Eli Pinkus
CS 155
Set 3

Problem 1

Part A

Note $\log x$ denotes the natural logarithm of x

Let $S = \{1,2,3,4\}$

To begin we can calculate the entropy with S and $p_S = \frac{3}{4}$:

$$L(S) = -4\left[\frac{3}{4}\log\frac{3}{4} + \frac{1}{4}\log\frac{1}{4}\right] = 2.25$$

Root Node

We have 3 binary inputs for each example so we begin at the root node and search for the way in which we can split the data according to one of those inputs to induce the greatest reduction in loss.

If we divide based on package type, we would have

$$S_{\text{canned}} = \{1,4\} \Rightarrow p_{S_{\text{canned}}} = \frac{1}{2}$$

$$\Rightarrow L(S_{\text{canned}}) = -2\left(\frac{1}{2}\log\frac{1}{2} + \left(\frac{1}{2}\log\frac{1}{2}\right)\right) = -2\log\frac{1}{2} \approx 1.39$$

$$S_{\text{bagged}} = \{2,3\} \Rightarrow p_{S_{\text{bagged}}} = 1 \Rightarrow L\left(S_{\text{bagged}}\right) = 0$$

$$L(S) = -2\log\frac{1}{2} \approx 1.39$$

If we divide based on whether or not the unit price is $> \$5$ we would have

$$S_{\leq\$5} = \{3,4\} \Rightarrow p_{S_{\leq\$5}} = 1$$

$$\Rightarrow L(S_{\leq\$5}) = -2[\log 1 + 0 \cdot \log 0] = 0$$

$$S_{>\$5} = \{1,2\} \Rightarrow p_{S_{>\$5}} = \frac{1}{2}$$

$$\Rightarrow L(S_{>\$5}) = -2\left[\frac{1}{2}\log\frac{1}{2} + \frac{1}{2}\log\frac{1}{2}\right] = -2\log\frac{1}{2} \approx 1.39$$

Eli Pinkus
CS 155
Set 3

$$L(S) = -2\log\frac{1}{2} \approx 1.39$$

If we divide based on whether or not the item contains > 5 grams of fat we would have

$$S_{>5} = \{1,3\} \Rightarrow p_{S_{>5}} = \frac{1}{2}$$

$$\Rightarrow L(S_{>5}) = -2\log\frac{1}{2} \approx 1.39$$

$$S_{\leq 5} = \{2,4\} \Rightarrow p_{S_{\leq 5}} = 1$$

$$\Rightarrow L(S_{\leq 5}) = 0$$

$$L(S) = -2\log\frac{1}{2} \approx 1.39$$

They all yield the same information gain so we choose arbitrarily

Say we divide based on whether or not the item contains > 5 grams of fat we would have

$$S_{>5} = \{1,3\}, \qquad S_{\leq 5} = \{2,4\}$$

Second Node

Since the input features are binary, we can easily see that dividing again by fat content would result in a 0 information gain so we ignore that possibility.

If we further divide based on package type we have:

