Project Report 2
Learning to Rank using Linear Regression

CSE-574 - Fall 2017

Deepika Chaudhary, Manpreet Dhanjal

(UB IT Name: d25, PN: 50248725), (UB IT Name: dhanjal, PN: 50248990)

Introduction:

The given project expects us to find the relevance between a given query and document. The technique is to find the gradient descent and minimize the cost function in order to get the relevance levels. For this we have used two methods: Closed form Solution and Stochastic Gradient Descent method. The analysis was done on 2 datasets (LeTOR and Synthetic) and the results were obtained as explained further in the report.

Data Partition:

Both the datasets were divided into 3 subsets. 80% of the dataset was used for training, 10% for validation and the rest 10% for the testing purposes. This was done by first taking 0 as lower bound and the upper bound was calculated by taking 80% of the total data. Next 10% was taken by setting lower bound to 80% data point and 90% data point as upper bound. The remaining 10% was used for testing purposes. Note that these datasets do not overlap. The calculation was performed on both LetTOR Data and synthetic data for processing.

Hyper Parameter Tuning:

1. Calculating μ and Σ:

For calculating μ, we performed k means clustering on the training data to find the centers. K means clustering is the method of randomly choosing k clusters and then analyzing rest of the data according to those clusters. We choose k equal to the number of radial basis functions we need. Initially we chose k=4, and at every iteration as M changes the number of centers also changes.

To calculate Σ, for every center the variance was calculated and adjusted into the diagonal of a D x D matrix where D are the number of features of the input data. σ^2 is proportional to the i-th dimension variance of the training data and is calculated as

$$\sigma_i^2 = \frac{1}{10}\text{var}_i(\mathbf{x}) \qquad \Sigma = \begin{pmatrix} \sigma_1^2 & & & \\ & \sigma_2^2 & & \\ & & \ddots & \\ & & & \sigma_D^2 \end{pmatrix}$$

The D x D variance matrices for each cluster is stacked according to their cluster label in a M x D x D matrix.

2. Adjusting M and λ:

In order to tune the number of basis function M and the regularization term λ we made use of our validation set.
Initially the closed form solution was obtained from the training set and validation set was fed to this model, and Erms was calculated. To find M, we used grid search in the range of 4 to 50. Initially M was chosen to be 4 and with every iteration we increased the it by 2 while analyzing the value of Erms. Finally, the value with minimum Erms was set as M for the optimal model.
λ which is used for regularization or say to control over fitting was also calculated by the grid search method. We took an array of expected values and at every iteration the value of Erms was calculated. The value with which minimum Erms was obtained was chosen to be the λ value.

- LeTOR dataset

Closed form Solution:
After validation the final value of M=42 and λ =0.1were obtained. The graph of Erms with different values of M and λ are as follows:

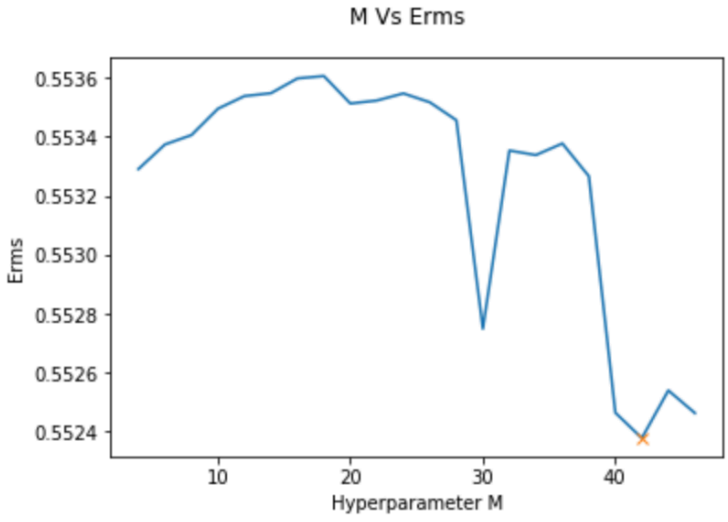


Figure: Tuning M for LeTOR dataset

- Synthetic dataset

Closed Form solution:

