Name- ARUN PRASSATH J

Internship Program- Machine Learning with Python Internship

Batch- Dec2022-Feb 2023
Certificate Code- TCRIG03R17
Date of submission- 30 March 2023

Technical Coding Research Innovation, Navi Mumbai,
Maharashtra, India-410206

USED CAR PRICE PREDICTIONUSING MACHINE LEARNING

A Case-Study Submitted for the requirement of
Technical Coding Research Innovation

For the Internship Project work done during
MACHINE LEARNING WITH PYTHON INTERNSHIP PROGRAM

by

PAVULURI TRIVIKRAM RAO (TCRIG03R04)
ARUNPRASSATH.J(TCRIG03R17)

Rutuja Doiphode
CO-FOUNDER &CEO
TCR innovation.

USED CAR PRICE PREDICTION USING MACHINE LEARNING

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 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, 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 need to developed a model which will be highly effective to predict the actual price 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, 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 price of a car due to a frequent change in the price of a fuel. Different features like exterior colour, door number, type of transmission, dimensions, safety, air condition, interior, whether it has 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 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 of 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, colour, mileage, engine capacity, brakes, Torque, Transmission type, Maximum power, Gearbox type, power steering, engine type, turbocharger, supercharger and price were included.

Abstract:

This project aims to build a model to predict used cars' reasonable prices based on multiple aspects, including vehicle mileage, year of manufacturing, fuel consumption, transmission, road tax, fuel type, and engine size.

This model can benefit sellers, buyers, and car manufacturers in the used cars market.

Upon completion, it can output a relatively accurate price prediction based on the information that users input. The model building process involves machine learning and data science. The dataset used was scraped from listings of used cars.

Various regression methods, including linear regression, polynomial regression, support vector regression, decision tree regression, and random forest regression, were applied in the research to achieve the highest accuracy.

Objective

- To scrap used car data of at least 5000 cars from cardekho.com
- To Analyze data to gain key insights about current used car market.
- To build Machine Learning model to predict price of used car

DATASET Information:

- Dataset is Scrap from www.cardheko.com
- Selenium web driver is used to Scrap data of around 10544 cars
- Raw data in excel file contain 10544 rows and 24 feature .
- Dataset contain some errors, so data cleaning operation was performed.
- Data integrity check is perform for missing values, duplicate data, data error

Review of Literature

Pudaruth [1] applied various machine learning algorithms, namely: k-nearest neighbours, 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 period less than one month, as time can have a noticeable impact on price of the car. He studied the following attributes: brand, model, cubic capacity, mileage in kilometre's, production year, exterior colour, transmission type and price. However, the author found out that Naive Bayes and Decision Tree were unable to predict and classify numeric values. Additionally, limited number of dataset instances could not give high classification performances, i.e., accuracies less than 70%.

Kanwal Noor and Sadaqat Jan [5] proposed 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 method. 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%.

Motivation for the 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 skillset in solving a real time problem has been the primary motivation.

Data needed for this project is require to scrap from 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 tells a used car's market value.

Analytical Problem Framing

Mathematical / Analytical Modelling of theProblem

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

Data Sources and their formats

Data is collected from cardheko.com using selenium and saved in CSV file.

```
# Importing dataset excel file using pandas.
df= pd.read_excel('Final Used Car Scrap data .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 : 10544
No. of Columns : 24
```

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

```
# As we have 24 columns lets sort columns by their datatype
df.columns.to_series().groupby(df.dtypes).groups
```

```
{int64: ['Make Year'], float64: ['No of Cylinder'], object: ['Car Model', 'Fuel Type', 'KMs driven', 'Engine Displacement(CC)',
'Transmission', 'Milage(kmpl)', 'Max Power(bhp)', 'Torque(Nm)', 'Seating Capacity', 'Color', 'Gear Box', 'Steering Type', 'Fron
t Brake Type', 'Rear Brake Type', 'Tyre Volume', 'Engine Type', 'Turbo Charger', 'Super Charger', 'Length(mm)', 'Width(mm)', 'H
eight(mm)', 'Price(Rs)']}
```

Price is our Target variable which is to be predicted. We can see that some of numerical variable like torque, milage are by default come with object datatypes.

Data Pre-processing

The dataset is large and it may contain some data error. In order to reach clean, error free data some pre-processing is done on data. First task is to finding error in data or data entry correction. Second task is to

convert datatypes in appropriate datatypes. 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 datatypes and Value are in format like 2,36,000 KMs. Here comma (' , ') is

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

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')
```

removed from data and converted into numeric datatypes.

- Some of data in 'Engine Displacement (CC)' comes with CC notation attached to value, for example, 1497 CC. So, CC is strip away and subsequently feature converted into numeric datatype.

```
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 Milage contain dash (-) and some of values in unit of km/kg or km/hr which need to converted into standard unit kmpl. This conversion in proper unit is done and finally milage

datatypes is converted into numeric datatypes.

```
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')
```

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 into object datatype. The conversion into appropriate numeric value requires for such entries which resolved using 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[['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 car is since first sold out by manufacturer?

```
df['Car_Age'] = 2021 - df['Make Year']
```

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

- The car model name we extracted from website contain both car brand and its variant. Some feature engineering performs to extract these additional features out of original car model feature using following piece of code.

```
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(',',' ')
```

```
df.drop(columns = 'Car Model', inplace = True)
```

Final datatypes of all feature after converting them into appropriate datatypes is shown in figure below:

```
# As we have 24 columns lets sort columns by their datatype after conversion into appropriate datatypes
df.columns.to_series().groupby(df.dtypes).groups
```

```
{int64: ['KMs driven', 'Engine Displacement(CC)', 'Car_Age'], float64: ['Milage(kmpl)', 'Max Power(bhp)', 'Torque(Nm)', 'Seatin
g Capacity', 'No of Cylinder', 'Length(mm)', 'Width(mm)', 'Height(mm)', 'Price (Rs.)'], object: ['Fuel Type', 'Transmission',
'Color', 'Gear Box', 'Steering Type', 'Front Brake Type', 'Rear Brake Type', 'Tyre Volume', 'Engine Type', 'Turbo Charger', 'Su
per Charger', 'Car_Brand', 'Car_Model']}
```

In next phase of data pre-processing involves data integrity check to identify presence of duplicates, whitespaces, null value.

