CSE 574: Introduction to Machine Learning (Fall 2017)

Saleem Ahmed
Person Number : 50247637
University at Buffalo

Problem Statement

The goal of this project is to use machine learning to solve a problem that arises in Information Retrieval, one known as the Learning to Rank (LeToR) problem. We formulate this as a problem of linear regression where we map an input vector x to a real-valued scalar target y(x,w).

There are four tasks:

1. Train a linear regression model on LeToR dataset using a closed-form solution.
2. Train a linear regression model on the LeToR dataset using stochastic gradient descent (SGD).
3. Train a linear regression model on a synthetic dataset using a closed-form solution.
4. Train a linear regression model on the synthetic dataset using SGD.

Closed Form Solution of Letor Training Set

```
### Performing kmeans on training input - Letor
kmeans_t_letor = computing_KMeans_clusters(k_forK_means, letor_ip_train)
train_input_cntr_syn = get_data_centers(kmeans_t_letor)
# print(np.shape(train_input_cntr_syn))
# print(train_input_cntr_syn)

### Center value arrays for training input Letor
train_input_cntr_letor = get_data_centers(kmeans_t_letor)
# print(np.shape(train_input_cntr_letor))
# print(train_input_cntr_letor)

### Generating Letor Training Cluster Dictionary
letor_ip_train_clusters_dict = get_cluster_dictionary(letor_ip_train, kmeans_t_letor)
letor_ip_train_clusters_dict

### Caculating Spreads for Synthetic Training Data
train_input_cov_letor = get_spreads(letor_ip_train_clusters_dict)
train_input_cov_letor[0]

### Finding design matrix for training input set Letor Training
phi_letor_t = compute_design_matrix(letor_ip_train, train_input_cntr_letor[:, None] , train_input_cov_letor)

### Calculating Wml - weight for maximum likelihood from the closed form sol for Letor
wml_letor = np.array(closed_form_sol(l_forLambda, phi_letor_t, letor_op_train)).reshape(1,-1)

### Getting predicted Values for Letor Training Input
prediction_set_t_letor = get_predicted_value (phi_letor_t, wml_letor)
#Getting Learning Error for Letor Data :
print("RMS Error of LETOR Training  SET  Closed Form  :  ", np.sqrt(mean_squared_error(letor_op_train, prediction_set_t_letor
```

```
RMS Error of LETOR Training  SET  Closed Form  :   0.539133334174
RMS Error of LETOR Validation SET  Closed Form  :   1.60940745482
RMS Error of LETOR Testing SET  Closed Form  :   0.954359423361
CLOSED FORM Solution For Letor Training Set
 [-0.37400409  4.55098667 -1.3233613   5.73696248 -1.83813829 -2.59197784
  0.11310325  0.99103294  3.9666247   2.01065357  0.03477407 -1.71052883
  0.90547698 -1.28668244 -0.13762361 -0.23554149 -3.01642492  0.80271825
  1.89873321 -1.91926409 -2.80511698 -0.0142868  -1.01352134 -6.16497662
  1.38277188 -2.56475399  4.30388974  0.73388473 -3.0272224  -0.31894984
  2.17003724  0.78292043  0.14055272  1.36759293  6.22409881 -0.3755304
  4.94536553 -2.99698066 -0.32561414 -3.0842482  -2.61817457  2.58934122
 -2.13600311  3.78766384 -2.99521629 -0.46990337 -1.83429755 -3.21966821
  0.17945114  0.70688103  0.57211058 -0.23280972 -2.84367887  0.14358031
  0.52194434  2.17762923 -5.35261728  0.93955985  1.64519492 -2.95509844
 -2.50286501  3.06231868  1.23129252 -2.49130588 -0.4853414  -3.31207985
  2.46076686  2.26721239  2.77709611  0.04330882  1.7098374   3.49191018
 -1.0162473   0.27365599  0.20553953  0.64050798 -4.702589   -0.92715994
  1.32749177  1.78043733 -1.47137729  0.37790303 -0.3634757  -1.85541091
  0.36706195  1.56890648  0.14361494  1.50213405  1.39361415  1.18428642
 -4.19396776  1.09759431 -0.3697221  -0.77382432  0.84324942 -1.27661283
  0.21857205  1.03655823 -3.76949046  3.90480859  0.49769571]
```

