CAR PRICE PREDICTION

Submitted by-:
NITISH KUMAR SHARMA

ACKNOWLEDGEMENT

It is great pleasure for me to undertake this project. I feel overwhelmed doing this project entitled – "Car Price Prediction".

Some of the resources that helped me to complete this project are as follows:

- Internet/web
- Stack overflow
- Analytics Vidhya
- Coding Ninjas
- Articles published in Medium.com
- Wikipedia

INTRODUCTION

BUSINESS PROBLEM FRAMING

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.

Due to increased price of new cars and financial incapability of customers who cannot afford one. So they look for used cars in the market according to their budget. But sellers has no idea about the car's existing value or the price he should be selling the car at.

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.

CONCEPTUAL BACKGROUND OF THE DOMAIN PROBLEM

The used car market is witnessing a boom around the globe. 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. 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 it's 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 we have developed a model which will be highly effective.
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. User Interface has also been developed which acquires input from any user and displays the Price of a car according to user's inputs.

REVIEW OF LITERATURE

Used cars are big business and it has been growing for many years. hence it becomes one of the prime fields to apply the concepts of machine learning to optimize and predict the prices with high accuracy.

Collected new raw dataset from different car websites. 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.

We have used different machine learning models to predict the above. Since we have continous target data so regression model algorithms has been used.

We will start our project with the train dataset which contains price as our target variable along with other independent variables. We will observe all the variables with following goals in mind:

- Relevance of the variables
- Distribution of the variables
- Data Cleaning of the variables
- Visualization of the variables
- Visualization of the variables as per target variable for data analysis

After having gone through all the variables and cleaning the dataset, we will move on to machine learning regression modelling:

- Pre-processing of the dataset for models
- Testing multiple algorithms with multiple evaluation metrics
- Select evaluation metric as per our specific business application
- Doing hyper-parameter tuning using GridSearchCV for the best model
- Finally saving the best model
- Loading and predicting with the loaded model

MOTIVATION FOR THE PROBLEM UNDERTAKEN

This project deals in used-car price prediction whose market is thriving. Through my analysis on the data it will be somehow able to decide whether a used car is worth the posted price or not as several factors including mileage, model, make year, kilometres driven, no of owners etc. can influence the actual worth of a car.

With more and more analysis on the data, it will improve my knowledge about this particular domain. It will be very productive for me to know how the features/variables present in the data are affecting the target variable, so that we can evaluate price of cars effectively & precisely based on the data collected.

ANALYTICAL PROBLEM FRAMING

MATHEMATICAL / ANALYTICAL MODELING OF THE PROBLEM

The dataset contains csv file . Train data has 14 attributes (13 variables and 1 target variable). The target (price) variable is a continous data. Thus it's a regression problem. The other variables are model, brand, variant, fuel type, make year, no of owners, transmission type, torque, maximum power, mileage, engine displacement, kilometres driven and location.

To know the overview/stats of the dataset , we will be using df.describe() function which gives informations like count, min, max, mean, standard deviation values of features . By plotting of heat map we can correlate features if they are highly inter-correlated or not. So we can drop columns if problem of multicollinearity arises (if vif >10) when we observe through variance inflation factor.

From an initial statistical overview of the dataset, we infer that variables like mileage, maximum power and torque has zero values in their minimum category which is not possible. So we have to deal with it by applying median/ mean value to it.

DATA SOURCES AND THEIR FORMATS

The train data that was in CSV format (Comma Separated Values). After reading the data by using function df=pd.read_csv (‘ ---file path --- ’) in jupyter notebook there were 5072 rows and 14 columns.

Dataset datatypes are as follow :

```
df.info() # to know datatype of each columns
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5072 entries, 0 to 5071
Data columns (total 14 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Model         5072 non-null   object 
 1   brand         5072 non-null   object 
 2   variant       5072 non-null   object 
 3   mfg           5072 non-null   int64  
 4   km_drvn       5065 non-null   float64
 5   fuel          5072 non-null   object 
 6   mileage       5072 non-null   float64
 7   engine        5072 non-null   int64  
 8   max_pwr       5071 non-null   float64
 9   torque        5062 non-null   float64
 10  owner         5072 non-null   object 
 11  transmission  5072 non-null   object 
 12  location      5072 non-null   object 
 13  Price         5072 non-null   int64  
dtypes: float64(4), int64(3), object(7)
memory usage: 554.9+ KB
```

- Count is not same for Km_drvn, max_pwr & torque. So there will be missing values.

```
# to count number of unique values in each columns
df.nunique()
```

```
Model            606
brand             29
variant         1397
mfg               23
km_drvn         2413
fuel               5
mileage          385
engine           125
max_pwr          325
torque           232
owner              9
transmission      20
location          13
Price           1509
dtype: int64
```

Categorical & Continous features in the train dataset are as follows :

```
# to list categorical features in our dataset
cat_features=[i for i in df.columns if df.dtypes[i]=='object']
cat_features
```

```
['Model', 'brand', 'variant', 'fuel', 'owner', 'transmission', 'location']
```

```
# to list continous features in our dataset
con_features=[i for i in df.columns if df.dtypes[i]=='int64' or df.dtypes[i]=='float64']
con_features
```

```
['mfg', 'km_drvn', 'mileage', 'engine', 'max_pwr', 'torque', 'Price']
```

- There are seven categorical & continuous variables present inside the dataset.

DATA PREPROCESSING DONE

- Data containing null values was visualized by is null().sum() function.

```
df.isnull().sum()  # to check null values
```

```
Model            0
brand            0
variant          0
mfg              0
km_drvn          7
fuel             0
mileage          0
engine           0
max_pwr          1
torque          10
owner            0
transmission     0
location         0
Price            0
dtype: int64
```

- Only three variables contained null values from data.

```
# treating null values
df['km_drvn']=df['km_drvn'].fillna(df['km_drvn'].median())
df['max_pwr']=df['max_pwr'].fillna(df['max_pwr'].median())
df['torque']=df['torque'].fillna(df['torque'].median())
```

- Treated the null values by filling it up by their median value.

```
# to count sum of 0's present
print('Total no of zeros in mileage is:',sum(df['mileage']==0))
print('Total no of zeros in max_pwr is:',sum(df['max_pwr']==0))
print('Total no of zeros in torque is:',sum(df['torque']==0))
```

- There were 19, 21 & 1 zeros present in mileage, max_pwr & torque variables.

```
# filling zero with median value
df['mileage']=df['mileage'].replace(0,df['mileage'].median())
df['max_pwr']=df['max_pwr'].replace(0,df['max_pwr'].median())
df['torque']=df['torque'].replace(0,df['torque'].median())
```

- So filled the zeros with their respective median values.

