FLIP ROBO

Project Report On

Car Price Prediction

Submitted By: OLIVER RAMAN | Internship 26

ACKNOWLEDGEMENT

The following research papers helped me understand the problem of car prices, the various factors affecting the pricing of a particular car, fluctuations in car prices and their causes & finally, helped me in my model building & predictions:

1."Used Cars Price Prediction using Supervised Learning Techniques"- Mukkesh Ganesh, Pattabiraman Venkatasubbu

The production of cars has been steadily increasing in the past decade, with over 70 million passenger cars being produced in the year 2016. This has given rise to the used car market, which on its own has become a booming industry. The recent advent of online portals has facilitated the need for both the customer and the seller to be better informed about the trends and patterns that determine the value of a used car in the market. Using Machine Learning Algorithms such as Lasso Regression, Multiple Regression and Regression trees, we will try to develop a statistical model which will be able to predict the price of a used car, based on previous consumer data and a given set of features.

2. "Price Prediction of Used Cars Using Machine Learning" - IEEE

This paper 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.

3. "Car Price Prediction Using Machine Learning" - Ketan Agrahari, Ayush Chaubey, Mamoor Khan, Manas Srivastava

The demand for used cars has increased significantly in the past decade and it is predicted that with Covid-19 outbreak this requirement will augment considerably. Hence to enhance the reliability, with the expansion of the used car market, a model that can forecast the current market price of a used automobile on the basis of a variety of criteria. This analysis can be used to study the trends in the industry, offer better insight into the market, and aid the community in its smooth workflow. The aim of this research paper is to predict the car price as per the data set (previous consumer data like engine capacity, distance traveled, year of manufacture, etc.). The result of these algorithms will be analyzed and based on the efficiency and accuracy of these algorithms, the best one of them can be used for the said purpose.

4. "Project Report: Predicting Used Car Prices" - Kshitij Kumbar, Pranav Gadre, Varun Nayak

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. We implement and evaluate various learning methods on a dataset consisting of the sale prices of different makes and models across cities in the United States. Our results show that Random Forest model and K-Means clustering with linear regression yield the best results, but are compute heavy. Conventional linear regression also yielded satisfactory results, with the advantage of a significantly lower training time in comparison to the aforementioned methods.

INTRODUCTION

Business Problem Framing

With the covid 19 impact in the market, we have seen a lot of changes in the car market. Now some cars are in demand hence making them costly and some are not in demand hence cheaper. Predicting the price of used cars is an important and interesting problem. Predicting the resale value of a car is not a simple task. It is trite knowledge that the value of used cars depends on a number of factors. The most important ones are usually the age of the car, its make (model), the origin of the car (location of the manufacturer), its mileage (the number of kilometers it has run) and its horsepower (amount of power that the engine produces). 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 a car price valuation model.

Conceptual Background of the Domain Problem

The prices of new cars 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 increasing prices of new cars and the inability of customers to buy new cars due to the lack of funds, used cars sales are on a global increase.

Car Price Prediction is really an interesting machine learning problem as there are many factors that influence the price of a car in the second-hand market. There is a need for a used car price prediction system to effectively determine the worthiness of the car using a variety of features.

Thus, it is of commercial interest to sellers to be able to predict the salvage value (residual value) of cars with accuracy. If the residual value is under-estimated by the seller at the beginning, the installments will be higher for the clients who will certainly opt for another seller. If the residual value is overestimated, the installments will be lower for the clients but then the seller may have much difficulty at selling these high-priced used cars at this over-estimated residual value. Thus, we can see that estimating the price of used cars is of very high commercial importance as well.

There are websites that offer this service, but their prediction method may not always be the best. It is important to know their actual market value while both buying and selling.
Here we are trying to help the client work with small traders, who sell used cars to understand the price of the used cars by deploying machine learning models. These models would help the client/sellers to understand the used car market and accordingly they would be able to sell the used car in the market.

Review of Literature

Several studies and related works have been done previously to predict used car prices around the world using different methodologies and approaches, with varying results of accuracy from 50% to 90%.

Galarraga et al. (2014) used the European labeling system as a new alternative indicator for energy efficiency for light cars that classify cars according to their relative fuel consumption levels. They applied the hedonic price method to estimate the price functions for cars and thus to obtain the marginal price of highly rated cars in terms of energy efficiency.

(Pudaruth, 2014) the researcher proposed to predict used car prices in Mauritius, where he applied different machine learning techniques to achieve his results like decision tree, K-nearest neighbors, Multiple Regression and Naïve Bayes algorithms to predict the used cars prices, based on historical data gathered from the newspaper.

(Gegic, Isakovic, Keco, Masetic, & Kevric, 2019) from the International Burch University in Sarajevo, used three different machine learning techniques to predict used car prices. Using data scraped from a local Bosnian website for used cars totalled at 797 car samples after pre-processing, and proposed using these methods: Support Vector Machine, Random Forest and Artificial Neural network.

Pal et al. (2018) used Random Forest, a controlled learning method to estimate the price of used cars. The model was chosen after careful exploration data analysis to determine the effect of each feature on the price.

(Monburinon, et al., 2018) Gathered data from a German e-commerce site that totalled to 304,133 rows and 11 attributes to predict the prices of used cars using different techniques and measured their results using Mean Absolute Error (MEA) to compare their results. Same training dataset and testing dataset was given to each model.

Dastan (2016) aimed to determine the factors affecting second hand car prices. For this purpose, horizontal cross-sectional data obtained from second hand car advertisements on websites were used. Indeed, it has been found that many features such as the front view camera, the brand, model of the car, age, traction, mileage, gear, fuel type, torque, width, fuel tank volume, ABS, panoramic glass roof, rear window defroster, power steering, start / stop, sunroof, cooled torpedo affect the price of the car.

Motivation for the Problem Undertaken

In this project, I have to build a model that calculates the price of used cars, with available independent variables. This model will then be used by the management to understand how exactly the prices vary with the variables.
This is my third internship project and the second one in which I have to build a machine learning model. Also, we had to scrape the data for the model using various web scraping techniques.
By doing this project I have got an idea about how to deal with web scraping, data exploration & model building, where I used my analytic skills to predict the optimal used car prices using ML models. Further, the model will be a good way for the management to understand the pricing dynamics of the 'used car' market.

ANALYTICAL PROBLEM FRAMING

Mathematical/Analytical Modeling of the Problem

As a first step I have scrapped the required data from cardekho.com website. I have fetched data for different locations and saved it to excel format & csv format.

In this particular problem I have "Price" as my target column and it was a continuous column. So clearly it is a regression problem and I have to use all regression algorithms while building the model.

- There were no null values in the dataset.
- Since we have scrapped the data from cardekho.com website the raw data was not in the format, so we have used feature engineering to extract the required feature format.
- To get better insight on the features I have used plotting like distribution plot, bar plot, reg plot, pie plot, scatter plot and count plot. With these plots I was able to understand the relation between the features in a better manner.
- Also, I found outliers and skewness in the dataset so I removed outliers using z-score method and I removed skewness using yeo-johnson method.
- I have used all the regression algorithms while building models, then tuned the best model and saved the best model. At last I have predicted the car-price using the saved model.

Data Sources & their formats

The data was collected from cardekho.com website in excel & csv format. The data was scraped using selenium. After scrapping required features the dataset is saved as an excel file & csv file.
Also, my dataset had 2880 rows and 6 columns including target. The information about features is as follows.

★ Brand : Brand Name of the car
★ Fuel : Type of fuel used for car engine
★ KMS_driven : Number of kilometers driven
★ Variant : Whether automatic or manual
★ Model : The car model/type
★ Manuf_Year : Year of manufacture of the car

★ Price : Price of the car

Data preprocessing done

- As a first step I have scrapped the required data using selenium from cardekho.com website.
- And I have imported required libraries and I have imported the dataset which was saved in csv format.
- Then I did all the statistical analysis like checking shape, columns, data types, nunique, value counts, info etc.
- While checking the data types of the columns I found all the columns to be of object data type, so I changed the data types of the columns to the correct data types.
- I also extracted the car manufacturing year from the brand column using feature extraction techniques.
- While checking for null values I found no null values in the dataset.
- There were no empty values present in the dataset.
- Performed univariate, bivariate and multivariate analysis to visualize the data. Visualized each feature using seaborn and matplotlib libraries by plotting several categorical and numerical plots like pie plot, count plot, bar plot, distribution plot, box plots and pair plot.
- Next, I used Label Encoder to encode all the object data type columns to make it easier for model building and visualization.
- Identified outliers using box plots and removed outliers in columns using Z Score Method and stored the data frame after removing outliers as "df_usedcars".
- Checked for skewness and removed skewness in numerical columns using power transformation method (yeo-johnson).
- Separated features and target variables and conducted feature scaling using Standard Scaler method to avoid any kind of data biasness.
- Checked Variance Inflation Factor (VIF) got rid of high multicollinearity issues if present.

