**MLBP 2019 - Quiz 5**

**06/11/2020 - 18/11/2020 at 23:59**

---

(1) (1.0 pt.) Assume 15 base learners with independent true risk 0.3 ($\epsilon = 0.3$), are grouped to form a strong learner via majority voting. What is the probability of having an incorrect aggregated prediction (using the formula given in page 3, lecture 8):
1- 0.01
2- 0.05
3- 0.09

(2) (1.0 pt.) Which option is NOT correct in AdaBoost
1- a base learner with less weighted training error has higher weight to form the final decision
2- the algorithm (page 20, lecture 8) can continue if a base learner weighted training error is higher than 0.5
3- it is prone to over-fitting in the presence of noise

(3) (1.0 pt.) Comparing loss functions in AdaBoost and LogitBoost, which option is NOT correct:
1- LogitBoost loss function is the logarithm of Adaboost loss function
2- AdaBoost exponentially penalizes the misclassified examples
3- in both AdaBoost and LogitBoost the examples with negative margin are penalized

(4) (2.0 pt.) Compute accuracy of AdaBoost algorithm given in slide 20 of lecture 8, using the given perceptron implementation as a weak learner. Yo need to choose the interval that contains the AdaBoost accuracy on the given data, if the result does not fall into any of the intervals then the closest one needs to be selected. For your AdaBoost implementation use the following information:
Number of base classifiers: 5
Note: consider learning the bias (augment your data with a column of ones).
Note: use all the given data as training set and also the asked accuracy should be computed on the training set.

Data: x1 and x2 are features, labels is the output class
x1 = [.1,.2,.4,.8, .8, .05,.08,.12,.33,.55,.66,.77,.22,.2,.3,.6,.5,.6,.25,.3,.5,.7,.6]
x2 = [.2,.65,.7,.6, .3,.1,.4,.66,.22,.65,.68,.55,.44,.1,.3,.4,.3,.15,.15,.5,.55,.2,.4]
labels = [1,1,1,1,1,1,1,1,1,1,1,1,1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1]

---

```
def perceptron (X,labels,sample_weight):
#Inputs:
#X: a 2d array, each row represents an example of the training set
#labels: vector of the examples labels
#sample_weight:vector of examples weights given by Adaboost
#Output:
#pred_labels: the label predicted for each example
d = np.shape(X)[1]
w = np.zeros(d)
i = 1
while(any([element<=0 for element in [labels[ind]*np.dot(w,x) for ind,x in enumerate(X)] ])):
   #misclassified examples
```

```
    mistakes = np.where([element<=0 for element in [labels[ind]*np.dot(w,x) for ind,x in
        enumerate(X)] ])[0]
    pairs = zip(mistakes, sample_weight[mistakes])
    sorted_pairs = sorted(pairs, key=lambda t: t[1], reverse = True)
    #use the misclassified example with maximum weight given by Adaboost
    misclass = sorted_pairs[0][0]
    #weight update
    w = w + labels[misclass]*X[misclass]
    #labels prediction
    pred_labels = [1 if x>0 else -1 for x in [np.dot(w,x) for x in X]]
    i+=1
    if (i>201):
      break
return pred_labels
```

---

1- $[0.9, 1.0]$

2- $[0.8, 0.9)$

3- $[0.7, 0.8)$