Fliprobo

# Used Car Price Prediction Model

**Report**


Submitted by:

**Arjun Verma,**

**Intern Data Scientist**

# ACKNOWLEDGEMENT

*I would like to express my greatest appreciation to the all individuals who have helped and supported me throughout the project. I am thankful to Fliprobo team for their ongoing support during the project, from initial advice, and encouragement, which led to the final report of this project.*

*A special acknowledgement goes to my institute Datatrained who helped me in completing the project and learning concepts.*

*I wish to thank my parents as well for their undivided support and interest who inspired me and encouraged me to go my own way, without whom I would be unable to complete my project.*

Below following are the other references:

www.towardsdatascience.com

www.medium.com

www.stackoverflow.com

Datatrained lectures

# 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

## ➢ Conceptual Background of the Domain Problem

Companies such as olx, cardekho, car24, etc which sells and buy used car from seller and vice versa. But before purchasing a used car they used to predict the current market value of the car. Similarly in the given task we have to build a model that can predict used car price from the extracted dataset from the present valuation of the car.

## ➢ Review of Literature

Data has been collected from various websites such as olx, cardekho, cars24.We collected most of the important dataset that can impact the price of the car. Model is created using the data by splitting the data as dependent and independent variable. These dataset are further split into test and train. The train data is trained through various regression algorithms. The algorithm having the lease difference between r2 score and cross val score will be used for hyperparameter tuning. The best parameters are used to tune the model. This model is given to the client in further using to visualise data for future car price prediction.

## ➢ Motivation for the Problem Undertaken

**Genuinely it's a need of the any seller to complete their goal with higher revenue and low expenditure. Hence this model can brings higher revenue because we can predict upcoming selling car prices and bid a price to the seller with lower amount, before their publishment of car prices.**

## ➢ Mathematical/ Analytical Modeling of the Problem

**Data is statistically analysed through variance inflation factor. Analysed through correlation and multicollinearity. Graphical modelling done through seaborn and matplotlib to understanding how different features impact dataset.**

Statistical models used

- ➢ Linear Regression
- ➢ DecisionTreeClassifier
- ➢ Random forest regressor
- ➢ Gradient Boosting Regressor
- ➢ Ada Boost Regressor
- ➢ PCA
- ➢ Standard Scalar

## ➢ Data Sources and their formats

**Datasets are extracted by various site like** olx, cardekho, car24, etc **for building machine learning model to predict selling price of cars based on given parameter.**

**Dataset is having 6019 rows and 8 columns including target.**

**The information about features are as follows**

```
'Name', 'Location', 'Mfg_Year', 'Kilometers_Driven', 'Fuel_Type',
'Transmission', 'Owners', 'Price'
```

1. Name: Name and brand model of the cars.
2. Location : A particular location or position where car can sold.
3. Mfg_Year : The production market classifies manufacturing years to specific vehicles.
4. Kilometers_Driven : The car is driven for particular distance
5. Fuel_Type : Types of fuel is used in car.
6. Transmission : The mechanism by which power is transmitted from an engine to the axle in a motor vehicle.
7. Owners : The car owners numbers specifies used owners of cars.
8. Price : Price of the car that seller want to sell.

```
df.head() # checking first 5 rows
```

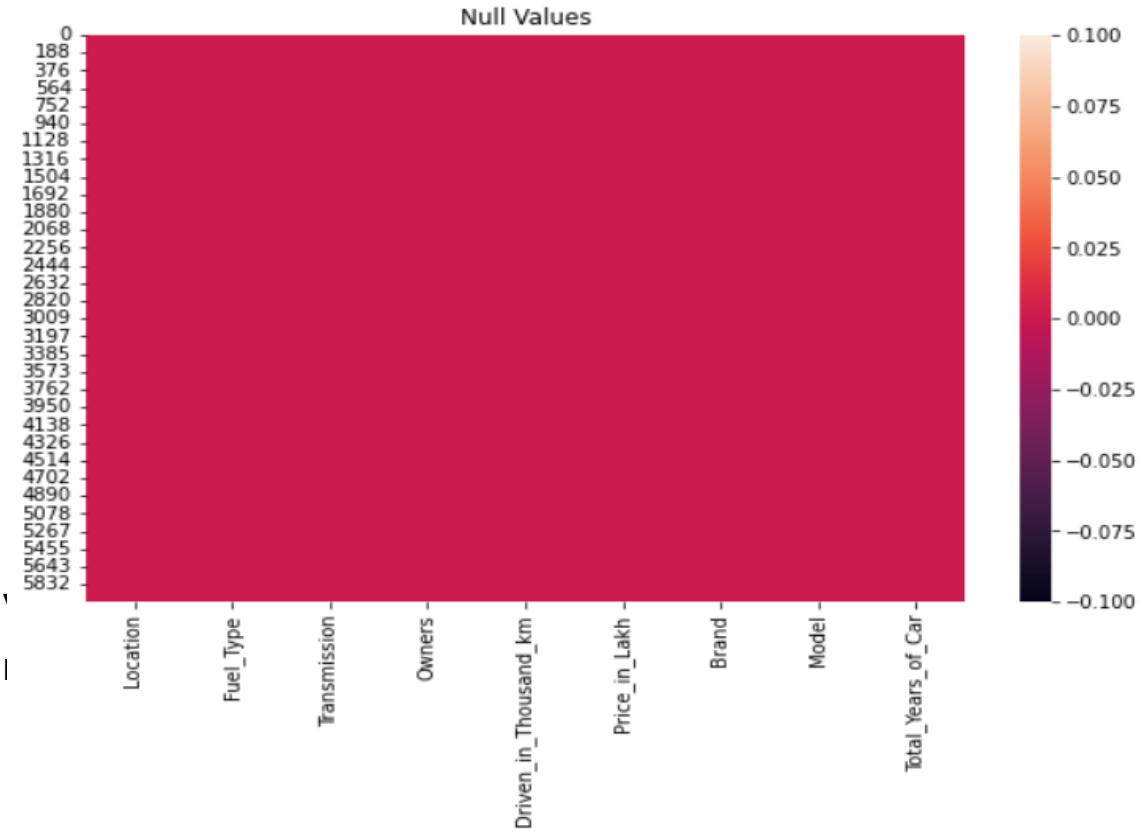
| | Name | Location | Mfg_Year | Kilometers_Driven | Fuel_Type | Transmission | Owners | Price |
|---|---|---|---|---|---|---|---|---|
| 0 | Maruti Wagon R LXI CNG | Mumbai | 2010 | 72000 | CNG | Manual | First | 175000 |
| 1 | Hyundai Creta 1.6 CRDi SX Option | Pune | 2015 | 41000 | Diesel | Manual | First | 1250000 |
| 2 | Honda Jazz V | Chennai | 2011 | 46000 | Petrol | Manual | First | 450000 |
| 3 | Maruti Ertiga VDI | Chennai | 2012 | 87000 | Diesel | Manual | First | 600000 |
| 4 | Audi A4 New 2.0 TDI Multitronic | Coimbatore | 2013 | 40670 | Diesel | Automatic | Second | 1774000 |

