FLIP ROBO

Used Car Selling Price Prediction

Submitted by

Raganee Verma

(Intern fliprobo batch 32)

ACKNOWLEDGMENT

I have referred to various articles in Stack overflow while cleaning the data and referred algorithms in sklearn while model building. At the time of web scraping, I phase lots of difficulties then I took help from selenium official website and some of the pictures for the projects are obtained from google.

INTRODUCTION

Due to the numerous elements that influence a used vehicle's market pricing, determining if the advertised price is accurate, is a difficult undertaking. The goal of this project is to create machine learning models that can properly forecast the price of a used car based on its attributes so that buyers can make educated decisions.

- Business Problem Framing

Due to Covid 19 Pandemic 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. The people who want to sell his used cars is facing problems with their previous car price valuation machine learning models. So, they are looking for new machine learning models from new data. I have made a car price valuation model using current dataset of used car selling company.

- Conceptual Background of the Domain Problem

The goal of this Project is to develop a model that can anticipate fair used car pricing based on a variety of factors such as vehicle model, year of manufacture, fuel type, Price, Kms Driven. In the used car market, this strategy can benefit vendors, purchasers, and car manufacturers. It can then produce a reasonably accurate price estimate based on the data that users provide. Machine learning and data science are used in the model-building process.

- Review of Literature

There are various companies such as olx, cardekho, car24,etc which performs Car price prediction for used cars. These predictions are done through various data models. This data model allows the person who is in the buyer and the seller side to understand the current market value of the used car.

- Motivation for the Problem Undertaken

If we talk about few years ago, we did not need a car as much as we do now. At the time of Covid-19, people were using only personal cars because all public transport had stopped. After Covid-19 the middle-class population also started feeling the need of a car earlier they used to travel by public transport. Many middle-class people are unable to buy a new car, so they want to buy used cars within their budget.

Car has become a significant part of most of the households, specially where the public transport is not advanced. Hence Used car plays the pivotal role among cars as it expands the market of cars to a wider population.

Analytical Problem Framing

- Mathematical/ Analytical Modelling of the Problem

The solution is divided into the following sections:

1. Data understanding and exploration
2. Data cleaning
3. Date Visualization
4. Data preparation
5. Model building and evaluation

Data understanding and exploration

I have scraped used cars data from CarDekho.com

The Dataset contains 5980 rows and 8 columns.

	Driven_kilometers	Fuel_type	Transmission	Sell_Price	Year	Brand	model_variant	Location
0	86,294	Diesel	Manual	4.11 Lakh	2014	Maruti	Swift Dzire VDI	Delhi
1	82,496	Petrol	Manual	8.72 Lakh	2019	Maruti	Ciaz Alpha BSIV	Delhi
2	1,04,537	Diesel	Manual	6.23 Lakh	2015	Mahindra	XUV500 W6 2WD	Delhi
3	73,656	Diesel	Automatic	6.51 Lakh	2016	Renault	Duster 110PS Diesel RxL AMT	Delhi
4	38,280	Diesel	Manual	7.90 Lakh	2015	Hyundai	Creta 1.6 CRDi SX	Delhi

1. Model_variant : A car's make is the brand of the vehicle, while the model refers to the name of a car product and sometimes a range of products.
2. Brand(Company name) : A type of product manufactured by a particular company under a particular name.
3. Year(Manufacture year): The Production market classifies years to specific vehicles, and model codes in place of the model year.
4. Diven_Kilometer : The car is driven for particular distance.
5. Fuel_Type : Types of fuel is used for particular car.
6. Transmission : the mechanism by which power is transmitted from an engine to the axle in a motor vehicle.
7. Sell_price : The car is quoted for particular price to sell it.
8. Location : A particular place or position where car is sold.

The column"**Price**" is the target variable and rest of the columns are independent variables. The independent variables are again divided into Categorical and Numerical variables.

Data cleaning and feature engineering

1.We've seen that there are no missing values in the dataset by heatmap visualization.

2. There are 44 duplicate value is present but no need to remove it because may be same car model is selling in same price.

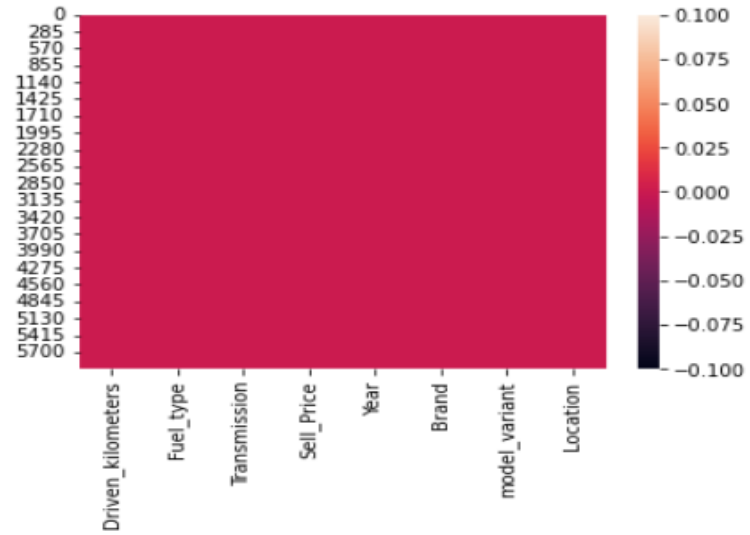
3. Add new column car_age that is calculated by using current year minus manufacturing year of car.

4. In ous dataset the sell_price data haveing different different unit like lakh,Cr,and thousand. So we convert it into one unit thousand and then convert it into object to intiger type data.