Data Inputs- Logic- Output Relationships

The dataset consists of a target and other features. The features are independent and the target is dependent, as the values of our independent variables change, so does our target variable change.
To analyze the relation between features and target I have done EDA where I analyzed the relation using many plots like bar plot, reg plot, scatter plot, pie plot, count plot, pair plot etc.
I have checked the correlation between the target and features using heat map and bar plot, where I got the positive and negative correlation between the label and features.

Important features that affect Price positively and negatively are as follows:

<u>Features having high Positive correlation with target:</u> Manuf_Year.

<u>Features having high Negative correlation with target:</u> Fuel & Variant.

Hardware and Software Requirements & Tools used

To build the machine learning projects it is important to have the following hardware and software requirements and tools.

Hardware required:

- Processor: core i5 or above
- RAM: 8 GB or above
- ROM/SSD: 250 GB or above

Software required:

- Anaconda - language used Python 3

Libraries Used:

- import numpy as np
- import pandas as pd
- import seaborn as sns
- import matplotlib.pyplot as plt
- from sklearn.preprocessing import LabelEncoder
- from sklearn.preprocessing import StandardScaler
- from statsmodels.stats.outliers_influence import variance_inflation_factor
- from sklearn.linear_model.LinearRegression
- from sklearn.tree import DecisionTreeRegressor
- from sklearn.neighbors.KNeighborsRegressor
- from sklearn.svm.SVR
- from sklearn.ensemble import RandomForestRegressor
- from xgboost import XGBRegressor
- from sklearn.ensemble import GradientBoostingRegressor
- from sklearn.ensemble import ExtraTreesRegressor
- from sklearn.model_selection import cross_val_score
- from sklearn.model_selection import GridSearchCV

MODEL DEVELOPMENT & EVALUATION

Identification of possible problem-solving approaches (methods)

I have used feature extraction & conversion methods to replace the data types and extract values from columns in the dataset. To encode the categorical columns I have used Label Encoding.
To remove outliers I have used the Z Score method. And to remove skewness I have used the yeo-johnson method.
Use of Pearson's correlation coefficient to check the correlation between dependent and independent features. Also I have used standardization. Then followed by model building with all regression algorithms.

Testing of Identified Approaches (Algorithms)

Since Price was my target and it was a continuous column so this particular problem was a regression problem. And I have used all regression algorithms to build my model.
By looking into the r2 score and cross validation score I found ExtraTreesRegressor as a best model with high scores. Also to get the best model we have to run through multiple models and to avoid the confusion of overfitting we have to go through cross validation.

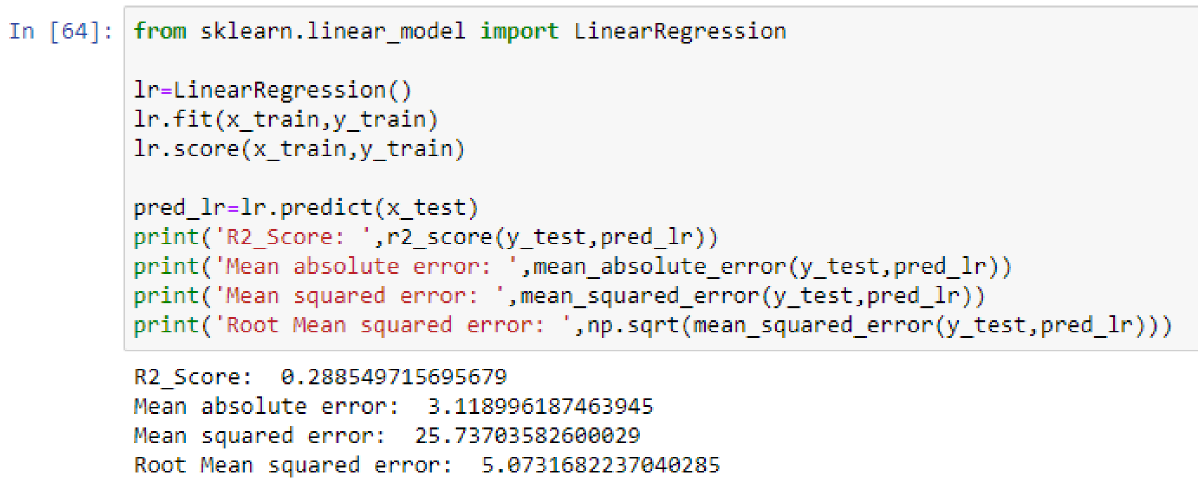
Below is the list of regression algorithms I have used in my project.

- LinearRegression
- SVR
- KNearestNeighborsRegressor
- RandomForestRegressor
- XGBRegressor
- ExtraTreesRegressor
- GradientBoostingRegressor
- DecisionTreeRegressor

Run and Evaluate selected models

Regression Models:

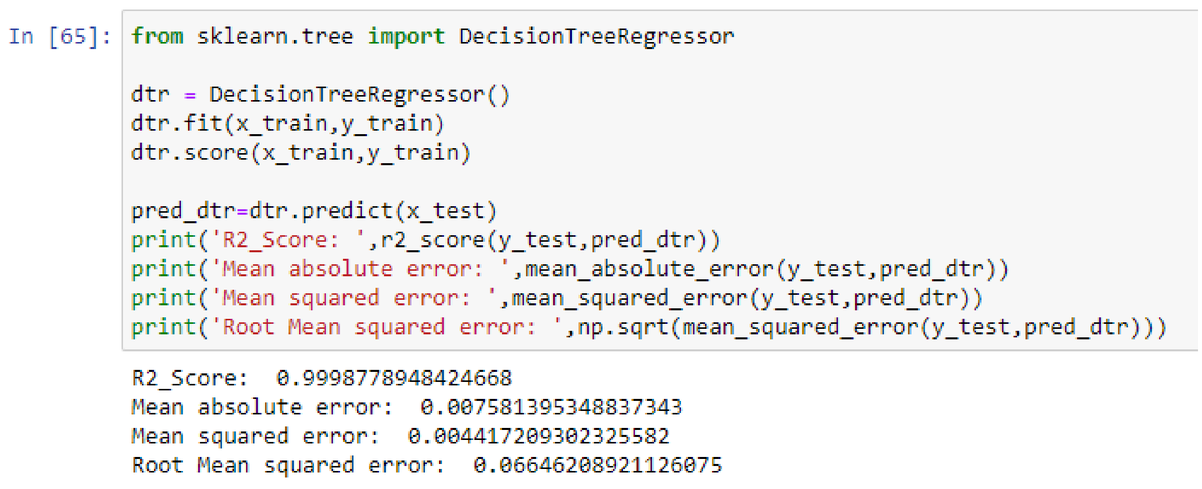
LINEAR REGRESSION:



```
In [64]: from sklearn.linear_model import LinearRegression

         lr=LinearRegression()
         lr.fit(x_train,y_train)
         lr.score(x_train,y_train)

         pred_lr=lr.predict(x_test)
         print('R2_Score: ',r2_score(y_test,pred_lr))
         print('Mean absolute error: ',mean_absolute_error(y_test,pred_lr))
         print('Mean squared error: ',mean_squared_error(y_test,pred_lr))
         print('Root Mean squared error: ',np.sqrt(mean_squared_error(y_test,pred_lr)))

         R2_Score:  0.288549715695679
         Mean absolute error:  3.118996187463945
         Mean squared error:  25.73703582600029
         Root Mean squared error:  5.0731682237040285
```

The Linear Regression model gave us an R2 Score of 28.85 %.

