Prateek Agrawal-A20358932

MACHINE LEARNING ASSIGNMENT 1

Table of Contents

[Problem Statement 3](#_Toc443587679)

[Proposed Solution 3](#_Toc443587680)

[Implementation Detail 4](#_Toc443587681)

[Results and Discussion 5](#_Toc443587682)

[References 10](#_Toc443587683)

# Problem Statement

We are trying to solve several problems here.

* For Single Variable Data Set
  + We compare the ready made linear regression model provide in the Scikit learn package of python with that build by us and check how better is our prediction on the basis of the training and the testing error.
  + We fit our data set with different degrees of no linear polynomial function and find the best fit for each data set.
  + We observe the effect of reduced data and see if the degree for the best fit changes from that of the above.
* For multiple Variable Data set
  + We try here to find the best model for data set with multiple features by comparing the testing error for different combinations of the features.
  + We compare the explicit solution from the above with the iterative solution of finding the optimal theta in terms of the testing error and the time taken for both the methods.
  + We fit a model using Gaussian Kernel to solve the dual problem and compare it with the primal problem in terms of testing error

# Proposed Solution

We are using the ready made linear regression algorithm provided in the Scikit learn package of python along with several manual algorithms to fit a predictive model.

The aim of all the algorithms used are to correctly predict the value of the last column of a given data set with single or multiple variables. For example, if a data set has 5 columns, our aim is to use the first 4 columns of each row to predict the 5th column of the respective row. This is done by first training the data and finding an optimal parameter called theta which when multiplied by the row values yield the predicted value.

For example, if X0, X1, X2, X3, Y are the first row if a data set. We find Q0, Q1, Q2, Q3 such that X0\*Q0 + X1\*Q1 + X2\*Q2 + Q3\*X3 = Y.

Algorithm follows the following steps:

* We define predicted Y as , were theta is the parameter we predict and Z is the feature matrix.

Example,

1. X11 X12 X13

1 X21X22X23

Z =

* To find the optimal value of theta we define an objective function 2 from 1 to m, were m is the number of examples we have in the dataset. Here m = 2.
* Now, we find the gradient of the above function and compute it to zero to get the optimal value of theta.
* Putting the value of theta in the first equation gives us the predicted value of Y.
* If we use the feature matrix as given in the data set we call it linear fit, but if we build a new matrix Z which has the combination of the features, we call it non linear fit.
* We use the above approach to fit different models and compare them with the ready made linear regression model in terms of the training and testing mean squared error.
* We call this solution as the primal problem and explicit solution.
* Another non trivial approach that we use here is the iterative approach by gradient decent.
* The objective function stays the same. But instead of equation its gradient to zero, we use the following iterative approach to find .

1 i-1  –

* We start by guessing the value of first and then keep on changing it iteratively until the value of stops changing.
* Here, is called the learning rate and is user defined.
* An alternative of solving non linear equation we use dual problem and convert the feature vector in higher dimension and represent as the product of . Where becomes our new parameter and X is the feature vector.
* is predicted as the product inverse of Gram matrix and Y.
* Once predicted we can use it to find the predicted Y values.
* Gram matric is a m by m matrix with each element as the value of similar between feature vectors.
* We used Gaussian Kernel to compute this similarity.

K(X, Y) = exp(-(X-Y)2/(2\*sigma\*\*2))

# Implementation Detail

The code attached with this report is written in such a way that is detects whether the data set passes in single variate or multivariate. And then performs the required operations.

It works for all the synthetic data set available on the course website and a real time data set on airfoil self noise from UCI database and compare the results of the varies algorithms on the mean square error and the time taken by each algorithm.

After the URL is read the dataset is converted into X and Y matrices with Y containing the last column and X containing all the other columns. Value of X is checked to see if the feature vector is multivariate or single variate.

If single variate data of found.

