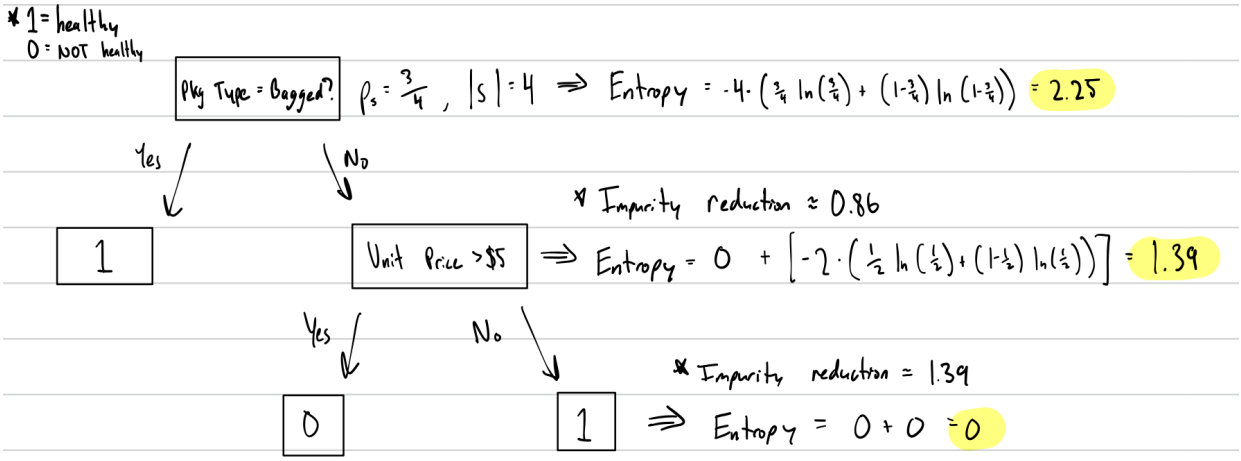
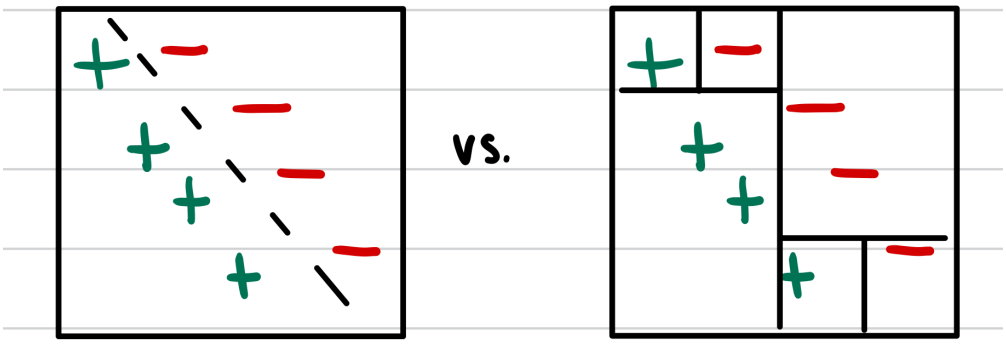
1 Decision Trees

Problem A



Problem B

No, a decision tree is not always preferred for classifications problems. See the following image for a simple example (same example from lecture 5). In this case, a linear SVM easily finds the maximum margin. On the other hand, a decision tree would require complex axis-aligned partitioning in order to classify the data perfectly which would not be preferred in this scenario.



Problem C

i.

See the following image for the drawing of the resulting tree in this scenario. The classification error is calculated through the following process beginning with entropy calculations:

$$L(S') = |S'|(1 - p_{S'}^2 - (1 - p_{S'}^2)) \quad \text{(Gini index as impurity measure)}$$

Thus, the entropy of the root level is clearly

$$L(S) = |4| \cdot (1 - 0.5^2 - 0.5^2) = 4 \cdot (0.5) = 2$$

At this point, with either decision there are two points in each section such that 1 point is misclassified in each. So, entropy at the next level would be

$$L(S) = |2| \cdot (1 - 0.5^2 - 0.5^2) + |2| \cdot (1 - 0.5^2 - 0.5^2) = 1 + 1 = 2$$