Variation of RMS (CF - Letor)Value with K (Number of Clusters)

K	Lambda	Training Error	Validation Error
10	0.1	1.24	0.789
100	0.1	0.53	1.60
500	0.1	0.38	0.84
1000	0.1	2.34	3.33

SGD Solution of Letor Training Set

```
### Calculating Stochastic Gradient Descent for Letor training set
learning_rate = 0.001
minibatch_size =  len(letor_op_train)
num_epochs = 10000
L2_lambda = 0.1
sgd_weights_letor_t = SGD_sol(learning_rate, minibatch_size, num_epochs,
                              L2_lambda, phi_letor_t,phi_letor_v, letor_op_train, letor_op_valid)

### Fetch predicted value for Letor Trainig Set
sgd_weights_letor_t = np.array(sgd_weights_letor_t).reshape(1,-1)

### Calculate Error
sgd_predicted_t_letor = get_predicted_value(phi_letor_t, sgd_weights_letor_t )
print("SGD Solution of Letor Training Set : \n", sgd_weights_letor_t)
print("SGD RMS Error of LETOR Training SET : ", np.sqrt(mean_squared_error(letor_op_train, sgd_predicted_t_letor)))

### Calculating Stochastic Gradient Descent for Letor Testing set
learning_rate = 0.001
minibatch_size =  len(letor_op_test)
num_epochs = 10000
L2_lambda = 0.1
sgd_weights_letor_te = SGD_sol(learning_rate, minibatch_size, num_epochs,
                               L2_lambda, phi_letor_te,phi_letor_v,  letor_op_test,letor_op_valid)

### Fetch predicted value for Letor Validation Set
sgd_weights_letor_te = np.array(sgd_weights_letor_te).reshape(1,-1)
sgd_predicted_te_letor = get_predicted_value(phi_letor_te, sgd_weights_letor_te )
print("SGD Solution of Letor Testing Set : \n", sgd_weights_letor_te)
print("SGD RMS Error of LETOR TESTING SET : ", np.sqrt(mean_squared_error(letor_op_test, sgd_predicted_te_letor)))
```

```
SGD Solution of Letor Training Set :
 [[-0.00103357  0.01667844  0.00664586  0.00120358 -0.00403304 -0.00043871
   0.00999837 -0.01683566  0.02428636 -0.00170707  0.00629461 -0.0194734
   0.01042991  0.00652239  0.01228022  0.01249077  0.00380685  0.02251948
  -0.01437053 -0.01336579  0.00730754  0.0061054  -0.008329   -0.01331314
   0.00533142  0.00034661 -0.01010128  0.01032937  0.00655535 -0.00316069
   0.02166535 -0.01507923 -0.01604209  0.00628306 -0.01447541  0.01048321
   0.02776697  0.01961281  0.01046179  0.00949686  0.01562264  0.00419061
  -0.0007524  -0.01600015  0.01081547  0.02784177  0.00175949  0.00620531
   0.00551473  0.01003411 -0.00435315  0.005318   -0.01121216  0.02153697
   0.0022966   0.00687282 -0.01451518  0.00791282  0.00861274  0.00698612
  -0.00698678 -0.00414083  0.0167972  -0.00997683 -0.02201014  0.01070117
   0.02722375  0.01061048 -0.00401033  0.00523283 -0.01506991 -0.01668196
   0.00676429 -0.0132883   0.00700033  0.00331011 -0.00044361  0.0159024
  -0.00652105  0.00576248  0.00819743 -0.00911289 -0.01122435 -0.01044179
   0.02057574 -0.00263499 -0.0007931   0.01647172  0.01327381 -0.00230396
   0.00843415  0.00457103  0.00497215  0.01565114  0.01668675  0.00921998
   0.00195389  0.01755624 -0.0013122   0.0031062   0.02488944]]
SGD RMS Error of LETOR Training SET :  0.560786414841
```

```
SGD Solution of Letor Testing Set :
 [[ 0.00134846 -0.01322206 -0.00116385  0.01684652  0.00988675  0.00944966
   0.01082627  0.00458081 -0.00926854  0.00211368 -0.00437169  0.00768672
   0.01383028  0.01317487  0.01281256 -0.00103382 -0.00424262 -0.00815876
  -0.00821111  0.00229753  0.01120159  0.00606538 -0.00729963  0.00533396
  -0.00024004  0.00011764  0.00723807  0.01536366 -0.00733477  0.01307409
   0.02613676  0.01898741 -0.01326677 -0.01204736  0.00573274  0.0082891
   0.00781311  0.01735149  0.00180801  0.01100578  0.01295843 -0.02449645
   0.01446676  0.00573353  0.01209703  0.01507032 -0.00012658  0.00997147
   0.00970178  0.00911023  0.00661205  0.00487153 -0.02123728  0.02011857
  -0.01012853 -0.0066352   0.00196193  0.01012018 -0.00484767  0.00658391
   0.00506468  0.00842562  0.01200315  0.0133034   0.01136146  0.00880489
   0.01016683 -0.00279475 -0.01490319  0.00644485  0.01317858  0.00947903
   0.01353108  0.00451174  0.01644102 -0.01853381  0.00882442 -0.01866275
  -0.02199176  0.00182156 -0.01821414 -0.00164033  0.00610455  0.00890445
  -0.01823941  0.00071038 -0.01677897 -0.01651829 -0.02007957  0.01976214
   0.02359996  0.0145524  -0.01097668  0.00668312  0.01975997  0.01026113
   0.00548575  0.01506888  0.00179848  0.00201635  0.00915009]]
SGD RMS Error of LETOR TESTING SET :  0.559681353914
```