Around 1658 duplicate entry exist in dataset and eventually these duplicate entries are drop out.

```
df.isin(['????','?????','-','null','NA',' ']).sum().any()
```

```
True
```

```
df.replace('-',np.nan, inplace = True)
df.replace('null',np.nan, inplace= True)
df.replace('????', np.nan, inplace = True)
df.replace('?????', np.nan, inplace = True)
df.replace(' ', np.nan, inplace = True)
```

After those values with entry like whitespaces, null, '?' are replace with np.nan. we are going to treat those finding as missing value.

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

```
df.duplicated().sum() # This will detect duplicate entries in dataset
```

```
1658
```

- **Dropping duplicates entries from dataset.**

```
# Droping duplicate entries
df.drop_duplicates(keep='last', inplace =True)
```

```
df.shape
```

```
(8886, 25)
```

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

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

There is correction in subcategories of Super Charger.

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

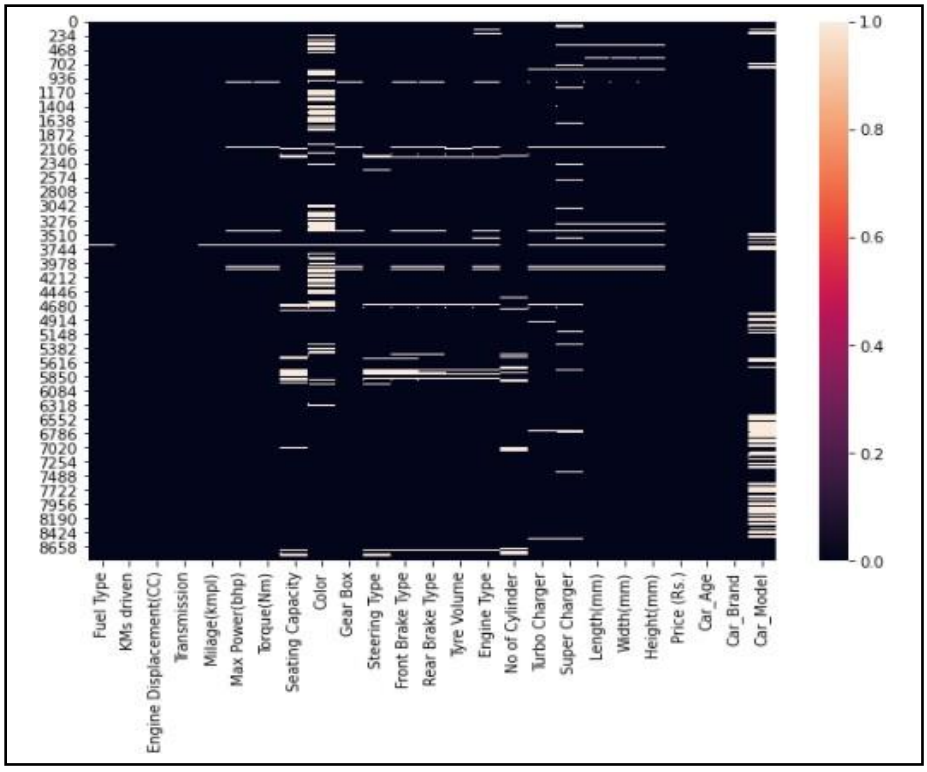
```
df['Super Charger'].value_counts()
```

```
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
- Gearbox
- Tyre Volume

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

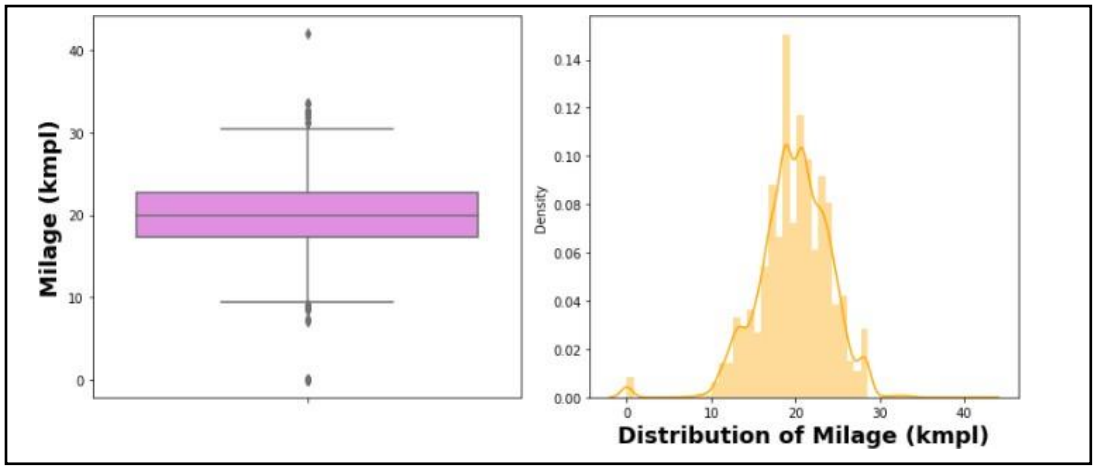


Lot features contain missing value. Categorical value can be imputation done with mode of categories.