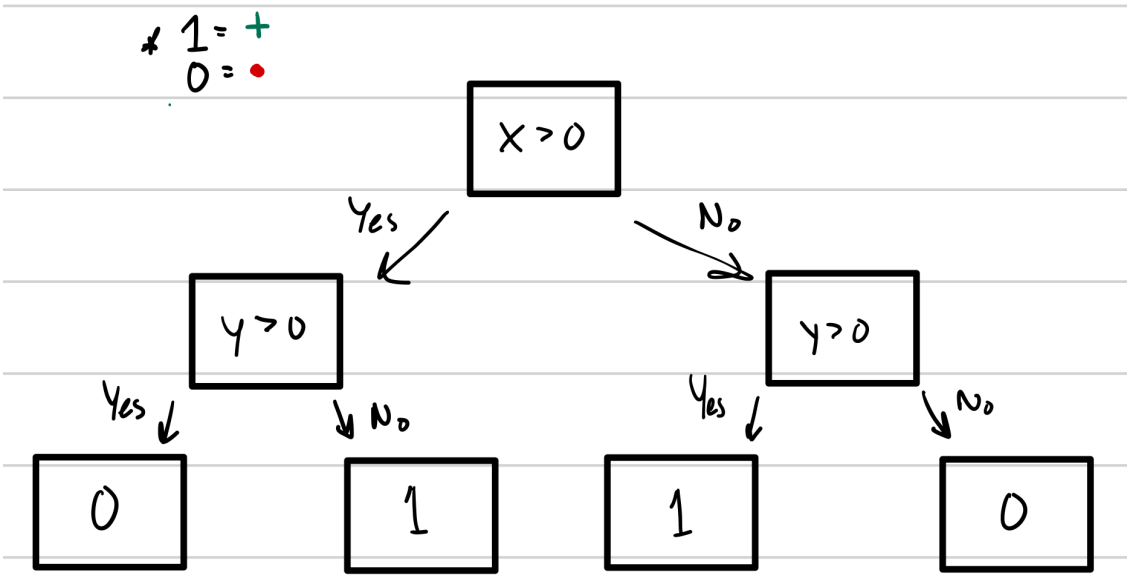
Therefore, we have no reduction in impurity with any possible splitting of the root node. Thus, the resulting tree is simply the root level node as depicted below. The number of misclassified points is 2 and the number of total points is 4, giving us the classification error of $\frac{2}{4} = \frac{1}{2}$.

ii.
See the following image for the drawing of a two-level decision tree that classifies the dataset with zero classification error. We could modify the Gini index slightly such that it would be an impurity measure that satisfies the necessary requirements:

$$L(S') = |S'|^2(1 - p_{S'}^2 - (1 - p_{S'}^2))$$

Notice that the difference is the $|S'|$ term is now $|S'|^2$ which allows us to use the same stopping condition with the drawn tree using top-down greedy induction.

- **Pros**: The points in this dataset would be classified correctly such that the decision tree would be able to split as much as needed even when the number of correctly classified datapoints stays the same with a split which could be the case with similar arbitrary datasets.
- **Cons**: A clear con with this impurity measure is its likeliness to over-fit since it would not generalize well with arbitrary datasets. This is related to the fact that this measure encourages splitting relative to the standard Gini index which might cause the model to over-fit the data as mentioned.



iii.

With $N = 100$ points in some 2-D dataset, the largest number of unique thresholds required in order to achieve zero classification training error is $N - 1 = 99$. In the worst case scenario, we can imagine that 99 internal nodes are needed in order to produce the 99 decision boundaries needed to separate the 100 points. In other words, for any number of points N in this scenario, the number of unique thresholds, or splits, needed is at most $N - 1$. With 2 points, we only need 1 unique threshold at the most to achieve zero classification training error. With 3 points, only 2 thresholds are needed at the most. So on and so forth...

