assignment09

May 29, 2019

0.0.1 Binary classifier using linear regression

0.0.2 Name : Park Jinhyuk

0.0.3 Student-ID : 20141775

In [ ]:
```
import matplotlib.pyplot as plt
import numpy as np
import random

file_data                   = "mnist_train.csv"
file_test_data          = "mnist_test.csv"
handle_file         = open(file_data, "r")
handle_test_file         = open(file_test_data, "r")
data                                = handle_file.readlines()
test_data                               = handle_test_file.readlines()
handle_file.close()
handle_test_file.close()

size_row        = 28    # height of the image
size_col          = 28    # width of the image
num_image         = len(data)
num_test_image= len(test_data)
count               = 0     # count for the number of images
numOfZero = 0
numOfNonZero = 0
test_numOfZero = 0
test_numOfNonZero = 0


#
# normalize the values of the input data to be [0, 1]
#
def normalize(data):

    data_normalized = (data - min(data)) / (max(data) - min(data))

    return(data_normalized)
```

```
list_image  = np.empty((num_image, size_row * size_col + 1), dtype=float)
list_test_image  = np.empty((num_test_image, size_row * size_col + 1), dtyp
list_label  = np.empty(num_image, dtype=int)
list_test_label  = np.empty(num_test_image, dtype=int)
classification = np.empty(num_image, dtype=float)
test_classification = np.empty(num_test_image, dtype=float)

for line in data:
    line_data   = line.split(',')
    if line_data[0] == '0':
        label = 1
        numOfZero = numOfZero + 1
    else:
        label = -1
        numOfNonZero = numOfNonZero + 1

    im_vector   = np.asfarray(line_data[1:])
    im_vector   = normalize(im_vector)
    list_label[count]       = label
    list_image[count, 0] = 1
    list_image[count, 1:]    = im_vector
    count = count + 1

count = 0
for line in test_data:
    line_test_data   = line.split(',')
    if line_test_data[0] == '0':
        label = 1
        test_numOfZero = test_numOfZero + 1
    else:
        label = -1
        test_numOfNonZero = test_numOfNonZero + 1

    im_vector   = np.asfarray(line_test_data[1:])
    im_vector   = normalize(im_vector)
    list_test_label[count]   = label
    list_test_image[count, 0] = 1
    list_test_image[count, 1:]= im_vector
    count = count + 1
```

