USED-CAR PRICE PREDICTION

N GOBINATH, G PHANI YESHWANTH, S NARENDRA KUMAR, V VINAY KUMAR, T SURENDRA

School of Electronics and Communication Engineering, Vellore Institute of Technology, Chennai

Keywords

Decision Tree
Random Forest
Extra tree Regressor
Bootstrap
Bagging

Abstract:

The focus of this project is to develop machine learning models that can accurately predict the price of a used car based on its features. In this project, we investigate supervised machine learning models to predict the price of used cars in India. The predictions are based on historical data of car details, taken from CarDekho.com. Different models like Random Forest, Extra Tree Regressor, Bagging Regressor and Decision Tree have been used to make the predictions. The predictions are then compared in order to find those which provide the best performances. All four methods provided comparable performance. The number of different attributes is measured, and also it has been considered to predict a more reliable and accurate result. We also compared the prediction accuracy of these models to determine the best one. Our results show that the Random Forest model yields the best result.

CHAPTER I

1.Introduction

1.1 Background and motivation

Predicting the price of used cars is an important thing. In recent years, people interested in buying used cars have increased. Moreover, the awareness about cars has increased, and people are using such predictors to take a more educated decision. Therefore, there is a need for efficient prediction techniques. Accurate car price prediction usually depends on many distinctive features and factors. Typically, the most significant ones are brand and model, age, horsepower and mileage. The fuel type used in the car and fuel consumption per mile highly affect the price of a car due to frequent fuel price changes. Different features like exterior colour, dimensions, safety, air condition, interior, and navigation will also influence the car price. From the perspective of a seller, it is also a dilemma to price a used car accurately. Based on existing data, the aim is to use machine learning models to develop models for predicting used car prices. By training statistical models for predicting the costs, one can quickly get a rough estimate of the worth without actually entering the small print.

1.2 Problem statement and objectives

The manufacturer fixes the prices of new cars in the industry with some additional costs incurred by the Government in taxes. However, due to the increased price of new cars and the incapability of customers to buy new cars due to the lack of funds, used cars sales are on a global increase. There is a need for a used car resale price prediction to determine the car’s worth effectively. The r egression model is mainly influenced by three factors, i.e. algorithm, number of explanatory variables, and number of samples. It is essential to understand their actual market price during both buying and selling.

The main objective of this paper is to us e three different prediction models to predict the retail price of a used car and compare their levels of accuracy.

CHAPTER II

2.1 PROPOSED/IMPLEMENT METHOD

In this project, we analyzed the data present in the data set by doing exploratory analysis on it. Later we applied different machine learning models in order to achieve higher precision of used car price prediction. We separated the data set into dependent and independent attributes. Dependent attribute is the selling price and rest are independent attributes. We found the correlation of each attribute with selling price to and found out which of them was affecting the re-sale price the most. Later we applied four different classification algorithms in order to find the accuracy of our project. The following are the essential elements of our project.

Correlation:

We calculated Correlation of each attribute with every other attribute to get insight into how the attributes were depended on each other and to identify the attributes which affect the prediction.

Algorithms used in order to find the accuracy:

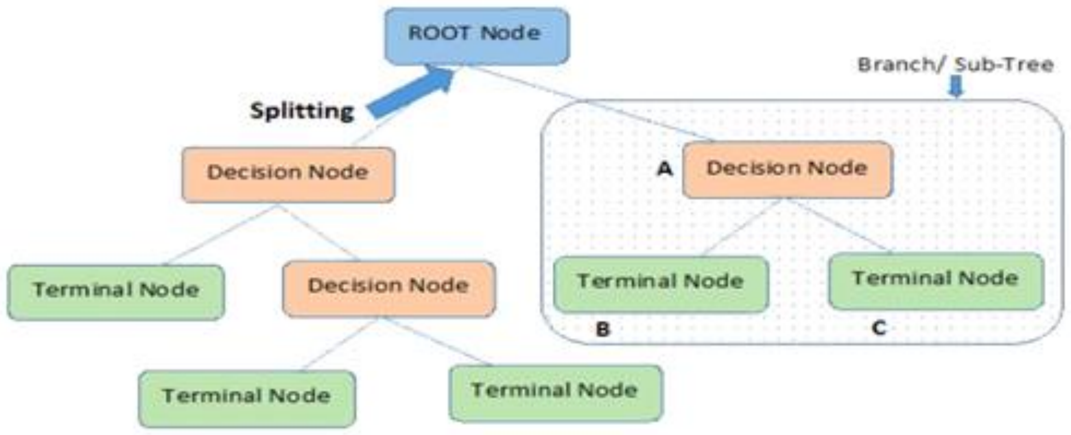
We used four classification algorithms in this project. All of them very similar in approach but with slight variations. Our aim was to find the algorithm that fits best with our dataset and to give the best possible prediction we could. The algorithms we used are briefly introduced and explained in the following segment.

1. Decision Tree

Classification may be a two-step process, learning step and prediction step, in machine learning. In the learning step, the model is developed supported given training data. In the prediction step, the model is employed to predict the response for given data.
Decision Tree is one among the simplest and popular classification algorithms to know and interpret.

The goal of employing a Decision Tree is to make a training model which will use to predict the category or value of the target variable by learning simple decision rules inferred from prior data (training data).

In Decision Trees, for predicting a category label for a record we start from the basis of the tree. We compare the values of the root attribute with the record’s attribute. On the idea of comparison, we follow the branch like that value and jump to subsequent node.



Attribute Selection Measures

If the dataset consists of N attributes then deciding which attribute to put at the basis or at different levels of the tree as internal nodes may be a complicated step. By just randomly selecting any node to be the basis can't solve the difficulty. If we follow a random approach, it's going to give us bad results with low accuracy. For solving this attribute selection problem, researchers worked and devised some solutions. They suggested using some criteria like:

Entropy,

Information gain,

Gini index,

Gain Ratio,

Reduction in Variance

Chi-Square

These criteria will calculate values for every attribute. The values are sorted, and attributes are placed within the tree by following the order i.e, the attribute with a high value (in case of data gain) is placed at the root.
While using Information Gain as a criterion, we assume attributes to be categorical, and for the Gini index, attributes are assumed to be continuous. In our project also it will calculate the root node and get the resultant output by following the algorithm given up.

2.Random Forest

Random forest (RF) also known as random decision forest belongs to the category of ensemble methods. One of the most important advantages of random forest is its versatility. It is often used for both regression and classification tasks, and it's also easy to look at the relative importance it assigns to the input features.
Random Forest is additionally a really handy algorithm because the default hyper parameters it uses often produce an honest prediction result. Understanding the hyper parameters is pretty straightforward, and there is also not that a lot of them.
One of the biggest problems in machine learning is over fitting, but most of the time this won't happen thanks to the random forest classifier. If there are enough trees within the forest, the classifier won't overfit the model.

Working:
Random forest builds multiple decision trees and merges them together to urge a more accurate and stable prediction.

One big advantage of random forest is that it are often used for both classification and regression problems, which form the bulk of current machine learning systems. Let's check out random forest in classification, since classification is usually considered the building block of machine learning.

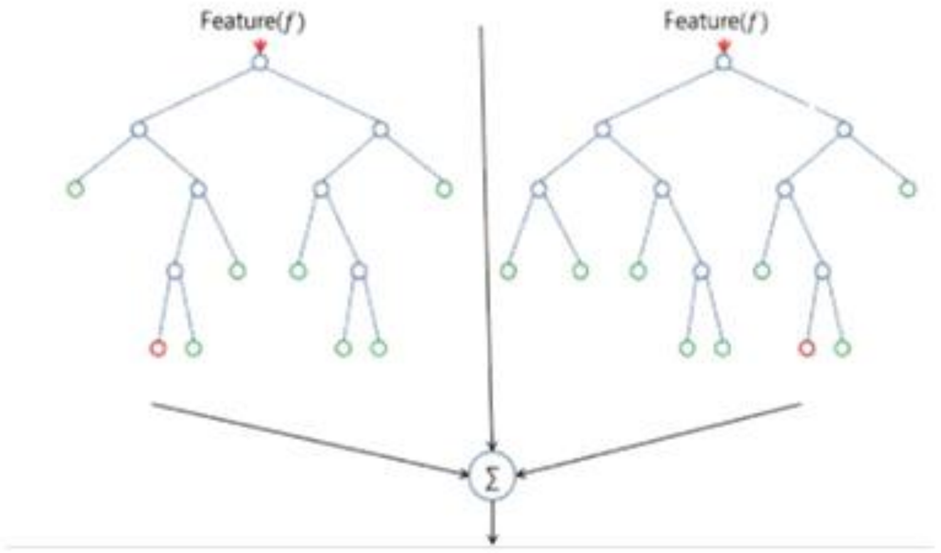


Fig. Random Forest Representation

Random forest has nearly an equivalent hyper parameter as a decision tree or a bagging classifier. Fortunately, there is no got to combine a choice tree with a bagging classifier because you'll easily use the classifier -class of random forest. With random forest, you'll also affect regression tasks by using the algorithm's regressor.