- To check unique elements present in transmission column

```
# to find unique elements
df['transmission'].unique()
```

```
array(['Manual', 'Automatic', 'RTO\nDL6C', 'RTO\nDL3C', 'RTO\nUP14',
       'RTO\nDL5C', 'RTO\nDL9C', 'RTO\nDL7C', 'RTO\nDL12', 'RTO\nDL2C',
       'RTO\nDL4C', 'RTO\nHR87', 'RTO\nDL1C', 'RTO\nHR29', 'RTO\nHR26',
       'RTO\nDL10', 'RTO\nHR51', 'RTO\nUP16', 'RTO\nDL8C', 'RTO\nHR77'],
      dtype=object)
```

- So we require only manual & transmission, rest must be corrected.

```
# Appending different values in the column into one value having maximum counts
df['transmission']=df['transmission'].replace(['RTO\nDL9C','RTO\nHR87','RTO\nDL3C','RTO\nHR26',
                                               'RTO\nDL6C','RTO\nDL10','RTO\nDL4C','RTO\nUP16','RTO\nHR29',
                                               'RTO\nUP14','RTO\nDL8C','RTO\nDL7C','RTO\nDL1C','RTO\nHR77','RTO\nDL5C',
                                               'RTO\nHR51','RTO\nDL2C','RTO\nDL12'],'Manual',regex=True)
```

- Hence appending all the unknown values into manual which was having the maximum count.

```
# to find unique elements
df['owner'].unique()
```

```
array(['First Owner', 'Second Owner', 'Third Owner',
       'Fourth & Above Owner', 'Test Drive Car', '1st Owner', '2nd Owner',
       '3rd Owner', '4th Owner'], dtype=object)
```

- The elements above like (First Owner – 1st Owner) represents the same thing, rest all are like that.

```
# Appending duplicate values into one
df['owner']=df['owner'].replace('First Owner','1st Owner',regex=True)
df['owner']=df['owner'].replace('Second Owner','2nd Owner',regex=True)
df['owner']=df['owner'].replace('Third Owner','3rd Owner',regex=True)
df['owner']=df['owner'].replace('Fourth & Above Owner','4th Owner',regex=True)

# Appending different values in the column into one value having maximum counts
df['owner']=df['owner'].replace('Test Drive Car','1st Owner',regex=True)
```

- So appending the like terms into one and the rare element (Test Drive Car) to the value having maximum counts in the column.

DATA INPUTS- LOGIC- OUTPUT RELATIONSHIPS

When visualizing the feature variables with the target (price) variable, It gave me insights that how these variables are so important for making used car price predictions.

Lot of variables given here explains that how price increases/decreases according to the location, brand, number of owners, fuel type etc of used cars. Which type of transmission type has maximum price value whether its manual or automatic. What bodytype is now in trend with its expected price range like hatchback, sedan, wagon, couple, SUV, MUV etc.
With increase in maximum power & torque the mileage decreases.

STATE THE SET OF ASSUMPTIONS (IF ANY) RELATED TO THE PROBLEM UNDER CONSIDERATION

```
df.describe().T # to get high understanding of dataset or to get overview/stats of the dataset
```

	count	mean	std	min	25%	50%	75%	max
mfg	5072.0	2015.049093	3.182132	1989.0	2013.00	2016.00	2017.00	2021.00
km_drvn	5065.0	54321.217572	39144.378292	1001.0	29867.00	49000.00	72000.00	1019000.00
mileage	5072.0	19.853760	4.055172	0.0	17.21	20.14	22.54	33.54
engine	5072.0	1424.110410	462.910685	72.0	1197.00	1248.00	1498.00	5461.00
max_pwr	5071.0	97.742749	41.697566	0.0	74.00	85.80	108.45	575.00
torque	5062.0	173.988123	104.090620	0.0	110.00	145.00	204.00	850.00
Price	5072.0	729037.054811	820864.945625	40000.0	371000.00	525000.00	725000.00	11700000.00

- In min category – engine, torque and max_pwr has zero value which is not possible because it must have some value as for running vehicles these are important factors.

HARDWARE / SOFTWARE REQUIREMENTS AND TOOLS USED

- Anaconda Navigator 1.10.0
- Jupyter Notebook 6.1.4 , Python 3

Libraries

```
import pandas as pd # for handling dataset
import numpy as np  # for mathematical computation

from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split,GridSearchCV,cross_val_score
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report, plot_roc_curve, roc_auc_score
from sklearn.ensemble import RandomForestRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.neighbors import KNeighborsRegressor
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
from sklearn.preprocessing import LabelEncoder
from scipy.stats import skew

# for visualization
import plotly.express as px
import matplotlib as mpl
import matplotlib.pyplot as plt
%matplotlib inline
from matplotlib import style
import seaborn as sns

# for saving & loading model
import pickle

import warnings
warnings.filterwarnings('ignore')
```

```
# converting objects into integers
lab_enc = LabelEncoder()
list1 = ['Model', 'brand', 'variant', 'fuel','owner', 'transmission', 'location']
for val in list1:
    df[val] = lab_enc.fit_transform(df[val].astype(str))
```

Libraries and Packages used:

- Pickle for saving & loading machine learning model.
- GridSearchCV for Hyper-parameter tuning.
- Cross validation score to cross check if the model is overfitting or not.
- Label Encoder to convert objects into integers for categorical variables.
- Seaborn and matplotlib for visualization.

MODEL/S DEVELOPMENT AND EVALUATION

IDENTIFICATION OF POSSIBLE PROBLEM-SOLVING APPROACHES (METHODS)

As we have to predict price of used cars with variables containing both categorical and continuous values but our target variable was of continuous data. Thus it's a regression problem and we have to apply regression algorithms and build a model giving good performance.

Removed some duplicate variables that represented same thing, thereby minimizing runtime of system.

Plotted heatmap to visualize which variables were showing good positive or negative correlation with the target variable. So that it could help us in further analysis.

Checked for skewness, outliers and handled them with 1.5 IQR method. Dropped variables that were highly correlated with each other to remove multicollinearity.