```
# Imputation of Categorical variable or ordinal variable with mode of category
df['Fuel Type'].fillna(df['Fuel Type'].mode()[0],inplace = True)
df['Seating Capacity'].fillna(df['Seating Capacity'].mode()[0],inplace = True)
df['Color'].fillna(df['Color'].mode()[0],inplace = True)
df['Gear Box'].fillna(df['Gear Box'].mode()[0],inplace = True)
df['Steering Type'].fillna(df['Steering Type'].mode()[0],inplace = True)
df['Front Brake Type'].fillna(df['Front Brake Type'].mode()[0],inplace = True)
df['Rear Brake Type'].fillna(df['Rear Brake Type'].mode()[0],inplace = True)
df['Color'].fillna(df['Color'].mode()[0],inplace = True)
df['Tyre Volume'].fillna(df['Tyre Volume'].mode()[0],inplace = True)
df['Engine Type'].fillna(df['Engine Type'].mode()[0],inplace = True)
df['No of Cylinder'].fillna(df['No of Cylinder'].mode()[0],inplace = True)
df['Turbo Charger'].fillna(df['Turbo Charger'].mode()[0],inplace = True)
df['Super Charger'].fillna(df['Super Charger'].mode()[0],inplace = True)
```

```
df['Car_Model'].fillna('Unknown',inplace = True)
```

Numerical value can be imputed with mean or median depending on sensitive to outliers. For example, in this project imputation of missing value can be done after examine 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
```

Milage (kmpl) is almost symmetrical in nature. Outliers are also spread to both lower & upper bound.

So, We will be imputate Milage (kmpl) with mean.

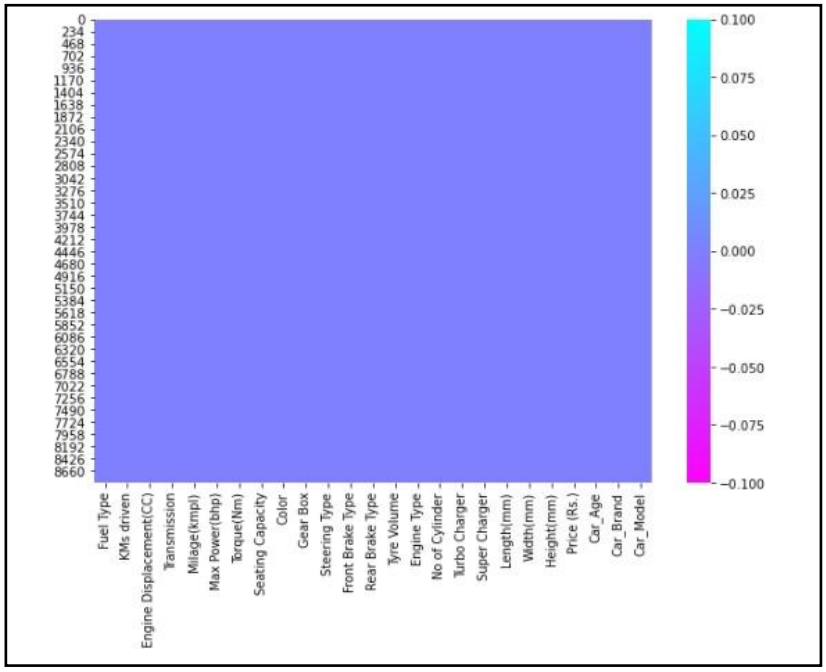
```
df['Milage(kmpl)'].fillna(df['Milage(kmpl)'].mean(), inplace =True)
```

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

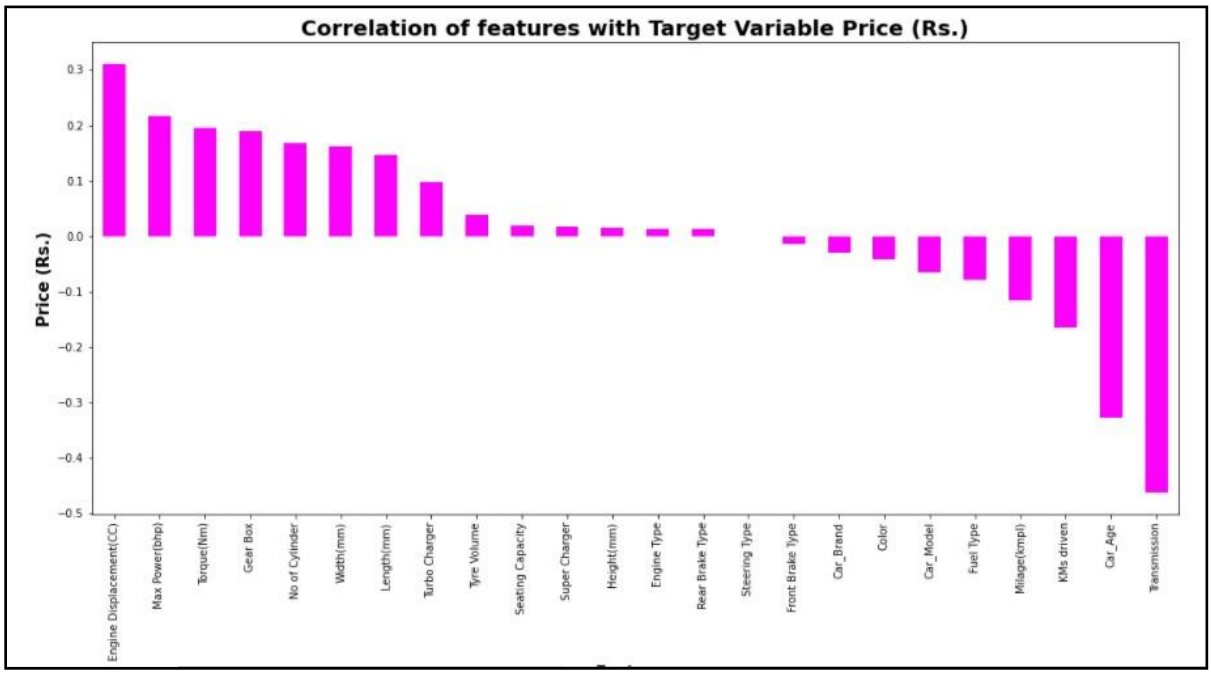


```
# Using Label encoder for transforming Categorical data
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
for i in Categorical:
    df[i] = le.fit_transform(df[i])
df.head()
```

