Homework 2

Shalin Shah

January 27, 2020

1 Splitting Heuristic for Decision Trees

1.a How many mistakes does the best 1-leaf decision tree make over 2^n examples?

The 1-leaf decision tree doesn't take into account anything besides the most popular value, which in this case will be 1. So, the model will guess 1 every time (since there is no split) and obviously be wrong many times. If we have n features, then we have 2^n examples. In the case that $n \geq 4$, we want to find a formula that will tell us how many times we'll guess incorrectly. When $n = 4$, like in the given example, we see that the model guesses incorrectly twice. This is because X_1, X_2, and X_3 are all 0's, but the decision tree will guess a 1 by default. This is $\frac{2}{16} = \frac{1}{8}$. For each extra variable we add, there will be double the cases that fail and double the values. The total number of incorrect guess will thus be 2^{n-3}.

1.b Is there a split that reduces number of mistakes by at least one?

No there isn't. If we're restricting our attention to splits that only consider a single attribute, we can consider solely X_1 as an example. If we make a tree out of that, we'll see that the path we would take is either left or right of X_1, but after that we're left with the same (in fact, worse) outcome, due to the fact that we take X_1's outcome and take that as the outcome of the rest of the values. So, we guess 0 half the time and 1 half the time. Half the times would be right, by default, but times where X_1 is 0 will only correspond to the others actually being 0 much less than the actual rate. We would guess incorrectly $3 * 2^{n-3}$ times, because a quarter of the guesses would be correct. If we choose one of the non-OR'd boolean features, we would see that the same problem from the last question would come up, where we're basically guessing at random. Most of the cases are 1's, but we only guess 1's half the times. In essence, we're making the guessing worse by splitting either way.

1.c Entropy for 1-leaf decision tree?

$\sum_k P(X = a_k) * logP(X = a_k) = P(X = 0)logP(X = 0) + P(X = 1)logP(X = 1) =$
$\frac{7}{8}log_2\frac{7}{8} + \frac{1}{8}log_2\frac{1}{8} = \mathbf{0.544}$

1.d Can we improve entropy?

While the amount of incorrect values will stay the same or increase if we branch X_1, the entropy may increase. Let's look at the logic for this. If we split up X_1 into 1 and 0 based on the values of X_1 , every time that X_1 is 1, we will automatically have a 1, and so the entropy for this case is $-(P(X = 1)logP(X = 1) + P(X = 0)logP(X = 0)) = 1 * 0 + 0 * (-\infty) = 0$. So, the 1 case has 0 entropy.

The 0 case is more complicated. We first have to find an explicit formula for the randomness that was described earlier. This represents half of the cases, since half is where $X_1 = 0$. So, our probability will be the probability of 0, divided by 2^{n-1}. From the last part, $\frac{3*2^{n-3}}{2^{n-1}} = \frac{3}{4}$. This is the probability of 1, $P(X = 1)$. $P(X = 0) = 1 - P(X = 1) = \frac{1}{4}$.
So, the entropy is $-(.75log.75 + .25log.25) = .811$. If we calculate the total conditional entropy H$[Y|X]$ (from the slidedeck), then we can calculate the entropy as follows: $.5 * 0 + .5 * .811 \approx .406$.

This is less than the entropy calcluated in part (c), so **Yes**, this split reduces the entropy of the output by a non-zero amount.

2 Entropy and Information

2.a Show that $0 \leq H(S) \leq 1$ and $H(S) = 1$ when $p = n$

$H(S) = B(\frac{p}{p+n}) = -\frac{p}{p+n}log_2\frac{p}{p+n} - (1 - \frac{p}{p+n})log_2(1 - \frac{p}{p+n}) =$
$-\frac{p}{p+n}log_2\frac{p}{p+n} - \frac{n}{p+n}log_2\frac{n}{p+n}$
If $p = n$, then $-\frac{p}{p+n}log_2\frac{p}{p+n} - \frac{n}{p+n}log_2\frac{n}{p+n} = -\frac{p}{p+p}log_2\frac{p}{p+p} - \frac{p}{p+p}log_2\frac{p}{p+p} =$
$-\frac{1}{2}log_2\frac{1}{2} - \frac{1}{2}log_2\frac{1}{2} = -\frac{1}{2}(-1) - \frac{1}{2}(-1) = \mathbf{1}\checkmark$
Else:
$-\frac{p}{p+n}log_2\frac{p}{p+n} - \frac{n}{p+n}log_2\frac{n}{p+n} = -\frac{1}{p+n}[plog_2p - plog_2(p+n) + nlog_2n - nlog_2(p+n)] = -[\frac{p}{p+n}log_2\frac{p}{p+n} + \frac{n}{p+n}log_2\frac{n}{p+n}] =$.
We can see from here that as $n \to 0$ the equation tends to 0, but as $n \to p$ the equations tends to 1. In any case in between, there will always be one of the two data points either equal, tending to 0, or with a little separation, and mathematically the sum will always be within that bound. We could also a differentiation definition, in which the global maximum will be at the center point of 0 and 1.

