Anubhav Maheshwari
2K19/EE/048
Department of Electrical Engineering,
Delhi Technological University

Anuj Majumdar
2K19/EE/050
Department of Electrical Engineering
Delhi Technological University

Used Car Price Prediction Using Machine Learning

Database Management System Innovative Project

1. Abstract

The price of a new car in the automobile industry is fixed keeping in mind the extra cost imposed by the government and the price set by the manufacturer . So, the worthiness of buying a new car can never be questioned. There are several reasons to buy a used car. The high cost of new cars and the lack of economic resources of the customers to buy them, Used Car sales are on a hike. Thus, we need a Used Car Price Prediction system which can efficiently determine the price of the car using machine learning techniques.

In this project, we investigate the application of supervised machine learning techniques to predict the price of used cars. Different techniques like linear regression, random forest regression and decision trees have been used to make the predictions. All these predictions are then evaluated and a comparative analysis is done to know which provides the best performances.

2. Introduction

Used Car price prediction is a very common and useful problem statement. For precise and accurate car price prediction it requires depth and sincere knowledge, because price usually depends on various features and factors. Different features like fuel type, exterior color , year of manufacture , model, architecture, assurance, air condition, interior, navigation and many more will affect the car price. In this paper, we applied different methods and techniques in order to achieve higher precision of the used car price prediction. As we can see, the price depends on a large number of factors. Unfortunately, all the data about all these factors are not always up-to-date and the customer is forced to make the decision to buy it at a certain price based on few factors only.

This is of economic profit to the manufactures and sellers to be able to predict the precise price of used cars with efficiency and accuracy. If the reused car value is fixed lower by the seller in the starting, the installments will be at great rates for the buyer who will certainly then opt for another seller. If the reused car's value is fixed higher than what it should be, the installments will be lower for the clients. Thus, we can see that estimating the price of used cars is of very high commercial importance as well.

This paper is organized as follows. In the next section, a review of related work is provided. Section III describes the related work and limitations while in section IV, we describe research methods. Section V is all about implementation of the working model which describes which machine learning techniques are best fitted to predict the price of used cars. Finally, we end the paper with a conclusion with some pointers towards future work.

3. Related work and limitations

Predicting the price of a used car has been researched extensively in various researches. Listian discussed, in her paper written for Master thesis, that the regression model that was modeled using Support Vector has been assigned with better precision than some simple multiple regression SVM or multivariate regression.

The Second paper is Car Price Prediction Using Machine Learning Techniques. Varieties of attributes are examined for the precise and accurate prediction. To have a model for predicting the price of used cars in Bosnia and Herzegovina, three machine learning techniques (Artificial Neural Network, Random Forest and Support Vector Machine) were used.

Another scope of idea was given by Richardson in his thesis work . His theory states that car manufacturers should manufacture more durable cars. Richardson applied multiple regression analysis and showed that hybrid cars hold their economic strength for a longer time. He was able to achieve a prediction accuracy of 98%.

4. Research Methods

Approach for car price prediction proposed in this paper is composed of several steps. First we got our dataset from Kaggle (Car Dekho). After getting the dataset, we used it in the project which was done in Python using Jupyter Notebook. After importing all the necessary libraries, we started working. Linear Regression and Random Forest Regression is used for predicting the prices of second hand cars. After the comparative analysis of the error and the plots, it was hence proved that Random Forest Regression gives a more accurate answer rather than Linear Regression.

Once the backend part was done, we started working on deploying the website. After training, the file was put as a pickle file(a serialized file that is to be used for deployment) and all the information was dumped into that file. Hence a .pkl file was created.

Then the environment was activated again in a new anaconda prompt and the directory was changed to the project directory. Then the requirements.txt file was created. A frontend was created and a basic web app was made using flask. All this was uploaded on github and heroku was used to host the github repository.

5. Implementation

5.1 Initial setup, refining the dataset

First, an anaconda prompt was opened and a new anaconda environment was created for our particular project rather than using the base environment so that we only use the libraries relevant to our project. Then this environment was activated in the working directory and JupyterNotebook was launched.

First Pandas was imported and it was used to import the dataset. The categorical features were identified from the dataset and the various unique values in them were identified. The null values were checked from the dataset. The ‘car name’ feature, which was irrelevant for the prediction of the car price, was dropped from the dataset.

A new derived feature, representing the number of years the car has been manufactured for, was created. For this, first a feature ‘current year’ was created, then ‘Year’ was subtracted from it to get ‘no_year’. The ‘Year’ and ‘current year’ features were dropped.

Next, the categorical features were converted into numerical data using one-hot encoding. Hence we obtain a final dataset on which we will train our model.