TESTING OF IDENTIFIED APPROACHES (ALGORITHMS)

Algorithms used:

- Linear Regression
- Decision Tree Regressor
- Random Forest Regressor
- KNN Regressor
- Gradient Boosting Regressor
- XGB Regressor

1. Logistic Regression

```
regression = LinearRegression()
regression.fit(x_train,y_train)
```

```
LinearRegression()
```

```
# Adjusted R2 score
regression.score(x_train,y_train)
```

```
0.5665753405656866
```

```
# To cross verify
y_pred = regression.predict(x_test)
```

```
r2_score(y_test,y_pred)
```

Training Result : 56.65% Test Result : 52.15%

2. Decision Tree Regressor

```
dt_reg = DecisionTreeRegressor()
dt_reg.fit(x_train,y_train)

print("\n===================Train Result=====================")

print(f"Accuracy Score: {dt_reg.score(x_train,y_train) * 100:.2f}%")
print("_____________________________________________")

#********************** Test Score ************************

y_pred = dt_reg.predict(x_test)
print("\n===================Test Result=====================")
print(f"Accuracy Score: {dt_reg.score(x_test,y_test) * 100:.2f}%")
print("_____________________________________________")
```

Training Result : 100% Test Result : 62.23%

3. Random Forest Regressor

```
rand_reg = RandomForestRegressor(n_estimators = 100,random_state=51)
rand_reg.fit(x_train,y_train)

print("\n===================Train Result=====================")

print(f"Accuracy Score: {rand_reg.score(x_train,y_train) * 100:.2f}%")
print("_____________________________________________")

#********************** Test Score ************************

y_pred = rand_reg.predict(x_test)
print("\n===================Test Result=====================")
print(f"Accuracy Score: {rand_reg.score(x_test,y_test) * 100:.2f}%")
print("_____________________________________________")
```

Training Result : 97.75% Test Result : 70.74%

4. KNN Regressor

```
knn=KNeighborsRegressor(n_neighbors=3)
knn.fit(x_train,y_train)

print("\n====================Train Result======================")

print(f"Accuracy Score: {knn.score(x_train,y_train) * 100:.2f}%")
print("_____________________________________________")

#********************** Test Score ************************

y_pred = knn.predict(x_test)
print("\n====================Test Result======================")
print(f"Accuracy Score: {knn.score(x_test,y_test) * 100:.2f}%")
print("_____________________________________________")
```

Training Result : 91.23% Test Result : 60.11%

5. Gradient Boosting Regressor

```
from sklearn.ensemble import GradientBoostingRegressor
np.random.seed(42)
gbr = GradientBoostingRegressor()
gbr.fit(x_train, y_train)

print("\n====================Train Result======================")

print(f"Accuracy Score: {gbr.score(x_train,y_train) * 100:.2f}%")
print("_____________________________________________")

#********************** Test Score ************************

y_pred = gbr.predict(x_test)
print("\n====================Test Result======================")
print(f"Accuracy Score: {gbr.score(x_test,y_test) * 100:.2f}%")
print("_____________________________________________")
```

Training Result : 89.88% Test Result :71.39%

6. XGB Regressor

```
from xgboost.sklearn import XGBRegressor
np.random.seed(42)
xgb = XGBRegressor()
xgb.fit(x_train, y_train)

print("\n====================Train Result=====================")

print(f"Accuracy Score: {xgb.score(x_train,y_train) * 100:.2f}%")
print("_____________________________________________")

#********************** Test Score ************************

y_pred = xgb.predict(x_test)
print("\n====================Test Result=====================")
print(f"Accuracy Score: {xgb.score(x_test,y_test) * 100:.2f}%")
print("_____________________________________________")
```

Training Result : 99.83% Test Result : 72.59%

KEY METRICS FOR SUCCESS IN SOLVING PROBLEM UNDER CONSIDERATION

Precision: It is the ratio between the True Positives and all the Positives. It can be seen as a measure of quality, higher precision means that an algorithm returns more relevant results than irrelevant ones

Recall : It is the measure of our model correctly identifying True Positives. It is used as a measure of quantity and high recall means that an algorithm returns most of the relevant results.

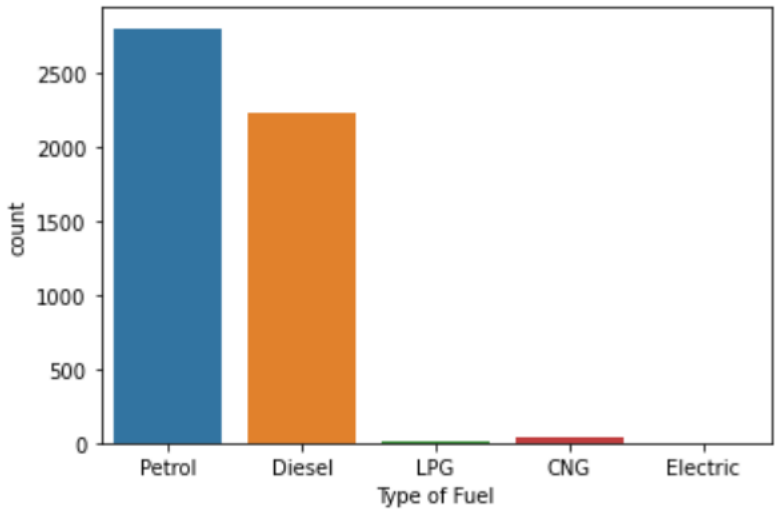
Accuracy score : It is the ratio of the total number of correct predictions and the total number of predictions. It is used when the True Positives and True negatives are more important. Accuracy can be used when the class distribution is similar

F1-score : It is used when the False Negatives and False Positives are crucial. Hence F1-score is a better metric when there are imbalanced classes.

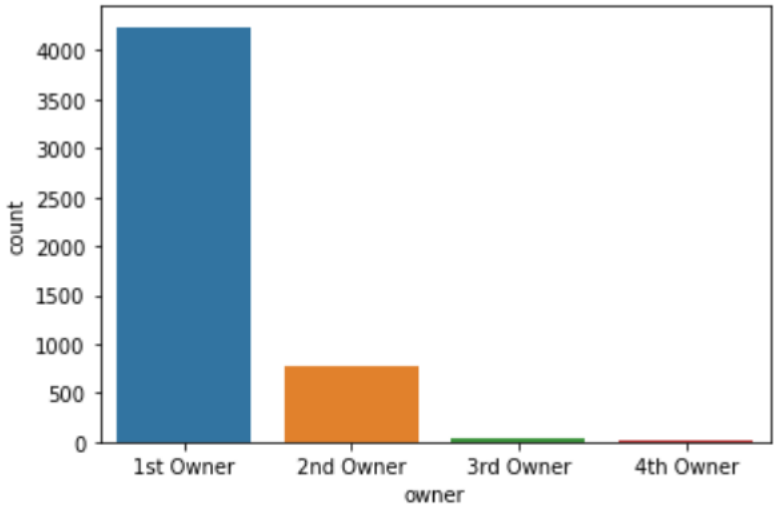
Cross_val_score : To run cross-validation on multiple metrics and also to return train scores, fit times and score times. Get predictions from each split of cross- validation for diagnostic purposes. Make a scorer from a performance metric or loss function.

roc _auc _score : ROC curve. It is a plot of the false positive rate (x-axis) versus the true positive rate (y-axis) for a number of different candidate threshold values between 0.0 and 1.0

