

15-381 - Fall 2001
Homework 5 Solutions

Due: Thursday, November 15, 2001

Grading: 4 points for each sub-problem.

Problem 1

There were a variety of examples for this problem. Anything reasonable was accepted.

Problem 2: Decision Tree Learning

- (a) The entropy of the initial set of examples is 1.
- (b) Information Gain (Home_Owner) = 0.1872
- (c) Information Gain (In_Debt) = 0
- (d) Information Gain (Rich) = 0.0623
- (e) Home_Owner, because the split yields the highest information gain.

Problem 3: Decision Tree Learning

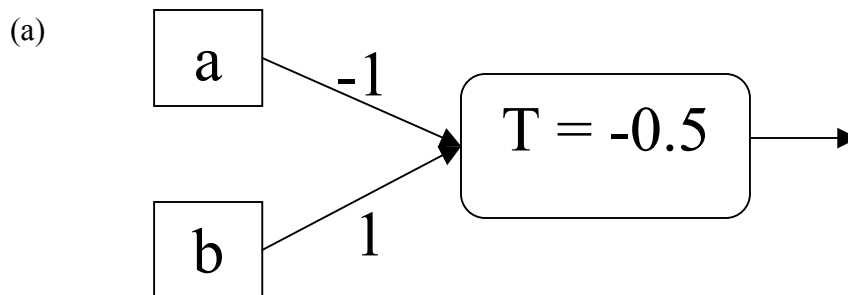
- (a) +, +, -, +
- (b) blue, circle, rotten egg => - : (2/3) -, 1/3 (+)
medium, red => + : (4/15) -, (11/15) +
- (c) The attribute will have a strictly positive information gain, unless $p_i/(p_i+n_i)$ is the same for all i subsets. If this is true, then each subset will have the same ratio of positive and negative examples as the original set of examples E . Splitting on this attribute will thus yield no information gain (the split has not helped to categorize examples). If $p_i/(p_i+n_i)$ is not the same for all i , the split is helping categorize the examples, hence there will be some positive information gain.

Problem 4: Inductive Learning

- (a) Cross validation is the technique that eliminates the danger of overfitting. The basic idea of cross validation is to try to estimate how well the current hypothesis will predict unseen data. This is done by setting aside some fraction of known data, and using it to test the prediction performance of a hypothesis induced from the rest of the known data. This can be done repeatedly with subsets of the data, and results averaged.

- (b) In supervised learning, both inputs and outputs of a component are perceived. So, the model knows the correct output for a given input. In unsupervised learning, there is no hint of the correct outputs. The learner must learn relationships among its percepts using supervised learning methods. It can learn to predict its future using previous percepts. It must first, however, have a utility function.
- (c) Training is a function of changing weights of input links. The change is gradual and the change in one unit affects other units. We have to train the net on multiple trials of the same examples in order to reach acceptable convergence.
- (d) Credit given for all reasonable answers.

Problem 5: Perceptrons and Neural Nets



- (b) There are numerous ways to do this. You can just add an additional unit at the end that flips the output of the current network.
- (c) Connect all m positive literals (with weight 1) and k negative literals (with weight -1) to a step thresholded perceptron with activation value $t = m - 0.5$.
- (d) Use the same connections as above, but change the activation to $t = -k + 0.5$.
- (e) Yes, propositional boolean functions can all be broken into a sum of products or a product of sums (ie conjunctions and disjunctions). We have already shown that a single layer can be used to create either of these, so two layers can be used to represent any propositional boolean function.

Problem 6: Neural Networks

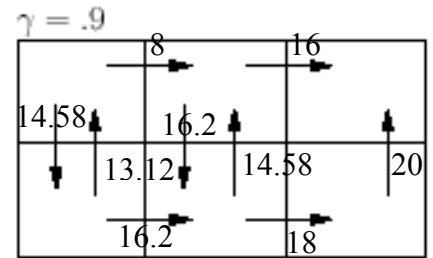
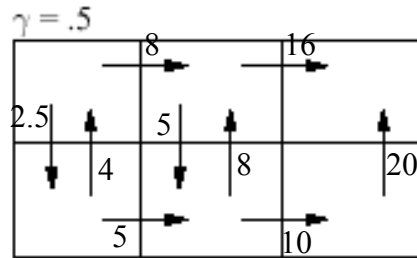
- (a) $(A \wedge \neg B \wedge \neg C) \vee (\neg A \wedge B \wedge \neg C)$
- (b) This is the same as $A \oplus B$ (which was the original network shown in 5b).

- (c) We accepted breadth-first, or hill-climbing (gradient descent). The way it does this is to generate an error surface depending on the different weights used. Then it would keep finding better weights with less error until weights that work are found (see p 580-582).

Problem 7: Reinforcement Learning

- (a) Only the value for the movement going southward in the center column changes (from 4 to 5).

- (b)



- (c)