3. Extra Tree Regressor

This class implements a meta estimator that matches variety of randomized decision trees on various sub-samples of the dataset and uses averaging to enhance the predictive accuracy and control over-fitting. The number of trees in the forest.

4.Bagging Regressor

Algorithm:
An ensemble method may be a technique that mixes the predictions from multiple machine learning algorithms together to form more accurate predictions than a person model.

Bootstrap Aggregation may be a general procedure which will be wont to reduce the variance for those algorithms that have high variance. An algorithm that has high variance are decision trees, like classification and regression trees (CART).

Decision trees are sensitive to the precise data on which they're trained. If the training data is changed (e.g. a tree is trained on a subset of the training data) the resulting decision tree are often quite different and successively the predictions are often quite different.

Bagging is that the application of the Bootstrap procedure to a high-variance machine learning algorithm, typically decision trees. When bagging with decision trees, we are less concerned about individual trees overfitting the training data. For this reason and for efficiency, the individual decision trees are grown deep (e.g., few training samples at each leaf-node of the tree) and therefore the trees aren't pruned. These trees will have both high variance and low bias. These are important characterize of sub -models when combining predictions using bagging. The only parameters when bagging decision trees is that the number of samples and hence the number of trees to incorporate . This can be chosen by increasing the number of trees on run after run until the accuracy begins to prevent showing improvement (e.g. on a cross validation test harness). Very large numbers of models may take an extended time to organize , but won't overfit the training data.

2.2 MERITS

The main aim of our project is accuracy. By applying different machine learning algorithms, we found the best on that works for this dataset. Moreover, our project needs very few attributes of a car to make a fairly accurate prediction, which is not the case with most of the existing algorithms.

2.3 CHALLENGES FACED

Finding the proper data set.

This is the most important challenge we faced in our project. For analyzing the data we need an appropriate data set which has all the parameters that we required and give more accurate results. We get our dataset from Kaggle which includes the all parameters that we required.

Finding the suitable algorithm to predict the accuracy of our project.

The main objective of our project is prediction accuracy. And many algorithms that we tried had their drawbacks either in accuracy or in prediction speed and efficiency.

2.4 CODE

```
#Import all required libraries
import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

import numpy as np
```

```
from sklearn.tree import DecisionTreeRegressor from sklearn.ensemble import
ExtraTreesRegressor from sklearn.model_selection import train_test_split from
sklearn.ensemble import RandomForestRegressor from sklearn import metrics
from sklearn.metrics import r2_score
from sklearn.ensemble import BaggingRegressor


df=pd.read_csv('car data.csv') #Read the dataset
df.shape            #Returns the dimensions of the dataset
df.columns #Returns all the feature columns
print(df['Fuel_Type'].unique() , df['Seller_Type'].unique() ,
  df['Owner'].unique() ,                          df['Transmission'].unique()) #Returns
distinct values of those specific features
df.isnull().sum()#Returns the count of null values in the
dataset
df.describe() #Returns useful metrics that describes the
dataset
final_dataset=df[['Year','Selling_Price','Present_Price','Kms_
Driven','Fuel_Type','Seller_Type','Transmission','Owner']]
final_dataset.head() #Car name feature column has been deleted
as it has no importance in the prediction
final_dataset['Current Year']=2021 #New current year column is
inserted to visualize the number of years passed from
manufactured date
final_dataset.head()#To visualize the top few observations
final_dataset['no_year']=final_dataset['Current Year']- final_ dataset['Year']#Returns number of years
passed from the year of manufacturing
final_dataset.head()#To visualize the top few observations
final_dataset.drop(['Year'],axis=1,inplace=True)#Year column
is dropped as it has no importance in prediction
final_dataset.head()#To visualize the top few observations
final_dataset=final_dataset.drop(['Current Year'],axis=1)
)#Current Year column is dropped as it has no importance in
prediction
```

final_dataset.head()#To visualize the top few observations

final_dataset=pd.get_dummies(final_dataset,drop_first=True)#To make every unique value as a feature using one hot encoding, drops the original variables.

final_dataset.head()#To visualize the top few observations

final_dataset.corr()#Return the pairwise correlation of all the features

X=final_dataset.iloc[:,1:]#Independent features

X.head()#To visualize the top few observations

y=final_dataset.iloc[:,0]#Target variable

y.head()#To visualize the top few observations

model = ExtraTreesRegressor()#Calling Extra tree Regressor class

model.fit(X,y)#Fitting the dataset into the model model.feature_importances_#Returns an array with feature importance values of each feature

feat_importances = pd.Series(model.feature_importances_, index =X.columns) #Convert feature importance values into 1D array for visualization

feat_importances.nlargest(5).plot(kind='barh') #To Plot a bar graph with five top most features by their values

g=sns.heatmap(final_dataset[top_corr_features].corr(),annot=Tr ue,cmap="RdYlGn")#To plot the heatmap of correlation values of all the features

sns.pairplot(final_dataset)#To obtain the pairplot for better understanding of the dataset

X_train, X_test, y_train, y_test = train_test_split(X, y, test _size=0.2, random_state=0)#To split the dataset into training and testing

X_train.shape #Returns the dimensions of the "X" array

#Decision Tree Regressor

decision_tree = DecisionTreeRegressor() #Calling Decision Tree class

decision_tree.fit(X_train, y_train) #Fitting the dataset into the model

predictions_decisiontree=decision_tree.predict(X_test)

#Predicts the selling price

print(predictions_decisiontree) #Prints the predictions of

```
selling price
print(y_test)
print('MAE:', metrics.mean_absolute_error(y_test, predictions_ decisiontree)) #Returns mean absolute
error
print('MSE:', metrics.mean_squared_error(y_test, predictions_d ecisiontree)) #Returns mean square error
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, pred ictions_decisiontree))) #Returns root
mean square error
r2 = r2_score(y_test,predictions_decisiontree) #To find the R-squared value
print('r2 score for the model is', r2) #Prints the R-squared value
errors = abs(predictions_decisiontree - y_test) #Residual error
mape = np.mean(100 * (errors / y_test)) #Mean absolute percentage error
accuracy_3= 100 - mape #Accuracy of the model
print('Accuracy:',accuracy_3, '%') #Prints the accuracy
#Random Forest Regressor
rf_random=RandomForestRegressor() #Calling
RandomForestRegressor class
rf_random.fit(X_train,y_train) #Fitting the dataset into the
model
predictions=rf_random.predict(X_test) #Predicts the selling price
print(predictions) #Prints the predictions of selling price
print(y_test)
print('MAE:',metrics.mean_absolute_error(y_test, predictions))
#Returns mean absolute error
print('MSE:',metrics.mean_squared_error(y_test, predictions))#
Returns          mean square error
print('RMSE:',np.sqrt(metrics.mean_squared_error(y_test, predi
ctions))) #Returns root mean square error

r2 = r2_score(y_test,predictions) #To find the R-squared value
print('r2 score for the model is', r2) #Prints the R-squared value
errors = abs(predictions - y_test) #Residual error mape = np.mean(100 * (errors / y_test))
#Mean absolute percentage error
accuracy_2= 100 - mape                    #Accuracy of the model
print('Accuracy:',accuracy_2, '%') #Prints the accuracy
#Extra Trees Regressor
```