# Dataset Information

**'Name', 'Location', 'Fuel_Type', 'Transmission', 'Owners' are of object type data columns.**

**'Mfg_Year', 'Kilometers_Driven', 'Price' are of numerical type data columns.**

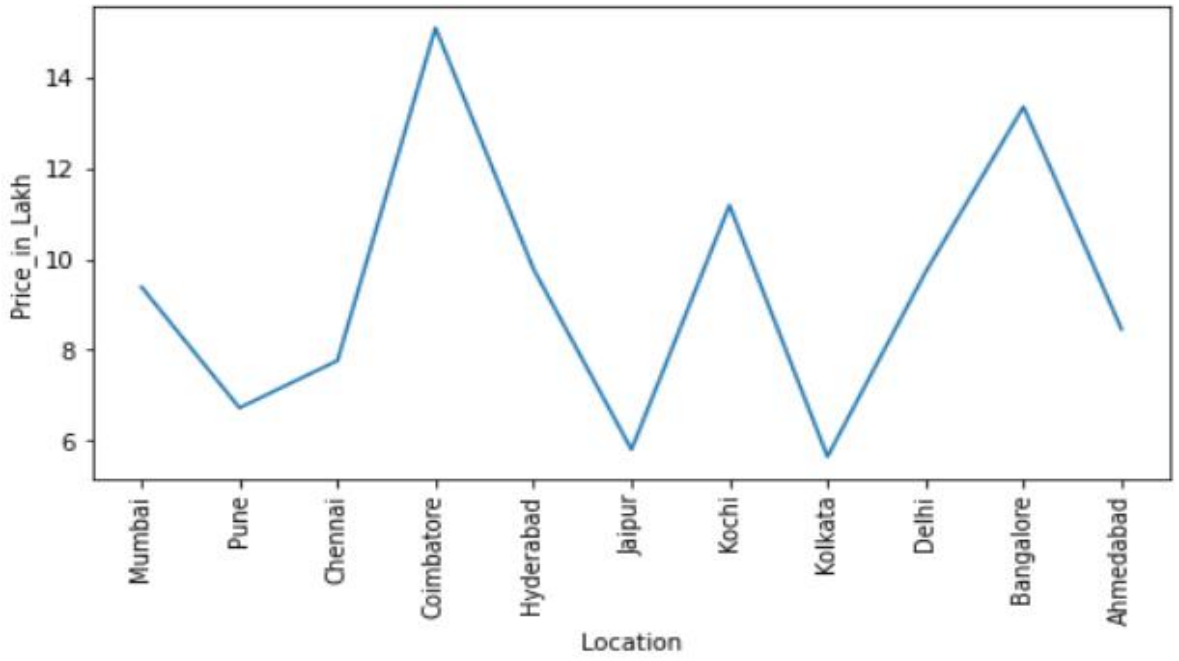
### Checking Null Values of the dataset




**There are no null values present in the dataset.**

# Visualization of important features for understanding

**Price in Lakh vs Location**




Note:- Here we convert price in Lakh because the value of price get converted into log in visualization and correlation becomes very high.

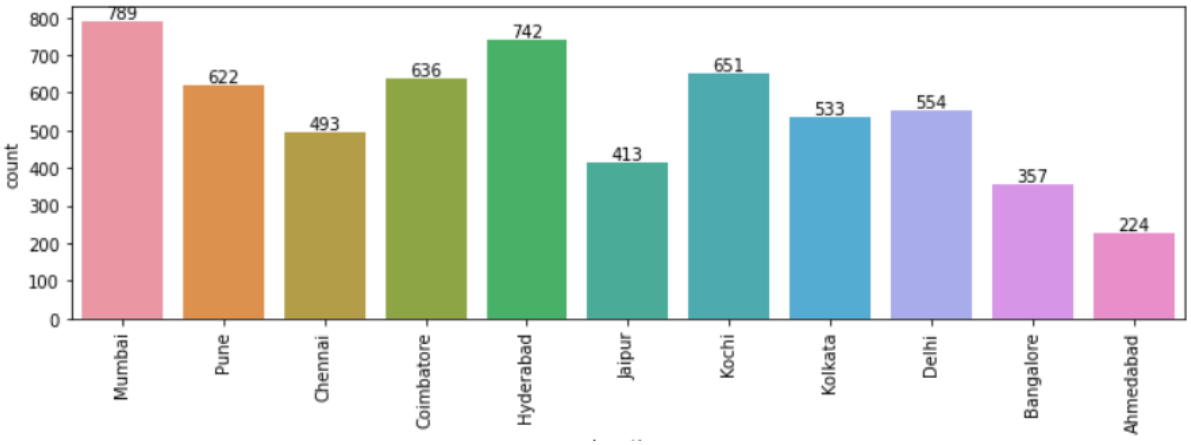
## Observations:

1. Location Coimbatore and Banglore having very high price values cars as compared to other locations.

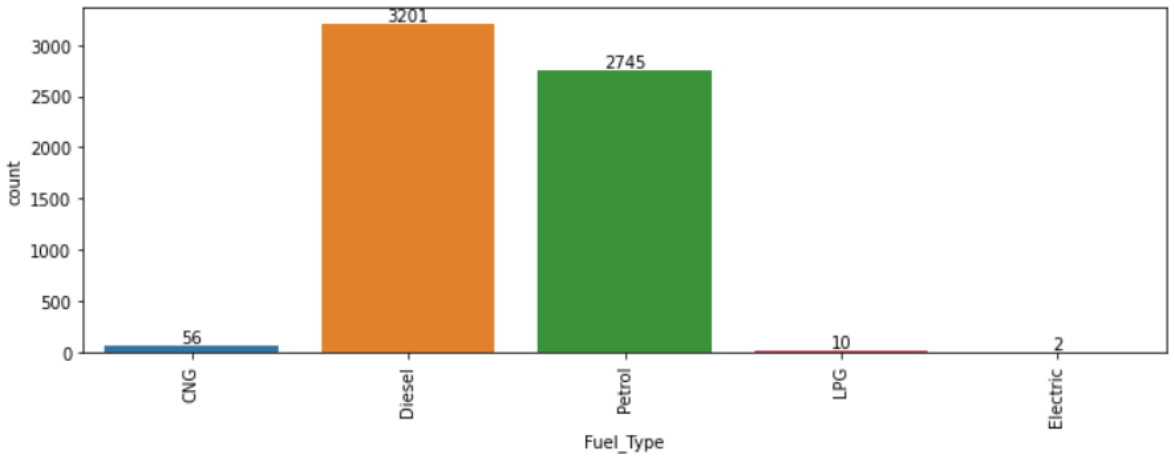
2. Pune, Jaipur and Kolkata having very less price values car as compared to other locations.