For $S_{>5}$

$$S_{\text{canned}} = \{1\} \Rightarrow p_{\text{canned}} = 0$$

$$\Rightarrow L(S_{\text{canned}}) = 0$$

$$S_{\text{bagged}} = \{3\} \Rightarrow p_{\text{bagged}} = 1$$

$$\Rightarrow L(S_{\text{canned}}) = 0$$

For $S_{\leq 5}$

$$S_{\text{canned}} = \{4\} \Rightarrow p_{\text{canned}} = 1$$

Eli Pinkus
CS 155
Set 3

$$\Rightarrow L(S_{\text{canned}}) = 0$$

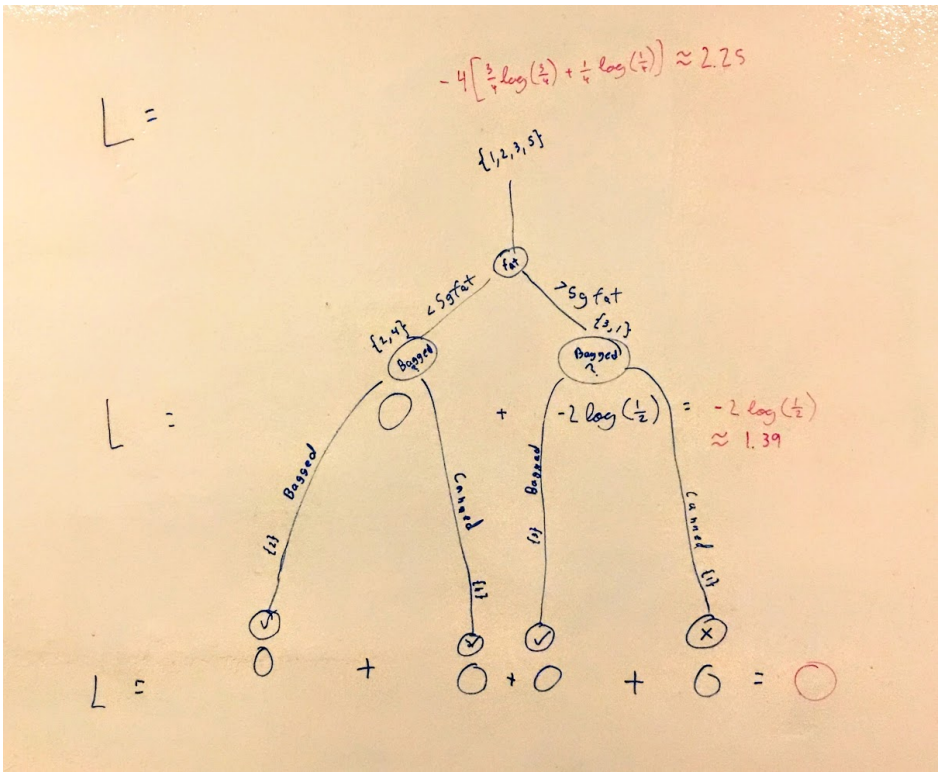
$$S_{\text{bagged}} = \{2\} \Rightarrow p_{S_{\text{bagged}}} = 1$$

$$\Rightarrow L\left(S_{\text{bagged}}\right) = 0$$

So the loss for the entire node if we choose this split would be:

$$L(S) = 0$$

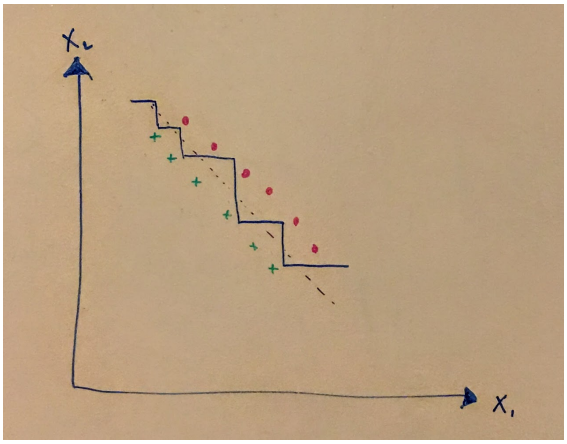
We can't do better than this so we can choose this as our second split. Our tree now has zero classification and training is complete.



Part B

A decision tree is not always preferred. There are cases were a simple linear classifier could do better than a fairly complex decision tree. This is partially due to the fact that decision trees result in dividing lines that are always axis aligned which makes it difficult for them to express diagonal natured decision boundaries.

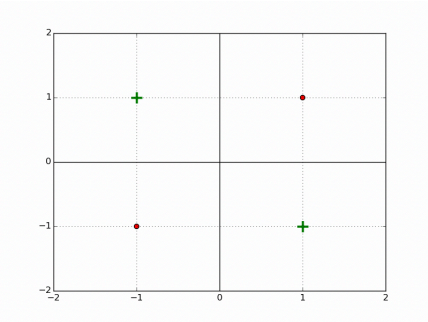
Eli Pinkus
CS 155
Set 3



Eli Pinkus
CS 155
Set 3

Part C

i)



I will denote the data set as $S = \{\text{top left, top right, bottom left, bottom right}\}$

