Learning to Rank using Linear Regression

Manishkumarreddy Jarugu

50206843

mjarugu@buffalo.edu

Abstract:

The main goal of this project is to use machine learning linear regression techniques to solve the Learning to Rank (LeToR) problems on a real dataset and a synthetic dataset. We train our model using the training set which comprises of 80% of the given data set and validate it using the 20 % validation data set and finally test it using the 20% testing data set.

Introduction:

In linear regression, the relationships are modeled using linear predictor functions whose unknown model parameters are estimated from the data. In our project, the linear regression model is being implemented using closed form solution and we are trying to optimize the values by tuning the hyper parameters.

The linear regression function is

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

where w is the weight vector which has to be learned

and the Gaussian radial basis function is shown below:

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

Including regularization yields,

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

We also need to calculate the stochastic gradient descent using the formula:

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

Where

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

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

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

Finally, its tested using the root mean square error as

$$E_{\mathrm{RMS}} = \sqrt{2E(\mathbf{w}^*)/N_{\mathrm{V}}}$$

Data Set Partition:

LeToR Data Set Partition:

The LeToR Data set has 69,623 rows with 46 features for each row. The data has been processed in the python as shown below:

```
#The following commented for the actual file from the microsoft server
with open("Querylevelnorm.txt") as data_file:
    for line in data_file:
        temparr=[]
        count=count+1
        Arr = line.split(" ")
        t.append(float(Arr[0]))
        for i in range(2,48):
            temp = Arr[i].split(":")
            temparr.append(float(temp[1]))
        X.append(temparr)
```

Later on the LeToR data set was preprocessed and gave as csv files, which are loaded as shown below:

```
firstfile  = open('Querylevelnorm_X.csv', "rU")
firstreader = csv.reader(firstfile)
for row in firstreader:
    temm=[]
    for col in row:
        temm.append(float(col))
    X.append(temm)

secondfile= open('Querylevelnorm_t.csv', "rU")
secondreader = csv.reader(secondfile)
for row in secondreader:
    t.append(float(row[0]))
```

The initial 80% of the rows are considered as the training set, the next 10% are considered as the validation set and the final 10% are considered as the testing set. The corresponding data splitting is shown below:

```
npAllX=np.array(X)
Num=len(npAllX)
lenX=int(Num*0.8)
npX=npAllX[0:lenX]
val_npX=npAllX[lenX:lenX+int(0.1*Num)]
tes_npX=npAllX[lenX+int(0.1*Num):Num]

nptAll=np.array(t)
npt=nptAll[0:lenX]
val_npt=nptAll[lenX:lenX+int(0.1*Num)]
tes_npt=nptAll[lenX+int(0.1*Num):Num]
```

For Synthetic data set:

Similarly, for the synthetic data set, the given input.csv and output.csv are being loaded in python using the csv file loader and are processed. And also the data is split into 3 sections for training, validation and testing with 80, 10 and 10 percentages respectively for the corresponding sections. The python code implementing the same is shown below:

For input.csv:

```
ifile  = open('input.csv', "rU")
reader = csv.reader(ifile)
Syn_X_all=[]
for row in reader:
    temm=[]
    for col in row:
        temm.append(float(col))
    Syn_X_all.append(temm)
np_Syn_X_all=np.array(Syn_X_all)
np_Syn_X=np_Syn_X_all[0:int(len(np_Syn_X_all)*(0.8))]
val_np_Syn_X=np_Syn_X_all[int(len(np_Syn_X_all)*(0.8)):int(len(np_Syn_X_all)*(0.9))]
tes_np_Syn_X=np_Syn_X_all[int(len(np_Syn_X_all)*(0.9)):len(np_Syn_X_all)]
```

Similarly for output.csv:

```
ifile_o  = open('output.csv', "rU")
reader_o = csv.reader(ifile_o)
Syn_t_all=[]
for row in reader_o:
    Syn_t_all.append(float(row[0]))
np_Syn_t_all=np.array(Syn_t_all)
np_Syn_t=np_Syn_t_all[0:int(len(np_Syn_t_all)*(0.8))]
val_np_Syn_t=np_Syn_t_all[int(len(np_Syn_t_all)*(0.8)):int(len(np_Syn_t_all)*(0.9))]
tes_np_Syn_t=np_Syn_t_all[int(len(np_Syn_t_all)*(0.9)):len(np_Syn_t_all)]
```

Thus the given data set is successfully split into

- Training set – 80%
- Validation set – 10%
- Testing set – 10%

Hyper parameter Tuning:

For M and λ:

The best method for choosing the values for M and λ are trial and error method. Initially a value of M and λ is being selected and then, the value of M is kept constant and the value of λ is changed over a range and the same process is being repeated for different values of M.