Label Encoding of categorical data:

4. Data Inputs- Logic- Output Relationships

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



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

Hardware & Software Requirements with ToolUsed

Hardware Used -

1. Processor — Intel i3 processor with 2.4GHZ
2. RAM — 4 GB
3. GPU — 2GB AMD Radeon Graphics card Software utilised -

1. Anaconda – Jupyter Notebook
2. Selenium - Webscraping
3. Google Colab – for Hyper parameter tuning

Libraries Used –

4. Different libraries used for web scraping as follow:

```
import pandas as pd
import numpy as pd
import time
import selenium
from selenium import webdriver
from selenium.common.exceptions import StaleElementReferenceException, NoSuchElementException
```

Different libraries are used while building ML model and Visualisation of data

```
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
```

```
import pandas as pd # for data wrangling purpose
import numpy as np # Basic computation library
import seaborn as sns # For Visualization
import matplotlib.pyplot as plt # ploting package
%matplotlib inline
import warnings # Filtering warnings
warnings.filterwarnings('ignore')
```

Models Development &Evaluation

4. IDENTIFICATION OF POSSIBLE PROBLEM-SOLVING APPROACHES (METHODS)

First part of problem solving is to scrap data from cardheko.com website which we already done. Second part of problem building machine learning model to predict price of used car. This become regression problem which can be solved using different regression-based algorithm. Out of them best model can be tuned using hyper parameter tuning to enhance R2 score of best models. At end we will save our final model using joblib.

5. 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 first part Scraping URL of Used car for different location in India.
3. Storing Scrap URL in excel file.
4. Selecting car feature to be scrap from website.
5. In second part Scraping data from individual URL 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

6. KEY METRICS FOR SUCCESS IN SOLVINGPROBLEM 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 accurate our model predict result, is going to important evaluation criteria along with Cross validation score.

4. RUN AND EVALUATE SELECTED MODELS

1. 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, V, 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: 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
```

Random forest Regressor gives R2 score of 96.46 %.

k-5 fold cross validation perform which gives CV score as 93.03 %.

Finally difference between R2 and CV Score 3.43 %.

1. 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: 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
```

2. Bagging Regressor using Machine Learning Techniques

```
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: 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
```

All model is giving us R2 score & Cross validation Score more than 90%, So we will select model which has less difference between these scores. On Basis of difference between R2 Score and Cross Validation Score Decision Tree Regressor is selected as best model with 91.79% r2_score. We will perform Hyper Parameter tuning over this model. We got best parameter for final model.

```
DecisionTreeRegressorGCV.fit(X_train,y_train)
```

```
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']})
```

Tunning the model using GCV.

```
GCV.best_params_
```

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

We got best parameter for final model.

Final Model

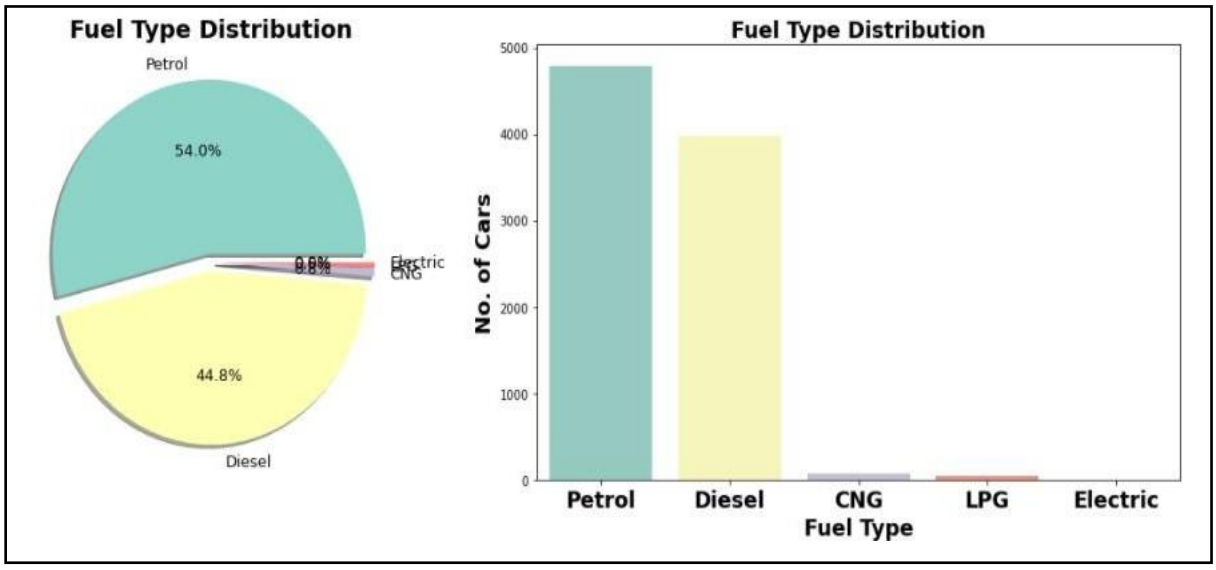
```
Final_mod=DecisionTreeRegressor(criterion='friedman_mse', max_depth=15, max_features='auto',
                                min_samples_split=4, splitter='random')
Final_mod.fit(X_train,Y_train)
pred=Final_mod.predict(X_test)
print('R2_Score:',r2_score(Y_test,pred)*100)
print('mean_squared_error:',metrics.mean_squared_error(Y_test,pred))
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: 92.28906442588051
mean_squared_error: 20125176994.634956
mean_absolute_error: 70467.88619495885
RMSE value: 141863.2334138587
```

