James Simmons
Machine Learning
Fall 2021

TASK ONE

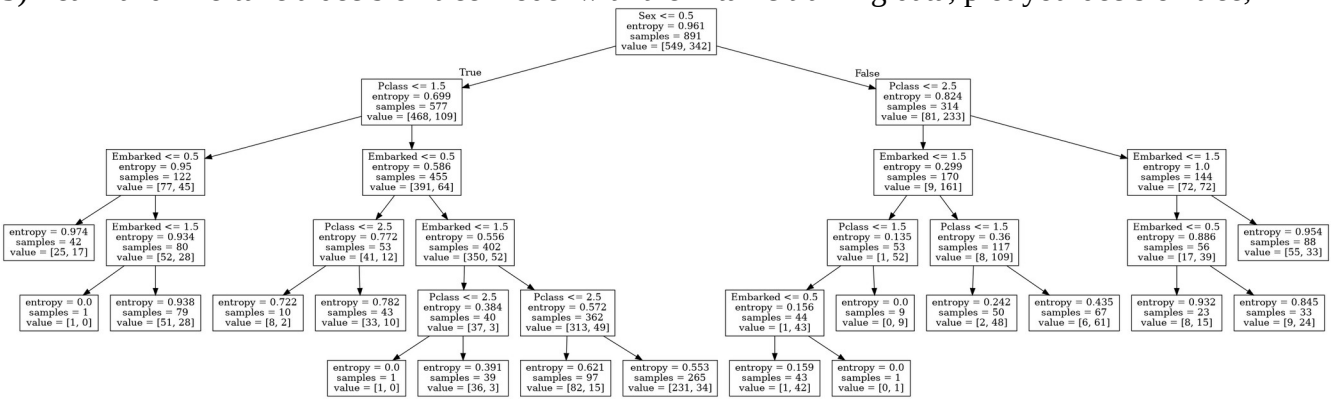
1) Preprocess your Titanic training data;

To preprocess the data, I removed the Name column because each record was a distinct category. I also removed the Cabin column because ¾ of the records were missing, and those that weren't were inconsistent and hard to work with. For Sex and Embarked, I changed them from string-based categories to numerical categores by setting male to 0 and female to 1, and by setting S, Q, and C to 0, 1, and 2, respectively. I resolved the missing Age records (about 1/6 of them) by setting them equal to the average age. Finally, since most of the elements in the Ticket category just had a number with no leading info, and the ones that did have leading info had very few categories, I stripped off everything but the number, and cast it from string to int. By the end, every column was an int or float, with missing values.

2) Select a set of important features. Please show your selected features and explain how you perform feature selection.

I expected that Sex and Pclass would be the most impactful features simply based on what I know about the evacuation procedures on sinking ships, but I wasn't satisfied with settling for this. I wrote a script to take the powerset of the features and try every possible combination, generating a tree for each, and sorting them by their average k-fold-cross-validation accuracy. This resulted in an ideal feature selection of Sex, Pclass, and Embarked. That said, the 5th best selection was simply Sex by itself, and the accuracy was less than 1% worse than the ideal; if a simpler tree is desired, simply splitting on Sex is nearly as good as the complex ideal tree. I also ran it with ALL features, just to see what it would be. Interestingly, it wasn't deterministic like the others, but instead had a range of different values that would appear.

3) Learn and fine-tune a decision tree model with the Titanic training data, plot yourdecision tree;



4) Applythe five-fold cross validation of yourfine-tuneddecision treelearning modelto the Titanic training datato extract average classification accuracy;

Average cross-validation accuracy for Sex, Pclass, Embarked: 0.7935483870967742
Average cross-validation accuracy for Sex alone: 0.7868162692847125
Average cross-validation accuracy for ALL features: 0.74 – 0.75

5) Applythe five-fold cross validation of your fine-tuned random forest learning modeltothe Titanic training data to extract average classification accuracy;

I did the "best of all possible combinations" for the random tree as well (it took much longer), the the result was unsurprisingly the same. I also computed the accuracy of just using ALL the features with random forest, and the results were nearly identical. Since the random forest is random, the accuracies are slightly different every time, so they are presented below as ranges.

Average cross-validation accuracy for Sex, Pclass, Embarked: 0.78 – 0.81
Average cross-validation accuracy for ALL features: 0.78 – 0.79

6) Which algorithm is better, Decision Tree or Random Forest?