The main objective is to reduce the root mean square error of the validation set

Validation root mean square for LeToR:

M / λ	0.01	0.05	0.1	0.15	0.25	0.35	0.5
2	0.552950332226	0.552951915491	0.552953888716	0.552955855476	0.55295976978	0.552963658754	0.552969445471
4	0.552950332226	0.552951915491	0.552953888716	0.552955855476	0.55295976978	0.552963658754	0.552969445471
6	0.552950332226	0.552951915491	0.552953888716	0.552955855476	0.55295976978	0.552963658754	0.552969445471
8	0.552950332226	0.552951915491	0.552953888716	0.552955855476	0.55295976978	0.552963658754	0.552969445471
10	0.552950332226	0.552951915491	0.552953888716	0.552955855476	0.55295976978	0.552963658754	0.552969445471
12	0.552950332226	0.552951915491	0.552953888716	0.552955855476	0.55295976978	0.552963658754	0.552969445471
14	0.552950332226	0.552951915491	0.552953888716	0.552955855476	0.55295976978	0.552963658754	0.552969445471
16	0.552950332226	0.552951915491	0.552953888716	0.552955855476	0.55295976978	0.552963658754	0.552969445471

SGD validation root mean square error for LeToR:

M / λ	0.01	0.05	0.1	0.15	0.25	0.35	0.5
2	0.566869312151	0.565565952783	0.564739834612	0.564406149085	0.564781425873	0.566101503304	0.56910009271
4	0.567346231397	0.56555903823	0.564740197552	0.564406132955	0.564781425867	0.566101503304	0.56910009271
6	0.566639926989	0.56556614186	0.5647396514	0.564406097985	0.564781425868	0.566101503304	0.56910009271
8	0.566796787282	0.565572287126	0.564740304495	0.564406141412	0.564781425837	0.566101503304	0.56910009271
10	0.56705466021	0.565561019734	0.564740438979	0.564406102966	0.564781425828	0.566101503304	0.56910009271
12	0.567024344147	0.565564157065	0.564739847065	0.564406154582	0.564781425848	0.566101503304	0.56910009271
14	0.566615282005	0.565553023072	0.564739405078	0.564406106722	0.56478142587	0.566101503304	0.56910009271
16	0.566754299209	0.565547730639	0.564740294215	0.564406146625	0.564781425837	0.566101503304	0.56910009271

Validation root mean square for Synthetic data:

M / λ	0.01	0.05	0.1	0.15	0.25	0.35	0.5
2	0.785353245364	0.785365069775	0.785379885986	0.785394703889	0.785424338261	0.785453969171	0.785498408236
4	0.785353272855	0.785365075201	0.785379888697	0.785394705696	0.785424339346	0.785453969947	0.78549840878
6	0.785353593131	0.785365131906	0.785379916605	0.785394724211	0.785424350419	0.785453977851	0.785498414315
8	0.785353589286	0.785365131136	0.785379916219	0.785394723953	0.785424350265	0.785453977741	0.785498414238
10	0.78534996718	0.785364404888	0.785379552971	0.785394481755	0.785424204924	0.785453873916	0.785498341552
12	0.78534997848	0.785364407148	0.785379554101	0.785394482509	0.785424205377	0.785453874239	0.785498341778
14	0.785349979842	0.785364407348	0.785379554197	0.785394482571	0.785424205414	0.785453874266	0.785498341796
16	0.785349760711	0.785364363498	0.78537953226	0.785394467939	0.785424196625	0.785453867981	0.78549833739

SGD validation root mean square error for Synthetic data:

M / λ	0.01	0.05	0.1	0.15	0.25	0.35	0.5
2	0.792000251589	0.795388012366	0.801494251866	0.80907347969	0.826856963143	0.846195949187	0.875269342913
4	0.79202884188	0.795393550366	0.801502537288	0.809073836841	0.8268569771	0.846195939854	0.875269342415
6	0.792121301228	0.795438957216	0.801506145227	0.809079440005	0.826857290313	0.846196004713	0.875269344752
8	0.792155756525	0.795412410885	0.801514060854	0.80907274515	0.826856838628	0.846195974566	0.875269342446
10	0.792108212996	0.795392532347	0.801495944288	0.809070134635	0.826856446699	0.846195888774	0.875269332627
12	0.791948713205	0.795334247845	0.80148536769	0.809066359507	0.826855622123	0.846195809453	0.87526933461
14	0.792073030476	0.795377909736	0.801472728799	0.809072392584	0.826856460371	0.846195760471	0.875269332262
16	0.792075582234	0.795377101219	0.801482131842	0.809070960261	0.82685604193	0.846195732286	0.875269328997

From all these tables, for the minimum root mean square, the value of M and λ are being selected.

