Allen Madsen
Project 3: Decision Tree

Implementation

My program is broken up into four modules. The first is data, which is tasked with reading in the csv files and splitting the data randomly into a test and training set. The second is tree, which has three classes: Builder, Node, and Leaf. Builder is the implementation of the decision tree algorithm in the book. Builder creates nodes for each branch in the decision making process and leafs for final decisions. The third, the chi module is responsible for deciding if the split produces significant results when branching in the builder module. The final module is analyze, which handles the building of the tree's with different sample sizes and testing the results of the classification against the known values. It also, calculates the averages of the total correct and total nodes across ten random samples for each sample size.

Discussion

I believe the trees have over fitting. For the Car data set in Figure 1 as the number of training instances increase, the number of correctly classified instances increase to a point and then plateau. However, in Figure 2 as the number of training instances in crease the number of nodes used increases almost linearly. What this is saying is that after about 75 training examples, the tree which uses about 39 nodes is as general as seeing 864 training examples and using about 412 nodes for the car data set. This means that there is likely some pruning that could occur to remove the unnecessary nodes between the two trees with 39 nodes and 412 nods.

For the Tic-Tac-Toe data set the results seem misleading. The trees generated without pruning seem to follow the same pattern as the Car data set. However, when pruning is employed, it does not improve the classification (Figure 3). It does, however, drastically decrease the amount of nodes necessary to get a classification rate near what the non-pruned version does. Generally, pruning should improve the classification rate. A possible reason for the poorer performance might be with the entire data set. Since it is a subset of all possible end games in tic-tac-toe, it is possible that the subset is skewed towards particular end games and outcomes. If that is the case, it would be possible to generate a tree with over fitting because the data set is structured in such a way that no matter how the data is split an over fitted tree will still be a good classifier for the test set. However, when applied to the full set of all possible end games and outcomes, it is likely the pruned tree will perform better than the non-pruned tree.

