Single Variable Regression

# Problem Statement

The single feature data set should be fit to a linear model. The data should be trained and tested using the Linear Regression model. Based on the fit, the labels have to be predicted and the predicted labels are compared to the actual values to measure the accuracy of the model. The fit represents finding the coefficient of the feature computed with training data which is then used to predict the unknown labels from the test data. The 10-fold cross validation has to be done to analyze the performance of the model. The accuracy of the model is analyzed with the training and testing error calculated by Mean Square Error. The fit and predict implemented should be compared with the readymade Python functions.

The data is split into different subsets and each subset is extended to a certain degree of polynomial and the performance of the model is evaluated through cross validation. The effect of change in the size of the training data set should is observed.

# Problem Solution

The solution to find the theta or the coefficient of a feature is done by implementing Linear Regression. The linear regression modeling is used to find the relationship between a scalar dependent variable also known as labels Y and one or more explanatory variables or also known as features which is denoted by X. The features and the labels are in the form of vectors. The predictor function is used for modeling. The predictor function should solve for the theta or coefficients from X and Y. The number of elements in the theta or the coefficients should correspond to the number of features.

Let X, Y be the feature vector and the label vector,

Θ = (XTX)-1Y

Solving the above will provide the coefficients which are also in the form of vector.

To find the unknown labels,

Y’ = Θ-1X’

Where X’ is the testing data.

The linear model is then extended to polynomial model with certain degrees and the same solution provided above is used to compute the result and the accuracy is measured.

# Implementation Details

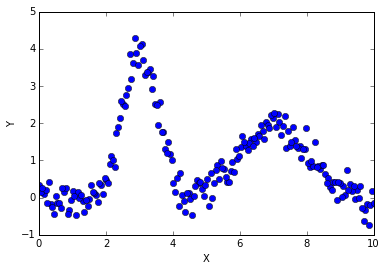
There are two ways of solving the single feature variable. The first one is the form the 2x2 matrix by sum up the values in the feature and a 1x2 matrix by summing up values in Y. the second one is by adding 1 in the first position of all rows and solve it with the above said method. Both will give the same result. Solving by the second method will be easier as the single function can be used for polynomial model too.

In the code the function “find\_theta\_linear” will solve by the first method and “find\_theta” will solve for coefficients by the second method.

I have implemented the code in ipython notebook. The filename has to be mentioned and in the tool bar option “Cell” -> “Run All” will implemented the whole file and the results will be printed.

# Results and Discussion

1. Plotting the data: file "svar-set4.dat.txt"



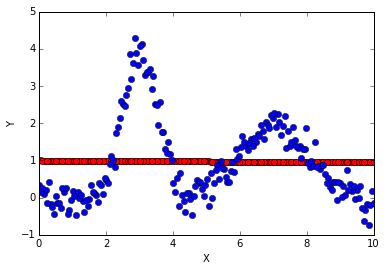
1. Fit the model and compute training and testing error

Theta [[ 0.99777523]

[-0.00537298]]

|  |  |
| --- | --- |
| MSE Train Error Mean | MSE Test Error Mean |
| 1.20020250224 | 1.21228704771 |

The training and the testing error is calculated using mean square errors. The plot of the regression model is as below,



The linear regression on a single feature shows a plot of straight line. This shows that the linear model is a plot of single line.

1. Comparing to Python readymade function

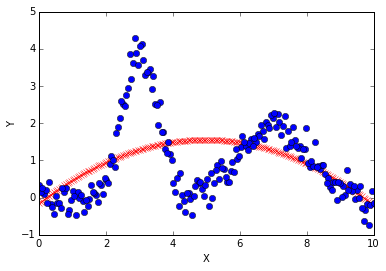
The mean error when evaluated with Python in built function turns out be the same as what I have implemented manually.

|  |  |
| --- | --- |
| MSE Train Error Mean | MSE Test Error Mean |
| 1.20020250224 | 1.21228704771 |

1. I split the data in four and have raised each sub part to polynomial of degree 2,3,4,5 and the cross validation for each subset has been done.

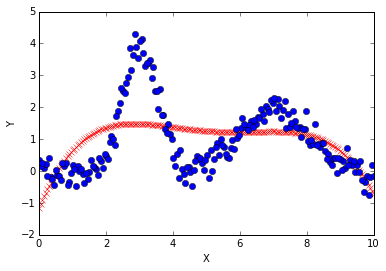
**Degree: 2**

|  |  |
| --- | --- |
| Train Error Mean | Test Error Mean |
| 1.09420984139 | 1.14892091518 |

****

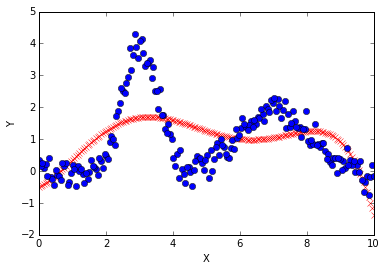
**Degree: 4**

|  |  |
| --- | --- |
| Train Error Mean | Test Error Mean |
| 0.667960023482 | 0.921087472516 |

****

**Degree: 5**

|  |  |
| --- | --- |
| Train Error Mean | Test Error Mean |
| 0.878769046682 | 1.08430321188 |

****

**The polynomial model I would choose is the one with degree 4.** Out of all the degree from my experiments the data when raised to polynomial of degree 4 has lowest testing and training error. Hence it would yield better result than other model.

1. Reducing the amount of Training data

Mean error when the training data is some percentage of original data

Percentage Mean\_Error

25 1.70496717318

90 1.72445473762

75 1.74098476595

50 1.74379150188

The percentage shows the percentage of actual data used for training.

# References

<https://en.wikipedia.org/wiki/Linear_regression>