After validation the final value of M=34 and λ =0.1were obtained. The graph of Erms with different values of M and λ are as follows:

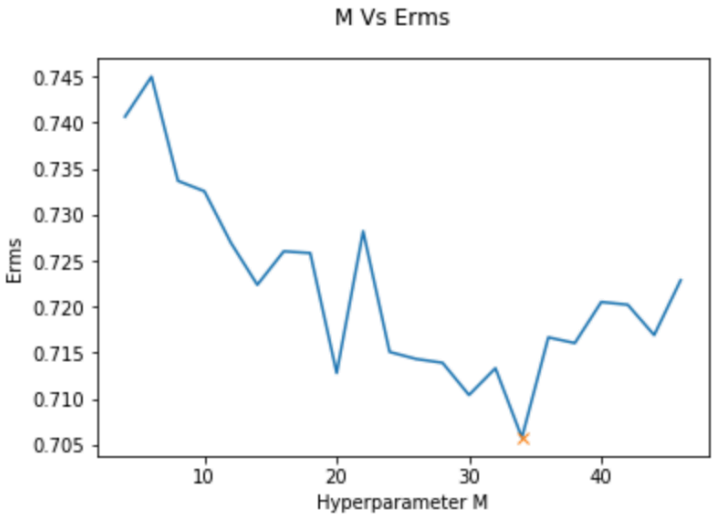


Figure: Tuning M for Synthetic dataset

3. Deciding Learning rate η :

For stochastic gradient descent the values of model parameter '**w**' were calculated by the formula below:

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

where $\Delta\mathbf{w}^{(\tau)} = -\eta^{(\tau)}\nabla E$

The learning rates are used to optimize the values of weights. Learning rate decides how fast the error function is reduced to zero. We can optimize the weights in 2 ways either by fixing the learning rate or by varying it. Fixing the learning rate would give a very poor performance. So we took the approach of learning rate adaptation. Initially we chose η to be 0.01 and analyzed the behavior of cost function. Then we made the value η 3 fold and repeatedly analyzed the behavior of cost function. Below are the results of cost function versus the learning rate with different data sets:

- LeTOR Datasets

We analyzed the behavior of the cost function(Error function) with different values of η and plotted the results. We can see that for $\eta > 0.1$ the cost function takes forever to reach minimum value. Therefore we can chose η to be 0.01 or 0.03 as we can calculate the weights easily in less number of iterations. For this project we used η to be 0.03

