Report For Project2

1.Data Split:

Letor Data Set : This data set has 69623 rows and 46 columns
Synthetic Data Set : This data set has 20000 rows and 10 columns

The Letor Data Set is prepared by reading from Querylevelnorm.txt
The Synthetic data Set is read from the both input.csv file and output.csv file using the csv.reader

The Letor Data Set and Synthetic data Set are shuffled using numpy.random.shuffle() so as to minimize the Error values for prediction for the three datasets.

For Splitting the data a user defined class named DataSplitClass is used with parameters

1. trainingrowsize : Size of the training data Set
2. validationrowsize = Size of the validation data Set
3. testingrowsize = Size of the Testing data set

4.colsize = Number of Columns or the number of the input features

5. trainingfeaturesmat = Matrix to hold the training feature data values
6. traininglabelsvec = Matrix to hold the training label labels

7.validationfeaturesmat = Matrix to hold the validation feature data values
8.validationlabelsvec = Matrix to hold the validation label values

9. testingfeaturesmat = Matrix to hold the testing feature data values
10. testinglabelsvec = Matrix to hold the testing label values

The dataset is partitioned into 80% , 10% and 10 % to form training , validation and Testing data Sets correspondingly .
Two objects of type DataSplitClass, are created one for the letor data set and another for the synthetic data set

The constructor of this class takes trainingrowsize, validationrowsize, testingrowsize and colsize as the input parameters ,
For Letor data set object
Trainingrowsize = 55698
validationrowsize = 6962
testingrowsize = 6963

For synthetic data set object
Trainingrowsize = 16000
validationrowsize = 2000
testingrowsize = 2000

The member function 'SplitData()' is called which copies the values from original data using start index and data set size(training or validation or testing) parameter into either training ,validation or testing data sets

For Letor
Training Data Starting Index = 0
Validation Data Starting Index = 55698
Testing Data Starting Index = 62,660

For Synthetic
Training Data Starting Index = 0
Validation Data Starting Index = 16000
Testing Data Starting Index = 18,000

Using the above mentioned two class objects (one for letor and another for synthetic) the corresponding dataset is accessed

2. Hyper Parameter tuning :

M and lambda:

As per the feature size, a range of M values are selected .
Considering the wiki link given for the grid search for the hyper-parameters in project specification
range of lambda values are selected

For Letor data , The Set of M values considered is M = {5,7,9,11,13 ,15} and lambda = {0.1,0.3,0.5,0.9}
For Synthetic data , The Set of M values considered is M = {3,5, 8} and lambda = {0.1,0.3,0.5,0.9}

Number of iterations in Letor n = 6*4 = 24
Number of iterations in Synthetic n = 3*4 = 12

(2 nested for loops are run) for every M and Lambda combination i.e. n times mentioned above
For Every iteration weights and training rms error are obtained from the training data set .This weights are given as input to calculate Erms on the validation data set .

For all the 'n' iterations the lowest of all the validation RMS Error values is chosen , the corresponding M , lambda and weight vector are selected as the Final values .

The above mentioned selected final parameters are used to calculate the Erms error on testing data set which is the final Erms error and benchmark for the prediction accuracy for the trained model

Above Steps are similar for closed form and the Stochastic Gradient except that the
the method of calculating weights differs

Closed form weight Determination :
The Basis function matrix is calculated as per the equation given in specification using Muj and Sigmaj .
This is discussed more elaborately in upcoming section .
The closed form solution Equation (with regularization parameter) is used to find the weights

Stochastic Gradient method of weight Determination :
As per the value of M , the random weight vector is prepared using numpy.random.random() . The size of this vector is M .This weight vector is used as input to the method that operates on training data set which calculates weights.
Learning Rate is selected as 1.

The gradient descent is run for the iterations equal to the size of the data set.

For first iteration Erms is calculated using the input random weight vector mentioned above. Thereafter
Erms is calculated using the weights determined in previous iteration.

If the current rms is greater than Previous rms , the learning rate is decreased(dividing by2) and the weights are not updated .

If the current rms is less than previous rms by a factor 0.001 , weights are updated and the for loop is exited .