Problem D

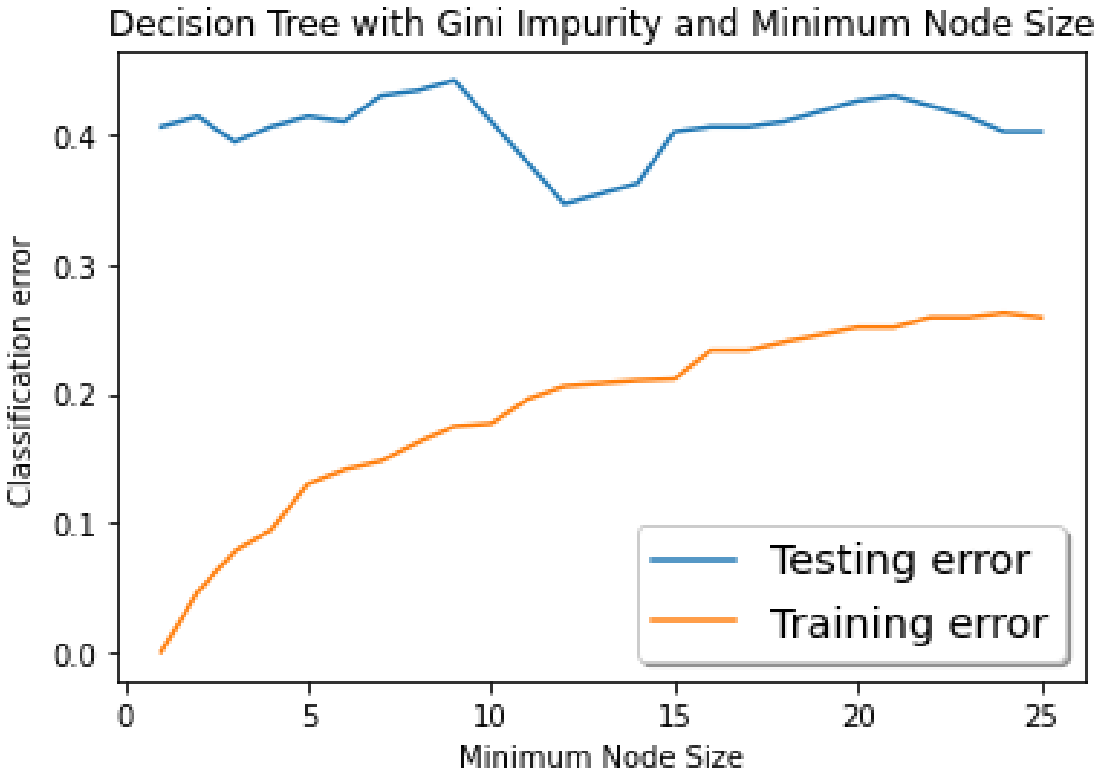
The worse-case complexity of the number of possible splits that need to be considered in order to find the one that most reduces impurity is $O(DN)$. Following lecture 5, we know that this is true since each of the N training points needs to be considered with each of the D continuous features. Again, this is in the worst case scenario (reinforced by big-O notation).

2 Overfitting Decision Trees

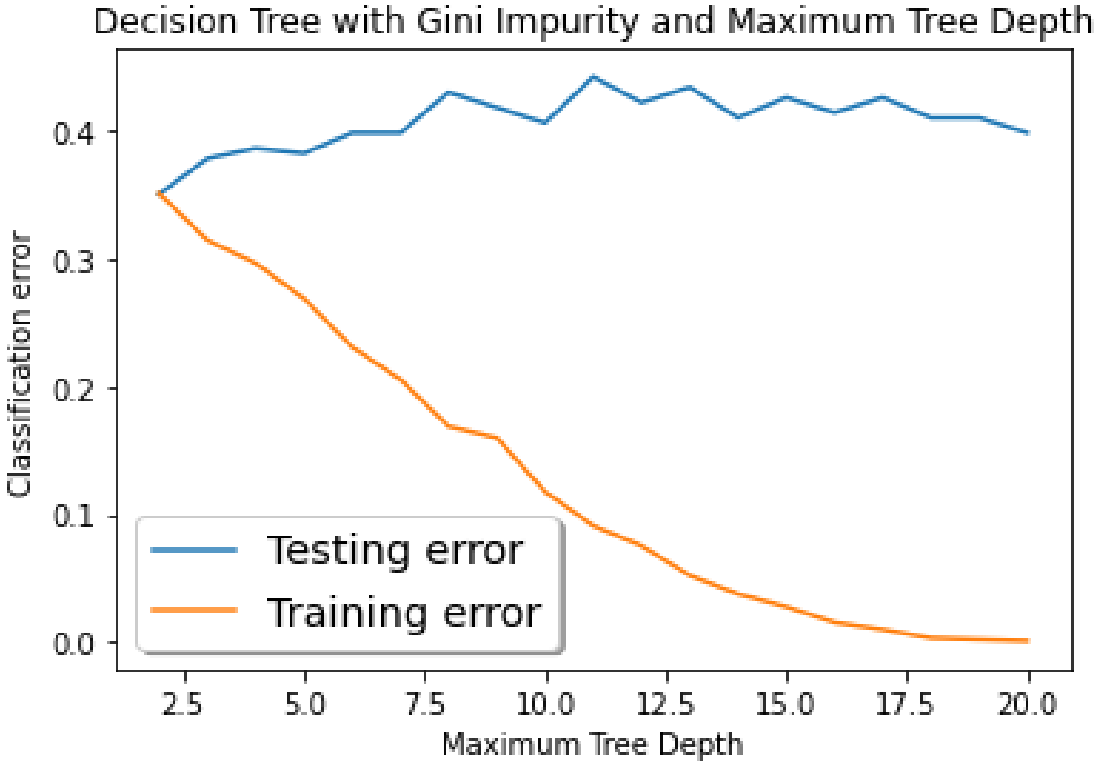
Question 2 Code

Problem A

i.



ii.