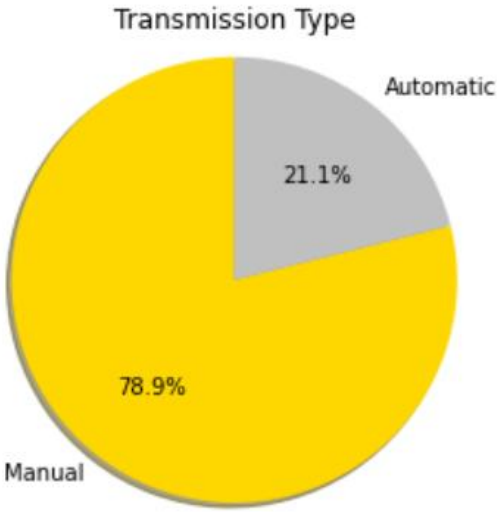
VISUALIZATIONS



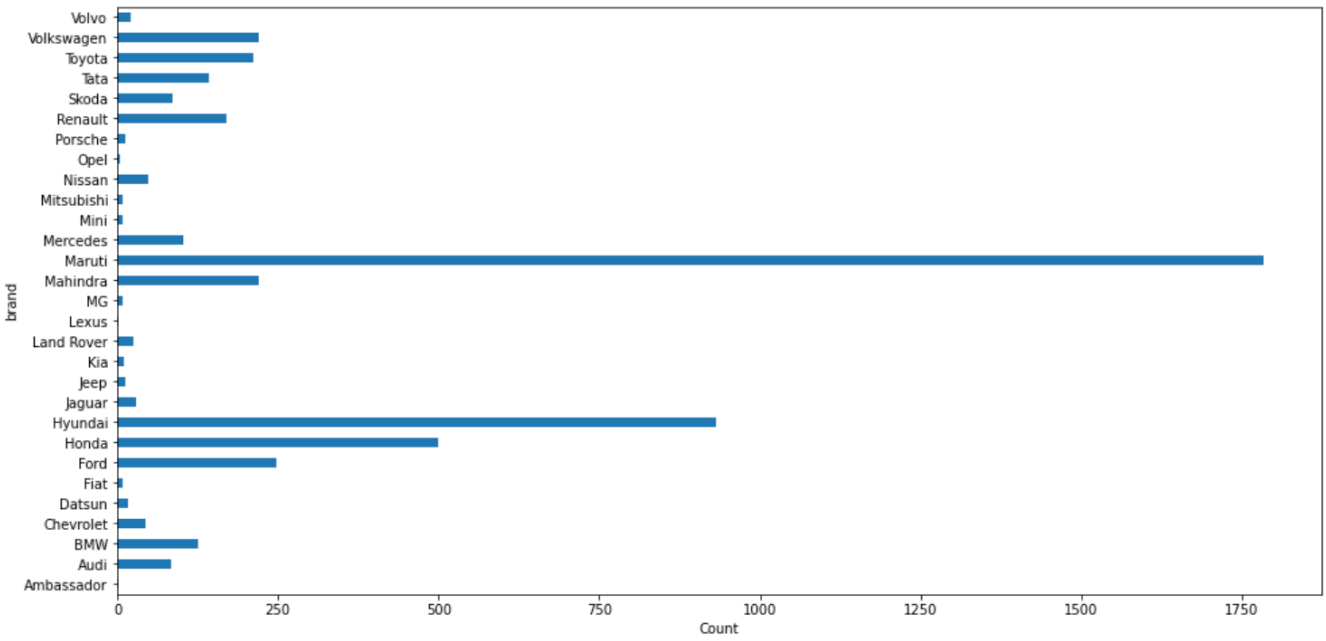
- Mostly used cars in the data is having petrol type of fuel then followed by diesel.



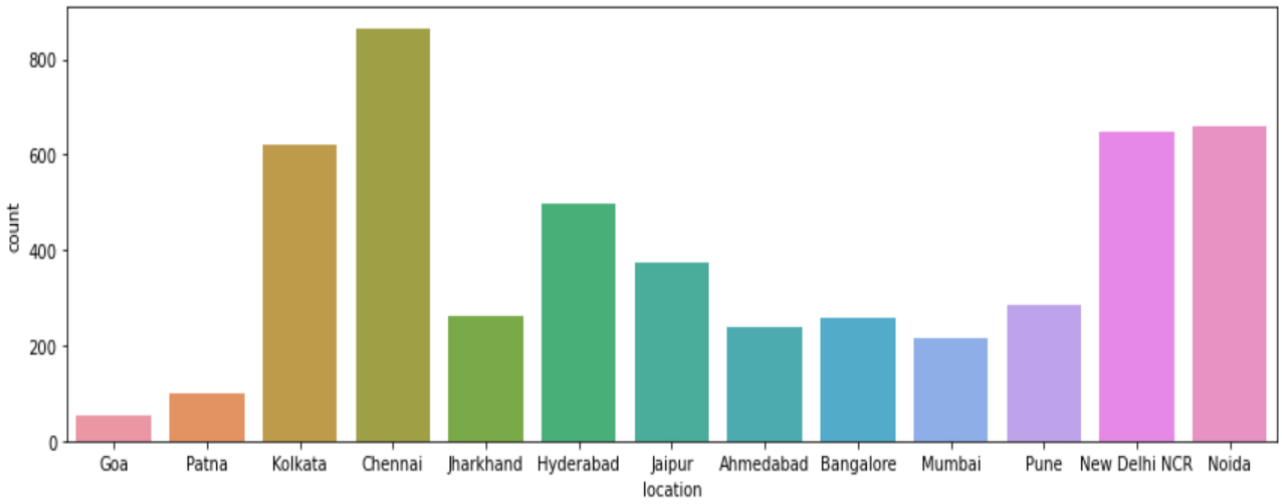
- Above plot shows that used cars have maximum only one owner.



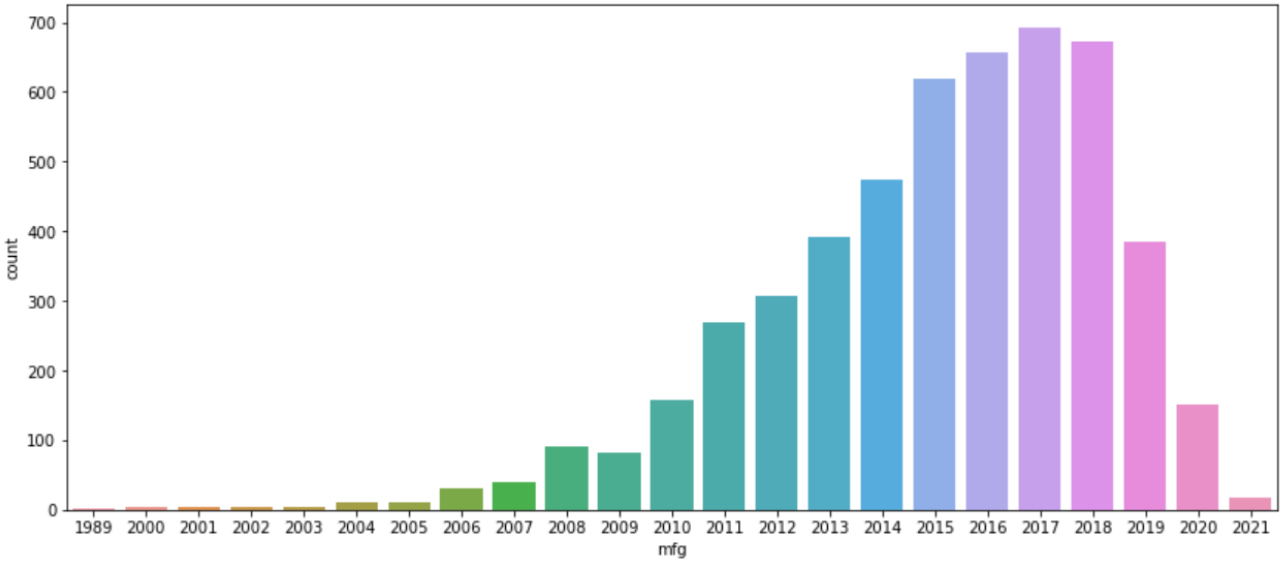
- Around 79% of used cars are of Manual type and rest 21% are of Automatic type.