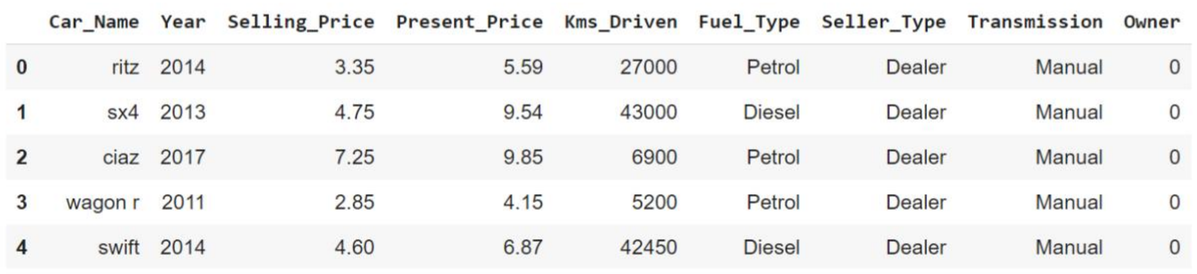
```
model = ExtraTreesRegressor() #Calling ExtraTreeRegressor class
model.fit(X_train,y_train) #Fitting the dataset into the model predictions_model=model.predict(X_test)
#Predicts the selling price
print(predictions_model) #Prints the predictions of selling price
print(y_test)
print('MAE:', metrics.mean_absolute_error(y_test, predictions_ model)) #Returns mean absolute error
print('MSE:', metrics.mean_squared_error(y_test, predictions_m odel)) #Returns mean square error
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, pred ictions_model))) #Returns root mean
square error

r2 = r2_score(y_test,predictions_model) #To find the R-squared value
print('r2 score for the model is', r2) #Prints the R-squared value
errors = abs(predictions_model - y_test) #Residual error mape = np.mean(100 * (errors /
y_test)) #Mean absolute percentage error
accuracy_2= 100 - mape #Accuracy of the model
print('Accuracy:',accuracy_2, '%') #Prints the accuracy
#Bagging Regressor
bagging = BaggingRegressor() #Calling ExtraTreeRegressor class bagging.fit(X_train, y_train) #Fitting
the dataset into the model
predictions_bagging=bagging.predict(X_test) #Predicts the selling price
print(predictions_bagging) #Prints the predictions of selling price print(y_test)
print('MAE:', metrics.mean_absolute_error(y_test, predictions_ bagging)) #Returns mean absolute error
print('MSE:', metrics.mean_squared_error(y_test, predictions_b agging)) #Returns mean square error
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, pred ictions_bagging))) #Returns root mean
square error

r2 = r2_score(y_test,predictions_bagging) #To find the R-squared value
print('r2 score for the is', r2) #Prints the R-squared value errors = abs(predictions_bagging - y_test)
#Residual error mape = np.mean(100 * (errors / y_test)) #Mean absolute percentage error
accuracy_4= 100 - mape                    #Accuracy of the model
print('Accuracy:',accuracy_4, '%') #Prints the accuracy
```

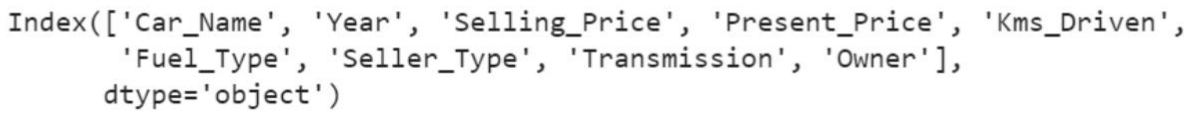
CHAPTER III

3.1 MAIN RESULTS



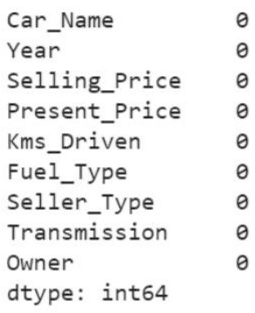
	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner
0	ritz	2014	3.35	5.59	27000	Petrol	Dealer	Manual	0
1	sx4	2013	4.75	9.54	43000	Diesel	Dealer	Manual	0
2	ciaz	2017	7.25	9.85	6900	Petrol	Dealer	Manual	0
3	wagon r	2011	2.85	4.15	5200	Petrol	Dealer	Manual	0
4	swift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

Fig 1: Car dataset



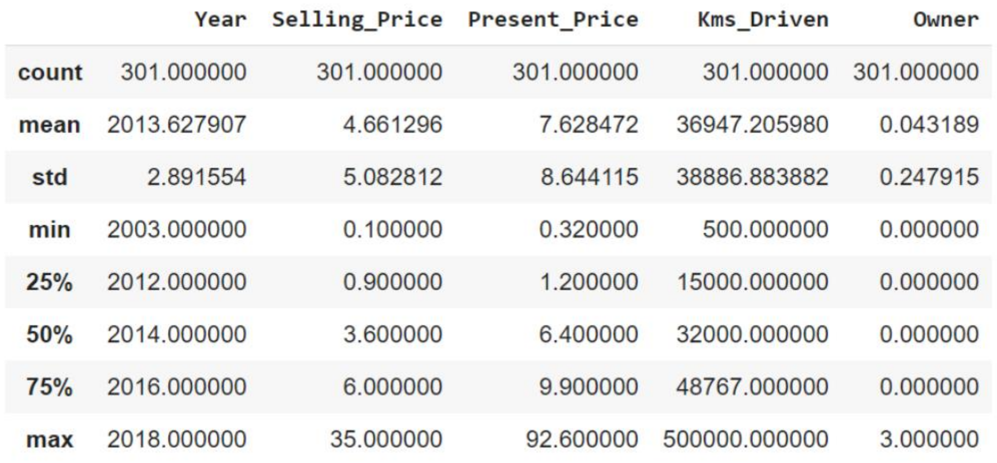
```
Index(['Car_Name', 'Year', 'Selling_Price', 'Present_Price', 'Kms_Driven',
       'Fuel_Type', 'Seller_Type', 'Transmission', 'Owner'],
      dtype='object')
```

Fig 2: Feature variables



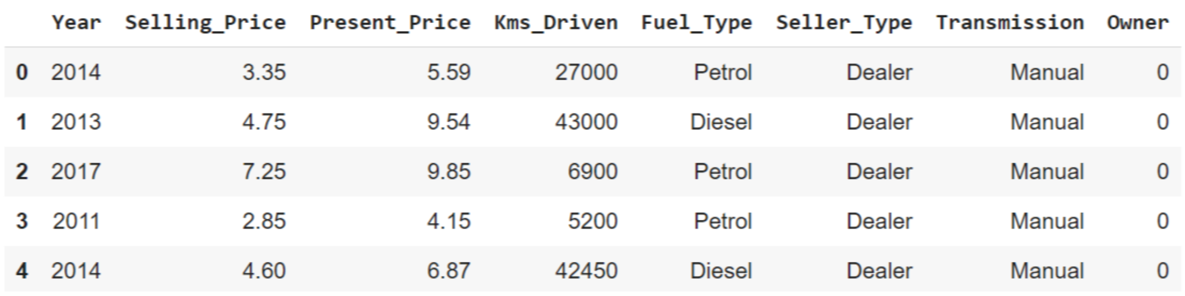
```
Car_Name         0
Year             0
Selling_Price    0
Present_Price    0
Kms_Driven       0
Fuel_Type        0
Seller_Type      0
Transmission     0
Owner            0
dtype: int64
```

Fig 3: Count of null values



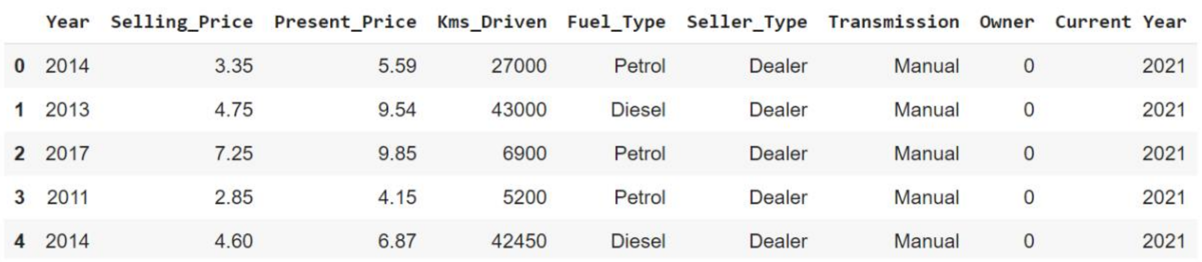
	Year	Selling_Price	Present_Price	Kms_Driven	Owner
count	301.000000	301.000000	301.000000	301.000000	301.000000
mean	2013.627907	4.661296	7.628472	36947.205980	0.043189
std	2.891554	5.082812	8.644115	38886.883882	0.247915
min	2003.000000	0.100000	0.320000	500.000000	0.000000
25%	2012.000000	0.900000	1.200000	15000.000000	0.000000
50%	2014.000000	3.600000	6.400000	32000.000000	0.000000
75%	2016.000000	6.000000	9.900000	48767.000000	0.000000
max	2018.000000	35.000000	92.600000	500000.000000	3.000000