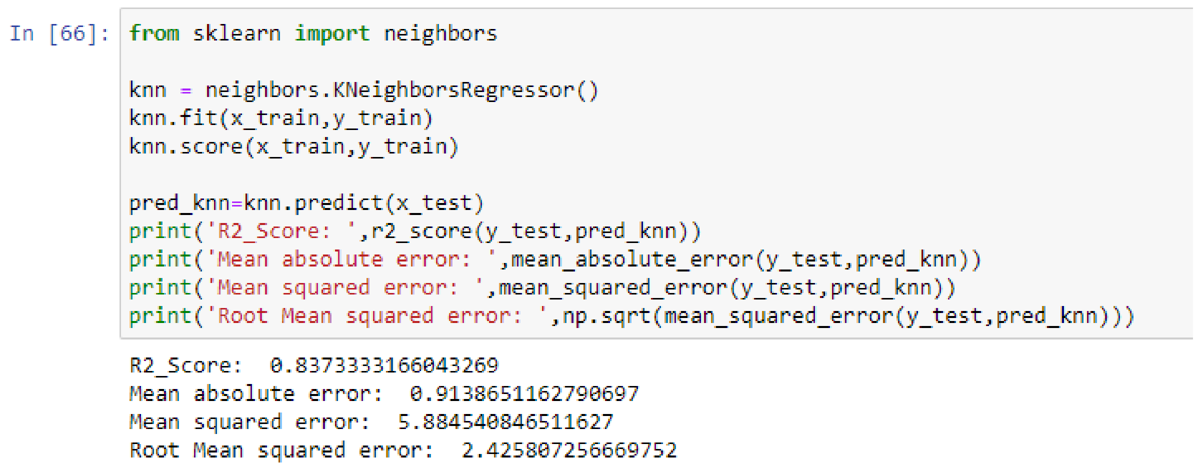
DECISION TREE REGRESSOR:



```
In [65]: from sklearn.tree import DecisionTreeRegressor

         dtr = DecisionTreeRegressor()
         dtr.fit(x_train,y_train)
         dtr.score(x_train,y_train)

         pred_dtr=dtr.predict(x_test)
         print('R2_Score: ',r2_score(y_test,pred_dtr))
         print('Mean absolute error: ',mean_absolute_error(y_test,pred_dtr))
         print('Mean squared error: ',mean_squared_error(y_test,pred_dtr))
         print('Root Mean squared error: ',np.sqrt(mean_squared_error(y_test,pred_dtr)))

         R2_Score:  0.9998778948424668
         Mean absolute error:  0.007581395348837343
         Mean squared error:  0.004417209302325582
         Root Mean squared error:  0.06646208921126075
```

The Decision Tree Regressor Model gave us a R2 Score of 99.98 %.

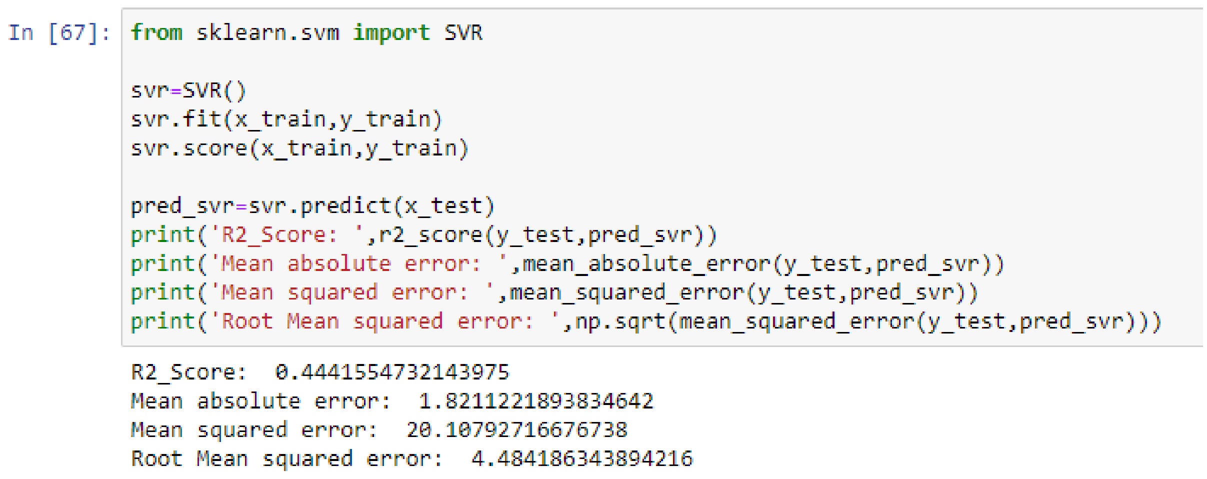
KNEAREST NEIGHBORS REGRESSOR:



```
In [66]: from sklearn import neighbors

         knn = neighbors.KNeighborsRegressor()
         knn.fit(x_train,y_train)
         knn.score(x_train,y_train)

         pred_knn=knn.predict(x_test)
         print('R2_Score: ',r2_score(y_test,pred_knn))
         print('Mean absolute error: ',mean_absolute_error(y_test,pred_knn))
         print('Mean squared error: ',mean_squared_error(y_test,pred_knn))
         print('Root Mean squared error: ',np.sqrt(mean_squared_error(y_test,pred_knn)))

         R2_Score:  0.8373333166043269
         Mean absolute error:  0.9138651162790697
         Mean squared error:  5.884540846511627
         Root Mean squared error:  2.425807256669752
```

The KNearest Neighbors Regression Model gave us a R2 Score of 83.73 %.

SUPPORT VECTOR REGRESSOR (SVR):



```
In [67]: from sklearn.svm import SVR

         svr=SVR()
         svr.fit(x_train,y_train)
         svr.score(x_train,y_train)

         pred_svr=svr.predict(x_test)
         print('R2_Score: ',r2_score(y_test,pred_svr))
         print('Mean absolute error: ',mean_absolute_error(y_test,pred_svr))
         print('Mean squared error: ',mean_squared_error(y_test,pred_svr))
         print('Root Mean squared error: ',np.sqrt(mean_squared_error(y_test,pred_svr)))

         R2_Score:  0.4441554732143975
         Mean absolute error:  1.8211221893834642
         Mean squared error:  20.10792716676738
         Root Mean squared error:  4.484186343894216
```

The SVR Model gave us a R2 Score of 44.41 %.

RANDOM FOREST REGRESSOR:

```
In [68]: from sklearn.ensemble import RandomForestRegressor

         rfr = RandomForestRegressor()
         rfr.fit(x_train,y_train)
         rfr.score(x_train,y_train)

         pred_rfr=rfr.predict(x_test)
         print('R2_Score: ',r2_score(y_test,pred_rfr))
         print('Mean absolute error: ',mean_absolute_error(y_test,pred_rfr))
         print('Mean squared error: ',mean_squared_error(y_test,pred_rfr))
         print('Root Mean squared error: ',np.sqrt(mean_squared_error(y_test,pred_rfr)))

         R2_Score:  0.9952502742515288
         Mean absolute error:  0.17314732558139714
         Mean squared error:  0.17182347726744163
         Root Mean squared error:  0.4145159553834347
```

The Random Forest Regression Model gave us a R2 Score of 99.52 %.