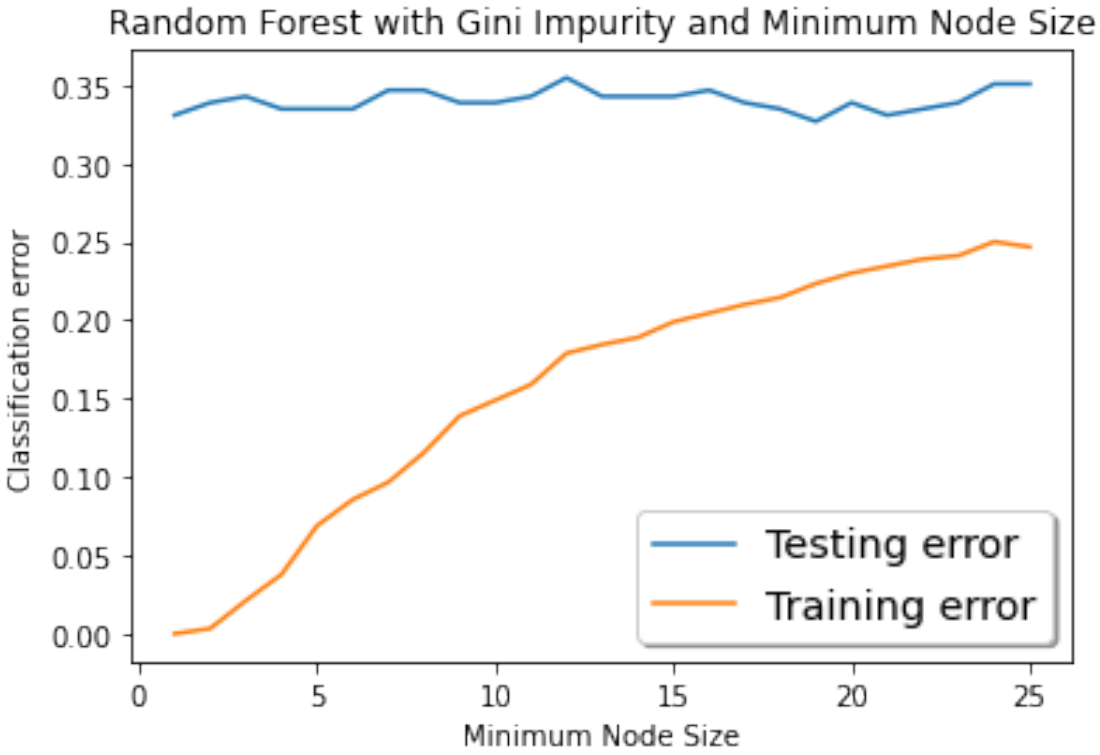


Problem B

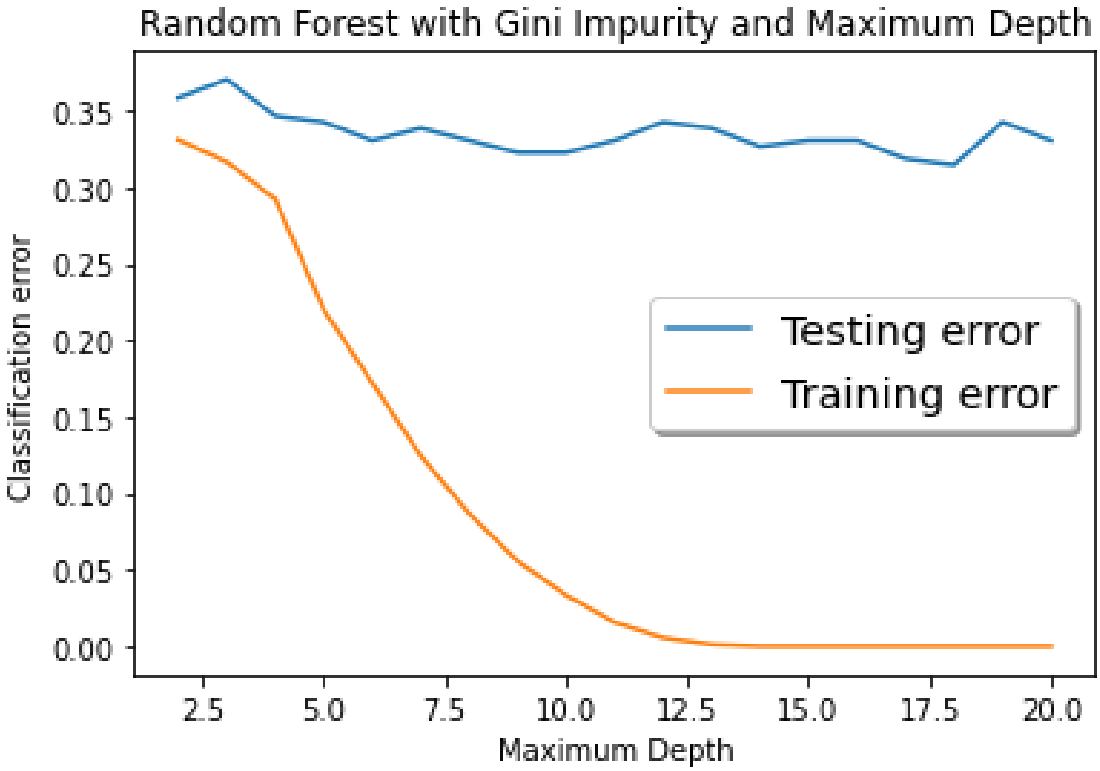
Let us consider *A.i* with the minimal leaf node size parameter. We have that the parameter value that minimizes the test error in this case is 12. Early stopping allows for better prevention of over-fitting. This is shown in the plot such that after the minimum node size of 12, the testing error increases, suggesting over-fitting. Thus, early stopping in the 10-15 range for minimum node size would likely result in optimal performance since this is where the testing error generally decreases. On the other hand, if the stopping occurs too early, then under-fitting is more likely to occur due to the rising of both testing and training error in the 0-10 range for minimum node size, as shown in the plot.

