Ex1 – Advanced Topics in Machine Learning

Name: Oren Cohen
ID: 305164295

I've run the 3 multi-class "classifiers" (OVA – one VS all, AP – all pairs, RAND – random) and tested the prediction accuracy on test file (the test file accidentally sent by you) **with two methods:**

1. hamming distance
2. loss based decision

I will write 3 specific epochs (start =1, mid= 25, end=50)

Hyper parameters (for all trains):
$\lambda = 0.1$
$\eta = 0.1$
Epochs = 50

1. OVA:

    ############ epoch: 1 ############
    hamming accuracy is: 84.60 %
    loss accuracy is: 92.30 %
    ############ epoch: 25 ############
    hamming accuracy is: 82.00 %
    loss accuracy is: 93.00 %
    ############ epoch: 50 ############
    hamming accuracy is: 91.10 %
    loss accuracy is: 93.70 %

2. AP:

    ############ epoch: 1 ############
    hamming accuracy is: 88.40 %
    loss accuracy is: 91.40 %
    ############ epoch: 25 ############
    hamming accuracy is: 93.80 %
    loss accuracy is: 95.60 %
    ############ epoch: 50 ############
    hamming accuracy is: 96.20 %
    loss accuracy is: 96.10 %

3. RAND:

    ############ epoch: 1 ############
    hamming accuracy is: 85.80 %
    loss accuracy is: 88.20 %
    ############ epoch: 25 ############
    hamming accuracy is: 91.20 %
    loss accuracy is: 93.70 %
    ############ epoch: 50 ############
    hamming accuracy is: 93.50 %
    loss accuracy is: 93.40 %

Ex1 – Advanced Topics in Machine Learning
Name: Oren Cohen
ID: 305164295

It is obvious to see, that in almost every epoch and every classifier, the **loss accuracy** was **much better then the hamming distance accuracy**.

And from all "classifiers" above, the most accurate was the **AP classifier** with ~96%!
In the second place was OVA, and in the last place RAND.

Its important to say, that the RAND classifier contains 16 classifiers, (f0, f1, … ,f15) and uniformly selected (generate matrix uniformly between 0 to 1 ($x \rightarrow [0,1]$), multiply by 2($x \rightarrow [0,2]$), sub -1 ($x \rightarrow [-1,1]$), and then round to nearest integer ($x \rightarrow \{-1, 0, 1\}$)) .
Therefor, it was very unstable (in some runs, the accuracy was around 50% and sometimes around 70%).

The training section:
I've created 3 matrix, one for each classifier (W1 – OVA, W2-AP, W3- RAND),
with 784 rows and #of classifiers columns.
W1 was (784x4), W3 was (784,16) ,W3 was (784x16).
for every input I've trained all classifiers in W1, W2, W3 (based on the true label)

The decoding section:
there was the "ideally classifier" matrix with 4 rows (row for each class) and #of classifiers columns (column for each classifier).
To get the prediction (classifier #) , I checked the "accepted row" from all f_i in the classifier (which was the prediction of the classifier).
for example:
in W1, there was 4 classifiers : f0, f1, f2, f3 , so we get f(x) = [f0(x), f1(x), f2(x), f3(x)]
then, for the class prediction, I used 2 methods (as mentioned).

1. hamming distance:
the goal was to choose the closest row/class in the "ideally classifier" to f(x) (the minimum hamming distance)

2. loss based decision:
the goal was to choose the row/class such that, the loss of f(x) and row #s will be the minimum (the minimum loss)

It seems to be that the more classifiers we put W (OVA – 4, AP – 6) the more accuracy we get.