GRADIENT BOOSTING REGRESSOR:

```
In [69]: from sklearn.ensemble import GradientBoostingRegressor

         gbr=GradientBoostingRegressor()
         gbr.fit(x_train,y_train)
         gbr.score(x_train,y_train)

         pred_gbr=gbr.predict(x_test)
         print('R2_Score: ',r2_score(y_test,pred_gbr))
         print('Mean absolute error: ',mean_absolute_error(y_test,pred_gbr))
         print('Mean squared error: ',mean_squared_error(y_test,pred_gbr))
         print('Root Mean squared error: ',np.sqrt(mean_squared_error(y_test,pred_gbr)))

         R2_Score:  0.90366480525037
         Mean absolute error:  1.3012580701924499
         Mean squared error:  3.484969242791666
         Root Mean squared error:  1.8668072323600169
```

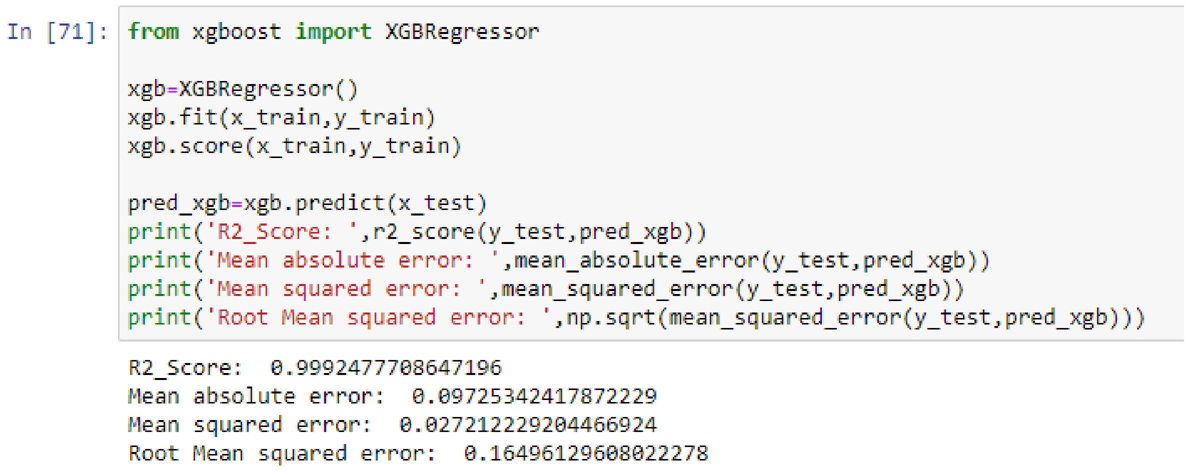
The Gradient Boosting Regressor Model gave us a R2 Score of 90.36 %.

EXTRA TREES REGRESSOR:

```
In [70]: from sklearn.ensemble import ExtraTreesRegressor

         etr=ExtraTreesRegressor()
         etr.fit(x_train,y_train)
         etr.score(x_train,y_train)

         pred_etr=etr.predict(x_test)
         print('R2_Score: ',r2_score(y_test,pred_etr))
         print('Mean absolute error: ',mean_absolute_error(y_test,pred_etr))
         print('Mean squared error: ',mean_squared_error(y_test,pred_etr))
         print('Root Mean squared error: ',np.sqrt(mean_squared_error(y_test,pred_etr)))

         R2_Score:  0.999771449398171
         Mean absolute error:  0.009780465116287402
         Mean squared error:  0.00826792139534876
         Root Mean squared error:  0.09092811113923328
```

The Extra Trees Regressor Model gave us a R2 Score of 99.97 %.

XGBOOST REGRESSOR:



```
In [71]: from xgboost import XGBRegressor

         xgb=XGBRegressor()
         xgb.fit(x_train,y_train)
         xgb.score(x_train,y_train)

         pred_xgb=xgb.predict(x_test)
         print('R2_Score: ',r2_score(y_test,pred_xgb))
         print('Mean absolute error: ',mean_absolute_error(y_test,pred_xgb))
         print('Mean squared error: ',mean_squared_error(y_test,pred_xgb))
         print('Root Mean squared error: ',np.sqrt(mean_squared_error(y_test,pred_xgb)))

         R2_Score:  0.9992477708647196
         Mean absolute error:  0.09725342417872229
         Mean squared error:  0.027212229204466924
         Root Mean squared error:  0.16496129608022278
```

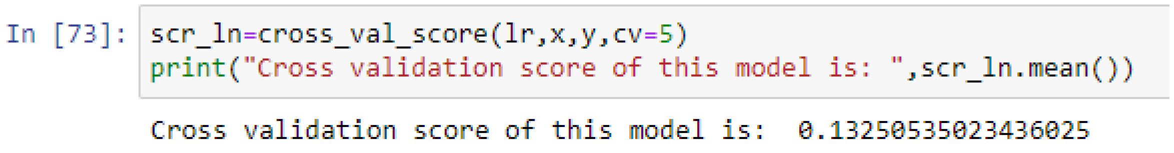
The XGBoost Regressor Model gave us a R2 Score of 99.92 %.

From the above regression models, the highest R2 score belongs to Decision Tree Regressor. Followed by Extra Trees Regressor, XGBoost Regressor & Random Forest Regressor.

Next, Gradient Boosting Regressor & KNearest Neighbors Regressor.

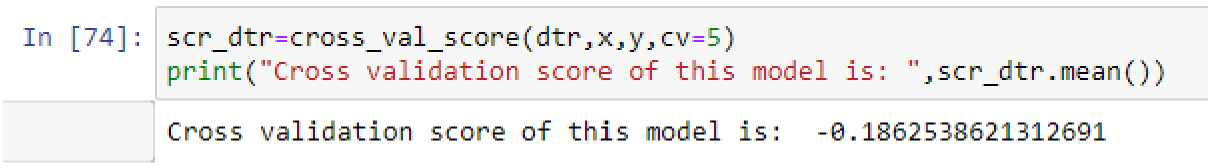
The lowest R2 scores belong to SVR Model & Linear Regression Model.

Cross Validation Scores:



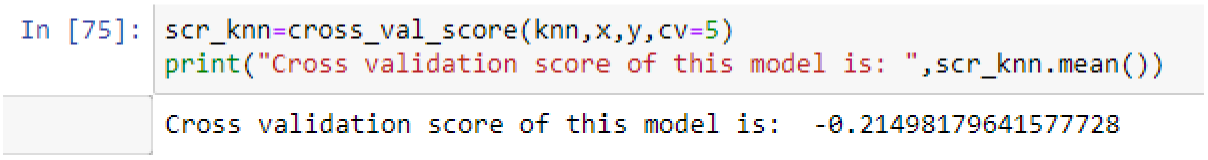
```
In [73]: scr_ln=cross_val_score(lr,x,y,cv=5)
         print("Cross validation score of this model is: ",scr_ln.mean())

         Cross validation score of this model is:  0.13250535023436025
```

The cross validation score of the Linear Regression Model is 13.25 %



```
In [74]: scr_dtr=cross_val_score(dtr,x,y,cv=5)
         print("Cross validation score of this model is: ",scr_dtr.mean())

         Cross validation score of this model is:  -0.1862538621312691
```

The cross validation score of the Decision Tree Regressor Model is -18.62 %.



```
In [75]: scr_knn=cross_val_score(knn,x,y,cv=5)
         print("Cross validation score of this model is: ",scr_knn.mean())

         Cross validation score of this model is:  -0.21498179641577728
```

The cross validation score of the KNearest Neighbors Regression Model is -21.49 %.

```
In [76]: scr_svr=cross_val_score(svr,x,y,cv=5)
         print("Cross validation score of this model is: ",scr_svr.mean())

         Cross validation score of this model is:  0.28240254110337226
```

The cross validation score of the SVR Model is 28.24 %.

```
In [77]: scr_rfr=cross_val_score(rfr,x,y,cv=5)
         print("Cross validation score of this model is: ",scr_rfr.mean())

         Cross validation score of this model is:  0.3055737115710514
```

The cross validation score of the Random Forest Regressor Model is 30.55 %.

```
In [78]: scr_gbr=cross_val_score(gbr,x,y,cv=5)
         print("Cross validation score of this model is: ",scr_gbr.mean())

         Cross validation score of this model is:  0.3141536785578286
```

The cross validation score of the Gradient Boosting Regressor Model is 31.41 %.

```
In [79]: scr_etr=cross_val_score(etr,x,y,cv=5)
         print("Cross validation score of this model is: ",scr_etr.mean())

         Cross validation score of this model is:  0.46656127020744903
```

The cross validation score of the Extra Trees Regressor Model is 46.65 %.

```
In [80]: scr_xgb=cross_val_score(xgb,x,y,cv=5)
         print("Cross validation score of this model is: ",scr_xgb.mean())

         Cross validation score of this model is:  0.28010972886126123
```

The cross validation score of the XGBoost Regressor Model is 28.01 %.

The highest cross validation score belongs to Extra Trees Regressor, followed by Gradient Boosting Regressor & Random Forest Regressor.

Next, XGBoost Regressor & SVR Model.

The cross validation scores of KNearest Neighbors Regressor, Decision Tree Regressor & Linear Regression Model are negative and are the lowest of all the scores.

Hyper Parameter Tuning:

Since the R2 Score & Cross Validation Score are both the highest in Extra Trees Regressor we shall consider it for hyper parameter tuning.