0.0.4 Compute optimal model parameter

```
In [21]: d = 28 * 28
         X = np.zeros(d+1)
         X = np.dot(np.linalg.pinv(list_image),list_label)
         print(X)
```

```
[-6.84406870e-01  2.37849111e-12  9.77649619e-13 -7.55606562e-13
  4.65302804e-13 -2.72493104e-12 -3.42955075e-12  2.44608609e-12
  1.46186912e-12  1.54540945e-12  2.34367809e-12 -3.67131030e-13
  1.68286852e-12  1.53458664e-01  1.98220986e-01 -1.53345669e-01
 -6.38940289e-03  4.36666199e-13  2.27434452e-13  3.19672081e-13
 -2.79285881e-13 -1.86745735e-12  2.64320482e-12 -9.20819970e-14
  8.40131115e-13 -4.18773815e-13  3.10040223e-12 -1.20469402e-12
  2.18646697e-13  1.46863171e-13  1.98674311e-12  8.17567744e-13
 -1.55858501e-13 -4.06289060e+00  1.05099744e+00  1.23101922e-01
  2.11306669e-01  6.51508445e-02  2.55392530e-03  5.52498152e-02
  1.15990558e-01 -7.23390275e-02  2.72683304e-01 -1.63118357e-01
  6.19145739e-02  3.85382309e-05  7.70688345e-02 -6.15820985e-02
  1.52778049e-02  8.53863822e-02 -1.41982541e-01  2.38270481e-01
  3.58333722e-02  3.33915605e-13  2.40022670e-12  2.57644850e-14
  9.76465682e-13 -1.72230906e-12 -3.78761513e-13 -8.36059853e+00
  9.77593181e+00  5.34144080e-01 -7.99936568e-01 -1.89682766e-01
  1.01087544e-01 -5.27782579e-02 -8.17534204e-02 -7.48630136e-02
  5.01256328e-02 -4.91614464e-02 -1.42921940e-02 -3.38311142e-02
  3.52755065e-03 -1.10751292e-01 -2.59229330e-02 -4.38081127e-02
 -4.70799587e-02 -9.88395498e-02  1.01588630e-01 -1.69319576e-01
  1.03897289e-01 -2.46427687e-01  4.10216976e-01 -2.13067125e-12
 -1.76047896e-12 -6.34506779e-13 -1.02063998e-12  3.95434813e+00
 -1.79359733e+00 -8.68499683e-02 -8.20073230e-02 -6.32433515e-02
 -1.82101353e-03  2.33142466e-02 -6.21453144e-02 -3.36839292e-02
  5.46815877e-02 -5.94164193e-02  4.61485594e-02 -4.58733621e-02
 -3.06237392e-02 -1.76278944e-02 -4.09481750e-02 -1.03436869e-01
 -4.58517277e-02 -1.08105115e-01 -1.05007043e-01 -6.90997253e-02
 -6.71496011e-02  1.07949793e-01 -1.11765369e-01  2.28174578e-01
  4.03741981e-14  1.20238975e-14  6.18091090e-01 -2.88022934e-01
 -5.60173775e-02  7.27680834e-02 -1.15032164e-02 -2.07682000e-02
 -4.97472829e-02  3.72950169e-02 -4.34850573e-02 -2.24350027e-02
 -2.82845323e-02  1.66644578e-03  3.06577987e-02 -3.48394437e-02
  3.91208914e-02 -1.66416499e-02  3.82942676e-03 -2.43450472e-03
  2.57119769e-02 -6.91423765e-03  2.78355493e-02 -2.46222020e-02
 -6.62873397e-02 -6.42904228e-02 -3.59056314e-02 -2.42324348e-01
  1.32591422e+00 -1.72974545e-12  6.66162405e-13 -9.18140767e-01
 -7.59158102e-02 -1.37934055e-02 -4.03775682e-02  3.84716436e-02
 -1.51256434e-02 -2.94318523e-02 -2.44712072e-02 -1.96691887e-02
  9.80477629e-04 -2.24541051e-02  3.57542879e-02  1.15025008e-04
  1.94593316e-02  9.74017026e-03  1.41481511e-02 -1.47587351e-02
  1.89919848e-02 -2.54640490e-02 -1.82647686e-02 -5.10601180e-02
 -6.28953006e-02 -2.35283865e-02 -2.87450168e-02 -1.11628448e-01
 -8.97641357e-01  2.30234131e-12 -2.54615777e+00  1.77054232e-01
  6.51639999e-03  1.89324682e-02 -2.88200629e-02  1.47532681e-02
 -1.67811983e-02  1.85848674e-02 -2.69291781e-02 -2.21421420e-02
 -1.06494634e-02 -1.71309113e-03  2.28669380e-02  2.94280678e-03
 -5.01942060e-03  1.20090491e-02 -1.13210503e-03  1.50315298e-02
 -1.38217775e-02  2.76423728e-03  1.92529974e-02  5.79218032e-03
```