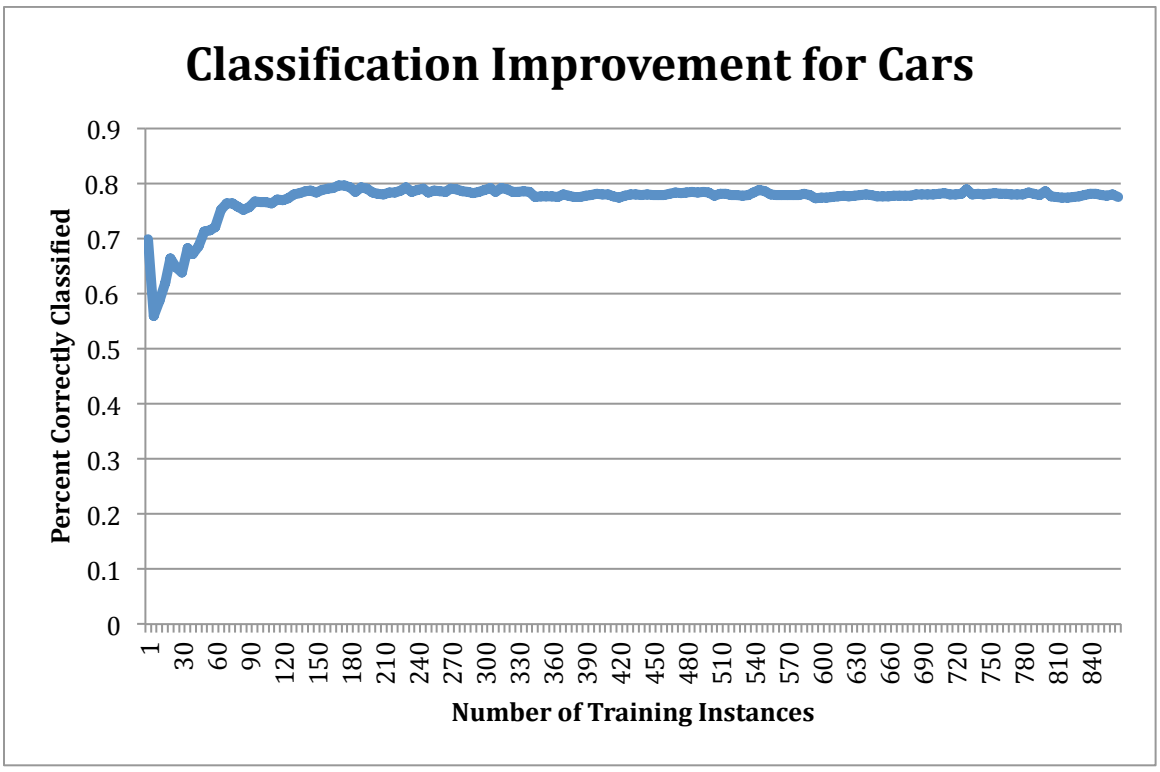

Figure 1

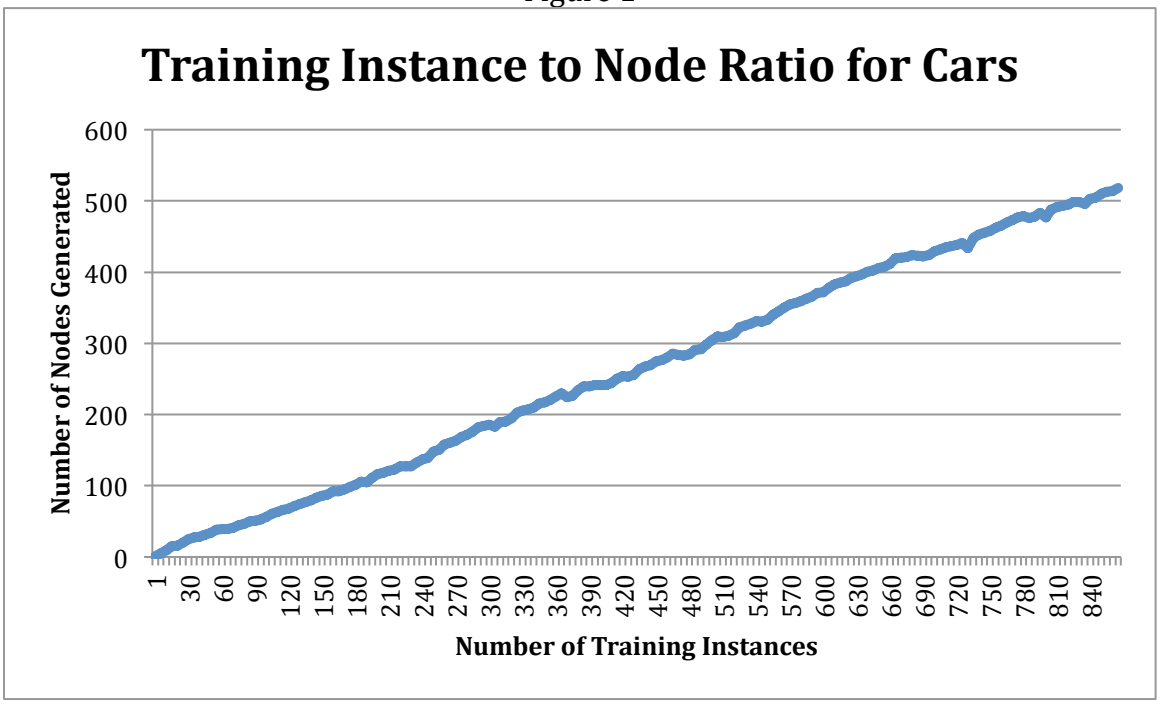

Figure 2