The final weight vector would consist the values for which the Erms error is lowest

Plots :
(note : The minimum values are already calculated in the code . Inference statement from plots shall reverfiy this)

Closed Form Solution For Letor Data :

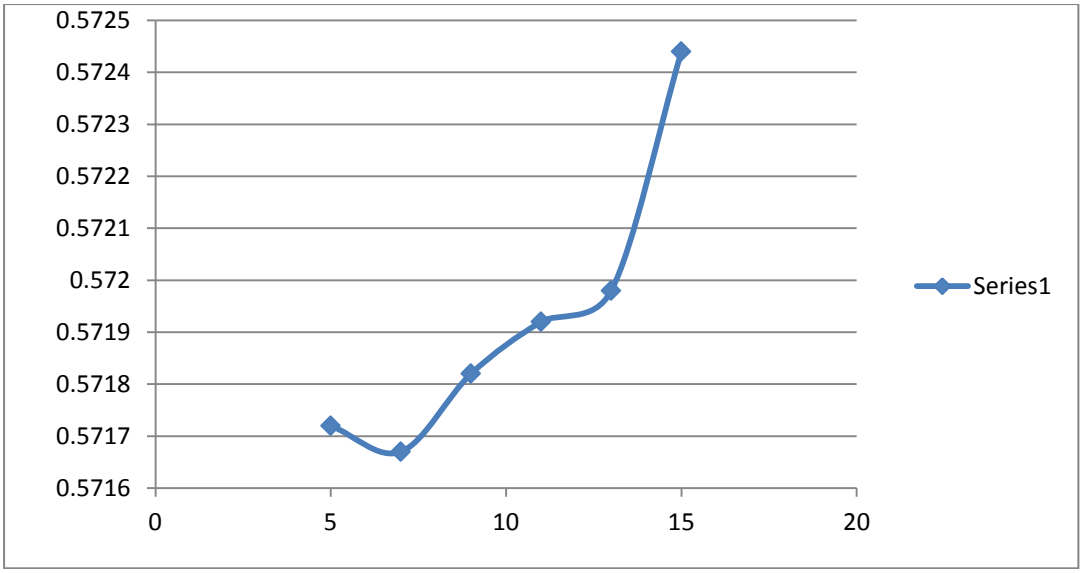
Table for M,lambda and Validation Rms error Values

		Validation

M	Lambda	Erms Error
5	0.1	0.57207
5	0.3	0.57172
5	0.5	0.57172
5	0.9	0.57174
7	0.1	0.57174
7	0.3	0.57176
7	0.5	0.57167
7	0.9	0.57216
9	0.1	0.57179
9	0.3	0.57239
9	0.5	0.57182
9	0.9	0.57175
11	0.1	0.57211
11	0.3	0.57242
11	0.5	0.57192
11	0.9	0.57186
13	0.1	0.57183
13	0.3	0.57221
13	0.5	0.57198
13	0.9	0.57214
15	0.1	0.57167
15	0.3	0.57173
15	0.5	0.57244
15	0.9	0.57238

Plot between M and Erms

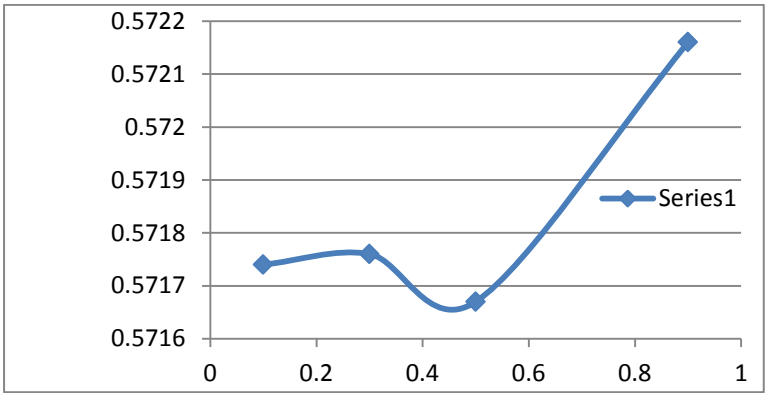
lambda = 0.5(minimum)						
M (X-axis)	5	7	9	11	13	15
Validation rms error(Y-axis)	0.57172	0.57167	0.57182	0.57192	0.57198	0.57244



Inference : M = 7 (minimum value) for minimum validation rms error = 0.57167

Plot Between lambda and Erms

M = 7 (minimum)				
lambda(X-axis)	0.1	0.3	0.5	0.9
Validation rms error(Y-axis)	0.57174	0.57176	0.57167	0.57216



lambda = 0.5 (minimum value) for minimum validation rms error = 0.57167

2.Plots for Closed Form Synthetic Data :

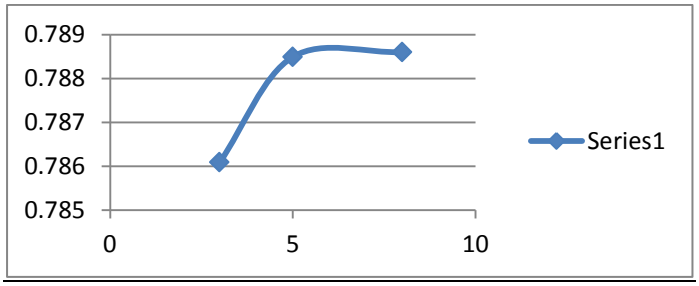
Table for M,lambda and Validation Rms error Values