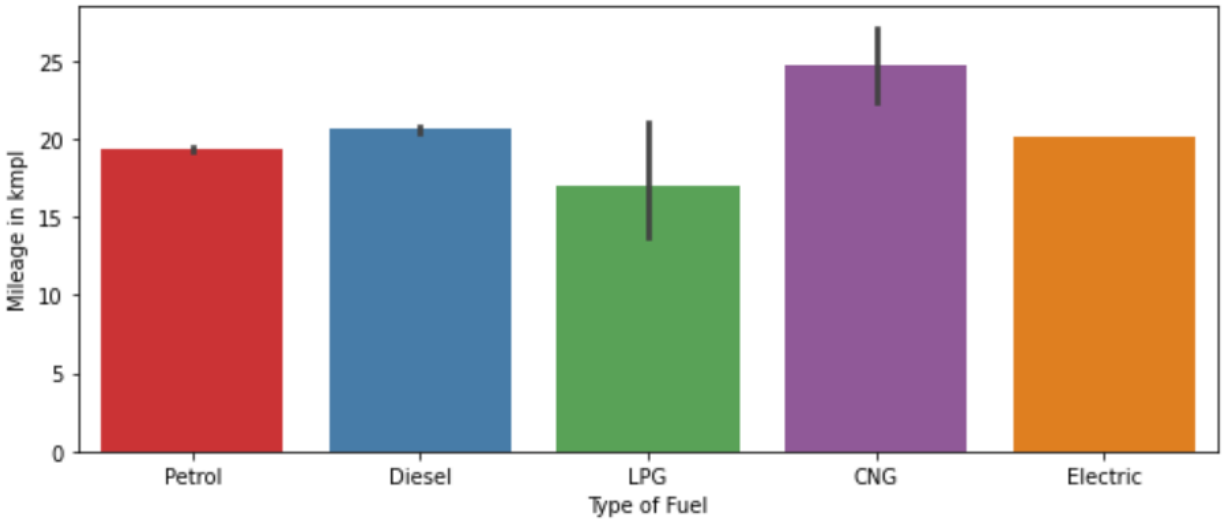
- Maruti brand type of used cars is present in maximum number than any other.



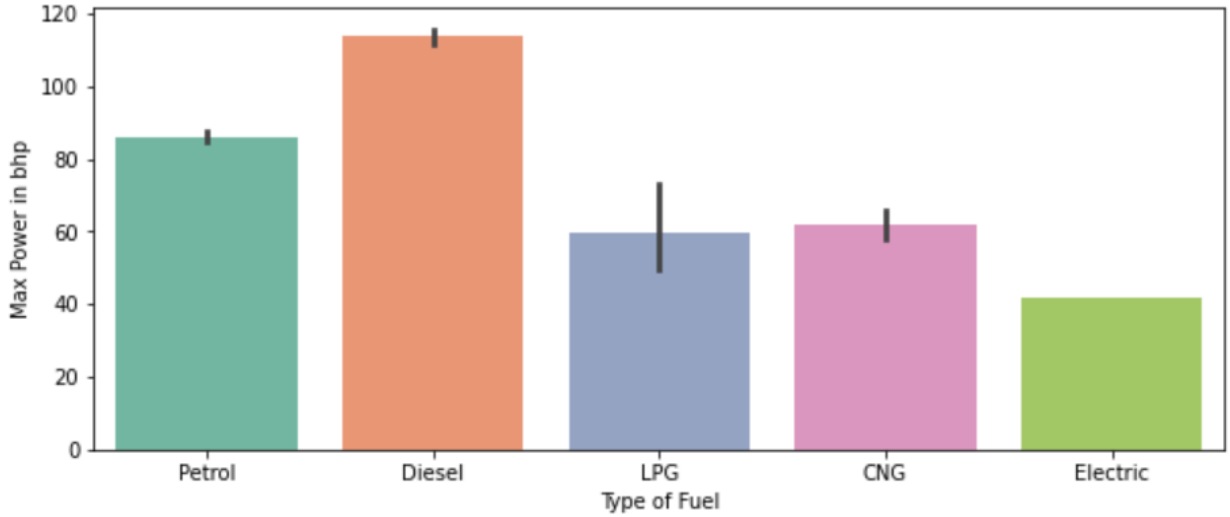
- Maximum used cars data have been scraped from chennai location.



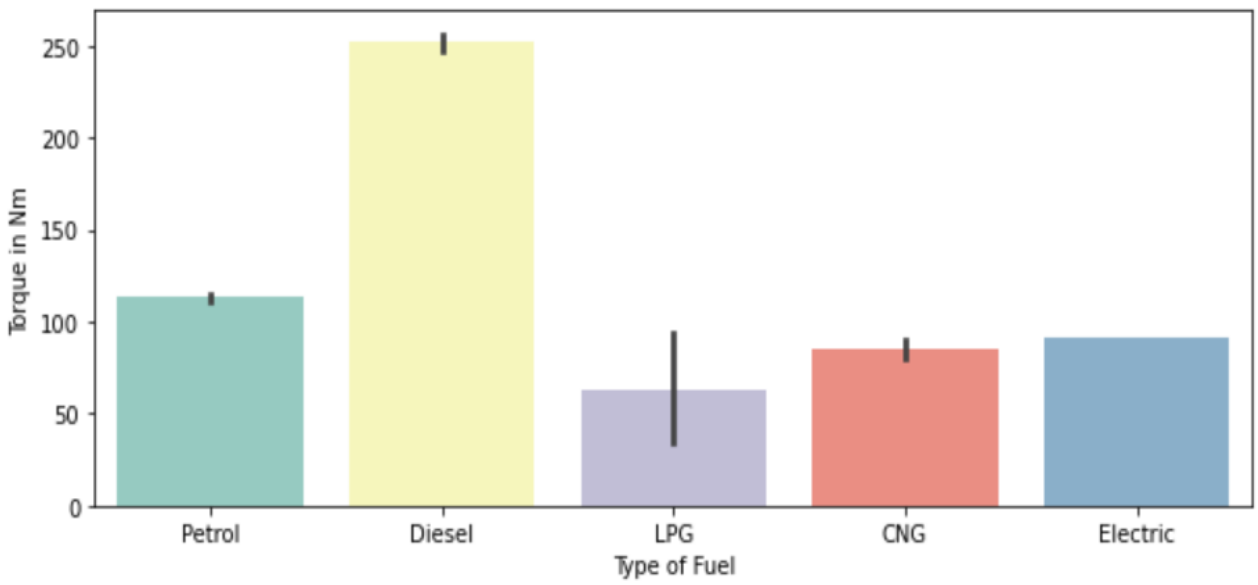
- Mostly the used cars make year lies between 2015-2018.