For the data set as a whole we will evaluate the Gini index

$$L(S) = |S|[1 - p_S^2 - (1 - p_S)^2]$$

$$p_S = \frac{1}{2}$$

$$L(S) = 4\left[1 - \frac{1}{4} - \frac{1}{4}\right] = 2$$

For the vertical splits we need only consider the split down the middle as all other splits will evidently have the same gini index as above:

With the vertical split we have:

$$S_{\text{left}} = \{\text{top left, bottom left}\} \Rightarrow p_{S_{\text{left}}} = \frac{1}{2}$$

$$S_{\text{right}} = \{\text{top right, bottom right}\}$$

$$\Rightarrow L(S_{\text{left}}) = L(S_{\text{right}}) = 2\left[1 - \frac{1}{4} - \frac{1}{4}\right] = 1$$

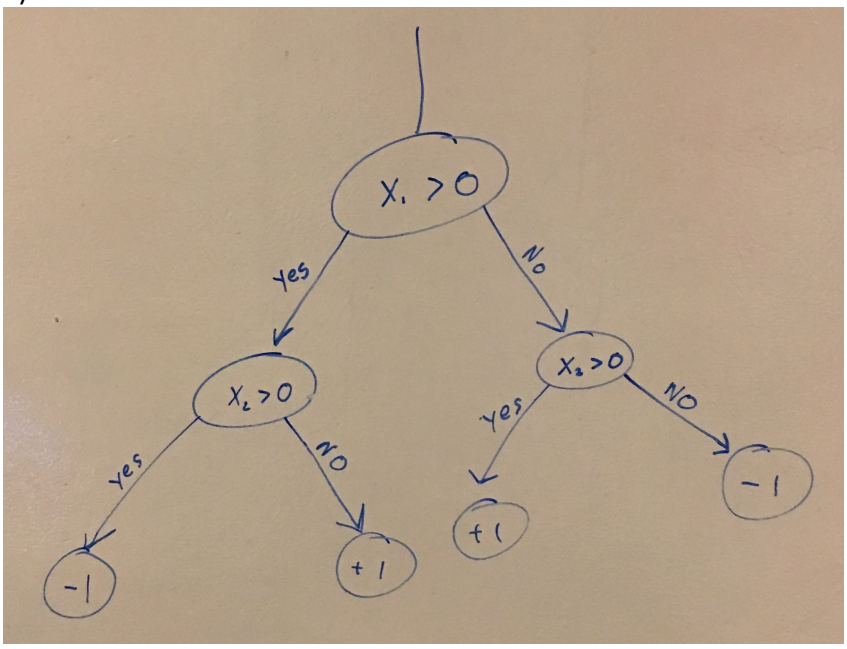
So

$$L(S) = L(S_{\text{left}}) + L(S_{\text{right}}) = 2$$

We use the same reasoning and we observe that the horizontal split down the middle has the same Gini index as the vertical split down the middle. Thus, there is no split of a node that results in any reduction in impurity and we don't make a split so two of the points will be misclassified. The tree is just a node with all points +1:

Eli Pinkus
CS 155
Set 3

ii)



