Sample Training Sessions

For the type of network we're using, the relevant Matlab command (from the Neural Network Toolbox) are: `feedforwardnet`, which initializes the network `train`, which trains the network, and `sim`, which simulates the network using new data. Here are a couple of sample training sessions using these commands:

Example:

Build a 1-10-1 network to model a dataset that comes with Matlab (shown below).

```
[x,t]=simplefit_dataset;  %Data comes with Matlab
%Plot the data to see what you're building:
plot(x,t);
%Initialize the network using 10 nodes in the hidden layer.
net=feedforwardnet(10);
net=train(net,x,t);
```

At this point, a dialog box comes up to illustrate how the training is proceeding. Once done, you might take a look at some of the plots- In particular, the "Performance" plot and the "Error Histogram" are kind of interesting.

Training Parameters

- Number of hidden layers and number of nodes in each hidden layer.

  The default option is a single hidden layer with 10 neurons:

  ```
  net=feedforwardnet;
  ```

  For example, to construct a 2-3-5-3-2 network:

  ```
  net=feedforwardnet([3 5 3]);
  ```

- The optional second argument is the name of the training function. Among the available options are:

  - `trainlm` is Levenburg-Marquardt (an optimization technique, the default value)
  - `traingdx` is optimization using a variable step gradient descent.
  - `trainbr` is Bayesian regularization (regularization has to do with the error on the testing set)

  The Levenburg-Marquardt is probably the most accurate, but can be slow over large networks. Gradient descent is fast, but is less accurate (more iterations needed).

  For example, to construct a 2-3-2 network using Levenburg-Marquardt, type:

```
net=feedforwardnet(3,'trainlm');
```

Training Performance

We will break up the data into three sets for training: There will be a set for "training", a set for "validation", and a set for "testing". Recall that the error will go down on the training set- However, we run the risk of training too much. That means that the network will perform terribly on new data.

To avoid this, we use a "validation" set to measure the performance of the training- The error will be strictly decreasing on the training set, but if we begin to "overtrain", the error on the validation set will begin to increase.

To see the error on the three datasets, after training, look for the "Plots" section of the dialog box, and choose "Performance".

Alternatively, obtain training data during training: `[net,tr]=train(net,x,t);`

The output `tr` is a Matlab structure that contains a lot of information. To see the plot from the training, you can type:

```
h=length(tr.perf);
plot(1:h,log(tr.perf),1:h,log(tr.vperf),1:h,log(tr.tperf))
```

Example Training

PROBLEM: Build an 8-15-15-2 feedforward neural network that uses `trainrp` for the training method, and uses a training goal of 0.05. Train the data on the Matlab file on our class website, `diabetes1.mat`

SOLUTION:

```
net=feedforwardnet([15 15]);
net.trainParam.goal=0.05;
net.trainFcn='trainrp';
load diabetes1;  %You have to download this
net=train(net,P',T');  %The data needs to be transposed.
```