INTRODUCTION

Predicting the price of cars is both an important and interesting problem. Estimating the price of used cars is of very high commercial importance too. Predicting the value of a car is not a simple task. It is trite knowledge that the value of cars depends on a number of factors. The most important ones are usually its make (and model), its mileage (the number of kilometers it runs) and its horsepower.

Other factors such as the type of fuel it uses, the braking system, acceleration, the volume of its cylinders (measured in cc), its size, number of doors, weight of the car, its physical state, wheels, etc. all may influence the price as well.

In this project I have considered only a small data set of the factors mentioned above.

This project is based on supervised machine learning techniques to predict the price of cars depending on some entities enclosed in the data set.

Different supervised machine learning techniques like Linear Regression, Regression version of Decision Tree and Random Forest analysis have been used to make predictions.

The predictions are then evaluated and compared in order to find those which provide the best performance.

Problem statement

To predict the price of a car based on the factors/entities provided through developing a model using various machine learning techniques.

Objective

Predicting the price of a car using different techniques like Linear Regression, Regression version of Decision Tree and Random Forest analysis.The predictions are then evaluated and compared in order to find those which provide the best performance.

Requirement Specifications

Hardware Requirements

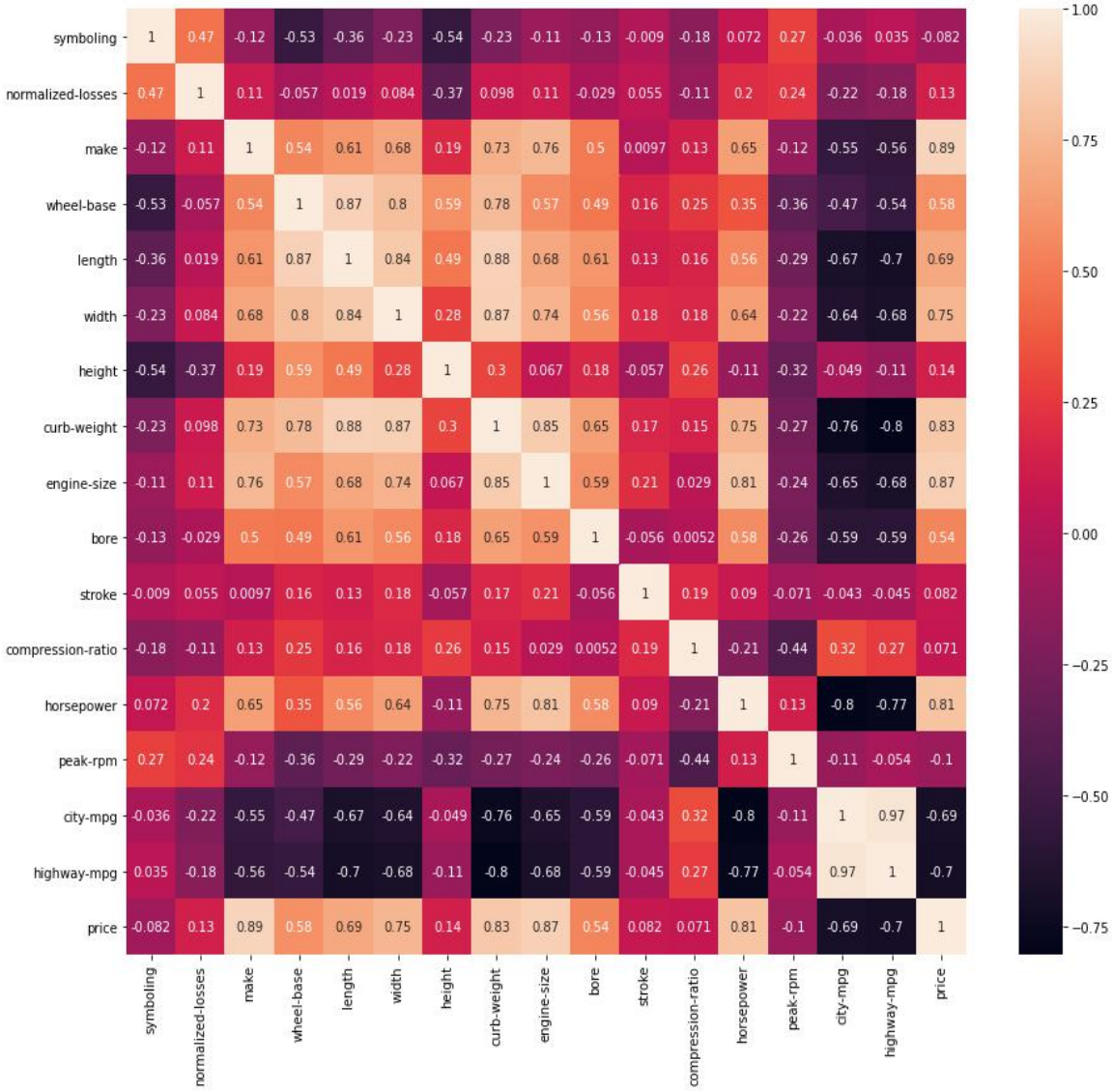
- Hard disk : 40GB Hard disk or higher.
- Processor : Intel i3 core/AMD Athlon or higher.
- Memory : 1 GB or higher.

