Problem Set 1

Problem 1: Splitting Heuristic for Decision Trees

(a) The best case situation is when we just guess Y=1. There are 16 options; 14 are Y=1 and 2 are Y=0. Thus, the mistakes are 2 when Y=1.

(b) If we consider only one attribute at a time, there is no split that would lessen the number of mistakes. Because each of the attributes happen half the time, all of the attributes would split the 14 of our already correct Y=1 decision by half; so worsening our mistakes.

(c) Our entropy is 0.544. Our current split is 14 Y=1 labels and 2 Y=0 labels. So, plugging into

$$-log_2\frac{14}{16} * \frac{14}{16} - log_2\frac{2}{16} * \frac{2}{16}$$

gives us 0.544.

(d) The split that would reduce the entropy would be splitting by X3. If we split by X3 we would have X3=1 would have 8 datapoints that all have the label Y=1 and X3=0 would have 6 datapoints that would be Y=1 and 2 datapoints that would be Y=0. Calculating conditional entropy would be

$$\frac{1}{2}(-log_2\frac{6}{8} * \frac{6}{8} - log_2\frac{2}{8} * \frac{2}{8}) + \frac{1}{2}(-log_2\frac{8}{8} * \frac{8}{8} - log_2\frac{0}{8} * \frac{0}{8})$$

which gives us a conditional entropy 0.406 giving us a better result than the one leaf split proving that splitting by entropy is much more informative than splitting by correctness (or reducing error).

Problem 2: Entropy and Information

(a) When p=n, this tells us that the number of positive examples is exactly equal to the number of negative examples which tells us that each type of example is exactly half of the total number of examples. So, here p contributes to 50% of p+n as well as n. Caluclating H(S) when p=n gives us

$$H(S) = (-log_2\tfrac{1}{2} * \tfrac{1}{2} - log_2\tfrac{1}{2} * \tfrac{1}{2}) = 1$$

To prove that $0 \leq H(S) \leq 1$, we need to show that the 100/0 split gives us a result of 0. Since we already proved that the 50/50 split gives us a maximum of 1, we need to show that the opposite gives us an answer of 0 to complete the premise. So, we show that

$$H(S) = (-log_2\tfrac{2}{2} * \tfrac{2}{2} - log_2\tfrac{0}{2} * \tfrac{0}{2}) = 0$$

which completes our premise.

(b) To prove that information gain is equal to 0, we need to prove that the entropy minus the conditional entropy is equal to 0 which is saying that the entropy before the split has to be equal to the conditional entropy. First, we show the regular entropy given that p is the number of total positive data points and n is the number negative data points

$$(-log_2\tfrac{p}{p+n} * \tfrac{n}{p+n} - \tfrac{n}{p+n} * log_2\tfrac{n}{p+n})$$

has to equal the conditional entropy of

$$\tfrac{p_{\mathrm{k}}+n_{\mathrm{k}}}{p+n}(-log_2\tfrac{p_{\mathrm{k}}}{p_{\mathrm{k}}+n_{\mathrm{k}}} * \tfrac{p_{\mathrm{k}}}{p_{\mathrm{k}}+n_{\mathrm{k}}} - log_2\tfrac{n_{\mathrm{k}}}{p_{\mathrm{k}}+n_{\mathrm{k}}} * \tfrac{n_{\mathrm{k}}}{p_{\mathrm{k}}+n_{\mathrm{k}}}) * k$$

.

Simplifying and distributing the conditional entropy gives us

$$(-log_2\tfrac{p_{\mathrm{k}}*k}{p+n} * \tfrac{p_{\mathrm{k}}*k}{p+n} - log_2\tfrac{n_{\mathrm{k}}*k}{p+n} * \tfrac{n_{\mathrm{k}}*k}{p+n})$$

.

and in this case $\mathrm{p_k}$ and $\mathrm{n_k}$ multiplied by k is equal to the total of p and n respectively. Thus, this being our final simplification we would get an equal entropy and conditional entropy giving 0 information gain by

$$InformationGain =$$
$$(-log_2\tfrac{p}{p+n} * \tfrac{n}{p+n} - \tfrac{n}{p+n} * log_2\tfrac{n}{p+n}) - (-log_2\tfrac{p}{p+n} * \tfrac{n}{p+n} - \tfrac{n}{p+n} * log_2\tfrac{n}{p+n}) = 0$$

.

Problem 3

(a) K = 1 would lead to the minimal training error of 0%. K=1 nearest neighbours means to assign each label to itself, which is why it's very unreasonable. It is over fitting by a high margin, so it is not reasonable to estimate the test error.