Hence the best value of M and λ for the LeToR data set is

M= 14

λ= 0.01

Similarly, one of the best value for M and λ for synthetic data set is

M=2

λ=0.01

For mean(μ) and Sigma(Σ):

We need to select random M means, where M is the complexity and each mean is a vector with number of features in it. The mean matrix is actually calculated by selecting M-1 data sets from the given data set in a random manner. This way we calculate mean so easily and the time taken for selecting these mean is relatively low. The code for achieving the same is given below

```
rand=random.sample(range(1,len(npX)), M-1)
Mu=[]
for i in range(0,len(rand)):
    Mu.append(X[i])
npMu=np.array(Mu)
```

The value of sigma is being calculated as its mentioned in the project document. This way of calculation of sigma ensures a lot of time is being saved on computation. We first calculate the variance and by representing the variance in the diagonal matrix we get sigma. Dimensions of sigma is F X F where F is the number of features.

We calculate sigma as follows:

$$\Sigma = \begin{pmatrix} \sigma_1^2 & & & \\ & \sigma_2^2 & & \\ & & \ddots & \\ & & & \sigma_D^2 \end{pmatrix}$$

Where,

$$\sigma_i^2 = \frac{1}{10}\mathrm{var}_i\left(\mathbf{x}\right)$$

This way we calculate the sigma matrix.

For η:

The value of η has been kept fixed for our observation. But for a fixed value of M and lambda, if a sweep is being applied to η, the root mean square values can be calculated and they are tabulated below:

η	0.01	0.02	0.05	0.1	0.2	0.3
Erms	0.568034327904	0.565164995197	0.573052601803	0.606801801909	0.637722821872	0.642291141616

So the root mean square is minimum when η=0.02 for LeToR Data set

η	0.01	0.02	0.05	0.1	0.2	0.3
Erms	0.792101497555	0.799993577602	0.824531302457	0.867493206809	0.963717140687	1.057616399556

So the root mean square is minimum when η=0.01 for the Synthetic data set

Evaluation and Results:

Output:

```
The actual Erms value is
0.563797174585
The value of Validation Erms is
0.552950332226
The value of test Erms is
0.552910814364


The actual value of SGD_Erms for training set is
0.564482758649
The value of SGD_Erms for validation set is
0.566869312151
The value of SGD_Erms for testing set is
0.695204640981


The actual Syn_Erms value for training set is
0.790263715328
The value of Validation Erms for Synthesised data is
0.785353245364
The value of test Erms for the synthesised set is
0.785353245364


The value of Syn_SGD_Erms for training set is
0.79190089587
The value of SGD_Erms for validation set  in synthesided data is
0.792000251589
The value of SGD_Erms for testing set in Synthesised data is
```

0.794098935358
end of M=2

The actual Erms value is
0.563782152069
The value of Validation Erms is
0.552950332226
The value of test Erms is
0.552910814364

The actual value of SGD_Erms for training set is
0.564483095796
The value of SGD_Erms for validation set is
0.567346231397
The value of SGD_Erms for testing set is
0.695620232336

The actual Syn_Erms value for training set is
0.790227166085
The value of Validation Erms for Synthesised data is
0.785353272855
The value of test Erms for the synthesised set is
0.785353272851

The value of Syn_SGD_Erms for training set is
0.791910017667
The value of SGD_Erms for validation set in synthesided data is
0.79202884188
The value of SGD_Erms for testing set in Synthesised data is
0.79356837064
end of M=4

The actual Erms value is
0.563773127924
The value of Validation Erms is
0.552950332226
The value of test Erms is
0.552910814364

The actual value of SGD_Erms for training set is
0.564482360245
The value of SGD_Erms for validation set is
0.566639926989
The value of SGD_Erms for testing set is
0.695016252483

The actual Syn_Erms value for training set is
0.790153746157
The value of Validation Erms for Synthesised data is
0.785353593131
The value of test Erms for the synthesised set is
0.785353592666

The value of Syn_SGD_Erms for training set is
0.791927559073
The value of SGD_Erms for validation set in synthesided data is
0.792121301228
The value of SGD_Erms for testing set in Synthesised data is
0.793272353713
end of M=6

The actual Erms value is
0.563764103553
The value of Validation Erms is
0.552950332226
The value of test Erms is
0.552910814364

The actual value of SGD_Erms for training set is
0.564482519183
The value of SGD_Erms for validation set is
0.566796787282
The value of SGD_Erms for testing set is
0.695158199136

The actual Syn_Erms value for training set is
0.790111296591
The value of Validation Erms for Synthesised data is
0.785353589286
The value of test Erms for the synthesised set is
0.785353588821

The value of Syn_SGD_Erms for training set is
0.791924748796
The value of SGD_Erms for validation set in synthesided data is
0.792155756525
The value of SGD_Erms for testing set in Synthesised data is
0.79386876962
end of M=8