5. Eliminate comma symbol from kilometer_driven and convert it into integer type .

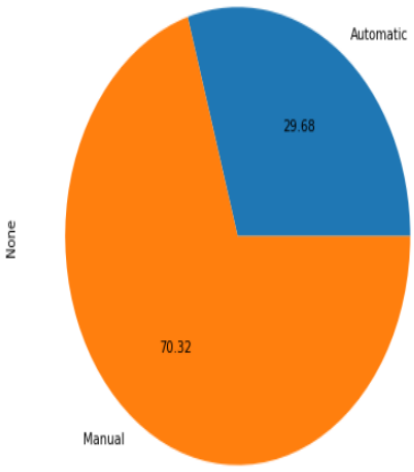
```
1 sns.heatmap(df.isnull())##check missing value through visualization
```

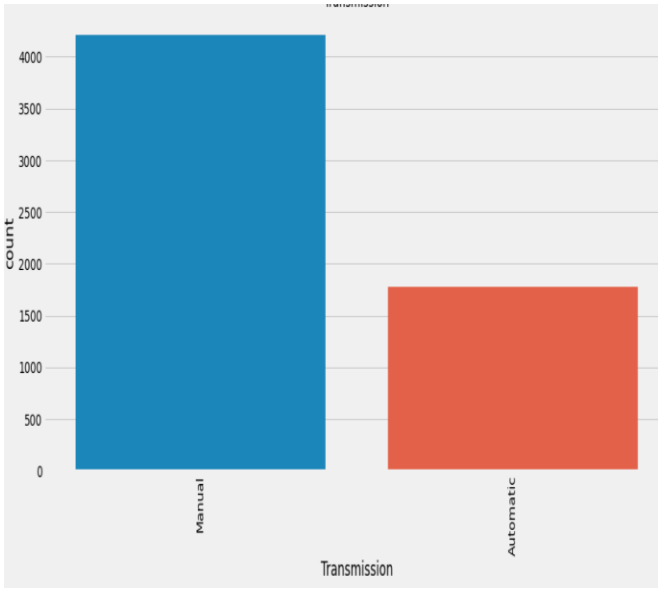
```
<AxesSubplot:>
```



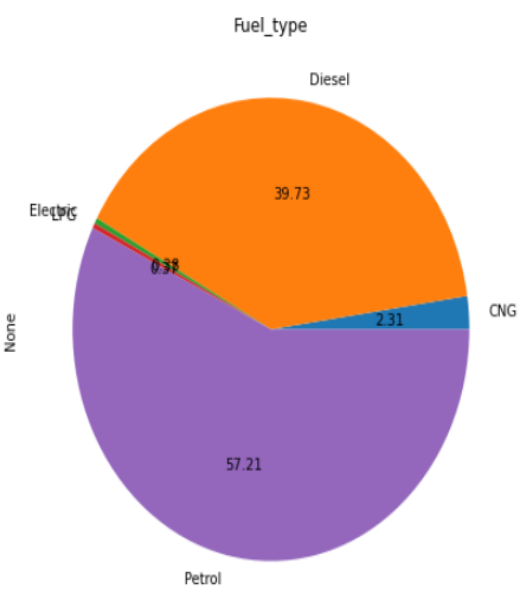
Date Visualization

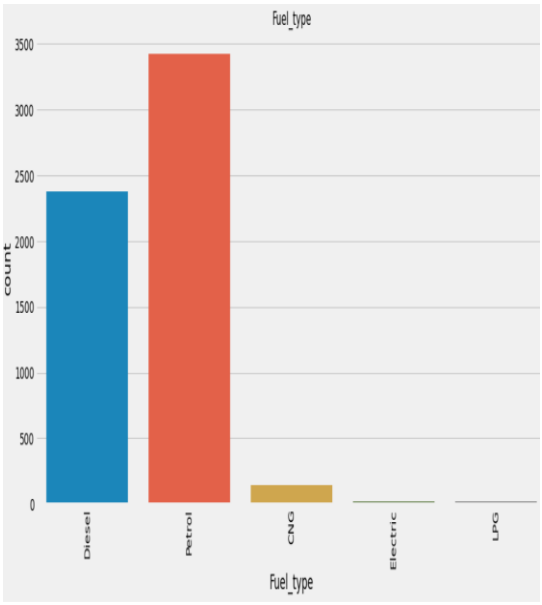
We are Analysing the data by various visualization technique



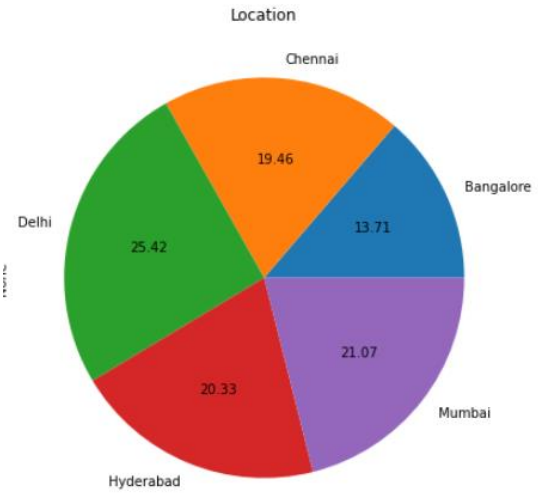


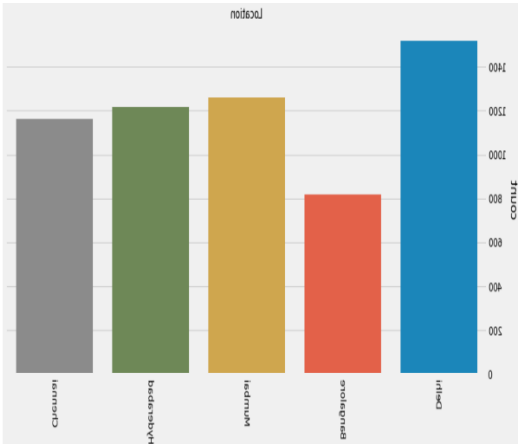
- Most of the used car is manual (70%)





- Higher no. of petrol car (57%) is there and very few cars is LPG(0.35%) fuel type .

