Project 1.2: Learning to Rank using Linear Regression

Arinjoy Bhattacharya
Department of Physics
University at Buffalo, SUNY
Buffalo, NY 14214
Person # 50168806
arinjoyb@buffalo.edu

Abstract

The LeToR is a very fundamental problem of linear regression which implements various fundamental aspect of machine learning such as clustering and Moore-Penrose matrix inversion. In this report, two ways of solving the problem have been enunciated: Stochastic Gradient Descent and Closed form solution.

1 Introduction

Linear regression is a very powerful tool which has been very effective in deep learning and information retrieval. Defining basis functions to tackle it and SGD form to train it has been used widely over the years

2 Model

2.1 Model: Dataset

The LeToR dataset derives the input vector from a query-URL pair and the target value is human value assignment about how well the URL corresponds to the query. It has been taken from the Microsoft search engine results. The relevance vector takes one of the discrete values 0, 1 or 2. The larger the relevance label, the better is the match between query and document. The objective output y of our linear regression will give a continuous value rather than a discrete one– so as to give a fine-grained distinction. Its dimensions are *46* x *69623*.

2.2 Model: Partitioning the dataset

Like in any other Machine Learning problem, we implement the method of training, validation and testing. So we partition the dataset into three parts:

- 80% training
- 10% validation
- 10% testing

These three datasets are distinct and don't overlap with each other.

2.3 Model: Linear regression model

In this case the Gaussian radial distribution model to map the linear regression model.

$$y(\boldsymbol{x}, \boldsymbol{w}) = w^T \phi(x)$$

Here, ϕ(x) are the Gaussian radial basis functions. Then a probabilistic approach is implemented to find the error associated with the output. Now the output is a normal distribution and the Error function can be given as:

$$E = \sum_{i=1}^{N} (t_i - w^T \phi(x_i))^2$$

2.4 Model: Closed form solution

The Error function in the matrix form can be represented as:

$$E = \frac{1}{2}\left(t^T - w^T \phi^T(x)\right)(t - \phi(x)w)$$

where t and ϕ(x) represent follows:

$$t = \begin{bmatrix} t_1 \\ \vdots \\ t_n \end{bmatrix} \text{ and } \phi(x) = \begin{bmatrix} \phi(x_{11}) & \cdots & \phi(x_{1n}) \\ \vdots & \ddots & \vdots \\ \phi(x_{n1}) & \cdots & \phi(x_{nn}) \end{bmatrix}$$

Each row in the ϕ(x) matrix correspond to a set of relu basis functions corresponding to t_1.

So, to decrease error, we equate the first order differential of E to 0.

$$\nabla E = \frac{\partial E}{\partial w} = \left(t^T - w^T \phi^T(x)\right)\phi(x) = 0$$
$$\Rightarrow t^T \phi(x) = w^T \phi^T(x)\phi(x)$$
$$\Rightarrow w^T = t^T \phi(x)(\phi^T(x)\phi(x))^{-1}$$
$$\Rightarrow w = \left(\phi^T(x)\phi(x)\right)^{-1}\phi(x)t$$

which is our form for the closed form solution.

2.5 Model: Regularization

There is need for regularization so as to prevent overfitting. In general, there is a regularization term added to the weighted error term so that the weights term minimizes the error along with the regularizer term.

$$E(w) = E_D(w) + \frac{1}{2}\lambda w^T w$$

Our closed form solution with a regularizer has a λ**I** term added to it where **I** is the identity matrix, and λ is our regularizer term.