Closed Form Solution of Synthetic Training Set

```
### Input - output_data - RX1 2d matrix - Target values from data set given
def closed_form_sol(L2_lambda, design_matrix, output_data):
    return np.linalg.solve( L2_lambda * np.identity(design_matrix.shape[1]) + np.matmul(design_matrix.T, design_matrix), np.r

### Calculating Wml - weight for maximum likelihood from the closed form sol Synthetic
wml_syn = np.array(closed_form_sol(l_forLambda, phi_syn_t, syn_op_train)).reshape(1,-1)

### Function To Return WXPHI - Predicted Values as generated from my leniar regression equation
def get_predicted_value(design_matrix, wml):
    return np.matmul(design_matrix, wml.T)

### Getting predicted Values for Synthetic Training Input
prediction_set_syn = get_predicted_value (phi_syn_t, wml_syn)
### Getting Learning Error for Synthetic Data :
print("RMS Error of Synthetic Training  SET Closed Form :  ", mean_squared_error(syn_op_train, prediction_set_syn))

### Validation Set Operations for Synthetic Data
kmeans_v_syn = computing_KMeans_clusters(k_forK_means, syn_ip_valid)
kmeans_v_cluster_dict_syn = get_cluster_dictionary(syn_ip_valid,kmeans_v_syn)
valid_input_cov_syn = get_spreads(kmeans_v_cluster_dict_syn)
valid_input_cntr_syn = get_data_centers(kmeans_v_syn)
phi_syn_v = compute_design_matrix(syn_ip_valid, valid_input_cntr_syn[:, None] , valid_input_cov_syn)

### Predicted Values for Synthetic Validation Set
prediction_set_v_syn = get_predicted_value (phi_syn_v, wml_syn)
print("RMS Error of Synthetic Validation SET  Closed Form  : ",mean_squared_error(syn_op_valid, prediction_set_v_syn))

### Operations for Synthetic Testing Set Data
kmeans_te_syn = computing_KMeans_clusters(k_forK_means, syn_ip_test)
kmeans_v_cluster_dict_syn_te = get_cluster_dictionary(syn_ip_test, kmeans_te_syn)
input_cov_syn_te = get_spreads(kmeans_v_cluster_dict_syn_te)
input_cntr_syn_te = get_data_centers(kmeans_te_syn)
phi_syn_te = compute_design_matrix(syn_ip_test, input_cntr_syn_te[:, None] , input_cov_syn_te)
```

```
RMS Error of Synthetic Training  SET Closed Form :   0.368433887321
RMS Error of Synthetic Validation SET  Closed Form  :  0.575057683804
RMS Error of Synthetic Testing SET  Closed Form  :  0.794070981068
CLOSED FORM Solution For Synthetic Training Set
 [ -4.31280965e+01   2.55535437e+00   2.40013414e+00   1.14086202e+00
  -5.88064109e+00  -1.29680112e+00   3.13620289e+00   4.87511531e+00
  -3.98243279e+00  -7.35787477e-02   2.24414616e+00   3.75670316e+00
  -2.73906973e+00   2.58184596e+00  -2.22015116e+00   2.28504824e-01
  -2.33326006e+00   2.49718885e+00  -3.84483237e-01   3.03281406e+00
  -3.00895492e+00  -2.79102144e+00   1.28197035e+00   3.81915449e+00
  -6.65917561e-01  -1.54952612e+00   1.09786522e+00   1.35391105e+00
   3.61041470e+00  -2.07230558e-01  -3.11033731e-01   2.31781137e+00
   3.03504934e+00   8.39446007e-01   3.17508692e+00   1.43906230e-01
  -2.12131758e+00   3.88388455e+00  -1.18718929e+00   2.93846994e+00
   1.09944186e+00   1.06473709e+00  -7.76572938e-01   6.52756936e+00
   8.55278391e+00  -8.89818103e-01  -1.20100049e+00  -4.38116992e+00
  -4.49106858e+00   2.57393279e+00  -2.84146432e+00  -3.76780191e+00
  -3.91010051e+00   1.91270386e+00  -7.17957582e-01  -6.52568393e-01
   3.89431549e+00   2.03376803e-01   7.47159876e-01  -2.10590164e+00
  -6.46473108e+00   1.02552527e-01   3.29004641e+00   3.41535717e+00
   2.63361226e+00   2.44825741e+00   2.44647683e+00   1.67639034e+00
  -1.19650348e+00   1.17945105e+00   6.70627175e-01  -2.65764427e+00
   1.30265170e+00  -1.50421706e+00   1.04071354e+00  -1.32037814e+00
   3.23053151e+00  -1.14928558e+00  -3.06030043e+00   3.54778620e+00
   2.92673532e+00   1.63028003e+00   9.25718751e+00   3.92328722e+00
  -4.03981799e-02  -3.02821510e+00  -3.11560322e-01   1.75227877e+00
  -6.96450343e-01  -6.05722326e-01  -1.79029435e+00   1.52919452e+00
  -7.55300061e-01   4.07837337e+00   1.99041196e+00   6.88088372e-01
  -1.66288085e+00  -4.86605767e+00  -3.48953661e+00  -1.33301721e-01
  -9.40497817e-01]
```