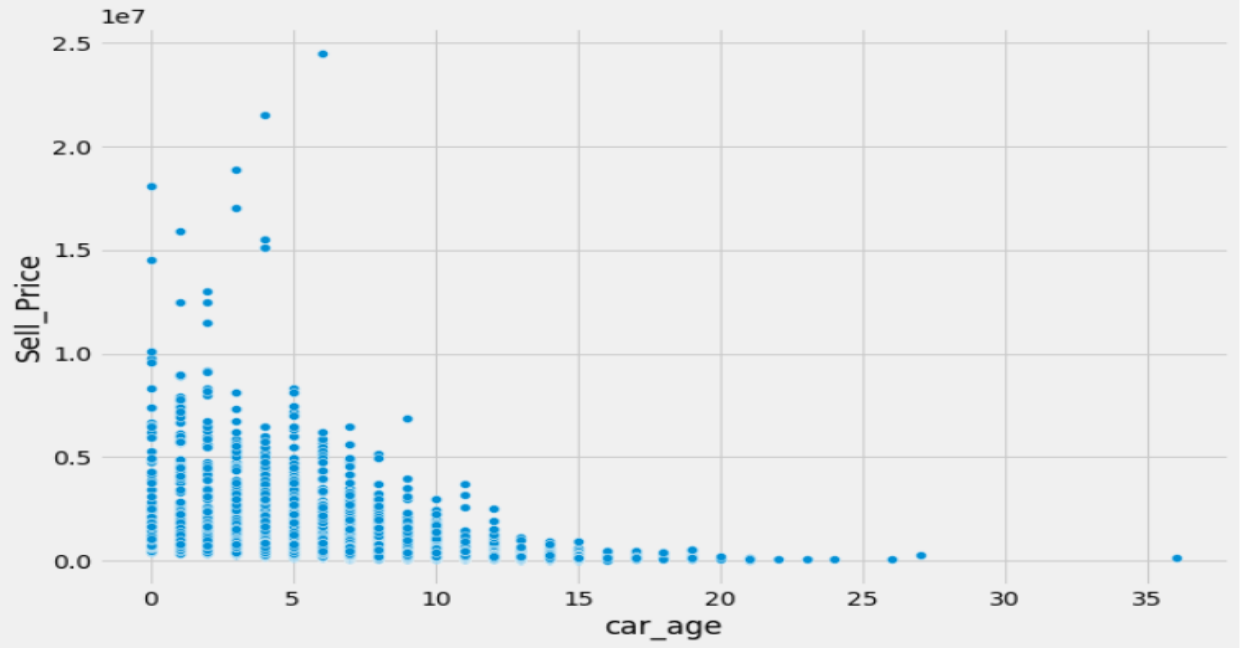




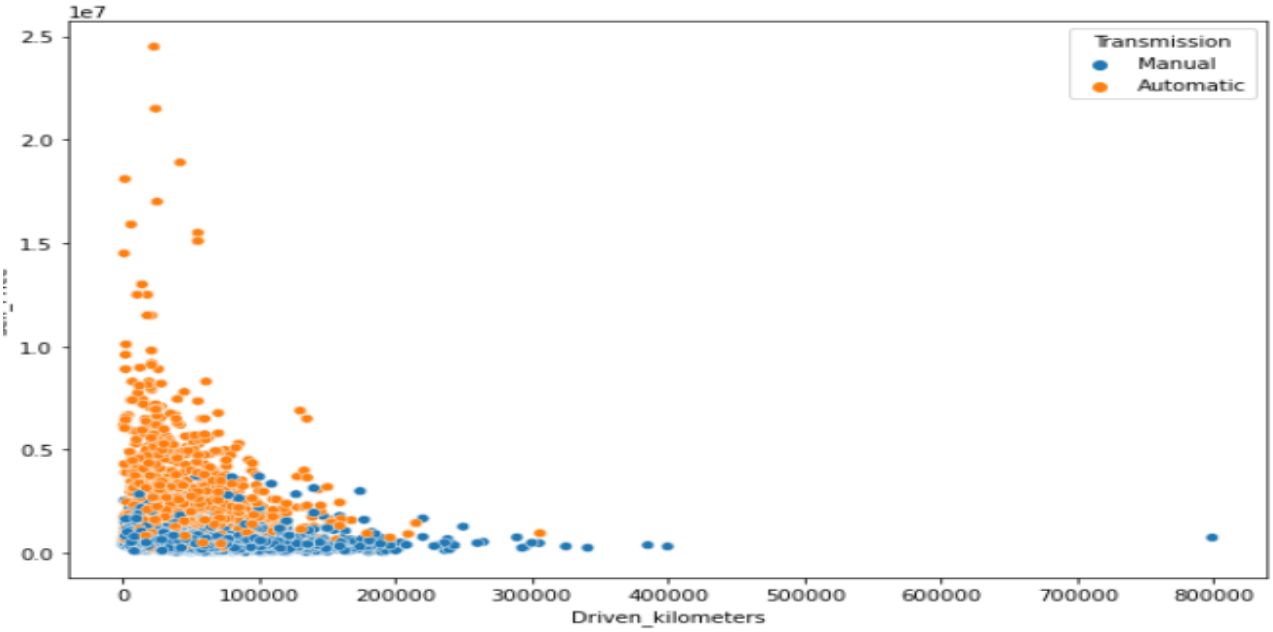
- Higher no. of used car is from Delhi(25%).