Final Model is giving us R2 Score of 92.29% which is slightly improved compare to earlier R2 score of 91.79%.

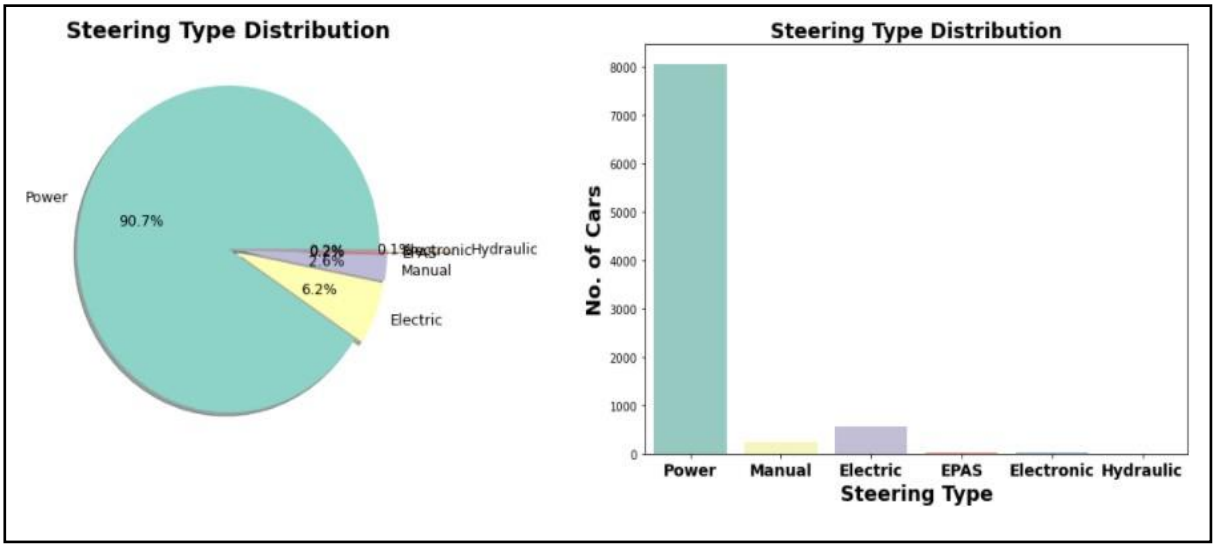
VISUALIZATIONS

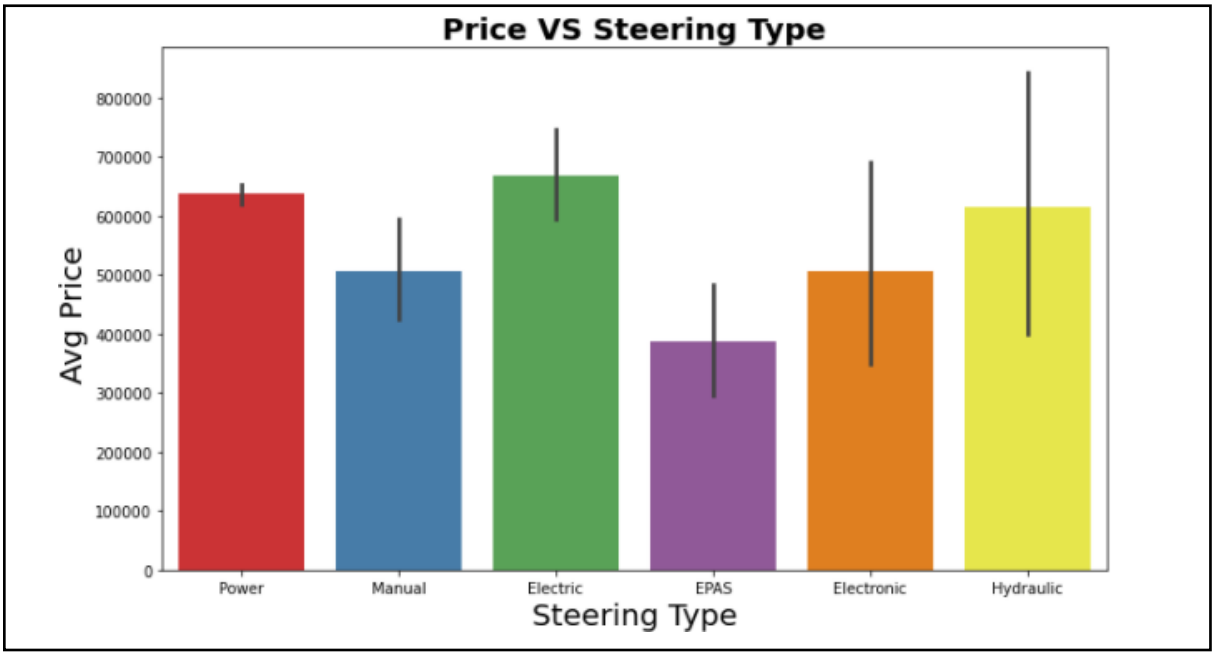
Let see key result from EDA, start with fuel type.