(b) K = 7 minimizes the LOOCV error for the data set. In this model, we'll miss classify the two outliers on either side of the dataset. The error would then be 28.5%. Cross validation error is a better measure since we use different test data sets every time.

(c) K = 1: 10/14 or 71.4% error because on either side of the graph there exists two data points that are in the extreme ends that are closest to their own class(2 circles on the top and 2 asterisks on the bottom). All the rest of the data points are actually closest to the opposite class, since there is always 2 in between 3 of the opposite class.

K = 13: 14/14 or 100% error because majority rules. For each time you take a data point out, there will be 7 of the opposite class and 6 of the same class. So, every time you try to classify it is guaranteed you are wrong.

Problem 4

1. A note on the data set is that there a far larger amount of people that earn less than 50k than more than 50k.

   (a) Sex: For the Sex feature, it seems that the data says that Females have a much higher ratio of earning more than 50k. Based on the data, even though Female data in total is higher, Males have a lesser ratio of earning more than 50k.

   (b) Workclass: The biggest ratios where earning less than 50k is significantly higher than earning more than 50k is in three workclass types: Unincorporated self employed, the private sector, incorporated self employed and state workers. All the other possible types are mostly even.

   (c) Education: The trend is pretty obvious here; where the ratios of people that earn more than 50k with bachelors, professional school, masters, doctorate, or any other form of higher education is higher than any person with highschool as their highest education.

   (d) Marital Status: An interesting correlation where we see the divorced ratio is the highest between more than 50k earners, while never married and seperated come after sequentially.

   (e) Occupation: There is a clear trend of three features that the big earners focus in: Sales, Management and Professional Specialty. All other occupations are dominated by low earners.

   (f) Relationship: Both husband and wife relationship status have high trends with the high earners; while others are very common with low earners.

   (g) Race: Asian-Pac-Islander and "Other" races have significant high earnings with respect to low earners. White has a lesser ratio with Black being even lower.

   (h) Native Country: The only countries with high earning showing is USA, Cambodia, England, Puerto Rico and Canada. The other countries are all dominated with low earners which include: China, Iran, Phillipines, Mexico and Portugal.

   (i) Age: We see high earners start off a little in the 30s range, peaking in the 40s and dipping into the 60s while non-existent after.

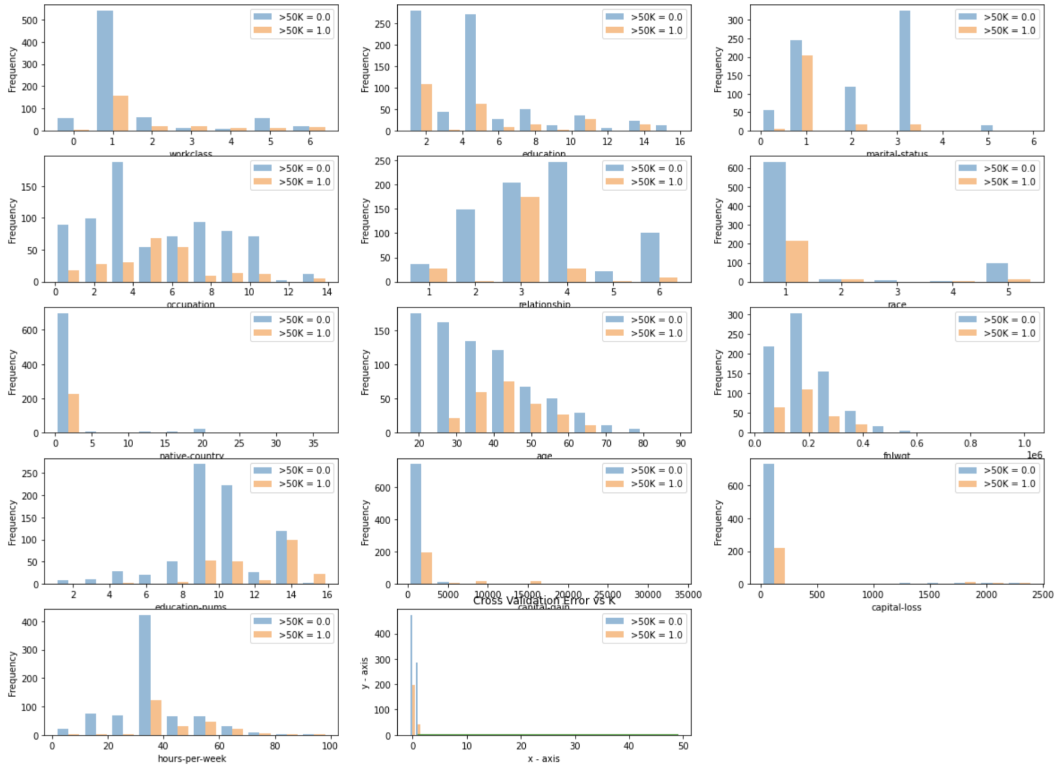
   (j) Fnlwt: Describes the general proportion of the two populations; concluding that the less than 50k earners represent a higher portion.