Bivariate Visualization

Scatter plot between descriptive variable and target variable

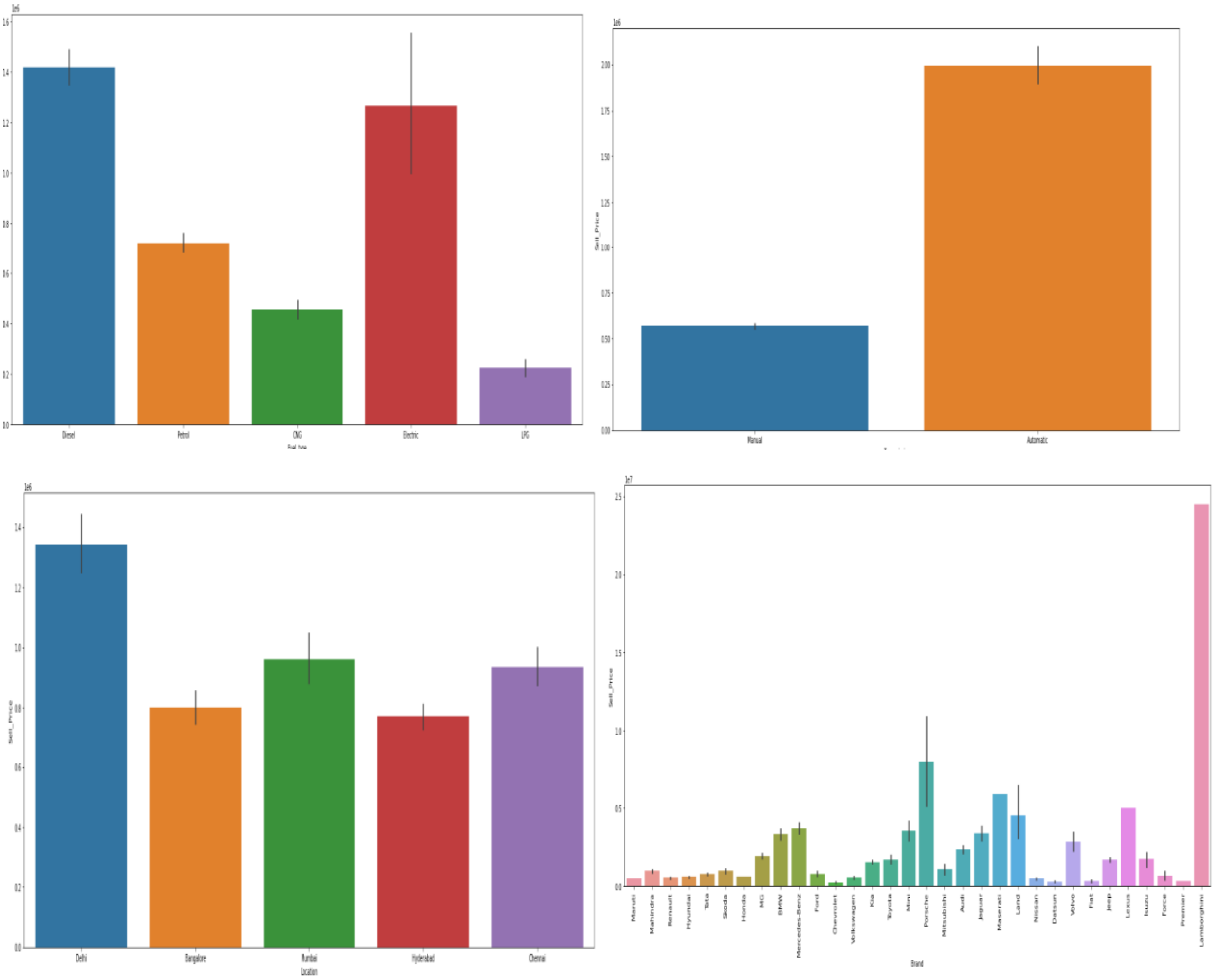


- Used car Price between zero to 7 year is good. The value of used car depreciates as soon as exceeds 10 years.



- If the car runs within 10000 kms then its price is fine but as soon as it goes above the value of car goes down a lot.
- Automatic car always cost more than manual car

Bar plot ategorical variable and target variable



Observation of above bar and bar plot:

1.The price of Diesel and electric car is higher approx. 5 to 25 lakhs.

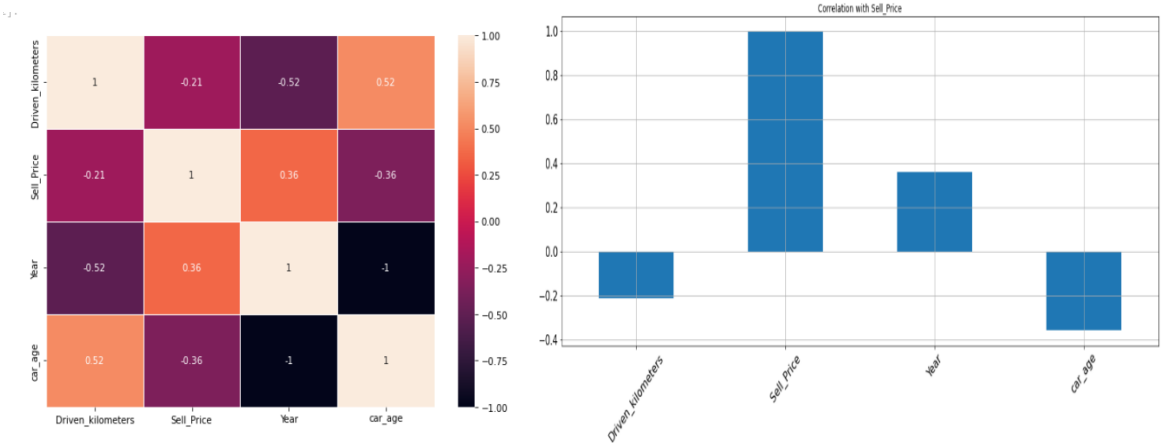
2.The price of an automatic car is almost double that of a manual car and the price maximum no. of used automatic cars lies betweem 10 to 30 lakh

3.Price of used car of almost brands is under 10 lakh except BMW, Mercedes_Benz, porsche,jaguar,audi,land,volvoand zeep.

