Aditya Kumar
Netid: AKE9173

Problem Set 1

1. Which other student, if any, is in your group?
Answer: No one

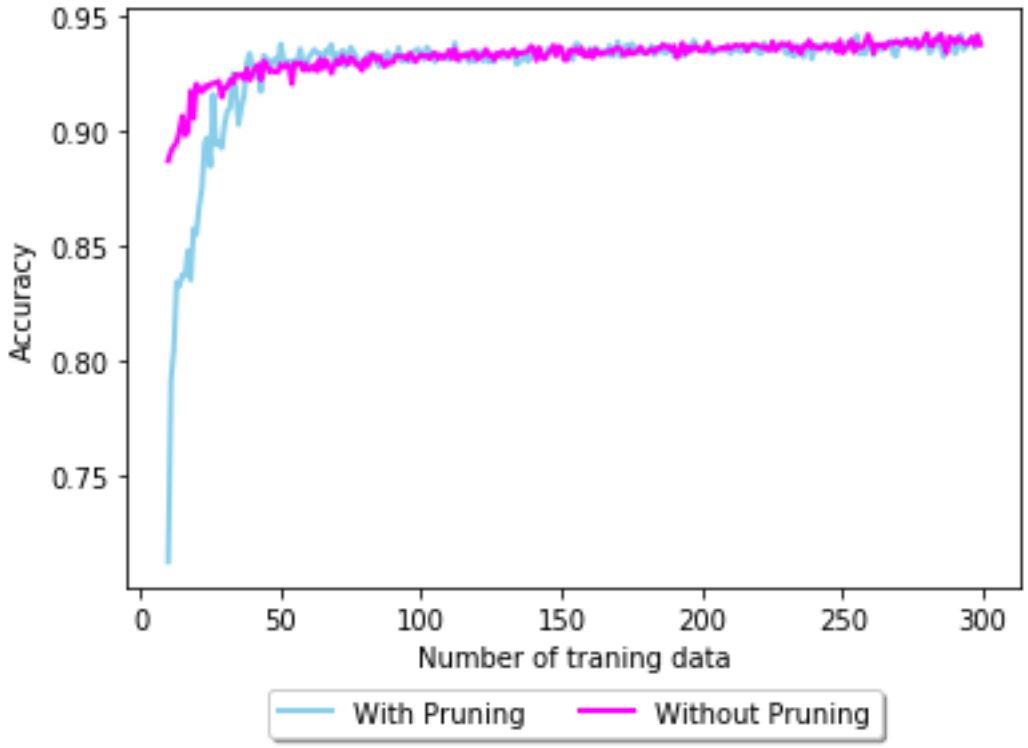
2. Did you alter the Node data structure? If so, how and why?
Answer: Yes, I altered the Node Data Structure. I added 3 more attribute to Node data structure. Those attribute are explained below:
a) isLeaf (Boolean) : To identify if the Node is a leaf node or not.
b) default (String) : To set the default value at each node in the case node path is not found then default value can be passed.
c) numberOfTimeTraversed (int) : To keep the count of which node is traversed how many times while training the tree so that I can use this count to prune my tree at later point of time.

3. How did you handle missing attributes, and why did you choose this strategy?
Answer: I handled missing attribute (?) as a child path for their respective node. I have choose this strategy to make my decision tree more flexible so that later these missing attributes can be changed according to training data. For example if one training data says that the missing attribute is say "0" and other training data can predict it as "1".

4. How did you perform pruning, and why did you choose this strategy?
Answer: I tried to perform pruning by using concept of *reduced error pruning* algorithm. I train my tree using the train data and keep the count of each node that has been traversed while executing that train data on my tree including leaf nodes. Then I used below algorithm to perform pruning:
Step 1: Use breadth first search to collect all nodes of the tree (excluding leaf nodes) in a list
Step 2: Traverse each node from list (from step 1), if all their nodes are leaf nodes then collect the number of times that leaf node is traversed (using the variable *numberOfTimeTraversed*).
Step 3: Sum all the count of traversal of each node (if their entire sub child is a leaf nodes).
Step 4: The leaf node with highest count is the famous child for that node.
Step 5: Take the backup of the node and replace the existing node with the famous leaf node.
Step 6: Measure the accuracy.
Step 7: If accuracy increased then continue from Step 1, else reverts the node with the original with the backup.

I choose this strategy because of the following reason:
1. This is bottom up approach and best way to replace a node using their famous child.
2. This is faster approach to prune a tree efficiently.

5. Now you will try your learner on the house_votes_84.data, and plot learning curves. Specifically, you should experiment under two settings: with pruning, and without pruning. Use training set sizes ranging between 10 and 300 examples. For each training size you choose, perform 100 random runs, for each run testing on all examples not used for training (see testPruningOnHouseData from unit_tests.py for one example of this). Plot the average accuracy of the 100 runs as one point on a learning curve (x-axis = number of training examples, y-axis = accuracy on test data). Connect the points to show one line representing accuracy *with* pruning, the other *without*. Include your plot in your pdf, and answer two questions:



a. In about a sentence, what is the general trend of both lines as training set size increases, and why does this make sense?
Answer: It is clear from the graph that the general trend of both lines is moving to the accuracy of 100% as the training data set increases. It make sense because as we give more data to our tree it will train themselves more accurately and make more authentic tree with less probability of error.

b. In about two sentences, how does the advantage of pruning change as the data set size increases? Does this make sense, and why or why not?
Answer: Advantage of pruning change as the data set size increases because the tree train themselves on that huge data set and prune themselves with only mostly used nodes and leafs. If any node's leafs are more common in the trained data then that node can be replaced with that common leaf.

In my opinion it makes sense as the node which has not been traversed or rarely traversed are not going to affect the accuracy of the tree if that node is pruned with their famous leaf on a huge data set.