Software Requirements

- Windows OS XP/7/8/10.
- Jupyter notebook(*).
- Python 3.6 and above.

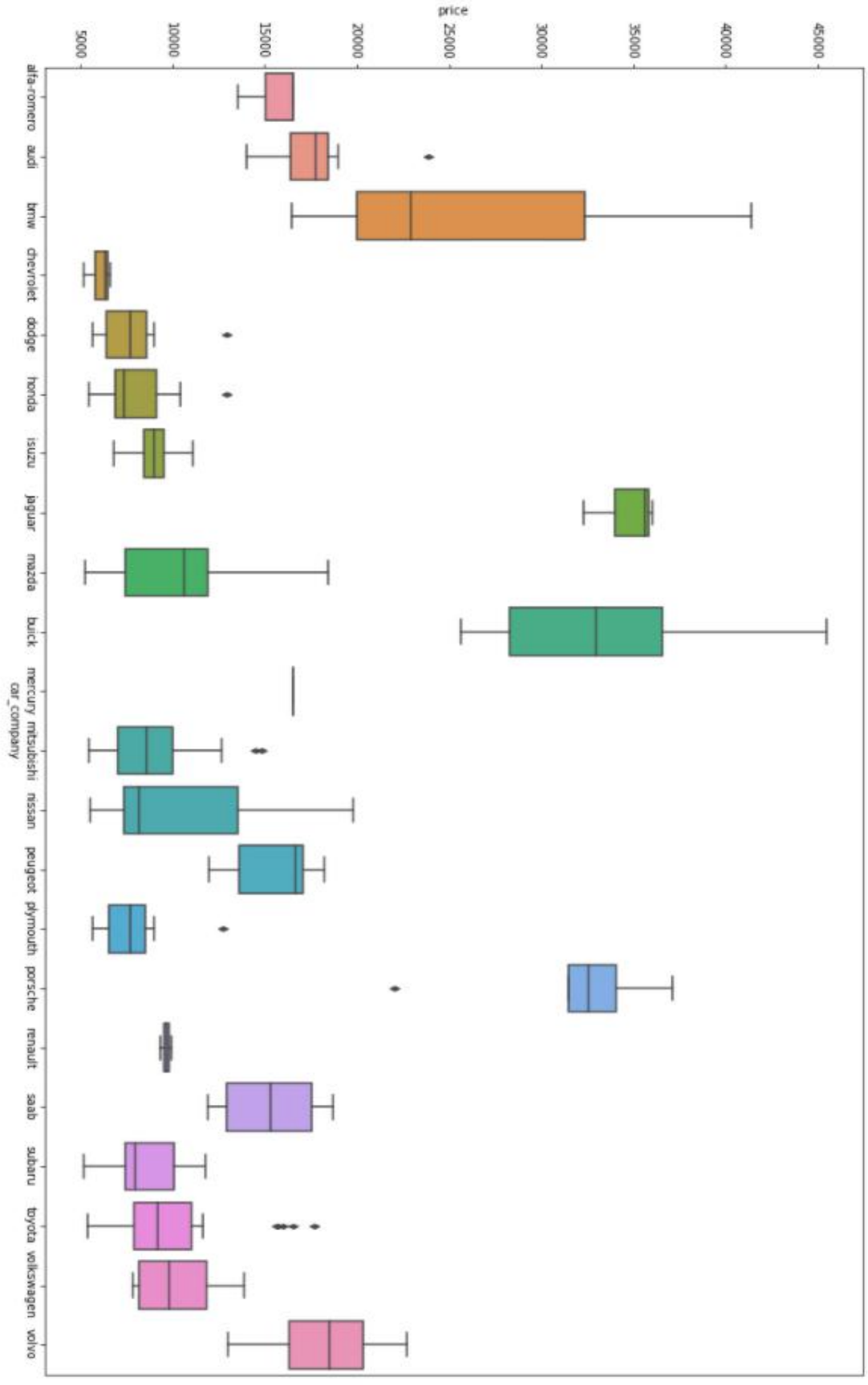
Data Analysis

Heatmap



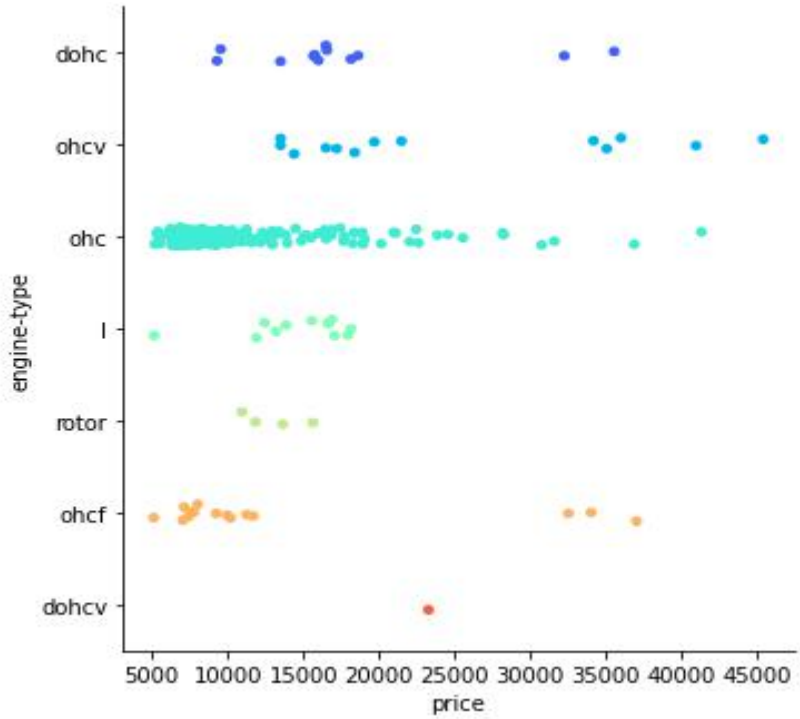
Darker shades and brighter shades are co-related. Darker shades are negatively co-related i.e inversely proportional. Brighter shades are positively co-related i.e directly proportional And those which are in mid-range have less or no co-relation.For example, ‘city-mpg’ and ‘price’ are negatively co-related. As ‘city-mpg’ lies in darker shade it is inversely