We will use GridSearchCV for hyper parameter tuning.

```
In [81]: from sklearn.model_selection import GridSearchCV
```

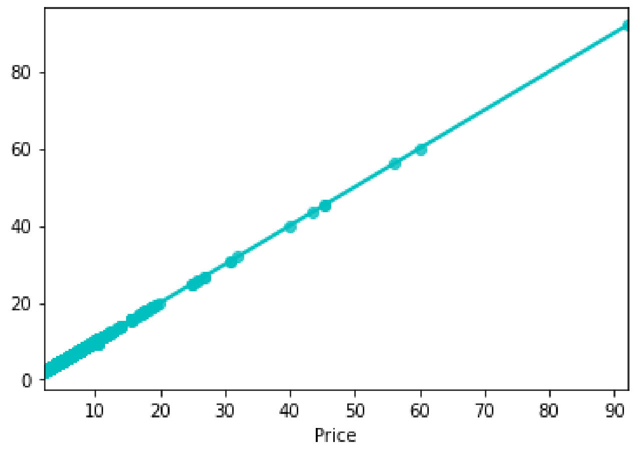
```
In [82]: parameters = {'n_estimators':[10,100,1000],
                       'criterion':['squared_error','mse','absolute_error','mae'],
                       'max_features':['auto','sqrt','log2'],
                       'n_jobs':[-2,-1,1]}
         grid_etr = GridSearchCV(etr, param_grid = parameters, cv = 5)
```

```
In [83]: grid_etr.fit(x_train, y_train)

Out[83]: GridSearchCV(cv=5, estimator=ExtraTreesRegressor(),
                      param_grid={'criterion': ['squared_error', 'mse', 'absolute_error',
                                                'mae'],
                                  'max_features': ['auto', 'sqrt', 'log2'],
                                  'n_estimators': [10, 100, 1000],
                                  'n_jobs': [-2, -1, 1]})
```

```
In [84]: grid_etr.best_params_

Out[84]: {'criterion': 'mse', 'max_features': 'log2', 'n_estimators': 10, 'n_jobs': 1}
```

```
In [85]: Best_model = ExtraTreesRegressor(criterion='mse',max_features='log2',n_jobs=1,n_estimators=10)
         Best_model.fit(x_train,y_train)

         pred = Best_model.predict(x_test)
         print('R2_Score: ',r2_score(y_test,pred))
         print('Mean absolute error: ',mean_absolute_error(y_test,pred))
         print('Mean squared error: ',mean_squared_error(y_test,pred))
         print('Root Mean squared error: ',np.sqrt(mean_squared_error(y_test,pred)))

         R2_Score:  0.9997575731119528
         Mean absolute error:  0.009632558139535638
         Mean squared error:  0.008769902325581388
         Root Mean squared error:  0.09364775665002012
```

After Hyper Parameter Tuning, we have got a better R2 score of 99.97 %.



Saving the final model and predicting the saved model

Now we shall save the best model.

```
In [87]: import joblib
         joblib.dump(Best_model,"Used_Car_Price_Prediction.pkl")

Out[87]: ['Used_Car_Price_Prediction.pkl']
```

```
In [88]: # Loading the saved model
         used_car_model=joblib.load("Used_Car_Price_Prediction.pkl")

         # Prediction
         prediction = used_car_model.predict(x_test)
         prediction

Out[88]: array([ 8.05 ,  3.71 ,  4.4  ,  4.6  ,  9.29 ,  3.7  , 11.79 ,  3.2  ,
                 4.1  ,  3.85 ,  4.4  ,  3.26 ,  9.75 ,  5.4  ,  5.5  ,  8.12 ,
                 5.54 ,  4.2  , 10.72 , 19.35 ,  6.15 ,  5.4  ,  5.98 ,  3.35 ,
                 3.95 ,  6.92 ,  5.04 ,  3.05 ,  7.05 ,  6.25 ,  5.52 ,  6.92 ,
                 5.74 ,  3.86 ,  2.57 ,  7.25 ,  6.25 ,  6.78 ,  9.95 ,  8.59 ,
                 2.61 , 32.   ,  9.26 ,  3.8  ,  4.4  ,  3.4  ,  3.28 , 11.7  ,
                 2.91 ,  4.69 ,  3.34 ,  3.8  ,  6.67 ,  5.11 ,  8.59 ,  8.6  ,
                 6.   ,  4.35 ,  4.83 ,  5.85 ,  8.59 ,  8.25 ,  4.4  ,  6.28 ,
                30.7  ,  9.8  ,  8.91 ,  7.6  ,  3.1  ,  8.34 ,  3.73 , 12.7  ,
                 7.25 ,  2.9  , 13.49 , 18.8  ,  7.3  ,  4.63 ,  2.46 ,  9.85 ,
                 8.23 ,  5.54 ,  9.06 ,  5.07 ,  5.15 ,  8.93 ,  3.4  ,  9.89 ,
                 5.04 ,  5.85 ,  4.95 ,  4.5  ,  3.8  ,  4.58 ,  4.35 ,  4.75 ,
                 7.03 ,  3.1  ,  9.85 ,  6.88 ,  4.96 ,  3.4  , 12.4  ,  4.312,
                15.59 , 10.23 ,  2.35 , 13.85 ,  6.26 ,  3.05 , 18.77 ,  3.8  ,
                 4.39 ,  2.45 ,  4.1  ,  4.82 ,  7.8  ,  7.05 , 25.73 ,  4.75 ,
```

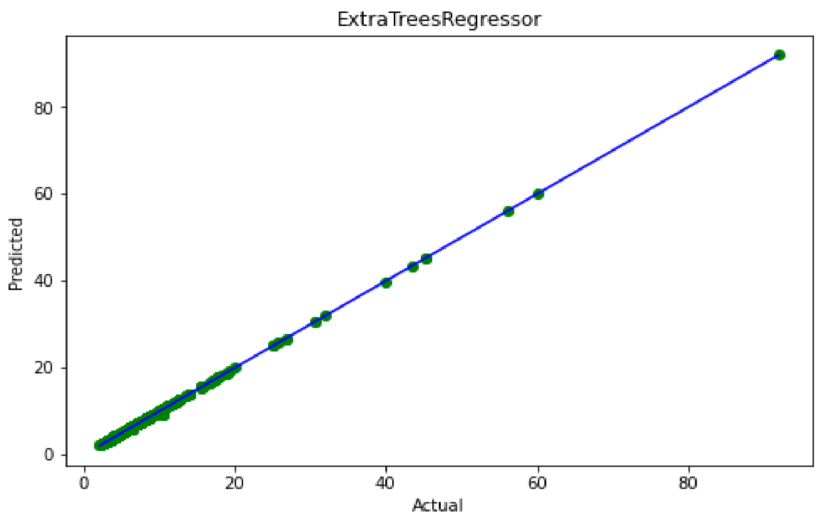
Putting the predicted & actual values in a dataframe.

```
In [89]: pd.DataFrame([used_car_model.predict(x_test)[:],y_test[:]],index=["Predicted","Actual"])

Out[89]:
```

	0	1	2	3	4	5	6	7	8	9	...	850	851	852	853	854	855	856	857	858	859
Predicted	8.05	3.71	4.4	4.6	9.29	3.7	11.79	3.2	4.1	3.85	...	5.13	6.47	3.75	5.85	25.0	8.01	4.75	12.24	4.75	3.41
Actual	8.05	3.71	4.4	4.6	9.29	3.7	11.79	3.2	4.1	3.85	...	5.13	6.47	3.75	5.85	25.0	8.01	4.75	12.24	4.75	3.41

2 rows × 860 columns

Graph of predicted and actual values.



The plot gives the linear relation between predicted and actual price of the used cars. The best fitting line gives the actual values and green dots give the predicted values.

