MACHINE LEARNING AND DATA SCIENCE

ENCS5341.

ASSIGNMENT#1.

PREPARED BY:

SALEH KHATIB – 1200991.

SECTION 1.

DR: YAZAN ABU FARHA.

DATE: 30/11/2023.

Table of contents.

1-) Read the dataset and examine how many features and examples does it have? (Hint: you can use Pandas to load the dataset into a dataframe)

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 398 entries, 0 to 397
Data columns (total 8 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   mpg           398 non-null    float64
 1   cylinders     398 non-null    int64
 2   displacement  398 non-null    float64
 3   horsepower    392 non-null    float64
 4   weight        398 non-null    int64
 5   acceleration  398 non-null    float64
 6   model_year    398 non-null    int64
 7   origin        396 non-null    object
dtypes: float64(4), int64(3), object(1)
memory usage: 25.0+ KB
None
```

Fig1: dataset information.

As we see when we apply this code (print(data_set.info())) we get all information of our dataset as we see above.

We have 8 features with 398 examples and more info as we see.

2-) Are there features with missing values? How many missing values are there in each one?

Yes from Fig1 horsepower have 6 missing values and origin have 2 missing values.

3-) Fill the missing values in each feature using a proper imputation method

For horsepower any one of mean, median, or mode can be chosen.

So, I choose median which equal: 130.0.

But for origin, which is a string, we can take the mode for it, and the our mode is: USA

Now after we fill the missing values, we get this:

```
RangeIndex: 398 entries, 0 to 397
Data columns (total 8 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   mpg           398 non-null    float64
 1   cylinders     398 non-null    int64
 2   displacement  398 non-null    float64
 3   horsepower    398 non-null    float64
 4   weight        398 non-null    int64
 5   acceleration  398 non-null    float64
 6   model_year    398 non-null    int64
 7   origin        398 non-null    object
dtypes: float64(4), int64(3), object(1)
memory usage: 25.0+ KB
None
```

Fig2: dataset information after filling.

4-) Which country produces cars with better fuel economy? (Hint: use box plot that shows the mpg for each country (all countries in one plot))

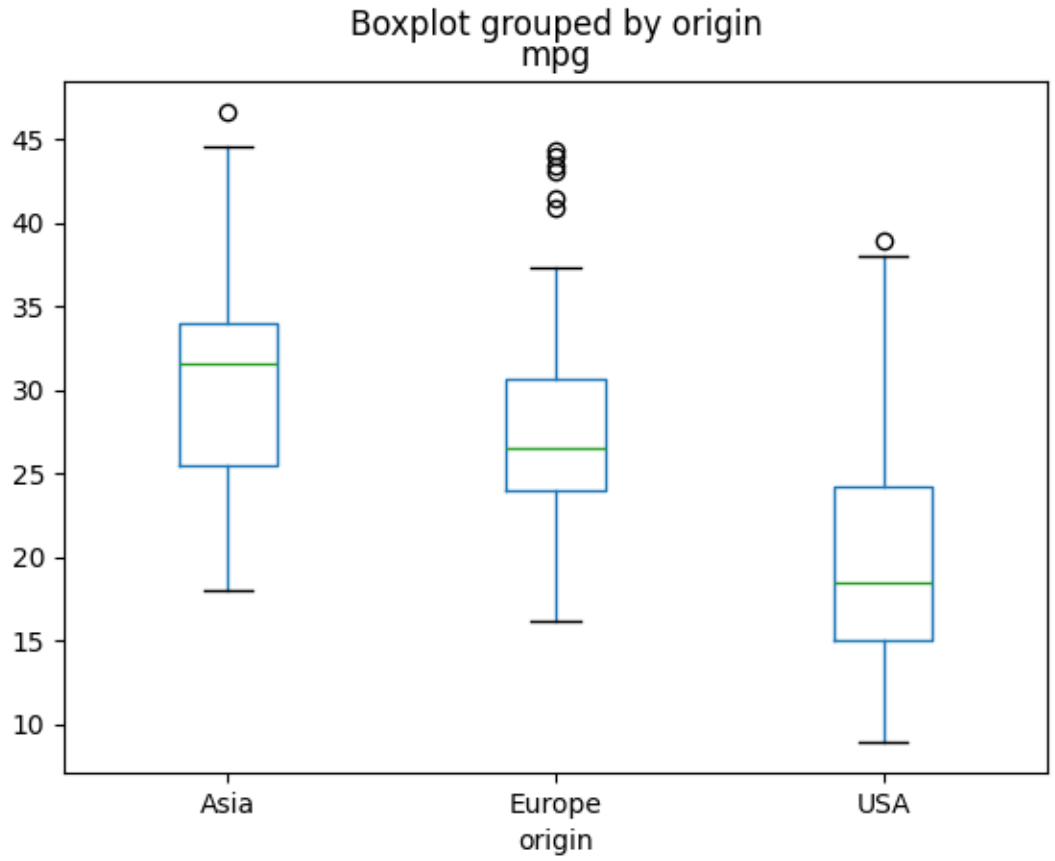

Fig3: box plot shows the mpg for each country.

As we see, Asia is the best one.

5-) Which of the following features has a distribution that is most similar to a Gaussian: ‘acceleration’, ‘horsepower’, or ‘mpg’? Answer this part by showing the histogram of each feature.

First, we will show the gaussian distribution graph:

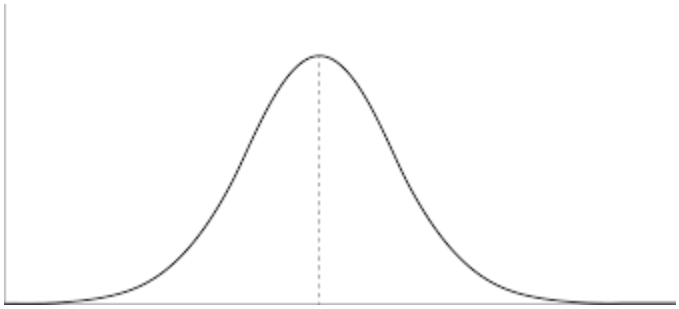

Fig4: Gaussian distribution graph.

Now we will see the histogram:

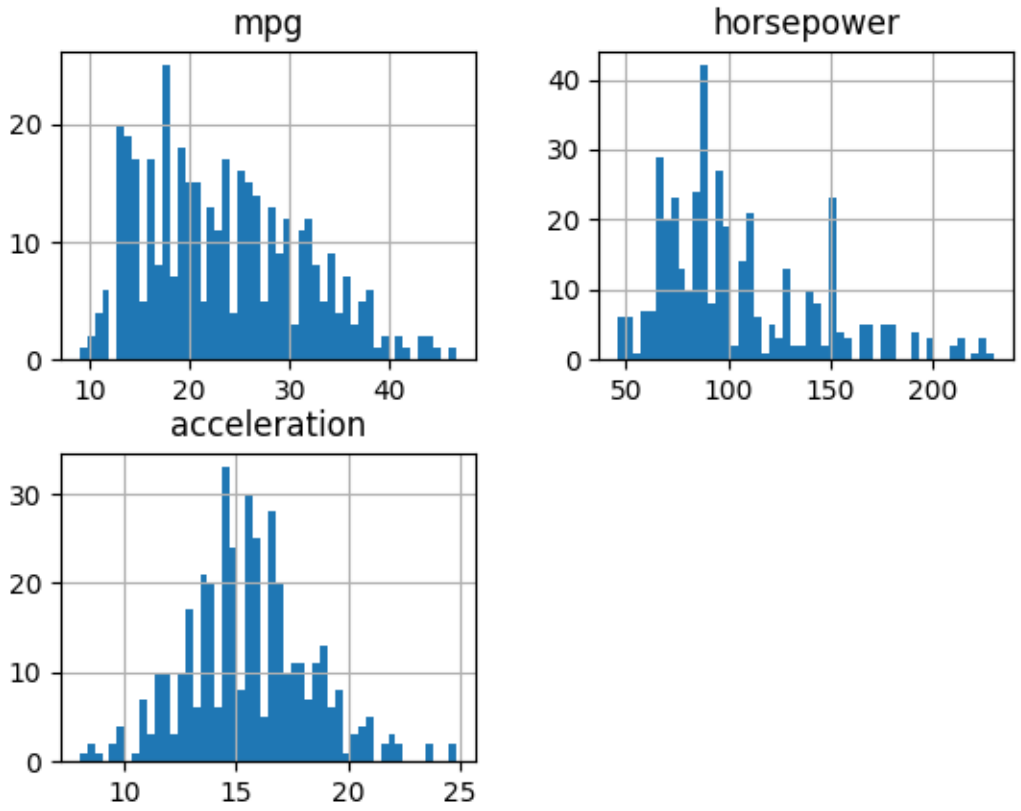

Fig5: histogram graphs.

As we see the most similar one is acceleration.

6-) Support your answer for part 5 by using a quantitative measure.

For this part we will apply Shapiro-wilk test as we see:

```
for mpg:  ShapiroResult(statistic=0.967965841293335, pvalue=1.1833407853600875e-07)
for horsepowr:  ShapiroResult(statistic=0.909908652305603, pvalue=1.1939947525151963e-14)
for acceleration:  ShapiroResult(statistic=0.9923787713050842, pvalue=0.039872437715530396)
```

Fig6: Shapiro-wilk test.

To say if a column come from gaussian distribution the pvalue must be under 0.05 as we see in acceleration.

7-) Plot a scatter plot that shows the ‘horsepower’ on the x-axis and ‘mpg’ on the y-axis. Is there a correlation between them? Positive or negative?

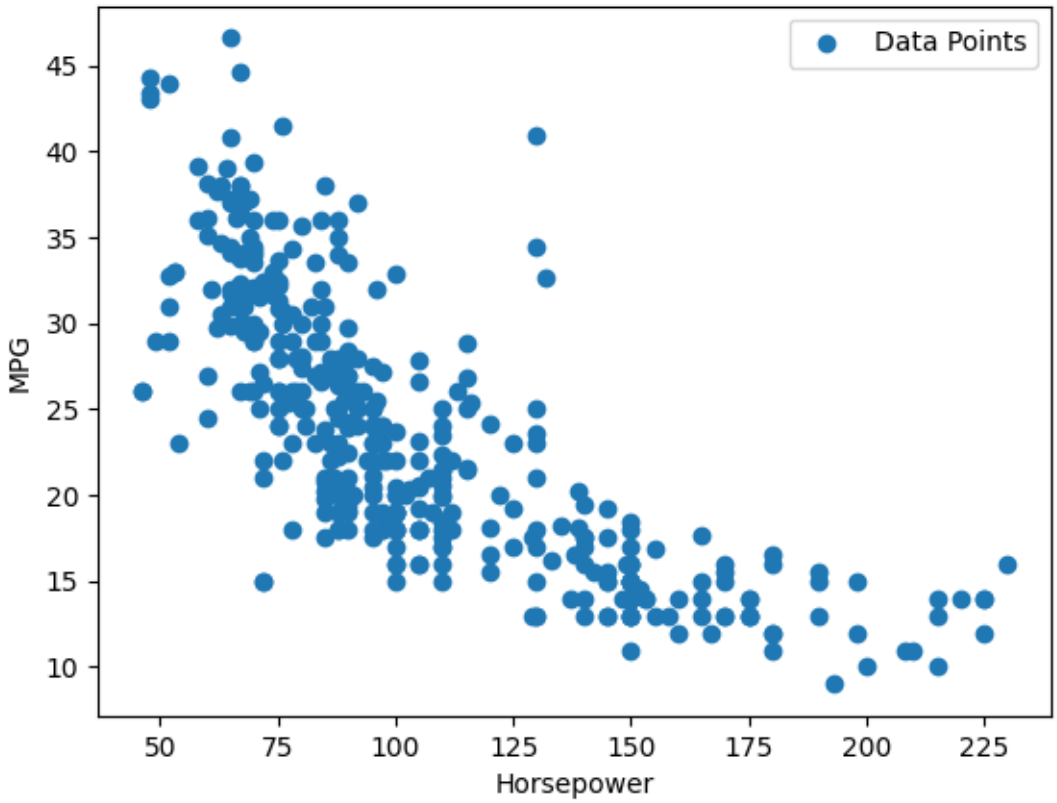

Fig7: scatter plot.

As we see there is negative correlation.

8-) Implement the closed form solution of linear regression and use it to learn a linear model to predict the ‘mpg’ from the ‘horsepower’. Plot the learned line on the same scatter plot you got in part 7.

When we apply linear regression, we get this:

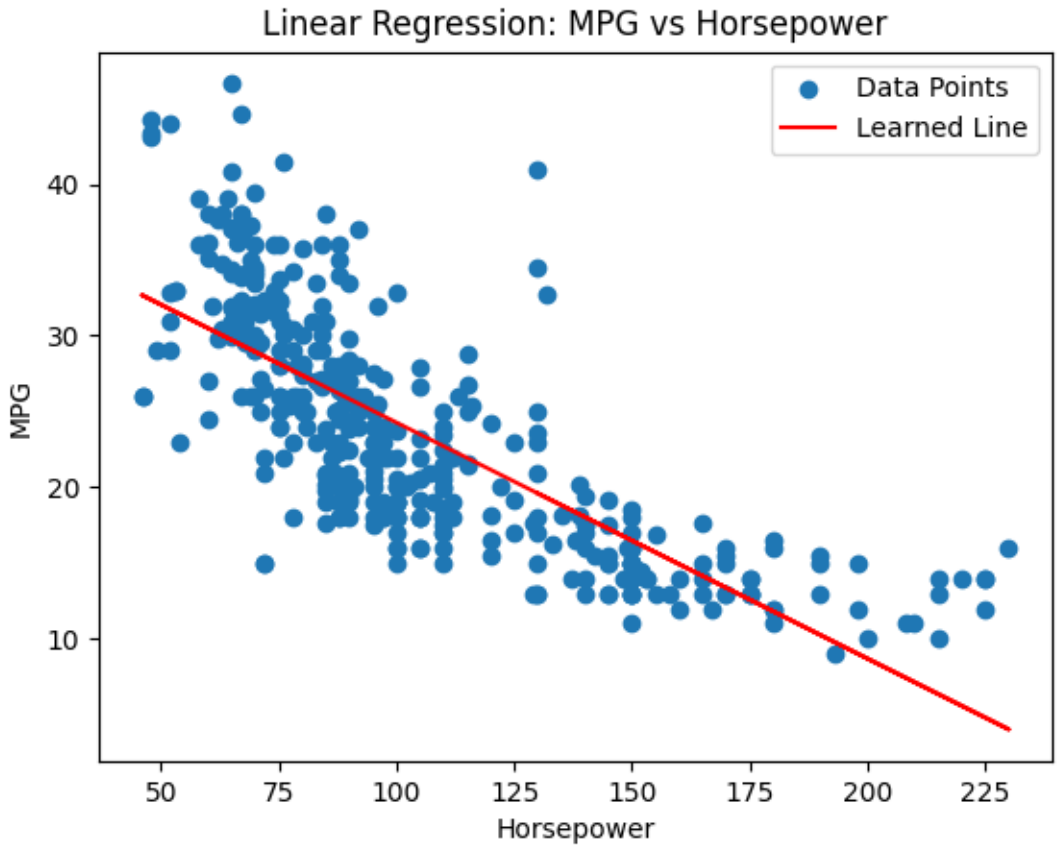

Fig8: linear regression.

And we get these weights: [39.83239818, -0.15562385].

We know F(X) = w1X + w0 = -0.155X + 39.83.

9-) Repeat part 8 but now learn a quadratic function of the form.