* Linear model in fit and the testing error is compared with that of the ready made python model.
* Same data is then fit with different degree of polynomials and the best model is selected (one that produces the minimum error).
* Effect of reducing the size of data was also observed.

If multivariate data Is found.

* Different combination of feature vectors is used to increase the dimension and find the one with the lowest mean square error.
* Different algorithms such as iterative approach and kernel method are evaluated and the best parameters for each approach is detected.

The most difficult issues I faced was of finding the parameters learning rate and sigma. It took a lot if iterations to reach to the optimal solution.

Different data sets used are:

* svar-set1.dat, svar-set2.dat, svar-set2.dat, svar-set3.dat
* mvar-set1.dat, mvar-set1.dat, mvar-set1.dat, mvar-set1.dat
* UCI data set on airfoil self noise

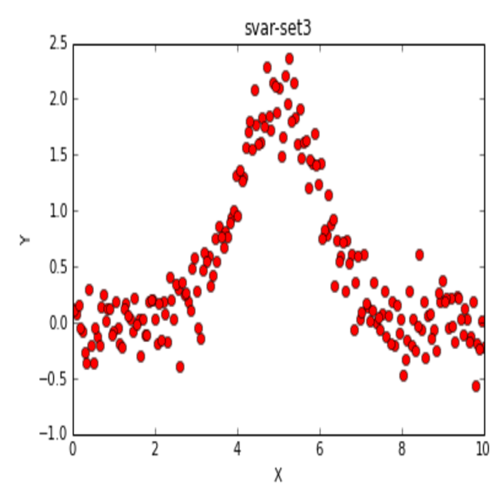
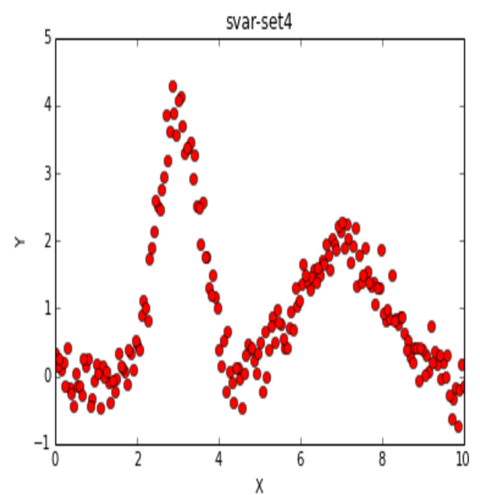
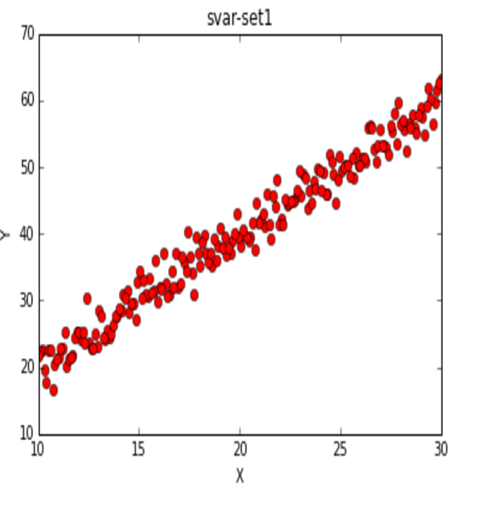
(URL:https://archive.ics.uci.edu/ml/machine-learning-databases/00291/airfoil\_self\_noise.dat)