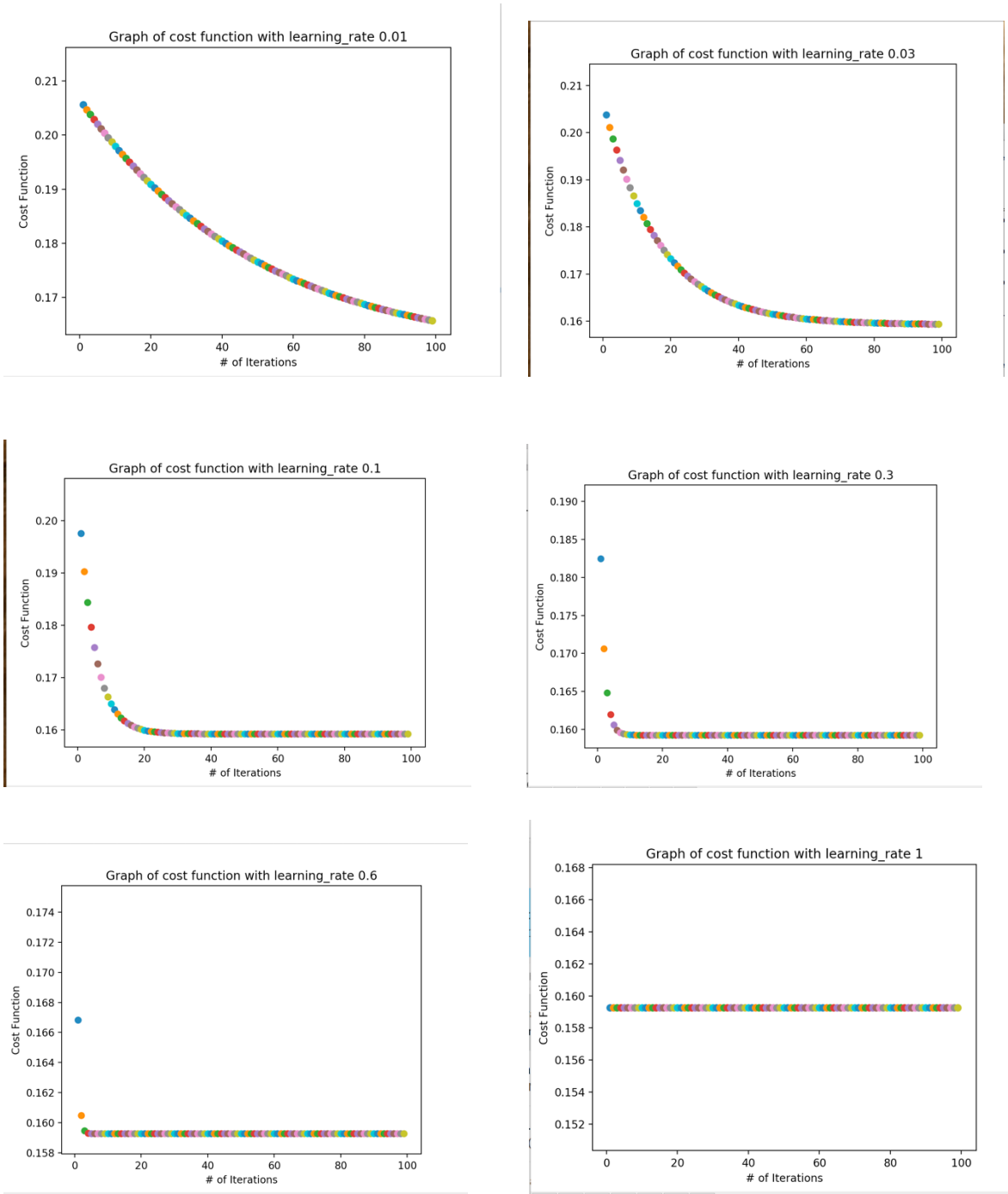
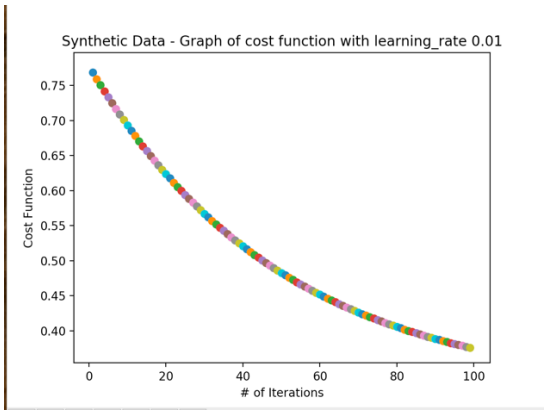
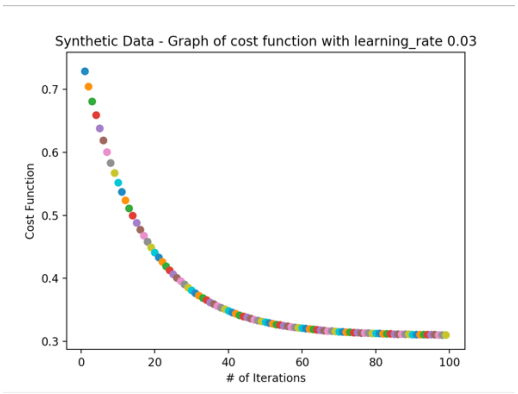