5.2 Finding and plotting among various features

The correlation function was applied to the final dataset and the correlation table was created among all the respective features, which showed the pairwise correlation of all columns in the data frame. To get a better, visual representation of the correlation, the Seaborn library was imported and the pairplot function was applied to the correlation dataset.

To get an even more clear picture of feature correlation, a heatmap was plotted. Again seaborn was used, but this time a new variable was used as the correlation of features. Then the index of this new variable was stored in yet a new variable and the figure size was defined. After that, matplotlib was imported to plot the heatmap in a red-yellow- green cmap styling.

Many conclusions can be drawn from the heatmap easily, about the likeliness of correlation of various parameters by observing the intensity of the color (green - highly positively correlated, red - highly negatively correlated).

5.3 Feature importances

The iloc function was then used to create two variables X and Y, which represent independent and dependent features respectively (as the selling price is dependent on all the other features).

From sklearn.ensemble, ExtraTreesRegressor was imported to check the feature importance. A model was created using this and then it was fit over X, Y. Feature importances were then printed.

To get a visual representation of the feature importances, the Series function from pandas was used on the model's feature importance, for the top 5 largest features.

5.4 Splitting data for training and testing

From sklearn.model_selection, train_test_split was imported. Train test split function was used to split the features into training and testing. Thus the variables X_train, X_test, Y_train, Y_test were created from X and Y. The test size was set to 0.2(20%). X_train.shape was printed to see the correct working of the previous step.

5.5 Training the model

5.5.1 Using Random Forest Regressor

From sklearn.ensemble, RandomForestRegressor was imported. It was used on a new variable 'rf_random'. To avoid underfitting or overfitting, hyperparameter tuning was done on the various parameters. For example, the parameter 'n_estimators', which is basically the decision trees used in our random forest regressor algorithm was tuned.

For this numpy was imported and a list comprehension was first created and the decision trees values were selected from 100 to 1200 using linspace from numpy. Then these values were printed. Similarly, hyperparameter tuning was done for features like max_features', 'max_deapth', 'min_samples_split' and 'min_samples_leaf'. Then, RandomizedSearchCV was imported and it was used for the hyperparameter tuning. Default criterion used is 'mse'.

Next a random grid was created taking these key parameters we had just tuned. A random forest regressor was initialized and a randomized search CV was applied. Here estimator is the random forest just initialized, parameter distributed is the random grid, scoring parameter is negative mean squared error, number of iterations was taken as 10, cross validation taken as 5.

After this model fitting is done on X_train and Y_train; and hence the model is successfully trained.

5.5.2 Using Linear Regression

The model was similarly trained by linear regression, this time importing LinearRegression from sklearn.linear_model and then training the model on 'mse' criterion.

5.6 Doing the Predictions

Predictions are made for both random forest regressor and linear regressor. Dislot and the scatter plot was plotted to make comparisons.

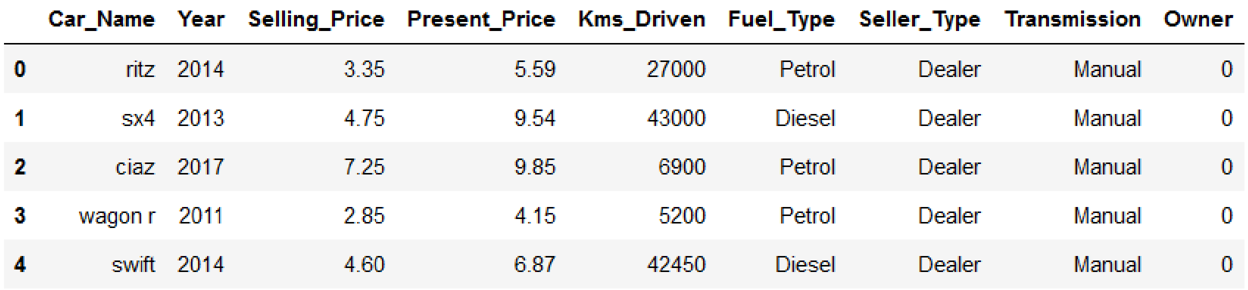
5.7 Deploying it as a webapp

After training, the file was put as a pickle file(a serialized file that is to be used for deployment) and all the information was dumped into that file. Hence a .pkl file was created.

Then the environment was activated again in a new anaconda prompt and the directory was changed to the project directory. Then the requirements.txt file was created. A frontend was created and a basic web app was made using flask. All this was uploaded on github and heroku was used to host the github repository.