4.The highest price of Porsche car.

Correlation

It is time to check for relationships among attributes. A correlation matrix was used to determine whether relationships exist.



- Both Driven_Kilometers and car_age is negative correlated with the target variable (car selling price) means the car which is older and driven higher is greater selling price

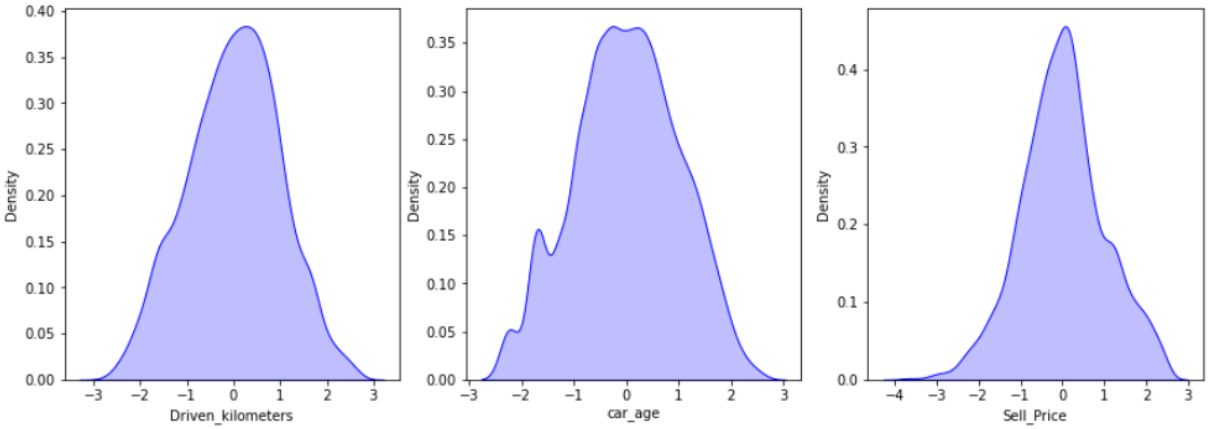
Data preparation

Pre-processing is a Data Mining technique that involves converting raw data into a comprehensible format. Pre-processing in Machine Learning is the process of modifying, calling, remove outliers, remove skewness, label encoding, drop unimportant column of data so that the machine can parse it more easily. Thus, the algorithm can now properly interpret the data.

I have done following steps for our data is to be clean:

- Drop unnecessary column.
- Remove outliers using z-score
- Remove skewness by using log Transform technique.
- Scaling our data by using standardisable.

After removing skewness our data is normalized .



Model building

since our target feature cell_price is an numerical value this became Regression problem.

Steps for machine learning model building:

- **Train Test Split**

```
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size = 0.20, random_state=100)
```

```
print("shape of x_train= ",x_train.shape)
print("shape of y_train= ",y_train.shape)
print("shape of x_test= ",x_test.shape)
print("shape of y_test= ",y_test.shape)
```

```
shape of x_train=  (4616, 6)
shape of y_train=  (4616,)
shape of x_test=  (1155, 6)
shape of y_test=  (1155,)
```

- **Finding Best Random State**

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
maxAccu=0
maxRS=0

for i in range(1,200):
    x_train, x_test,y_train, y_test=train_test_split(x,y,test_size=.25,random_state=i)
    LR=LinearRegression()
    LR.fit(x_train,y_train)
    predrf=LR.predict(x_test)
    r2=r2_score(y_test,predrf)
    if r2>maxAccu:
        maxr2=r2
        maxRS=i

print("Best r2 score is", maxr2,"on Random State",maxRS)
```

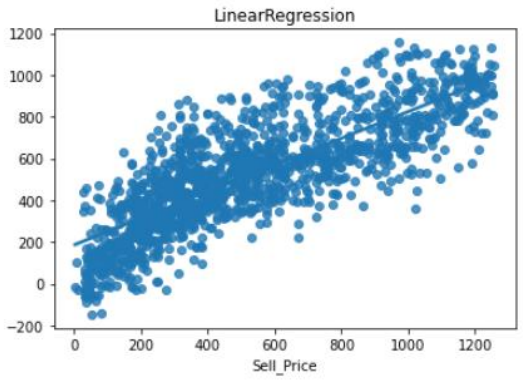
```
Best r2 score is 0.6437230029098344 on Random State 199
```

- **Testing various Regression models**

Linear Regression

Linear Regression Linear Regression was chosen as the first model due to its simplicity and comparatively small training time. The features, without any feature mapping, were used directly as the feature vectors. No regularization was used since the results clearly showed low variance

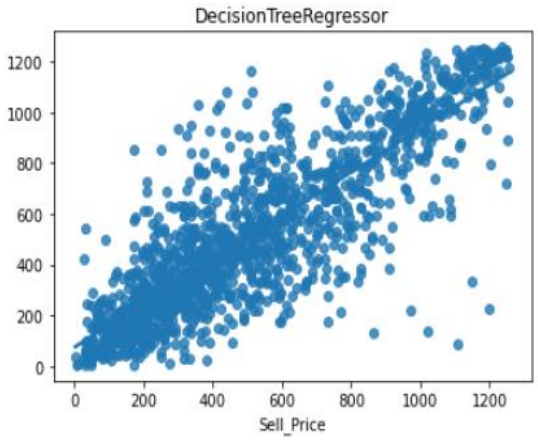
```
LinearRegression()