   (k) Education_nums: Very clear trend with the lesser earners are highly correlated with less number of years in education; and the opposite dominate the higher number of years.

   (l) Capital_gain: Another clear trend with the lesser earners population condensed into the 0-5000 range while the higher earners dominating the 5000+ earnings.

(m) Capital_loss: The complete inverse relation of complete_gain with some exceptions of high earners with high capital loss.

(n) Hours_per_week: The 0-40 hour range is fairly dominated by the lesser earners and the ratios start to even out by the 50 hour mark. After that, it is pretty skewed towards the higher earners.



2. (a) Already solved

   (b) 4.2(b) Got training error of 0.374.

   (c) 4.2(c) For decision tree classifier, I obtained a training error 0. This makes sense because the default implementation of decision tree allows the model to go to as deep with no restrictions, meaning it can classify the data perfectly.

   (d) 4.2(d) For KNN, for k = 3 I had a training error 0.153, for k = 5 I had a training error of 0.193, and for k = 7 I had a training error of 0.213.

   (e) 4.2(e) These were the results per the four models:

      i. MajorityVoteClassifier: Training Error: 0.240, Test Error: 0.240

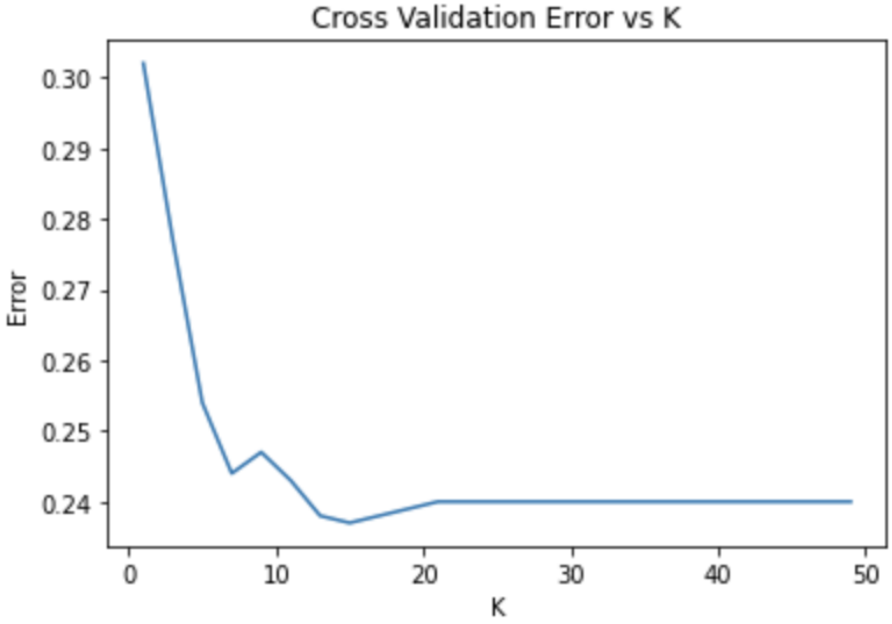
      ii. RandomClassifier: Training Error: 0.375, Test Error: 0.382

