Homework #7: Chapters 18 - 19

The following exercises are due at the beginning of class on Friday, April 27. Note, this homework is continued on the reverse side of the paper.

1. *[25 pts.]* Consider the training set in Table 1 below.

Example	Type	Garage	Bedrooms	Bathrooms	Goal Predicate
X_1	TownHouse	Yes	2	1.5	Yes
X_2	Condo	No	2	2	No
X_3	Apartment	Yes	2	1	No
X_4	TownHouse	Yes	4	2	No
X_5	TownHouse	Yes	3	1.5	Yes
X_6	Condo	Yes	1	1	No
X_7	TownHouse	No	3	1.5	No
X_8	Apartment	Yes	3	1	No

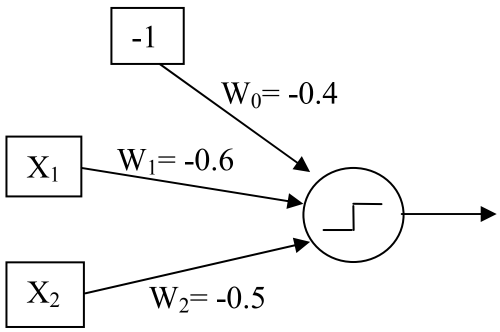
Table 1. Housing Training Set

a) *[4 pts.]* Calculate the information requirement for the training set.
b) *[16 pts.]* Calculate the information gain for each of the four attributes: *type*, *garage*, *bedrooms*, and *bathrooms*. **Hint:** If you don't have a calculator capable of doing base 2 logarithms, you can calculate them using the natural logarithm: $\log_2 x = \ln x / \ln 2$.
c) *[5 pts.]* Based on your findings in part b), draw a partial decision tree that includes the attribute on which the first test should be performed and its immediate child nodes. You may use a question mark for the attribute of any nodes that cannot be completely classified by the first attribute test.

2. *[35 pts.]* Consider the training set in Table 1 again.

a) *[8 pts.]* Convert the training set into a set of first-order logic description and classification sentences. Use the predicates *Type(x,t)*, *Garage(x,g)*, *Beds(x,b)* and *Baths(x,b)* in your description sentences and *Q(x)* for your goal predicate. Note that by using a binary predicate for garage (instead of a unary predicate), we are able to restrict our hypothesis space to only include candidate definitions that are positive conjunctive sentences.
b) *[27 pts.]* Use current best-hypothesis search learning on the training set. As was demonstrated in class, assume the algorithm is implemented as a depth-first search. The initial hypothesis should either be "True" or "False", and each node in your search tree should show the current hypothesis and how it classifies the current example (i.e., true positive, true negative, false positive or false negative). When a node is expanded, its children are determined by the consistency of its hypothesis. If it is consistent, then it has a single child with the same hypothesis considering the next example. If it is inconsistent, then the children's hypotheses are the immediate specializations (or generalizations) that are consistent with all examples to that point. Assume that the examples are received in the order given and that the hypothesis space only contains hypotheses whose candidate definitions are positive conjunctive sentences. When there is a choice of otherwise equivalent nodes to expand, always choose the one that adds the leftmost remaining condition from the attributes in the table.

3. *[25 pts.]* Use the perceptron learning algorithm to teach the perceptron below to recognize the boolean NAND function. Assume that a **threshold activation function** is being used (i.e., the function returns 1 when its input is ≥ 0 and returns 0 otherwise). For initial weights, use $W_0 = -0.4$, $W_1 = -0.6$, and $W_2 = -0.5$. For the learning rate, use $\alpha = 0.1$. Use only the examples in the table to the right of the network in your learning process (note, Y is the correct output for the example). Stop the training once the weights remain unchanged for one full pass through the examples. The examples must be used in the order given by the table below. Start again with the first example whenever you exhaust all of the examples but have not yet reached the stopping criteria. Show all of the intermediate calculations and values (not just the answer or the updated weights after each example).



Training Set		
X_1	X_2	Y
0	1	1
1	1	0
0	0	1
1	0	1

4. *[15 pts.]* Construct by hand (i.e., not following the steps of any particular algorithm) a feed-forward neural network that computes the exclusive or (XOR) function of two inputs. Assume that 1 = true and 0 = false for all inputs and outputs. Be sure to say which activation function you are using.