R2 Score is: 64.37230029098345


Cross Validation Score: 63.007135356069476


R2 Score - Cross Validation Score is 1.3651649349139703


RMSE Score is: 193.4145465329447
```



Decision Tree Classifier

Decision tree regression observes features of an object and trains a model in the structure of a tree to predict data in the future to produce meaningful continuous output. Continuous output means that the output/result is not discrete, i.e., it is not represented just by a discrete, known set of numbers or values

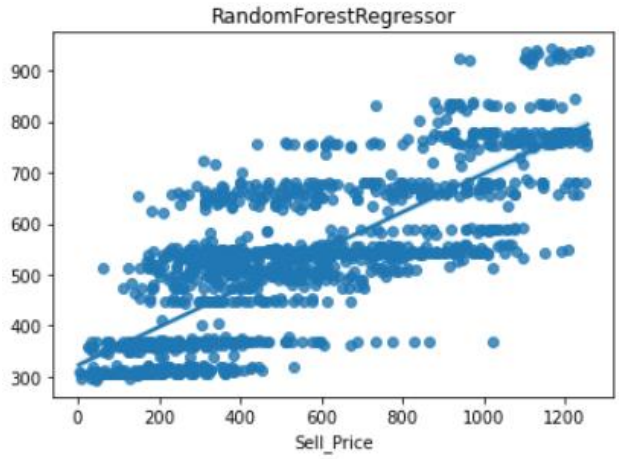
```
DecisionTreeRegressor(criterion='poisson', random_state=111)

R2 Score is: 70.65844002907404

Cross Validation Score: 63.98079267320844

R2 Score - Cross Validation Score is 6.677647355865602

RMSE Score is: 175.52413528692307
```



Random Forest regressor

Random Forest Random Forest is an ensemble learning based regression model. It uses a model called decision tree, specifically as the name suggests, multiple decision trees to generate the ensemble model which collectively produces a prediction. The benefit of this model is that the trees are produced in parallel and are relatively uncorrelated, thus producing good results as each tree is not prone to individual errors of other trees. This uncorrelated behavior is partly ensured by the use of Bootstrap Aggregation or bagging providing the randomness required to produce robust and uncorrelated trees. This model was hence chosen to account for the large number of features in the dataset and compare a bagging technique with the following gradient boosting methods.

```
RandomForestRegressor(max_depth=2, max_features='sqrt')

R2 Score is: 52.648487948191644

Cross Validation Score: 52.81625597207559

R2 Score - Cross Validation Score is -0.1677680238839443

RMSE Score is: 222.97807823473417
```

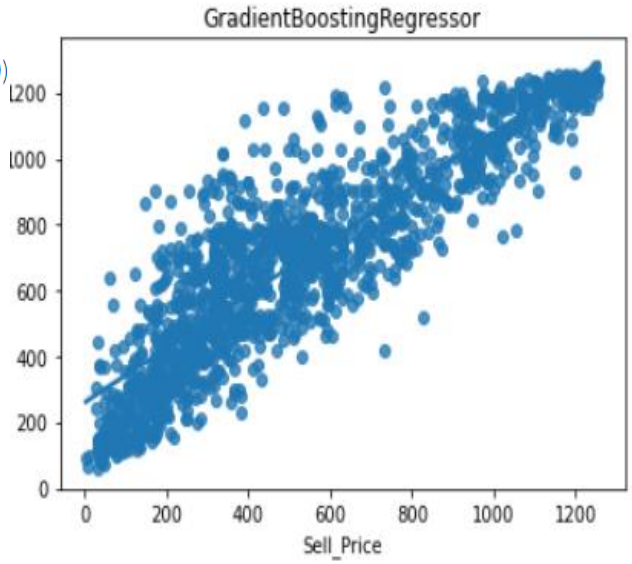


Gradient Boosting Regressor

Gradient Boost Gradient Boosting is another decision tree based method that is generally described as “a method of transforming weak learners into strong learners”. This means that

like a typical boosting method, observations are assigned different weights and based on certain metrics, the weights of difficult to predict observations are increased and then fed into another tree to be trained. In this case the metric is the gradient of the loss function. This model was chosen to account for non-linear relationships between the features and predicted price, by splitting the data into 100 regions

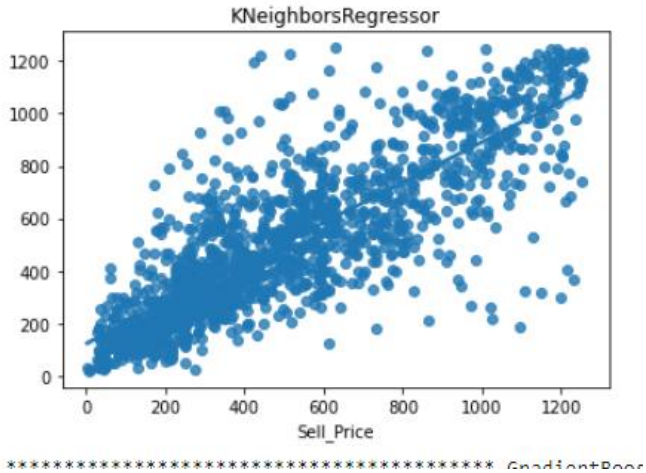
```
GradientBoostingRegressor(loss='quantile', max_depth=5, n_estimators=200)

R2 Score is: 50.65468384455703

Cross Validation Score: 51.28759329579864

R2 Score - Cross Validation Score is -0.6329094512416091

RMSE Score is: 227.6240835164656
```



K-Nearest Neighbors Algorithm

KNN regression is a non-parametric method that, in an intuitive manner, approximates the association between independent variables and the continuous outcome by averaging the observations in the same neighbourhood

```
KNeighborsRegressor(algorithm='kd_tree', n_neighbors=2)