# Results and Discussion

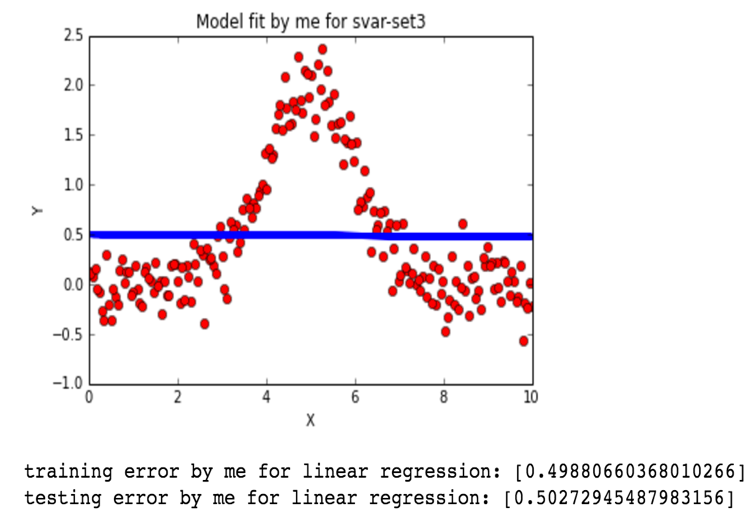
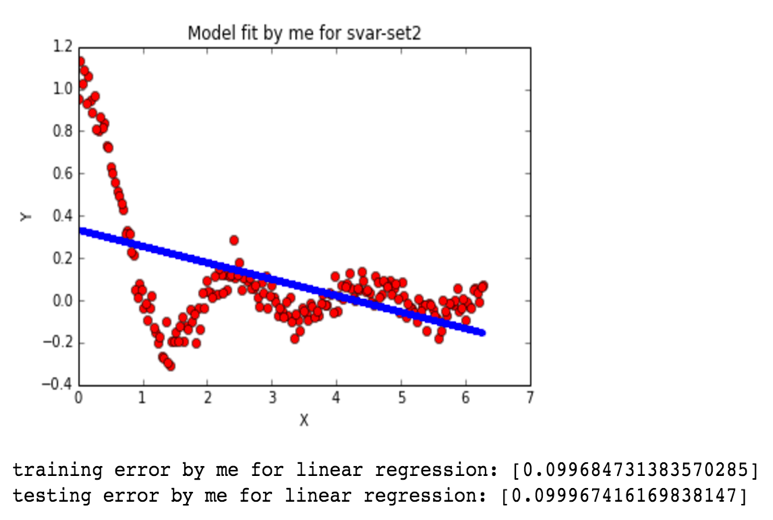
Following are the operation details on single variate feature examples.

* Each model passed is plotted to gain some knowledge on the complexity of the model.

For example:

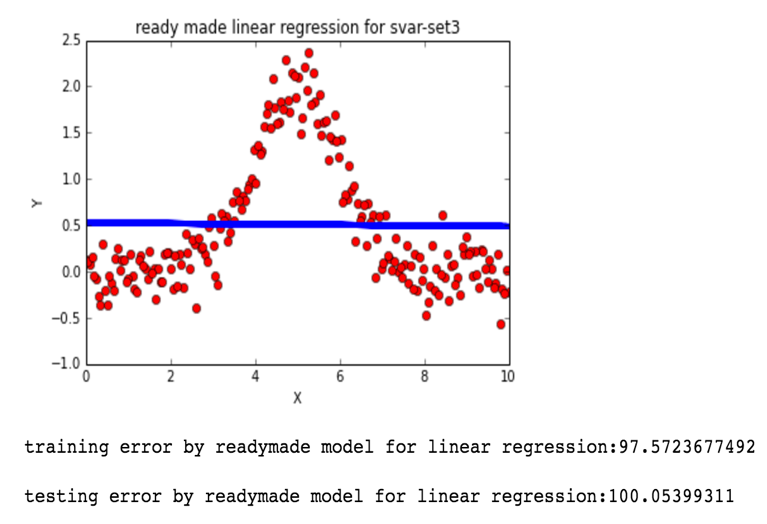
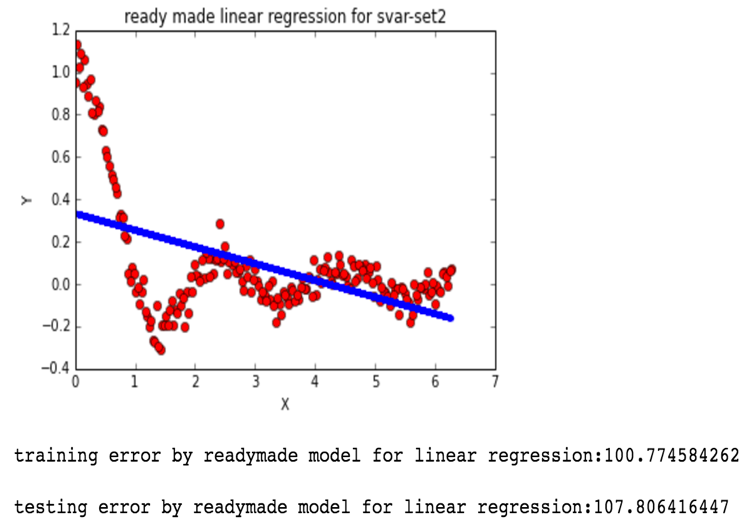


