Tom and Jack: Irrational Agents Write-up

Implementation:

Decision Tree Algorithm:

For this algorithm, we used the framework provided in the class notes.
Our general approach to implementing the decision tree learning algorithm was as follows:

1. Read the data from the csv files into dictionaries.
2. As we progress through the tree, reduce the examples to subsets of the wider examples set.
3. When we reach a leaf node, we use the base cases to determine the value that it will possess. We implemented leaf nodes as strings.
4. We also made a Node() class for the nodes in the tree. The decision tree learning algorithm just returns the head node of the tree.

χ2 Pruning:

For chi squared pruning, we did not have an algorithm from the notes sheet. Instead, we constructed the algorithm by following the steps we took on the homework and in the in class example.
Since we already had a tree made, we traversed to the bottom of the tree first. There is only one situation where you can prune a node, which is when all of its children are leaf nodes. That said, if we prune a node below our node, we have to re-check if the current node needs to be pruned. This was the biggest conceptual challenge in coding this algorithm. Before that change, our algorithm would only try to prune the left-hand-side of our tree. (There is no literal left hand side, but it would only prune one line of branches)

Congressional Voting Data Set:

For all of these confusion matrices, we train our tree on the training data set. Each underlined set represents the set that we tested our tree on.

Confusion Matrices:

Unpruned Trees:

Testing on: Training Set

True republican: 112	True democrat: 192
False republican: 0	False democrat: 0

Recognition Rate: 1.0

Testing on: Test Tree

True republican: 48	True democrat: 72
False republican: 3	False democrat: 8

Recognition Rate: 0.916030534351145

Pruned Trees:

Testing on: Training Set

Significance = 0.05:

True republican: 112 True democrat: 185
False republican: 7 False democrat: 0
Recognition Rate: 0.9802631578947368

Significance = 0.01:

True republican: 112 True democrat: 185
False republican: 7 False democrat: 0
Recognition Rate: 0.9769736842105263

Testing on: Test Set

Significance = 0.05:

True republican: 49 True democrat: 72
False republican: 3 False democrat: 7
Recognition Rate: 0.9312977099236641

Significance = 0.01:

True republican: 50 True democrat: 72
False republican: 3 False democrat: 6
Recognition Rate: 0.9236641221374046

Results:

The results from running this program match up with the results we expected to receive. In particular, when we tested our (unpruned) decision trees on the same dataset that we trained them with, we always got a 100% recognition rate. Also, when we pruned our trees, they generally did better on the testing set than the unpruned trees. This is because by pruning our tree we were no longer overfitting our decision tree to the training set data.

As a consequence of overfitting on the unpruned tree, when we tested our *pruned* tree on the same data that we trained it with, it performed worse than the unpruned tree. Once again, we expect to see this, because while the unpruned tree is a perfect classifier for the training set data, the pruned tree is not. These results can be seen reflected in the congressional voting data set above.

Example Tree:

```
Testing physician-fee-freeze
        Branch y
        Testing synfuels-corporation-cutback
                Branch n
                Testing duty-free-exports
                Branch n
                Testing export-administration-act-south-africa
                        Branch y
                        Node with value republican
                        Branch ?
                        Testing adoption-of-the-budget-resolution
                        Branch y
                        Node with value democrat
                        Branch ?
                        Node with value republican
                        Branch n
                        Node with value republican
                        Branch n
                        Node with value republican
                Branch y
                Node with value republican
                Branch ?
                Node with value republican
                Branch y
                Testing education-spending
                Branch y
                Testing export-administration-act-south-africa
                        Branch y
                        Node with value republican
                        Branch ?
                        Node with value republican
                        Branch n
                        Node with value republican
                Branch n
                Node with value democrat
                Branch ?
                Node with value republican
                Branch ?
                Node with value republican
        Branch n
        Node with value democrat
        Branch ?
        Testing mx-missile
                Branch y
                Node with value democrat
                Branch ?
                Node with value republican
                Branch n
                Node with value democrat
Total Nodes: 25
Decision Nodes: 17
Maximum Depth: 5
Minimum Depth: 1
Average Depth of Root-to-Leaf: 3.2941176470588234
```