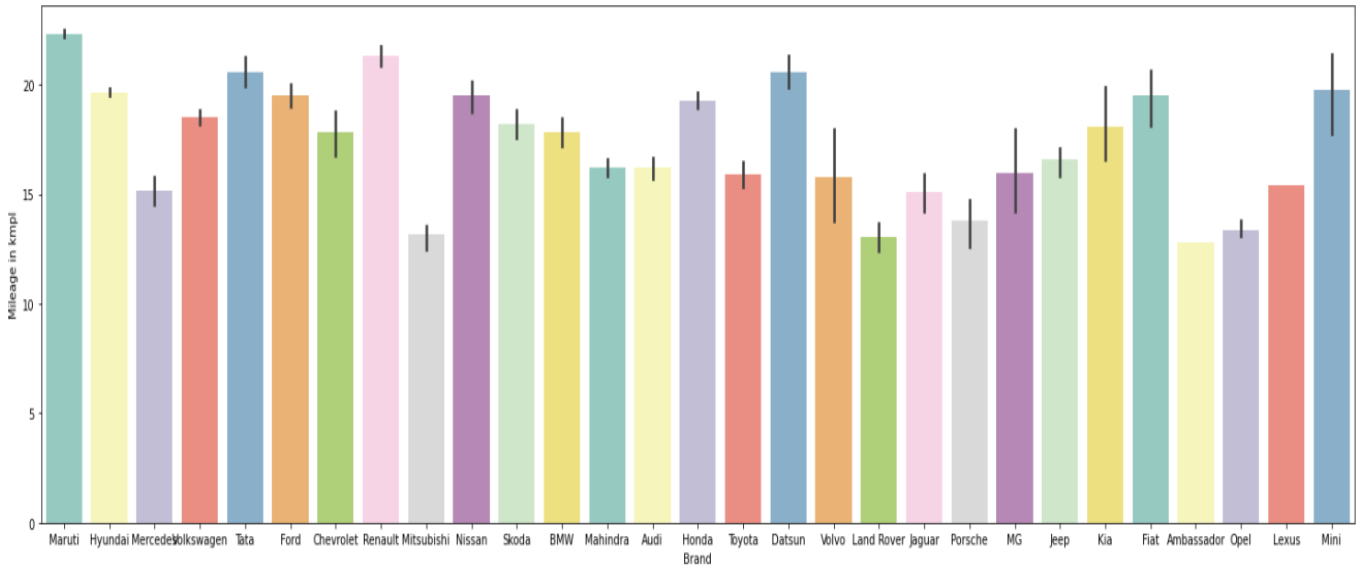
- CNG fuel gives more mileage than other fuels.



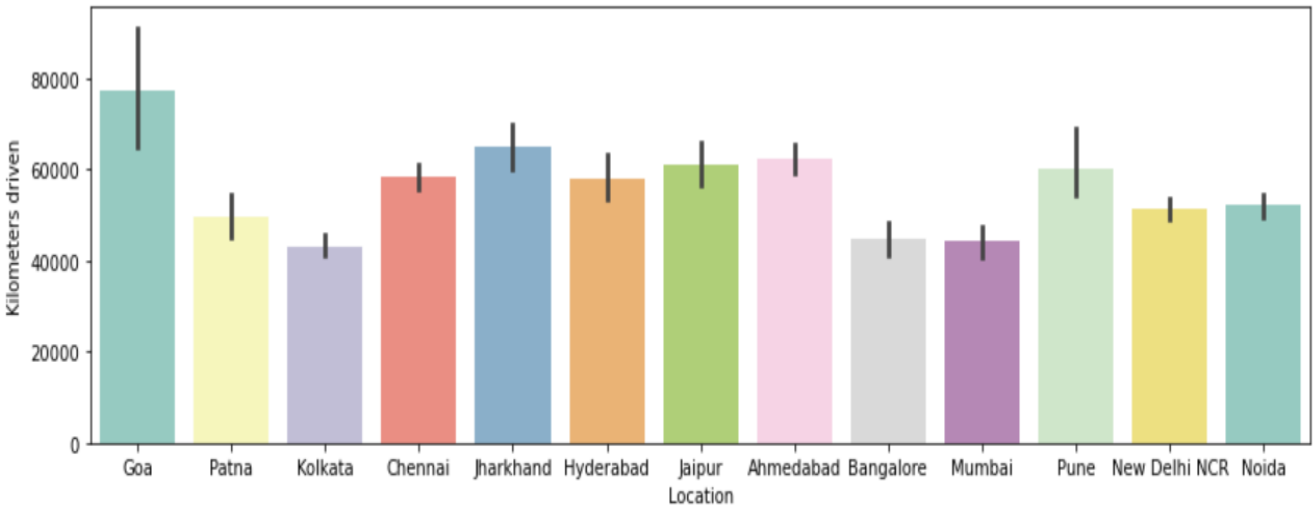
- Diesel fuel gives maximum power to cars than any other fuels.



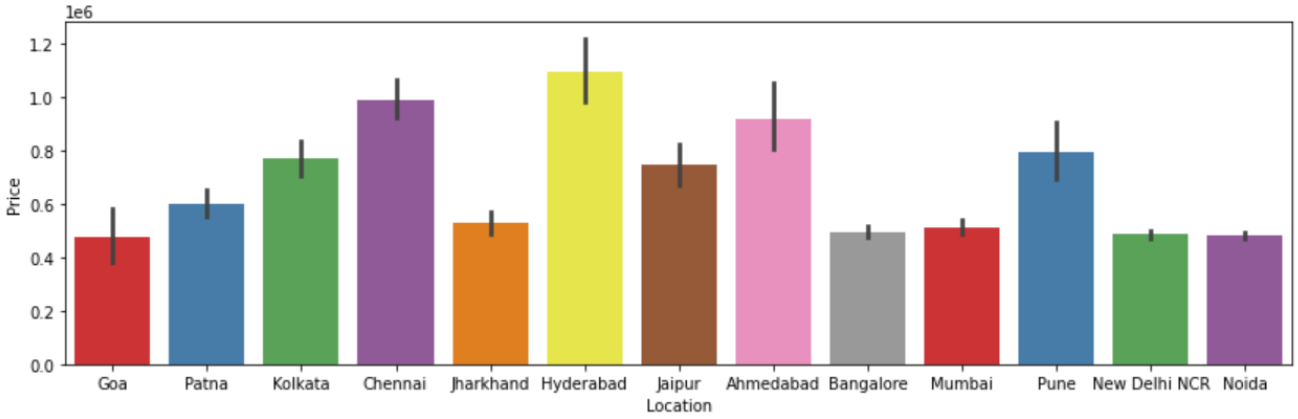
- Diesel fuel provides maximum torque to cars than any other fuels.



