Homework Set #5

Due: 12/10/03

Note: You do not have to turn in this homework assignment. If you do, you may get extra credit. Your final homework score will be computed by discarding you lowest score and averaging over the remaining scores.

1 Problem: error correction learning

In the class we derived a gradient descent method for perceptron learning that assumes we use a sigmoid function $g(in)$.

Consider now a slightly different formulation of the learning problem. Assume we want to devise a learning rule that does the following:

1. Activation function $g(in)$ is the (hard) threshold function.
2. Assume (incorrectly) $g'(in) = 1$, for all in.

Derive a gradient learning rule that minimizes the sum of square errors $E(w) = 1/2 \sum_{k=1}^{K} (y_k - g(in_k))^2$.

Discuss how this learning rule updates the network weights and how it is different from the gradient learning rule we talked about in the class (assume that after we train the network weights using the rule you just derived, we add back the threshold function so that the network output once again becomes 0 or 1.)

2 Problem: Widrow-Hoff learning

Consider yet another formulation of the learning problem. Assume we want to devise a learning rule that does the following:

1. Instead of labeling points 1 or 0 it labels them as $+1$ or -1.
2. Activation function $g(in)$ is linear, i.e., $g(in) = in$.
3. After the weight vectors are learned using the above two assumptions we use the old threshold activation function $g(in)$ to classify any new points.

Derive a gradient learning rule that minimizes the sum of square errors $E(w) = 1/2 \sum_{k=1}^{K} (y_k - g(in_k))^2$.

3 Problem:

The following training set is linearly separable:

```
input  output
1 0 0    1
0 1 1    0
1 1 0    1
1 1 1    0
0 0 1    0
1 0 1    1
```

Train (by hand) a linear threshold element on this training set. Your unit will have four inputs counting the one that implements the threshold. Assume that the initial values of all weights are zero. Train your unit with the learning procedure in Problem 1 until it converges to a solution. Show the set of weights at the end of each pass through a training cycle. Draw a sketch of a 3-D cube with the preceding inputs as vertices, and sketch in the separating plane corresponding to the final weight set. Assume α (the learning rate) is one.

Extra credit: Use the Widrow-Hoff algorithm on this problem (you can write a short program that implements Widrow-Hoff). Assume the learning rate $\alpha = 0.1$. Compare results with those of the error-correcting rule.

4 Problem:

Prove that it is possible to implement any Boolean function of N inputs with a network of threshold elements consisting of just one hidden layer.