R2 Score is: 66.67754649320115

Cross Validation Score: 61.86440200558106

R2 Score - Cross Validation Score is 4.813144487620086

RMSE Score is: 187.05259070892788
```



Ada Boost Regressor

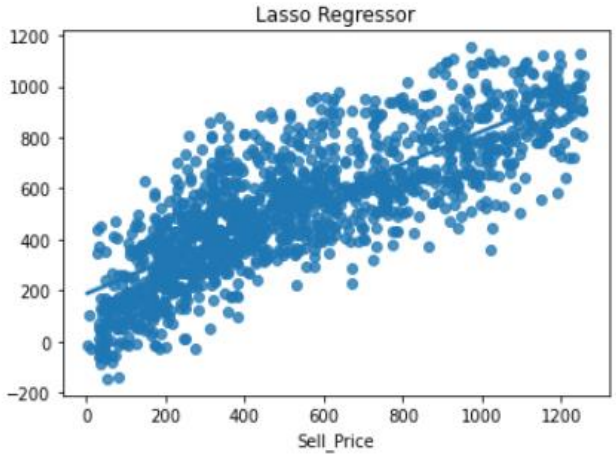
An AdaBoost regressor is a meta-estimator that begins by fitting a regressor on the original dataset and then fits additional copies of the regressor on the same dataset but where the weights of instances are adjusted according to the error of the current prediction.

```
AdaBoostRegressor(learning_rate=1.05, n_estimators=200, random_state=42)

R2 Score is: 62.52980973390236

Cross Validation Score: 63.75491513296419

R2 Score - Cross Validation Score is -1.2251053990618317

RMSE Score is: 198.35273422976647
```

Lasso Regression

Lasso regression is **a regularization technique**. It is used over regression methods for a more accurate prediction. This model uses shrinkage. Shrinkage is where data values are shrunk towards a central point as the mean. The lasso procedure encourages simple, sparse models

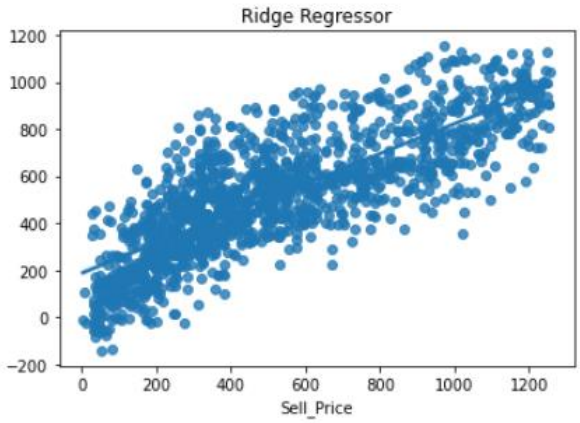
```
Lasso(alpha=0.01, max_iter=15, normalize=True)

R2 Score is: 64.35144878924018

Cross Validation Score: 63.005041224964216

R2 Score - Cross Validation Score is 1.3464075642759639

RMSE Score is: 193.47113724568743
```



Ridge Regression

Ridge regression is a method of estimating the coefficients of multiple-regression models in scenarios where the independent variables are highly correlated. It has been used in many fields including econometrics, chemistry, and engineering

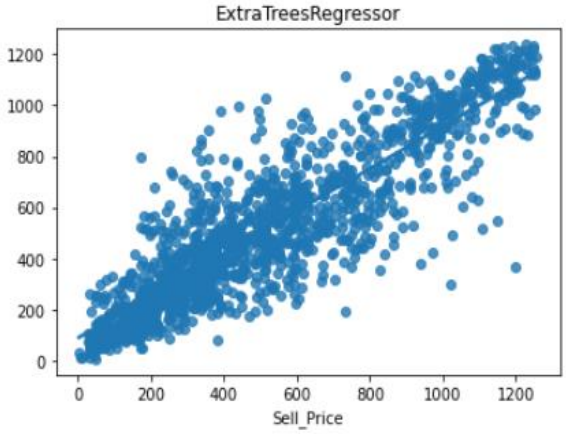
```
Ridge(alpha=0.01, normalize=True)

R2 Score is: 64.37625351111362

Cross Validation Score: 63.01758358324244

R2 Score - Cross Validation Score is 1.3586699278711762

RMSE Score is: 193.40381567610612
```



ExtraTreesRegressor

An extra-trees regressor. This class implements a meta estimator that fits a number of randomized decision trees (a.k.a. extra-trees) on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting

```
ExtraTreesRegressor(criterion='mae', n_jobs=1, random_state=199)

R2 Score is: 78.192665037211

Cross Validation Score: 76.34475791753688

R2 Score - Cross Validation Score is 1.8479071196741188

RMSE Score is: 151.32007208262257
```



- **Compare All Model Performance**

Comparing all the model and choose best model by r2-score and difference between CV and R2 score is minimum.

	Model	r2score	Cross_val_score	RMSE score	Difference between cv score and cross_val score
0	LinearRegression	64.372300	63.007135	193.414547	1.365165
1	Ridge Regressor	64.376254	63.017584	193.403816	1.358670
2	Lasso Regressor	64.351449	63.005041	193.471137	1.346408
3	DecisionTreeRegressor	70.658440	63.980793	175.524135	6.677647
4	RandomForestRegressor	52.648488	52.816256	222.978078	-0.167768
5	KNeighborsRegressor	66.677546	61.864402	187.052591	4.813144
6	GradientBoostingRegressor	50.654684	51.287593	227.624084	-0.632909
7	AdaBoostRegressor	62.529810	63.754915	198.352734	-1.225105
8	ExtraTreesRegressor	78.192665	76.344758	151.320072	1.847907

- **Hyperparameter tuning of best model**

ExtraTreesRegressor is our best model and difference between CV and R2 score is minimum for this particular model

```
# Choosing Extra Trees Regressor

param = {'criterion' : ['mse', 'mae'],
               'min_samples_split' : [2, 3],
                'n_estimators' : [100,200,500],
               'n_jobs' : [-1, 1]
              }
```