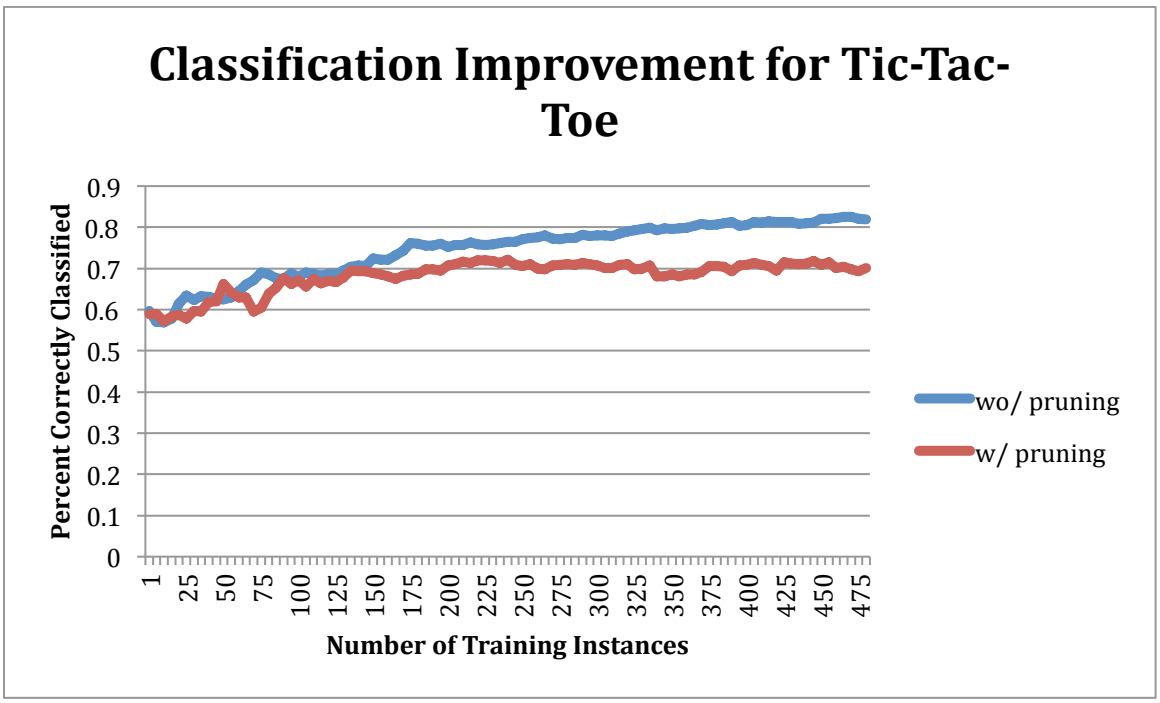

Figure 3

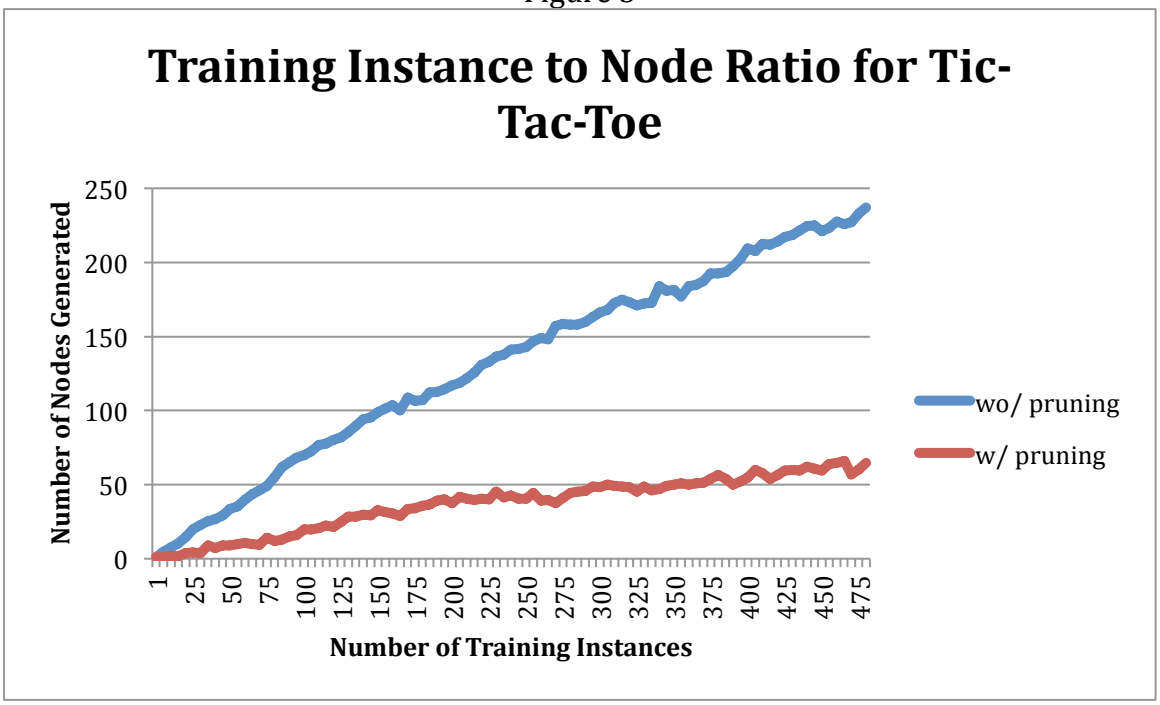

Figure 4