M	lambda	Validation Rms Error
3	0.1	0.78623
3	0.3	0.78609
3	0.5	0.78667
3	0.9	0.78692

5	0.1	0.78685
5	0.3	0.78849
5	0.5	0.78748
5	0.9	0.78707
8	0.1	0.79003
8	0.3	0.78756
8	0.5	0.7886
8	0.9	0.78727

Plot between M and Validation Rms Error

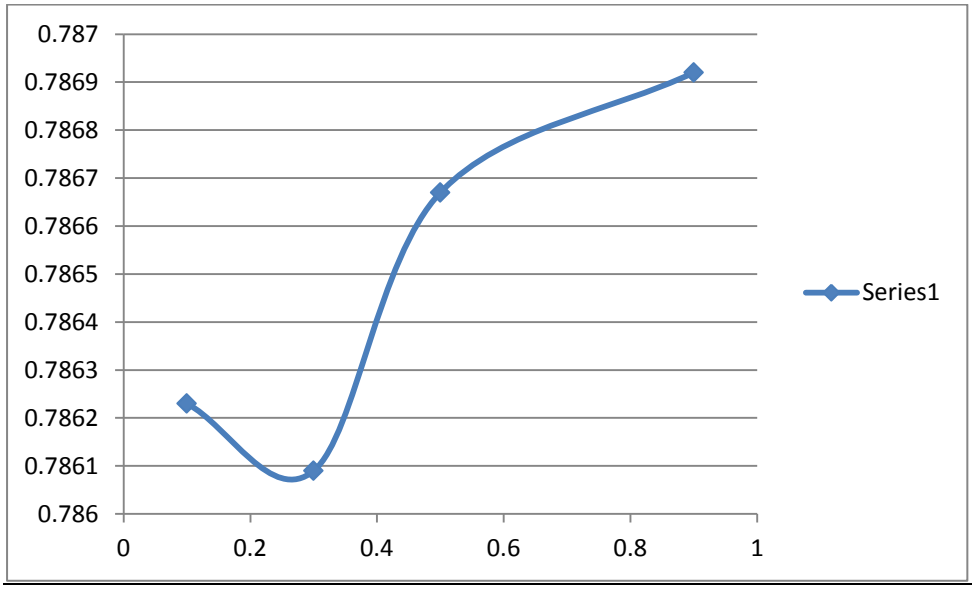
lambda(min) =0.3			
M(X-axis)	3	5	8
Validation rms error(Y-axis)	0.78609	0.78849	0.7886



Inference : M =3 is the minimum value for minimum validation error = 0.78609

Plot between lambda and Validation Rms Error

M = 3(minimum)				
lambda (X-axis)	0.1	0.3	0.5	0.9
Validation rms error(Y-axis)	0.78623	0.78609	0.78667	0.78692



Inference :lambda(minimum) =0.3 for minimum Validation Rms error = 0.78609

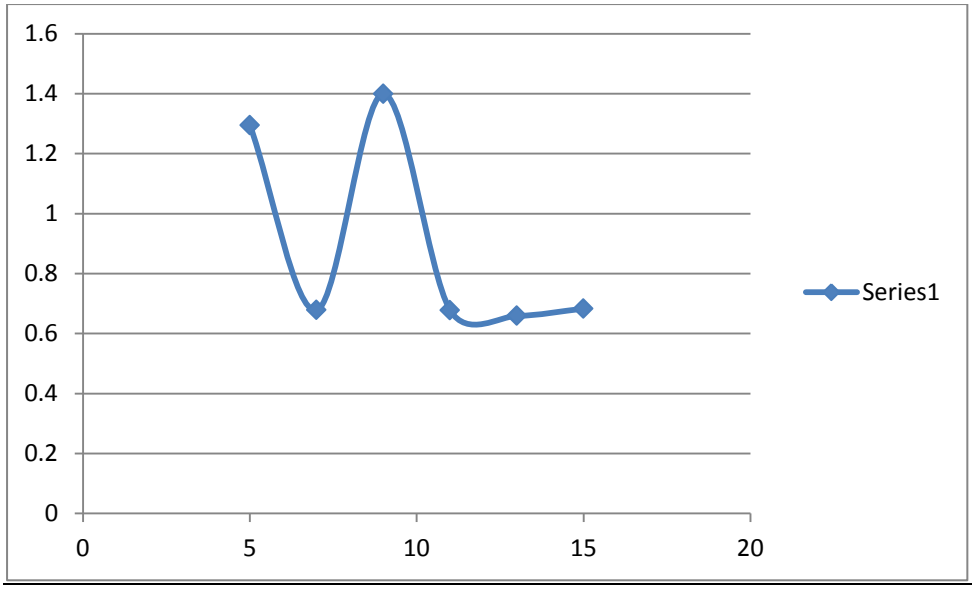
Plots for stochastic gradient descent Letor Data :

Table for M ,lambda and Validation Rms Error Values

M	lambda	validation rms error
5	0.1	1.29495
5	0.3	0.7637
5	0.5	0.71221
5	0.9	1.19777
7	0.1	0.67821
7	0.3	0.77685
7	0.5	0.68723
7	0.9	0.94671
9	0.1	1.39909
9	0.3	0.69484
9	0.5	0.66973
9	0.9	0.85657
11	0.1	0.67792
11	0.3	0.70981
11	0.5	0.72145
11	0.9	0.79617
13	0.1	0.65897
13	0.3	0.78726
13	0.5	0.90892
13	0.9	0.7014
15	0.1	0.68309
15	0.3	0.70674
15	0.5	0.86993
15	0.9	0.75042

Plot between M and Minimum Validation Rms Error

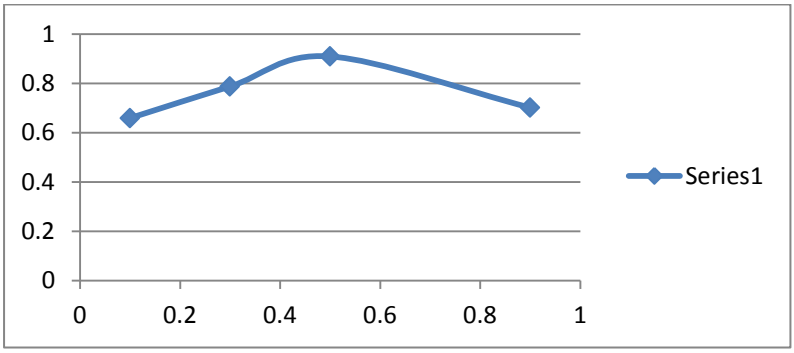
lambda(min) =0.1						
M(X-axis)	5	7	9	11	13	15
Validation rms error(Y-axis)	1.29495	0.67821	1.39909	0.67792	0.65897	0.68309



Inference : M(minimum) = 13 for the minimum validation rms error = 0.65897

Plot between lambda and Minimum Validation Rms Error

M = 13(minimum)				
lambda (X-axis)	0.1	0.3	0.5	0.9
Validation rms error(Y-axis)	0.65897	0.78726	0.90892	0.7014



Inference : lambda(minimum) = 0.1 for minimum validation error = 0.65897

Plots for stochastic gradient descent Synthetic Data :

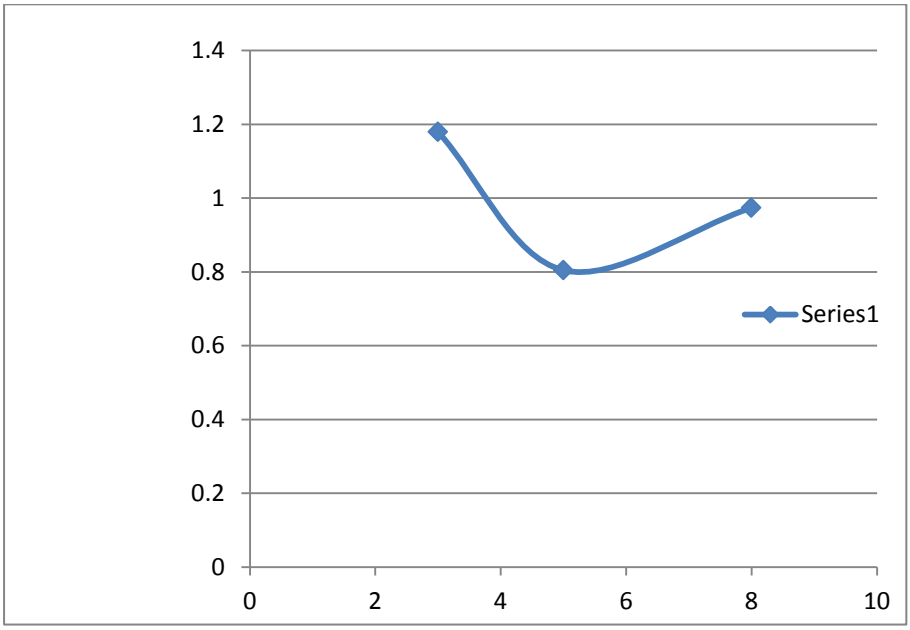
Table for M , lambda and Validation Rms Error Values

		validation rms error
M	lambda	
3	0.1	1.25442
3	0.3	1.26854
3	0.5	1.00844

3	0.9	1.17993
5	0.1	1.27484
5	0.3	1.12645
5	0.5	1.27911
5	0.9	0.80472
8	0.1	1.24579
8	0.3	1.16682
8	0.5	0.9768
8	0.9	0.97407

Plot between M and validation error

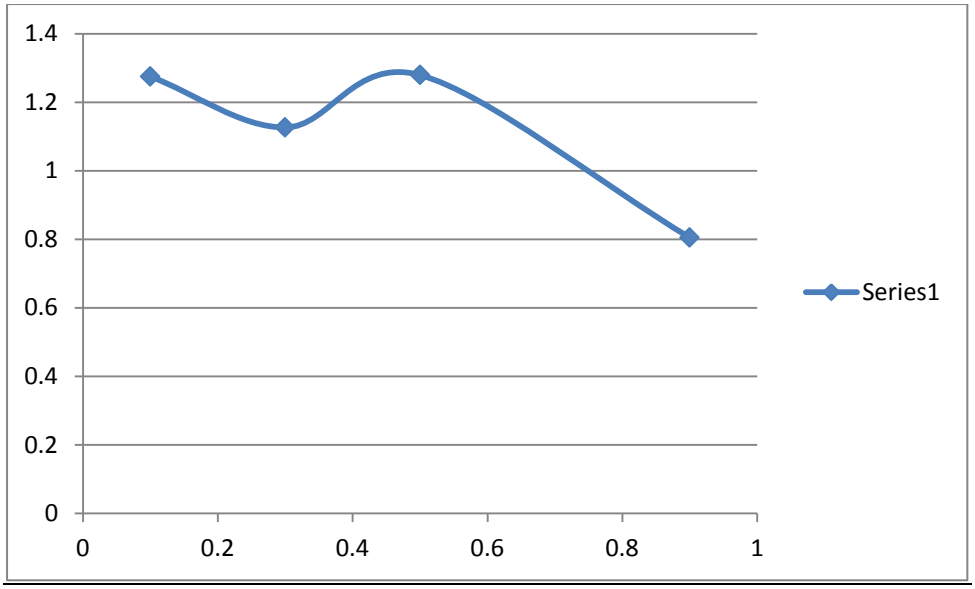
lambda(min) =0.9			
M(X-axis)	3	5	8
Validation rms error(Y-axis	1.17993	0.80472	0.97407



Inference : M(minimum) = 5 for minimum validation error = 0.80472

Plot between lambda and Validation Rms Error

M = 5(minimum)				
lambda (X-axis)	0.1	0.3	0.5	0.9
Validation rms error(Y-axis	1.27484	1.12645	1.27911	0.80472



Inference : lamda(minimum) = 0.9 for minimum validation error = 0.80472

Muj and Sigmaj :

Muj:

The Muj Matrix contains 'M' rows which are drawn randomly from the corresponding partition are found using the following function mujmat = featuresmat[np.random.choice(featuresmat.shape[0],M, replace=False),:] . It contains M number of rows and columns of this matrix is equal to the number of features . The Basis Function is calculated using the MujMat .The Basis function Matrix contains M columns and number of rows is equal to size of the data set . The first column in the basis function is '1' for every row of the Basis Function Matrix

SigmaJ :

The variance of every column in given input data set is calculated . Variance is divided by 10 as mentioned in main.pdf project specification document(to maintain proportionality) . If the variance value is equal to zero , it is substituted by a very small value i.e.0.001 so as to prevent Sigmaj from becoming a singular matrix. This variance value of every column is taken as diagonal element in the diagonal matrix . The dimension of this matrix is ' Number of columns by Number of columns' .

Using the numpy.linalg.inv() function the inverse of the sigma matrix is calculated which is used in the calculation of the basis function matrix

Learning Rate (eta):

In stochastic Gradient Descent , for both cases i.e. letor data and synthetic data ,the initial learning rate value is chosen as '1' as suggested in the recitation class . Though the code is present to handle the division of eta by2 when the present rms error is greater than previous rms error , it has been observed that convergence was achieved in few iterations and there was never a situation where eta has to be divided . Hence eta (learning rate) was kept constant through out for Stochastic Gradient descent in Letor or Synthetic data cases.

Evaluation Results :

Closed Form Solution For Letor Data

M=5
lambda_reg=0.1

letor_trainingrmserror=[[0.5718]]
letor_validationrmserror=[[0.57207]]
minwweight=
[[0.31438085]
 [0.62327833]
 [0.62329014]
 [0.62329149]
 [-0.28580159]]
M=5
lambda_reg=0.3
letor_trainingrmserror=[[0.57182]]
letor_validationrmserror=[[0.57172]]
minwweight=
[[0.31442785]
 [-0.25100952]
 [-0.24186757]
 [-0.26611259]
 [-0.24250344]]
M=5
lambda_reg=0.5
letor_trainingrmserror=[[0.57179]]
letor_validationrmserror=[[0.57172]]
M=5
lambda_reg=0.9
letor_trainingrmserror=[[0.57181]]
letor_validationrmserror=[[0.57174]]
M=7
lambda_reg=0.1
letor_trainingrmserror=[[0.57181]]
letor_validationrmserror=[[0.57174]]
M=7
lambda_reg=0.3
letor_trainingrmserror=[[0.57178]]
letor_validationrmserror=[[0.57176]]
M=7
lambda_reg=0.5
letor_trainingrmserror=[[0.57181]]
letor_validationrmserror=[[0.57167]]
minwweight=
[[0.3144177]
 [-0.21118822]
 [-0.2096118]
 [-0.20961558]
 [-0.2096118]
 [-0.20959667]
 [0.45704477]]
M=7
lambda_reg=0.9
letor_trainingrmserror=[[0.57177]]
letor_validationrmserror=[[0.57216]]
M=9
lambda_reg=0.1
letor_trainingrmserror=[[0.57179]]
letor_validationrmserror=[[0.57179]]
M=9
lambda_reg=0.3
letor_trainingrmserror=[[0.57172]]
letor_validationrmserror=[[0.57239]]
M=9
lambda_reg=0.5
letor_trainingrmserror=[[0.57181]]
letor_validationrmserror=[[0.57182]]
M=9

lambda_reg=0.9
letor_trainingrmserror=[[0.57181]]
letor_validationrmserror=[[0.57175]]
M=11
lambda_reg=0.1
letor_trainingrmserror=[[0.57178]]
letor_validationrmserror=[[0.57211]]
M=11
lambda_reg=0.3
letor_trainingrmserror=[[0.57176]]
letor_validationrmserror=[[0.57242]]
M=11
lambda_reg=0.5
letor_trainingrmserror=[[0.57178]]
letor_validationrmserror=[[0.57192]]
M=11
lambda_reg=0.9
letor_trainingrmserror=[[0.57115]]
letor_validationrmserror=[[0.57186]]
M=13
lambda_reg=0.1
letor_trainingrmserror=[[0.57179]]
letor_validationrmserror=[[0.57183]]
M=13
lambda_reg=0.3
letor_trainingrmserror=[[0.57178]]
letor_validationrmserror=[[0.57221]]
M=13
lambda_reg=0.5
letor_trainingrmserror=[[0.57115]]
letor_validationrmserror=[[0.57198]]
M=13
lambda_reg=0.9
letor_trainingrmserror=[[0.57122]]
letor_validationrmserror=[[0.57214]]
M=15
lambda_reg=0.1
letor_trainingrmserror=[[0.57179]]
letor_validationrmserror=[[0.57167]]
M=15
lambda_reg=0.3
letor_trainingrmserror=[[0.57179]]
letor_validationrmserror=[[0.57173]]
M=15
lambda_reg=0.5
letor_trainingrmserror=[[0.57156]]
letor_validationrmserror=[[0.57244]]
M=15
lambda_reg=0.9
letor_trainingrmserror=[[0.57171]]
letor_validationrmserror=[[0.57238]]

min_rmserror=[0.57167]
min_lambda=0.5
min_M= 7

Final_M=7
Final_lambda=0.5
Final minimum Validation_rmserror=[[0.57167]]
Final Weight =
[[0.3144177]
 [-0.21118822]
 [-0.2096118]

[-0.20961558]
[-0.2096118]
[-0.20959667]
[0.45704477]]

FinalTesting rms Error = 0.56787

Explanantion :

The above highlighted results shows that the testing rms error is lower than minimum validation rms error for Letor Data using Closed Form Solution which is desirable as the Testing Rms Error should be lower . When above M ,lambda and Weights are used ,57% of the predictions are incorrect while 43% of predictions are correct which shows this model is better comparing to the below mentioned models , but this is not the excellent prediction

Closed Form Solution For Synthetic Data :

M=3
lambda_reg=0.1
synth_trainingrmserror=[[0.79047]]
synth_validationrmserror=[[0.78623]]
minweight=
[[0.96074456]
[0.03568897]
[0.95000236]]
M=3
lambda_reg=0.3
synth_trainingrmserror=[[0.79046]]
synth_validationrmserror=[[0.78609]]
minweight=
[[0.9607907]
[0.79943254]
[-0.73913073]]
M=3
lambda_reg=0.5
synth_trainingrmserror=[[0.79053]]
synth_validationrmserror=[[0.78667]]
M=3
lambda_reg=0.9
synth_trainingrmserror=[[0.79052]]
synth_validationrmserror=[[0.78692]]
M=5
lambda_reg=0.1
synth_trainingrmserror=[[0.79048]]
synth_validationrmserror=[[0.78685]]
M=5
lambda_reg=0.3
synth_trainingrmserror=[[0.7904]]
synth_validationrmserror=[[0.78849]]
M=5
lambda_reg=0.5
synth_trainingrmserror=[[0.79042]]
synth_validationrmserror=[[0.78748]]
M=5
lambda_reg=0.9
synth_trainingrmserror=[[0.7905]]
synth_validationrmserror=[[0.78707]]
M=8
lambda_reg=0.1
synth_trainingrmserror=[[0.79029]]
synth_validationrmserror=[[0.79003]]
M=8
lambda_reg=0.3

