Eric Sun
EECS 349

1. We used an array to represent the data as a binary tree. Before pruning and cleaning the tree is a full binary tree with the left subtree of a node at index i is at 2i and the right subtree of a node at i is a 2i + 1. Every level of the tree doubles the size of the array.

The values in the tree include the value for a certain attribute, the index of the attribute, whether the node is a leaf, the count of class '0' associated with the node, the count of class '1' associated
with the node, and the index in the training set where the attribute was split.

2. We represent the examples from the training, testing, and validation sets as a 2d array. Each row represents a line in the dataset and each column in a row represnts the value of a particular attribute.

3. Each node was chosen by iterating over a list of available attributes, (using all attributes if enabled (the default setting) or a subset if disabled). For each attribute we find the best entropy by iterating over the training data set. We sort about the pivot associated with the node to find the splitting attribute with the least entropy. This took about O(n + nlogn) time for each node where n is the number of training examples associated with a node.

4. We handled missing attributes in the training set by finding the average value of that attribute over the training set and substituting that value in for the missing value. For the validation set we tried two different methods. We tried substituting the average value found in the training to replace the missing value in the missing data set. We also tried a method where if there is a missing value, we move to the node in the decision tree with the majority associated training examples.

5. The termination criteria varied for pruning vs unpruned. For pruned, termination can occur if certain thresholds used in pruning are met. See quesiton 8 for more details. Otherwise termination will occur once the maximum number of levels has been reach, which depends on the number of attributes.

6. See attached document (need to complete later)

7. |||= Split on attribute  opprundifferential < 27.0 into left subtree, else right subtree

This is a 4th level node, but not a leaf node. At this node we check the test examples value for opprundifferential and if it is less than 27 we go to the left, else we go to the right.

8. The pruning method is modeled after reduced error pruning. We used a greedy strategy during tree generation, with two different methods used. If the ratio of one class is greater than threshold, no subtrees are created and that class is used as the estimate and the node becomes a leaf. If there are relatively few examples associated with a node compared to the overall number of training examples, then no subtrees are generated, the majority class for that node is then used and thenode becomes a leaf. In addition the tree is trimed from the bottom up in both pruned and unpruned trees to collapse the tree of redunant leaves into their parent nodes.

9. (To do later)

10. We have 19 nodes on our decision tree when it is set to use a maximum level of 12 and set to pruned and cleaned vs 39 nodes on a decision tree using maximum 12 levels, pruned and unclean.

We have 55 nodes for a tree using a maximum of 6 levels and unpruned (we use 6 here because 12 takes way too long unpruned) and cleaned. We have 127 nodes for a tree using again a maximum of 6 levels and unpruend but this time uncleaned.

11. For unpruned cleaned six level tree we get an accuracy of 61% (60.996%). For pruned cleaned 12 level tree we get an accuracy of 62.546%. This occured because more attributes where used in the six level tree but they were not neccesarily helpful. This probably cuased the decrease in accuracy. Overfitting on the training set also may have been a factor in causing the decreased accuracy.

12. (to do)

13. (to do)

14. It appears number injured and oppnuminjured and winpercent where the most important attributes in the decision tree. The least helpful attributes turned out to be weather and tempeature along with startingpticher and oppstartingpitcher. The rest of the attributes where more or less the same in importance.
Weather and temperature make sense because they affect both teams, the fact that pitchers were not useful was kind of surprising. This can be partly explained by the fact that maybe most teams play games with the same quality of pitcher starting. (for example its possible a team's fourth rotation guy would play against an opposing teams bottom rotation pitchers) and the fact that there probably isn't a noticeable difference between the quality of pitchers lower in the roation say the third and fifth guys. Numinjured was good and makes sense as being an important attribute, unfortunately when resusing attrbiutes, it made the tree more inaccurate because numinjured became overused on the tree. It might be better if looking at a

fuction of numinjured and oppnuminjured together than alone. It also might be helpful to look at a comparison between one team's pitcher vs another team's pitcher. (3vs 4 may not make a big difference but 1vs 5 probably would).

15. I worked on the learning curve, classification, validation, and bulding the tree. Chris worked on building the tree, printing the tree, pruning, classficiation and cleaning.