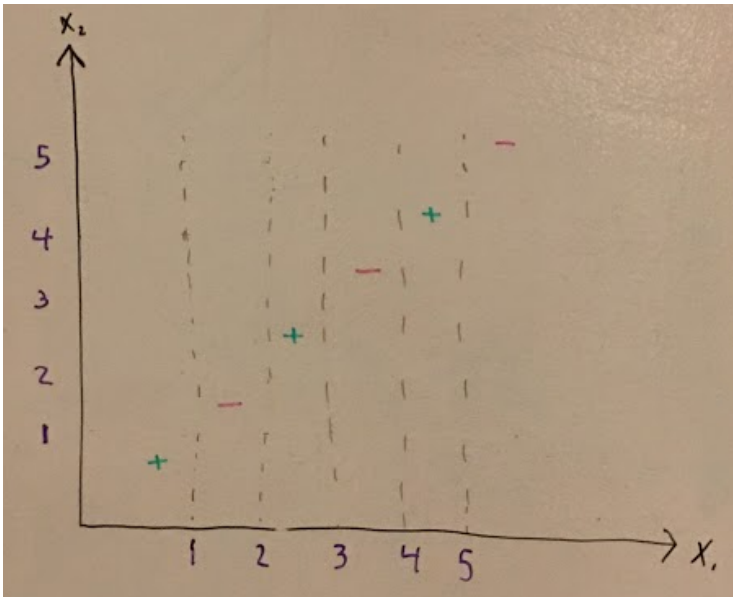
we could use

$$L(S') = |S'|^2\left(1 - p_{S'}^2, - (1 - p_{S'})^2\right)$$

In general this probably wouldn't be a great loss function to use with this stopping condition since it heavily favors depth in a tree which could lead to the tree overfitting very quickly and aggressively.

iii)
One could conceive of a 100 point data set that requires an axis aligned dividing line in between any two adjacent points. Here is an example with $N = 6$

Eli Pinkus
CS 155
Set 3



As we can see, you would need $N - 1 = 5$ splits in order to properly classify this data.

We can easily see that additional data point ⇒ additional split could be required.

Thus, in the general case you would need at minimum $N - 1$ splits in a given dimension to properly classify a similarly arranged data set. Thus, for $N = 100$, we could need up to 99 unique thresholds to reach zero classification training error.

Part D

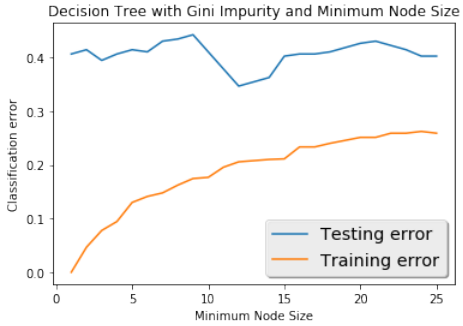
If we take any particular dimension in a D dimensional space with N data points, we know that we could produce as many as N splits in that dimension alone by placing a divider in any of the $N - 1$ gaps in between points as well as one divider that encompasses all of the data. Further for all D dimensions collectively, it could be as many as DN splits. This is the total number of possible queries which is the same as worst case complexity

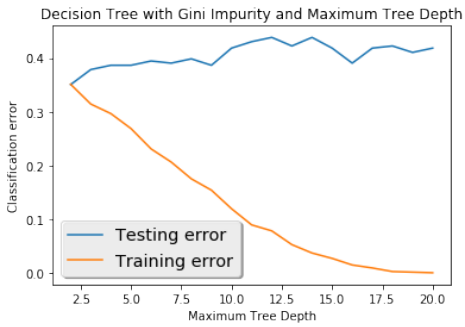
$$DN \Rightarrow O(DN)$$

Eli Pinkus
CS 155
Set 3

Question 2

Part B





Part C

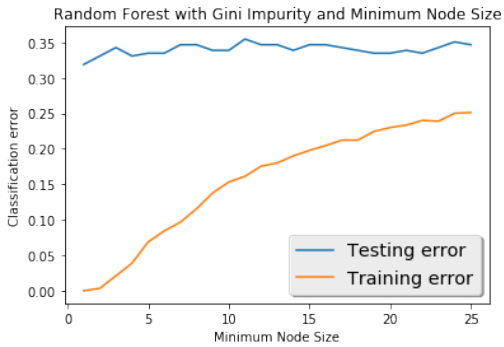
```
Test error minimized at min_samples_leaf = 12
Test error minimized at max_depth = 2
```

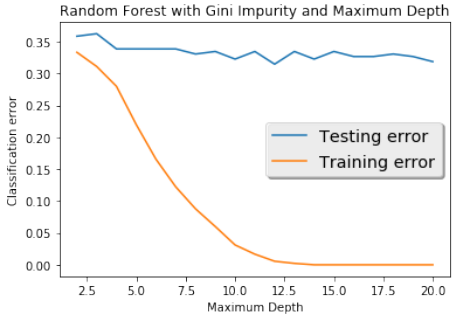
In both cases early stopping serves as regularizer that prevents overfitting. Although training error decreases with less early stopping, at some point test error increases which is emblematic of overfitting. Stopping early by either min samples leaf or max depth criteria, which essential limits the complexity of the model, helps mitigate this effect.