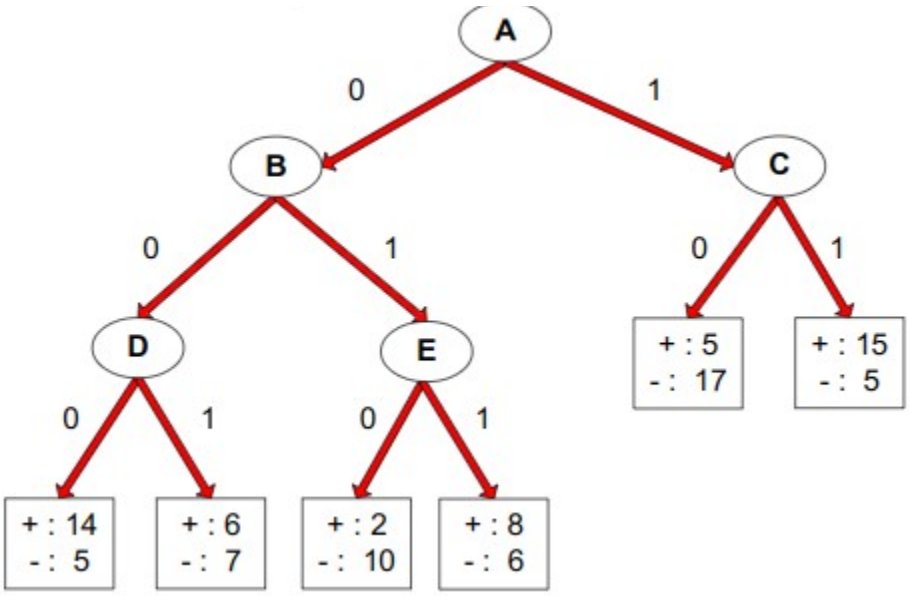
Random Forest appears to be better. Often, it is prohibitively difficult to precicely calculate the best feature combination. This is because a dataset with n features has 2^n feature combinations, which is calculable in this case when n is only 8, but quickly becomes incalcuable as n gets bigger. Random Forest had an "All Features" accuracy very close to Decision Tree's ideal accuracy, but was much easier and faster to compute.

7) What are your observations and conclusions from the algorithm comparison and analysis?

Bagging seeks to increase accuracy and resist overfitting compared to Descision Trees by using an ensemble process, but intoduces a major problem of its own: the forest it produces don't have enough variation, which defeats the point of ensemble learning. Random Forest resolves this shortcomming, resulting in an algorithm that is nearly as good as a perfectly optimized Descision Tree, but with considerably less computational effort required.

TASK TWO

Consider the descision tree below:



(a) What is the training error rate for the tree? Explain how you get the answer?

Using entropy, we calculate the entropy for each leaf. We then combine leafs based on the formula M = sum over i {n_i * M_i / n). We do this to calculate the entropies for nodes D, E, and C, then combine D and E to form B, then combine B and C to form A, which is the overall entropy of the tree. Actually doing this calculation gives an entropy of 0.836.

The error rate of the tree is the percent of total instances that are classified incorrectly based on the tree. The is equal to the total number of instances that are in the minority of their respective leaves, divided by the total number of training instances overall. This is 29 / 100 = 0.29 or 29% error.

(b) Given a test instance T={A=0, B=1, C=1, D=1, E=0}, what class would the decision tree above assign to T? Explain how you get the answer?

Starting at the root of the tree (node A), we see that feature A of instance T is 0, so we go to the left, which takes us to node B. Feature B of instance T is 1, so we go right to node E. Feature E is 0, so we go left, bringing us to a leaf with class distribution {+: 2, -: 10}. Since the majority of instances in this leaf are of the - class, the – class is assigned to instance T.

TASK THREE

Consider the following data set for a binary class problem.

Q1: What is the overall entropy before splitting?

- [4/10 log(4/10) + 6/10 log(6/10) = 0.971

Q2: What is the gain in entropy after splitting on A?

New Entropy: - [4/7 log (4/7) + 3/7 log (3/7)] - [0] = 0.690
Gain = Overall – New = 0.971 – 0.690 = 0.281