synth_trainingrmserror=[[0.79033]]
synth_validationrmserror=[[0.78756]]
M=8
lambda_reg=0.5
synth_trainingrmserror=[[0.79039]]
synth_validationrmserror=[[0.7886]]
M=8
lambda_reg=0.9
synth_trainingrmserror=[[0.79044]]
synth_validationrmserror=[[0.78727]]

min_rmserror=[0.78609]
min_lambda=0.3
min_M= 3

Final minimum validation rmserror =[[0.78609]]
Final lambda=0.3
Final M=3
Final weight=[[0.96074456]
[0.03568897]
[0.95000236]]

Final Testing rmserror = [0.80026]

Explanantion :
The above highlighted results shows that the testing rms error is lower than minimum validation rms error for Synthetic Data using Closed Form Solution which is not desirable as the Testing Rms Error should be lower .The prediction accuracy would have been better . When above M ,lambda and Weights are used ,80% of the predictions are incorrect while 20% of predictions are correct which shows this model needs more data samples for sufficient training

Stochastic Gradient Solution For Letor Data :

Convergence is achieved
M=5
lambda_reg=0.1
letor_trainingrmserror=[1.29827]
letor_validationrmserror=[1.29495]
minwweight=
[-0.85098548 0.6410765 0.31364313 0.18663343 0.50722349]
Convergence is achieved
M=5
lambda_reg=0.3
letor_trainingrmserror=[0.76751]
letor_validationrmserror=[0.7637]
minwweight=
[-0.19696979 0.6817619 0.52301359 0.26289888 0.18633128]
Convergence is achieved
M=5
lambda_reg=0.5
letor_trainingrmserror=[0.71615]
letor_validationrmserror=[0.71221]
minwweight=
[-0.11602345 0.31574379 0.34443905 0.2230774 0.31839392]
Convergence is achieved
M=5

lambda_reg=0.9
letor_trainingrmserror=[1.2009]
letor_validationrmserror=[1.19777]
Convergence is achieved
M=7
lambda_reg=0.1
letor_trainingrmserror=[0.68233]
letor_validationrmserror=[0.67821]
minwweight=
[-0.05706061 0.31274085 0.63169951 0.41235758 0.11172135 0.79176589
0.55040891]
Convergence is achieved
M=7
lambda_reg=0.3
letor_trainingrmserror=[0.78063]
letor_validationrmserror=[0.77685]
Convergence is achieved
M=7
lambda_reg=0.5
letor_trainingrmserror=[0.69135]
letor_validationrmserror=[0.68723]
Convergence is achieved
M=7
lambda_reg=0.9
letor_trainingrmserror=[0.95021]
letor_validationrmserror=[0.94671]
Convergence is achieved
M=9
lambda_reg=0.1
letor_trainingrmserror=[1.4022]
letor_validationrmserror=[1.39909]
Convergence is achieved
M=9
lambda_reg=0.3
letor_trainingrmserror=[0.69881]
letor_validationrmserror=[0.69484]
Convergence is achieved
M=9
lambda_reg=0.5
letor_trainingrmserror=[0.67384]
letor_validationrmserror=[0.66973]
minwweight=
[-0.04115007 0.39376918 0.23272234 0.13661678 0.43999043 0.05327082
0.15255113 0.22347034 0.10971165]
Convergence is achieved
M=9
lambda_reg=0.9
letor_trainingrmserror=[0.86027]
letor_validationrmserror=[0.85657]
Convergence is achieved
M=11
lambda_reg=0.1
letor_trainingrmserror=[0.68192]
letor_validationrmserror=[0.67792]
Convergence is achieved
M=11
lambda_reg=0.3
letor_trainingrmserror=[0.71381]
letor_validationrmserror=[0.70981]
Convergence is achieved
M=11
lambda_reg=0.5
letor_trainingrmserror=[0.72537]

