1 Description

1.1 Part A

Solve the following problems from Chapter 3 of Mitchell’s machine learning book. This chapter can be accessed from the following link:


1. Exercise 3.1

2. Exercise 3.2

3. Consider the samples in the Play-tennis dataset in Figure 1. If you calculate the information-gain for all of the attributes of this set, you will observe that the attribute "Outlook" has the largest information-gain, which is equal to 0.246. Therefore, the attribute "Outlook" is the best heuristic choice for the root node.

   (a) List the label of new tree branches below the root node.

   (b) Which partition of the data will be assigned to each branch by ID3? Please list the sample ID’s that will be assigned to each branch.

   (c) Calculate the information gain for the remaining attributes in each branch, and determine which attribute will be chosen as the root of the sub-tree in each branch.

4. Consider a machine learning algorithm, which is designed by a bank to decide if crediting a new customer is risky or not. The decision will be based on some attributes of the new customer such as his/her income, employment status, etc. This algorithm learns two independent decision trees for risk assessment in the training time. At the time of deployment, if both of the decision trees classify a customer as non-risky then the bank will credit the customer; otherwise, his or her application will be declined. If the bank had hired you to answer the following questions, what would be the right answers?

   (a) Describe an algorithm for converting this pair of decision trees into a single decision tree that makes the same predictions (that is, it predicts non-risky only when both of the original decision trees would have predicted non-risky).
Let $n_1$ and $n_2$ be the number of leaves in the first and second decision trees, respectively. Provide an upper bound on $n$, the number of leaves in the single equivalent decision tree, expressed as a function of $n_1$ and $n_2$. Argue if it is possible to find the equivalent single tree such that $n \leq n_1 + n_2$.

### 1.2 Part B

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). You may implement the algorithm in your choice of programming language. 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 Chi-squared test 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 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.

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.

## 2 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.
3 What To Turn In

Please hand-in a hard-copy of your solutions for part A in class on January 21st, and submit your code and a Readme file for compiling it to ali@cs.uoregon.edu by 11pm on January 23rd. Your code should follow the file format and interface discussed above.

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.