CSEP 573: Applications of Artificial Intelligence

Homework Assignment #3

Due: In class on Monday, March 1, 2010

Turn-in procedure: Bring a hardcopy containing your answers to class on March 1. Make sure you write your name on the hardcopy.
If you are unable to make it to class, use the dropbox:
https://catalysttools.washington.edu/collectit/dropbox/afriesen/8677
and submit your answers before class time on March 1. Use the naming convention HW3_<LastnameFirstname> for your file(s).

Reading: Sections 15.1 & 15.2 and Chapter 18 in AIMA (2nd/3rd ed.)

Problems:

1. [30 points for *a* and *b*] *Bayesian Filtering*. Consider the umbrella/rain example in the textbook with the probabilities given in Fig. 15.2. We know the director came with an umbrella both yesterday and the day before. Suppose the director walks in today *without* an umbrella. Assuming we started out with a uniform distribution for rain two days ago,
   a. What is the probability that it is raining today?
   b. What is the probability that it will rain tomorrow?
   (Hint: The situation for the first two days is already worked out in the textbook)

   Optional Extra Credit (10 points each):
   i. Compute the probability that it will rain *k* days into the future for *k* = 1,..,20 and plot the results. What value does the probability appear to be converging to?
   ii. Prove algebraically that the value in (i) above is a fixed point for the predicted rain distribution.

2. [40 points] *Decision Trees*. Exercise 18.6 in AIMA (3rd ed.).

3. [30 points] *Neural Networks*. Exercise 18.22 (all parts) in AIMA (3rd ed.).

Optional Extra Credit Problem (10 points): Exercise 18.17 in AIMA (3rd ed.).

Next two pages contain scans from AIMA (3rd ed.)
[for those who have AIMA 2nd ed. only]

18.6 Consider the following data set comprised of three binary input attributes (A_1, A_2, and A_3) and one binary output:

Example	A_1	A_2	A_3	Output y
$\mathbf{x}_1$	1	0	0	0
$\mathbf{x}_2$	1	0	1	0
$\mathbf{x}_3$	0	1	0	0
$\mathbf{x}_4$	1	1	1	1
$\mathbf{x}_5$	1	1	0	1

Use the algorithm in Figure 18.5 (page 702) to learn a decision tree for these data. Show the computations made to determine the attribute to split at each node.

function DECISION-TREE-LEARNING(*examples*, *attributes*, *parent_examples*) **returns** a tree

if *examples* is empty **then return** PLURALITY-VALUE(*parent_examples*)
else if all *examples* have the same classification **then return** the classification
else if *attributes* is empty **then return** PLURALITY-VALUE(*examples*)
else
  $A \leftarrow \text{argmax}_{a \in attributes}$ IMPORTANCE(a, *examples*)
  tree $\leftarrow$ a new decision tree with root test A
  for each value v_k of A **do**
    $exs \leftarrow \{e : e \in examples \text{ and } e.A = v_k\}$
    subtree $\leftarrow$ DECISION-TREE-LEARNING(exs, $attributes - A$, *examples*)
    add a branch to *tree* with label $(A = v_k)$ and subtree *subtree*
  return *tree*

Figure 18.5 The decision-tree learning algorithm. The function IMPORTANCE is described in Section 18.3.4. The function PLURALITY-VALUE selects the most common output value among a set of examples, breaking ties randomly.

18.22 Suppose you had a neural network with linear activation functions. That is, for each unit the output is some constant c times the weighted sum of the inputs.

a. Assume that the network has one hidden layer. For a given assignment to the weights **w**, write down equations for the value of the units in the output layer as a function of **w** and the input layer **x**, without any explicit mention of the output of the hidden layer. Show that there is a network with no hidden units that computes the same function.

b. Repeat the calculation in part (a), but this time do it for a network with any number of hidden layers.

c. Suppose a network with one hidden layer and linear activation functions has n input and output nodes and h hidden nodes. What effect does the transformation in part (a) to a network with no hidden layers have on the total number of weights? Discuss in particular the case $h \ll n$.

Extra credit:

18.17 Construct a support vector machine that computes the XOR function. Use values of +1 and –1 (instead of 1 and 0) for both inputs and outputs, so that an example looks like $([-1, 1], 1)$ or $([-1, -1], -1)$. Map the input $[x_1, x_2]$ into a space consisting of x_1 and $x_1\, x_2$. Draw the four input points in this space, and the maximal margin separator. What is the margin? Now draw the separating line back in the original Euclidean input space.