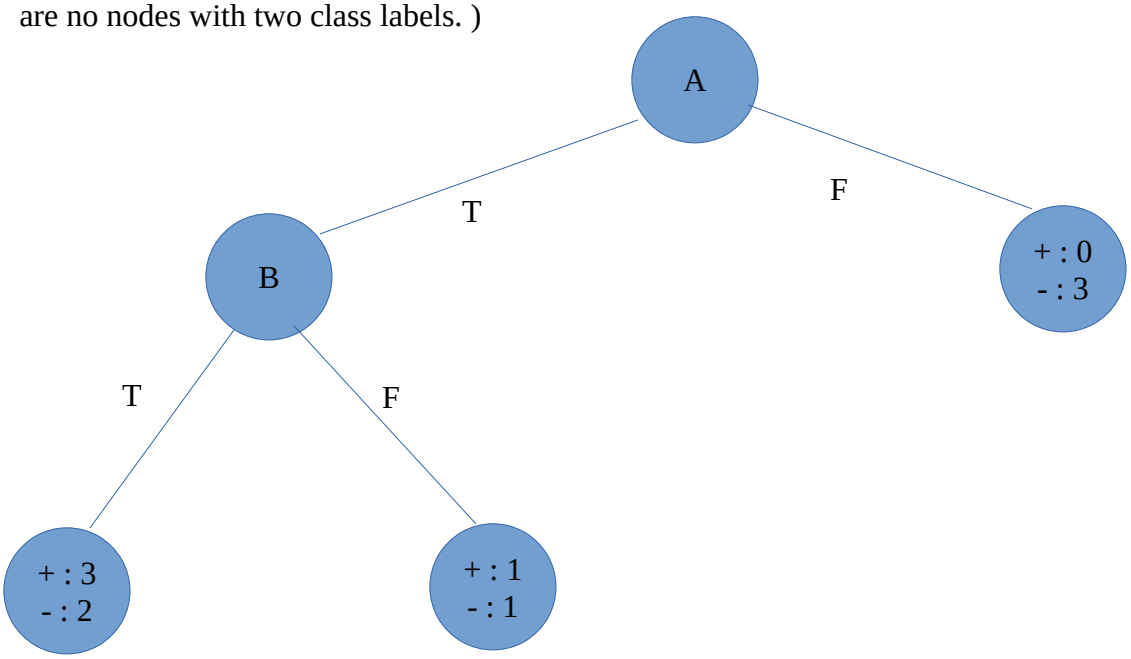
Q3: What is the gain in entropy after splitting on B?

New Entropy: - [3/5 log (3/5) + 2/5 log (2/5)] - [1/5 log (1/5) + 4/5 log (4/5)] = 0.846
Gain = Overall – New = 0.971 – 0.846 = 0.125

Q4: Which attribute would the decision tree choose?

0.281 > 0.125, therefore the decision tree would choose attribute A to maximise entropy gain.

Q5: Draw the full decision tree that would be learned for this dataset. You do not need to show any calculations. (We want to split first on the variable which maximizes the information gain until there are no nodes with two class labels.)

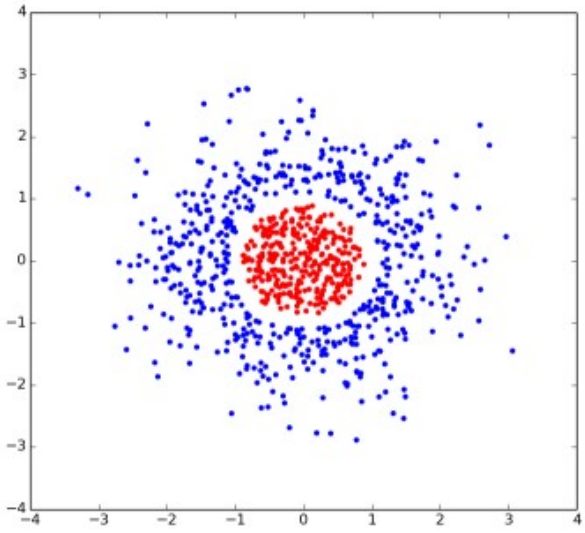


TASK FOUR

Q1: Are decision trees a linear classifier?

Decision trees are nonlinear. Consider the plot below. A linear classifier would not be able to separate the red and blue classes. A decision tree, however, could come up with the following rules:

y > 1: Blue
y < -1: Blue
-1 < y < 1:
x > 1: Blue
x < -1: Blue
-1 < x < 1: Red



Q2: What are the weaknesses of decision trees?

Decision trees are highly prone to overfitting. This is because, by default, a decision tree will continue to split until all leaves are pure. This can be mitigated somewhat by setting constraints on the tree, such as the mininum number of leaf nodes, or a minimum amount of information gain for a split. That said, it is often better to just use a different approach, such as random forest.

Q3: Is Misclassification errors better than Gini index as the splitting criteria for decision trees?

No. For descision trees, the best splitting criteria are Entropy, Gini Index, and Misclassification Error, in that order. The factors to consider in this determination are sensitivity (slope when probability is close to 0 or 1) and range (the difference between the highest and lowest values attainable). Gini and Misclassification Error both have the same range (0.5), but Gini is more sensitive.

CODE

All code available at https://github.com/Mesaj2000/Machine_Learning_5610/tree/master/hw2