proportional to price i.e if ‘city-mpg’ decreases ‘price’ increases and vice-verse.

Boxplot


price
5000
10000
15000
20000
25000
30000
35000
40000
45000
car_company
alfa-romero
audi
bmw
chevrolet
dodge
honda
isuzu
jaguar
mazda
buick
mercury
mitsubishi
nissan
peugeot
plymouth
porsche
renault
saab
subaru
toyota
volkswagen
volvo

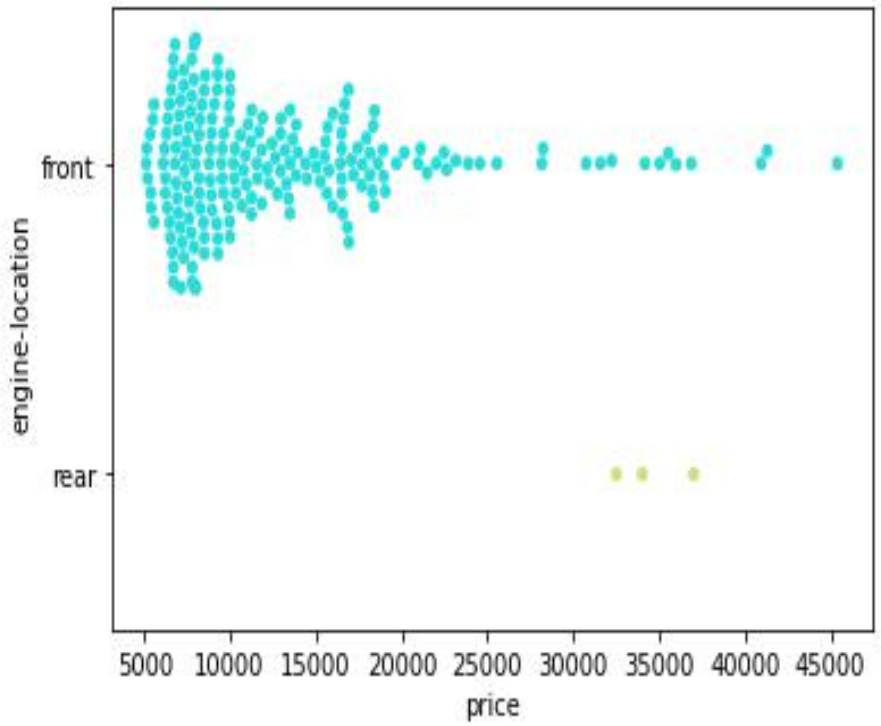
From this boxplot, we can derive car manufacturer (‘make’) with most expensive cars i.e to which ‘make’ does the most priced car belongs to. And we can derive the price range of each car.