Key Metrics for success in solving problem under consideration

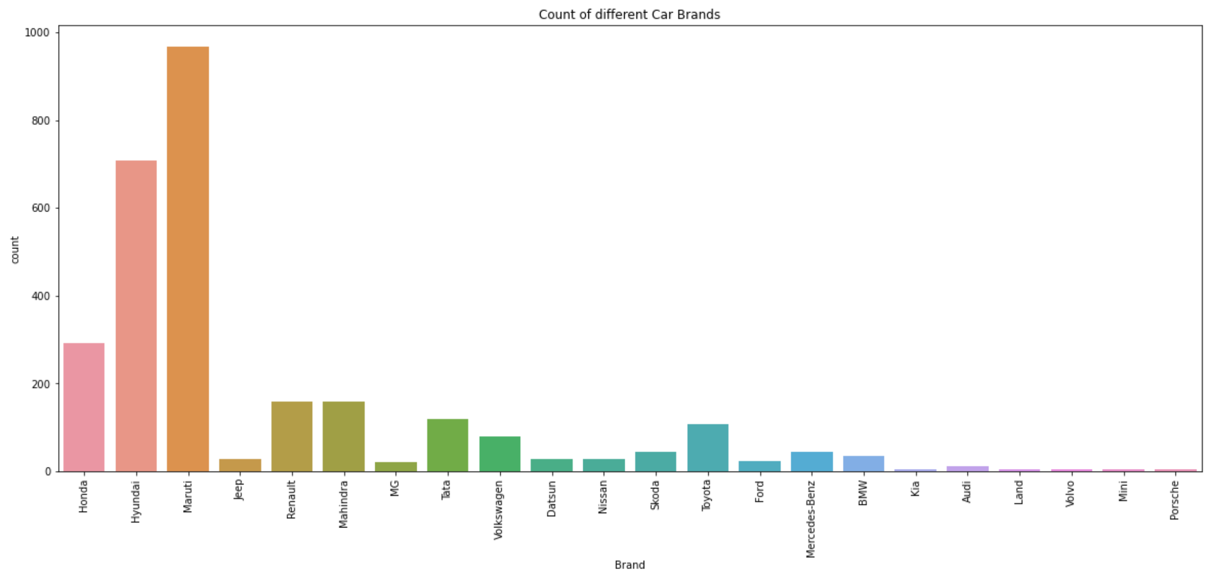
I have used the following metrics for evaluation:

- I have used an r2 score which tells us how accurate our model is.
- I have used mean absolute error which gives a magnitude of difference between the prediction of an observation and the true value of that observation.
- I have used root mean square deviation as one of the most commonly used measures for evaluating the quality of predictions.

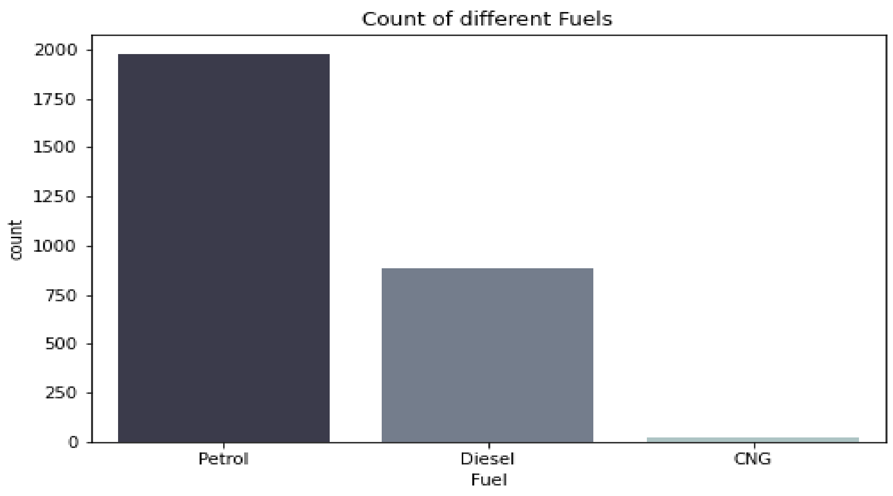
Visualizations

I have used count plots & pie charts to conduct univariate analysis of the features. I used a distribution plot for the univariate analysis of the target variable.

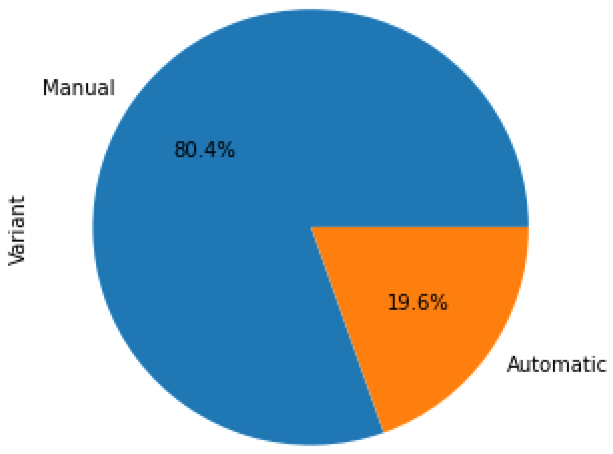
UNIVARIATE ANALYSIS:



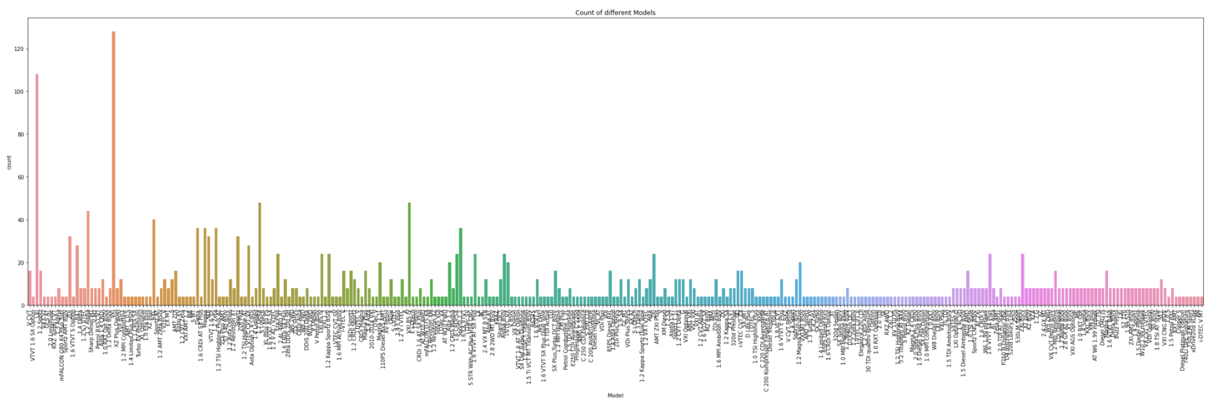
Observation: Most of the used cars are of the brand Maruti, followed by Hyundai, Honda, Renault, Mahindra, Tata & Toyota. There are very few luxury brand used cars.



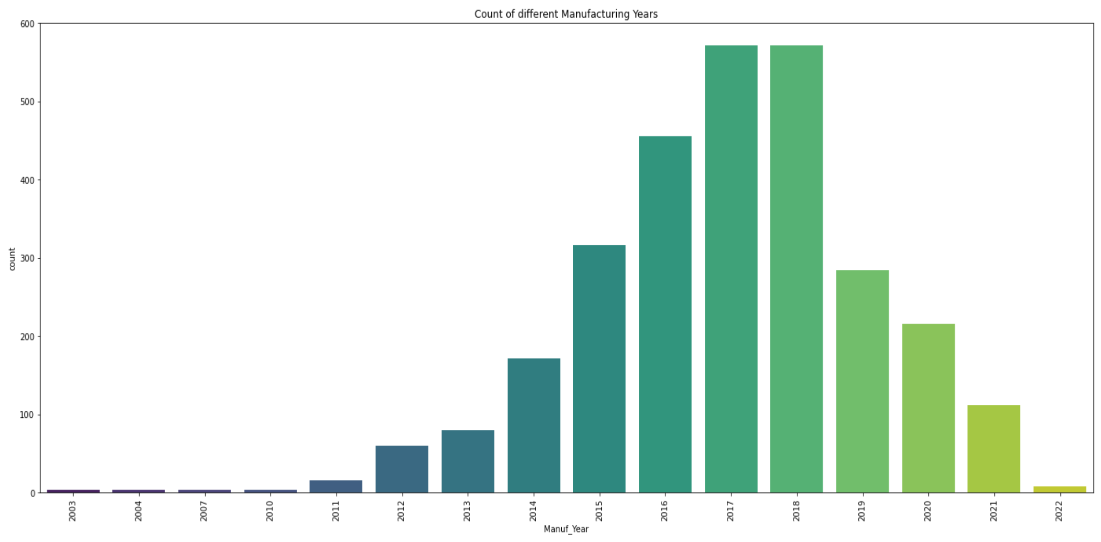
<u>Observation:</u> The major fuel type is Petrol, followed by Diesel and lastly, CNG is the fuel type of the least number of used cars.