Fig 4: Summarizing the dataset with various metrics



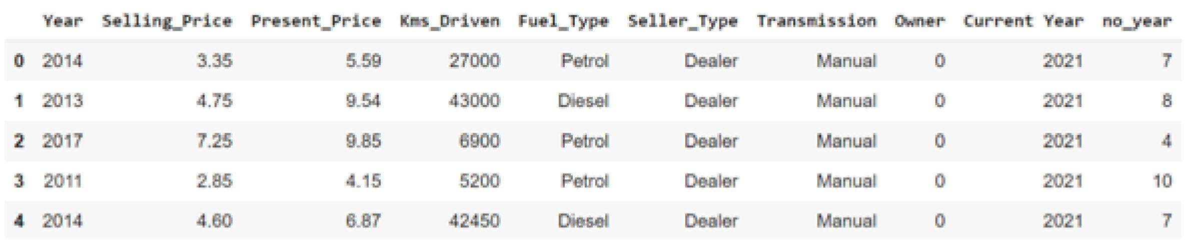
	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner
0	2014	3.35	5.59	27000	Petrol	Dealer	Manual	0
1	2013	4.75	9.54	43000	Diesel	Dealer	Manual	0
2	2017	7.25	9.85	6900	Petrol	Dealer	Manual	0
3	2011	2.85	4.15	5200	Petrol	Dealer	Manual	0
4	2014	4.60	6.87	42450	Diesel	Dealer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

Fig 5: Feature variable named “Car_name” is dropped



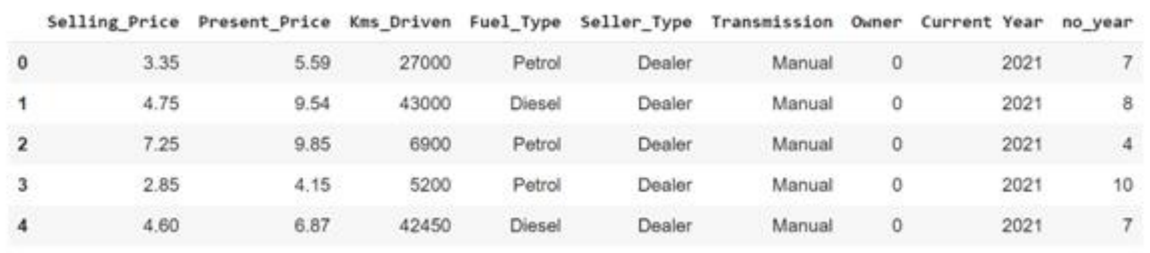
	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner	Current Year
0	2014	3.35	5.59	27000	Petrol	Dealer	Manual	0	2021
1	2013	4.75	9.54	43000	Diesel	Dealer	Manual	0	2021
2	2017	7.25	9.85	6900	Petrol	Dealer	Manual	0	2021
3	2011	2.85	4.15	5200	Petrol	Dealer	Manual	0	2021
4	2014	4.60	6.87	42450	Diesel	Dealer	Manual	0	2021

Fig 6: Feature variable named “Current Year” is appended



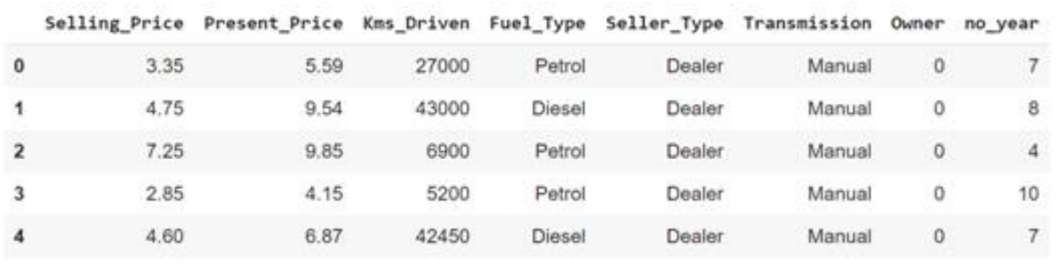
	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner	Current Year	no_year
0	2014	3.35	5.59	27000	Petrol	Dealer	Manual	0	2021	7
1	2013	4.75	9.54	43000	Diesel	Dealer	Manual	0	2021	8
2	2017	7.25	9.85	6900	Petrol	Dealer	Manual	0	2021	4
3	2011	2.85	4.15	5200	Petrol	Dealer	Manual	0	2021	10
4	2014	4.60	6.87	42450	Diesel	Dealer	Manual	0	2021	7

Fig 7: Feature variable named “no_year” is appended



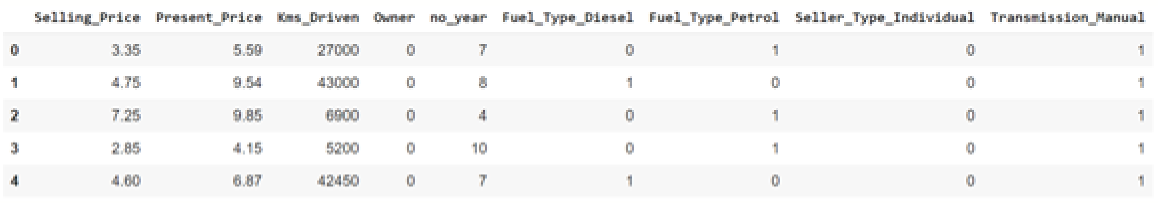
	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner	Current Year	no_year
0	3.35	5.59	27000	Petrol	Dealer	Manual	0	2021	7
1	4.75	9.54	43000	Diesel	Dealer	Manual	0	2021	8
2	7.25	9.85	6900	Petrol	Dealer	Manual	0	2021	4
3	2.85	4.15	5200	Petrol	Dealer	Manual	0	2021	10
4	4.60	6.87	42450	Diesel	Dealer	Manual	0	2021	7

Fig 8: Feature variable named “Year” is dropped



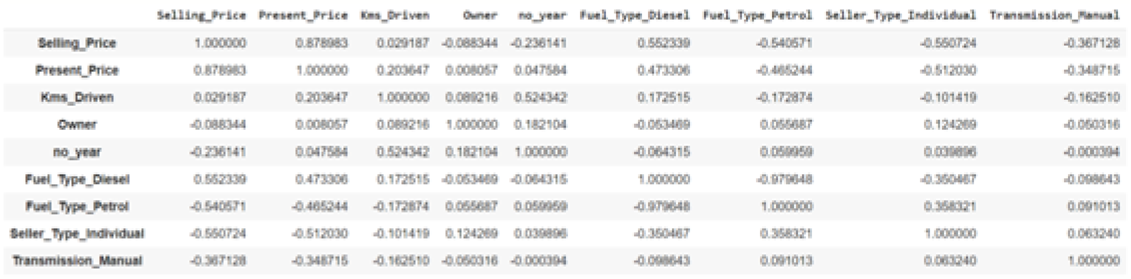
	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner	no_year
0	3.35	5.59	27000	Petrol	Dealer	Manual	0	7
1	4.75	9.54	43000	Diesel	Dealer	Manual	0	8
2	7.25	9.85	6900	Petrol	Dealer	Manual	0	4
3	2.85	4.15	5200	Petrol	Dealer	Manual	0	10
4	4.60	6.87	42450	Diesel	Dealer	Manual	0	7

Fig 9: Feature variable named “Current Year” is dropped



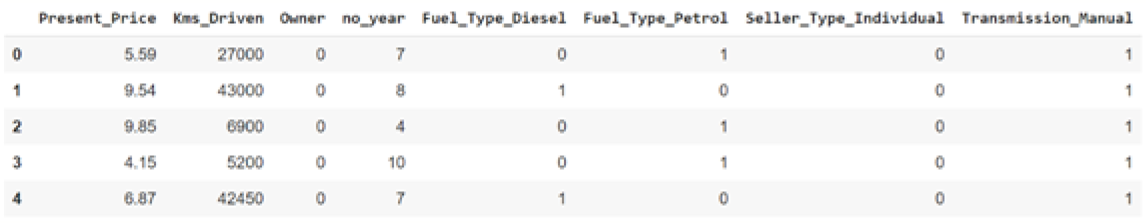
	Selling_Price	Present_Price	Kms_Driven	Owner	no_year	Fuel_Type_Diesel	Fuel_Type_Petrol	Seller_Type_Individual	Transmission_Manual
0	3.35	5.59	27000	0	7	0	1	0	1
1	4.75	9.54	43000	0	8	1	0	0	1
2	7.25	9.85	6900	0	4	0	1	0	1
3	2.85	4.15	5200	0	10	0	1	0	1
4	4.60	6.87	42450	0	7	1	0	0	1