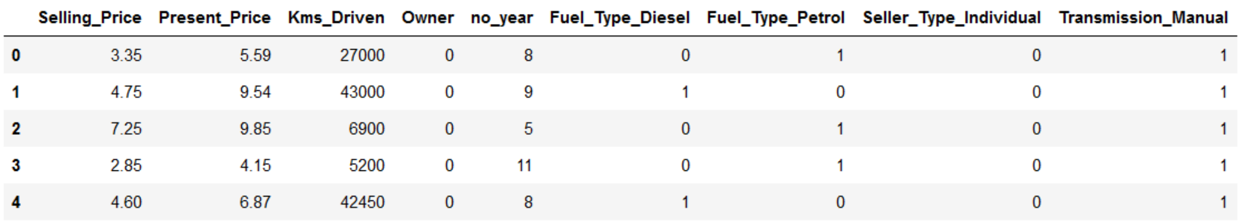
6. Results and Discussions

The dataset was refined from



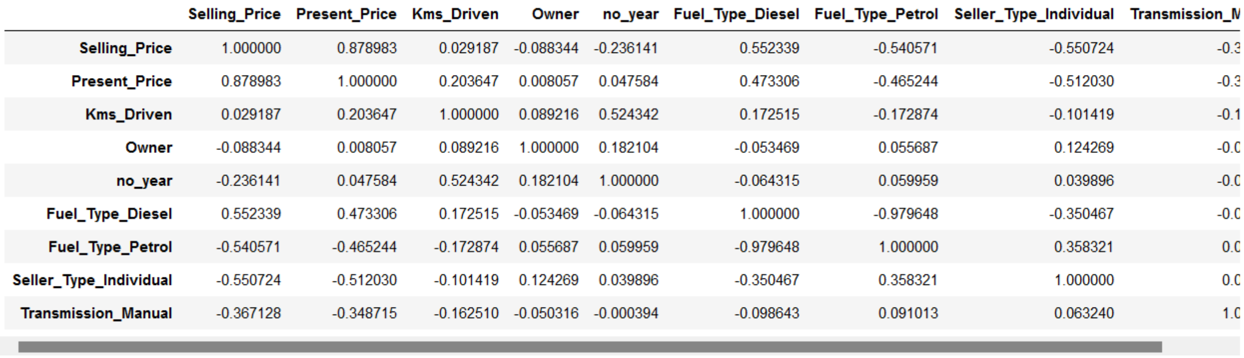
	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner
0	ritz	2014	3.35	5.59	27000	Petrol	Dealer	Manual	0
1	sx4	2013	4.75	9.54	43000	Diesel	Dealer	Manual	0
2	ciaz	2017	7.25	9.85	6900	Petrol	Dealer	Manual	0
3	wagon r	2011	2.85	4.15	5200	Petrol	Dealer	Manual	0
4	swift	2014	4.60	6.87	42450	Diesel	Dealer	Manual	0

