Intro. to Machine Learning

Part 2:

1. Why does a decision tree have a tendency to overfit to the training set? Is it possible for a decision tree to reach a 100% accuracy in the training set? please explain. List and describe at least 3 strategies we can use to reduce the risk of overfitting of a decision tree.

   Decision trees are prone to overfitting, especially when the trees are deep. This is because as the trees go deeper, they are focusing on the specificity of small samples. Yes, when we didn't set any limit of the depth, it is possible for a decision tree to reach a 100% accuracy.

   1. Set the max depth (avoid the tree to go to the full steps)
   2. Post-pruning (prune the tree after it grows to the full steps, based on the Cost Complexity Index)
   3. Random forest (grow many trees with randomness, the final classification is from the voting of all trees)

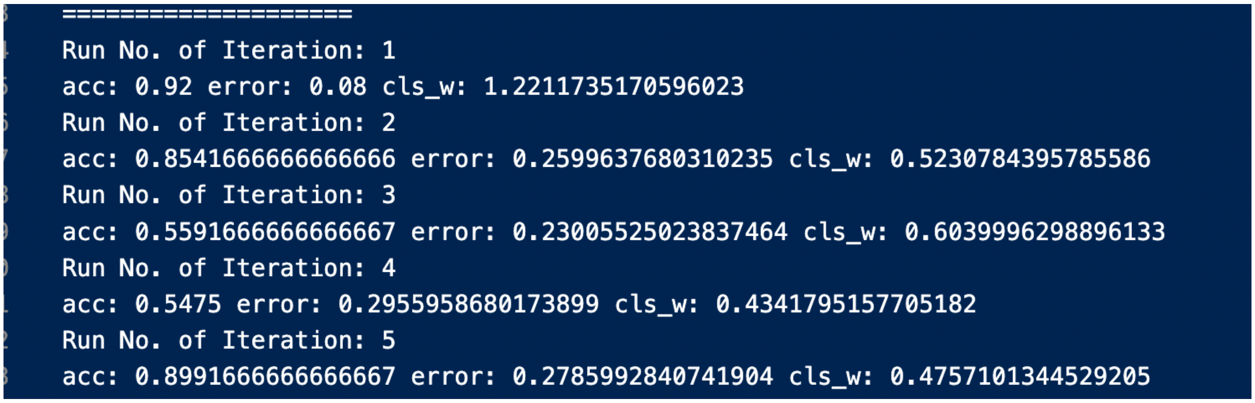
2. This part consists of three True/False questions. Answer True/False for each question and briefly explain your answer.

   a. In AdaBoost, weights of the misclassified examples go up by the same multiplicative factor.

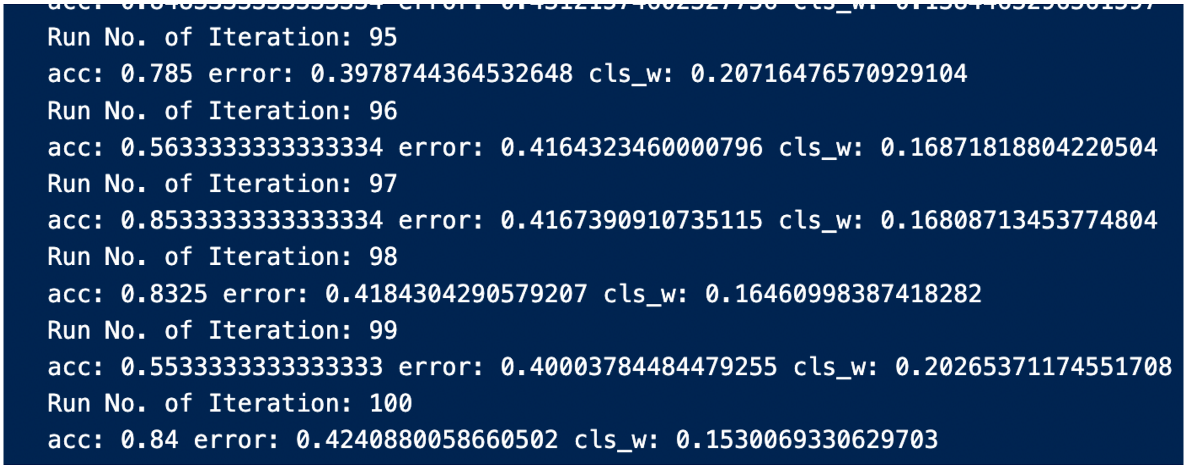
      True, they are all multiplied by *exp(0.5 * log((1-error) / error))*

