Exercises

2. Hand-simulating backpropation of error as in the previous example, repeat the calculation for the following two cases:

   High output-layer weights: $w_{11}^{(1)} = 3.0, w_{12}^{(1)} = -3.0, w_{21}^{(1)} = 3.0, w_{22}^{(1)} = 3.0$
   Small output-layer weights: $w_{11}^{(1)} = 0.3, w_{12}^{(1)} = -0.3, w_{21}^{(1)} = 0.3, w_{22}^{(1)} = 0.3$

   Observe the relative changes in the weights in each case.

Give it Some Thought

1. How will you generalize the technique of backpropagation of error so that it can be used in a *multilayer perceptron* with more that one hidden layer?
2. Section 5.1 suggested that all attributes should be normalized here to the interval $[-1.0, 1.0]$. How will the network's classification and training be affected if the attributes are not so normalized? (hint: this has something to do with the sigmoid function)

Computer Assignments

1. Write a program that implements backpropagation of error for a predefined number of output and hidden neurons. Use a fixed learning rate, η.
2. Apply the program implemented in the previous task to some benchmark domains from the UCI repository.[3] Experiment with different values of η, and see how they affect the speed of convergence.