CSE574 : Introduction to Machine Learning(Fall 2014)

**Project 1: Linear Regression with Basis Functions**

Naveen Narayan

UB ID : nnarayan

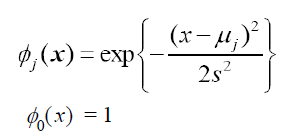
UB Person # : 50134647

**Aim :** The aim of this project is to, given 46 relevant features, predict the corresponding relevancy scores. The hypotheses used to predict the output of a new sample is derived using several supervised machine learning approaches to the task of linear regression.

**Model :** The Closed Form Solution and Stochastic Gradient Descent algorithms are implemented to derive the weight parameters of the hypotheses function.

1. **CFS :**

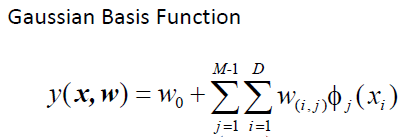
The design matrix 'phi' is derived using the below formula :



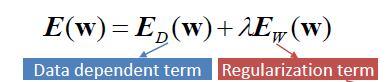
The parameters for the closed form hypotheses is derived using the below formula :

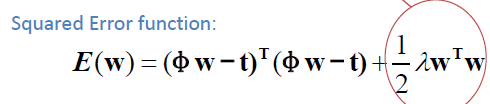


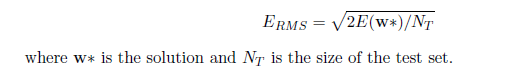
The closed form solution is derived using the below formula :



The root-mean-square error is calculated using the below formulae :







1. **Stochastic Gradient Descent**

The Stochastic Gradient Descent solution employs the below formulae :

The parameters for the Stochastic Gradient Descent hypotheses is given by :



Calculate ERMS using the above formulae.

If ERMS increases, decrease n ( eta ) by a factor of 2.

**Implementation :**

The data provided is cleaned up and loaded into a matrix.

**CFS :**

1. Loading data :

80% of the data is used to train the the model while the remaining 20% of the data is split for validation and training purposes.

The training data is divided into 2 matrices, TrainingData ( 59699 X 46 ) and and TrainingTarget ( 59699 X 1).

The validation data

1. Finding Mu :

The dimension ( number of columns ) of Mu depends on the number of features provided in the data set ( in this case, 46).

Fetch 200 random rows from the data set and find the mean for each column. This Mean vector ( 1 X 46) is the mean vector.

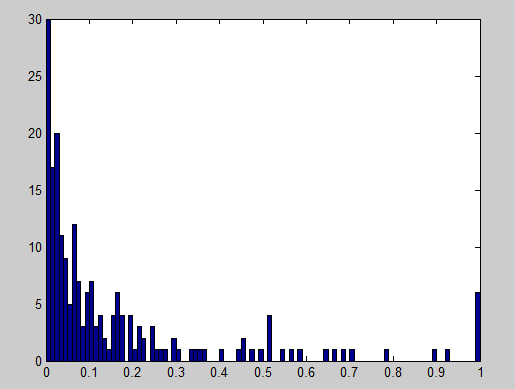


Fig 1 : Plot of values for 1 feature( column) for 200 sample data rows.

Mean = 0.16

1. Creating the Design Matrix :

The first column for all rows of the design matrix will be assigned to 1.

By subtracting each row with this Mean vector and applying the Gaussian function on this row vector, a scalar is generated. Repeat this step for all the rows in the dataset.

This process is repeated M-1 times to generate a N X M deimensional design matrix.

N - # of rows in the given data set

M – Model complexity

1. Finding W ( Paramaters ) :

The parameter vector ( M X 1 ) is found by matrix multiplications using the formula mentioned in the above section.

1. Calculate ERMS :

Using the hypotheses function, and the paramaters ( derived from the above step ), predict the output value for each of the row.

Calculate the difference in the predicted value and the observed value and hence find ERMS.

1. Validation :

If the ERMS is lesser than the acceptable limit, perform validation by calculating the ERMS for Validation data. Use the hypotheses derived from the above steps to predict the output for the data in the validation set.

If the difference in the ERMS values calculated for the Training data and Validation data is not significant, we assume the hypothese to be validated.

1. Testing :

Predict the output for the Testing data and calculate ERMS.

If the ERMS value is acceptable, the implementation is successful.

1. Critical observations :
2. For lower values of M, the ERMS value is large. This is due to under-fitting of data by the generated hypotheses. As M increases, the hypotheses fits the data better as the degree of the freedom of the curve fitting the data increases.

This can be illustrated using the below graph.

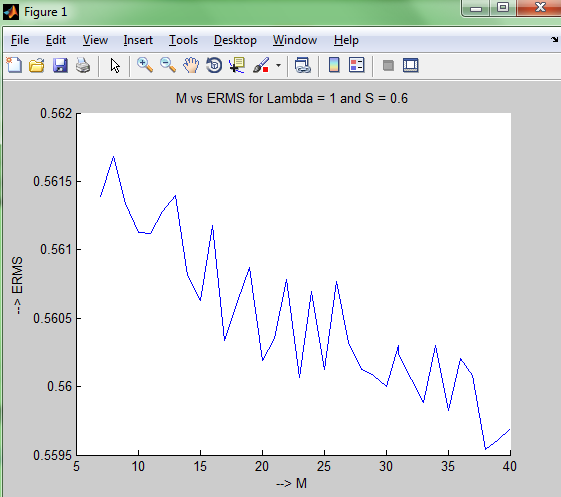


Fig 2 : ERMS vs M

1. As seen from the above graph, as M increases, the hypotheses fits the data better. Although this might seem appropriate, higher values of M imply lower probability of predicting the output for new data.

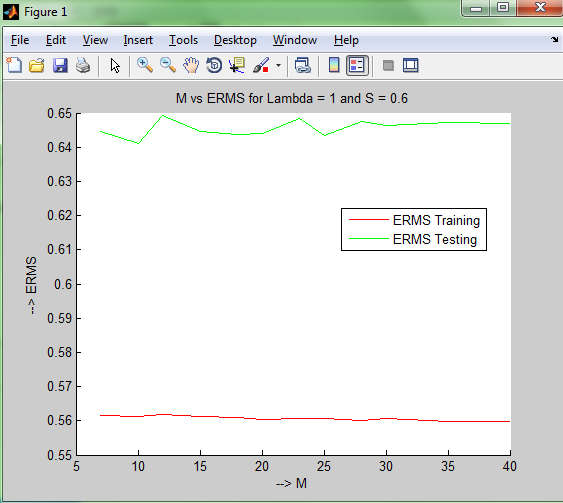


Fig 3 : ERMS for Training, ERMS for Testing vs M

This feature can be observed more prominently by considering only two values for M.

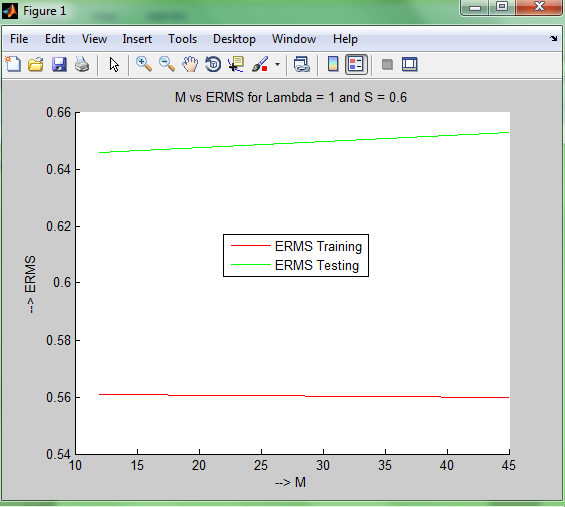


Fig 4 : ERMS for Training, ERMS for Testing vs M ( M = 12, M = 45 )

As M increases, Lamda ( the regularization paramater ) increases to resolve the issue of over-fitting.

For ex : For M = 07, optimal Lambda = 01

For M = 12, optimal Lambda = 10

For M = 40, optimal Lambda = 20 ( the largest value that Lambda could take )

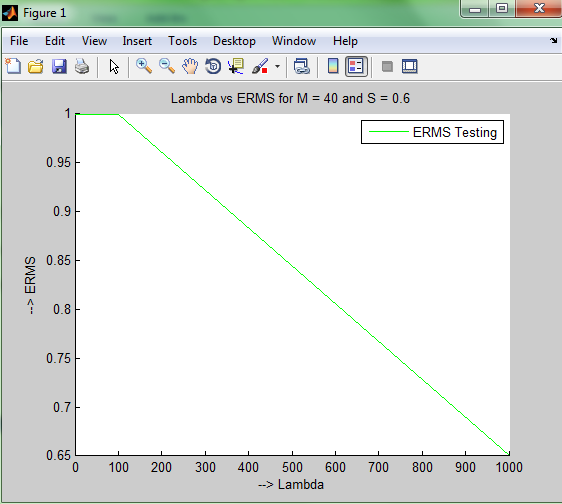
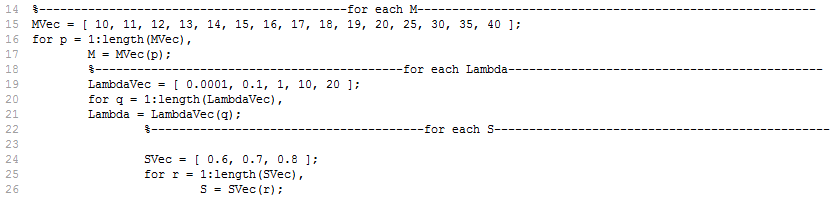


Fig 5 : Lambda vs ERMS for a large value of M ( M = 40 )

1. Tuning :

For a given value of M, we tune the values of S and Lambda.

This can be seen from the below code snippet :



We find the optimal M by choosing M which gives us the least ERMS.

=====================================================================

**Stochastic Gradient Descent :**

1. Loading data :

Same as CFS.

1. Finding Mu :

Same as CFS.

1. Creating the Design Matrix :

Same as CFS.

1. Finding W ( Paramaters ) and calculating ERMS:

Initialise W to a M X 1 dimensional vector with random values ( or some convenient values ).

For each row of data, calculate W using Wprev ( value of W in the previous iteration ).

Calculate ERMS using the new parameters (W).

If ERMS is greater than ERMSprev decrease n ( eta ) by a factor of 2.

If the difference between ERMS and ERMSprev is less than a pre-defined acceptable limit, stop the iteration procedure and accept W to be the optimal paramaters.

Matrix multiplications are used to compute W from Wprev.

1. Validation :

Using W, calculate ERMS for the validation data set. If the difference between ERMSTraining for the training data set and ERMSValidation is not significant, we assume the hypothese to be validated.

1. Testing :

Predict the output for the Testing data and calculate ERMS.

If the ERMS value is acceptable, the implementation is successful.

1. Critical observations :

As the number of iterations increases, the difference between ERMS values from 2 consecutive calls decreases. When this difference reaches a pre-defines minimum the algorithm stops the iterative procedure and outputs the result.

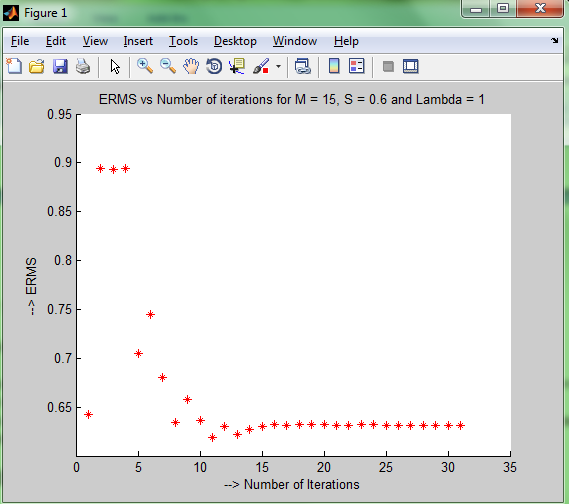


Fig 6 : Number of iterations vs ERMS for Stochastic Gradient Descent.

=====================================================================

**Performance comparision of CFS and Stochastic Gradient Descent :**

Stochastic Gradient Descent is faster than the Closed form approach.

Since ERMS being calculated at each iteration reduces, W can be derived before all the rows are considered.

This approach is applicable for problems with large data sets, where applying the closed form solution means multiplication of matrices of high order and hence is inefficient.

Closed Form appoach can be used when the size of the data set is relatively small.