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

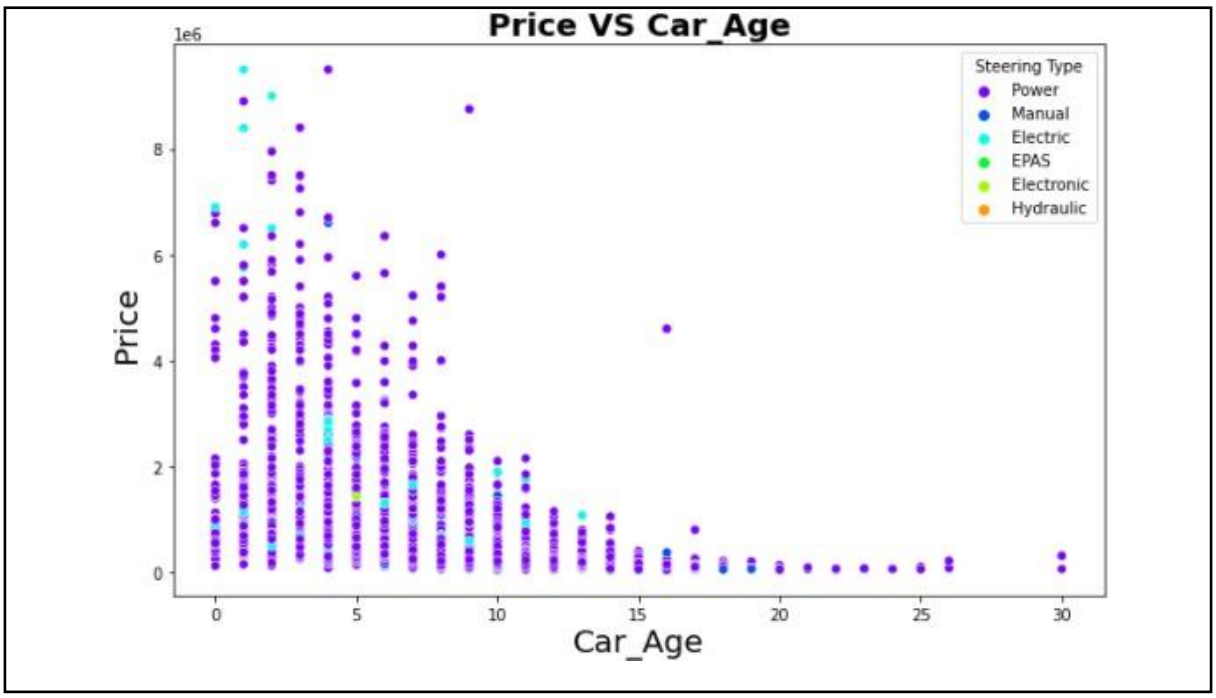
➢ Price Vs Steering Type



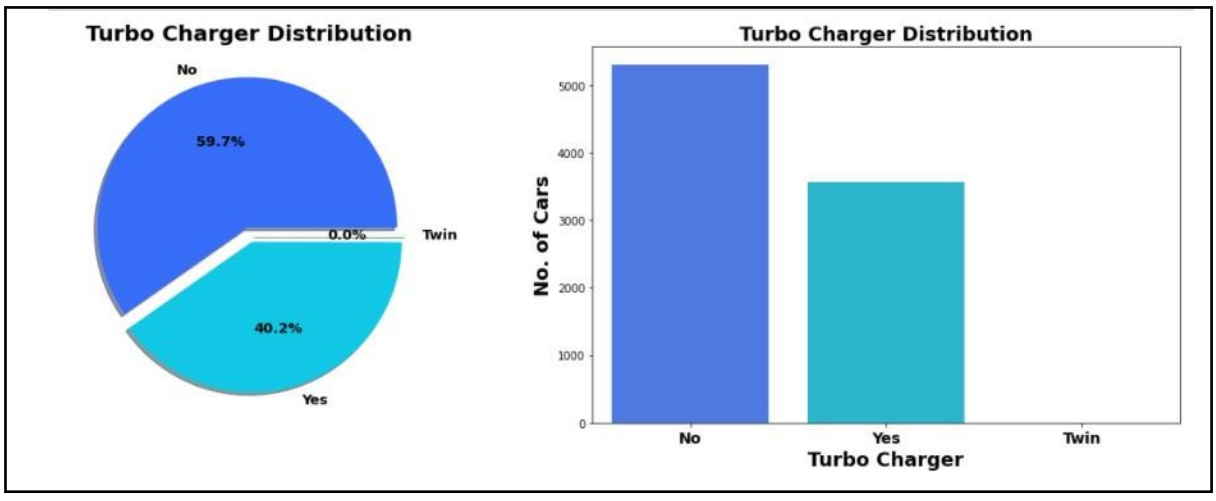


- 6.2% car based on electric steering, which is costly compare to others.
- Very small section of car still uses Manual Steering, Most probably they belong to old model.