# Count Plot

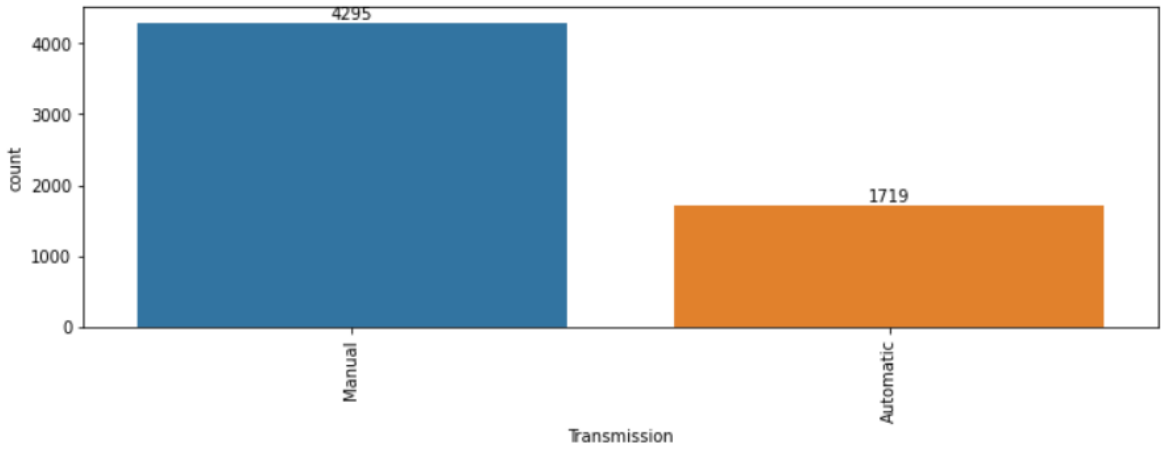
**Location**



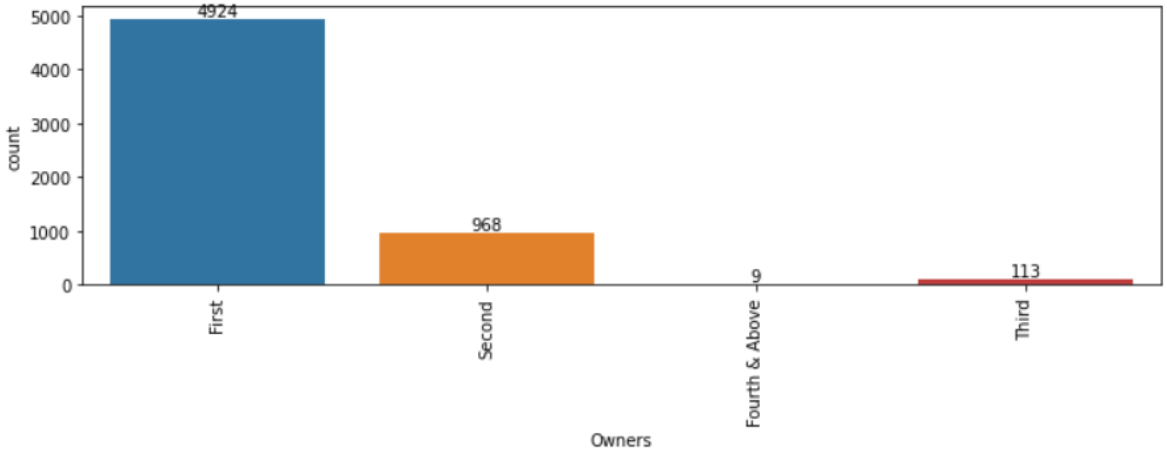

Fuel Type



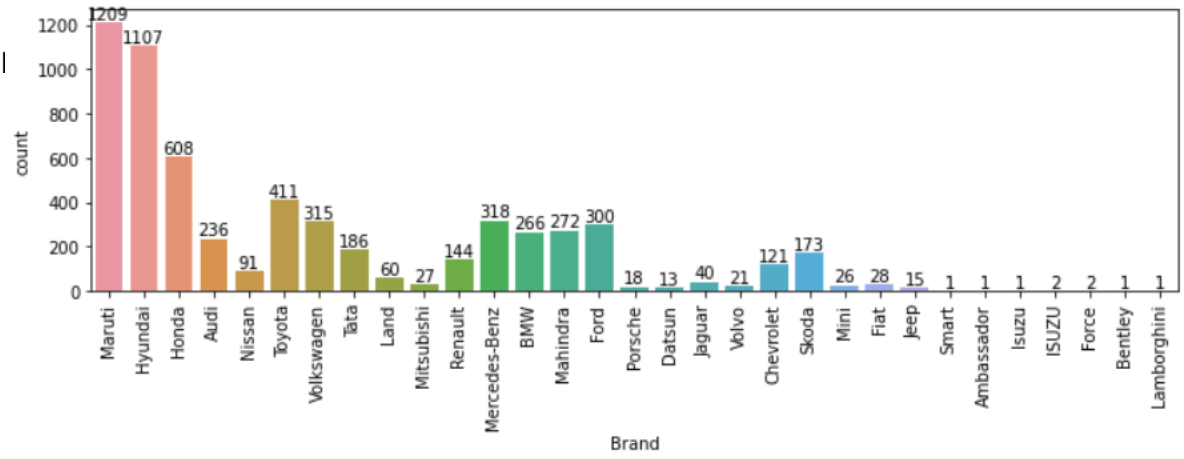
Transmission




Owners



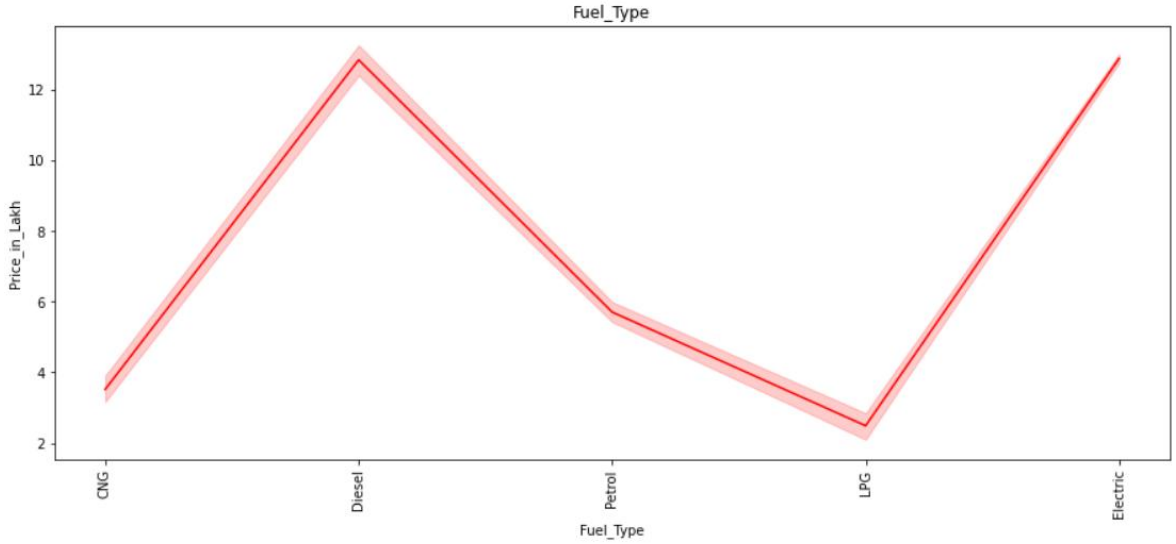
Brand



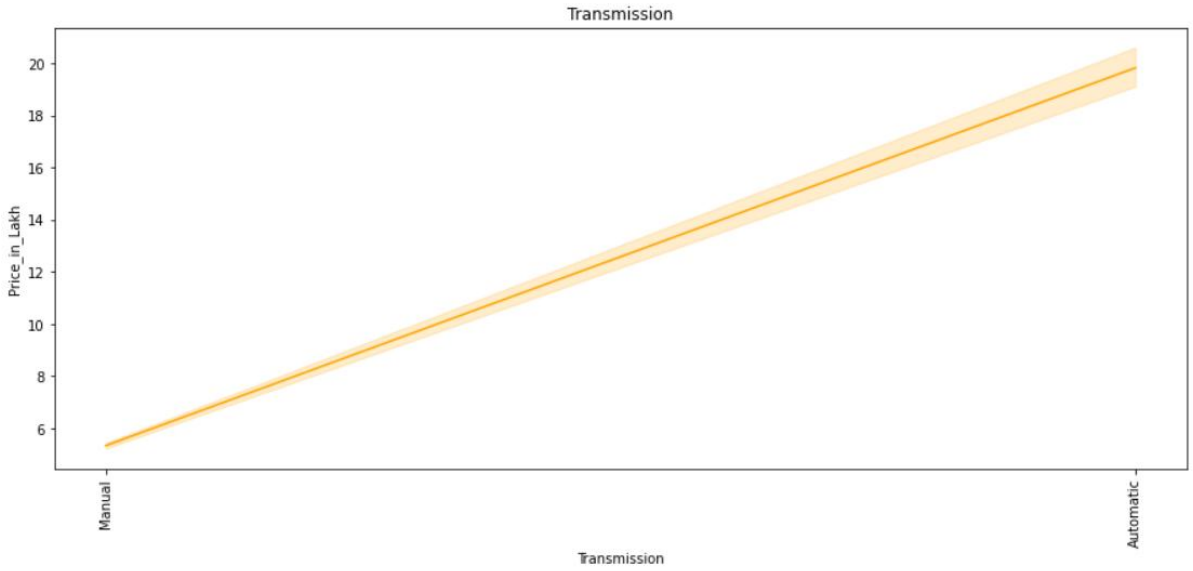

## Observations

- Most of the seller are lies in the location Mumbai, Hyderabad, Coimbatore, Kochi, Pune as compared to other location.
- Diesel and Petrol cars are more in the sell dataset.
- Manual cars are more as compared to automatic cars.
- First owners cars are more as per dataset.
- Maruti and Hyundai brands are more in sell as compared to other brands.