With min samples leaf, we put a restriction on how divided the space can be relative to the number of data points. In doing so, the model is less able to fit outliers and we have a min point around 12 min samples leaf where which a more restrictive regularize begins to produce underfitting.

With max depth, we put a restriction on the depth of the tree regardless of data points, which while it can prevent overfitting, seems like an arbitrary choice if the parameter is not chosen in a principled manor. Our graph indicates that a greater restriction on the depth of the tree helps out of sample performance with an best performance with depth restriction 2.

Part E





Eli Pinkus
CS 155
Set 3

Part F

```
Test error minimized at min_samples_leaf = 1
Test error minimized at max_depth = 10
```

In both cases, testing error seems to be best when the effects of early stopping are less. In the case of min samples in a leaf, allowing the tree to split as far as 1 point per split seems to give the best results in this case. With max depth restrictions, the testing error doesn't change a whole lot although there is a vague downward trend as max depth increases which seems makes sense with the results of min samples in leaf.

Part G

This behavior is different from the regular decision tree in that it seems like this model class is not seeing huge benefits from regularization. This makes sense considering the stochastic nature of Random Forest training. By randomly choosing a feature to split on at each step and then bootstrapping a handful of such trees, the variance of the hypothesis sets is drastically reduced which suggests that there should be less overfitting. In a sense, this algorithm already incorporates regularization which means that when further regularization is applied, the effects aren't necessarily helpful for performance.

Eli Pinkus
CS 155
Set 3

Question 3

Part A

We want:

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

where $H(x) = \operatorname{sign} f(x)$

We can show this inequality by simply showing

$$\exp\left(-y_i f(x_i)\right) \geq \mathbf{1}(H(x_i) \neq y_i)$$

for an arbitrary x_i, y_i

We have two cases:

Either $H(x_i) \neq y_i$ is true or it is false

If $H(x_i) = y_i$ we have $\mathbf{1}(H(x_i) \neq y_i) = 0$ and $\operatorname{sign} f(x_i) = \operatorname{sign} y_i \Rightarrow -y_i f(x_i) < 0$

So since y_i and $f(x_i)$ must be finite, we know that $0 < e^{-y_i f(x_i)} < 1$ so in this case

$$\exp\left(-y_i f(x_i)\right) \geq \mathbf{1}(H(x_i) \neq y_i)$$

If $H(x_i) \neq y_i$ we have $(H(x_i) \neq y_i) = 1$ and $\operatorname{sign} f(x_i) \neq \operatorname{sign} y_i \Rightarrow -y_i f(x_i) > 0$

$$\Rightarrow \exp(-y_i f(x_i) > 1 \geq \mathbf{1}(H(x_i) \neq y_i)$$

Thus, generally,

$$\exp\left(-y_i f(x_i)\right) \geq \mathbf{1}(H(x_i) \neq y_i)$$

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

Part B

We have the updating rule:

$$D_{T+1}(i) = \frac{D_T(i)\exp\{-\alpha_T y_i h_T(x_i)\}}{Z_T}$$

and

Eli Pinkus
CS 155
Set 3

$$Z_T = \sum_{i=1}^{N} D_T(i) \exp(-\alpha_T y_i h_T(x_i)$$

Because of this multiplicative recursive definition we can write:

$$D_{T+1} = \left(\prod_{t=1}^{T} \frac{\exp\{-\alpha_t y_i h_t(x_i)\}}{Z_T} \right) \cdot D_1(i)$$

We know $D_1(i) = \frac{1}{N}$

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

Problem C

We have:

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

we know that

$$f(x) = \sum_{i=1}^{T} \alpha_t h_t(x)$$

So substituting for $f(x)$ gives:

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

as desired.