The actual Erms value is
0.563755078956

The value of Validation Erms is
0.552950332226
The value of test Erms is
0.552910814364

The actual value of SGD_Erms for training set is
0.564482836048
The value of SGD_Erms for validation set is
0.56705466021
The value of SGD_Erms for testing set is
0.695382107649

The actual Syn_Erms value for training set is
0.790027058602
The value of Validation Erms for Synthesised data is
0.78534996718
The value of test Erms for the synthesised set is
0.785349989864

The value of Syn_SGD_Erms for training set is
0.791913169598
The value of SGD_Erms for validation set in synthesided data is
0.792108212996
The value of SGD_Erms for testing set in Synthesised data is
0.794016777662
end of M=10

The actual Erms value is
0.563752035121
The value of Validation Erms is
0.552950332226
The value of test Erms is
0.552910814364

The actual value of SGD_Erms for training set is
0.564482825403
The value of SGD_Erms for validation set is
0.567024344147
The value of SGD_Erms for testing set is
0.695352778817

The actual Syn_Erms value for training set is
0.789942981251
The value of Validation Erms for Synthesised data is
0.78534997848

The value of test Erms for the synthesised set is
0.785350001166

The value of Syn_SGD_Erms for training set is
0.791864098896
The value of SGD_Erms for validation set in synthesided data is
0.791948713205
The value of SGD_Erms for testing set in Synthesised data is
0.795412956606
end of M=12

The actual Erms value is
0.563743010382
The value of Validation Erms is
0.552950332226
The value of test Erms is
0.552910814364

The actual value of SGD_Erms for training set is
0.564482410977
The value of SGD_Erms for validation set is
0.566615282005
The value of SGD_Erms for testing set is
0.695010117078

The actual Syn_Erms value for training set is
0.789906451798
The value of Validation Erms for Synthesised data is
0.785349979842
The value of test Erms for the synthesised set is
0.785350002524

The value of Syn_SGD_Erms for training set is
0.791898950994
The value of SGD_Erms for validation set in synthesided data is
0.792073030476
The value of SGD_Erms for testing set in Synthesised data is
0.794136392106
end of M=14

The actual Erms value is
0.563721445703
The value of Validation Erms is
0.552950332226
The value of test Erms is

0.552910814364

The actual value of SGD_Erms for training set is
0.564482351889
The value of SGD_Erms for validation set is
0.566754299209
The value of SGD_Erms for testing set is
0.695144573191

The actual Syn_Erms value for training set is
0.789869872696
The value of Validation Erms for Synthesised data is
0.785349760711
The value of test Erms for the synthesised set is
0.785349783397

The value of Syn_SGD_Erms for training set is
0.791897608813
The value of SGD_Erms for validation set in synthesided data is
0.792075582234
The value of SGD_Erms for testing set in Synthesised data is
0.79378587863
end of M=16

The actual Erms value is
0.563718869639
The value of Validation Erms is
0.552950332226
The value of test Erms is
0.552910814364

The actual value of SGD_Erms for training set is
0.564482752577
The value of SGD_Erms for validation set is
0.567176130764
The value of SGD_Erms for testing set is
0.695507800001

The actual Syn_Erms value for training set is
0.789827839547
The value of Validation Erms for Synthesised data is
0.785349737838
The value of test Erms for the synthesised set is
0.785349760524

```
The value of Syn_SGD_Erms for training set is
0.791882151736
The value of SGD_Erms for validation set  in synthesided data is
0.791999385846
The value of SGD_Erms for testing set in Synthesised data is
0.793877209478
end of M=18
```

In all of the above results, the value of the root mean square in the testing set is comparable to the value of the root mean square of the training set and the validation set. We made a series of changes to the M and lambda and finally took a value and normalized it. So the value with the minimum root mean square is being taken and fixed.

Hence, the final output is

```
C:/Users/manis/PycharmProjects/LinearRegression/main.py
The value of M is choosen 12
the value of lambda is choosen 0.01
The actual Erms value is
0.563752035121
The value of Validation Erms is
0.552950332226
The value of test Erms is
0.552910814364


The actual value of SGD_Erms for training set is
0.564482684185
The value of SGD_Erms for validation set is
0.566969193148
The value of SGD_Erms for testing set is
0.695317608501


The value of M for the synthetic data is choosen 5
The value of lambda for the synthetic data is choosen 0.01
The actual Syn_Erms value for training set is
0.790190213872
The value of Validation Erms for synthetic  data is
0.785353587836
The value of test Erms for the synthetic set is
0.785353587371


The value of Syn_SGD_Erms for training set is
```

```
0.791909052516
The value of SGD_Erms for validation set  in synthetic data is
0.792046199389
The value of SGD_Erms for testing set in synthetic  data is
0.793835234189
The end of the program
```