To



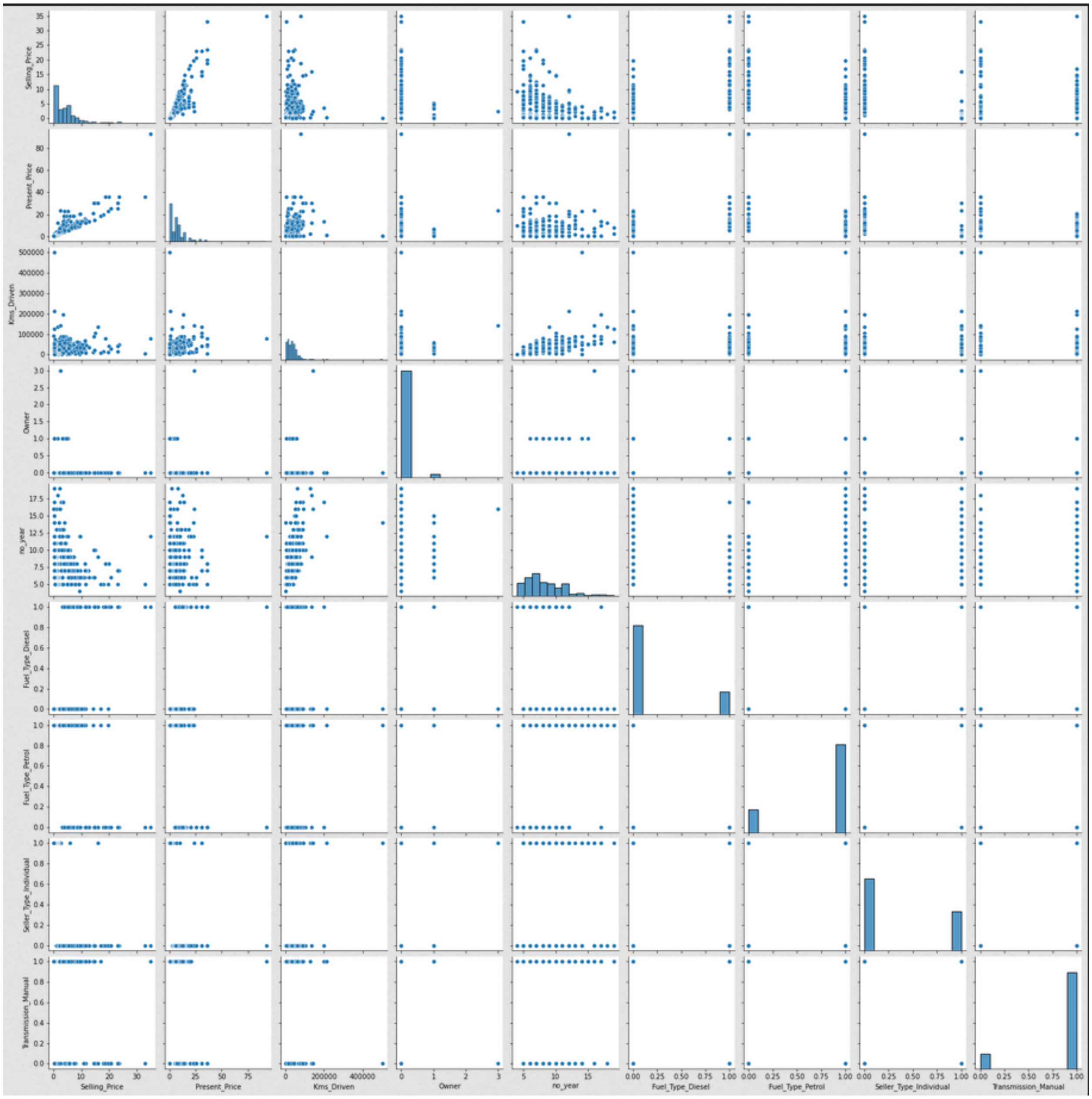
	Selling_Price	Present_Price	Kms_Driven	Owner	no_year	Fuel_Type_Diesel	Fuel_Type_Petrol	Seller_Type_Individual	Transmission_Manual
0	3.35	5.59	27000	0	8	0	1	0	1
1	4.75	9.54	43000	0	9	1	0	0	1
2	7.25	9.85	6900	0	5	0	1	0	1
3	2.85	4.15	5200	0	11	0	1	0	1
4	4.60	6.87	42450	0	8	1	0	0	1