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

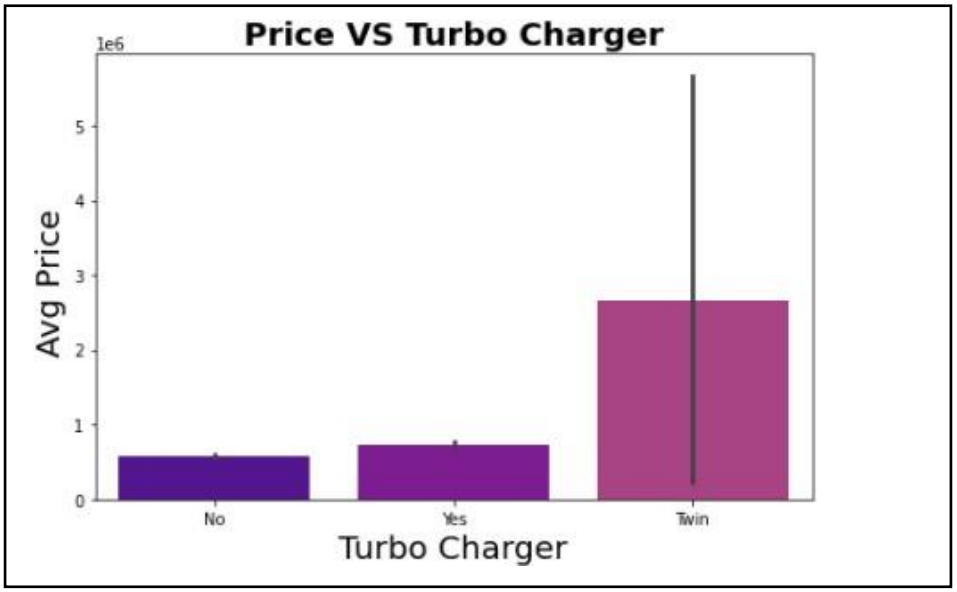


Here we got confirmation of prediction in previous section, almost all manual steering-based car at least 10-year-old.

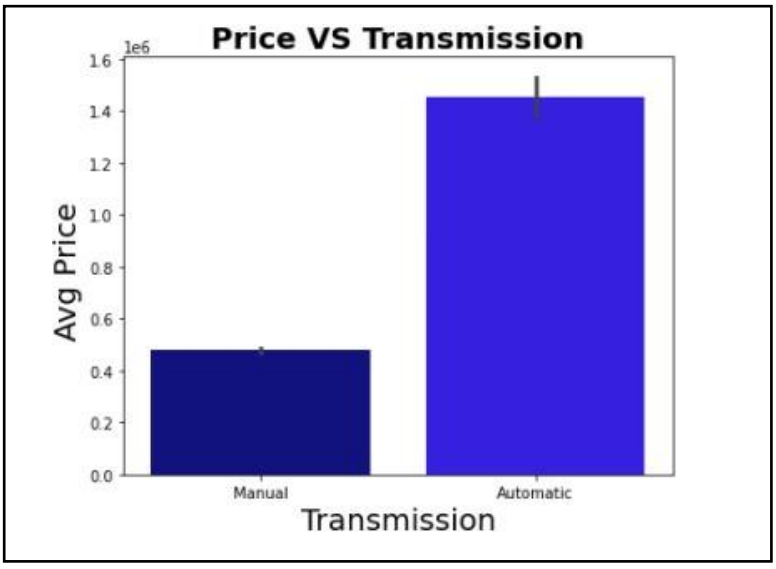


40% cars are with turbo charger & almost less than 1 % car with twin

facility



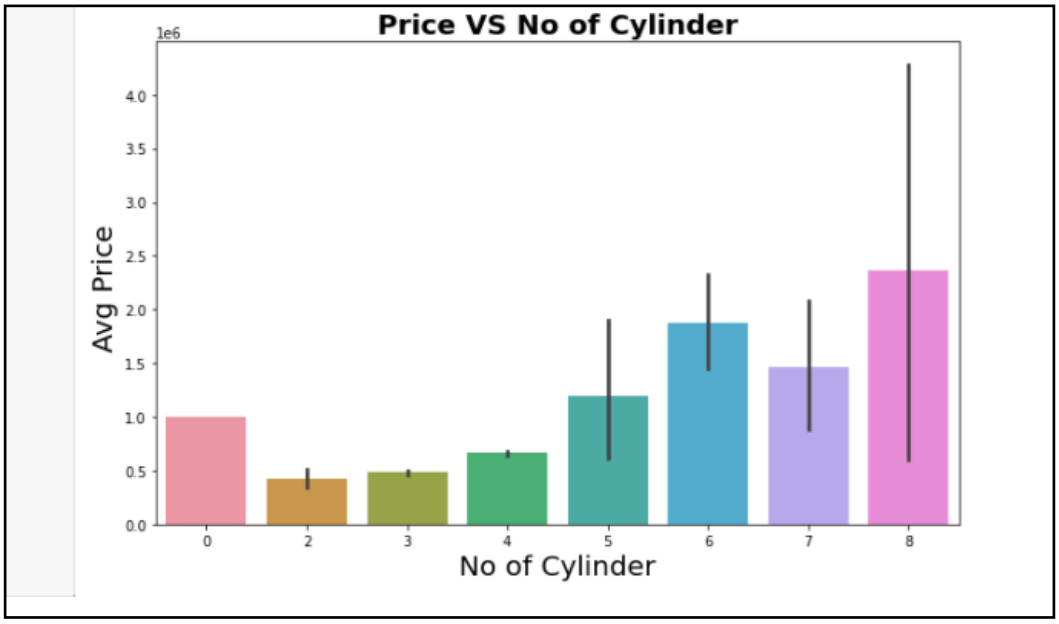
As expected, Max price for car based on Twin engine followed by with turbocharger



Observation:

- Most of car are with manual transmission.
- The price of Automatic transmission is much greater than manual transmission.