Variation of RMS (CF - Synthetic) Value with K (Number of Basis Functions)

K	Lambda	Training Error	Validation Error
10	0.1	0.209	1.5
100	0.1	0.36	0.57
500	0.1	0.78	0.2.004
1000	0.1	0.94	0.302

SGD Solution of Letor Training Set

```
SGD Solution of Synthetic Training Set :
 [[ -5.17209020e-04   6.94439421e-03   1.64141065e-02   1.67527017e-02
   -1.11980431e-02   3.97815862e-03   1.32596937e-02   2.09509879e-02
   -2.66019460e-03   4.90545235e-03   1.20401563e-02   1.51296866e-02
    1.82955082e-06   1.63458046e-02   3.07570398e-03   1.39388225e-02
    4.49008464e-03   1.82684831e-02  -2.25319505e-03   1.61928269e-02
    8.25595920e-05   2.89107010e-03   5.98131562e-03   2.42400771e-02
    1.88784210e-03  -1.20761720e-03   9.46571511e-03   2.06151178e-02
    1.64723379e-02   1.84778879e-02   4.38327683e-03   1.49399583e-02
    1.14418418e-02   2.21999121e-02   2.85869331e-02   8.84368631e-03
   -2.23027508e-03   2.39244348e-02   1.15525268e-02   1.32864924e-02
    8.05577056e-03   9.81004667e-03   7.31550511e-03   2.92173810e-02
    2.56358850e-02   8.65233119e-03  -4.55088731e-05   1.21624984e-03
   -1.63947316e-04   1.69373616e-02   2.50413270e-03  -4.86254728e-03
    6.61911625e-03   7.21334673e-03   5.64574387e-03   1.15591042e-02
    9.38071575e-03   1.95578017e-02   7.65798394e-03   7.09902355e-03
    8.36095302e-04   5.09152749e-03   2.27638223e-02   1.06017516e-02
    1.66198429e-02   1.64917493e-02   1.29648245e-02   8.56281054e-03
    7.21937713e-03   1.89442552e-02   5.45909861e-03   9.22276870e-03
    1.12788620e-02   2.75323654e-03   1.39968410e-02   2.75989758e-04
    9.38992887e-03   1.43063635e-02   3.71395499e-03   1.02016793e-02
    1.82240447e-02   1.26551060e-02   2.87169036e-02   2.16999781e-02
    2.31267434e-02  -7.49748827e-03   6.32818628e-03   1.19887881e-02
    3.64583078e-03   1.16685728e-03  -6.15929380e-03   2.14343594e-02
    1.57868323e-02   1.11112790e-02   9.69710317e-03   6.63963000e-03
    5.92605142e-03  -3.38691576e-03  -8.06985955e-04   1.89563061e-02
    7.58032926e-03]]
SGD RMS Error of Synthetic Training SET :  0.787031088591
```

```
SGD Solution of Synthetic Testing Set :
 [[ 0.00978914  0.00959517  0.00962176  0.00966738  0.00960338  0.0097963
   0.00965393  0.00966128  0.00967196  0.00971025  0.00966205  0.00972248
   0.00968853  0.00963433  0.00965267  0.00960245  0.00961683  0.00964288
   0.00964718  0.00959224  0.00959356  0.00966876  0.00951612  0.00972199