Catplot



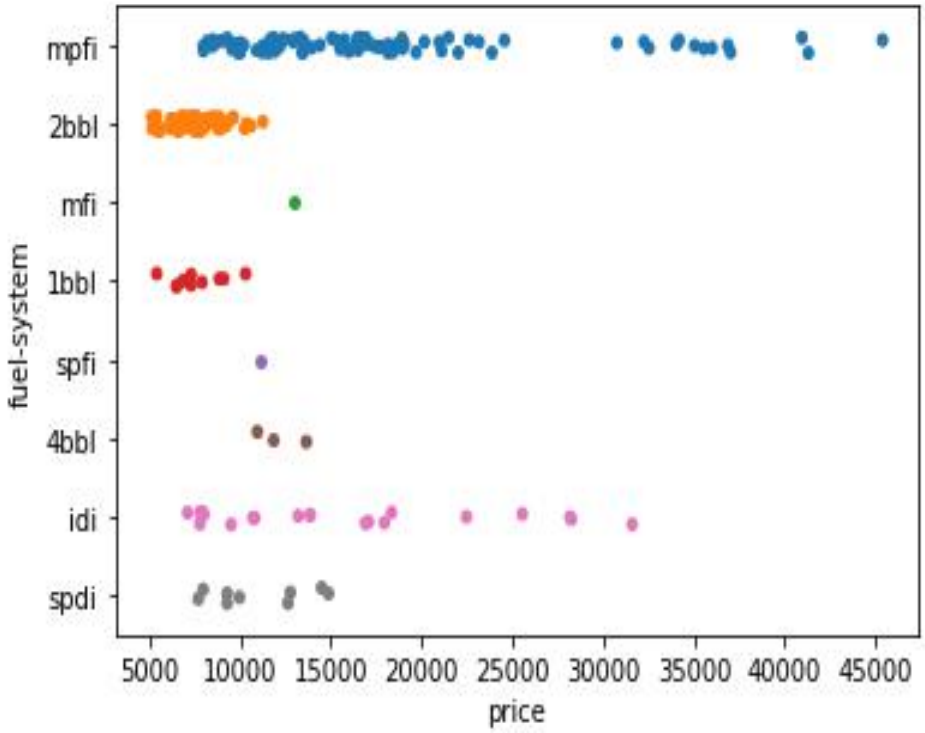
Catplot is used to plot a categorical data. In the above catplot, we can observe that ‘engine-type’ is a categorical data i.e it has different indicatory values. From this we can derive which indicatory data is of what price value .Most of the cars have ohc type of engine.