The obtained correlation table was

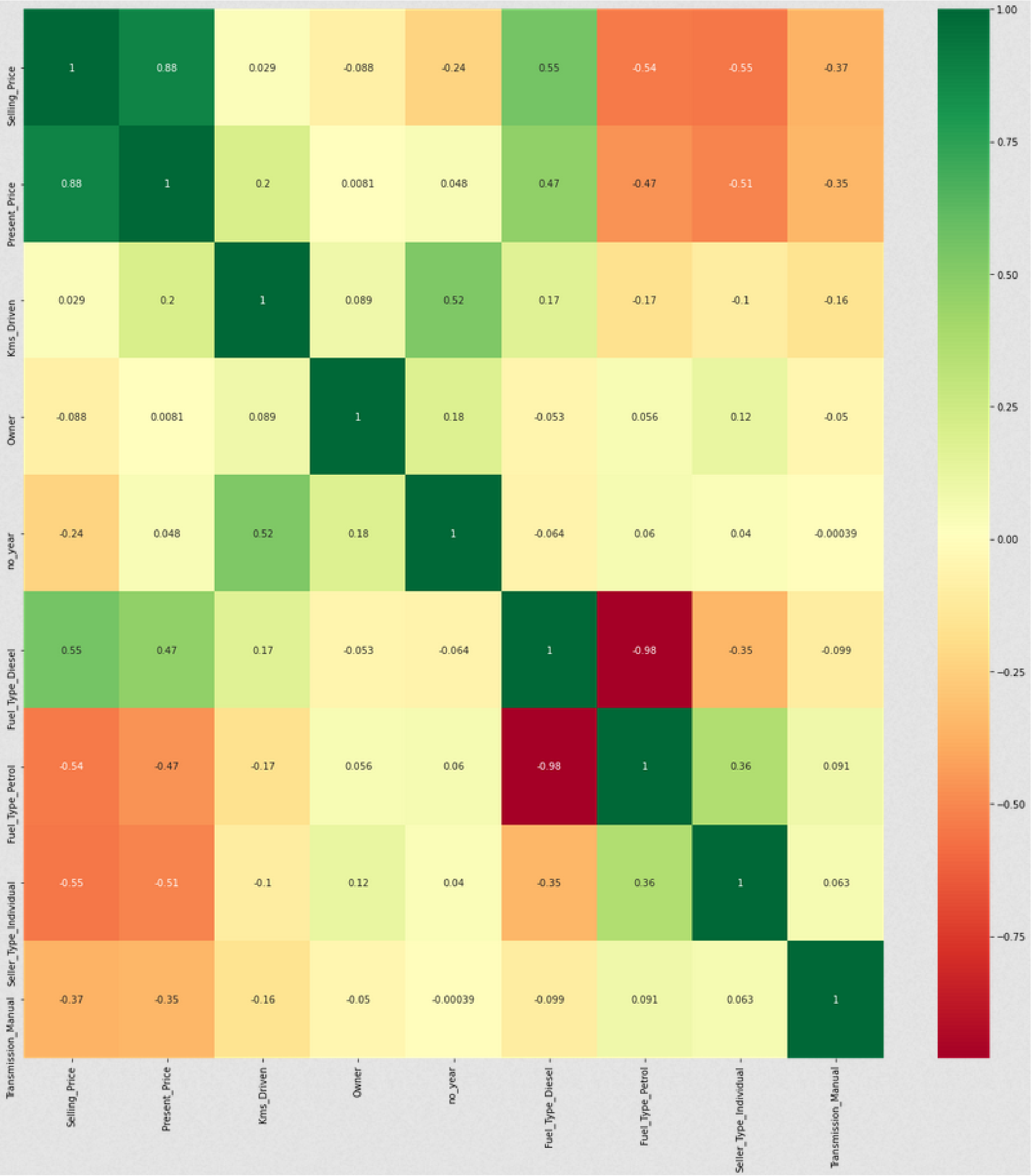


	Selling_Price	Present_Price	Kms_Driven	Owner	no_year	Fuel_Type_Diesel	Fuel_Type_Petrol	Seller_Type_Individual	Transmission_M
Selling_Price	1.000000	0.878983	0.029187	-0.088344	-0.236141	0.552339	-0.540571	-0.550724	-0.3
Present_Price	0.878983	1.000000	0.203647	0.008057	0.047584	0.473306	-0.465244	-0.512030	-0.3
Kms_Driven	0.029187	0.203647	1.000000	0.089216	0.524342	0.172515	-0.172874	-0.101419	-0.1
Owner	-0.088344	0.008057	0.089216	1.000000	0.182104	-0.053469	0.055687	0.124269	-0.0
no_year	-0.236141	0.047584	0.524342	0.182104	1.000000	-0.064315	0.059959	0.039896	-0.0
Fuel_Type_Diesel	0.552339	0.473306	0.172515	-0.053469	-0.064315	1.000000	-0.979648	-0.350467	-0.0
Fuel_Type_Petrol	-0.540571	-0.465244	-0.172874	0.055687	0.059959	-0.979648	1.000000	0.358321	0.0
Seller_Type_Individual	-0.550724	-0.512030	-0.101419	0.124269	0.039896	-0.350467	0.358321	1.000000	0.0
Transmission_Manual	-0.367128	-0.348715	-0.162510	-0.050316	-0.000394	-0.098643	0.091013	0.063240	1.0