Problem C

i.


Random Forest with Gini Impurity and Minimum Node Size
Classification error
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0
5
10
15
20
25
Minimum Node Size
Testing error
Training error

ii.



Problem D

Let us consider *C.ii* with the maximum depth parameter. We have that the parameter value that minimizes the random forest test error in this case is 18. Early stopping in this case does not seem to much of a beneficial impact since the testing error relatively stays at the same level regardless of the value of the maximum depth parameter. We also see that the training error becomes zero around the value of 12.5 for this parameter. Thus, it does not appear that early stopping prevents over-fitting in this case, in contrast to problem *B*.

Problem E

It appears that the testing errors in the random forest curves are more constant (less fluctuation) relative to those in the decision tree curves. It also appears that early stopping has more of a beneficial impact with the decision tree curves in comparison to the random forest curves mainly due to these trends we see with the testing errors. This all makes sense because the goal of random forests is to reduce the variance as trees are further decorrelated. Let us also note that the training errors are similar between the two sets of curves.

Problem F

Both options completed for both **Problem A** and **Problem C**.

Problem G

For *A.ii*, the maximum depth parameter value of 2 minimizes the decision tree error. For *C.i*, the minimum leaf node size parameter value of 19 minimizes the random forest test error.

3 The AdaBoost Algorithm

Question 3 Code

Problem A

Want to show: $E = \frac{1}{N}\sum_{i=1}^{N} \exp(-y_i f(x_i)) \geq \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}(H(x_i) \neq y_i)$

* Equivalent to showing $\exp(-y_i f(x_i)) \geq \mathbb{1}(H(x_i) \neq y_i)$ for each (x_i, y_i)

- Case 1: y_i & $f(x_i)$ have same signs
  - $\Rightarrow \mathbb{1}(H(x_i) \neq y_i) = 0$
  - $\Rightarrow -y_i f(x_i)$ is non-positive
    - $\Rightarrow \exp(-y_i f(x_i)) \geq 0$
  - $\Rightarrow \exp(-y_i f(x_i)) \geq \mathbb{1}(H(x_i) \neq y_i)$
- Case 2: y_i & $f(x_i)$ have opposite signs
  - $\Rightarrow \mathbb{1}(H(x_i) \neq y_i) = 1$
  - $\Rightarrow -y_i f(x_i)$ is non-negative
    - $\Rightarrow \exp(-y_i f(x_i)) \geq 1$ * $\exp(0) = 1$
  - $\Rightarrow \exp(-y_i f(x_i)) \geq \mathbb{1}(H(x_i) \neq y_i)$

$\Rightarrow \exp(-y_i f(x_i)) \geq \mathbb{1}(H(x_i) \neq y_i)$ holds in both cases for each (x_i, y_i)

$\Rightarrow E = \frac{1}{N}\sum_{i=1}^{N} \exp(-y_i f(x_i)) \geq \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}(H(x_i) \neq y_i)$ holds

Problem B

Find $D_{T+1}(i)$ in terms of $Z_t, \alpha_t, x_i, y_i, h_t$

- Given $t \in \{1, \dots, T\}$, $Z_t = \sum_{i=1}^{N} D_t(i) \exp\left(-\alpha_t y_i h_t(x_i)\right)$
- We know $D_1(i) = \frac{1}{N}$, $D_{t+1}(i) = \frac{D_t(i) \exp\{-\alpha_t y_i h_t(x_i)\}}{Z_t}$

$$\Rightarrow D_2(i) = \frac{D_1(i) \exp\left(-\alpha_1 y_i h_1(x_i)\right)}{Z_1} = \frac{1}{N}\left(\frac{\exp\left(-\alpha_1 y_i h_1(x_i)\right)}{Z_1}\right)$$