-3.81634649e-02	-1.03944358e-01	-3.14676432e-02	-7.65759887e-02
-8.50549840e-02	-3.03249788e+01	-5.54541295e-02	-3.13129791e-01
4.15213998e-02	-1.29074041e-02	1.49939310e-02	-7.85612034e-03
4.49740552e-03	-2.59042120e-02	3.58688890e-02	-1.58589842e-02
1.42294856e-02	1.44020017e-02	1.37852173e-02	3.68645005e-02
3.21009183e-02	1.84545540e-03	3.82445587e-02	2.14842305e-02
1.88490310e-02	-2.09385855e-02	-7.04304552e-03	3.74623018e-03
-1.69926825e-02	-1.09216750e-01	-4.70023874e-02	-3.34857107e-02
1.18512098e-01	7.73735685e+00	7.99143648e-02	3.22226999e-02
-3.82070853e-02	-4.49128403e-03	-1.08295897e-02	3.77210140e-02
-1.01925718e-02	4.94795889e-03	-4.63040005e-03	1.23028027e-03
-1.29121106e-02	1.50728522e-02	4.12847137e-03	2.00257205e-02
1.81355516e-02	5.21674636e-02	1.40269931e-02	1.59614918e-02
9.93535240e-04	3.12223004e-02	-2.69868252e-02	1.21752778e-02
-7.24205297e-04	-1.22883530e-01	-1.26552514e-01	-2.67225984e-02
-7.38626850e-02	-6.27322344e-01	1.27248200e-01	-5.28187663e-02
4.45343641e-03	-1.71999311e-02	-1.74501373e-02	-2.30254058e-03
2.25724809e-02	1.17416898e-02	5.02298624e-03	1.09617011e-02
-1.80156042e-02	-6.30583859e-03	2.91698268e-02	2.58267363e-02
3.28687784e-02	4.22587482e-03	1.17818029e-02	8.77213799e-03
4.14944303e-02	1.05755469e-02	2.09568952e-02	-3.49942692e-02
2.64943604e-02	-3.32182580e-02	-2.19752946e-01	3.58305269e-03
-7.48816740e-02	4.18957867e-01	-2.18045061e-01	-6.24291898e-03
-2.11050407e-02	4.25264894e-03	2.24552106e-02	-1.20506272e-03
-1.36266811e-02	-1.63404825e-02	2.85703697e-02	-3.93904658e-03
1.09725370e-02	1.73111813e-02	9.11564016e-03	8.90299627e-03
3.04191955e-02	1.91257431e-02	3.60257052e-02	3.42702070e-02
7.87979252e-03	6.67227735e-02	3.30600147e-02	4.33365216e-02
7.53291073e-02	-7.62440689e-02	-2.23405116e-01	-1.68173775e-01
3.89237123e-01	-3.80811748e-01	2.97387609e-01	-5.61442283e-02
-3.43231122e-02	2.94597817e-03	-6.96512694e-02	8.98677407e-04
1.92926765e-02	1.03144421e-02	-1.29220519e-02	2.35269623e-02
1.45621788e-02	-2.05347095e-02	-1.51127791e-03	-4.22912417e-02
-2.55523189e-02	-2.08643574e-02	7.58496119e-04	9.88128237e-03
4.17800111e-02	2.00277180e-02	6.42581226e-02	1.47980509e-02
7.06556316e-02	1.07036530e-01	-3.40858900e-01	1.45597629e-02
-6.71918956e-01	1.33406797e-01	-2.36439043e-01	-7.46582387e-02
2.28924283e-02	-7.51003437e-02	-9.99749446e-03	1.87249191e-02
-1.54045077e-02	1.97633852e-02	9.56839230e-03	1.75595175e-02
-2.31656086e-02	1.47696624e-02	-1.09907080e-02	-7.64654750e-02
-2.89851250e-02	-4.27671985e-02	-3.35512445e-02	-3.72411965e-02
-2.87444652e-02	2.67571439e-02	1.62979898e-02	2.82561441e-02
1.28250228e-01	8.37250697e-02	-3.81218645e-02	-3.28052397e-01
7.05674148e-01	3.63110341e-01	7.10313597e-01	2.41499138e-02
-2.05539209e-03	-1.29324611e-02	-1.07866613e-02	-2.09651390e-02
1.95231675e-02	-2.23026408e-02	1.53915500e-02	1.00517438e-02
7.87008735e-03	-2.81797455e-02	-4.15333942e-02	-5.19998437e-02
-2.62643585e-02	-2.30192098e-02	-4.85780741e-02	-4.99575948e-02

