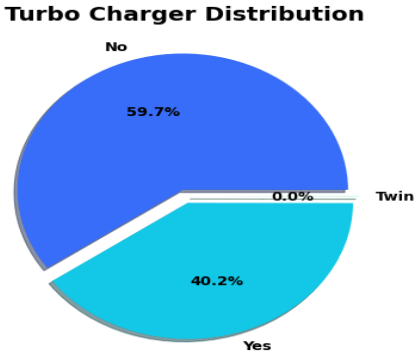


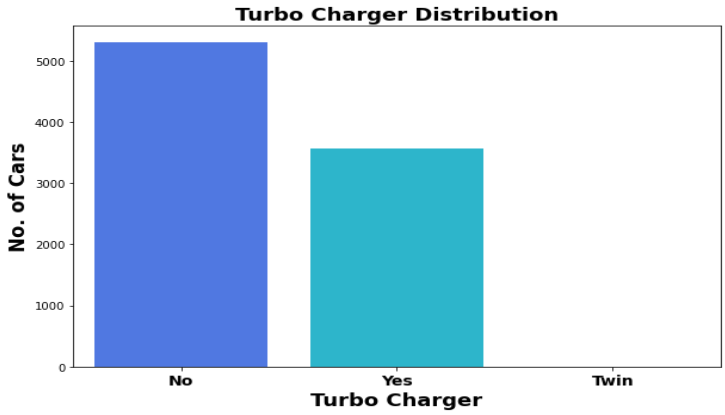
• 6.2% of cars are based on electric steering, which is costly compared to others. • Very small section of car still uses Manual Steering, Most probably they belong to the old model.

Let check predication in last point by plotting Car age Vs Price based on steering types

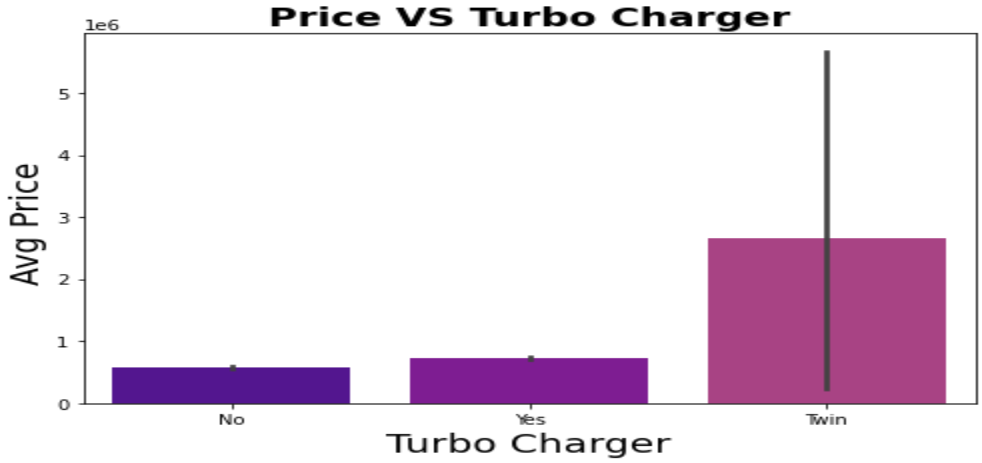


Here we got confirmation of prediction in previous section, almost all manual steering-based car at least 10-year-old

