```
CS578 Fall 2002
Empirical Methods in Machine Learning and Data Mining
Homework Assignment #3
Due: Thursday, November 21, 2002

The goal of this assignment is to experiment with artificial neural
nets trained with backpropagation, early stopping, and 5-fold cross
validation.  For this assignment use the same data set used in HW2.
The data is still available from the web page if you want to get a
clean copy.  The goal is to predict the same boolean variable (col 1)
from the 143 inputs (cols 2-144).

You may implement backprop yourself, or use a commercial/public domain
implementation.  Note that it probably will take more time to install,
learn to use, and modify someone else's implementation than to program
backprop yourself, so we encourage you to code backprop yourself.  In
fact, implementing bp yourself counts as extra credit.  If you decide
to use someone else's package, it is up to you to make sure that it
will support the experiments needed for this assignment.  One public
domain package you might want to consider that runs on a variety of
platforms is SNNS: Stuttgart Neural Network Simulator.  There also
is a Matlab toolbox for neural nets that is supposed to be pretty
good.

EXPERIMENTS:

 0: Scale each attribute so that the min value of the attribute is
    is 0 and the max value is 1: new_val = (val-min)/(max-min).
    Your code from HW2 might help you here.

 1: For neural nets, you need train sets (backprop sets), early
    stopping sets (technically still part of the train set), and
    test sets.  Use 5-fold cross validation for the train/test
    sets.  The early stopping set should be held out of the train
    set.  One way to do this is to split the data into 5 folds.
    Do backprop on folds 1-3 (3/5 of the train data), use fold
    4 for early stopping (1/4 of the train data), and test on
    fold 5.  Repeat this process 5 times for 5-fold CV. There
    are other ways to do this.  Carefully explain how you choose
    to do 5-fold CV.  A diagram or table would be helpful.

 2: Train fully-connected feedforward neural nets using vanilla
    backpropagation with momentum.  Every backprop implementation
    defines learning rate and momentum somewhat differently, and
    the definitions often vary when using batch mode (updating
    once per epoch -- full pass through the training set) or when
    updating per pattern, so you'll have to experiment with the
    parameter  settings to find values that work well with your code.
    You can use batch mode, per pattern, or per group of pattern
    updating.  (If the nets are fully trained after less than
    100 passes through the train set you're probably training too
    fast.  If the nets are taking more than 10^5 passes through the
    train set you're probably training slower than necessary.)

 3: Compute the accuracy and RMSE on the train, early stopping,
    and test sets. Show graphs of performance vs number of epochs
    for the train and early stopping sets.  The performances on the
    test sets should be reported at the early stopping point.  Is the
    early stopping point for accuracy the same as the early stopping
```

```
   point for RMSE?

4: Do some quick experiments to show how performance varies with the
   train set size.  Plot a learning curve.

5: Experiment with different numbers of hidden units.  You might try
   1,2,4,8,16,32,64,... or even 1,4,16,64,...  What size net yields
   best generalization performance?


EXTRA CREDIT -- do one or more of the following::

 - 5-fold CV leaves you with 5 or more trained neural nets. compare
   the average prediction of these nets with the performance of each
   of the nets alone to see which works better.  to do this experiment
   right you'll need to either use an extra level of cross validation,
   or hold out a final test set. (HINT: if you might do this extra
   credit, hold out the final test set(s) *before* doing the assigned
   experiments above so that you don't have to repeat the runs!)
 - do a study of the effect of altering the learning rate and
   momentum on the generaization performance of the nets
 - try nets with two or more hidden layers.  do they perform better
   than nets with one hidden layer?  are they harder to train?
 - compare weight decay with early stopping.  does one perform better
   than the other?  is one easier to use than the other?
 - do feature selection to find a subset of the features that seems
   to perform better than using all the features
 - do a sensitivity analysis to figure out what inputs the trained
   nets use most.  sensitivity analysis can be done by looking at
   derivatives of the output of the net with respect to the inputs,
   or by experimenting with injecting noise into the inputs one at a
   time
 - take variable type into account when coding inputs to the nets.
 - implement vanilla backprop with momentum for fully-connected
   feedforward neural nets containing one hidden layer and trained
   with squared error

Hand in a brief summary of the results with enough documentation so
that we can see what you did and how you did it.  Do not write a paper
or anything like that.  This is homework, not a class project.  You'll
probably want to use the neural net code later in the class project,
so effort spent now to write good code or become familiar with
whatever package you use should pay off later.

Have fun.
```