2.6 Model: Clustering

The dataset is too huge to work with it by itself. Thus it is broken up into smaller matrices using k-means clustering so that they can be fitted into Gaussian basis functions. Then these basis functions are used for regression by utilizing their mean values and the Gaussian spreads. The means are denoted by μ and the spread as a matrix called Σ.

$$\boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\sigma_1^2} & \cdots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & \boldsymbol{\sigma_D^2} \end{pmatrix}$$

2.7 Model: Stochastic Gradient Descent solutions

This algorithm decreases the weights based on a particular learning rate which is provided by the user. Therefore, the change in the learning rate affects the change in the error function accordingly. The change in error function is inversely proportional to the change in the value

of learning rate.

$$\Delta w = -\eta \nabla E$$

3 Experiment

For the experiment, the linear regression model was trained with the various hyperparameters and are sequentially explained with plots below.

3.1 Experiment: different number of relu basis functions

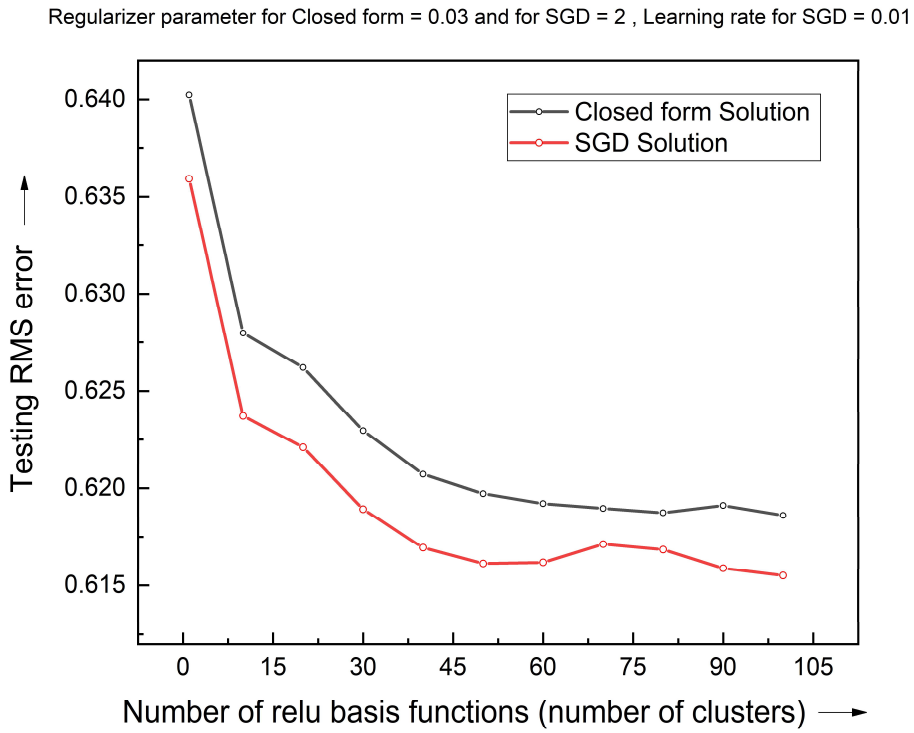

Figure 1: Testing E_{RMS} Vs number of relu basis functions which in turn is the number of clusters.

From the figure shown above, it clearly shows that increasing the number of basis functions leads to decrease of the root sum of square errors. Also, an important observation to be made here is that our stochastic gradient method gives out less error that the closed form one which might be due to implementation of k-means clustering rather than the use of inverse Moore-penrose matrix.

3.2 Experiment: different regularization parameters for SGD solution

As we see from the figure below, the RMS error blows up at very small value of regularizer. But after around $\lambda = 1$ to higher values, we get our desired value of $E_{RMS} = 0.6$. The reason may be evident from the formula of the error function for SGD.

$$\nabla E = -\big(t_i - w^T \phi(x_i)\big)\phi(x_i) + \lambda w$$

To reduce error, we have to equate it to 0, so for lower λ values, the error function of the weights increase and hence the testing error blows up.