Fig 10: Final dataset after pre-processing

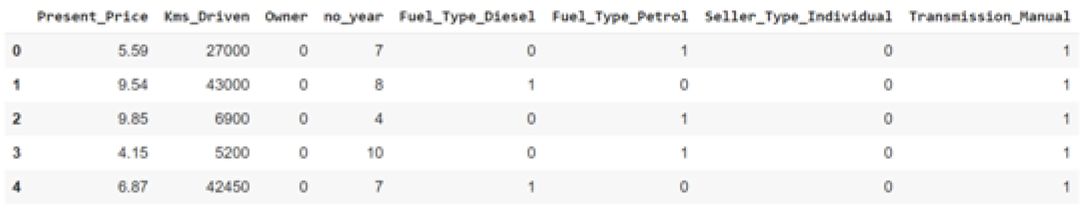


	Selling_Price	Present_Price	Kms_Driven	Owner	no_year	Fuel_Type_Diesel	Fuel_Type_Petrol	Seller_Type_Individual	Transmission_Manual
Selling_Price	1.000000	0.878983	0.029187	-0.088344	-0.236141	0.552339	-0.540571	-0.550724	-0.367128
Present_Price	0.878983	1.000000	0.203647	0.008057	0.047584	0.473306	-0.465244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

Fig 11: Correlation between the features

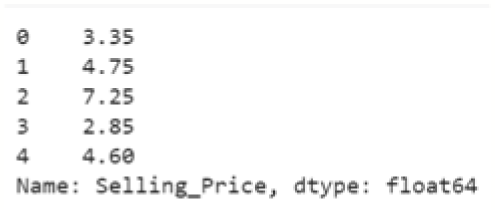


	Present_Price	Kms_Driven	Owner	no_year	Fuel_Type_Diesel	Fuel_Type_Petrol	Seller_Type_Individual	Transmission_Manual
0	5.59	27000	0	7	0	1	0	1
1	9.54	43000	0	8	1	0	0	1
2	9.85	6900	0	4	0	1	0	1
3	4.15	5200	0	10	0	1	0	1
4	6.87	42450	0	7	1	0	0	1



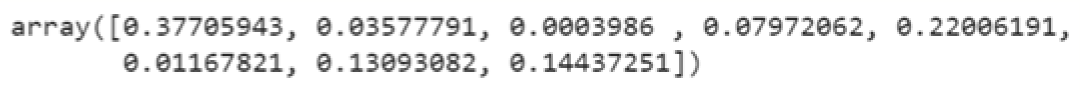
	Present_Price	Kms_Driven	Owner	no_year	Fuel_Type_Diesel	Fuel_Type_Petrol	Seller_Type_Individual	Transmission_Manual
0	5.59	27000	0	7	0	1	0	1
1	9.54	43000	0	8	1	0	0	1
2	9.85	6900	0	4	0	1	0	1
3	4.15	5200	0	10	0	1	0	1
4	6.87	42450	0	7	1	0	0	1

Fig 12: Independent Variables



```
0    3.35
1    4.75
2    7.25
3    2.85
4    4.60
Name: Selling_Price, dtype: float64
```

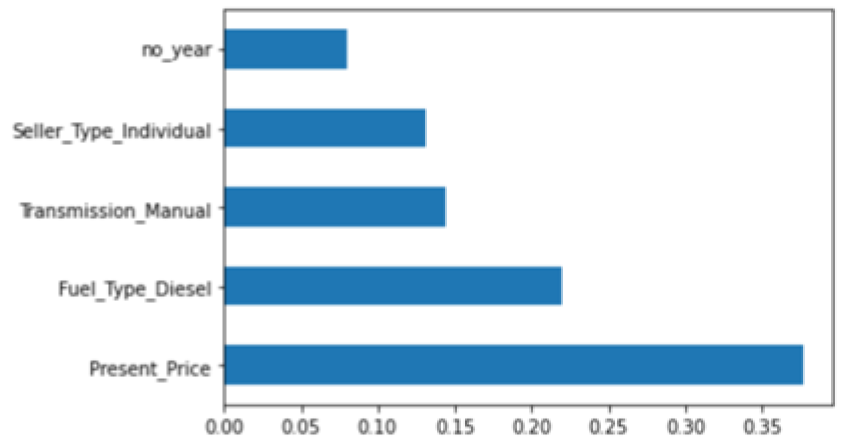
Fig 14: Target Variable



```
array([0.37705943, 0.03577791, 0.0003986 , 0.07972062, 0.22006191,
       0.01167821, 0.13093082, 0.14437251])
```