The corresponding pair plot obtained was

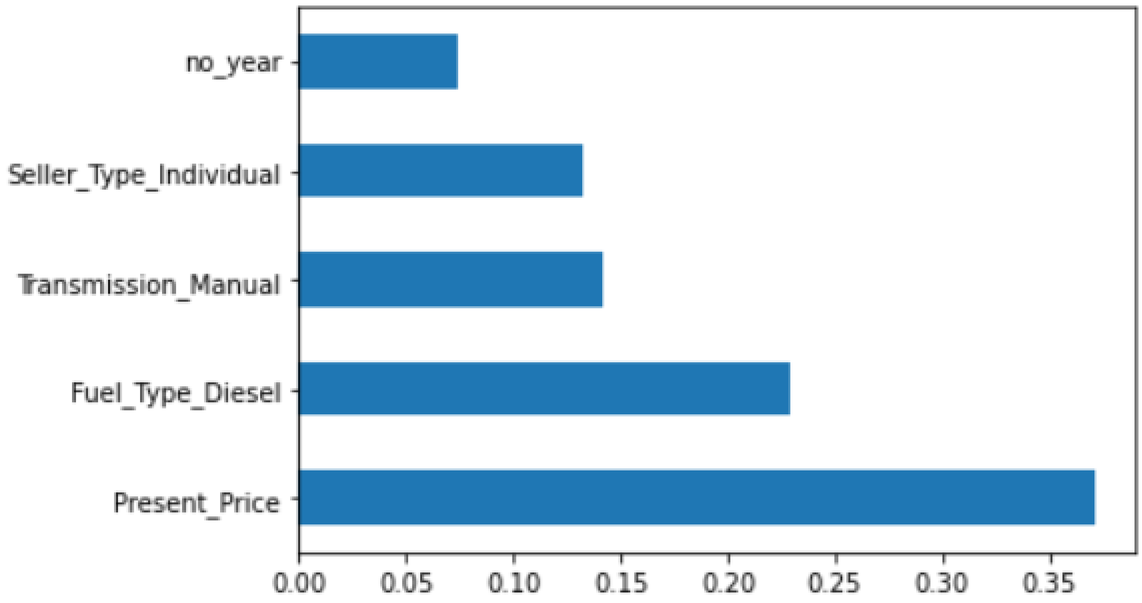


And the heatmap obtained was

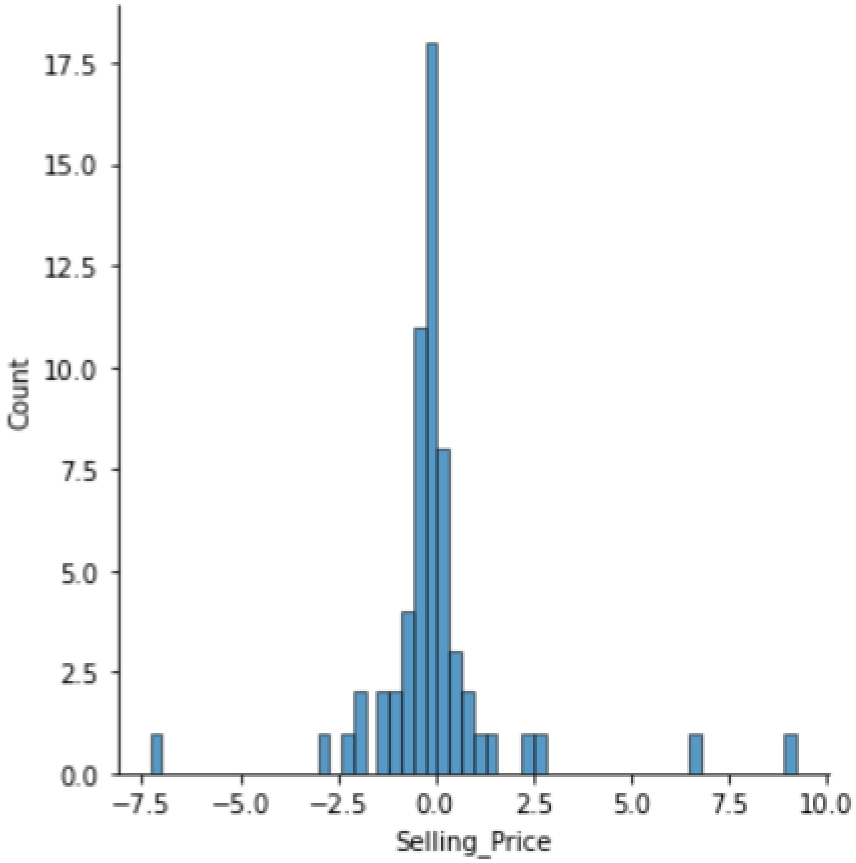


	Selling_Price	Present_Price	Kms_Driven	Owner	no_year	Fuel_Type_Diesel	Fuel_Type_Petrol	Seller_Type_Individual	Transmission_Manual
Selling_Price	1	0.88	0.029	-0.088	-0.24	0.55	-0.54	-0.55	-0.37
Present_Price	0.88	1	0.2	0.0081	0.048	0.47	-0.47	-0.51	-0.35
Kms_Driven	0.029	0.2	1	0.089	0.52	0.17	-0.17	-0.1	-0.16
Owner	-0.088	0.0081	0.089	1	0.18	-0.053	0.056	0.12	-0.05
no_year	-0.24	0.048	0.52	0.18	1	-0.064	0.06	0.04	-0.00039
Fuel_Type_Diesel	0.55	0.47	0.17	-0.053	-0.064	1	-0.98	-0.35	-0.099
Fuel_Type_Petrol	-0.54	-0.47	-0.17	0.056	0.06	-0.98	1	0.36	0.091
Seller_Type_Individual	-0.55	-0.51	-0.1	0.12	0.04	-0.35	0.36	1	0.063
Transmission_Manual	-0.37	-0.35	-0.16	-0.05	-0.00039	-0.099	0.091	0.063	1