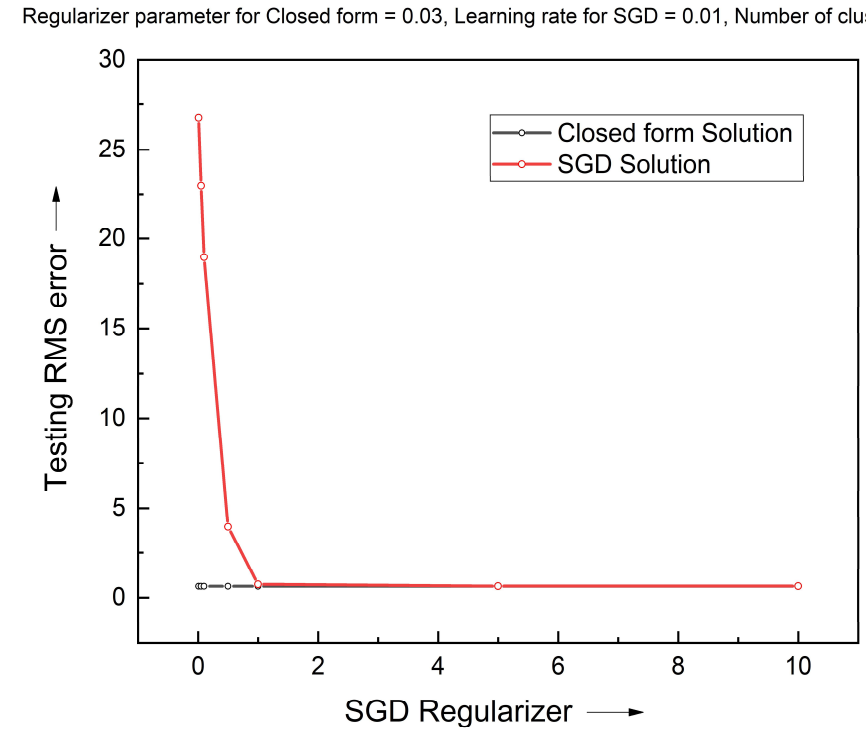

Figure 2: Testing E_{RMS} Vs different values of the SGD regularizer (λ).

3.3 Experiment: different regularization parameters for Closed form solution

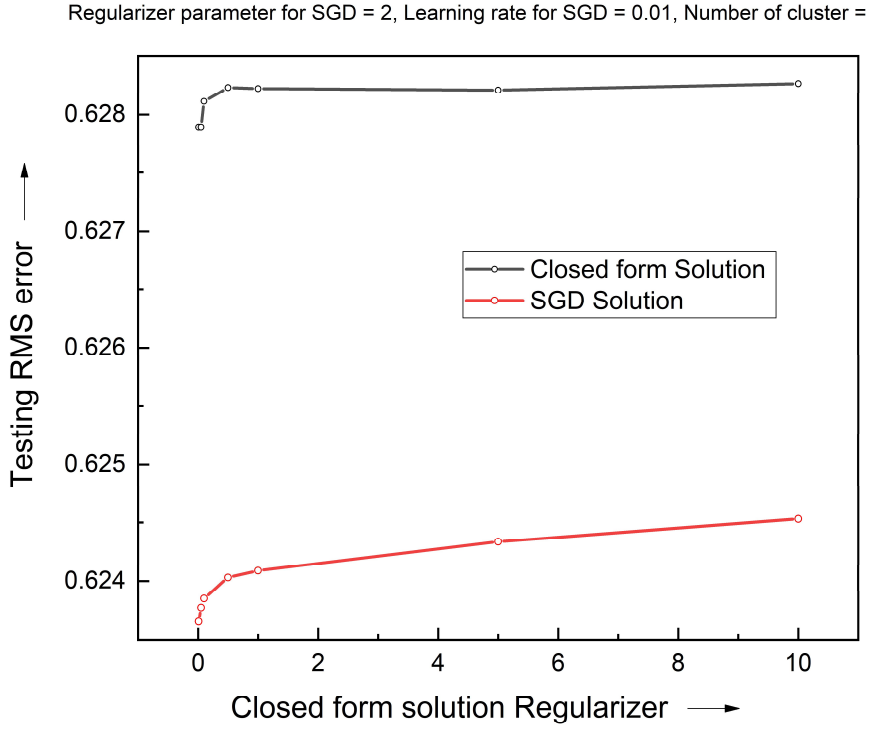

Figure 3: Testing E_{RMS} Vs different values of the Closed form regularizer (λ_c).