Let check effect of number of cylinders on price

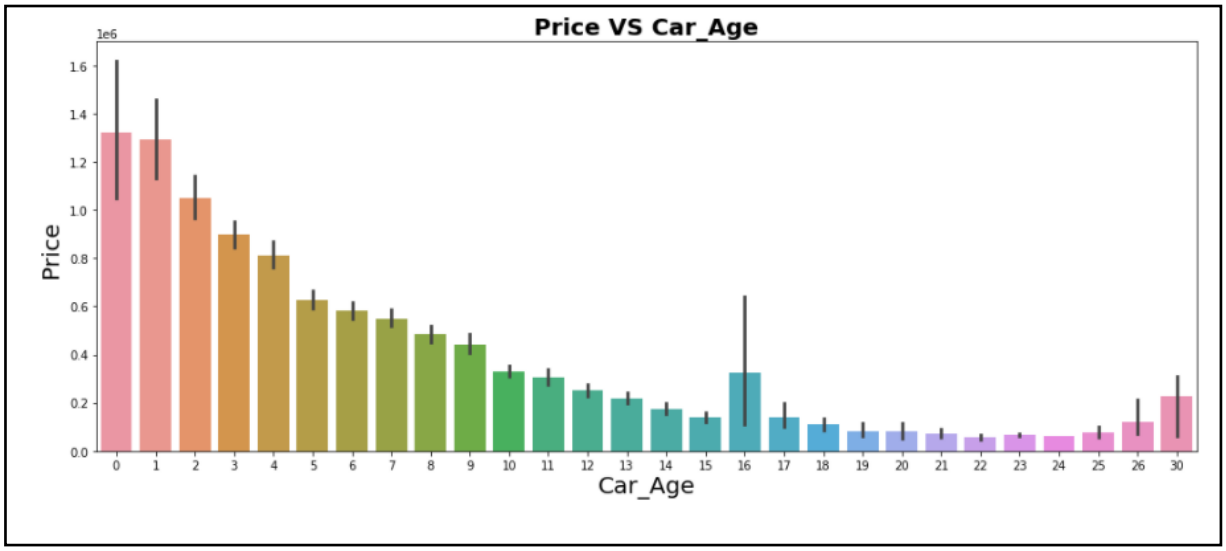


Observation:

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

We know that as car price decrease as car model become older. Let’s

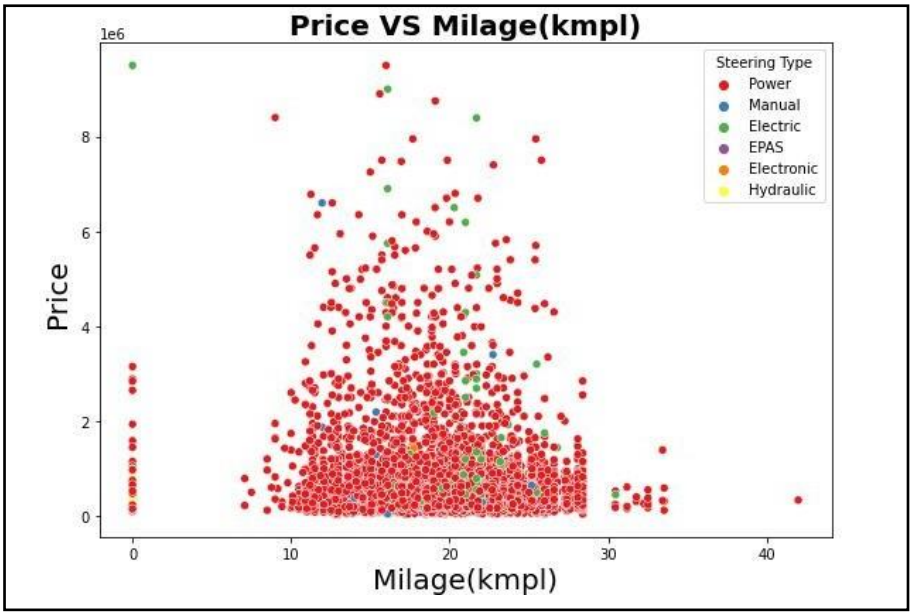
plot bar plot of Price vs Car age to verify it.



We know that as car price decrease as car model become older. Let’s

plot bar plot of Price vs Car age to verify it.

As car get older, price of car deprecates



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

Interpretation of the Results

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

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

Conclusion

Key Findings and Conclusions of the Stud

Algorithm	R2 Score	CV Score	R2 Score - CV Score
Random Forest Regressor	96.46	93.03	3.4337
XGB Regressor	96.85	93.24	3.6109
Gradient Boosting Regressor	94.93	90.19	4.7391

Decision Tree Regressor	91.79	88.83	2.955
Bagging Regressor	95.67	92.60	3.072

- On Basis of difference between R2 Score and Cross Validation Score Decision Tree Regressor is selected as best model with 91.79% R2_score.
  - Final Model is giving us R2 Score of 92.29% which is slightly improved compare to earlier R2 score of 91.79%.

Learning Outcomes of the Study in respect of Data Science

- Scraping data for project from www.cardheko.com. This first of such kind of project for me. Web scraping such 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 lot discrepancies in data scrap with different units, different names for same sub-categories. Data cleaning was big part of this project.

Limitations of this work and Scope for FutureWork

- 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 predication.
- Here we Scrap almost 24 features. But there are many different kinds of safety, comfort, entertainment features to which buyer weight while buying car. We can also include such more feature in future.