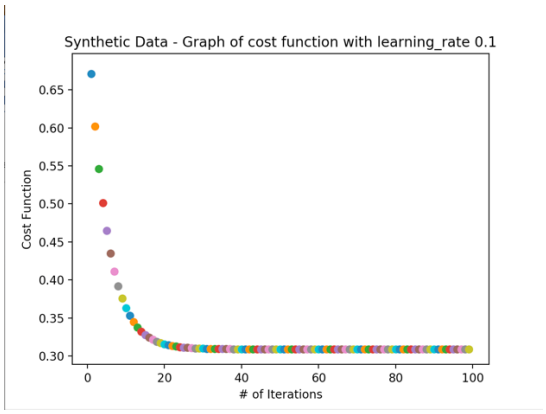
Figure: Tuning learning rate for LeTOR dataset

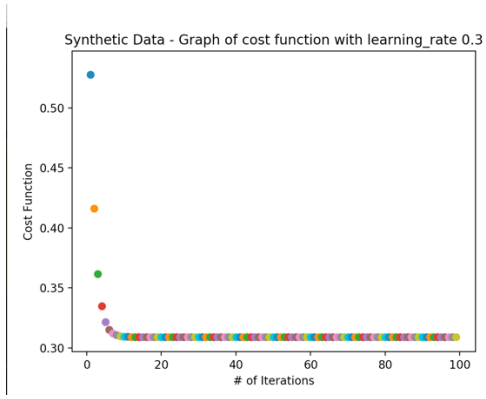
- Synthetic Datasets

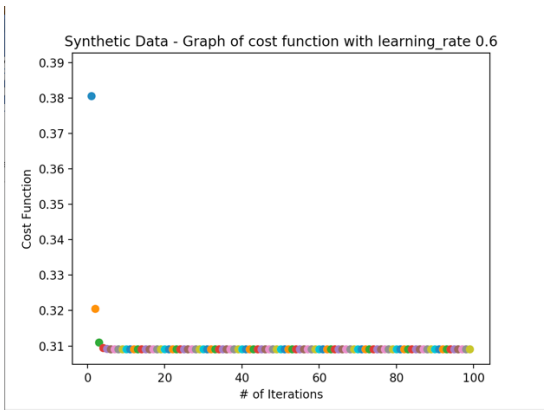
From the results we can see that the behavior of cost function is best depicted with η = 0.03











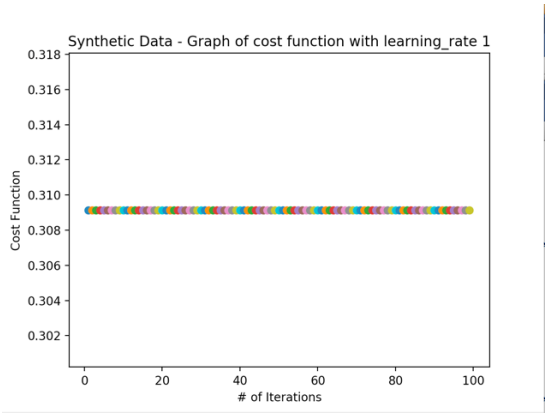


Figure: Tuning learning parameter for Synthetic dataset

4. Optimal number of iterations

To get the optimal number of iterations while performing stochastic gradient descent, we use early stop. In early stop, we take a patience number which is equal to the number of times we can tolerate the worsening validation error before we stop the training for stochastic gradient descent and we store the weights and number of iterations corresponding to the minimum validation error.

Results and Analysis:

1. Closed Form Solution:

In the closed form solution, we try to build a linear model that describes our input sets given the output dataset. The main goal is to obtain a linear function by minimizing the error.
The weight vectors are obtained using the equation

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

where phi is the matrix of radial functions and can be described as

$$\mathbf{\Phi} = \begin{bmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \phi_2(\mathbf{x}_1) & \cdots & \phi_{M-1}(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & \phi_2(\mathbf{x}_2) & \cdots & \phi_{M-1}(\mathbf{x}_2) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \phi_1(\mathbf{x}_N) & \phi_2(\mathbf{x}_N) & \cdots & \phi_{M-1}(\mathbf{x}_N) \end{bmatrix}$$

Finally, the estimated output of any new dataset can be derived using the equation

