Homework 1
CS 436/580L: Introduction to Machine Learning

Instructor: Arti Ramesh

1 Instructions

1. You can use either C/C++, Java or Python to implement your algorithms.

2. **Your implementations should compile on remote.cs.binghamton.edu.**

3. Make sure remote.cs.binghamton.edu has the packages that you require before starting to implement.

4. This homework requires you to implement decision tree. Using existing decision tree package is not allowed.

5. Your homework should contain the following components:

   (a) README.txt file with detailed instructions on how to compile and run the code.

   (b) Code source files

   (c) Type-written document containing the results on the datasets.

   (d) For complete credit, you need to report correct results on both the datasets, for both the heuristics, with and without pruning. If you fail to report results for all the above-mentioned combinations, points will be deducted.

6. Submit the homework as a **single zip file:** $firstname\_lastname\_hw1.zip$.

Inducing Decision Trees

In this homework you will implement and test the decision tree learning algorithm (See Mitchell, Chapter 3).

- Download the two datasets available on myCourses. Each data set is divided into three sets: the *training set*, the *validation set* and the *test set*. Data sets are in CSV format. The first line in the file gives the attribute names. Each line after that is a training (or test) example that contains a list of attribute values separated by a comma. The last attribute is the class-variable. Assume that all attributes take values from the domain {0,1}.

- **(60 points)** Implement the decision tree learning algorithm. As discussed in class, the main step in decision tree learning is choosing the next attribute to split on. Implement the following two heuristics for selecting the next attribute.

    1. Information gain heuristic (See Class slides, Mitchell Chapter 3).
    2. Variance impurity heuristic described below.

       Let K denote the number of examples in the training set. Let K_0 denote the number of training examples that have $class = 0$ and K_1 denote the number of training examples that have $class = 1$. The variance impurity of the training set S is defined as:

       $$VI(S) = \frac{K_0}{K}\frac{K_1}{K}$$

       Notice that the impurity is 0 when the data is pure. The gain for this impurity is defined as usual.

       $$Gain(S, X) = VI(S) - \sum_{x \in Values(X)} \Pr(x) VI(S_x)$$

       where X is an attribute, S_x denotes the set of training examples that have $X = x$ and $\Pr(x)$ is the fraction of the training examples that have $X = x$ (i.e., the number of training examples that have $X = x$ divided by the number of training examples in S).

- **(40 points)** Implement reduced error post pruning algorithm on the tree from Question 1. The steps in the algorithm are given below. (See also Mitchell, Chapter 3).

1. Consider each node for pruning.
2. Pruning = removing the subtree at that node, make it a leaf and assign the most common class at that node.
3. A node is removed if the resulting tree performs no worse then the original on the *validation* set.
4. Nodes are removed iteratively choosing the node whose removal most increases the decision tree accuracy on the graph.
5. Pruning continues until further pruning is harmful.

Your implementations should be called from the command line to get complete credit. Points will be deducted if it is not possible to run all the different versions of your implementation from the command line.

- Implement a function to print the decision tree to standard output. We will use the following format.

  ```
  wesley = 0 :
  | honor = 0 :
  | | barclay = 0 : 1
  | | barclay = 1 : 0
  | honor = 1 :
  | | tea = 0 : 0
  | | tea = 1 : 1
  wesley = 1 : 0
  ```

  According to this tree, if wesley = 0 and honor = 0 and barclay = 0, then the class value of the corresponding instance should be 1. In other words, the value appearing before a colon is an attribute value, and the value appearing after a colon is a class value.

- Once we compile your code, we should be able to run it from the command line. Your program should take as input the following five arguments:

  ```
  .\program <training-set> <validation-set> <test-set> <to-print>
  to-print:{yes,no} <prune> prune:{yes, no}
  ```

It should output the accuracies on the test set for decision trees constructed using the two heuristics. If to-print equals yes, it should print the decision tree in the format described above to the standard output.

- A README.txt file with detailed instructions on compiling the code.
- A document containing the accuracy on the training, validation, and test set in both datasets for decision trees constructed using the two heuristics mentioned above, with and without pruning.