CIS 472/572, Winter 2018
Programming Homework 1: Decision Trees

DUE DATE: Submit via Gradescope by Wednesday, January 31st at 11:00pm.

In this assignment, you will implement the ID3 algorithm for learning decision trees. You may assume that the class label and all attributes are binary (only 2 values). Please use the provided skeleton code in Python to implement the algorithm. You may look at open-source reference implementations, such as WEKA, but please **do not copy code from open-source projects**. Your code must be your own. Undergraduates may complete the assignment in teams of 2. Graduates must complete the assignment alone.

The ID3 algorithm is similar to what we discussed in class: Start with an empty tree and build it recursively. Use information gain to select the attribute to split on. (Do not divide by split information.) Use a threshold on the information gain to determine when to stop. The full algorithm is described in this classic paper (with over 11,000 citations):

`http://dept.cs.williams.edu/~andrea/cs374/Articles/Quinlan.pdf`

You code should run from the command line on ix and accept the following arguments:

```
./id3 <train> <test> <model>
```

Where `train` is the name of a file containing training data, `test` contains test data to be labeled, and `model` is the filename where you will save the model for the decision tree.

The data files are in CSV format. The first line lists the names of the attributes. The last attribute is the class label. Examples will be posted on the class web page. An example tree for at least one dataset will be posted online as well.

This year, we are providing skeleton code in Python that handles input, output, and some of the internal data structures. Please use it as the starting point because we'd be using that API to grade.

For saving model files, please 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 three command line arguments, as shown below:

```
./id3 <training-set> <test-set> <model-file>
```

It should output the accuracy of the decision tree on the test set and write the decision tree in the format described above to the specified file.

1 Extra Credit

If you have the time, there are lots of ways you can extend this assignment to get a few points of extra credit:

- Run another decision tree learner, such as the J48 learner in Weka (which is an implementation of the C4.5 algorithm). How does the accuracy compare? How does the learned model compare?
- Implement a different split criterion, such as GINI or one-step lookahead accuracy. Do they lead to trees that are substantially more or less accurate?
- Implement reduced error pruning. In reduced error pruning, you grow a full decision tree (with pure leaves) and then prune the leaves as long as the accuracy on the validation set continues to increase.
- Find one or more datasets of interest and run your algorithm on them. Report the results and discuss how well it works.

These are all optional, but doing at least one is highly recommended.

If you choose to do the extra credit, please turn in your code and write-up **separately**. Your extra credit code may follow a different interface.