Part D

We have

$$Z_T = \sum_{i=1}^{N} D_T(i) \exp\big(-\alpha_T y_i h_T(x_i)\big)$$

Subbing in for D_t from B:

Eli Pinkus
CS 155
Set 3

$$Z_T = \sum_{i=1}^{N}\left[\left(\prod_{t=1}^{T-1}\frac{\exp\{-\alpha_t y_i h_t(x_i)\}}{Z_t}\right)\cdot\frac{1}{N}\exp\left(-\alpha_T y_i h_T(x_i)\right)\right]$$

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

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

moving terms gives

$$Z_T\cdot\left(\prod_{t=1}^{T-1}Z_t\right) = \frac{1}{N}\sum_{i=1}^{N}\left[\exp\left(-y_i\sum_{t=1}^{T}\alpha_t h_t(x_i)\right)\right]$$

subbing in from C gives:

$$Z_T\cdot\left(\prod_{t=1}^{T-1}Z_t\right) = E \Rightarrow E = \prod_{t=1}^{T}Z_t$$

As desired.

Part E

$$Z_T = \sum_{i=1}^{N} D_T(i)\exp\left(-\alpha_T y_i h_T(x_i)\right)$$

We can break this up into the two cases using the indicator function:

$$= \sum_{i=1}^{N} D_T(i)\mathbf{1}(h_T(x_i)\neq y_i)\exp\left(-\alpha_T y_i h_T(x_i)\right) + \sum_{i=1}^{N} D_T(i)\mathbf{1}(h_T(x_i)= y_i)\exp\left(-\alpha_T y_i h_T(x_i)\right)$$

We know if $h_T(x_i)\neq y_i$ then $y_i h_T(x_i) = -1$ else if $h_T(x_i) = y_i$ then $y_i h_T(x_i) = 1$
So,

$$Z_T = \sum_{i=1}^{N} D_T(i)\mathbf{1}(h_T(x_i)\neq y_i)\exp(\alpha_T) + \sum_{i=1}^{N} D_T(i)\mathbf{1}(h_T(x_i)= y_i)\exp(-\alpha_T)$$

Eli Pinkus
CS 155
Set 3

Since the weighting are normalized to equal 1, we can say that

$$\sum_{i=1}^{N} D_T(i)\mathbf{1}(h_T(x_i) = y_i) = 1 - \sum_{i=1}^{N} D_T(i)\mathbf{1}(h_T(x_i) \neq y_i)$$

we also have $\sum_{i=1}^{N} D_T(i)\mathbf{1}(h_T(x_i) \neq y_i) = \epsilon_T$

So

$$Z_T = \exp(\alpha_T)\sum_{i=1}^{N} D_T(i)\mathbf{1}(h_T(x_i) \neq y_i) + \exp(-\alpha_T)\sum_{i=1}^{N} D_T(i)\mathbf{1}(h_T(x_i) = y_i)$$

$$Z_T = \epsilon_T \cdot \exp \alpha_T + (1 - \epsilon_T)\exp(-\alpha_T)$$

Part F

We differentiate w.r.t. Z_T

$$\frac{dZ_T}{d\alpha_T} = \epsilon_T \cdot \exp \alpha_T - (1 - \epsilon_T)\exp(-\alpha_T) = \epsilon_T - (1 - \epsilon_T)\exp(-2\alpha_T) = 0$$

Rearranging gives:

$$\exp(-2\alpha_T) = \frac{\epsilon_T}{1 - \epsilon_T} \Rightarrow \alpha = \frac{1}{2}\ln\left(\frac{\epsilon_T}{1 - \epsilon_T}\right)$$

Part H

Gradient Boosting

As we increase the number of classifiers used, training loss goes down consistently and fairly continuously. This makes sense because each additional classifier helps the model to more precisely fit the training data. Test loss, on the other hand hits a minimum with around 60 classifiers before it starts to rise. This is because the model begins to overfit as it adds more classifiers and becomes more complex. The test loss curve is also less smooth then training loss. The fluctuation overall is because we are using binary loss as our loss function so a change in the hypothesis has discrete changes to the performance.

Part I

Adaboost had a best testing loss of slightly less than 0.2 whereas gradient boosting had a best testing loss of slightly greater than 0.2 so Adaboost had the best performance.

Eli Pinkus
CS 155
Set 3

Part J

The dataset weights are the largest on the points that are more regularly misclassified whereas the weights are the smallest on the points that are consistently correctly classified.