   0.00970872  0.00959826  0.00960788  0.00973531  0.00964721  0.00969997
   0.0096724   0.00969153  0.00967563  0.00978954  0.00971159  0.00964922
   0.0097682   0.00963235  0.00971751  0.00963223  0.00969699  0.00965577
   0.00960798  0.00967008  0.00968188  0.00966687  0.00969859  0.00971611
   0.00972786  0.00970925  0.00971474  0.00960561  0.00964458  0.00960366
   0.00972107  0.0095612   0.00960677  0.00974417  0.00968175  0.00960773
   0.00969792  0.00981232  0.00952717  0.00961955  0.00967642  0.0096718
   0.00959762  0.00965811  0.00969261  0.00952082  0.00972411  0.00957645
   0.00955668  0.00961691  0.00971361  0.00958958  0.00960627  0.00963743
   0.00954652  0.00963002  0.00964826  0.00964659  0.00970105  0.00967028
   0.00964986  0.00962873  0.00965039  0.00961916  0.00976375  0.00973555
   0.00969129  0.00968143  0.00957968  0.00964307  0.00968709  0.00966525
   0.00976777  0.00968037  0.00967486  0.0096234   0.0096952 ]]
SGD RMS Error of Synthetic Testing SET :  0.790453888026
```

```
    predicted_values = get_predicted_value(phi, temp_wts)
    return np.sqrt(mean_squared_error(op, predicted_values))

### Calculating Stochastic Gradient Descent for synthetic training set
learning_rate = 0.01
minibatch_size =  len(syn_op_train)
num_epochs = 1000
L2_lambda = 0.1
sgd_weights_syn_t = SGD_sol(learning_rate, minibatch_size, num_epochs, L2_lambda, phi_syn_t, phi_syn_v, syn_op_train, syn_op_

### Fetch predicted value for Synthetic Training Set
sgd_weights_syn_t = np.array(sgd_weights_syn_t).reshape(1,-1)

### Calculate Error
sgd_predicted_val_syn = get_predicted_value(phi_syn_t, sgd_weights_syn_t )
print("SGD Solution of Synthetic Training Set : \n", sgd_weights_syn_t)
print("SGD RMS Error of Synthetic Training SET : ", np.sqrt(mean_squared_error(syn_op_train, sgd_predicted_val_syn)))

### Calculating Stochastic Gradient Descent for synthetic testing set
learning_rate = 0.001
minibatch_size =  len(syn_op_test)
num_epochs = 10000
L2_lambda = 0.1
sgd_weights_syn_te = SGD_sol(learning_rate, minibatch_size, num_epochs, L2_lambda, phi_syn_te, phi_syn_v, syn_op_test, syn_op

### Fetch predicted value for Synthetic TEsting Set
sgd_weights_syn_te = np.array(sgd_weights_syn_te).reshape(1,-1)

### Calculate Error
sgd_predicted_val_syn = get_predicted_value(phi_syn_te, sgd_weights_syn_te )
print("SGD Solution of Synthetic Testing Set : \n", sgd_weights_syn_te)
print("SGD RMS Error of Synthetic Testing SET : ", np.sqrt(mean_squared_error(syn_op_test, sgd_predicted_val_syn)))
```