-2.31449969e-02	-4.54315280e-02	1.83590882e-02	8.82262779e-02
7.19746907e-02	-9.63454256e-02	4.64711972e-04	1.18632499e-01
-4.11982927e-01	-3.35332957e-01	-4.34764244e-01	-6.17469782e-02
-6.03415505e-02	-1.38495558e-01	2.08704780e-02	3.93140973e-02
2.41313134e-02	1.45085147e-02	2.05524243e-02	5.01242979e-02
1.01364241e-04	-4.30134977e-02	5.17781332e-03	-6.97470077e-02
5.63711764e-03	-4.00069758e-02	-4.28528257e-02	-2.95278733e-02
-3.82561159e-02	1.85397753e-02	8.03620483e-02	3.46008304e-04
2.37996144e-02	2.94729437e-02	-5.23216875e-02	-5.90680735e-02
6.62195895e-01	-4.29210243e-01	2.52607443e-01	1.73402328e-02
-3.14452026e-02	4.84402580e-02	5.70984466e-02	-3.66184434e-02
3.51417996e-02	3.31553407e-02	3.73607626e-02	3.17000887e-02
-1.43734465e-02	-3.23869638e-02	-2.20822703e-02	-5.77931322e-02
-8.11349417e-03	-4.95793602e-02	-4.01245350e-02	-2.96377915e-02
-1.42454933e-02	3.16969883e-02	-5.00143928e-03	-1.38310361e-03
2.56643222e-02	-2.54677540e-02	-5.12622825e-02	-3.16903947e-02
1.48476771e-01	-1.82520546e+00	5.17803502e-02	3.00535671e-01
-2.36293327e-01	7.56630711e-03	1.00367779e-02	2.92923949e-02
6.97245018e-03	2.72067325e-02	9.55875478e-03	2.10924880e-02
-1.01636105e-02	-5.61730362e-02	-3.69367142e-02	-3.53812007e-02
-2.99457580e-02	-4.09076093e-02	-4.48277347e-02	-5.58197081e-03
1.54182004e-02	1.39168119e-02	1.06910131e-02	-5.47042901e-02
2.46556418e-02	2.35494096e-02	-1.27004520e-02	-6.38621386e-02
-4.25018506e-01	-2.87320505e-13	-1.29223160e-01	-1.87415458e-01
-2.74218442e-02	1.67616152e-02	2.42206933e-02	2.83859045e-02
2.48848750e-02	-4.18491369e-03	1.52699735e-02	1.68604462e-02
-1.50302662e-02	-1.44521357e-02	-4.43427204e-02	-5.59167160e-02
-4.79870056e-02	-1.81934611e-02	1.09120628e-02	9.09260197e-04
2.17567215e-02	-1.44962744e-02	-1.60245832e-02	5.41224889e-02
-2.42767738e-02	5.00929933e-03	-1.80489732e-02	1.65186943e-01
1.64785941e+00	-2.19509739e-01	-1.31617157e-01	2.54456139e-01
-2.36282055e-01	-4.86076891e-03	3.42595156e-02	-8.52251158e-03
4.06953455e-02	3.33248092e-02	3.99209311e-03	-7.80583438e-03
8.52431406e-03	-2.65320802e-02	-1.83140904e-02	-3.06411374e-02
2.91128260e-03	-1.30926158e-02	-1.13170294e-02	-1.43181116e-02
-1.24840887e-03	-2.74338544e-02	-5.38618589e-03	-6.68669799e-03
-7.08162299e-02	6.47336340e-02	-3.47759836e-02	-1.91659491e-01
-1.23578611e+00	-1.12312577e+00	2.42183248e-01	-9.17405306e-02
1.25022287e-02	-6.37620262e-03	-2.72170606e-02	1.56103307e-02
2.49191492e-02	1.22834419e-02	3.45549891e-02	-5.27491250e-03
1.81049784e-02	1.31972083e-02	1.15739178e-02	2.38121837e-03
-1.19633821e-02	-3.66833657e-03	2.28370791e-03	1.65742660e-05
-1.33416002e-02	7.12521178e-03	-8.66952595e-03	8.03540302e-04
1.28174931e-02	-1.80418601e-02	-5.29804019e-02	4.08863923e-02
8.58968333e-01	-2.33329246e-13	9.90390513e-01	1.01516547e-01
-1.02853460e-01	-2.74799620e-02	1.71246672e-02	-1.59466059e-02
-4.41256562e-04	1.11776718e-02	1.64047290e-03	2.09760583e-02
5.39196690e-04	2.18820131e-02	1.26726631e-02	9.80587122e-05