Swarmplot



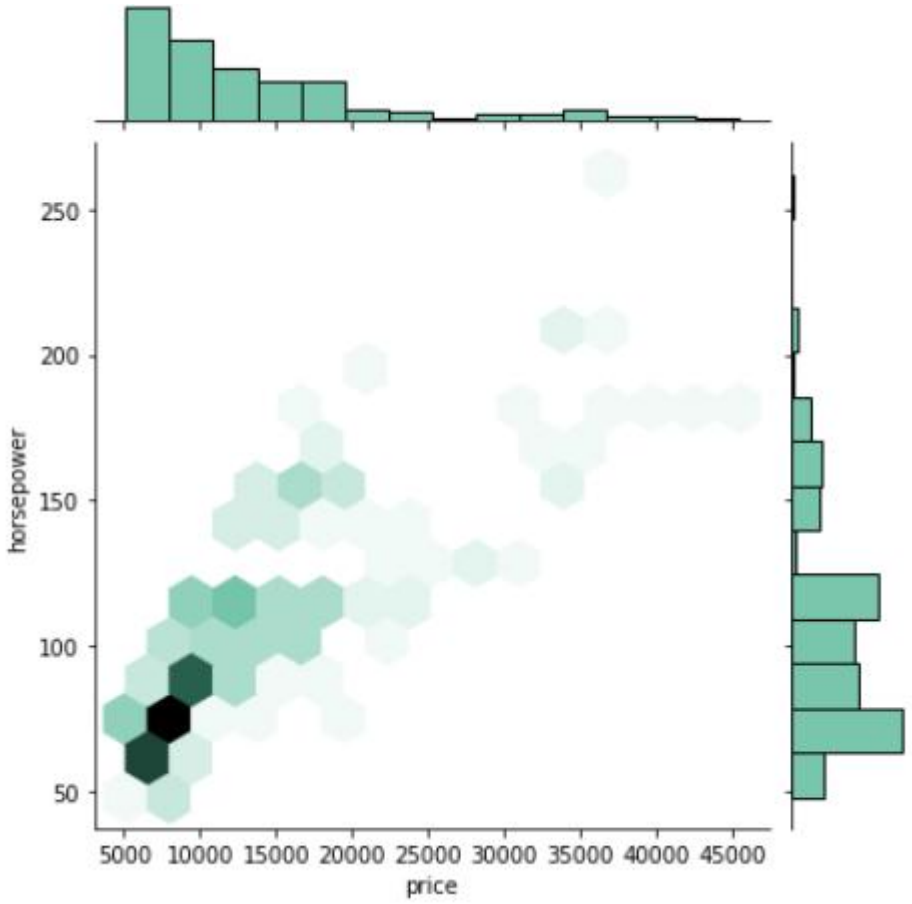
From the above swamplot, we can observe that most of the cars are having engine placed in the front i.e ‘engine-locatioin’ : front. Only small count of cars have engine placed in rear .Since most of the cars have their engine in the front, it doesn’t impact on car’s price. So as, we can drop this particular entity.

Stirplot



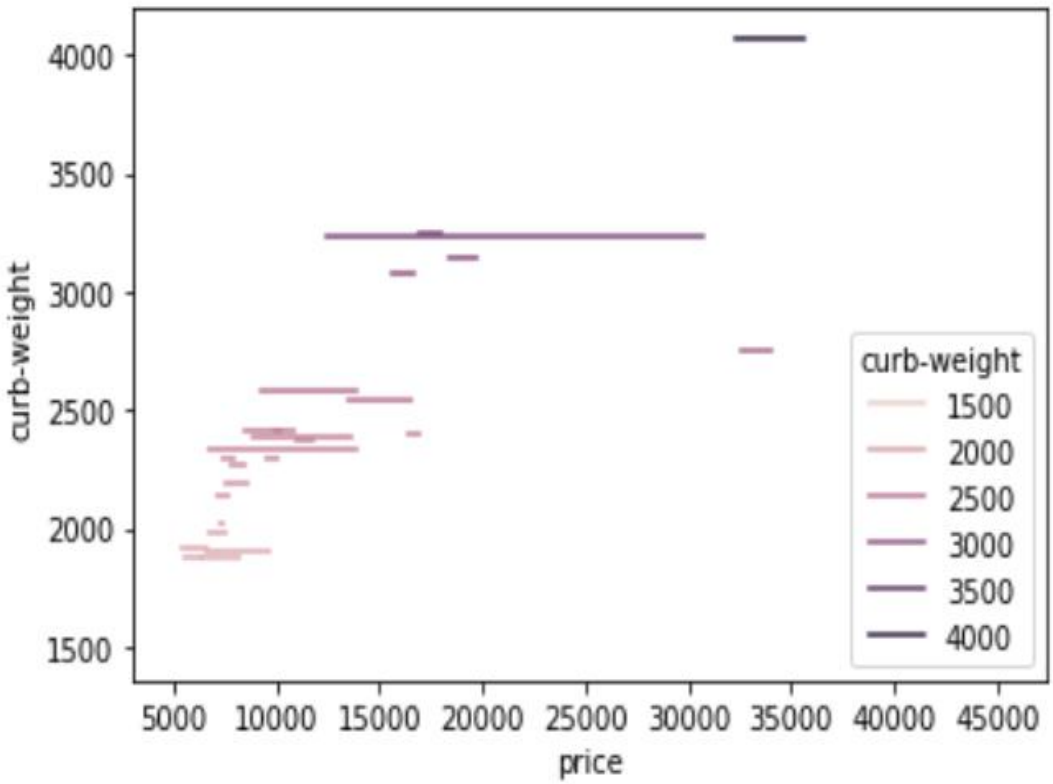
From the above stirplot, we can derive the price of car depending on the 'fuel-system' used in that car.Hence this entity has a role in determining price of the car. Most of the cars have mpfi fuel-system and are priced from10000 to 45000.