When we apply nonlinear regression, we get this:

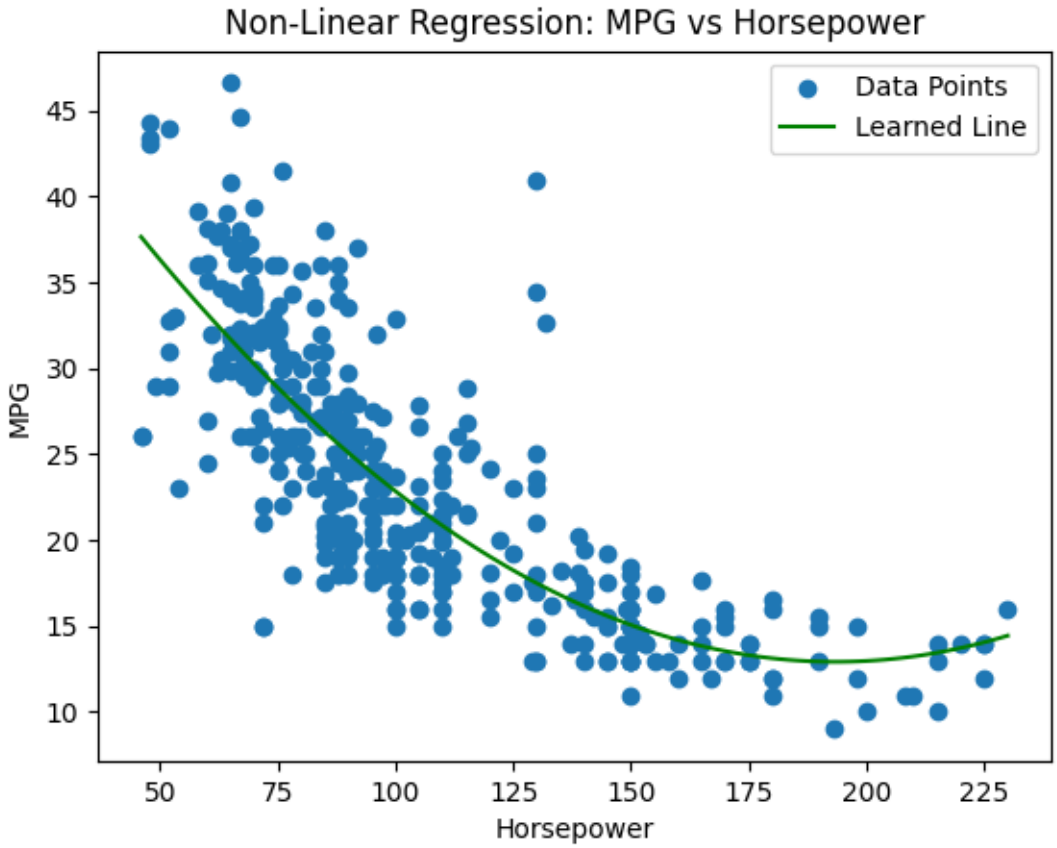

Fig9: nonlinear regression.

And we get these weights: [5.54924942e+01, -4.40179210e-01, 1.13781661e-03].

10-) Repeat part 8 (simple linear regression case) but now by implementing the gradient descent algorithm instead of the closed form solution.

When we apply gradient descent algorithm for 1000 iteration and alpha = 0.05, we get this:

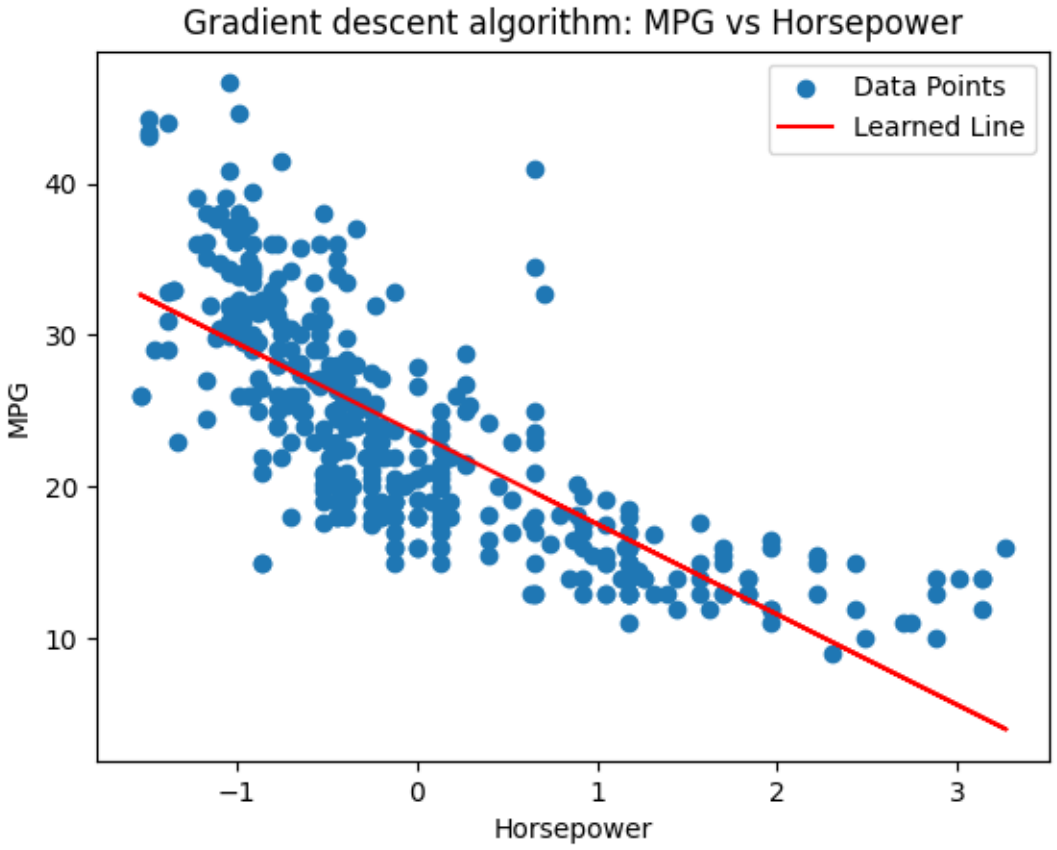

Fig10: gradient descent algorithm.

And we get these weights: [23.51457286, - 5.95693826].

We know F(X) = w1X + w0 = -5.95X + 23.51.