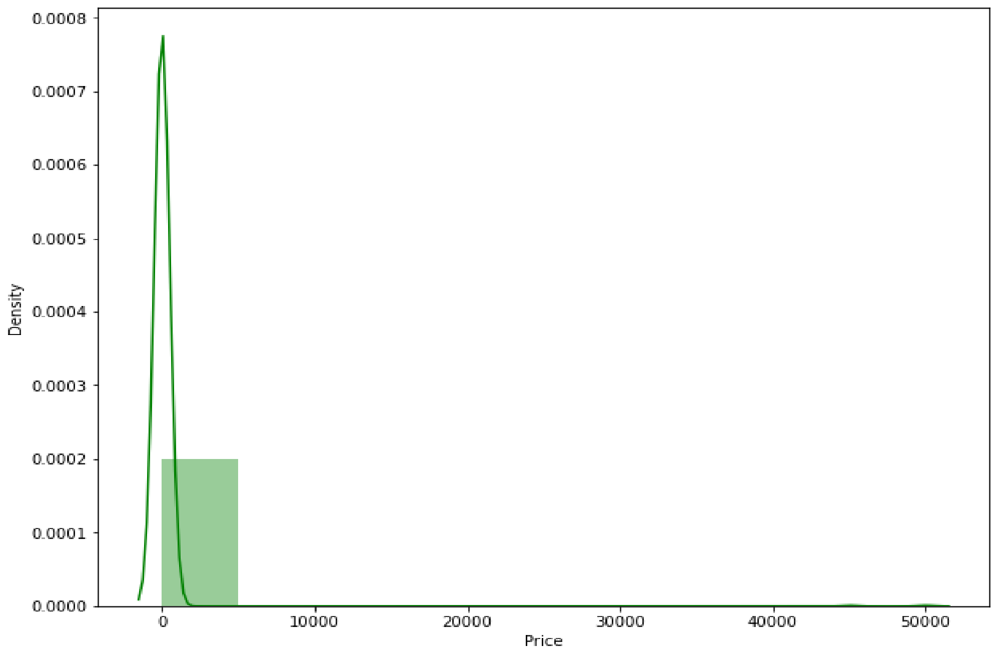
<u>Observation:</u> Most of the Used cars are of manual Variant (80.4 %). The rest are Automatic (19.6%).



Observation: The counts of **VXI, LXI, VXI BS IV, 1.2 Delta & 1.2 Alpha** are highest among all the models.

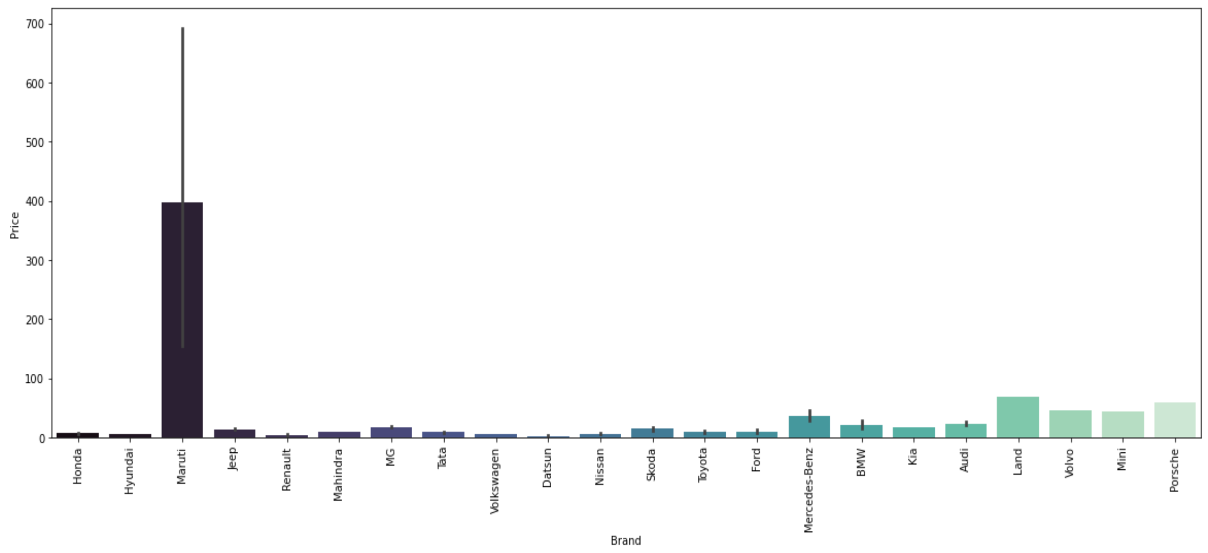


Observation: Most of the Used cars are manufactured in 2017 and 2018. Followed by 2016, 2015 & 2019.



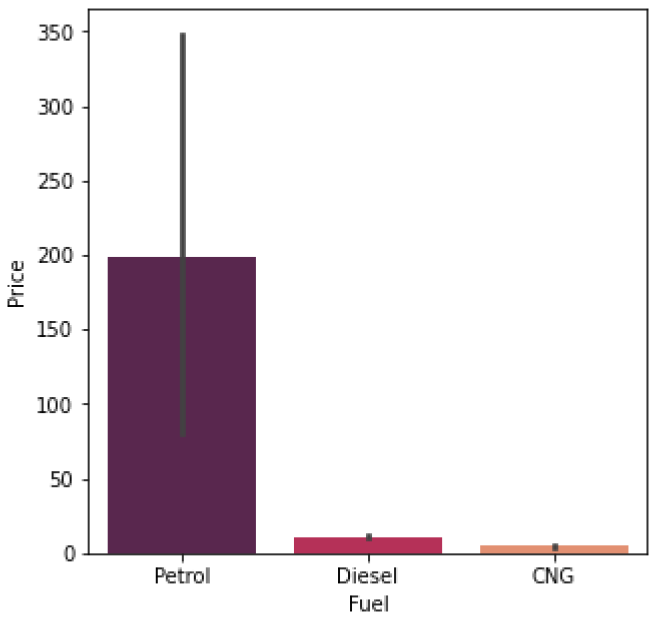
Observation: The distribution of Price is normal but it is skewed to the right.

BIVARIATE ANALYSIS:



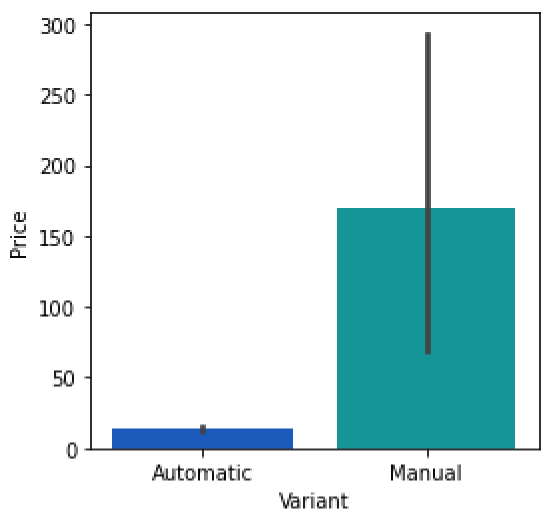
Observation:

Price vs Brand: Most of the used cars are of the Maruti Brand. The Prices of Maruti, Mercedes Benz & Audi are high.



