

CSE 5693 Machine Learning HW2
Due 6:30pm, Feb 25, 2009
Submit Server: course=ml , project=hw2

1. Written assignment:

- (a) 2.4
- (b) 2.7
- (c) 2.9
- (d) 3.4
- (e) Consider two attributes Outlook (sunny, rainy, cloudy) and Humidity (high, low) and outcome PlayTennis (yes, no) for the instance space (X) and an unbiased hypothesis space (H):
 - i. Enumerate all possible “Yes” hypotheses (h_1, h_2, \dots) in terms of subsets of instances.
 - ii. For each hypothesis, represent it as a boolean expression.
 - iii. Identify which hypotheses are in the biased hypothesis space if an attribute can only have a value, ?, or \emptyset .

2. Programming assignment: Decision Tree

- (a) Allow more than binary outcomes and continuous-valued attributes.
- (b) Allow the option of pre-pruning (handout)
- (c) Two data sets: PlayTennis in the book (Table 3.2) and another with continuous attributes and multiple outcomes
e.g. from <http://www.ics.uci.edu/~mlearn/MLRepository.html>
- (d) Separate the data set into a training set and a test set, report the accuracy on the two disjoint sets (with and without pre-pruning).
- (e) For the second data set, corrupt the class labels of training examples from 0% to 50% (5% increment), by changing from the correct class to another class. Compare the accuracy of the tree with and without pre-pruning on uncorrupted test data.
- (f) Implementation:
 - i. Use one of these programming languages: C, C++, Java, or LISP.
 - ii. You would have two (maybe three) executables:
 - A. Learner: input training examples/instances, output a tree
 - B. Classifier/predictor: input a tree and labeled instances, output the classifications/predictions and how accurate the tree is with respect to the correct labels (% of correct classifications).
 - C. Tree printer: if the output from the learner is human-readable, no need for a tree printer; otherwise, build a tree printer so that we can see the built tree.

- iii. You might want to think of a scheme of reading in examples/instances, which can be reused for the next two assignments (of course, if you want, you can rebuild the example/instance-reading part again). That is, you might want to fix the input file format for examples/instances. For example, each line is an instance and the class label is always at the end (like Table 3.2 on page 59):

```
Sunny Hot High Weak No  
Sunny Hot High Strong No
```

or the class labels are always in the front:

```
No Sunny Hot High Weak  
No Sunny Hot High Strong
```

You can drop Day since it's a unique id for each example.

- iv. You might want to put the attribute names and all their possible values in a separate file (sometimes called the “dictionary file” or you can put them at the beginning of the example file.) This allows your program to be run on different problems (not just the “play tennis” problem). You need that for tree building/printing and checking for invalid values. For example,

```
PlayTennis Yes No  
Outlook Sunny Overcast Rain  
Temperature Hot Mild Cool  
...
```