1. (0.5 points) Which other student, if any, is in your group? (either names or netIDs is fine)

Matt Engelken (mpe291) & Ryan Rawitscher (rar612)

2. (0.5 points) Did you alter the Node data structure? If so, how and why? (2 sentences)

We added an attribute array, and a classification integer. The attribute array is a list of the attribute values that pertain to the examples passed into ID3 that of attribute type *Node.Label,* and a classification integer to record a classification mode at any point in our tree and a class for every leaf in our tree if a test falls to a leaf.

3. (1 point) How did you handle missing attributes, and why did you choose this strategy? (2 sentences)

To handle missing attributes we created a mode field on each node so that if we get to a split on an attribute that is missing our ID3 method returns the most common class among the examples that were used to construct that tree up to that point. Only the examples that were relevant to the attribute splits leading up to that split with the missing attribute play into the mode calculation, which returns the most common class of examples to try to guess the correct class.

4. (1 point) How did you perform pruning, and why did you choose this strategy? (4 sentences)

   We performed pruning by starting at the leaves and replacing each of the parent nodes with the most common classification of its leaves. If this new tree had a higher accuracy than the original tree, we kept the changes. We then ran the pruning method on the new tree to repeat the process. We stop when we get a new tree whose accuracy is not better than the current tree.

5. (2 points) 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?

When we ran ID3 on our training set for each size of training set+validation set, we were surprised to see that our accuracy for the unpruned tree stayed at a high constant value of around .93., though we expected to see the accuracy of without pruning tree to increase as the training set increased. We did see that our pruned tree increased in accuracy as we increased the size of the training and validation set, a trend we expected to see. We were looking to find a

point when the pruned tree became better than the unpruned tree for a set of training examples but unpruned tree stayed above pruned tree for every training set value for our ID3 method.

2. 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?

As the data set size increases, pruning becomes more effective. This is because the tree becomes too complicated due to overfitting from being built using so many examples. Pruning simplifies the tree to a more general form that utilizes less but more impactful splits to increase overall accuracy.

6. *Note: depending on your particular approach, pruning may not improve accuracy consistently or may decrease it. You can still receive full credit for this as long as your approach is reasonable and correctly implemented.*