CS 349 Homework 1 Write-Up [CHAT GPT HELPED W/ DOCUMENTATION – prof said OK]

Coel Morcott and Vatsal Bhargava

1. Did you alter the Node data structure? If so, how and why?

We added splitting_attribute. This is initialized as None but will eventually be used to store the value of the node that the node will split on. An example would be Cuisine – this attribute would be stored in Node.splitting_attribute, then when predicting a value the tree would split on Cuisine into 4 paths for each cuisine.

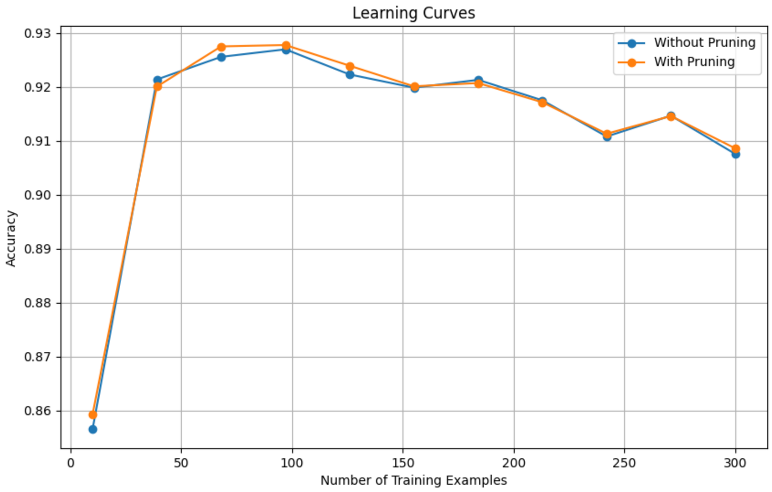
2. How did you handle missing attributes, and why did you choose this strategy?

We know from the test data that missing attributes are when the data has a "?" assigned to that attribute. To handle this, we have an if statement that checks for this, and does not include that attributes outcome to our dictionary when creating our data structures to calculate entropy. We went for this strategy based off the pseudo code and the fact that very few cases are like this, so overall it does not affect our entropy calculation much. This occurs in the find_best_attribute function where we basically ignore that example.

3. How did you perform Pruning, and why did you choose this strategy?

We chose the reduced error pruning strategy, based on advice from lecture (we were told it is the best) and the fact that we could implement this recursively without much trouble. By taking this bottom up recursive strategy, we can check which nonleaf nodes effectively decrease our error, and if it does, include it, if not, take it out to avoid overfitting.

4.



a. What is the general trend of both lines as training set size increases, and why does this make sense?

Both lines increase rapidly up until about 100 trials because as more information comes, the tree will start to become more and more accurate. After this point, both of them begin to decline steadily through 300 examples. We believe this occurs because of the small number of attributes in our examples, ID3 is known to overfit as it is, and with only 5 attributes, the number of examples can begin to overwhelm the tree and cause overfitting with all the noise.

Also, given our original ID3 algorithm already having a very high accuracy rate – increasing accuracy past 91% already is very difficult and even seeing a 2% increase made sense for us.

b. How does the advantage of pruning change as the dataset size increases? Does this make sense, and why or why not?

Regarding our specific scenario – pruning did not play much of a factor at all. For the small datasets, it was about equal until it surpassed the tree without pruning for the dataset size 60-150 then, after that, it was about the same as the tree without pruning. In general, however, as the dataset size increases, the value of pruning should shine through more because of the removal of overfitting. In our situation, though, the numbers do make sense. The prime range for the number of trials was around 100, which is when the pruned tree outperformed the original tree by the most, then decreased as the number of trials went up and overfitting began to set in.

5. (optional 2.0 points) Use your ID3 code to construct a Random Forest classifier using the candy.data dataset. You can construct any number of random trees using methods of your choosing. Justify your design choices and compare results to a single decision tree constructed using the ID3 algorithm.

We created a random forest in the file random_foreset.py. Here are our results:

Random forest accuracy: 0.7216
Single tree accuracy: 0.685

In constructing the Random Forest classifier decided to build multiple decision trees, each on a random subset of the training data, a practice known as bootstrap sampling, which aims to bring in diversity among the trees, thereby making the ensemble model more robust and less prone to overfitting. We selected 20 trees for our forest, which is a moderate choice balancing between computational efficiency and the model's performance. For classification, we used a majority voting strategy, the mode to aggregate predictions from individual trees to arrive at a final decision. This strategy often helps in reducing the variance of predictions, leading to a more accurate and stable model.

The results indicate a better performance from the Random Forest with an accuracy of 0.7216 compared to the single tree's accuracy of 0.685, over 50 testing examples. The improved performance of the Random Forest can be attributed to the collective learning from multiple trees, which tends to generalize better on unseen data, as opposed to a single decision tree which might capture noise and overfit to the training data.