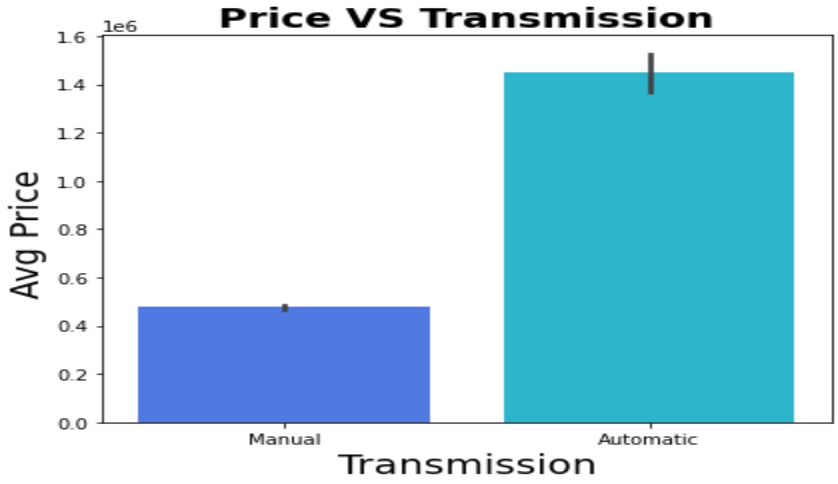




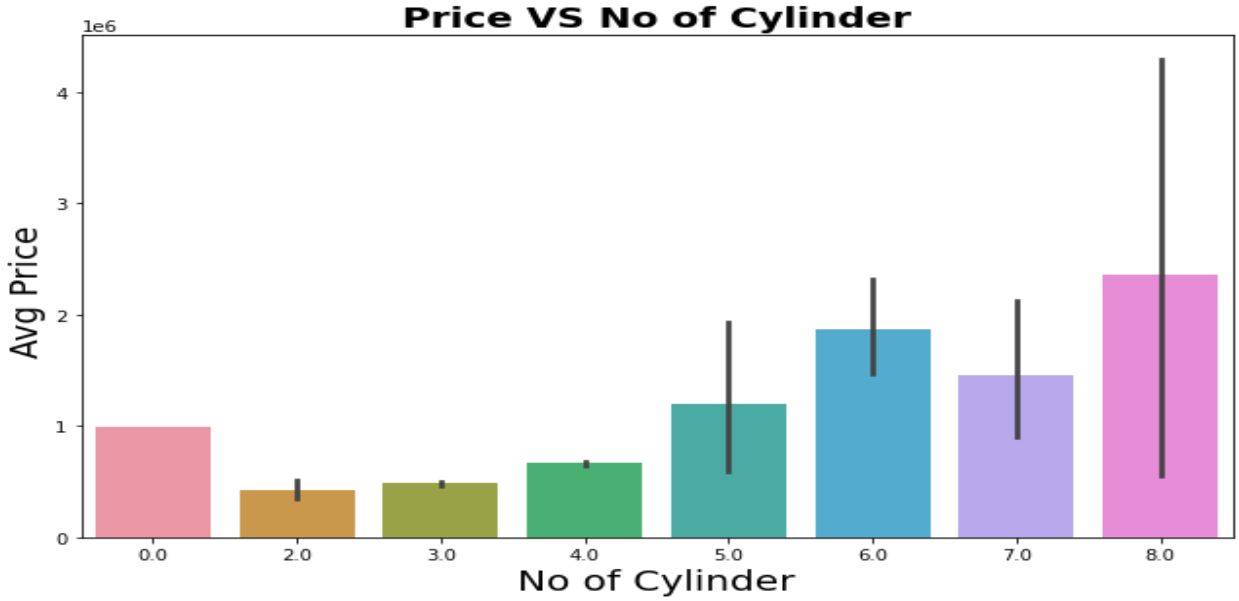
40% cars are with turbo chargers & almost less than 1 % cars with twin facilities.



As expected, Max price for cars based on Twin engine followed by turbocharger.

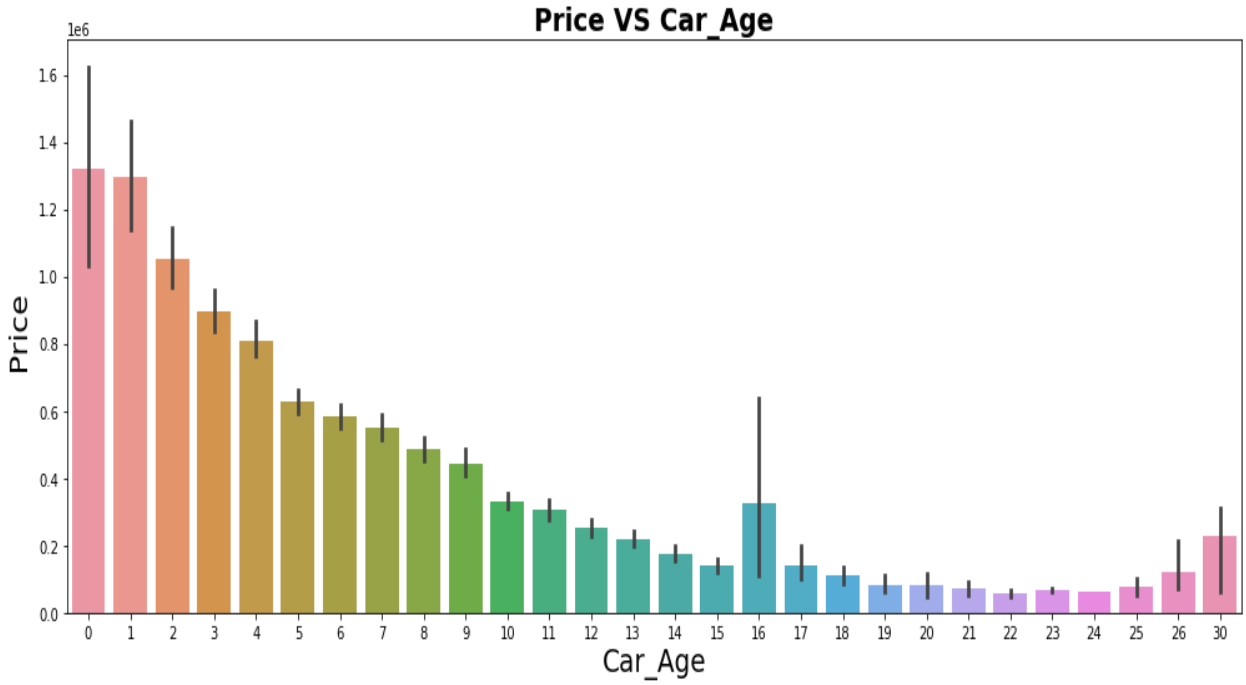


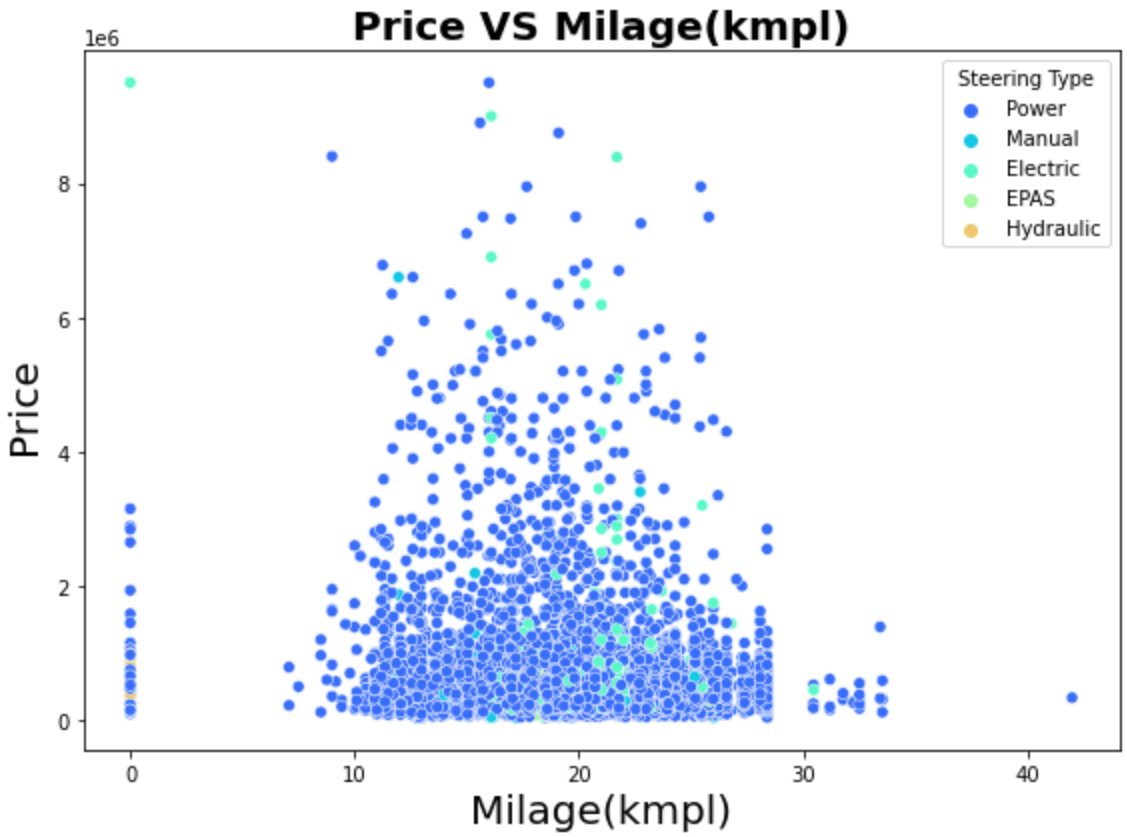
Let's check the effect of the number of cylinders on price.



• From value counts we got information that most Cars with are 4-cylinder engines followed by 3-cylinder engines.
• In terms of Avg. Price as the number of cylinders increases the average price increases.

We know that as car prices decrease as car models become older. Let's plot the Price vs Car age to verify it.





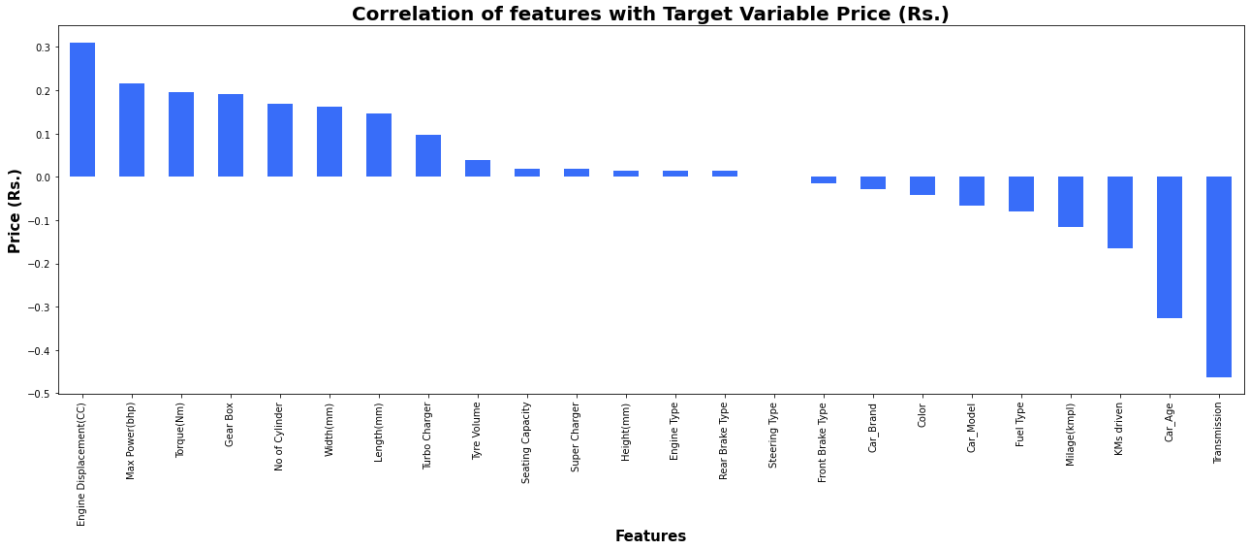
Mileage (kmpl) varies between 10 to 25 kmpl for most cars. For Majority cars, the price is below 0.5e6. We didn't get any other significant relation between price and steering types

Label Encoding of categorical data:

```
# Using Label encoder for transforming Categorical data
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
for i in Categorical:
    df[i] = le.fit_transform(df[i])
df.head()
```

Data Inputs- Logic- Output Relationships

To gain more insight about the relationship between input & output heatmap of correlation and bar plot of correlation of labels with independent features is plotted.



We can see most features are either poorly or moderately correlated with target variable Price.

Models Development & Evaluation

```
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import ExtraTreesRegressor
from xgboost import XGBRegressor
from sklearn.ensemble import AdaBoostRegressor
from sklearn.ensemble import  GradientBoostingRegressor
from sklearn.ensemble import BaggingRegressor
from sklearn.metrics import mean_squared_error, mean_absolute_error,  r2_score
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
import sklearn.metrics as metrics
```

IDENTIFICATION OF POSSIBLE PROBLEM SOLVING APPROACHES (METHODS)

Following metrics used for evaluation:
1. Mean absolute error which gives magnitude of difference between the prediction of an observation and the true value of that observation.
2. Root mean square error is one of the most commonly used measures for evaluating the quality of predictions.
3. R2 score, which tells us how accurately our model predicts the result, is going to be an important evaluation criteria along with Cross validation score.

Testing of Identified Approaches (Algorithms)

Phase 1 Web Scraping Strategy employed in this project as follow:
1. Selenium will be used for web scraping data from cardheko.com
2. In the first part Scraping URL of Used car for different locations in India.
3. Storing Scrap URL in excel file.
4. Selecting car features to be scrapped from the website.
5. In the second part Scraping data from individual URLs in excel file.
6. Exporting final data in Excel file.

The different regression algorithm used in this project to build ML model are as below:

- ❖ Random Forest Regressor ❖ Decision Tree Regressor
- ❖ XGB Regressor
- ❖ Gradient Boosting Regressor
- ❖ Bagging Regressor

KEY METRICS FOR SUCCESS IN SOLVING PROBLEM UNDER CONSIDERATION

Following metrics used for evaluation:
1. Mean absolute error which gives magnitude of difference between the prediction of an observation and the true value of that observation.

2. Root mean square error is one of the most commonly used measures for evaluating the quality of predictions.

3. R2 score, which tells us how accurately our model predicts result, is going to be an important evaluation criteria along with Cross validation score.

RUN AND EVALUATE SELECTED MODELS

Random Forest Regressor

```
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 = 5).mean()*100
print("\nCross validation score :", scores)

# Difference between Accuracy and CV Score
diff = R2_score - scores
print("\nR2_Score - Cross Validation Score :", diff)
```

```
R2_score: 78.84376114476599
mean_squared_error: 108914244056.0833
mean_absolute_error: 142084.60197711497
root_mean_squared_error: 330021.5811974776

Cross validation score : 57.12655258661263

R2_Score - Cross Validation Score : 21.71720855815336
```

XGB Regress

```
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 on XGB Model
scores = cross_val_score(XGB, X, Y, cv = 5).mean()*100
print("\nCross validation score :", scores)

# Difference between Accuracy and CV Score
diff = R2_score - scores
print("\nR2_Score - Cross Validation Score :", diff)
```

```
R2_score: 85.70263386925507
mean_squared_error: 73604142720.19276
mean_absolute_error: 120818.41830068233
root_mean_squared_error: 271300.83435218694

Cross validation score : 69.99678963054534

R2_Score - Cross Validation Score : 15.70584423870973
```

Gradient Boosting Regressor:

```
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 on Gradient Boosting
scores = cross_val_score(GBR, X, Y, cv = 5).mean()*100
print("\nCross validation score :", scores)

# Difference between Accuracy and CV Score
diff = R2_score - scores
print("\nR2_Score - Cross Validation Score :", diff)
```

```
R2_score: 74.27591900399247
mean_squared_error: 132429911332.01727
mean_absolute_error: 176969.22631321248
root_mean_squared_error: 363909.2075394868

Cross validation score : 60.85257600192355

R2_Score - Cross Validation Score : 13.423343002068918
```

Decision Tree Regressor

```
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 = 5).mean()*100
print("\nCross validation score :", scores)

# Difference between Accuracy and CV Score
diff = R2_score - scores
print("\nR2_Score - Cross Validation Score :", diff)
```

```
R2_score: 54.20127236379708
mean_squared_error: 235776020178.24817
mean_absolute_error: 182982.39786156444
root_mean_squared_error: 485567.7297537885

Cross validation score : 19.636757355284402
```

All models are giving us R2 score & Cross validation Score more than 80%, So we will select a model which has less difference between these scores. On Basis of difference between R2 Score and Cross Validation Score XGBRegressor is selected as the best model with 85 % r2_score. We will perform Hyper Parameter tuning over this model.

```
from sklearn.model_selection import GridSearchCV
```

```
parameter = {
         'learning_rate': [0.01, 0.1],
         'max_depth': [3, 5, 7, 10],
         'min_child_weight': [1, 3, 5],
         'subsample': [0.5, 0.7],
         'colsample_bytree': [0.5, 0.7],
         'n_estimators' : [100, 200, 500],
         'objective': ['reg:squarederror']
    }
```

```
GCV=GridSearchCV(XGBRegressor(),parameter,cv=5,n_jobs = -1)
```

```
GCV.fit(X_train,Y_train)
```

```
GCV.best_params_
```

```
{'colsample_bytree': 0.7,
 'learning_rate': 0.1,
 'max_depth': 5,
 'min_child_weight': 1,
 'n_estimators': 500,
 'objective': 'reg:squarederror',
 'subsample': 0.7}
```

```
Final_mod=XGBRegressor(colsample_bytree = 0.7,
 learning_rate = 0.1,
 max_depth = 5,
 min_child_weight = 1,
 n_estimators = 500,
 objective = 'reg:squarederror',
 subsample = 0.7)
Final_mod.fit(X_train,Y_train)
pred=Final_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: 87.61357120438944
mean_squared_error: 63766463314.18555
mean_absolute_error: 118907.88022958286
RMSE value: 252520.22357463877
```

Interpretation of the Results

▪ As car models get old eventually its price reduces with time.
▪ In terms of Avg. Price as the number of cylinders increases the average price increases.
▪ The price of Automatic transmission is much greater than manual transmission.
▪ 40% cars are with turbo chargers & almost less than 1 % car with twin facility.

▪ Almost all manual steering-based cars are at least 10-year-old. ▪ More than 90 % of car users prefer Power steering compared to others.
▪ Very small segment of electric cars and also the price is quite high compared to petrol based.

Conclusion

➢ On Basis of difference between R2 Score and Cross Validation Score XGBRegressor is selected as best model with **87.61%** R2_score.
➢ Final Model is giving us R2 Score of **87.61%** which is slightly improved compare to earlier R2 score of 85 %.

Learning Outcomes of the Study in respect of Data Science

➢ Scraping data for project from www.cardekho.com. This is the first of such a project for me. Web scraping such a huge amount of data challenge my scraping skill.
➢ Data cleaning or data pre-processing aspect of project is good hands on for me in this area. There were a lot of discrepancies in data scrape with different units, different names for same sub-categories. Data cleaning was a big part of this project.

Limitations of this work and Scope for Future Work

➢ Around data for more than 10000 car scrap from cardheko.com
➢ We can scrap more data from different online platform like olx, car24. More data obviously means more accurate prediction.
➢ Here we Scrap almost 24 features. But there are many different kinds of safety, comfort, entertainment features to which buyers weigh while buying a car. We can also include such more feature in future