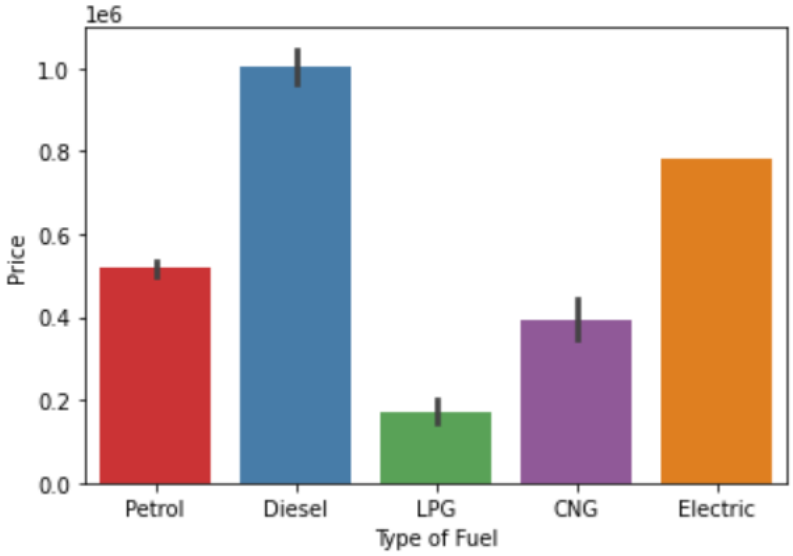
- Maruti gives above 20kmpl mileage.
- Rest all Tata, Renault, Hyundai, Datsun, Ford, Honda, Nissan, Fiat type of used cars gives mileage between (17-20)kmpl.



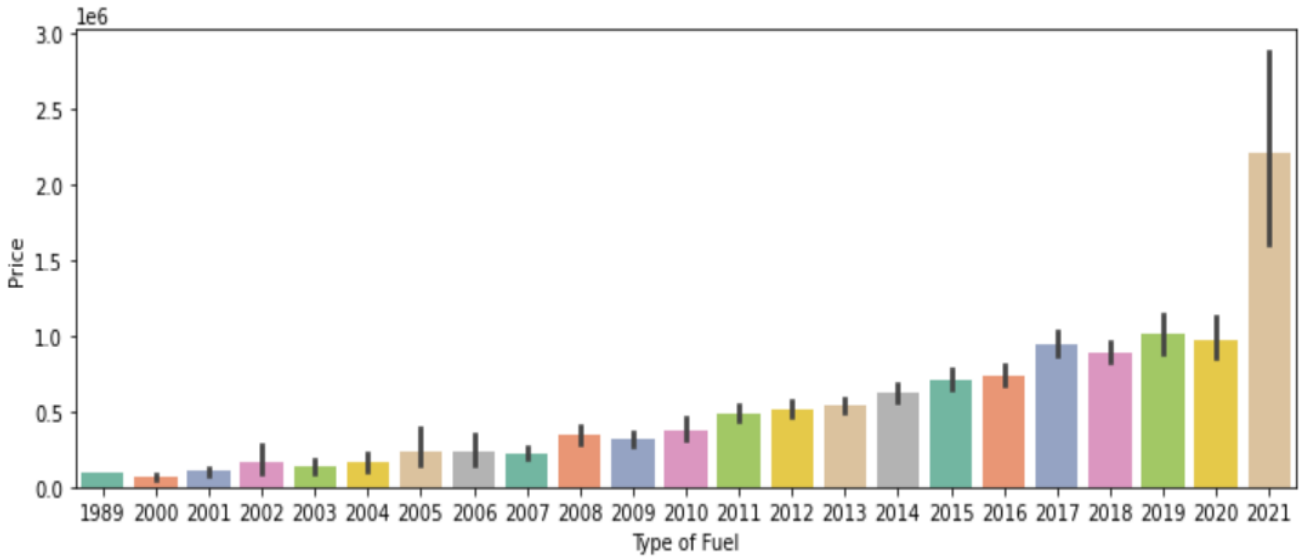
- Maximum kilometers driven used cars is from Goa location.



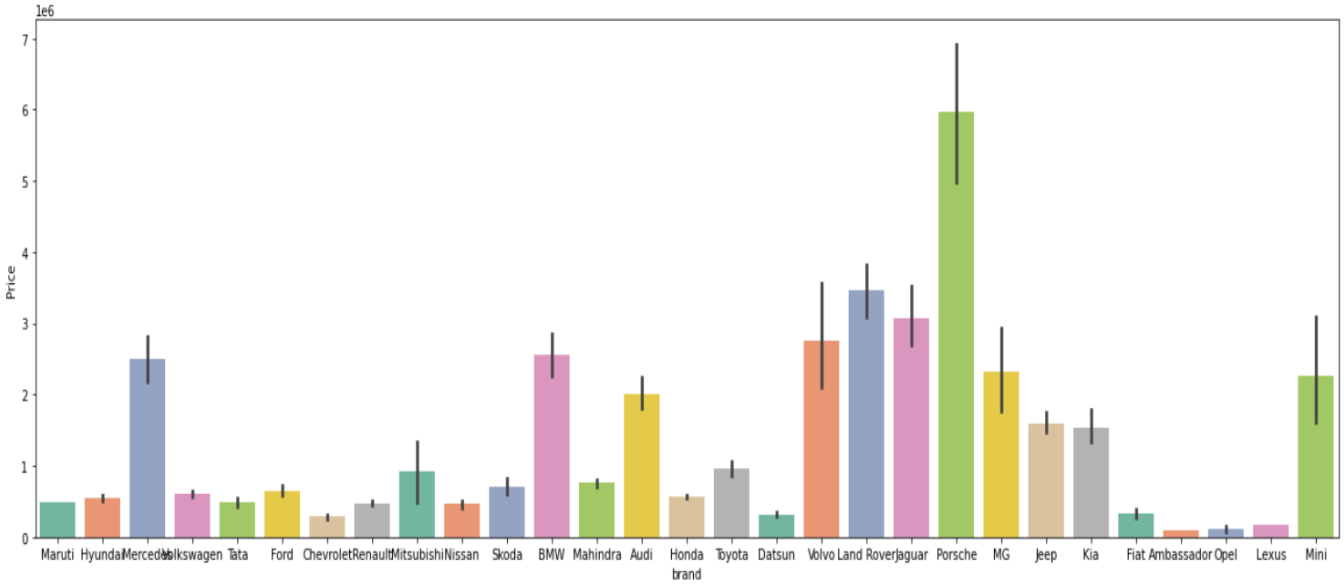
- Price of used cars is maximum in Hyderabad & Chennai and least in Goa, New Delhi, Noida, Bangalore, Mumbai & Jharkhand.