      iii. DecisionTreeClassifier: Training Error: 0.000, Test Error: 0.207

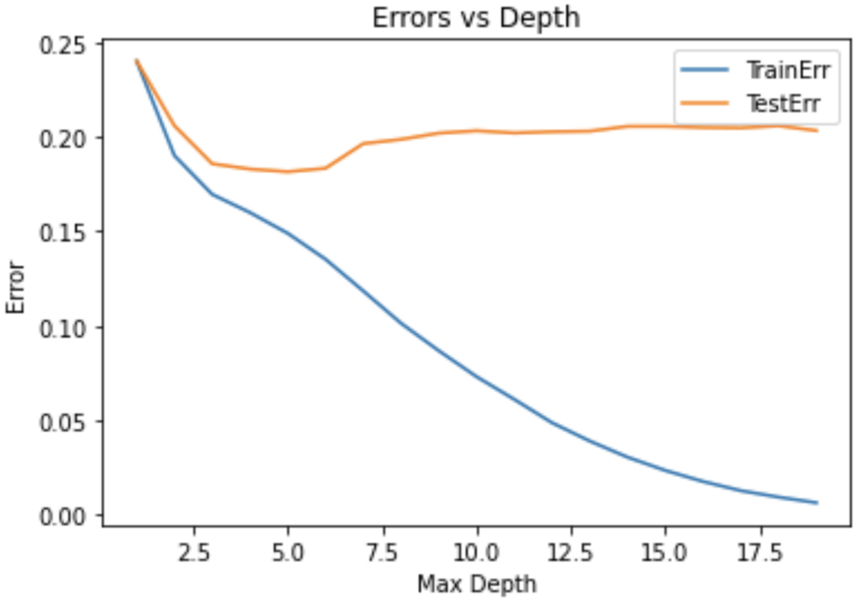
      iv. KNeighborsClassifier(K=5): Training Error: 0.202, Test Error: 0.259

   (f) 4.2(f) I plotted the CV error using the cross_val_score() function embedded in the CV class; while taking the mean of each K number because it returns an array.

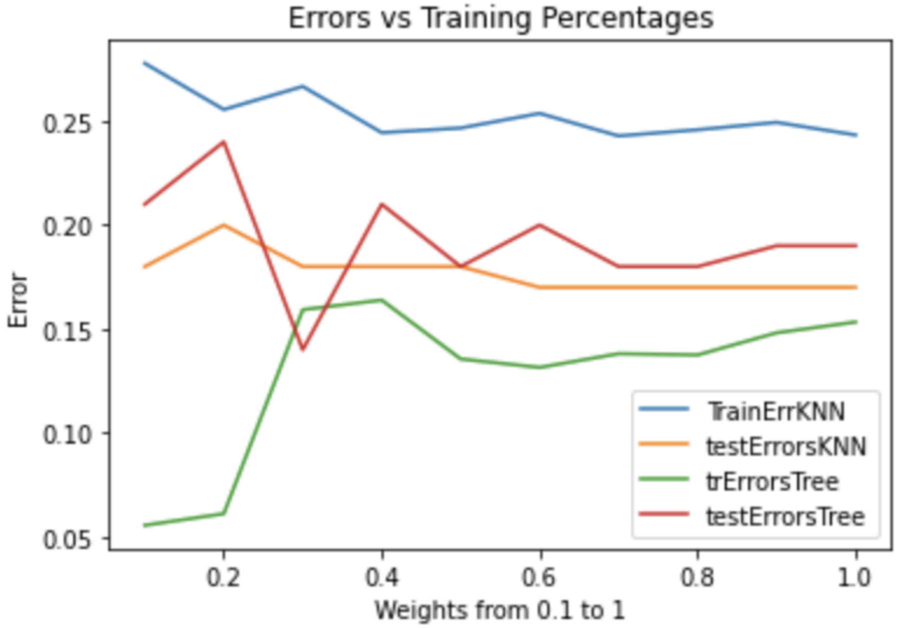
Based on the graph, it seems it under-fits or has a significant training error for the first couple of K numbers and it goes down to K=15 where we have the minimum error.



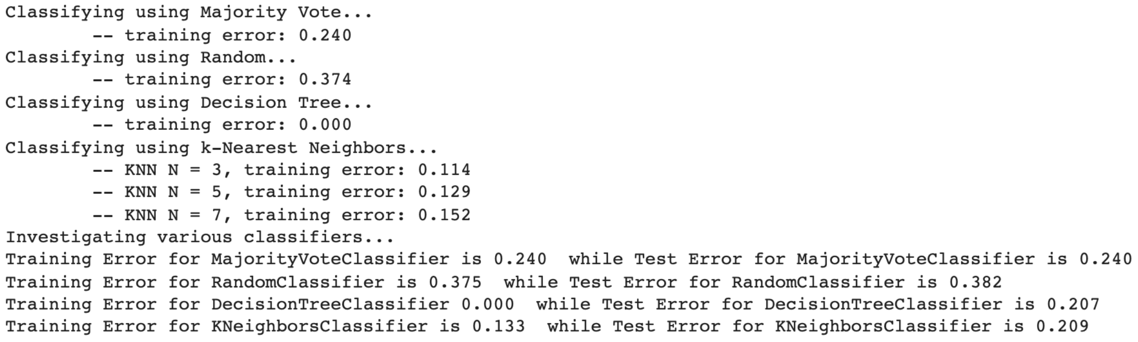
(g) 4.2(g) Looking at the graph below, the training error and testing error initially start severely under fitting for tree depth up to 3. The sweet spot is around tree depth of 5 and it starts over-fitting by a big margin after a depth of 7.