There is general trend for increase of the RMS error with the increase in λ_c. This might be explained by the formula.

$$w^* = (\lambda I + \Phi^T\Phi)^{-1}\Phi^T t$$

With increase in λ_c the w* increases which in turn results in increasing Error.

3.4 Experiment: different values of learning rate of SGD.

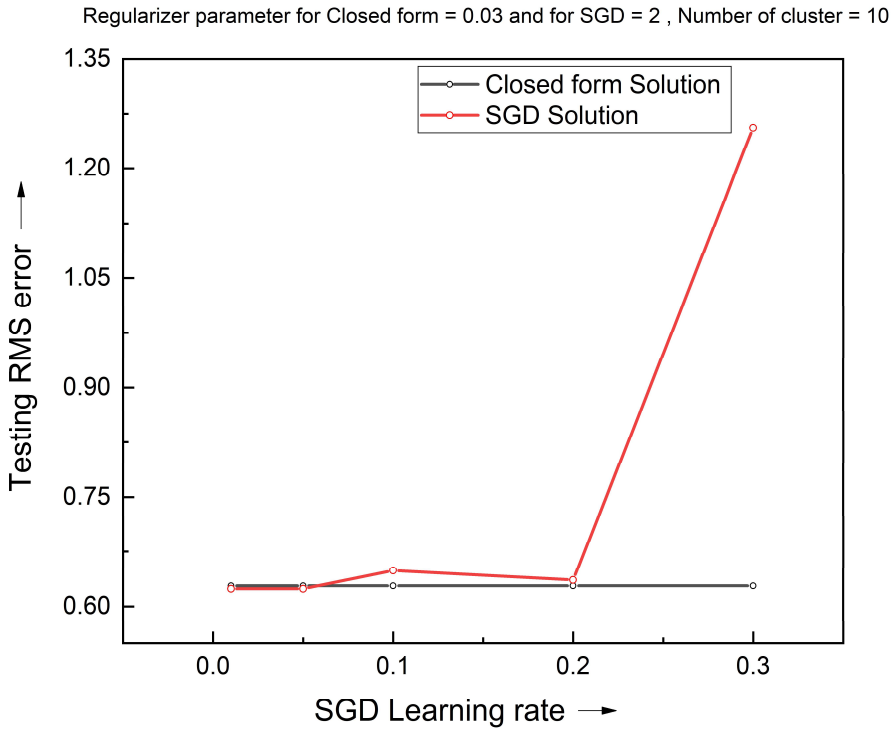

Figure 4: Testing E_{RMS} Vs different values of learning rate (η)

The learning rate is also detrimental in changing the error of the Error function which can be seen to blow up after the value of 0.2. Now learning rate basically determines how much should the error function of SGD decrease, or in other words how much should the weights be changed so as to get optimal error.

$$\nabla w = -\eta \nabla E \Rightarrow w_{new} = w_{prev} + \Delta w$$

3.5 Experiment: the perfect parameter

For the least RMS error, the following parameters seemed proper:

- 100 or more relu basis functions (clusters in k-means)
- SGD learning rate of 0.01.
- Closed form regularizer of 0.01
- SGD Regularizer of around 2.
- Number of iterations in the SGD calculation = 6000.

3 Conclusion

The SGD solution offers less error in testing the LeToR data set rather than that of the closed form solution but the closed form is way faster.

Acknowledgments

I thank the professor Dr. Srihari and the TAs for helping me complete this report.

References

[1] CM Bishop, TM Mitchell. (2014) *Pattern Recognition and Machine Learning*, Springer.