# Assignment 1

**Question 1:**

Using matplotlib, I plotted the x_train and the y_train data. On visualizing the plot, it was clear that it was not a linear function, but rather a polynomial function. Answering questions on problem1:

1. The clues that I got on plotting the data, was that I was able to notice that the variables had a polynomial relationship, and not a linear relationship. There was no straight line in the plot, to prove that it was a linear relation.

2. a) The relationship is not linear.

b) Yes. To want the line to have non-linearity to fit the data, we should apply basis functions to it.

i. We can engineer these features by using a basis function.
ii. We can try with different orders of x_train. First, start with x_train data. Then, move towards x_train^2 basis function. Check if the model got better or worse. Again try x_train^3 basis function and observe the graph. Continue this process until you find the right amount of basis functions to employ. You can find the optimal amount, by observing the cost function associated with the basis functions.
1) I plotted x_train data from first order to eleventh. The model gradually kept getting better until the sixth order. After that, the cost began to increase.

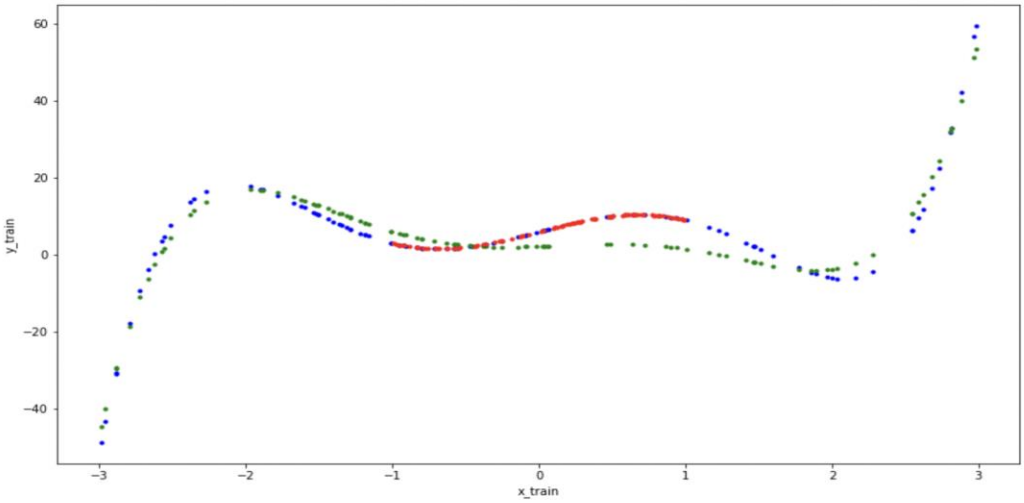
To summarize, I created a basic linear regression model using gradient descent. I added features to it using the basis function from first order to eleventh. The fit gradually kept getting better until the sixth order. Then the cost began to increase.

After training, I observed the following values. The function that gives the relationship between x and y can be written as:

```
Estimated Weight: [[ 1.4209838   1.74609314 -4.36017364 -0.16242367  0.67777258]] -- Estimated Bias: [2.22130366]
```

$$y = 0.68x^5 – .16x^4 – 4.36x^3 + 1.75x^2 + 1.42x + 2$$

The graph was observed as below:




Green is the predicted model. Red represents the test data, while blue is the data from training. The cost or error turned out to be 387.212 LMS, which I believe is a good fit.

**Question 2:**

1. The average least squares error for the given data was calculated to be 972.38.

2. Garages seem to have the most effect on the final value. I figured this out in two ways. First, taking each weight and raising it to the power of the respective order reveals which coefficient carries most weight. Second, is through deduction. All other factors have a highly varying interval. Garage on the other hand, only has a range from 0 – 2. This carries a lot of weight since its range is very small compared to other factors. No, you cannot use only this feature to predict the price of the houses.

3. The age of the home has the least effect on the final value. This is because it has the smallest weight 0.7. Raising it to the power of the order basically gives you a value of 0. Removing this feature didn't seem to have that much impact on the performance. However, it is always a good idea to have more data and variety in the data sets.

**Question 3:**

1. Basis function is not required while using the locally weighted approach.

2. In question 1, we had a fixed set of parameters, and we tried to find the optimal values for those parameters (theta). After finding the parameters, we can completely erase the data, and use our model with only the existing parameters. However, in question 3, i.e., non-parametric algorithm, we need the

data and the parameters until we make predictions. This may not be optimal for computer memory when you have a large dataset.