Observation:

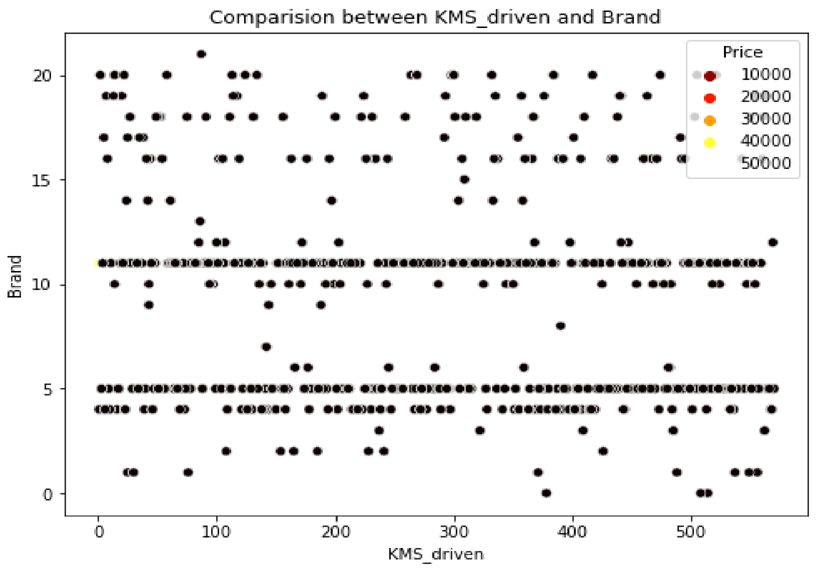
Price vs Fuel: Most of the cars use petrol as their fuel type, they are also the ones with the highest prices. Next, Diesel and CNG are the lowest .



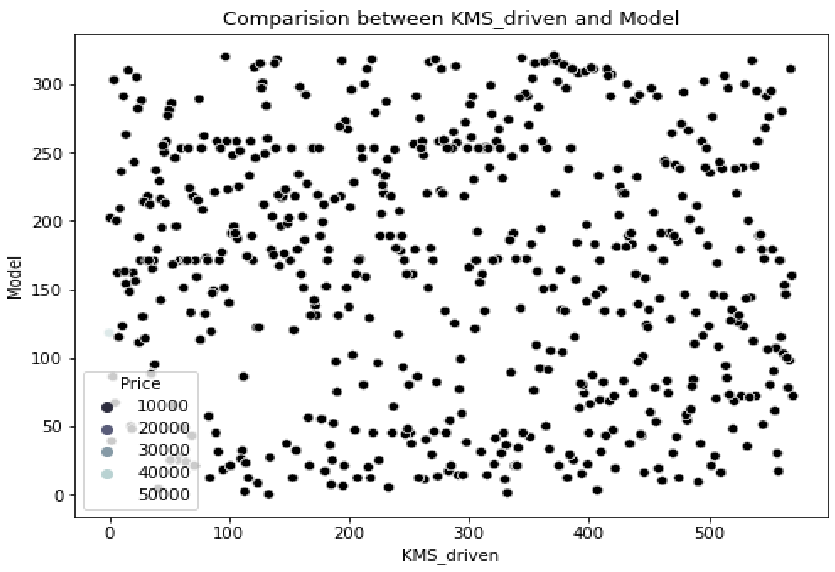
Observation:

Price vs Fuel: Most of the used cars are Manual, they are also the ones with the highest prices.

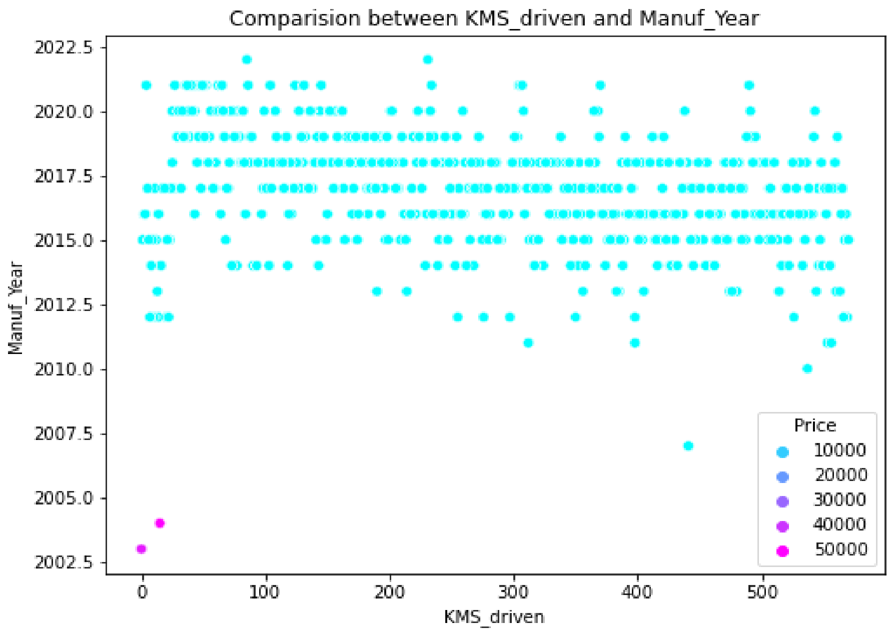
MULTIVARIATE ANALYSIS:



Observation: The above visualization shows the relationship between KMS_driven and Brand in terms of Price. The majority of the cars have kms driven in between 100 & 500. There are a few outliers present in Brand. There is no positive or negative correlation between either Brand or KMS_driven towards price.



Observation: The above visualization shows the relationship between KMS_driven and Model in terms of Price. There is no relation of the two variables with the target as the Price is scattered all around.



Observation: The above visualization shows the relationship between KMS_driven and Manuf_Year in terms of Price. Most of the car's manufacturing years are between 2015 & 2020. There are outliers present in the Manuf_Year column. Manuf_Year is positively correlated with our target Price.

Interpretation of the Results

In univariate analysis I have used count plots and pie plots to visualize the counts in categorical variables and distribution plots to visualize the numerical variables.

In bivariate analysis I have used bar plots, to check the relation between target and the other features. Next I used scatter plots for multivariate analysis & used pair plots to check the pairwise relation between the features.

Detected outliers and skewness with the help of box plots and distribution plots respectively. And I found some of the features skewed to right as well as to left.

The heat map and bar plot helped me to understand the correlation between dependent and independent features. Also, heat maps helped to detect the multicollinearity problem and feature importance.

Then, scaling both train and test dataset has a good impact like it will help the model not to get baised.

Lastly, we have to use multiple regression algorithms while building various models using a train dataset to get the best model out of it. And we have to use multiple metrics like mae, mse, rmse and R2 Score which will help us to decide the best model.

I found ExtraTreesRegressor as the best model with 99.97 % as its R2 Score. Then, I improved the accuracy of the best model by running hyper parameter tuning which changed the R2 Score slightly to 99.97 % .

CONCLUSION

Key Findings and Conclusions of the Study

In this project report, we have used machine learning algorithms to predict the prices of used cars. After the completion of this project, we got an insight of how to collect data, pre-processing of the data, analyzing the data, cleaning the data and building a model.

In this study, we have used multiple machine learning models to predict the sale price of the used cars. We have gone through the data analysis by performing feature engineering, finding the relation between features and target through visualizations. And got the important features and we used these features to predict the car price by building ML models.

After training the model we checked CV score to overcome the overfitting issue. Performed hyper parameter tuning on the best model and the best model's R2 score increased and was giving R2 score as 99.97 %. We have also got good prediction results of car prices.

Learning Outcomes of the Study in respect of Data Science

While working on this project I learned many things about the features of cars and about the various car selling platforms. By building the machine learning models have helped to predict the price of used cars which provides greater understanding into the many advantages & disadvantages of selling old cars.

I found this problem to be quite interesting to handle as it contains all types of data in it and that the data had to be scraped from cardekho.com website using selenium.

The data visualization has helped us in understanding the data by graphical representation. It has made me understand what data is trying to say.

This study is an exploratory attempt to use 8 machine learning algorithms in estimating housing prices, and then compare their results.

Finally, our aim was achieved by predicting the sale price of used cars and building a car price prediction model that could help clients to understand the future price of used cars. New analytical techniques of machine learning can be used in used car price research.

Limitations of this work and Scope for Future Work

LIMITATIONS:

- First drawback is scraping the data as it is a fluctuating process.
- Second was the data not having the correct data type.
- Followed by, more outliers and skewness these two will reduce our model accuracy. We have tried best to deal with outliers & skewness.
- So it looks quite good that we have achieved an accuracy of 99.97 % even after dealing with all these drawbacks.

FUTURE WORK:

- As for future work, we intend to collect more data and to use more advanced techniques like artificial neural networks and genetic algorithms to predict car prices.
- In future this machine learning model may bind with various websites which can provide real time data for price prediction.