1. **Which other student, if any, is in your group?**

**Answer:** No one

1. **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:

1. isLeaf (Boolean) : To identify if the Node is a leaf node or not.
2. default (String) : To set the default value at each node in the case node path is not found then default value can be passed. It can be removed and this default value can be assigned globally.
3. 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.
4. **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”.

1. **How did you perform pruning, and why did you choose this strategy?**

**Answer:** I tried to perform pruning by using some 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. Then I used below algorithm to perform pruning:

1. Initialize pruning\_factor = 0.1
2. Traverse the tree and update all nodes with the number of time it has been traversed (Using node attribute “*numberOfTimeTraversed”*).
3. Check for the node which has been traversed count less than pruning\_factor (total number of count of current node/total number of count of root node)
4. Checked the accuracy of pruned tree.
5. If accuracy < actual accuracy of the tree

break

6. Else

Increase the pruning\_factor by 0.1 (now it is 0.2)

start again from step 3

1. 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:
   1. In about a sentence, what is the general trend of both lines as training set size increases, and why does this make sense?

Answer:

* 1. 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?