2.b Show that gain is 0.

gain $= H[Y] - H[Y|X]$.
$H[Y] - H[Y|X] = -[\frac{p_k}{p_k+n_k} + log_2\frac{p_k}{p_k+n_k} + \frac{n_k}{p_k+n_k} + log_2\frac{n_k}{p_k+n_k}] = B(\frac{p}{p+n}) - B(\frac{p}{p+n})[\frac{\Sigma_{k=1}(p_k+n_k)}{p+n}]$, but it holds that $p_k + n_k = p + n$, so:
$B(\frac{p}{p+n}) - B(\frac{p}{p+n}) * \frac{p+n}{p+n} = B(\frac{p}{p+n}) - B(\frac{p}{p+n}) = 0\checkmark$

3 k-Nearest Neighbor

3.a What value of k minimizes training set error?

To minimize training set error, ideally we'd have it so that there is no bias or noise. Lower values of k will make the model more susceptible to noise, whereas higher values of k will not add much value to our model. A value of 5 for k minimizes the training set error. With k = 5, we have a training set error of 4, which is due to the fact that the circles at the bottom right and the asterices at the top left would be classified as data points that they're not. With a lower rate, however, we would risk compromising the two contiguous 5 object rows, especially in the middle of the contiguous rows. Training set error is not a reasonable estimate of test set error, however, because the model has been fitted to this exact data. So, it knows a lot more about this data and thus may seem like a perfect model but actually isn't. With this value of k, we could very well see many astrecies in a row in the top left corner so this could end up being the wrong model. Since we don't know the test, we really don't know if this model is the best for the data or not.

3.b What value of k minimizes leave-one-out cross-validation error?

A value of k = 3 will minimize the error in the leave-one-out cross-validation error case. This is because the model will run 12 times (once for each point as a "test" point) and so when the models are averaged out we will see that there will be no error. Cross-validation is a better measure of test set performance because it includes multiple iterations and chooses which models best fit the data that is there, and it also goes through all of the data eventually. So, we know that the model that is chosen cross-validation is much better suited for the data since there are multiple cross checks of the validity of the model.

3.c LOOCV errors for lowest and highest k for data set?

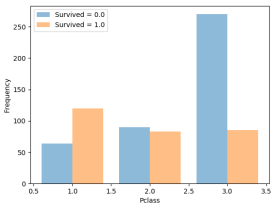
If we use k = 1, for example, then the error will be 10, because 10 of the data points will choose their neighbor of the other classification and thus lose out on their actual value. If we use k = 11, then the error will be 12, because every single point will think that the majority vote of its closest 11 pairs (which are all of the rest of the data points) represents that point, but it won't because there are currently an even number of circles and astrices, and so removing one for the LOOCV will make the balance uneven and the object will think it should be classified as the other object. These both are good examples of why using a large or small value of k would be bad; if we choose a value that's too big then we don't really gain anything from the model, it's computationally heavy, and also could be flat out wrong (like in this case) and if we choose a value that's too small then the model is susceptible to much more bias and noise than we would

like it to be because each individual point next to it has much more power than it would if more points (with a larger k) were there for the model.

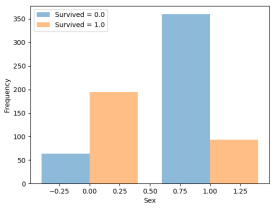
4 Programming Exercise : Applying decision trees

4.a Visualization

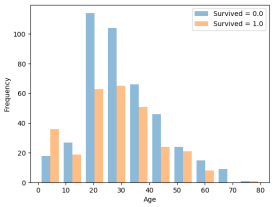
These are the plots from part (a):



Here, we see that if we look at class status, that a greater percentage and raw number of 1st class riders based on socio-economic status (from the kaggle website) survived the crash. Middle class and lower class riders had about the same number of people survive, but percentage wise, more lower class riders didn't survive as compared to upper and middle class.