Fig 15: Feature importance values

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff435c67990>
```



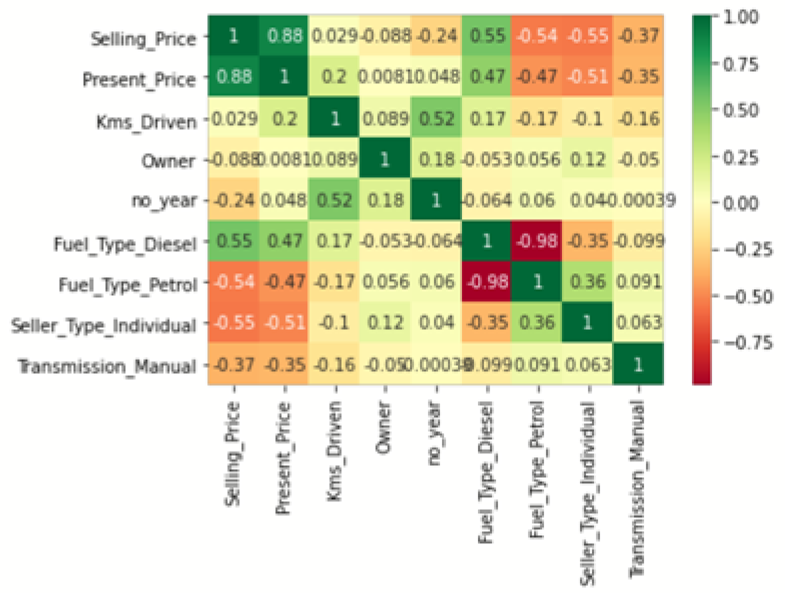


Fig 16&17: Bar plot of features w.r.t feature importance and Heat map showing correlation between the variables present in the dataset.

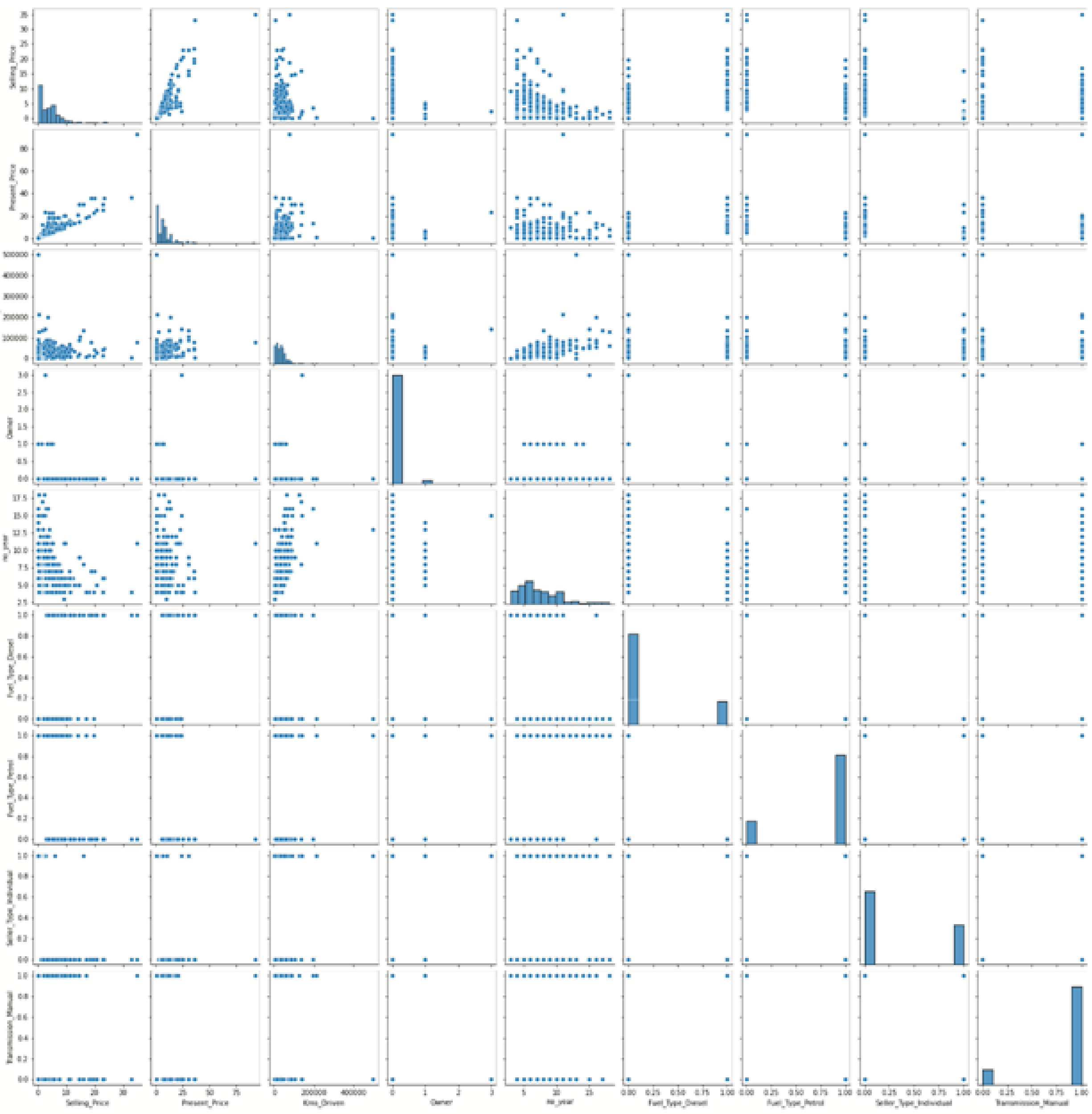
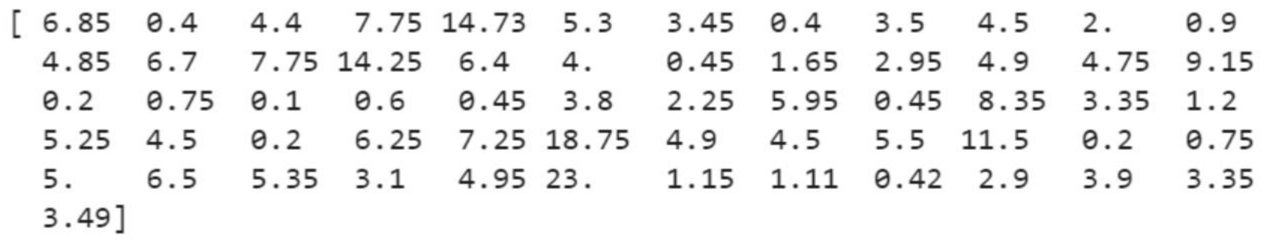
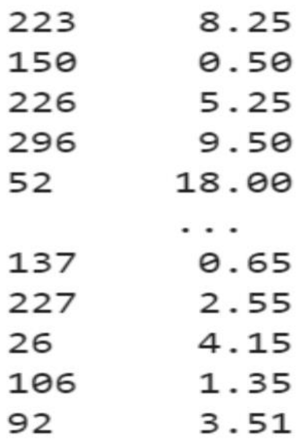


Fig 18: Paiplot off all the features

DECISION TREE REGRESSOR

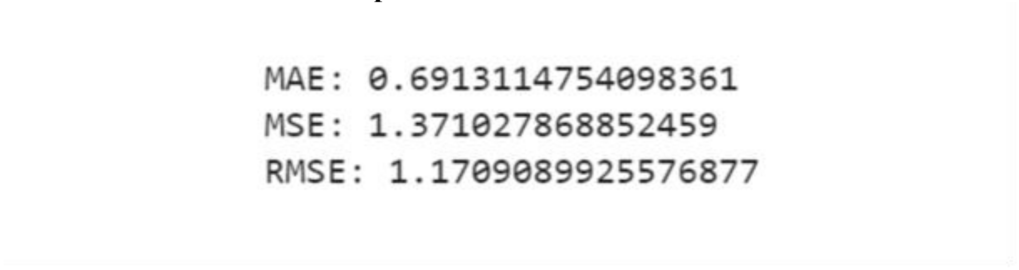


```
[ 6.85  0.4   4.4   7.75 14.73  5.3   3.45  0.4   3.5   4.5   2.    0.9
  4.85  6.7   7.75 14.25  6.4   4.    0.45  1.65  2.95  4.9   4.75  9.15
  0.2   0.75  0.1   0.6   0.45  3.8   2.25  5.95  0.45  8.35  3.35  1.2
  5.25  4.5   0.2   6.25  7.25 18.75  4.9   4.5   5.5  11.5   0.2   0.75
  5.    6.5   5.35  3.1   4.95 23.    1.15  1.11  0.42  2.9   3.9   3.35
  3.49]
```



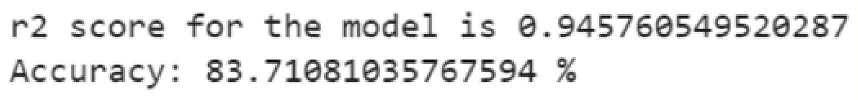
```
223      8.25
150      0.50
226      5.25
296      9.50
52      18.00
         ...
137      0.65
227      2.55
26       4.15
106      1.35
92       3.51
```

Fig 19 & 20 : Predicted values of selling price by decision tree model and Actual values of selling price from the dataset



```
MAE: 0.6913114754098361
MSE: 1.371027868852459
RMSE: 1.1709089925576877
```

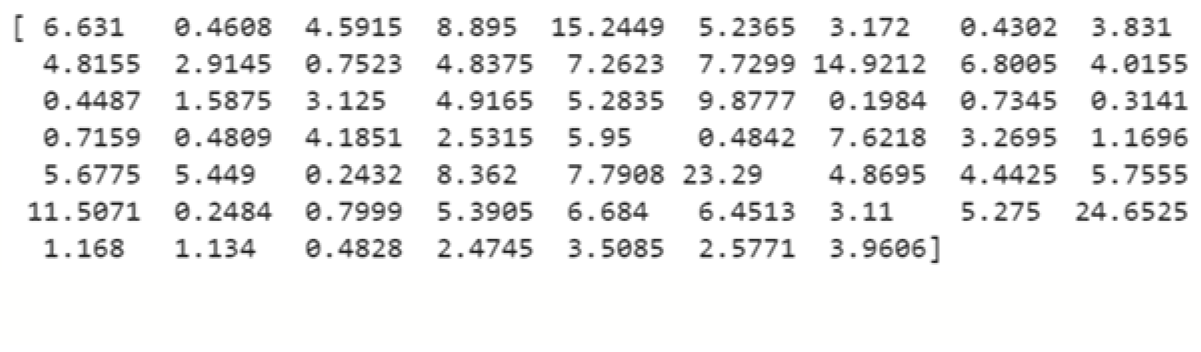
Fig 21: MAE,MSE,RMSE values for decision tree model



```
r2 score for the model is 0.945760549520287
Accuracy: 83.71081035767594 %
```

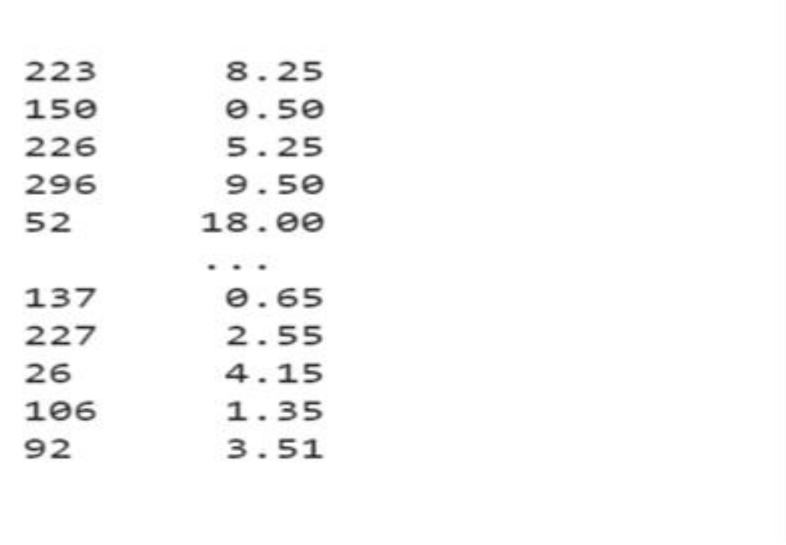
Fig 22: R-Squared score of the decision tree model

RANDOM FOREST REGRESSOR



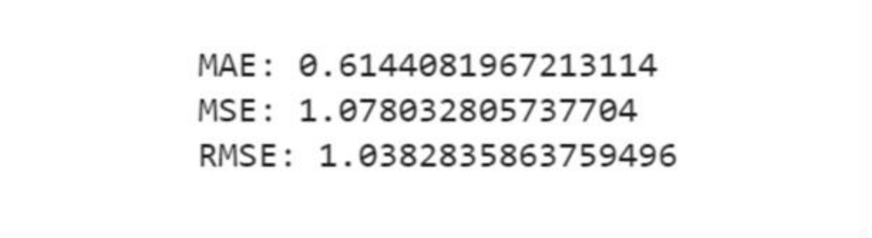
```
[ 6.631   0.4608  4.5915  8.895  15.2449  5.2365  3.172   0.4302  3.831
  4.8155  2.9145  0.7523  4.8375  7.2623  7.7299 14.9212  6.8005  4.0155
  0.4487  1.5875  3.125   4.9165  5.2835  9.8777  0.1984  0.7345  0.3141
  0.7159  0.4809  4.1851  2.5315  5.95    0.4842  7.6218  3.2695  1.1696
  5.6775  5.449   0.2432  8.362   7.7908 23.29    4.8695  4.4425  5.7555
 11.5071  0.2484  0.7999  5.3905  6.684   6.4513  3.11    5.275  24.6525
  1.168   1.134   0.4828  2.4745  3.5085  2.5771  3.9606]
```

Fig 23: Predicted values of selling price by random forest model



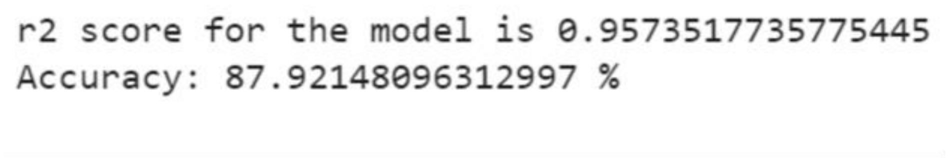
```
223     8.25
150     0.50
226     5.25
296     9.50
52     18.00
       ...
137     0.65
227     2.55
26      4.15
106     1.35
92      3.51
```

Fig 24: Actual values of selling price from the dataset



```
MAE: 0.6144081967213114
MSE: 1.078032805737704
RMSE: 1.0382835863759496
```

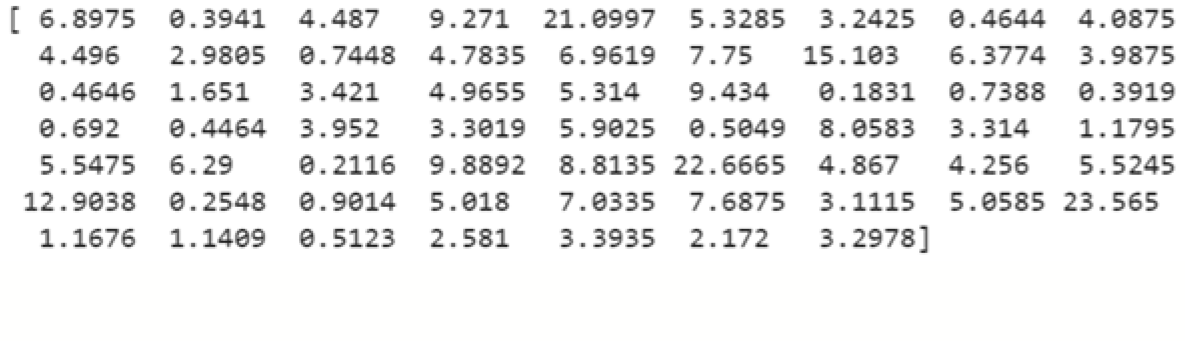
Fig 25: MAE,MSE,RMSE values for random forest model



```
r2 score for the model is 0.9573517735775445
Accuracy: 87.92148096312997 %
```

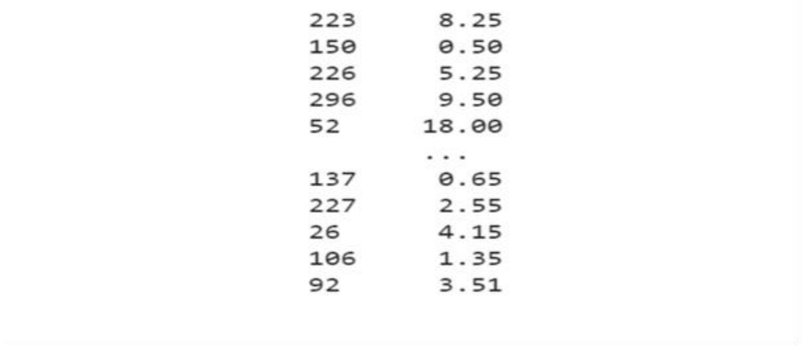
Fig 26: R-Squared score of the random forest model

EXTRA TREES REGRESSOR



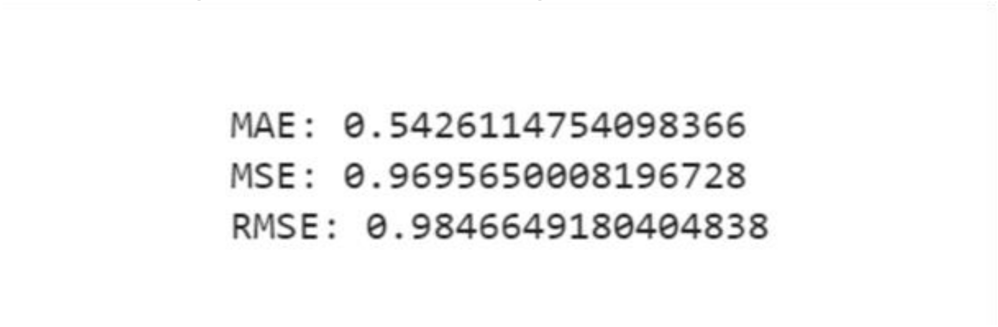
```
[ 6.8975  0.3941  4.487   9.271  21.0997  5.3285  3.2425  0.4644  4.0875
  4.496   2.9805  0.7448  4.7835  6.9619  7.75   15.103   6.3774  3.9875
  0.4646  1.651   3.421   4.9655  5.314   9.434   0.1831  0.7388  0.3919
  0.692   0.4464  3.952   3.3019  5.9025  0.5049  8.0583  3.314   1.1795
  5.5475  6.29    0.2116  9.8892  8.8135 22.6665  4.867   4.256   5.5245
 12.9038  0.2548  0.9014  5.018   7.0335  7.6875  3.1115  5.0585 23.565
  1.1676  1.1409  0.5123  2.581   3.3935  2.172   3.2978]
```

Fig 27: Predicted values of selling price by Extra Trees regressor model



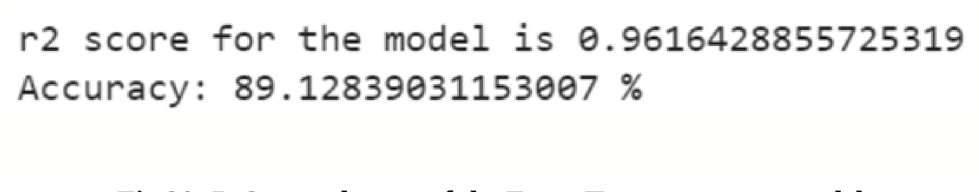
```
223      8.25
150      0.50
226      5.25
296      9.50
52      18.00
         ...
137      0.65
227      2.55
26       4.15
106      1.35
92       3.51
```

Fig 28: Actual values of selling price from the dataset



```
MAE: 0.5426114754098366
MSE: 0.9695650008196728
RMSE: 0.9846649180404838
```

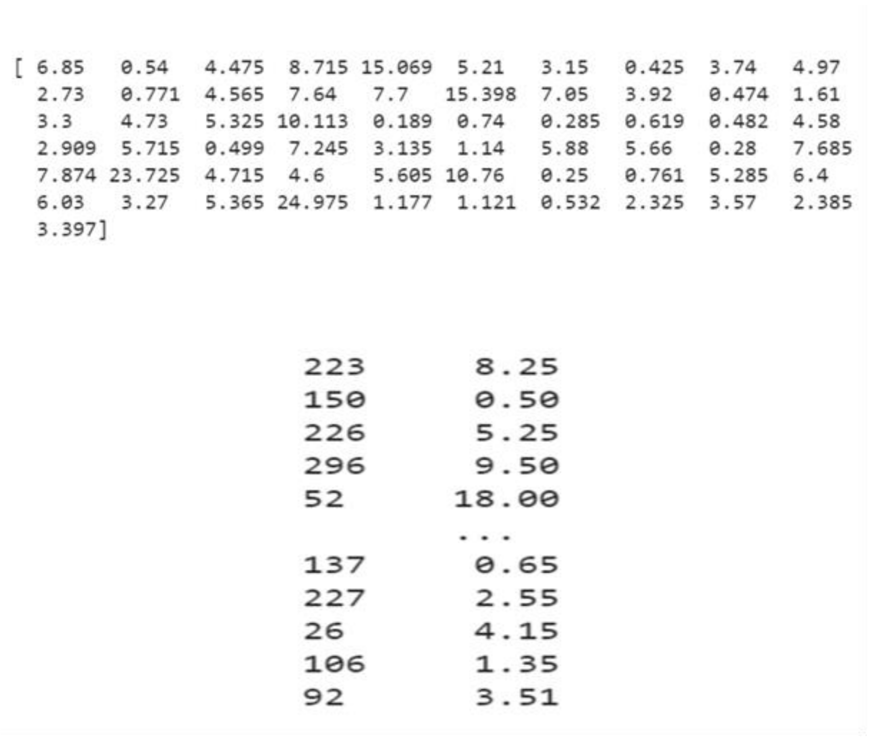
Fig 29: MAE,MSE,RMSE values for model



```
r2 score for the model is 0.9616428855725319
Accuracy: 89.1283903115300 7 %
```

Fig 30: R-Squared score of the Extra Trees regressor model

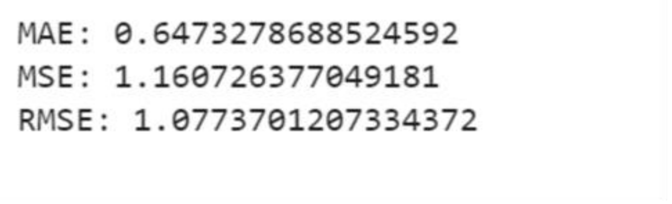
BAGGING REGRESSOR



```
[ 6.85   0.54   4.475  8.715 15.069  5.21   3.15   0.425  3.74   4.97
  2.73   0.771  4.565  7.64   7.7   15.398  7.05   3.92   0.474  1.61
  3.3    4.73   5.325 10.113  0.189  0.74   0.285  0.619  0.482  4.58
  2.909  5.715  0.499  7.245  3.135  1.14   5.88   5.66   0.28   7.685
  7.874 23.725  4.715  4.6    5.605 10.76   0.25   0.761  5.285  6.4
  6.03   3.27   5.365 24.975  1.177  1.121  0.532  2.325  3.57   2.385
  3.397]
```

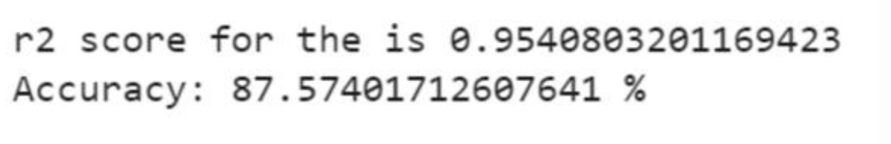
```
223     8.25
150     0.50
226     5.25
296     9.50
52     18.00
        ...
137     0.65
227     2.55
26      4.15
106     1.35
92      3.51
```

Fig 31 & 32: Predicted values of selling price by Bagging regressor model
And
Actual values of selling price from the dataset



```
MAE: 0.6473278688524592
MSE: 1.160726377049181
RMSE: 1.0773701207334372
```

Fig 33: MAE,MSE,RMSE values for Bagging regressor model



```
r2 score for the is 0.9540803201169423
Accuracy: 87.57401712607641 %
```

Fig 34: R-Squared score of the Bagging regressor model

3.2 INFERENCES

From the correlation matrix it is determined that selling price mainly depends on the present showroom price, fuel type and seller type also have a desired impact.

Name of the Regressor	MAE	MSE	RMSE	R-Squared Score	Accuracy (%)
Decision Tree Regressor	**0.6913**	**1.3710**	**1.1709**	**0.9457**	**83.71**
Random Forest Regressor	**0.6144**	**1.0780**	**1.0382**	**0.9573**	**87.92**
Extra Trees Regressor	**0.5426**	**0.9695**	**0.9846**	**0.9616**	**89.12**
Bagging Regressor	**0.6473**	**1.1607**	**1.0773**	**0.9540**	**87.57**

MEAN ABSOLUTE ERROR – The Error is the difference between Actual/True Value and Predicted Value. Absolute Error is the positive value of the error at that instant. The Mean Absolute error is the average of the absolute error of all the observations. MAE = (True Value – Predicted Value) / (Number of Observations).

MEAN SQUARED ERROR – MSE is calculated by taking the average of the square of the difference between True Value and Predicted Value.

MSE = (True Value – Predicted Value)^2 / (Number of Observations).

ROOT MEAN SQUARE VALUE - The Square root of Mean Squared Value (MSE) is the Root Mean Squared Value (RMSE). If RMSE value is greater than 1, then there is low accuracy.

R SQUARE SCORE – It is the statistical measure representing the proportion of variance of dependent variable with respect to the independent variables of the regression model. It ranges from -1 to 1. If the R square score is close to 1, there exists a high accuracy.

CHAPTER IV

4.1 CONCLUSION

Car sales are more often in metropolitan cities where the young software engineers, business man are showing interest to buy the new cars with the updated technology. So, there are high chances that they would not want to keep the old car with them and they opt to sale it. On the other side, new learners and middle-class families don't show interest in buying the new expensive cars and opt to buy the refurbished cars.

Refurbished Car price prediction is a challenging task as it involves various parameters in terms of car specifications and customers satisfaction for solving a high accuracy in predicting the sales values. The major step involved in this project is data analysing and pre - processing. It involved the study of various attributes, cleaning, inserting and deleting some of the important and redundant parameters for a better understanding and implementing the machine learning algorithms.

For this project, the supervised algorithms have been used and four different types of regression algorithms has been implemented. The algorithms are implemented on the sam e dataset in terms of testing data. The error calculations such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE) and R-square score were calculated to predict the high accuracy among the algorithms. Extra Tree Regressor has outperformed decision tree and bagging regressor. By the fact that both random forest and extra trees regressor are equally robust the latter slightly dominated. Extra trees regressor is more optimal algorithm for this problem statement as it has comparatively less time complexity and also similar accuracy.

4.2 RECOMMNDATIONS FOR FUTURE WORK

As a future work, we intend to collect more attributes to the data according to the emerging trends in the latest technologies in terms of human comfort and cars specifications. The advanced techniques like artificial neural networks can be implemented to predict the refurbished car prices.

REFERENCES

1. https://www.kdnuggets.com/2020/01/decision-tree-algorithm-explained.html#:~:text=Decision%20Tree%20algorithm%20belongs%20to%20the%20family%20of%20supervised%20learning%20algorithms.&text=The%20goal%20of%20using%20a,prior%20data(training%20data)

2. https://builtin.com/data-science/random-forest-algorithm

3. https://machinelearningmastery.com/bagging-and-random-forest-ensemble-algorithms-for-machine-learning/

4. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

5. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.BaggingRegressor.html

6. https://scikit-learn.org/stable/auto_examples/tree/plot_tree_regression.html

7. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesRegressor.html