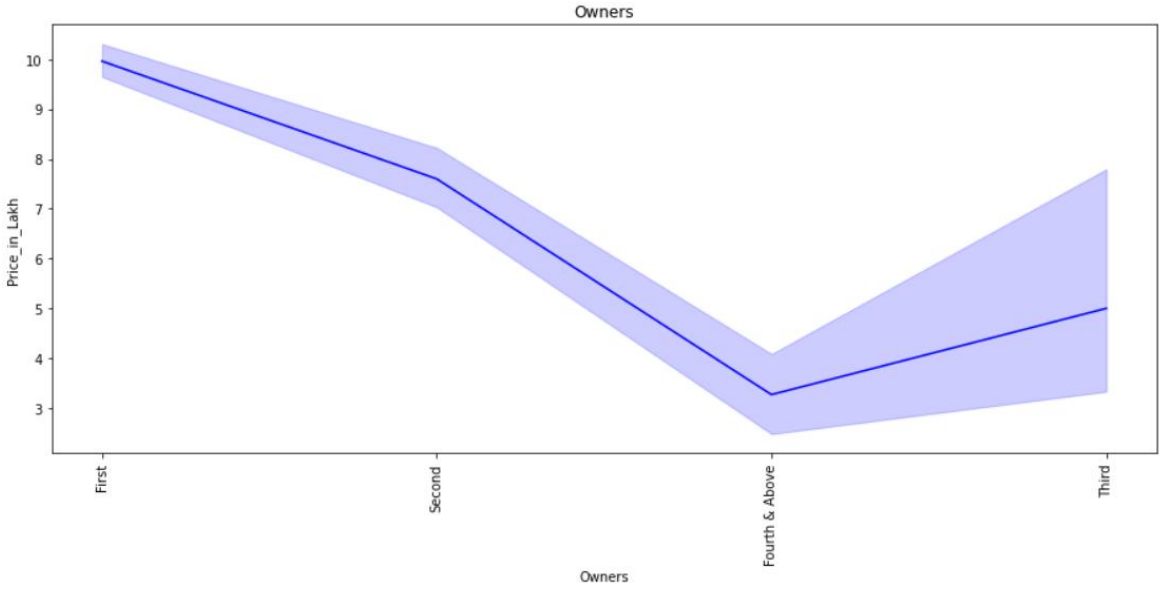
Price vs Fuel



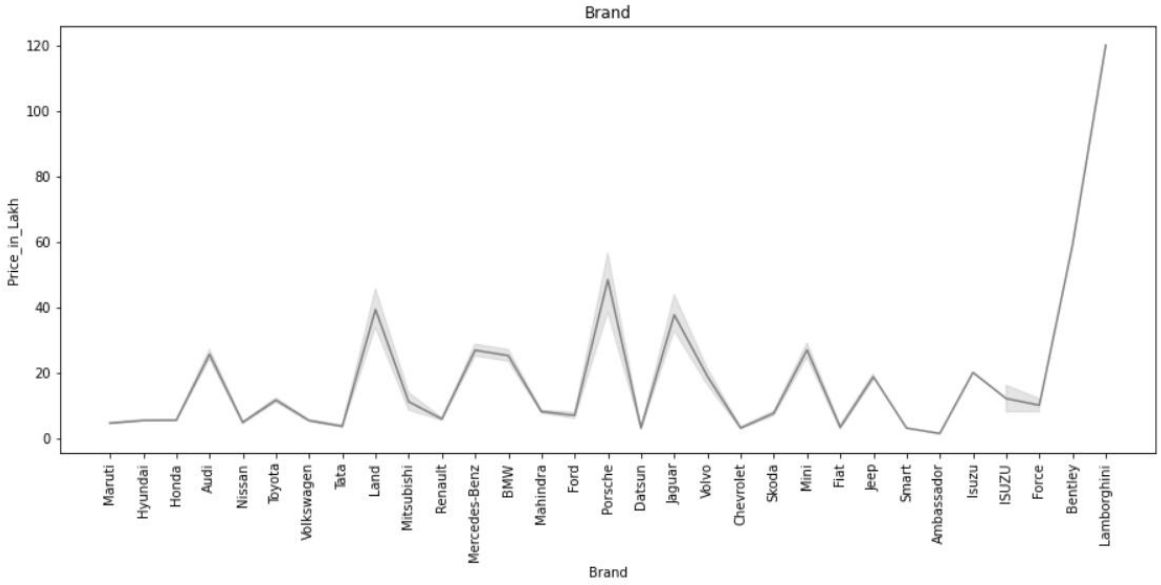

Transmission vs Price



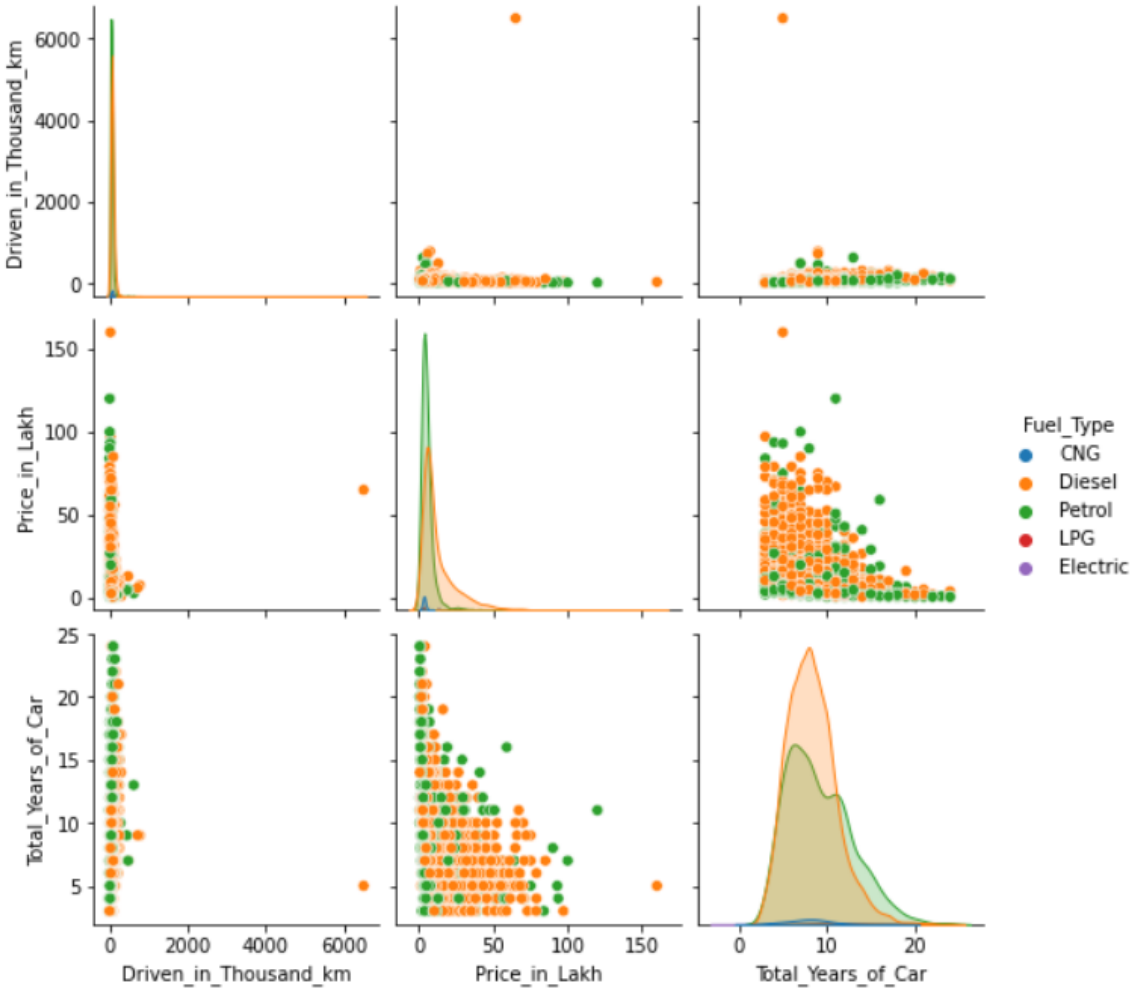