$$\Rightarrow D_3(i) = \frac{D_2(i) \exp\left(-\alpha_2 y_i h_2(x_i)\right)}{Z_2} = \frac{1}{N}\left(\frac{\exp\left(-\alpha_1 y_i h_1(x_i)\right)}{Z_1}\right)\left(\frac{\exp\left(-\alpha_2 y_i h_2(x_i)\right)}{Z_2}\right)$$

$$\Rightarrow D_4(i) = \dots$$

$$\Rightarrow D_{T+1} = \frac{1}{N} \prod_{t=1}^{T} \frac{\exp\left(-\alpha_t y_i h_t(x_i)\right)}{Z_t}$$

Problem C

Want to show that $E = \sum_{i=1}^{N} \frac{1}{N} e^{\sum_{t=1}^{T}\left(-\alpha_t y_i h_t(x_i)\right)}$

- We know $f(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$ from Problem A

$$\Rightarrow E = \frac{1}{N} \sum_{i=1}^{N} \exp\left(-y_i f(x_i)\right)$$

$$= \frac{1}{N} \sum_{i=1}^{N} \exp\left(-y_i \sum_{t=1}^{T} \alpha_t h_t(x_i)\right)$$

$$= \sum_{i=1}^{N} \frac{1}{N} \exp\left(\sum_{t=1}^{T} -\alpha_t y_i h_t(x_i)\right)$$

$$= \sum_{i=1}^{N} \frac{1}{N} e^{\sum_{t=1}^{T} -\alpha_t y_i h_t(x_i)} \quad \checkmark$$

Problem D

Want to show $E = \prod_{t=1}^{T} Z_t$ (recall $\sum_{i=1}^{N} D_t(i) = 1$)

- We know $E = \sum_{i=1}^{N} \frac{1}{N} \exp\left(\sum_{t=1}^{T} -\alpha_t y_i h_t(x_i)\right)$, $D_{T+1} = \frac{1}{N} \prod_{t=1}^{T} \frac{\exp(-\alpha_t y_i h_t(x_i))}{Z_t}$

$$\Rightarrow D_{T+1}(i) = \frac{1}{N} \prod_{t=1}^{T} \left(\exp(-\alpha_t y_i h_t(x_i)) \cdot \frac{1}{Z_t} \right)$$

$$D_{T+1}(i) = \frac{1}{N} \prod_{t=1}^{T} \exp(-\alpha_t y_i h_t(x_i)) \cdot \prod_{t=1}^{T} \frac{1}{Z_t}$$

$$D_{T+1}(i) = \frac{1}{N} \prod_{t=1}^{T} \exp(-\alpha_t y_i h_t(x_i)) \cdot \frac{1}{\prod_{t=1}^{T} Z_t}$$

$$\left(\prod_{t=1}^{T} Z_t\right) D_{T+1}(i) = \frac{1}{N} \prod_{t=1}^{T} \exp(-\alpha_t y_i h_t(x_i))$$

$$\Rightarrow E = \sum_{i=1}^{N} \frac{1}{N} \exp\left(\sum_{t=1}^{T} -\alpha_t y_i h_t(x_i)\right) = \sum_{i=1}^{N} \frac{1}{N} \prod_{t=1}^{T} \exp(-\alpha_t y_i h_t(x_i))$$

$$\Rightarrow E = \sum_{i=1}^{N} \left(\prod_{t=1}^{T} Z_t\right) D_{T+1}(i)$$

$$= \prod_{t=1}^{T} Z_t \cdot \sum_{i=1}^{N} D_{T+1}(i)$$

$$= \prod_{t=1}^{T} Z_t \cdot 1$$

$$E = \prod_{t=1}^{T} Z_t \quad \checkmark$$

Problem E

Want to show Z_t can be written as:

$$Z_t = (1-\epsilon_t)\exp(-\alpha_t) + \epsilon_t \exp(\alpha_t)$$

where $\epsilon_t = \sum_{i=1}^{N} D_t(i)\,\mathbb{1}(h_t(x_i) \neq y_i)$, $Z_t = \sum_{i=1}^{N} D_t(i)\exp(-\alpha_t y_i h_t(x_i))$

- Case 1: $h_t(x_i) \neq y_i$

$\Rightarrow h_t(x_i) y_i = -1$ $\Rightarrow \mathbb{1}(h_t(x_i) \neq y_i) = 1$

$\rightarrow Z_t = \sum_{i=1}^{N} D_t(i)\exp(\alpha_t)$ $\Rightarrow \epsilon_t = \sum_{i=1}^{N} D_t(i)$

