Project 1.2 Report

Redwan Ibne Seraj Khan(UBITname: rikhan)
Department of Computer Science and Engineering
University at Buffalo
Buffalo, NY-14228
rikhan@buffalo.edu

Abstract

In this project machine learning was used to solve a problem that arises in Information Retrieval, one known as the Learning to Rank (LeToR) problem. This problem was formulated as a problem of linear regression where we map an input vector x to a real-valued scalar target y(x,w).

1 Closed Form Solution

At first the data was preprocessed. The target data was collected and separated. All the null variant features were removed so that it is possible to inverse the co-variance matrix. Then the data was split into Training, Validation and Testing sets. Training consisted of 80% of the data and both validation and testing consisted 10% of the data each.

t =w^Tφ(**x**) is our linear regression model, where t is the target value. From this equation we get w = $(\varphi^T \varphi)^{-1} \varphi^T$t . As φ(**x**) is not a square matrix we need the pseudo inverse.

In the LeToR dataset there were 69623 query document pairs with 46 features. We deleted 5 features as they had all zeros. Otherwise we would not be able to inverse the co variance matrix. After the preprocessing of data, the training dataset had a shape of (41,55699), the validation data had a shape of (41,6962) and the test data had a shape of (41,6961).

The computation needed basis function for converting input vector x into a scalar value. The number of basis functions was taken as 10 as the starting point. Gaussial radial basis function was taken for working on this project.

$$\phi_j(\mathbf{x}) = \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_j)^\top \Sigma_j^{-1}(\mathbf{x}-\boldsymbol{\mu}_j)\right)$$

Here the dimensions of the quantities that would be found from computation are shown below.

x - μ_j : 1x41

Σ_j^{-1} : 41 x41

So the calculation of the above basis function would give a scalar value of dimension 1x1.

BigSigma is a diagonal matrix which was created with the variances of feature vectors. The dimension of BigSigma was 41x41 as we had removed 5 of the features while processing the raw data. The value of μ is obtained by clustering of data.

Let us take the example of training set data to clarify the solution. The design matrix φ(**x**) for training data will have the dimension of 55699x10. The matrix multiplication for the calculation of w is shown below.

$$(\varphi^T \varphi)^{-1} \varphi^T t$$
$$=((10x55699).(55699x10))^{-1} (10x55699).(55699x1)$$
$$= \quad (10x10) \quad . \quad (10x1)$$
$$= \quad (10x1)$$

After doing all the computation RMS was used for testing the results.

2 Gradient Descent Solution

For implementing the gradient descent solution at first the weights were initialized.

$$[W]_{10x1}$$

Then the weights were updated for each data point.

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} + \Delta\mathbf{w}^{(\tau)}$$

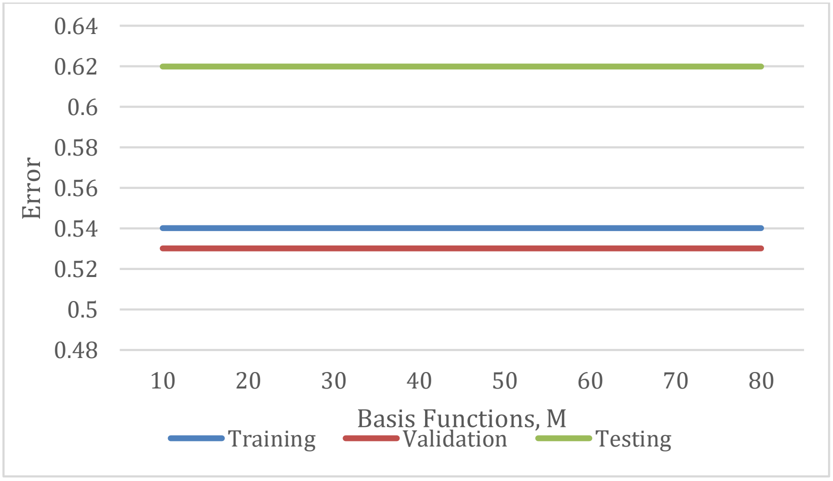
At last the RMS values were calculated for the training set, validation set and testing sets of data.