* A regression model was fit on the data set. Training and testing error was computed and compared using the mean square error.



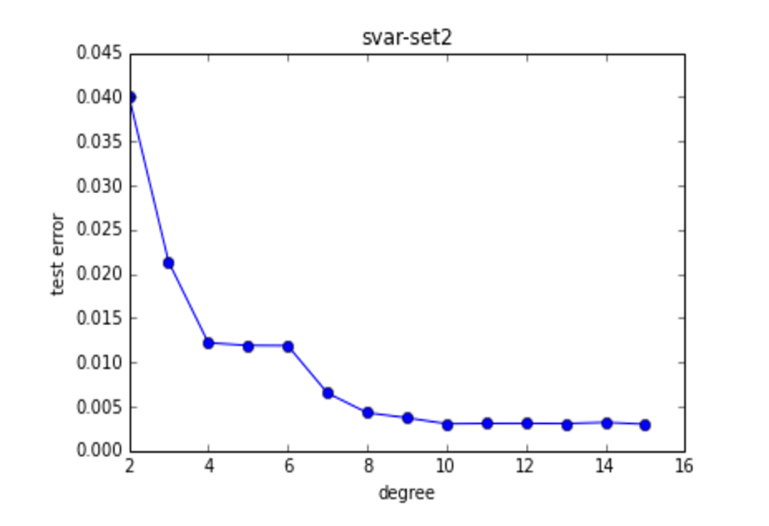
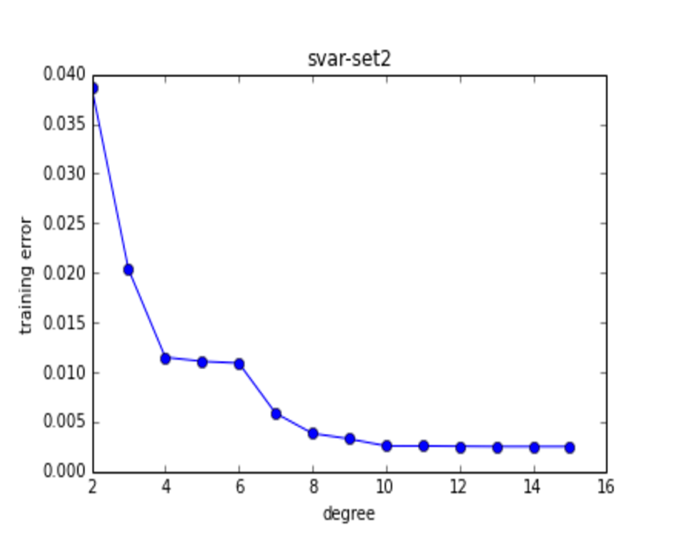
We can see here that there is not much difference the training error and testing error. This shows that the model fit my us works well.