$= \exp(\alpha_t)\sum_{i=1}^{N} D_t(i)$ $\Rightarrow \epsilon_t = 1$

$Z_t = \exp(\alpha_t)$ ✓ $\Rightarrow Z_t = (1-\epsilon_t)\exp(-\alpha_t) + \epsilon_t \exp(\alpha_t)$

$= (1-1)\exp(-\alpha_t) + \exp(\alpha_t)$

$Z_t = \exp(\alpha_t)$ ✓

- Case 2: $h_t(x_i) = y_i$

$\Rightarrow h_t(x_i) y_i = 1$ $\Rightarrow \mathbb{1}(h_t(x_i) \neq y_i) = 0$

$\Rightarrow Z_t = \sum_{i=1}^{N} D_t(i)\exp(-\alpha_t)$ $\Rightarrow \epsilon_t = 0$

$= \exp(-\alpha_t)\sum_{i=1}^{N} D_t(i)$ $\Rightarrow Z_t = (1-\epsilon_t)\exp(-\alpha_t) + \epsilon_t \exp(\alpha_t)$

$Z_t = \exp(-\alpha_t)$ ✓ $= (1-0)\exp(-\alpha_t) + (0)\exp(\alpha_t)$

$Z_t = \exp(-\alpha_t)$ ✓

$\Rightarrow Z_t = (1-\epsilon_t)\exp(-\alpha_t) + \epsilon_t \exp(\alpha_t)$ is a valid form to express Z_t

Problem F

Want to show that choosing α_t greedily to minimize Z_t at each iteration leads to choices:

$$\alpha_t^* = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$$

- We know $Z_t = (1-\epsilon_t)\exp(-\alpha_t) + \epsilon_t \exp(\alpha_t)$

$$\Rightarrow \frac{\partial}{\partial \alpha_t}(Z_t) = 0$$

$$\Rightarrow 0 = \frac{\partial}{\partial \alpha_t}\left((1-\epsilon_t)\exp(-\alpha_t) + \epsilon_t \exp(\alpha_t)\right)$$

$$= -(1-\epsilon_t)\exp(-\alpha_t) + \epsilon_t \exp(\alpha_t)$$

$$\epsilon_t \exp(\alpha_t) = (1-\epsilon_t)\exp(-\alpha_t)$$

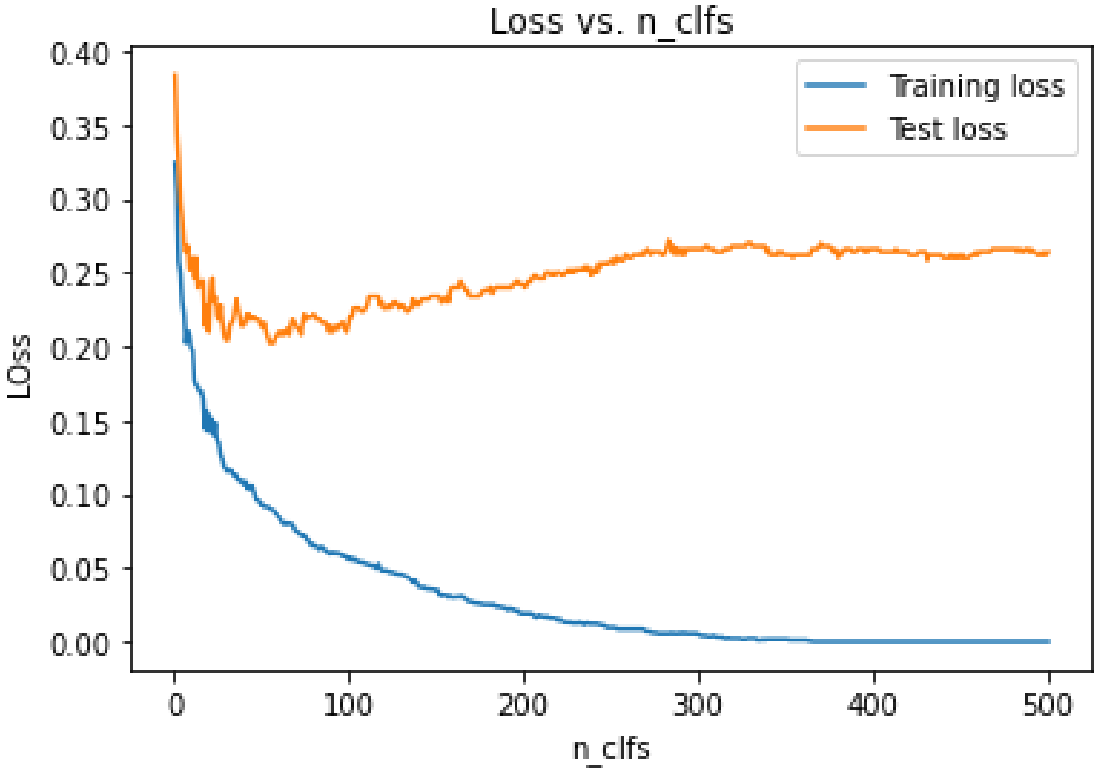
$$\epsilon_t \exp(2\alpha_t) = (1-\epsilon_t)$$