Price vs Owners



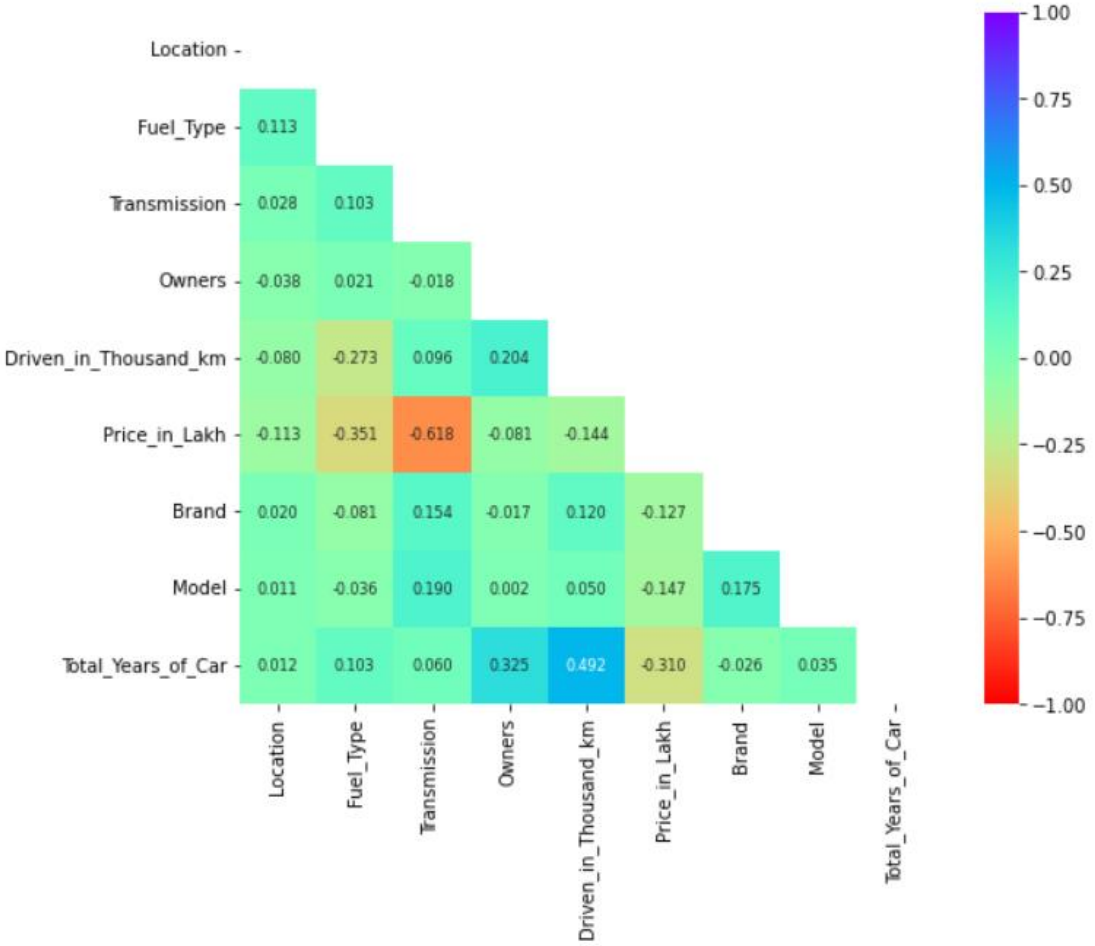

Brand vs Price



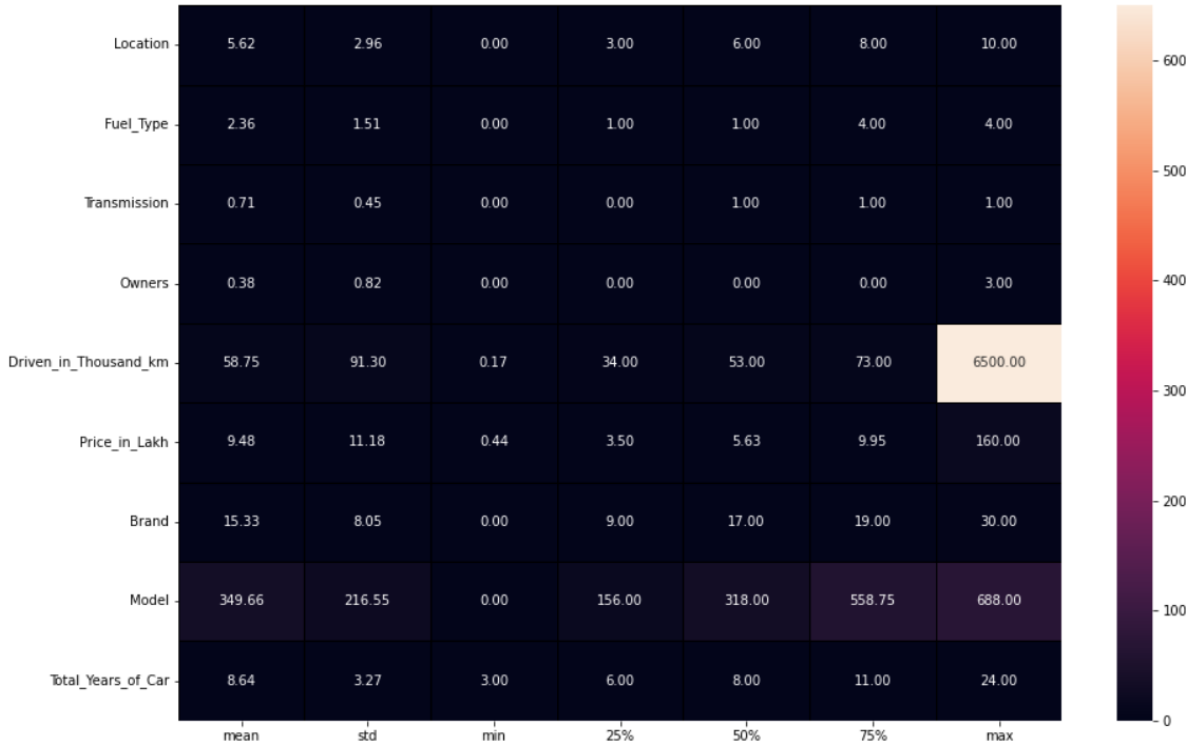
Different numerical columns via pair plot



# Correlation of the Dataset



# Describe of the Dataset




**Converting objects dataset into numerical form we are using Ordinal Encoder**

```
from sklearn.preprocessing import OrdinalEncoder
onc = OrdinalEncoder()
```

```
for i in df.select_dtypes(include = 'object').columns:
    df[i] = onc.fit_transform(df[i].values.reshape(-1,1))
```

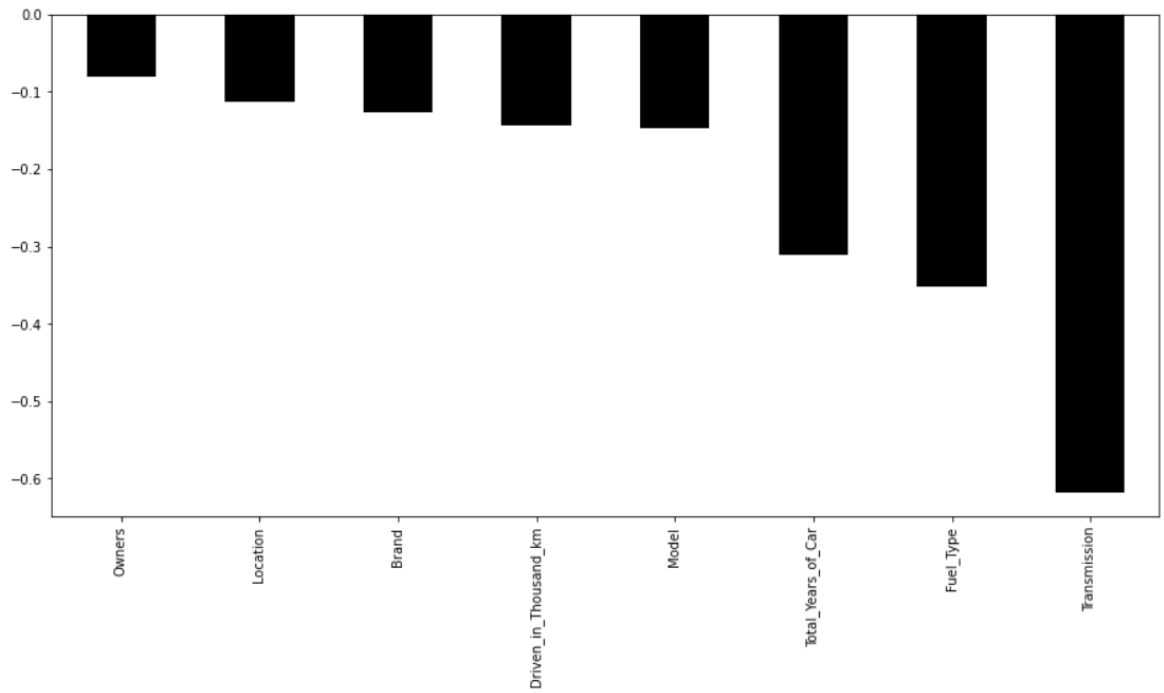
```
df.head(1) # checking result
```

| | Location | Fuel_Type | Transmission | Owners | Driven_in_Thousand_km | Price_in_Lakh | Brand | Model | Total_Years_of_Car |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 9.0 | 0.0 | 1.0 | 0.0 | 72.0 | 1.75 | 18.0 | 616.0 | 12 |

# Outliers

**We have applied Z score and Interquartile method for outlier removal and find that Z score gives lesser data lose hence we consider it.**

# Checking Positive and Negative Correlation



# Dividing data for feature selection

```
#Splitting the indepedent and target variable in x and y
x= df_z.drop('Price_in_Lakh',axis=1)
y= df_z['Price_in_Lakh']
```

## Checking Mutlicollinearity

```
def calc_vif(x):

    vif = pd.DataFrame()
    vif['Variance'] = x.columns
    vif["VIF Factor"] = [variance_inflation_factor(x.values, i) for i in range(x.shape[1])]
    return vif
```

```
calc_vif(x)  # checking VIF of numerical columns
```

| | Variance | VIF Factor |
|---|---|---|
| 0 | Location | 4.306790 |
| 1 | Fuel_Type | 4.272252 |
| 2 | Transmission | 3.859315 |
| 3 | Owners | 1.342835 |
| 4 | Driven_in_Thousand_km | 1.526596 |
| 5 | Brand | 4.282165 |
| 6 | Model | 3.650128 |
| 7 | Total_Years_of_Car | 8.745327 |

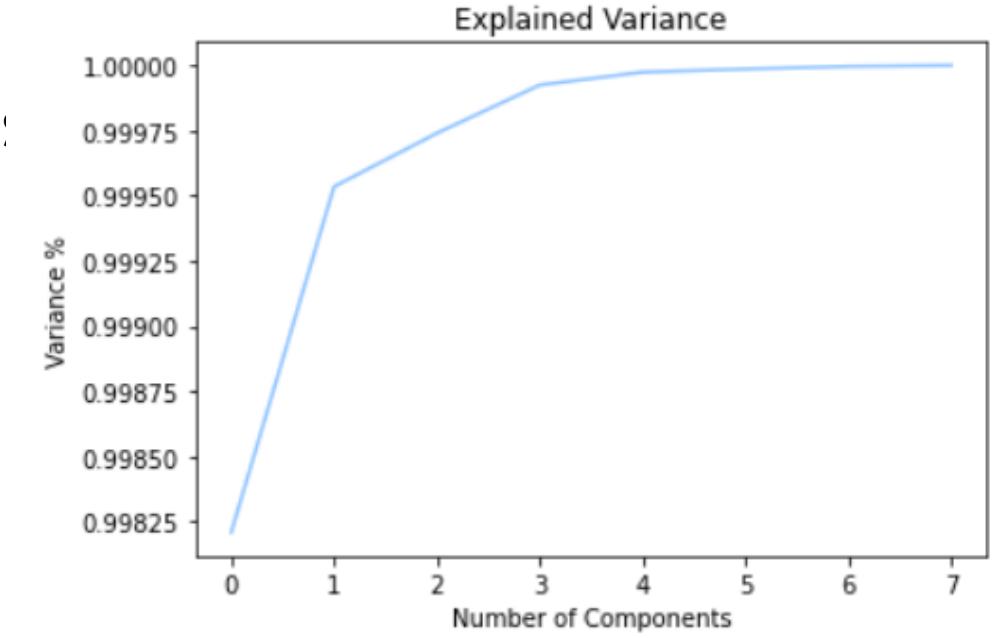
**Total year of car show some multicollinearity but due to limited features in dataset we are not removing it.**

