AUTOMOBILE PRICE TRACKER

Name : Tharun Manikonda

USN : 1KS18CS045

EMAIL : tharunmanikonda885@gmail.com

Company : Innovation Creations

INSTRUCTOR : Abhishek C

ACKNOWLEDGEMENT

I would like to express my special thanks of gratitude to my mentor Mr. Abhishek c, who gave me the opportunity to do this project of Automobile Price Tracker, who also helped me in completing my project. I came to know about so many new things I am thankful to them.

Secondly, I would like to thank “Take It Easy Engineers” for providing internships for us. Being this my first internship I’m very happy to be tied with TIE.

Finally, I would like to thank my parents and friends who helped me a lot in finalizing this project within a limited time frame.

ABSTRACT

The project is about to predict the price of the Automobiles by using a Machine learning algorithm.

The project titled “Automobile Price Tracker” the purpose of the project is to build a machine learning model to predict the best price on the give data. The data consists of rows and columns of real-world data. With the given data I have tried to plot graphs like heatmap, pair plot joint plot.

To find the model I choose three machine learning algorithms linear regression, decision trees, random forest, and all three algorithms that are trained and tested. Our final agenda is to find the R_2 score of the data. I have tried all three algorithms and I got the R_2 score. The R_2 score of the linear regression algorithm is 0.8264, for decision tree 0.8936, for random forest 0.935.

About The Company

ICS is a digital service provider that aims to provide software, designing and marketing solutions to individuals and businesses. At ICS, we believe that service and quality is the key to success.

We provide all kinds of technological and designing solutions from Billing Software to Web Designs or any custom demand that you may have. Experience the service like none other!

Some of our services include:

Development - We develop responsive, functional and super fast websites. We keep User Experience in mind while creating websites. A website should load quickly and should be accessible even on a small view-port and slow internet connection.

Mobile Application - We offer a wide range of professional Android, iOS & Hybrid app development services for our global clients, from a startup to a large enterprise.

Design - We offer professional Graphic design, Brochure design & Logo design. We are experts in crafting visual content to convey the right message to the customers.

Consultancy - We are here to provide you with expert advice on your design and development requirement.

Videos - We create a polished professional video that impresses your audience.

image.

INDEX

INTRODUCTION

The goal of the project is to find the desired value by analyzing and exploring the real world data of automobiles by a machine learning model. For analyzing the data I have used few data set models heat map, joint plot, pair plot. By using this I have plotted some graphs.

Hereby analyzing the data I found that the price increases linearly as few data sets increase. So I have chosen these three machine learning algorithms.

1. Linear regression
2. Decision tree
3. Random forest

The internship helps a lot, it has been a great opportunity for us to learn, previous I haven't done any internship this is my first internship it helps me a lot. In this internship first, we go through some basics of python language after that we jumped into the actual syllabus. We started with NumPy and we go through pandas, seaborn, and a few more various libraries to understand the whole machine learning process, statistics which plays a crucial role in understanding the logic behind it.

Problem Statement and Objective

PROBLEM STATEMENT: Finding the R2_score of the given real-world data set by using the machine learning regression model.

OBJECTIVE STATMENT: Understanding the given data set and modifying the libraries according to the data set, and plotting the graphs for the data set and predicting the value.

Requirement Specification

Hardware requirements:

- Ram : 8 GB
- Processer : i5

Software Requirements:

- Python libraries
  - NumPy
  - Pandas
  - Matplotlib
  - Seaborn
- Annaconda
- Jupyter