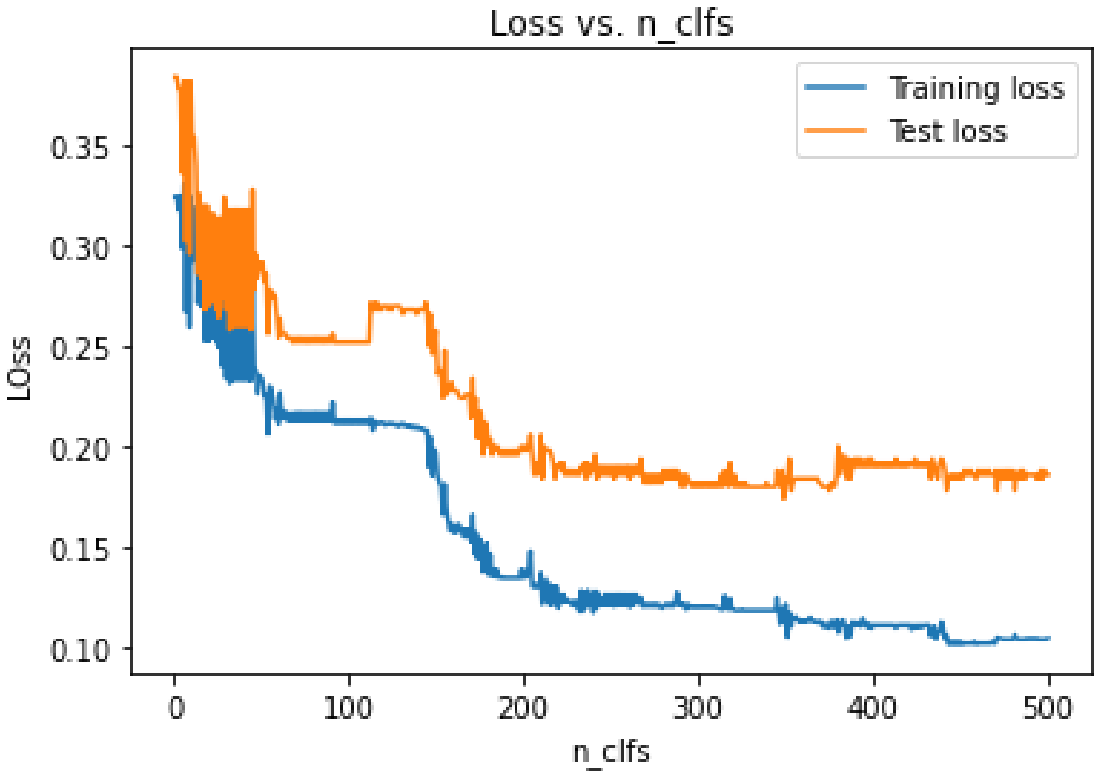
$$\ln(\exp(2\alpha_t)) = \ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$$

$$2\alpha_t = \ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$$

$$\alpha_t^* = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right) \quad \checkmark$$

Problem G


Loss vs. n_clfs
Training loss
Test loss
LOss
n_clfs
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0
100
200
300
400
500



Problem H

For the loss curves for gradient boosting (1st plot from G), we less fluctuation and more smoothness. Also, the training loss approaches 0 in the final quarter of the plot range while the testing error is not at its minimum in this range. Thus, there is evidence of over-fitting in this sense. On the other hand, for the loss curves with AdaBoost (2nd plot), we see much more fluctuation (especially in the earlier ranges) and less smoothness. The differences we see in smoothness are perhaps due to gradient boosting fitting on the residuals. Furthermore, the AdaBoost training loss curve does not approach 0 (approaches approx. 0.1), in contrast to gradient boosting. The final values for the test loss curve are lower than those with gradient boosting, and there is less evidence for over-fitting, thus it appears that AdaBoost performed better overall.

Problem I

For gradient boosting, the final training loss value was 0 and the final testing loss value was 0.264. For AdaBoost, the final training loss value was 0.1045 and the final testing loss value was 0.186. While gradient boosting had a better performance with the training set, AdaBoost had a better performance on the testing set. Overall, the performances were similar. However, considering our comments about the over-fitting evidence for gradient boosting in addition to these final values, it seems like AdaBoost is the more optimal choice.

Problem J

For AdaBoost, the dataset weights are the largest for the points that are more likely to be misclassified (closer to decision boundary), while the weights are the smallest for the points that are more likely to be classified properly (further from decision boundary).