Jointplot



Jointplot is used to compare both bivariate(comparing 2 features) and univariate data. In the above jointplot, we can observe that cars having 'horsepower' 50 to 100 are priced in the range 5000 to 15000, out of which more cars with that horsepower are priced between 5000 to 10000. And the plot on top and right indicates the number of cars and horsepower density respectively.

Lineplot



With the help of line plot it is possible to plot several semantic groupings. From the above line plot we can observe that cars priced bet-ween 5000 to 10000 have ‘curb-weight’ of 1500 to 2000 Most of the cars priced between 15000 to 30000 have ‘curb-weight’ of 3000 to 3500.

Machine Learning Model Training

Predicting the price of a car depends on determining the strength and character of the relationship between one dependent variable ('Price') and series of other independent variables ('make','num-of-doors','fuel-type','horsepower' etc.). To achieve this we use Regression approach.

1. Linear Regression

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
model = LinearRegression()
model.fit(x_train,y_train)
prediction = model.predict(x_test)

r2 = r2_score(y_test, prediction)
print(round(r2,4))
```

0.8247

We got an accuracy of 82 %.

2. Decision Tree (Regression Version)

```
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import r2_score
model_2 = DecisionTreeRegressor()
model_2.fit(x_train,y_train)
prediction_2 = model_2.predict(x_test)

r2 = r2_score(y_test, prediction_2)
print(round(r2,4))
```

0.8811
We got an accuracy of 88 %.

3. Random Forest (Regression Version)

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import r2_score
model_3 = RandomForestRegressor()
model_3.fit(x_train,y_train)
prediction_3 = model_3.predict(x_test)

r2 = r2_score(y_test, prediction_3)
print(round(r2,4))
```

0.9416
We got an accuracy of 94 %.

4. ExtraTree (Regressioin Version)

```
from sklearn.tree import ExtraTreeRegressor
from sklearn.metrics import r2_score
model_4 = ExtraTreeRegressor()
model_4.fit(x_train,y_train)
prediction_4 = model_4.predict(x_test)

r2 = r2_score(y_test, prediction_4)
print(round(r2,4))
```

0.8115
We got an accuracy of 81 %.

Model Chart

Sl.no	Algorithm Name	Metric used
1.	Random Forest (Regression Version)	r2_score (0.9416)
2.	Decision Tree (Regression Version)	r2_score (0.8811)
3.	Linear Regression	r2_score (0.8247)
4.	Extra Tree (Regression Version)	r2_score (0.8115)

Hurdles

Jupyter notebook has a disadvantage regarding how many times a cell is run by the user. If one tries to re-run the same cell again which contains some data types conversion code, it throws an error stating ‘cannot convert again’.And I was unaware of this problem. I tried to re-run the cell and it again showed the same error.

To overcome this, one can read the data set again into another variable.

While splitting the data as train and test using train_test_split module, if we pass an argument called ‘shuffle’=True, the data set is first shuffled and then it splits.The accuracy i.e r2_score depends on the train and test data sets. I tried both the options with ‘shuffle’=True and ‘shuffle’=False and observed a major change in r2_score. So its a good practice to pass the argument ‘shuffle’=True to get most accurate r2_score.

Conclusion

In this project, four different machine learning techniques have been used to forecast the price of cars.All the four techniques gave acceptable results.

Among the four techniques used Random Forest (Regression Version) gave higher accurate results.

As future work, one can intend to collect more data and can use more advanced techniques like neural networks.

Bibliography

Website

1. https://github.com
2. https://seaborn.pydata.org
3. https://www.kaggle.com

Books

1. Hands on Machine Learning with Scikit-Learn.