Top 5 feature importances obtained by series function were



Displot for Random Forest Regression was obtained as:



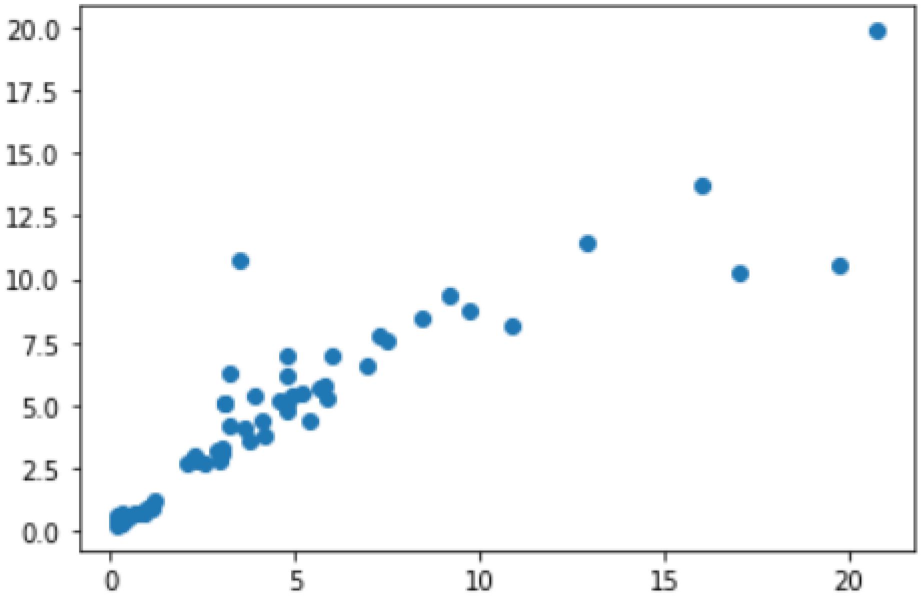
Errors for Random Forest Regression:

MAE: 0.9402548243559634

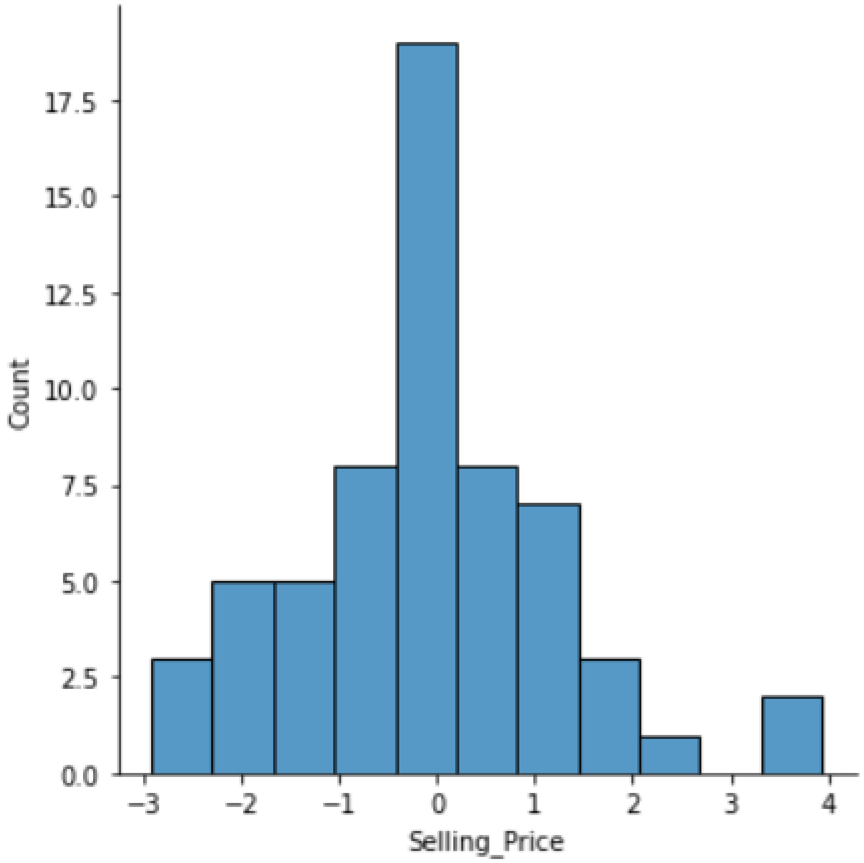
MSE: 3.826857163362346

RMSE: 1.9562354570353606

Scatter plot for Random Forest Regressor was obtained as:



Displot for Linear Regression was obtained as:



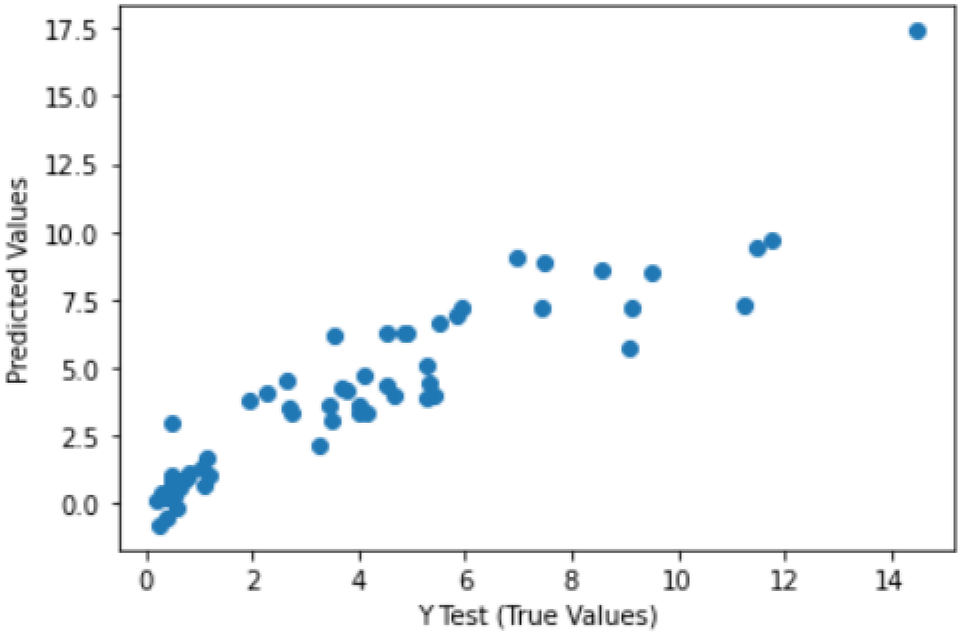
Errors for Linear Regression:

MAE: 0.9506014820849296

MSE: 1.6992388405952439

RMSE: 1.3035485570531096

Scatter plot for Linear Regressor was obtained as:



For Random Forest Regressor we got a better normal distribution (more close gaussian distribution graph), better scatter plot (less scattering from a straight line) and less errors (mean absolute error, mean squared error, root mean squared error) as compared to Linear Regression. Hence we can conclude that it is better suited for training this particular model.

7. Future Scope

As in this dataset, most of the data were of the cars manufactured after the year 2008. So if we use the data of cars which are quite old and manufactured before 2008, the result might not be very precise and accurate. More work needs to be done in training the data more accurately and data cleaning needs to be done with more precision to get more accurate results.

8. Conclusion

This model was based on the machine learning algorithms and we predicted the selling price of the used cars based on the dataset provided at Kaggle. To predict the selling price value, two machine learning algorithms i.e. Random Forest and Linear Regression were used. The prediction of this model was further compared with the test dataset created by picking random values from the original dataset and the evaluation of the prediction is further evaluated using different methods. After a complete evaluation of the predictive model, it was concluded that the accuracy of the Random forest regression model is very high and Random Forest is one of the best algorithms for regression problems. A comparative difference in between these two algorithms was clearly observed on the basis of accuracy and prediction speed irrespective of the size of the dataset. Finally the platform was hosted as a webapp on heroku.

9. References

1. Dataset- https://www.kaggle.com/nehalbirla/vehicle-dataset-from-cardekho

2. Pandas- https://realpython.com/pandas-python-explore-dataset/

3. Numpy- https://realpython.com/numpy-tutorial/

4. Matplotlib- https://realpython.com/python-matplotlib-guide/

5. Seaborn- https://realpython.com/lessons/plotting-seaborn/

6. Random Forest Regression- https://towardsdatascience.com/an-implementation-and-explanation-of-the-random-forest-in-python-77bf308a9b76

7. Linear Regression- https://realpython.com/linear-regression-in-python/

8. Krish Naik's Youtube Channel- https://www.youtube.com/channel/UCNU_lfiiWBdtULKOw6X0Dig

9. Stanford CS229: Machine Learning Autumn 2018- https://www.youtube.com/playlist?list=PLoROMvodv4rMiGQp3WXShtMGgzqpfVfbU

10. StatQuest with Josh Starmer Youtube Channel- https://www.youtube.com/c/joshstarmer