6.68817542e-03	-2.57719503e-02	8.27967543e-03	-1.95479269e-02
1.78533707e-03	-1.86333285e-02	3.21933245e-02	-1.98862439e-02
-5.78130510e-02	8.28494936e-03	9.78974481e-02	-1.08752359e-01
-3.06954502e-01	3.58949236e-01	-1.03849184e+00	-3.07864377e-02
-6.31754505e-03	5.62501274e-02	-2.56597721e-02	-7.91721714e-03
-9.75980198e-03	-9.24629337e-03	2.74498536e-02	1.69854320e-02
2.12478202e-02	-7.89461426e-04	-1.25203659e-02	3.22711764e-03
2.68066041e-02	5.74733337e-03	-2.35661763e-02	6.23550517e-03
-8.57166899e-03	-3.89784316e-03	-1.61280583e-02	1.18028401e-02
-4.36724031e-02	2.19057467e-02	-1.08059735e-02	2.61187616e-01
5.74405642e-01	3.47732072e-01	-1.02562345e+00	-1.76792965e-02
9.90417699e-03	-1.16172693e-01	2.55762439e-02	3.26737750e-02
1.00548008e-02	5.54314170e-03	1.70297693e-02	2.79392902e-03
1.93708086e-02	1.47361004e-02	4.21844802e-02	1.08704349e-02
-1.52939597e-02	-6.02194830e-03	-5.77472198e-03	-3.49974095e-02
1.63384816e-02	-4.85612262e-02	2.03114851e-02	1.32151781e-02
-1.49572657e-02	1.82686709e-02	1.04598116e-02	3.54365368e-01
1.06044118e+00	-6.13234417e-14	3.40955276e-14	1.25388339e-01
-2.30958457e-02	-5.88113771e-02	-1.84364883e-03	-6.04406353e-02
2.54449826e-02	-1.90813667e-02	1.31559119e-02	3.17327666e-03
2.14089830e-02	2.56670589e-02	3.40819846e-03	3.02783250e-02
3.78463391e-02	2.17140210e-02	2.12289652e-02	-1.79607760e-02
-2.20404529e-02	-1.11672821e-02	-5.30621084e-02	9.34300821e-03
1.83625686e-03	1.66758756e-02	-1.47400731e-01	-6.10764180e-01
-7.10528334e-14	8.96684410e-15	-5.54429911e-14	-6.74037263e-01
1.67844952e-01	2.25941586e-02	-8.85053906e-02	7.25145926e-02
-3.58382594e-02	5.00568473e-03	-2.72535815e-02	-4.91354993e-03
-4.56896716e-02	-1.39108544e-02	-1.80468775e-02	-3.17183546e-02
-1.37951641e-02	-4.01226467e-03	4.11089472e-03	3.87584387e-02
-8.91965524e-03	4.45736982e-02	3.25695059e-03	-1.01384643e-01
2.72593517e-02	-1.41423179e-01	2.49092543e-01	4.43747443e+00
1.23107803e-14	1.11255734e-14	3.91521696e-14	5.74024081e-01
7.10073769e-02	-1.14904287e-03	-1.36248522e-01	-5.52168592e-02
-1.05269463e-01	-5.52386399e-02	-4.80398149e-02	-6.66121557e-02
-2.69727298e-02	-6.47967538e-02	-3.07931417e-02	-4.40256645e-02
-1.10227227e-02	-3.02929742e-02	-3.42579844e-02	-2.18200615e-02
1.09956572e-02	-7.18229144e-02	8.45752821e-03	1.42200871e-01
-1.70038171e-01	-1.81931548e-01	-4.97548713e-01	-4.23939234e+00
0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
-9.25135496e-01	2.61352157e-01	-2.06542730e-01	7.77944182e-03
-9.17599613e-02	-4.73427311e-02	-1.16567776e-01	-4.30764643e-02
-1.01269701e-01	-3.37441281e-02	-9.01871025e-02	-4.68745660e-02
-9.76980636e-02	-5.40824984e-02	-3.27357791e-02	-7.45309295e-02
-7.49540276e-03	8.34344478e-03	-1.55967392e-01	-6.34283335e-02
-2.63607005e-01	2.06109092e+00	-2.73970042e-01	0.00000000e+00
0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
0.00000000e+00	8.04406795e-01	2.37610340e-02	-3.77192111e-01
6.07099203e-02	-1.29168168e-01	-1.13140632e-01	-3.49107977e-02

```
 -1.77844159e-02 -1.36109932e-01 -7.60964262e-03 -1.35766440e-01
 -2.44375949e-02 -4.33457087e-02 -1.14834013e-01 -4.48240493e-02
  1.13894563e-02 -1.02555619e-01  1.83726663e-01 -3.49763778e-01
 -2.08560473e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00
  0.00000000e+00]
```