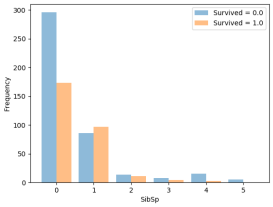


Here we see that a higher percentage and number of individuals of sex 0 survived. We see that frequency wise there were about 350 individuals of sex 1 that didn't survived as opposed to 75 that did while there were 175 individuals of sex 0 that did survive as opposed to 60 that didn't. There were way more of individuals of sex 1 onboard, but way less of them survived.

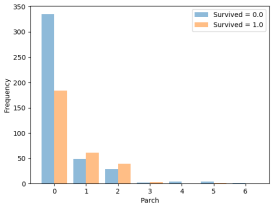


Here we see that there were mostly 20-40 year olds on the ship, but percentage wise they had the highest mortality rate. Out of the few individuals under

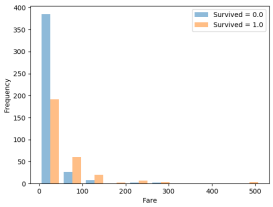
10 years of age, over $\frac{2}{3}$ of them survived. There weren't many individuals of very old age, but of them only 1 of the 2 survived.



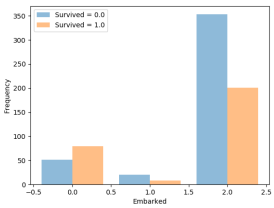
Here we see the number of individuals that had sibilngs or spouses on the ship and their mortality rate. The data doesn't show much, but it seems that there were many bachelors and bachelors had the highest mortality rate.



Here we see the number of individuals that had parents or children on the ship, and again this data doesn't show much but people without children or parents had the highest number of people and about $\frac{2}{3}$ died.



Here we see how the fare of the passengers affected mortality rate. Again this data doesn't show much but people that paid the lowest fares had the highest number of people and about $\frac{2}{3}$ died.



Here we see the port of embarkation and see that most people embarked from port 2 and about $\frac{2}{3}$ of them died.

4.b Evaluation

After implementing and training RandomClassifier, I got an error of 0.485.

4.c

After calling and running DecisionTreeClassifier, I got an error of 0.014.

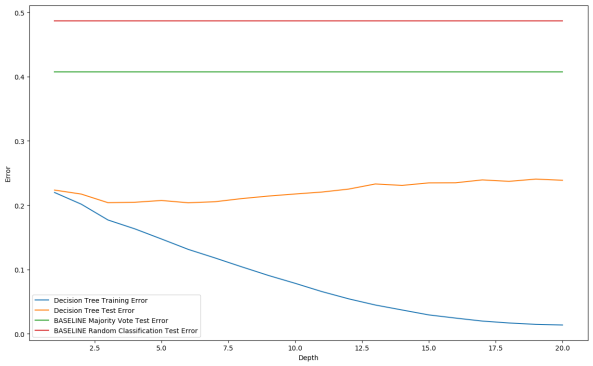
4.d

After implementing error() and modifying the main, these are the values that I got for training and test error for majority vote, random classifier, and decision tree:

```
Investigating various classifiers...
        -- majority vote training error: 0.404
        -- majority vote test error: 0.407
        -- random classifier training error: 0.489
        -- random classifier test error: 0.487
        -- decision tree training error: 0.012
        -- decision tree test error: 0.241
```

4.e

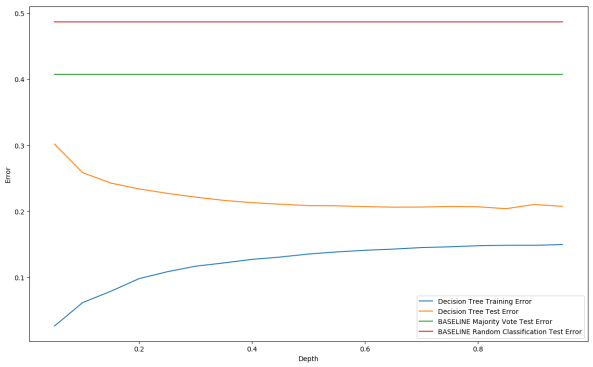
The plot, with all of the axes labeled, can be seen here:



It seems the best depth to use is 5. The reason I say this is that the two graphs sort of deviate from one another, but are still close enough to that ideal distance from one another that was described in class. There is definitely overfitting in this model, and we can see that since the two graphs are very far in distance towards the end of the graph, towards the depth of 20.

4.f

The plot with all of the axes labeled, can be seen here:



Here, we can see that training and test error are come closer as the depth increases. When there's less depth, there is more error.