

```

1  import struct
2  import numpy.random as random
3  import numpy as np
4  import matplotlib.pyplot as plt
5
6  # weights generated randomly which are used for step (f) and (g)
7  W = np.zeros((10,784))
8  for i in range(10):
9      temp = (1-(-1))*random.sample(784) - 1
10     W[i] = temp
11  mat_Wg=np.matrix(np.array(W))
12
13  # weights generated randomly which are used for step (i) first run
14  W = np.zeros((10,784))
15  for i in range(10):
16      temp = (1-(-1))*random.sample(784) - 1
17      W[i] = temp
18  mat_Wi=np.matrix(np.array(W))
19
20  # weights generated randomly which are used for step (i) second run
21  W = np.zeros((10,784))
22  for i in range(10):
23      temp = (1-(-1))*random.sample(784) - 1
24      W[i] = temp
25  mat_Wii=np.matrix(np.array(W))
26
27  # weights generated randomly which are used for step (i) third run
28  W = np.zeros((10,784))
29  for i in range(10):
30      temp = (1-(-1))*random.sample(784) - 1
31      W[i] = temp
32  mat_Wiii=np.matrix(np.array(W))
33
34  # function for multi-category Perceptron Training Algorithm for training images
35  def MCPTA_training(n,eta,epsilon,mat_W):
36  # importing training labels file
37      train_lbl = open('train-labels.idx1-ubyte', 'rb')
38      for i in range(2):
39          train_lbl.read(4)
40      des=[]
41      for i in range(n):
42          des.append(struct.unpack('>B', train_lbl.read(1))[0])
43
44      flag=1
45      epoch=0
46      epoch_list=[]
47      error_list=[]
48
49  # do epoch iterations until misclassifications converges below set threshold (i.e.
50  epsilon)
51      while(flag==1):
52          error_epoch=0
53  # importing training images file
54          train_img = open('train-images.idx3-ubyte', 'rb')
55          for i in range(4):
56              train_img.read(4)
57  # 1 grey scale image = 28X28 pixels
58          for x in range(n):
59              xi=[]
60              for p in range(784):
61                  xi.append(struct.unpack('>B', train_img.read(1))[0])
62              mat_xi=np.matrix(np.array(xi))
63
64              mat_xi_T=np.transpose(mat_xi)
65              mul = np.dot(mat_W,mat_xi_T)
66              act = np.argmax(mul)
67  # updating weights in case of misclassification
68              diff_mat = np.zeros((10,1))
69              if des[x]!=act :
70                  diff_mat[act]==-1
71                  diff_mat[des[x]]=1
72                  error_epoch = error_epoch+1

```

```

72         mat_W = mat_W + np.dot((eta*diff_mat),mat_xi)
73
74     epoch = epoch +1
75     if(epoch==100): break
76     epoch_list.append(epoch)
77     error_list.append(error_epoch)
78     print("Epoch:",epoch,"Error: ",error_epoch)
79     if (error_epoch/n)>epsilon:
80         flag=1
81     else:
82         flag=0
83     print("For ",n," training images, epoch list: ",epoch_list)
84     print("For ",n," training images, error list: ",error_list)
85     # plotting graph for number of epochs v/s number of misclassifications
86     plt.title("Epoch V/S Miss")
87     plt.xlim(0,epoch+1)
88     plt.plot(epoch_list,error_list,'o-')
89     plt.xlabel('Epoch')
90     plt.ylabel('Misclassification')
91     plt.show()
92     return mat_W
93
94 # function for multi-category Perceptron Training Algorithm for test images
95
96 def MCPTA_test(n,mat_W1):
97     # importing test labels file
98     test_lbl = open('t10k-labels.idx1-ubyte', 'rb')
99     for i in range(2):
100         test_lbl.read(4)
101     des=[]
102     for i in range(n):    ###60000
103         des.append(struct.unpack('>B', test_lbl.read(1))[0])
104
105     test_error=0
106     # importing test images file
107     test_img = open('t10k-images.idx3-ubyte', 'rb')
108     for i in range(4):
109         test_img.read(4)
110     for x in range(n):
111         xi=[]
112         for p in range(784):
113             xi.append(struct.unpack('>B', test_img.read(1))[0])
114         mat_xi=np.matrix(np.array(xi))
115
116         mat_xi_T=np.transpose(mat_xi)
117         mul = np.dot(mat_W1,mat_xi_T)
118         act = np.argmax(mul)
119     # Calculating total misclassifications for test image
120     diff_mat = np.zeros((10,1))
121     if des[x]!=act :
122         diff_mat[act]= -1
123         diff_mat[des[x]]=1
124         test_error = test_error + 1
125
126     print("Total errors for ",n," test images: ",test_error)
127     print("Percentage misclassification for ",n," test images: ",(test_error/n)*100)
128     return
129
130 print("\r\n-----Multi-category PTA for 50 training images
-----\r\n")
131 mat_W1= MCPTA_training(50,1,0,mat_Wg)
132 print("\r\n-----Multi-category PTA for 50 test images
-----\r\n")
133 MCPTA_test(50,mat_W1)
134 print("\r\n-----Error Comparison of whole Test with 50 test images
-----\r\n")
135 MCPTA_test(10000,mat_W1)
136
137 print("\r\n-----Multi-category PTA for 1000 training images
-----\r\n")
138 mat_W1= MCPTA_training(1000,1,0,mat_Wg)
139 print("\r\n-----Multi-category PTA for 1000 test images

```

```

140 MCPTA_test(1000,mat_W1)
141 print("\r\n-----Error Comparison of whole Test with 1000 test images
-----\r\n")
142 MCPTA_test(10000,mat_W1)
143
144 print("\r\n<<<<<<<<<<<<<<<< step (h)- Terminating after 100 epochs as graphs
don't converge >>>>>>>>>>>>>>>>>>>\r\n")
145 MCPTA_training(60000,1,0,mat_Wg)
146
147 print("\r\n-----Multi-category PTA for 60000 training images with
different weights (say w1) ----- \r\n")
148 mat_Wl= MCPTA_training(60000,1,0.107,mat_Wi)
149 print("\r\n-----Multi-category PTA for 10000 test images with different
weights (say w1) ----- \r\n")
150 MCPTA_test(10000,mat_W1)
151
152 print("\r\n-----Multi-category PTA for 60000 training images with
different weights (say w2) ----- \r\n")
153 mat_Wl= MCPTA_training(60000,1,0.107,mat_Wii)
154 print("\r\n-----Multi-category PTA for 10000 test images with different
weights (say w2) ----- \r\n")
155 MCPTA_test(10000,mat_W1)
156
157 print("\r\n-----Multi-category PTA for 60000 training images with
different weights (say w3) ----- \r\n")
158 mat_Wl= MCPTA_training(60000,1,0.107,mat_Wiii)
159 print("\r\n-----Multi-category PTA for 10000 test images with different
weights (say w3) ----- \r\n")
160 MCPTA_test(10000,mat_W1)

```