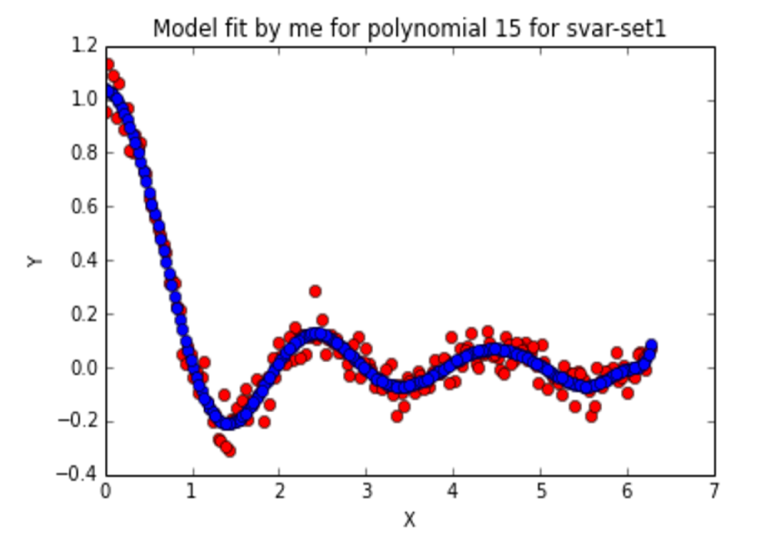
* Next we will compare the above performance with that of the ready made linear regression in Scikit learn package. And check how accurate were we in our prediction.



We see here that ready made linear regression had a very high testing and training error in comparison to the model fit by us.

* Now, we work towards decreasing these errors to more extend by fitting a non linear model on these data set. We do this by fitting the model for each polynomial degree from 2 to 15 and finding the one with the least testing error.

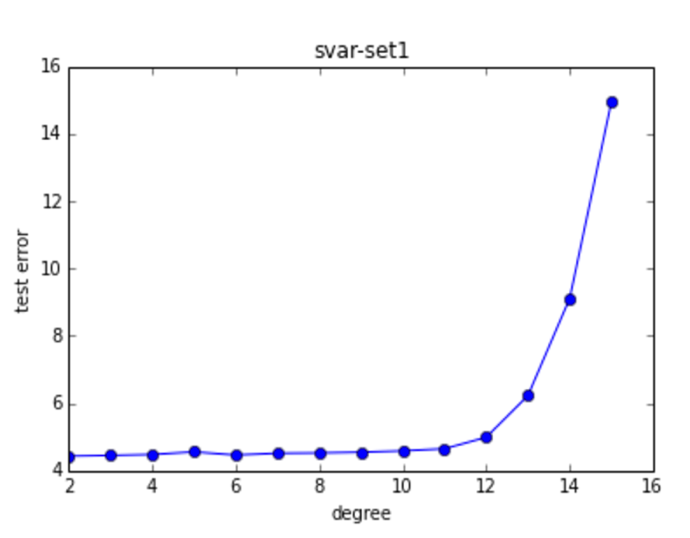




From the above three graphs we can see that as we increase the degree of the polynomial the training and testing error keeps on decreasing.

It was found that testing error was minimum with polynomial degree 15 with a value of 0.003. This value is far less then that we obtained from linear fit. Hence, we conclude that non linear fits the model better.

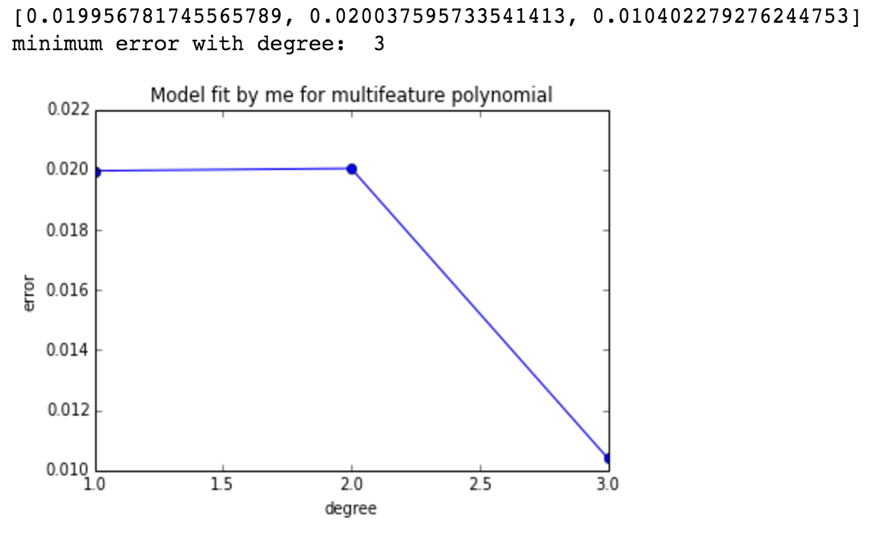
However, it is not true for all the cases. Choosing very high polynomial degree for some data set may over fit the data and produce high error. For example, if we use the first data set to fit higher polynomial we will see that the testing error increases exponentially.



* We also observed what happens to the testing error if we decrease the amount of data.
* Taking only 50 percent of the data we see that the testing error does not change much in case of linear model. But testing error shoots high for non linear model. Concluding that the more data we have, the better our model will be.

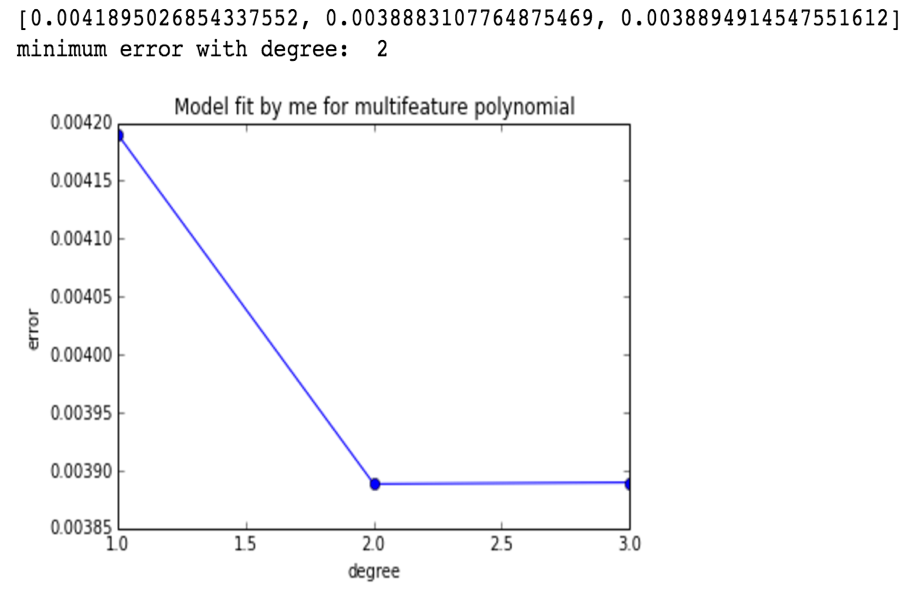
Following are the operations carried out for multivariate datasets.

* Multivariate dataset was taken and fit to higher dimension for the combination of 1, 2 and 3. For example, if number of features in X was 2. Then three different Z were prepared with dimensions 2,4 and 9.
* All these Z were trained and tested using 10 cross fold validation and the dimension with the minimum testing error was recognized.



We see here that dimension with polynomial degree of three gives the minimum error of 0.010422.