3 Hyper-Parameter Tuning

3.1 Effect of different number of Basis Function and regularization term

After increasing the number of basis functions there was no significant improvement in performance.

```
M = range (10,80)
Lambda = range (0.03,0.9)
E_rms Training   = 0.5494405428794945
E_rms Validation = 0.5384969256212522
E_rms Testing    = 0.6279439886025582
```



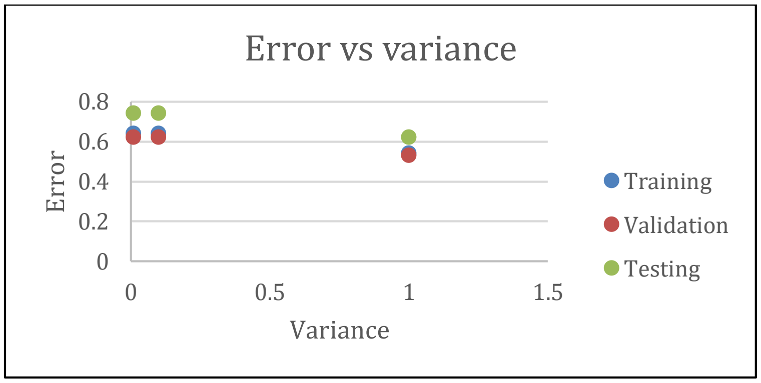
3.2 Effect of center of Gaussian radial basis function

There was no notable effect on choosing the centers of Gaussian radial basis function in a random way.

3.3 Effect of spread of spread of Gaussian radial basis function

Changing the variances to be 1/10 decreased the accuracy to some extent.

```
E_rms Training   = 0.6427528185481821
E_rms Validation = 0.628262808818991
E_rms Testing    = 0.7408882388927146
```

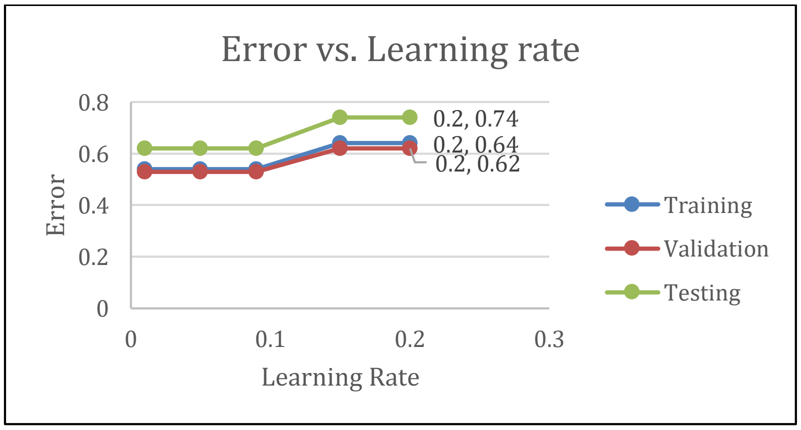


However, making the variances even smaller to 1/100 did not produce any worse results.

3.4 Effect of Learning rate and La

After increasing the learning rate gradually to 0.2 the performance deteriorated to a little extent but then remained constant no matter how much the learning rate was increased. Having a very big learning rate caused the computation to go out of bounds. No notable change was observed on changing the value of La as well. The end result is shown below.

```
E_rms Training   = 0.64275
E_rms Validation = 0.62826
E_rms Testing    = 0.74089
```



4 Analysis of Results

The accuracy obtained was not up to the mark even after changing the hyper parameter in different ways. This shows that to improve the performance we perhaps need more data or better algorithms rather than linear regression.