b. In AdaBoost, weighted training error ε_t of the t_{th} weak classifier on training data with weights D_t tends to increase as a function of t.

True, as the training error tends to increase as the iterations grow.



```
====================
Run No. of Iteration: 1
acc: 0.92 error: 0.08 cls_w: 1.2211735170596023
Run No. of Iteration: 2
acc: 0.8541666666666666 error: 0.2599637680310235 cls_w: 0.5230784395785586
Run No. of Iteration: 3
acc: 0.5591666666666667 error: 0.23005525023837464 cls_w: 0.6039996298896133
Run No. of Iteration: 4
acc: 0.5475 error: 0.2955958680173899 cls_w: 0.4341795157705182
Run No. of Iteration: 5
acc: 0.8991666666666667 error: 0.2785992840741904 cls_w: 0.4757101344529205
```

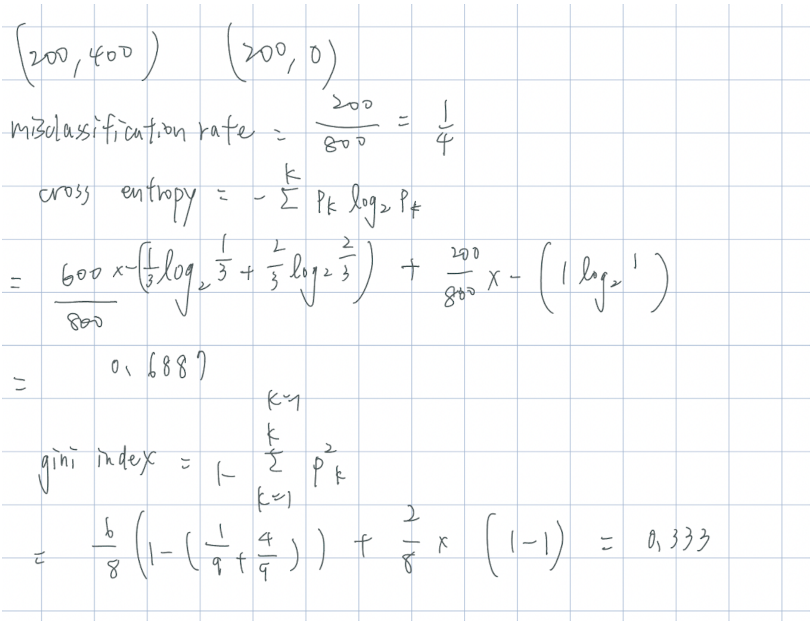


```
Run No. of Iteration: 95
acc: 0.785 error: 0.3978744364532648 cls_w: 0.20716476570929104
Run No. of Iteration: 96
acc: 0.5633333333333334 error: 0.4164323460000796 cls_w: 0.16871818804220504
Run No. of Iteration: 97
acc: 0.8533333333333334 error: 0.4167390910735115 cls_w: 0.16808713453774804
Run No. of Iteration: 98
acc: 0.8325 error: 0.4184304290579207 cls_w: 0.16460998387418282
Run No. of Iteration: 99
acc: 0.5533333333333333 error: 0.40003784484479255 cls_w: 0.20265371174551708
Run No. of Iteration: 100
acc: 0.84 error: 0.4240880058660502 cls_w: 0.1530069330629703
```

c. AdaBoost will eventually give zero training error regardless of the type of weak classifier it uses, provided enough iterations are performed.