letor_validationrmserror=[0.72145]
Convergence is achieved
M=11
lambda_reg=0.9
letor_trainingrmserror=[0.8]
letor_validationrmserror=[0.79617]
Convergence is achieved
M=13
lambda_reg=0.1
letor_trainingrmserror=[0.66293]
letor_validationrmserror=[0.65897]
minwweight=
[-0.02001276 0.0771293 0.10899979 0.65586121 0.39351467 0.7837358
0.40337427 0.05893762 0.50965642 0.49628657 0.11680944 0.08320048
0.77428499]
Convergence is achieved
M=13
lambda_reg=0.3
letor_trainingrmserror=[0.79115]
letor_validationrmserror=[0.78726]
Convergence is achieved
M=13
lambda_reg=0.5
letor_trainingrmserror=[0.91276]
letor_validationrmserror=[0.90892]
Convergence is achieved
M=13
lambda_reg=0.9
letor_trainingrmserror=[0.70546]
letor_validationrmserror=[0.7014]
Convergence is achieved
M=15
lambda_reg=0.1
letor_trainingrmserror=[0.687]
letor_validationrmserror=[0.68309]
Convergence is achieved
M=15
lambda_reg=0.3
letor_trainingrmserror=[0.71084]
letor_validationrmserror=[0.70674]
Convergence is achieved
M=15
lambda_reg=0.5
letor_trainingrmserror=[0.87381]
letor_validationrmserror=[0.86993]
Convergence is achieved
M=15
lambda_reg=0.9
letor_trainingrmserror=[0.75435]
letor_validationrmserror=[0.75042]

min_rmserror=[0.65897]
min_lambda=0.1
min_M= 13

Final_M=13
Final_lambda=0.1
Final minimum Validation_rmserror=[0.65897]
Final Weight =
[-0.02001276 0.0771293 0.10899979 0.65586121 0.39351467 0.7837358
0.40337427 0.05893762 0.50965642 0.49628657 0.11680944 0.08320048
0.77428499]

FinalTesting _rms Error = [0.66016]

Explanation:
The above highlighted results shows that the testing rms error is higher than minimum validation rms error for Letor Data using Stochastic Gradient Solution which is not desirable as the Testing Rms Error should be lower .This happened even after doing a random shuffle . However the difference between them is minimum .The prediction accuracy is of average standard . When above M ,lambda and Weights are used ,66% of the predictions are incorrect while 34% of predictions are correct which shows this model needs more data samples for sufficient training

Stochastic Gradient Solution For Synthetic Data :

Convergence is achieved
M=3
lambda_reg=0.1
synth_trainingrmserror=[1.24981]
synth_validationrmserror=[1.25442]
minweight=
[1.93089273 0.73667915 0.30937971]
Convergence is achieved
M=3
lambda_reg=0.3
synth_trainingrmserror=[1.26415]
synth_validationrmserror=[1.26854]
Convergence is achieved
M=3
lambda_reg=0.5
synth_trainingrmserror=[1.00499]
synth_validationrmserror=[1.00844]
minweight=
[1.5837407 0.39630155 0.2337651]
Convergence is achieved
M=3
lambda_reg=0.9
synth_trainingrmserror=[1.17548]
synth_validationrmserror=[1.17993]
Convergence is achieved
M=5
lambda_reg=0.1
synth_trainingrmserror=[1.27035]
synth_validationrmserror=[1.27484]
Convergence is achieved
M=5
lambda_reg=0.3
synth_trainingrmserror=[1.12233]
synth_validationrmserror=[1.12645]
Convergence is achieved
M=5
lambda_reg=0.5
synth_trainingrmserror=[1.2738]
synth_validationrmserror=[1.27911]
Convergence is achieved
M=5
lambda_reg=0.9
synth_trainingrmserror=[0.80403]
synth_validationrmserror=[0.80472]
minweight=
[1.11208520e+00 1.00175337e-03 7.56676761e-02 9.09355132e-03
9.18295757e-02]
Convergence is achieved

M=8
lambda_reg=0.1
synth_trainingrmserror=[1.23979]
synth_validationrmserror=[1.24579]
Convergence is achieved
M=8
lambda_reg=0.3
synth_trainingrmserror=[1.16185]
synth_validationrmserror=[1.16682]
Convergence is achieved
M=8
lambda_reg=0.5
synth_trainingrmserror=[0.97364]
synth_validationrmserror=[0.9768]
Convergence is achieved
M=8
lambda_reg=0.9
synth_trainingrmserror=[0.97065]
synth_validationrmserror=[0.97407]

min_rmserror=[0.80472]
min_lambda=0.9
min_M= 5

Final_lambda=0.9
Final_M=5
Final minimum Validation rmserror=[0.80472]

Final weight = [1.11208520e+00 1.00175337e-03 7.56676761e-02 9.09355132e-03
9.18295757e-02]

FinalTesting Rms_Error = 0.80399

Explanation:
The above highlighted results shows that the testing rms error is lower than minimum validation rms error for Synthetic Data using Stochastic Gradient Solution which is desirable as the Testing Rms Error should be lower .The prediction accuracy would have been better . When above M ,lambda and Weights are used ,80% of the predictions are incorrect while 20% of predictions are correct which shows this model needs more data samples for sufficient training