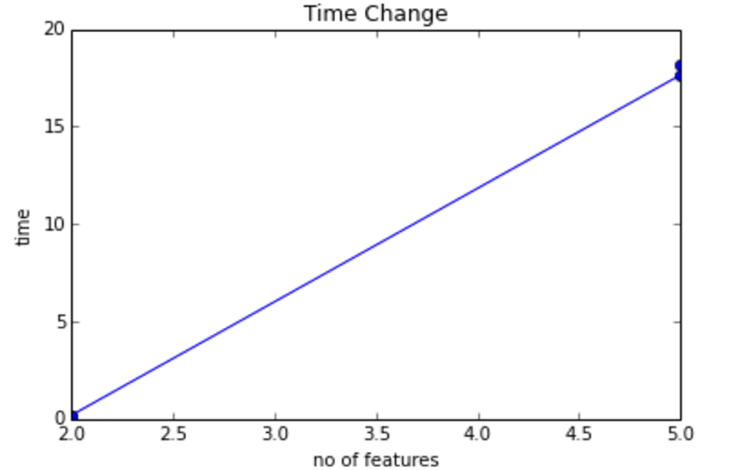
Next we checked if this trend stays the same of all the data sets.



In the above example the number of dimensions with degree 2 is 25 and 3 is 125. But we see that dimension 25 has slightly less error than that of dimension 125.

Concluding that as the dimension increases the testing error decreases but may eventually lead to overfitting and increase the error. So we need to find the optimal dimension by testing between different dimensions and then train the data. We cannot choose one dimension for all the data and use it. Each data set will have its own optimal dimension. Also linear dimension not always produces more error than that of the higher dimensions.

* Time taken by the algorithm increases rapidly as the dimension increases.



We see that the time taken by the algorithm increases from 0.2 seconds to 18 seconds as the number of features increase.

* We also tried solving the regression problem with the help of the iterative approach and computed the time taken by the algorithm along with the optimal value of learning rate.

Starting with a two data set with two features each we computed the optimal learning rate and see if the leaning rate depends on the data or the no of features.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Learning rate | Dataset1 with 2 features | Dataset2 with 2 features | Dataset3 with 5 features | Dataset4 with 5 features |
| 0.1 | False | False | False | False |
| 0.01 | False | False | False | False |
| 0.001 | False | False | False | False |
| 0.00001 | True with error 0.29 | True with error 0.013 | False | False |
| 0.000001 | True with error 0.59 | True with error 0.15 | False | False |
| 0.0000001 | True with greater error | True with greater error | True with error 0.65 | True with error 0.039 |

In the above table True says that the value of theta conversed and False says the opposite. We summarize our finding on different values of learning rate(LR). We found the value of LR depends on how complex the data set is. As we can see here that data set with two features converges at 0.00001 but the data set with 5 features do not.

Further increasing the LR for already converged dataset increases the testing error. Higher the number of features, lower the value of LR is required.

Also, if we compare iterative approach with the traditional approach we see that the optimal LR gives almost same testing error but takes significantly less time.

As show by the table below.

|  |  |  |
| --- | --- | --- |
| Data set | Traditional approach | Iterative approach |
| 2 features | 0.20 | 0.028 |
| 2 features | 0.20 | 0.053 |
| 6 features | 18.779 | 0.32 |
| 6 features | 18.9 | 0.57 |

Iterative approach finds the solution very fast, but before running this we need to find the optimal LR for that particular data set which will penalize us with significant time.

* Kernel method is used next to see if dual problem of linear regression is efficient than that of the primal problem.

Biggest issue of the kernel problem was of predicting the value of sigma. Which I was not able to predict correctly with several iterations.

Taking the value of sigma as 3 gave me a large error. Plus, the time taken by kernel method to build the model and do tem cross fold to predict was around 100 seconds. With this I conclude that if we find the correct value of sigma we might be able to use kernel method to increase the dimension of the feature matrix and fit a linear model in the higher dimension, which will be same a fitting non linear model in lower dimension.

# References

* Stackoverflow.com
* Pythonprogramming.net
* Coursera.com
* Element of statistical learning.