**Removing Skewness by 'yeo- Johnson' method.**

## Principle Component Analysis

```
from sklearn.decomposition import PCA
```

```
pca = PCA()
principleComponents = pca.fit_transform(x)
plt.figure()
plt.plot(np.cumsum(pca.explained_variance_ratio_))
plt.xlabel('Number of Components')
plt.ylabel('Variance %')
plt.title('Explained Variance')
plt.show()
```




**4 components explain around 95% variance in data**

## Selecting Kbest Features

```
from sklearn.feature_selection import SelectKBest, f_classif
```

```
bestfeat = SelectKBest(score_func = f_classif, k = 'all')
fit = bestfeat.fit(x,y)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(x.columns)
```

```
fit = bestfeat.fit(x,y)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(x.columns)
dfcolumns.head()
featureScores = pd.concat([dfcolumns,dfscores],axis = 1)
featureScores.columns = ['Feature', 'Score']
print(featureScores.nlargest(10,'Score'))
```

```
                 Feature     Score
2           Transmission  4.681590
7     Total_Years_of_Car  3.768128
1              Fuel_Type  1.890408
5                  Brand  1.317362
6                  Model  1.246838
4  Driven_in_Thousand_km  1.158665
3                 Owners  0.980556
0               Location  0.841045
```

**Since all the dataset show some scores hence we are not dropping anyone of them**

# Model Building and Results

| | Model | r2score | Cross_val_score | RMSE score | Difference between cv score and cross_val score |
|---|---|---|---|---|---|
| 0 | LinearRegression | 58.108984 | 53.744877 | 5.003015 | 4.364107 |
| 1 | Ridge Regressor | 58.087022 | 53.745034 | 5.004326 | 4.341988 |
| 2 | Lasso Regressor | 54.884926 | 50.704005 | 5.191970 | 4.180921 |
| 3 | DecisionTreeRegressor | 38.925076 | 45.254096 | 6.040916 | -6.329021 |
| 4 | RandomForestRegressor | 52.891159 | 45.589804 | 5.305454 | 7.301355 |
| 5 | KNeighborsRegressor | 87.482964 | 86.476775 | 2.734778 | 1.006189 |
| 6 | GradientBoostingRegressor | 70.658796 | 64.203586 | 4.187071 | 6.455210 |
| 7 | AdaBoostRegressor | 50.377123 | 31.542691 | 5.445181 | 18.834432 |
| 8 | ExtraTreesRegressor | 87.012598 | 83.854995 | 2.785688 | 3.157603 |
| 9 | XGBRegressor | 91.460950 | 91.107464 | 2.258791 | 0.353486 |
| 10 | LGBMRegressor | 90.141832 | 89.121988 | 2.426998 | 1.019844 |

Best models

**- KNeighborsRegressor : Model shows low r2 score in training and testing accuracy hence we cannot consider it.**

**- DecisionTreeRegressor: Same as DecisionTreeRegressor shows very much difference in training and testing accuracy hence we cannot consider it. Model becames underfit.**

**- XGBRegressor: Same as above two model it shows similar in CV score and testing r2 score hence we can consider it.**

**- GradientBoostingRegressor: GradientBoostingRegressor shows closer R2 testing and crpss val score but still training score it much greater than it testings R2 score which makes model underfit also R2 score not good yet from all models hence we can't consider it.**

**- LGBMRegressor : Moder shows very close differenct R2 score of testing obtaining also CV score is also not good hence we can consider it for model building.**

# Final Model XGB Regressor

## Hyperparameter tuning of XGB Regressor

```
param = {'booster' : ['gbtree','dart','gblinear'],
              'importance_type' : ['gain','split'],
              'n_estimators' : [100,200,500],
              'eta' : [0.001, 0.01, 0.1]
             }
GSCV = GridSearchCV(XGBRegressor(), param, cv=5)
GSCV.fit(x_train,y_train)
```

```
At random state 58 model giving best accuracy score

R2 Score:-  91.12973255504855


---------------------------------------
Mean squared error:-  4.925504986058744
Mean absolute error:-  1.1391381090438795
Root Mean squared error:- 2.2193478740519126

Text(0.5, 1.0, 'LGBMRegressor()')
```

# Model Deployment

## Deploy Model

```
import pickle

filename = "Used_Car_Price.pkl"
pickle.dump(final_model, open(filename, 'wb'))
```

## Loading Model

```
load = pickle.load(open('Used_Car_Price.pkl', 'rb'))
result = load.score(x_test, y_test)
print(result)
```

```
0.9182635688728165
```

```
conclusion = pd.DataFrame()
conclusion['Predicted Car price'] = np.array(final_model.predict(x_test))
conclusion['Actual Car price'] = np.array(y_test)
```

```
conclusion.sample(10)
```

| | Predicted Car price | Actual Car price |
|---|---|---|
| **572** | 5.000850 | 5.53 |
| **320** | 10.908600 | 12.00 |

## ➢ Hardware and Software Requirements and Tools Used

**Operating System: Window 11**

**RAM: 8 GB**

**Processor: i5 10th Generation**

**Software: Jupyter Notebook**

**Python Libraries: Mainly**

**Pandas: This library used for dataframe operations .**

**Numpy: This library gives statistical computation for smooth functioning .**

**Matplotlib: Used for visualization.**

**Seaborn: This library is also used for visualization.**

**Sklearn: This library having so many machine learning module and we can import them from this library.**

**Pickle: This is used for deploying the model.**

**Xgboost: Extreme Gradient Boosting, is a scalable, distributed gradient-boosted decision tree (GBDT) machine learning library**

**Lightgbm: Light version of Gradient Boosting Machine.**

# CONCLUSION

## ➢ Key Findings and Conclusions of the Study

**This project has built a model that can predict upcoming Sale Prices of Cars. For this company can reduces loses in Investment. The challenge behind Sale Price finding in machine learning is the number of features in dataset. Also some other issues like imputation understandings and so many values are zeros.**

## ➢ Learning Outcomes of the Study in respect of Data Science

**Data cleaning is the most important part in this model building as we see above there are so many NULL values we fill with imputation and ranges some of column dataset for better observations. This project has gives so much information about parameters that how a single parameter can increase or decrease prices of house.**

## ➢ Limitations of this work and Scope for Future Work

**Model work with similar parameters as we build the whole model if some of the parameters missed then we need to train model with remains parameter after that we can predict upcoming sales of houses hence we need to up to date all the parameters as per training dataset.**