```
GSCV = GridSearchCV(ExtraTreesRegressor(),param, cv=5)
print(GSCV.fit(x_train,y_train))
print(GSCV.best_params_)
```

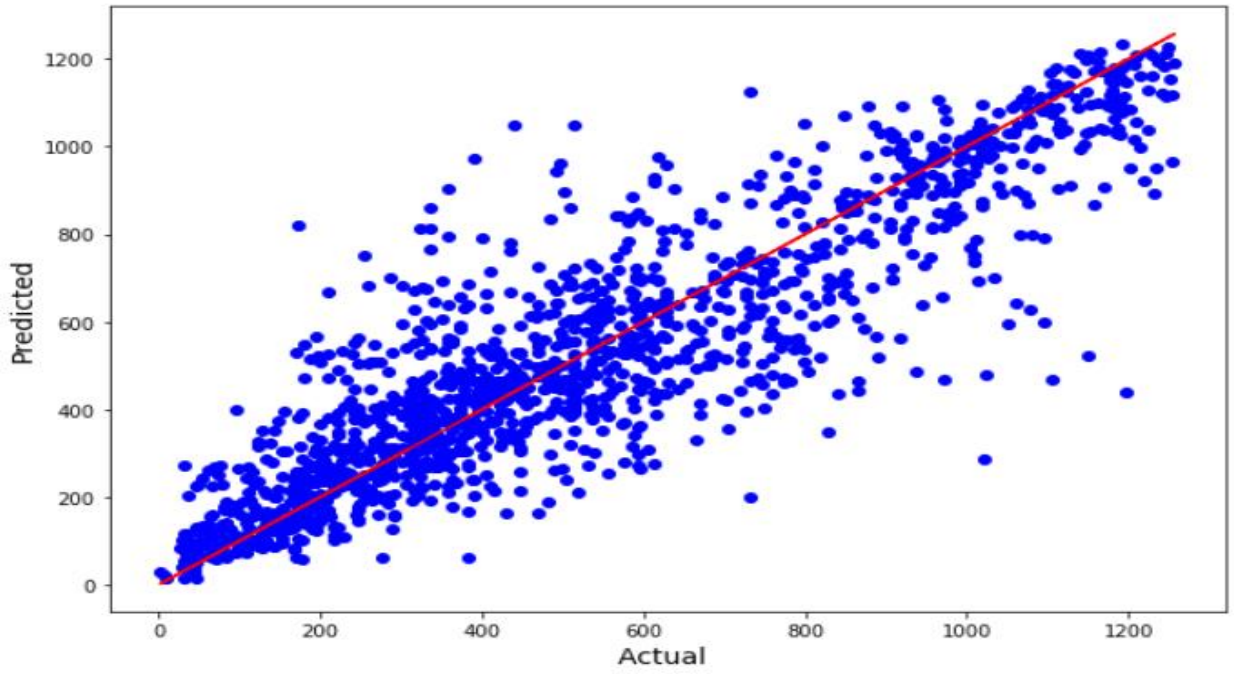
```
GridSearchCV(cv=5, estimator=ExtraTreesRegressor(),
             param_grid={'criterion': ['mse', 'mae'],
                         'min_samples_split': [2, 3],
                         'n_estimators': [100, 200, 500], 'n_jobs': [-1, 1]})
{'criterion': 'mae', 'min_samples_split': 3, 'n_estimators': 100, 'n_jobs': 1}
```

- **Model building and best line for Selected model**

Model building for best model by using selected parameter and best fit line.

```
Final_model= ExtraTreesRegressor(criterion='mae', min_samples_split= 3, n_estimators= 100, n_jobs= 1, random_state= 199)
#n_estimatorsint, default=100 The number of trees in the forest.

# Training the model
Final_model.fit(x_train, y_train)

# Predicting b_test
pred_Final_model= Final_model.predict(x_test)

# RMSE - a lower RMSE score is better than a higher one
rmse = mean_squared_error(y_test, pred_Final_model, squared=False)
print("RMSE Score is:", rmse)

# R2 score
r2 = r2_score(y_test, pred_Final_model, multioutput='variance_weighted')*100
print("R2 Score is:", r2)
```

```
RMSE Score is: 148.59512685610235
R2 Score is: 78.97099861706816
```



- **Saving, Loading and Conclusion the best Regression ML model**

Save and load the final model and predict the output by using test data

```
import pickle
filename_1="UsedCars_price.pkl"
pickle.dump(Final_model,open(filename_1,"wb"))
```

```
loaded_model=pickle.load(open("UsedCars_price.pkl","rb"))
result=loaded_model.score(x_test,y_test)
print(result*100)
```

```
78.97099861706816
```

	predicted	original
0	631.930	631.930
1	1029.460	1029.460
2	899.885	899.885
3	913.985	913.985
4	204.075	204.075
...	...	...
1438	534.700	534.700
1439	774.555	774.555
1440	209.735	209.735
1441	77.755	77.755
1442	289.880	289.880

1443 rows × 2 columns

Conclusion

Car price prediction can be a challenging task due to the high number of attributes that should be considered for the accurate prediction. The major step in the prediction process is collection and pre-processing of the data. In this project various methods of python were built to normalize, standardize and clean data to avoid unnecessary noise for machine learning algorithms.

Data cleaning is one of the processes that increases prediction performance, yet insufficient for the cases of complex data sets as the one in this research. Applying single machine algorithm on the data set accuracy was less than 60%. Therefore, the ensemble of multiple machine learning algorithms has been proposed and then the ExtaTree Regression ML methods gains accuracy of 78%. This is significant improvement compared to single machine learning method approach. However, the drawback of the proposed system is that it consumes much more computational resources than single machine learning algorithm. Although, this system has achieved astonishing performance in used car price prediction problem our aim for the future research is to test this system to work successfully with various data sets. We will extend our data with Car Dekho used cars data sets and validate the proposed approach.