(h) 4.2(h) For splitting the data I used train_test_split() to split the initial 90/10 train test and used it for the second split which has increments of 0.1. Based on the graph, I think the data is a little rougher since it's not stratified and used over ntrials. However, it seems that decision tree is learning a little better as it converges to somewhere around 0.17. KNN seems to converge just with a bigger margin.

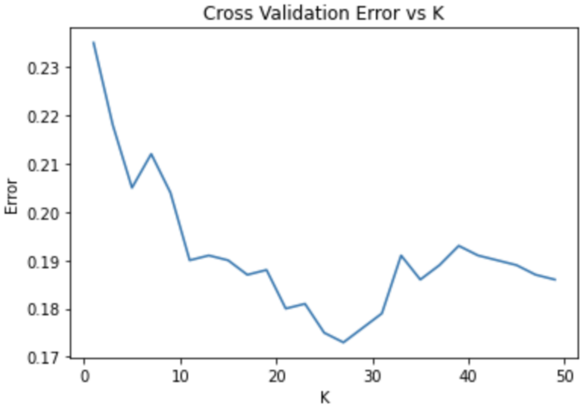


(i) 4.2(i) Standardization: These are the different errors using both the accuracy metric and the the error function that was implemented. We can see that the majority, random classifier and decision tree accuracy errors didn't change that much. The KNN error responded well to the standardization and was decreased. For the error function, it was the same behavior with KNN being the only model increasing in accuracy.

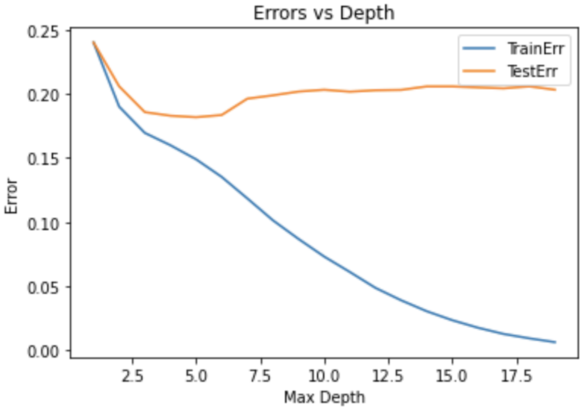


```
Classifying using Majority Vote...
        -- training error: 0.240
Classifying using Random...
        -- training error: 0.374
Classifying using Decision Tree...
        -- training error: 0.000
Classifying using k-Nearest Neighbors...
        -- KNN N = 3, training error: 0.114
        -- KNN N = 5, training error: 0.129
        -- KNN N = 7, training error: 0.152
Investigating various classifiers...
Training Error for MajorityVoteClassifier is 0.240  while Test Error for MajorityVoteClassifier is 0.240
Training Error for RandomClassifier is 0.375  while Test Error for RandomClassifier is 0.382
Training Error for DecisionTreeClassifier 0.000  while Test Error for DecisionTreeClassifier is 0.207
Training Error for KNeighborsClassifier is 0.133  while Test Error for KNeighborsClassifier is 0.209
```

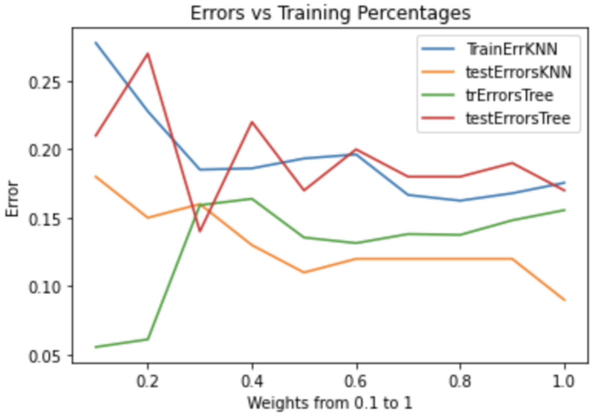
For the hyper parameters we get really intriguing data as the decision tree stayed at 5 depth while the KNN increased to K=26.



```
Investigating depths...
```



The last pieces of data show the different training weights. After adjusting the K=26, it seems that KNN especially converged pretty well comparatively with no standardization, while the Tree model seems unaffected.


Errors vs Training Percentages
TrainErrKNN
testErrorsKNN
trErrorsTree
testErrorsTree
Error
Weights from 0.1 to 1