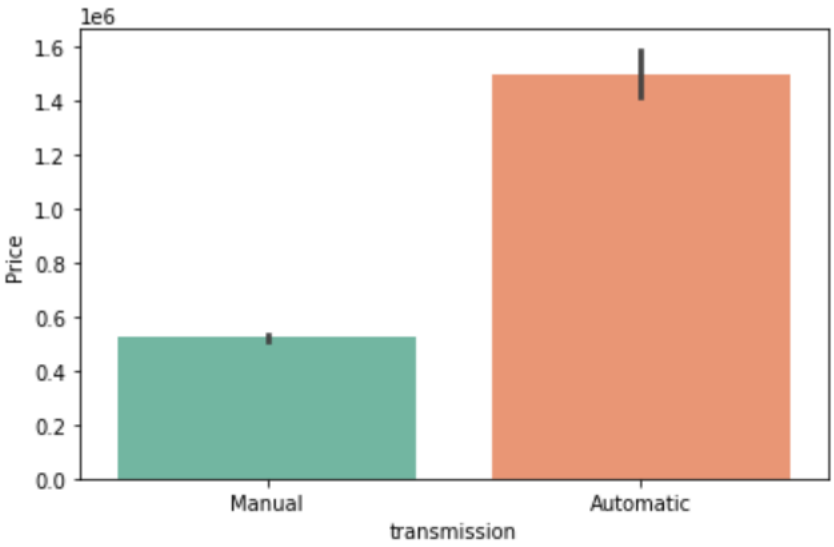
- Diesel used cars price is high than others.



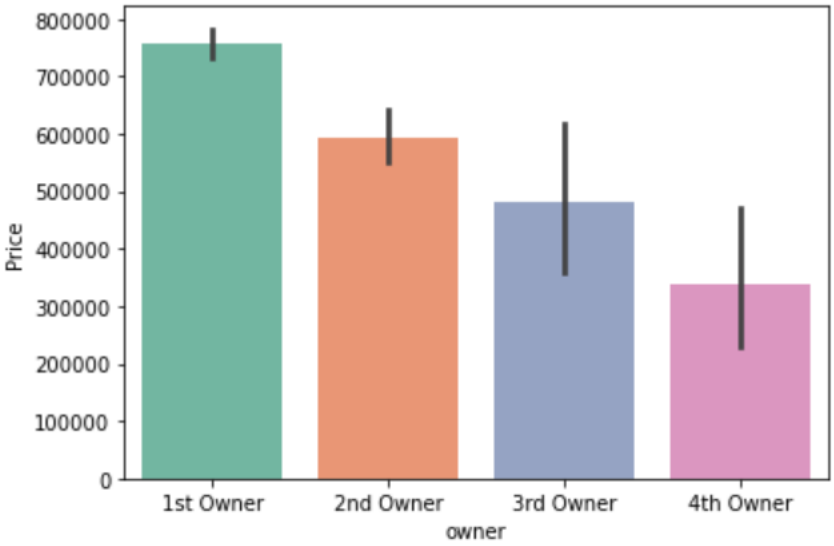
- Older manufacturing date used cars price decreases, whereas used cars with newer manufacturing date price increases.



- Used cars of Porsche brand has maximum price than other brand of cars.



- Automatic type of used cars price is very much high than manual ones.



- More the number of owners of car, lesser will be its price.

INTERPRETATION OF RESULTS

From visualization it was observed that used car prices changes according to the location. Customers will find maximum used cars running on petrol & diesel and few used cars running on electric, CNG & LPG. If a used car has maximum number of owners then its price value decreases. CNG gives better mileage. Make year & kilometers driven is important for checking used car efficiency and its condition.

From preprocessing it was observed that it's a regression problem as we have to predict the price of used cars with the scraped dataset. Where we have to build models on train data. The dataset contained null values which was treated accordingly.

Doing label encoding to convert categorical variables into machine language for the dataset. Through variance inflation factor removed variables to deal with multicollinearity problems.
Splitting the train dataset into two, keeping 25% for testing and rest 75% for training to build ML models.

From modelling it was observed that XGB Regressor was our best model because the difference between its accuracy and CV score was least among all models. Then applied hyper parameter tuning on our best model and the accuracy increased by 3.68%.

CONCLUSION

KEY FINDINGS AND CONCLUSIONS OF THE STUDY

From the whole project evaluation these are the inferences that I could draw from the visualization of data.

- The analysis showed that engine, maximum power and torque variables were important determinants of target (Price) variable for used cars.
- In the car market, mostly used cars available are of petrol & diesel type variant.
- Used cars were having first owner maximum.
- Mostly used cars make year was between 2015-2018.
- In terms of mileage, CNG is better than other fuels.
- Diesel used cars price is high and provides maximum power & torque than any other fuels.
- Automatic transmission type of used cars price is very much high than compared to manual ones.
- Used cars of Hyderabad was very high in price compared to New Delhi, Noida and Goa were price was least.
- Kilometers driven column gives the idea about the used car overall condition and maintenance.
- Porsche brand of used cars have maximum price than any other car brands.

LEARNING OUTCOMES OF THE STUDY IN RESPECT OF DATA SCIENCE

Through data visualization lot of variables depicted a lot of story telling when visualization was done for all of them. Visualization gives meaning to a raw data which helps drawing inference from it.

Challenges faced was that some of variables like owner which was having duplicate values and transmission column which was having few different elements, hence treated them accordingly.

To handle outliers 1.5 IQR method was used and also kept in mind to not lose 7-8% of data.

Lasso and Ridge regression gave same r2 score which means our linear regression model has been well trained over the training data and there is no overfitting

XGB Regressor was the best model to be deployed because the difference between its accuracy and CV value was least among all models.

LIMITATIONS OF THIS WORK AND SCOPE FOR FUTURE WORK

The dataset that I scraped from different websites was less. As lot of time was consumed in data scraping precisely.

The dataset contained few but important features/variables that were required for prediction of used cars price.

If I could add some features/variables it might be tyre type, steering type, color, rear brake type etc which would be important for futher price prediction analysis of used cars.