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

- LeTOR dataset

M = 42
Lambda = 0.1
Erms Train = 0.5635
Erms Validation = 0.5525
Erms Test = 0.6384

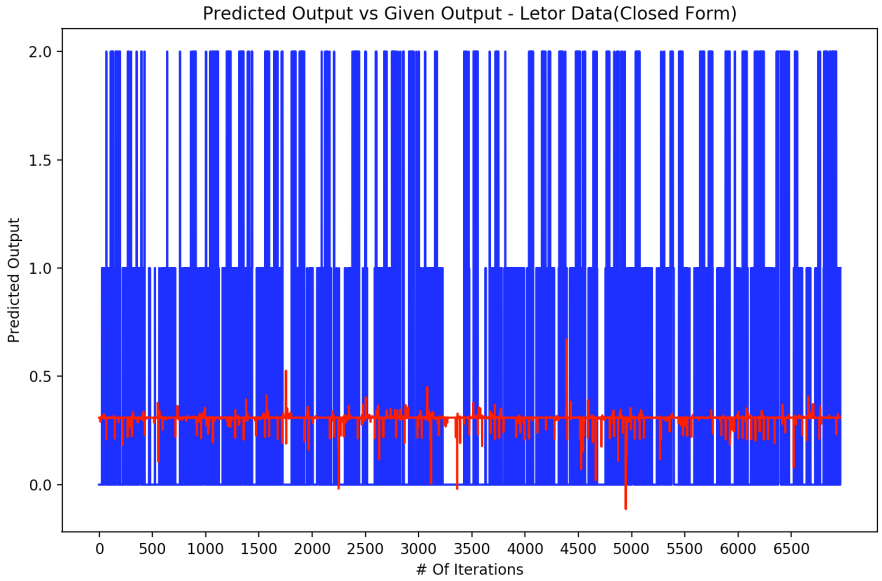


Figure: Closed Form Solution - LeTOR

- Synthetic dataset

M = 34
Lambda = 0.1
Erms Train = 0.7098
Erms Validation = 0.7117
Erms Test = 0.7164

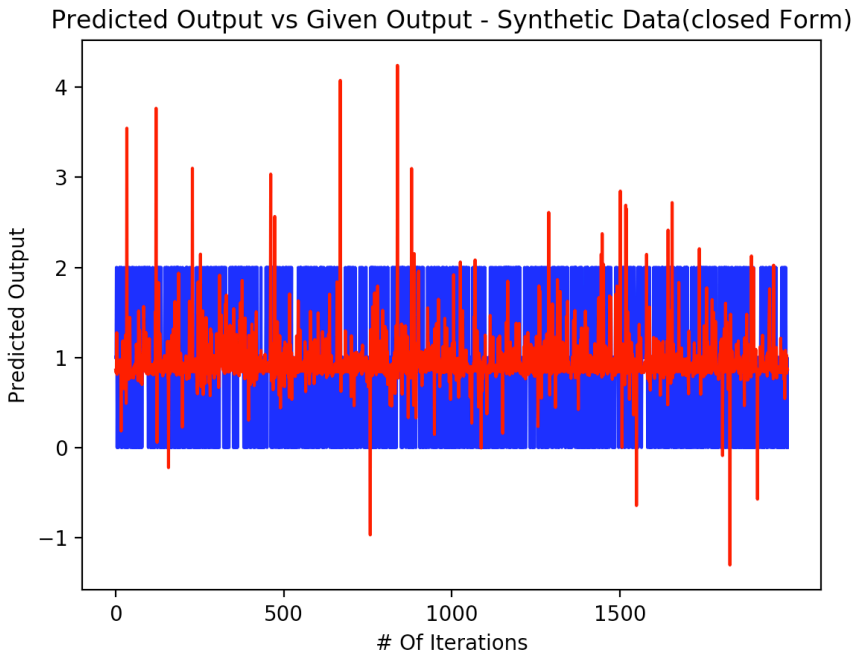


Figure: Closed Form Solution – Synthetic data

2. Stochastic Gradient Descent:

Stochastic gradient is an iterative approach of training a linear regression model for given dataset. Here, we start with random weight vectors and iteratively update them while minimizing a cost function. We have used mean square error as the cost function.
The weight vectors are updated according to the formula

