Computer Science 152, Fall 2010

Assignment 1

Due Tue. 7 September 2010

Construct a self-contained program for training a perceptron. You may code in any language of your choice, although you might want to focus on higher-level, interactive, languages such as Matlab, Python, Scheme, etc. to enable concentration on essential features rather than low-level aspects, which require more infrastructure coding. Also, there will be more problems along these lines, so aim for reusability of code.

Construct the perceptron, activation, and learning functions from scratch, rather than using any pre-constructed functions or demos.

I will put on the home page several data sets that you should run. These will be in S-expression format, but you may edit them to suit. Here is an example of the formatting:

```
(define or-example '(
        (0 0 0)
        (1 0 1)
        (1 1 0)
        (1 1 1)
))
```

In each case, the sample points consist of a single output value first, followed by the corresponding input vector. For example, above is an or-gate, so one sample point is (1 1 0), meaning that the output is 1 for an input of (1 0). A perceptron can be trained to implement this function exactly, so this is a good first test case. A much bigger example is the "cancer" data set. Here the inputs are not just 0's and 1's, but rather integers. The integers are results of clinical tests, while the output is indicator of benign (0) or malignant (1). We expect good, but not exact classification in this case.

```
(define cancer-example '(
   (0 5 1 1 1 2 1 3 1 1)
   (0 5 4 4 5 7 10 3 2 1)
   (0 3 1 1 1 2 2 3 1 1)
   (0 6 8 8 1 3 4 3 7 1)
   (0 4 1 1 3 2 1 3 1 1)
   (1 8 10 10 8 7 10 9 7 1)
   . . . abridged .. . .
))
```

You might want to start with a subset of the data, until you are convinced that your program is working. The cancer example is derived from one in the UCI Machine Learning Repository, which you can visit to learn more about the meaning of the data: http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/

What to submit (please use paper for this assignment):

1. We are interested in an exposition, rather than just raw evidence.
2. Submit code or code fragments that show your approach.
3. For each test case, show number of misclassified samples after training, % error, number of epochs taken, and final weights. Do not submit reams of output, but rather excerpts of nominal output to show that your program is working correctly.
4. Most importantly, summarize your results and what you learned, in prose.