CSC 261 **Lab 10: Decision Tree Analysis** **Spring 2018**

Assigned: Tuesday 17 April

Due: Monday 23 April, 11:59 pm

Objectives: Understand decision tree **classification** and practice techniques for **analyzing** classifiers

Collaboration: This homework assignment will be completed in pairs assigned by the instructor.

You have built a decision tree learner. How can you use it to make classifications? How does its performance scale with training data set size?

Background

The starter code for this assignment also contains these files you copied from the MathLAN directory

```
~weinman/courses/CSC261/code/dtree
```

`analysis.scm` Routines for helping you perform some analysis. Namely, splitting the example data into train/test subsets and selecting a (smaller) random subset of examples for training.

`classify.scm` Documentation for the procedure you will write.

How do you classify an instance using a decision tree? Given the tree's recursive structure, classification is straightforward. If the decision tree starts with a `cons` cell (which you can test using the predicate `pair?`), then you know you need to apply an attribute test. You will need to

- get the attribute test specified in the decision tree's root,
- get the value of that attribute in the instance (using `assoc`),
- follow the branch of the decision tree having that value to get the next decision tree (again using `assoc`), and then
- recursively apply the classification routine on the next decision (sub)tree.

Otherwise, the decision tree is simply a classification value, which may be directly returned. For example:

```
> (load "restaurant.scm")
> (load "dtree.scm")
> (load "learning.scm")
> (load "classify.scm")
> (define restaurant-tree  (decision-tree-learning restaurant-examples
                                                   restaurant-attributes #t)
> (cdar restaurant-examples)
(("Alt" . #t) ("Bar" . #f) ("Fri" . #t) ("Hun" . #t) ("Pat" . "Some")
 ("Price" . "$$$") ("Rain" . #f) ("Res" . #t) ("Type" . "French")
 ("Est" . "0-10"))
> (decision-tree-classify restaurant-tree (cdar restaurant-examples))
#t
```

Assignment

Problem 1: `decision-tree-classify`

Implement the `decision-tree-classify` procedure documented in `classify.scm`. Note the structure of the procedure outlined above and consult Exercise D.2 of Lab: Decision Trees.

Problem 2: Analysis

You now have everything you need to train *and* test your supervised learning classifier. Note that `dtree.scm` includes the procedure `(decision-tree-accuracy` *`decision-tree examples`*`)`, which allows you to test the accuracy (a number between 0 and 1) on some examples. Of course, the examples need not be the same ones used to train the given decision tree.

Compiled implementations (which you copied with the starter code) of `decision-tree-learning` and `decision-tree-classify` are optionally available. To use them, add the following to your analysis program before loading `analysis.scm` or `dtree.scm`:

```
(require "weinman-dtree.scm")
```

Part A

On the last assignment you built a tree from the entire mushroom data set. What is the accuracy of the resulting decision tree on the same data? Explain why this is necessary.

Part B

Here you will apply the methodology for assessing a learning algorithm using the mushroom data. Namely, this requires us to:[1]

1. Collect a large set of examples (which we have already done).
2. Divide the examples into two disjoint sets, one for training and another for testing.
3. Apply `decision-tree-learning` to a training set, generating a hypothesis "function" h.[2]
4. Measure the accuracy on the test set made by h
5. Repeat steps 2 to 4 for different sizes of training sets and different randomly selected sets of each size.

Note that this process empirically assesses the "generalizability" of the learned hypothesis function (i.e., decision tree) by measuring the accuracy on testing data that was not used to train the classifier.

The procedure `(run-trials` *`examples attributes default num-trials`*`)` in `analysis.scm` does precisely this sort of assessment, measuring the **error rate** (not accuracy) on a *single* test set with randomly selected training subsets of lengths 0,5,10,15,20,25,...,85,90,95,100. It reports the average error rate over several trials for each length.

Use this procedure and display your learning curve in a graph, much like Figure 18.7 (p. 703). Title your graph, clearly label your axes, and include a caption briefly explaining the data (noting its source), evaluation procedure, and any relevant learning and evaluation parameters.

What to turn in

Your submission should include the following

- Your completed `classify.scm`
- The short `driver.scm` program that demonstrates your classification procedure is correct and also generates the raw data for Problem 2
- A transcript `output.txt` of your driver program's output
- A PDF file `analysis.pdf` with your answer to Problem 2(A) and your graph (etc.) of Problem 2(B)

[1] Adapted from "Assessing the performance of the learning algorithm," AIMA 2/e pp. 660–661.

[2] Note that for our problem, this is not a function in the `lambda` sense. It is the decision tree itself (which can be used as an input to the function `decision-tree-classify`).