False, the training error will approach close to 0.5 as the training iterations.

3. Consider a data set comprising 400 data points from class C_1 and 400 data points from class C_2. Suppose that a tree model A splits these into (200, 400) at the first leaf node and (200, 0) at the second leaf node, where (n, m) denotes that n points are assigned to C_1 and m points are assigned to C_2. Similarly, suppose that a second tree model B splits them into (300, 100) and (100, 300). Evaluate the misclassification rates for the two trees and hence show that they are equal. Similarly, evaluate the cross-entropy $Entropy = -\sum_{k=1}^{K} p_k log_2 p_k$ and Gini index $Gini = 1 - \sum_{k=1}^{K} p_k^2$ for the two trees. Define p_k to be the proportion of data points in region R assigned to class k, where k = 1, . . . , K.



(200, 400) (200, 0)

misclassification rate $= \frac{200}{800} = \frac{1}{4}$

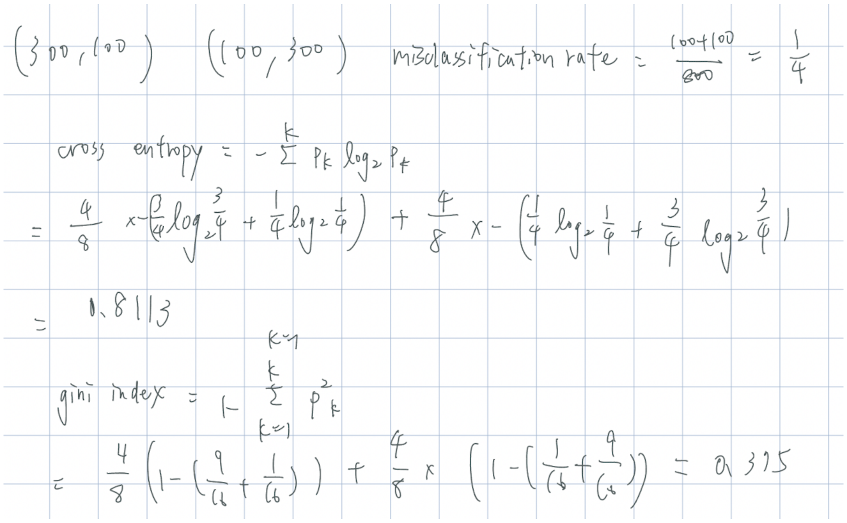
cross entropy $= -\sum^{k} p_k \log_2 p_k$

$= \frac{600}{800} \times -\left(\frac{1}{3}\log_2\frac{1}{3} + \frac{2}{3}\log_2\frac{2}{3}\right) + \frac{200}{800} \times -\left(1 \log_2 1\right)$

$= 0.6887$

gini index $= 1 - \overset{k=1}{\sum_{k=1}^{k}} p_k^2$

$= \frac{6}{8}\left(1-\left(\frac{1}{9}+\frac{4}{9}\right)\right) + \frac{2}{8} \times \left(1-1\right) = 0.333$



(300, 100) (100, 300) misclassification rate $= \frac{100+100}{800} = \frac{1}{4}$

cross entropy $= -\sum^{k} p_k \log_2 p_k$

$= \frac{4}{8} \times -\left(\frac{3}{4}\log_2\frac{3}{4} + \frac{1}{4}\log_2\frac{1}{4}\right) + \frac{4}{8} \times -\left(\frac{1}{4}\log_2\frac{1}{4} + \frac{3}{4}\log_2\frac{3}{4}\right)$

$= 0.8113$

gini index $= 1 - \overset{k=1}{\sum_{k=1}^{k}} p_k^2$

$= \frac{4}{8}\left(1-\left(\frac{9}{16}+\frac{1}{16}\right)\right) + \frac{4}{8} \times \left(1-\left(\frac{1}{16}+\frac{9}{16}\right)\right) = 0.375$