Exploratory data analysis

```
In [8]: AMD.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205 entries, 0 to 204
Data columns (total 26 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   symboling          205 non-null    int64
 1   normalized-losses  205 non-null    int64
 2   make               205 non-null    object
 3   fuel-type          205 non-null    object
 4   aspiration         205 non-null    object
 5   num-of-doors       205 non-null    object
 6   body-style         205 non-null    object
 7   drive-wheels       205 non-null    object
 8   engine-location    205 non-null    object
 9   wheel-base         205 non-null    float64
 10  length             205 non-null    float64
 11  width              205 non-null    float64
 12  height             205 non-null    float64
 13  curb-weight        205 non-null    int64
 14  engine-type        205 non-null    object
 15  num-of-cylinders   205 non-null    object
 16  engine-size        205 non-null    int64
 17  fuel-system        205 non-null    object
 18  bore               205 non-null    float64
 19  stroke             205 non-null    float64
 20  compression-ratio  205 non-null    float64
 21  horsepower         205 non-null    int64
 22  peak-rpm           205 non-null    int64
 23  city-mpg           205 non-null    int64
 24  highway-mpg        205 non-null    int64
 25  price              205 non-null    int64
dtypes: float64(7), int64(9), object(10)
memory usage: 41.8+ KB
```

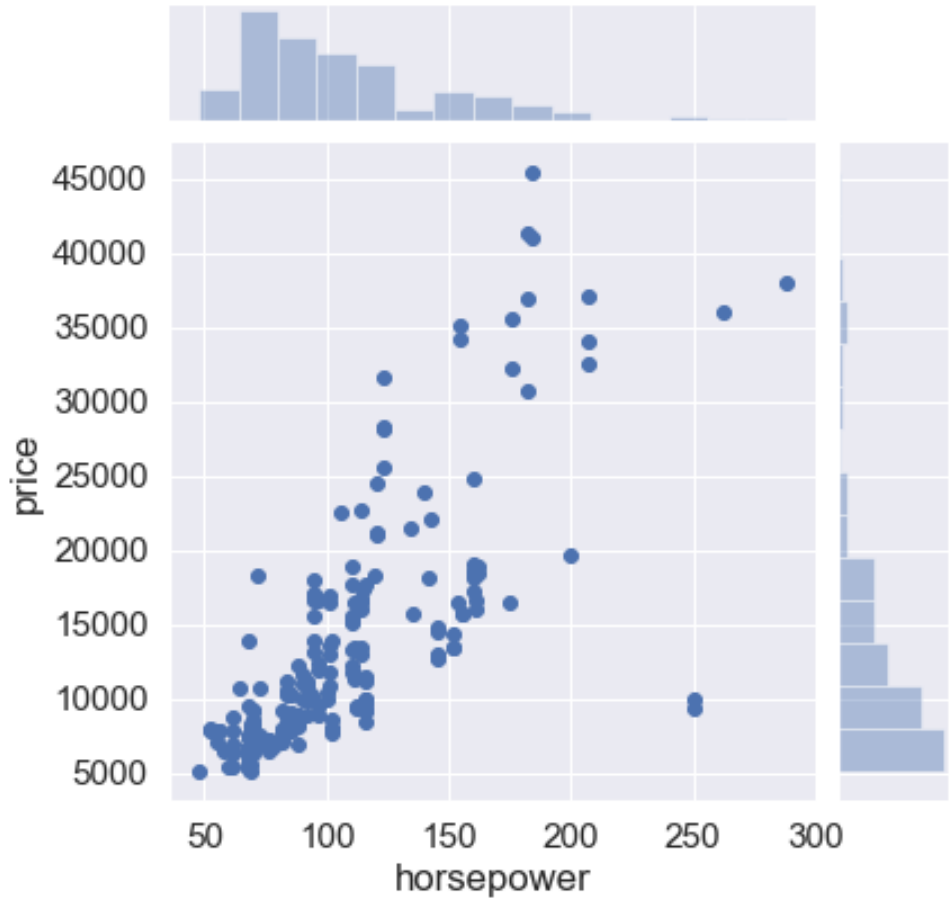
Conclusion:

here it has range index of two hundred and entries and data columns of 26

Joint plot between price and horse power:

```
In [32]: sns.jointplot(x='horsepower',y='price',data =AMD)
         sns.jointplot(x='bore',y='price',data =AMD)
         sns.jointplot(x='stroke',y='price',data =AMD)
         sns.jointplot(x='normalized-losses',y='price',data =AMD)
         sns.jointplot(x='peak-rpm',y='price',data = AMD)
         sns.jointplot(x='city-mpg',y='price',data =AMD)
         sns.jointplot(x='highway-mpg',y='price',data =AMD)

Out[32]: <seaborn.axisgrid.JointGrid at 0x1e5f48a9220>
```



Conclusion:

X-axis = horsepower

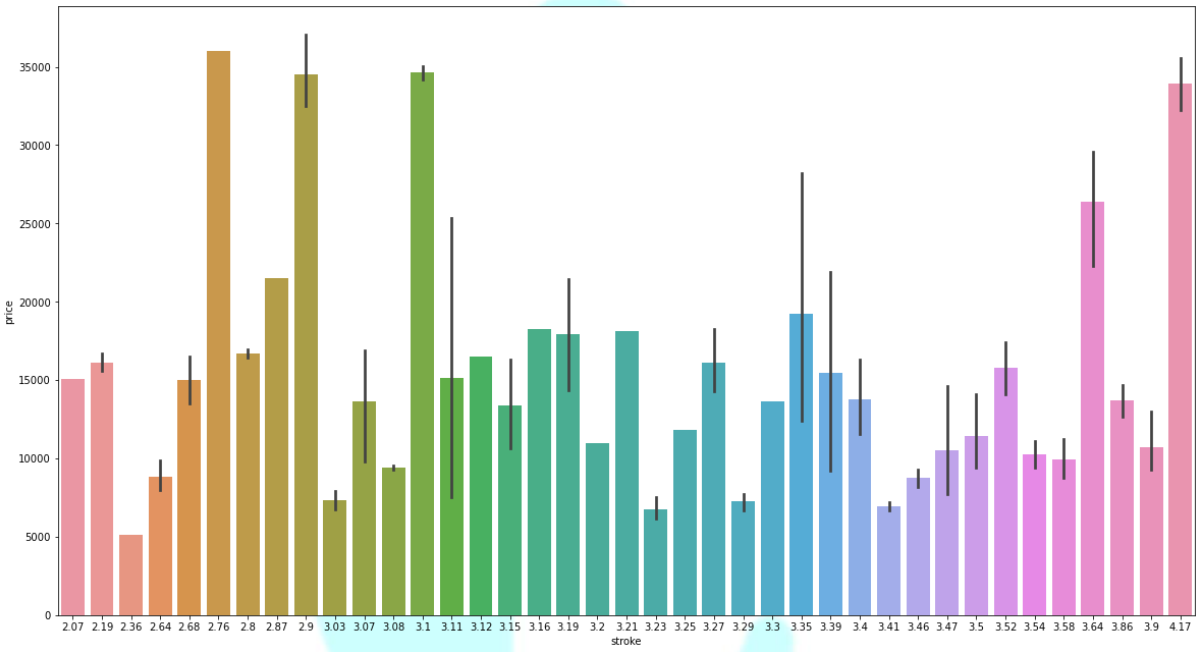
Y-axis = price

Here we can observe when horsepower increases the price will also increase .on average the most used horsepower is 100 ,as we can see there is a price increase at that range .

Bar plot between horsepower and stroke:

```
In [7]: sns.barplot(x='stroke',y='price',data=AMD)
        #plt.rcParams["figure.figsize"] = (20,11)

Out[7]: <matplotlib.axes._subplots.AxesSubplot at 0x1a0cca008e0>
```



Conclusion:

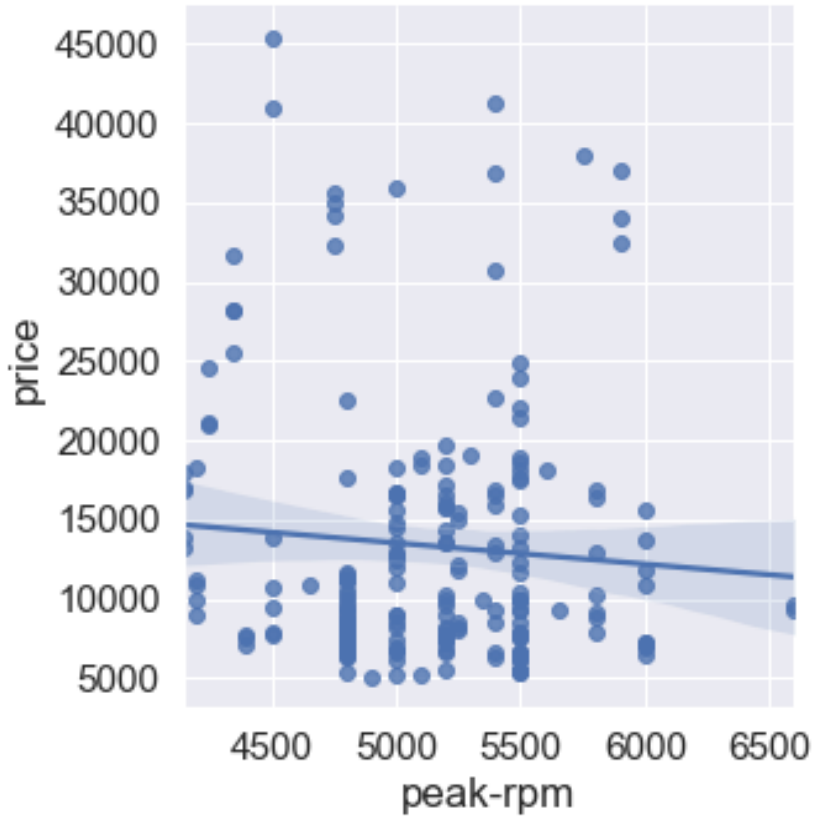
- X-Axis: stroke
- Y-Axis: price

In a barplot each bar represents a number. The following graph shows between price and stroke. As we can see the costliest car has a stroke of 2.68. the lowest stroke 2.36 had a price of 4900.

Lm plot between price and peak-rpm;

```
In [33]: sns.lmplot(x='horsepower',y='price',data = AMD)
         sns.lmplot(x='bore',y='price',data = AMD)
         sns.lmplot(x='stroke',y='price',data =AMD)
         sns.lmplot(x='normalized-losses',y='price',data =AMD)
         sns.lmplot(x='peak-rpm',y='price',data =AMD)
         sns.lmplot(x='city-mpg',y='price',data = AMD)
         sns.lmplot(x='highway-mpg',y='price',data =AMD)

Out[33]: <seaborn.axisgrid.FacetGrid at 0x1e5f7ff9400>
```



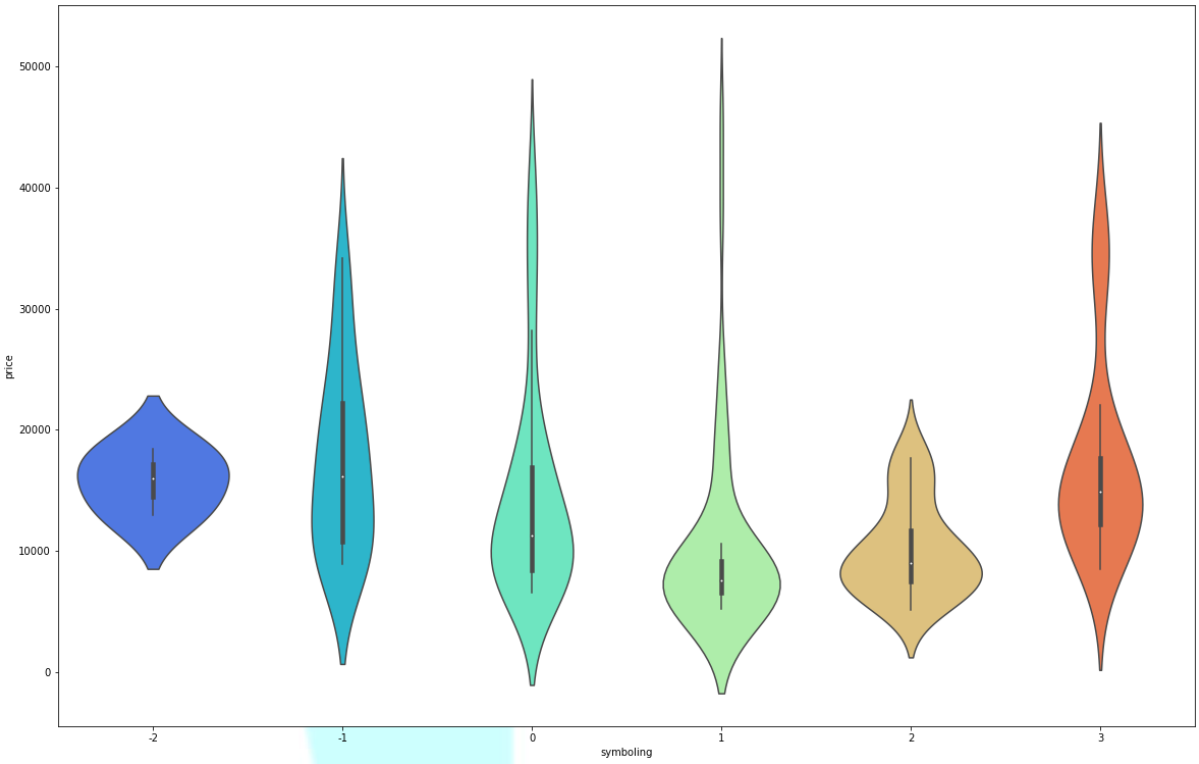
Conclusion :

- X-Axis : peak rpm
- Y-Axis : price

Here we can see the line going down when the peak-rpm values on the x-axis getting increase.

Voilinplot :

```
In [12]:  sns.violinplot(x="symboling", y="price", data=AMD,palette='rainbow')
          plt.rcParams["figure.figsize"] = (20,13)
```



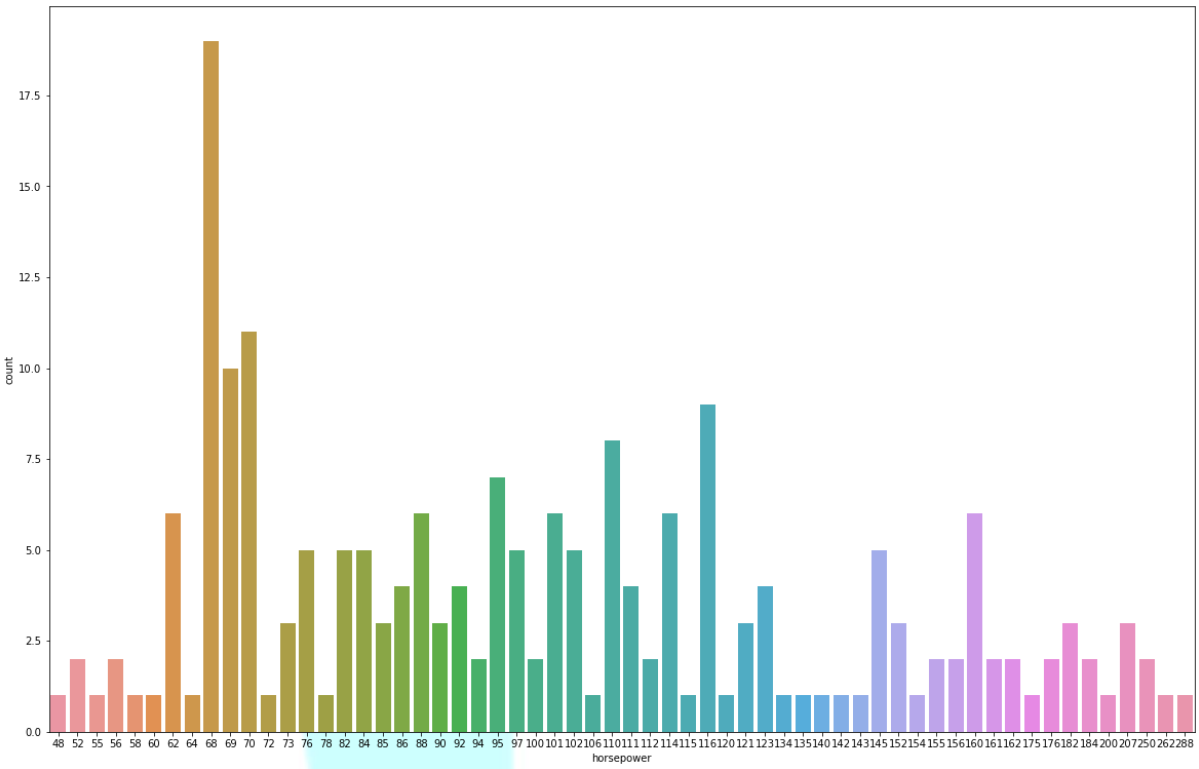
Conclusion:

- X-Axis : symboling
- Y-axis: price

In the above graph at symboling -2 the price range is between 10000 and 20000 and the width also increases so there is more number of users at that range, at -1 the price getting increased and the range is between 0 to40000 and as the price increases, the user's width decreases, at 0 the price range is between 0 to 49000 and in this range the price is also high, at 1 it touches the highest price, at 2 it has lowest price nut at a range of 10000 to 20000 this has more users and at 3 again it increases.

Countplot of horsepower:

```
In [31]:  sns.countplot(x='horsepower',data=AMD)
          plt.rcParams["figure.figsize"] = (20,15)
```



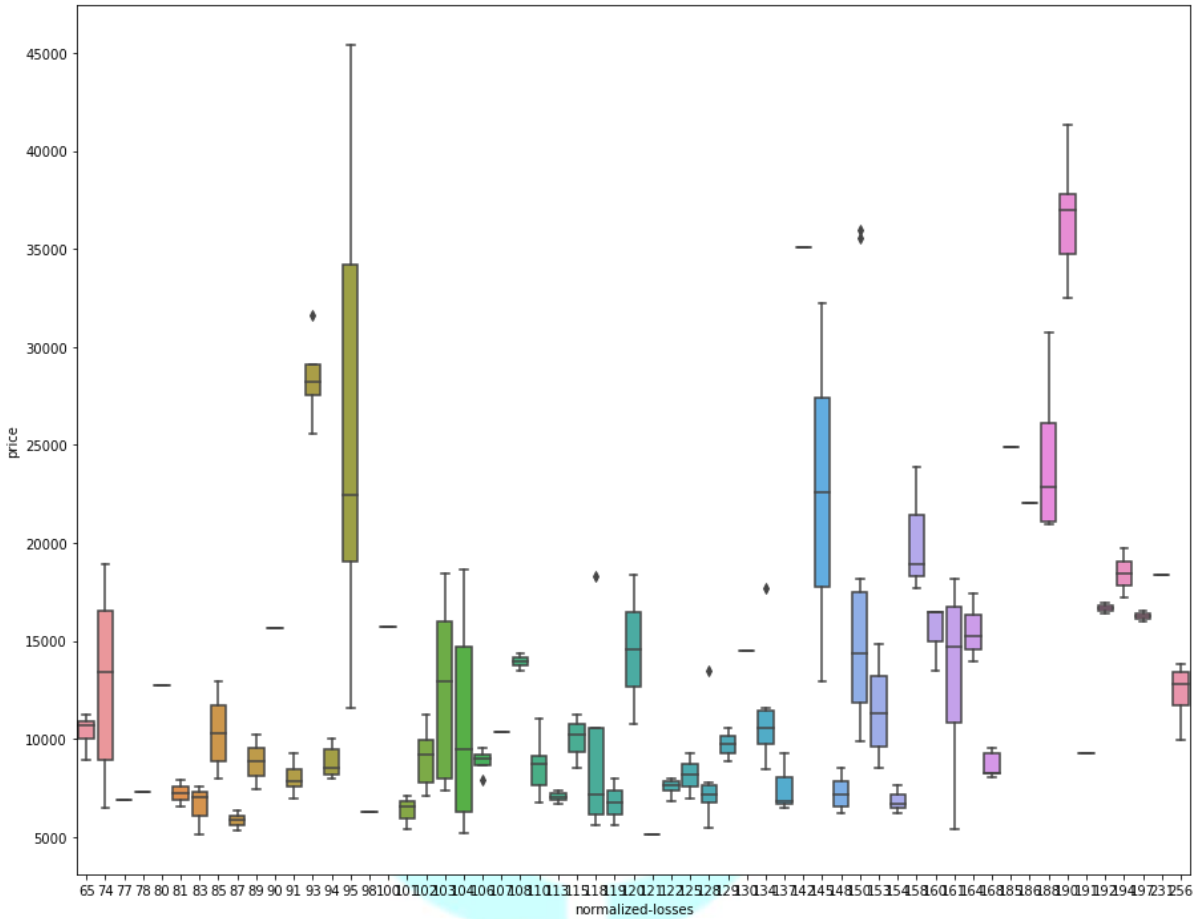
Conclusion :

- X-Axis :count
- Y-Axis:horsepower

Here we can see at horsepower 40 the count is very low it continues up to 64 at the same range and at 66 it touches the highest count and till 120 it shows medium values then it starts decreasing as the horsepower increases the count values are decreasing.

Boxplot graph between normalized-losses and price:

```
In [36]:  sns.boxplot(x="normalized-losses",y="price",data=AMD)
          plt.rcParams["figure.figsize"] = (15,12)
```



Conclusion:

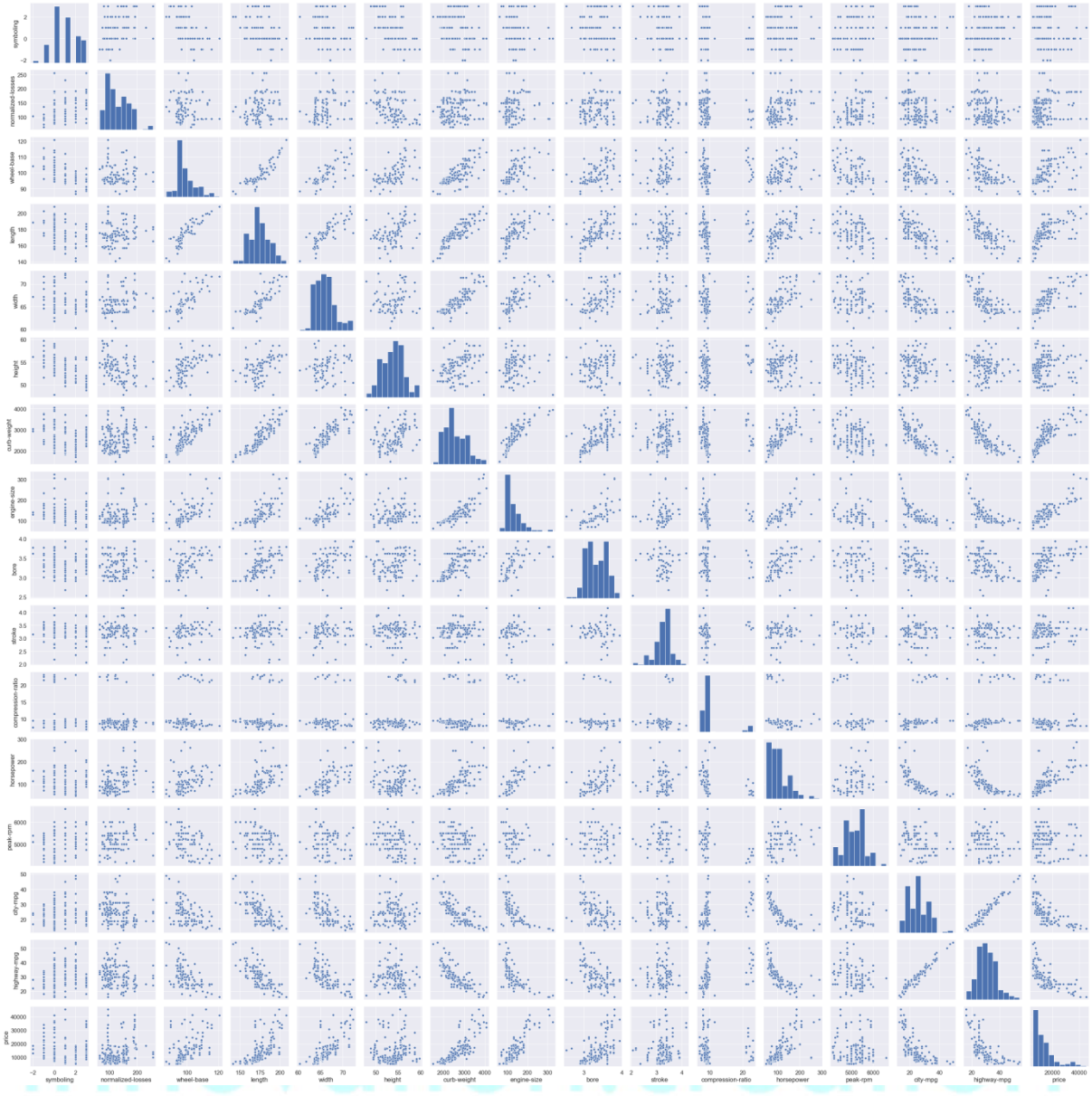
X-Axis : normalized losses

Y-Axis : price

A box plot or whisker plot shows the distribution of quantitative data in a way that facilitates comparisons between variables or across levels of a categorical variable. The box shows the quartiles of the dataset while the whiskers extend to show the rest of the distribution, except for points that are determined to be “outliers” using a method that is a function of the inter-quartile range.

Pair plot for the dataset:

```
In [56]:  sns.pairplot(AMD)
          plt.rcParams["figure.figsize"] = (15,7)
```



Conclusion:

The data set has 16 measurements, it creates a 16x16 plot

This dataset is often used in machine learning, because the measurements and classes provide an excellent way to distinguish classes.

Preparing Machine Learning model

Random forest Regressor Model:

- **Training The model:**

```
In [36]: from sklearn.ensemble import RandomForestRegressor

In [37]: rfc = RandomForestRegressor()

In [38]: rfc
Out[38]: RandomForestRegressor()

In [40]: rfc.fit(x_train,y_train)
Out[40]: RandomForestRegressor()
```

- **Testing The model:**

```
In [49]: print("the train set and the test set is")
         print(d_tree.score(x_train,y_train))
         print(d_tree.score(x_test,y_test))
         print("The R2_score is")
         print(metrics.r2_score(y_test,pred_rf))

         the train set and the test set is
         0.9999381242304528
         0.8936303952309881
         The R2_score is
         0.9350373038203226

In [3]: sol = round(0.9350373038203226,4)

In [4]: print(sol)

         0.935
```

ML model chart

Serial no	Ml Algorithm	R2_score
1.	Linear regression	0.8264
2.	Decision trees	0.8936
3.	Random forest	0.9350

Hurdles

- Time was given but managing with our examinations was difficult.
- I was little confused with the graphs because I don't have enough knowledge about cars and it's specifications and data concerned with the axis pattern
- Understanding Ml algorithm required to know statistics methods , learning those concepts improved me to understand the logic of algorithm.

Conclusion

Overall the project was pretty successful although we did face some problems.

Finding the values for the blank one was a stressful job, by finding the mean of data set we did filled the blanks. Then plotting the graphs for data set by using the python 3 libraries seaborn and matplotlib.

I have used three machine learning algorithms to predict the R2_score of the data set.

- For Linear regression I have got R2_score of 0.8264.
- For Decision trees I have got R2_score of 0.8936.
- For Random forest I have got R2_score of 0.9350.

BIBLIOGRAPHY

Douglas C. Montgomery, Elizabeth A. Peck, G. Geoffrey Vining

- This book gives a fairly standard introduction to simple and multiple linear regression, and then it devotes most of the text to dealing with their practical problems

Machine Learning With Random Forests And Decision Trees: A Visual Guide For Beginners.

- The book has a nice flow. All the concepts have been properly explained. I feel it's a very level zero kind of book. The fundamentals are way more important than the implementation.