0.0.5 Compute tp,fp,tn,fn on training set

```
In [23]: for i in range(num_image):
             if np.dot(list_image[i], X) > 0:
                 classification[i] = 1
             else:
                 classification[i] = -1

         tp = 0
         fp = 0
         fn = 0
         tn = 0

         for i in range(num_image):
             if list_label[i] == 1:
                 if classification[i] == 1:
                     tp = tp + 1
                 else:
                     fn = fn + 1

             if list_label[i] != 1:
                 if classification[i] == 1:
                     fp = fp + 1
                 else:
                     tn = tn + 1

         print('tp : ', tp / numOfZero)
         print('fp : ', fp / numOfNonZero)
         print('fn : ', fn / numOfZero)
         print('tn : ', tn / numOfNonZero)
```

```
tp :  0.8723619787269965
fp :  0.003310094864729922
fn :  0.12763802127300355
tn :  0.9966899051352701
```

0.0.6 Compute tp,fp,tn,fn on testing set

```
In [24]: for i in range(num_test_image):
             if np.dot(list_test_image[i], X) > 0:
```

```
                test_classification[i] = 1
            else:
                test_classification[i] = -1

        tp = 0
        fp = 0
        fn = 0
        tn = 0

        for i in range(num_test_image):
            if list_test_label[i] == 1:
                if test_classification[i] == 1:
                    tp = tp + 1
                else:
                    fn = fn + 1

            if list_test_label[i] != 1:
                if test_classification[i] == 1:
                    fp = fp + 1
                else:
                    tn = tn + 1

        print('tp : ', tp / test_numOfZero)
        print('fp : ', fp / test_numOfNonZero)
        print('fn : ', fn / test_numOfZero)
        print('tn : ', tn / test_numOfNonZero)
```

```
tp :  0.8836734693877552
fp :  0.004767184035476719
fn :  0.11632653061224489
tn :  0.9952328159645233
```

In [ ]: