CSE 5693 Machine Learning HW3
Due 6:30pm, Mar 17, 2015
Submit Server: Class = cse5693 , Assignment = hw3

1. Written assignment (from the textbook):
   (a) 4.1
   (b) 4.2: (by hand with at most one decimal place for the weights, not by a program to gain a better understanding) specify the weights and include two tables for the values in the perceptron: rows=input combinations
      i. columns=input, output values (before and after threshold)
      ii. columns=input, hidden, output values (before and after threshold)
   (c) 4.9
   (d) With the programming assignment:
      i. discuss the hidden values in testIdentity using 3 and 4 hidden units (Why do 4 hidden units also work? What do the hidden values represent? Is the magnitude of hidden values significantly different between 3 and 4 hidden units? If so, why?)
      ii. compare performance of using validation set to not using it in testIrisNoisy

2. Programming assignment: Implement the back propagation algorithm for a feed forward artificial neural network with one hidden layer.
   (a) Your implementation should include at least these input parameters:
      i. number of hidden units
      ii. learning rate
      iii. momentum
      iv. stopping criterion (e.g. number of iterations)
   (b) Test your implementation with the following data sets:
      i. Identity (on course web site)
      ii. Tennis (same as HW2)
      iii. Iris (same as HW2)
   (c) For each of the following experiments, provide a script/program/function (using parameter values you found are appropriate) for running the test:
      i. testIdentity: output accuracy on training set and hidden values for each input (similar to Figure 4.7) using 3 and 4 hidden units
      ii. testTennis: output accuracy on training and test sets.
      iii. testIris: output accuracy on training and test sets.
      iv. testIrisNoisy: corrupt 0% to 20% of class labels, with 2% increment, in the training set (similar to HW2); for each level of noise, output accuracy on the uncorrupted test set; use a validation set and not use a validation set (optionally use weight decay)
   (d) For discrete input/output attributes, you might want to have a pre-processor to convert them to 1-of-n representation.
   (e) The same program should be able to handle the different data sets.
   (f) Use C (GNU gcc), C++ (GNU g++), Java (Oracle Java), LISP (CLISP), or Python. If you don't have a preference, use Java since it's more portable.
   (g) Your program preferrably runs on code.fit.edu (linux).
   (h) Submission:
      i. README.txt: how to compile and run the four tests on code.fit.edu
      ii. source code