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

using the following

$$\nabla E = \nabla E_D + \lambda \nabla E_W$$

$$\nabla E_D = -(t_n - \mathbf{w}^{(\tau)\top}\boldsymbol{\phi}(\mathbf{x}_n))\boldsymbol{\phi}(\mathbf{x}_n)$$

$$\nabla E_W = \mathbf{w}^{(\tau)}$$

- LeTOR Dataset

Learning rate = 0.03
Lambda = 0.1
Erms Train = 0.591
Erms Valdiation = 0.605
Erms Test = 0.643

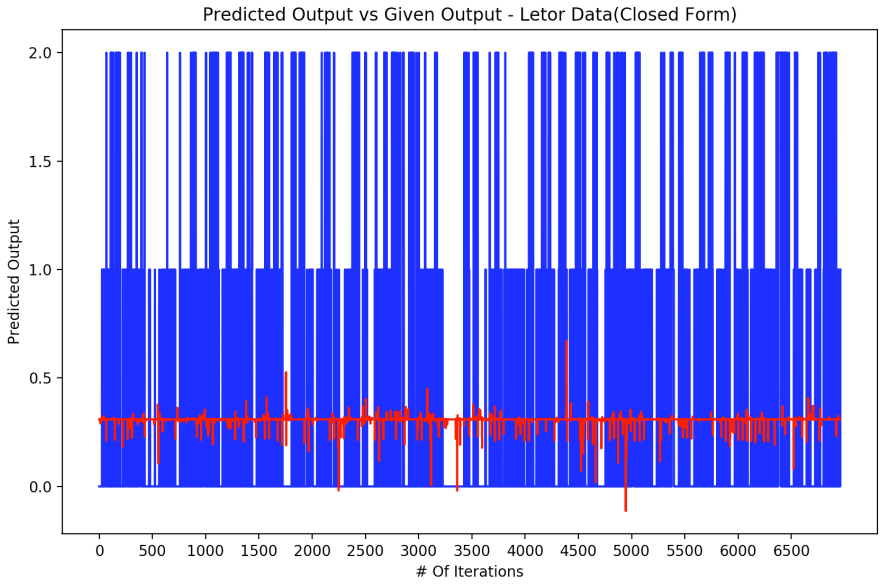


Figure: SGD – LeTOR Data

- Synthetic Data

Learning rate = 0.03
Lambda = 0.1
Erms Train = 0.727
Erms Validation = 0.7809
Erms Test = 0.7824

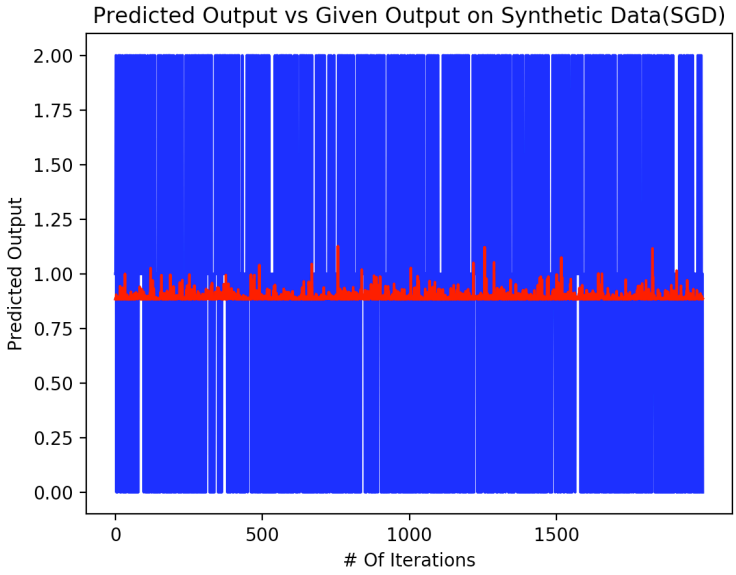


Figure: SGD – Synthetic Data