Date: 15th April, 2014
M.S. Raghavendra Sriram

1000854840

EE 5322 Intelligent Control Systems
Assignment no 4

Neural Networks

Question 1:

1- Consider the following training data set

```
x = [5 4 6  5  2  1  2  1

    0 -1 0 -1 -1 -2 -2 -3]; % inputs

y = [0 0 0 0 1 1 1 1 ]; % Target values
```

a- Train a perceptron to classify the data set into two classes. Plot points and decision boundaries.

b- Use a single neuron with sigmoid activation function to do this classification problem. Plot the points and decision boundary and compare the results to part a.

Solution 1 a :

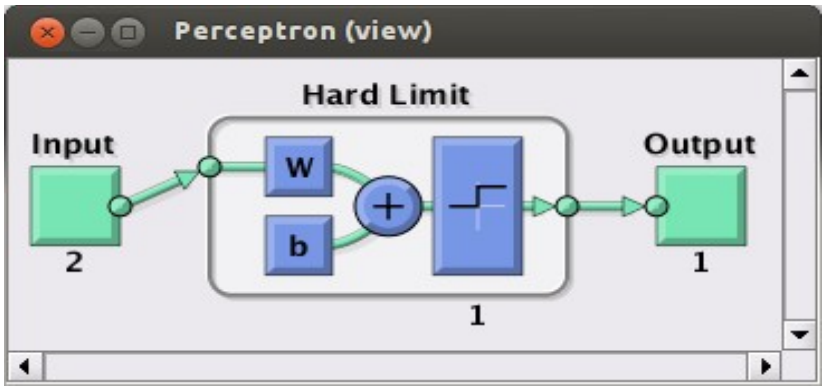
Code :

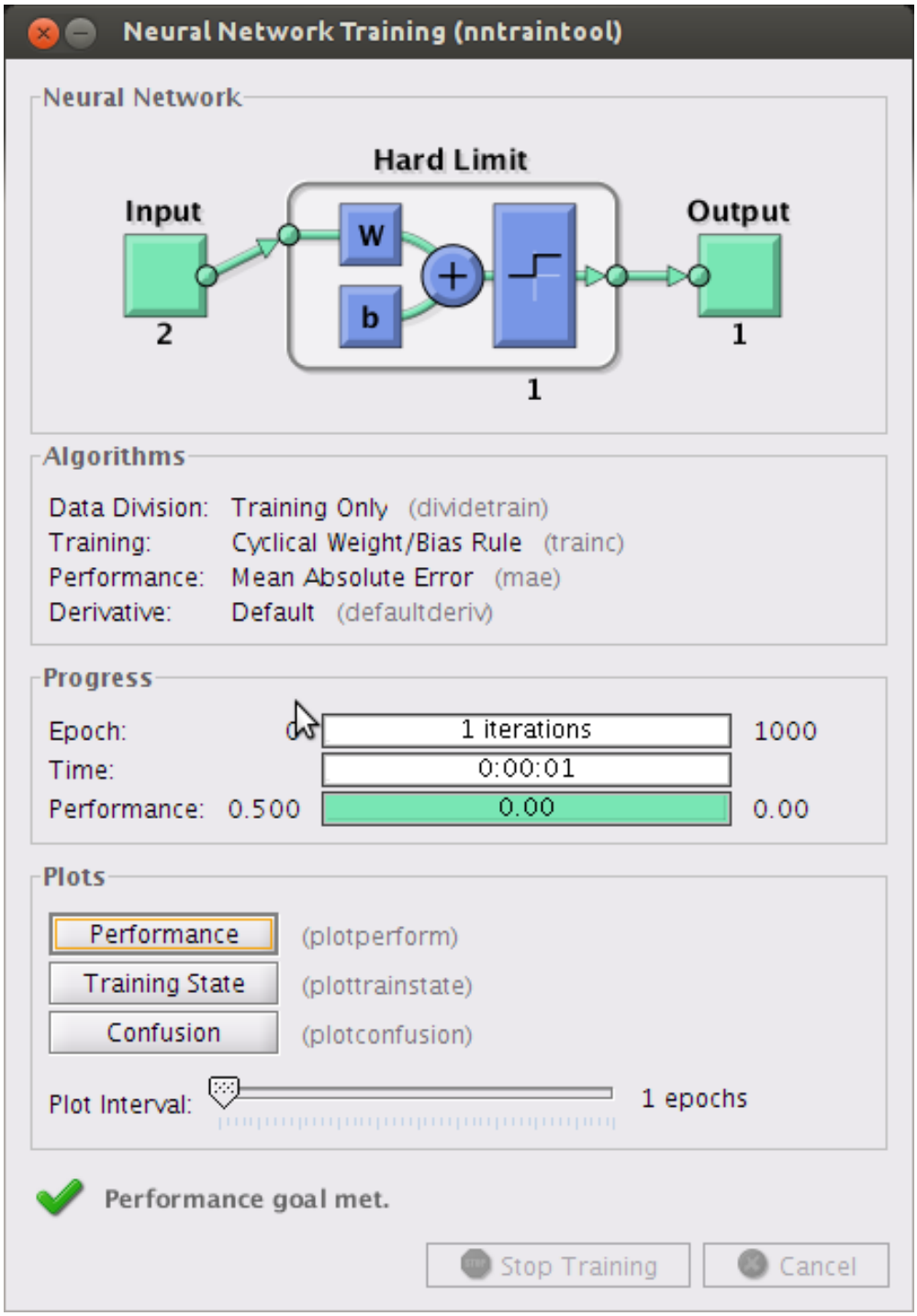
```
close all,
clear all,
clc,
format compact
% number of samples of each class
N = 8;
% define inputs and outputs

x = [5 4 6 5 2 1 2 1
    0 -1 0 -1 -1 -2 -2 -3]; % inputs
y = [0 0 0 0 1 1 1 1 ];
% outputs
% Plot input samples with PLOTPV (Plot perceptron input/target vectors)
figure(1)
plotpv(x,y);
net = perceptron;
net = train(net,x,y);
view(net);
figure(1)
plotpc(net.IW{1},net.b{1});
y = net(x)
```

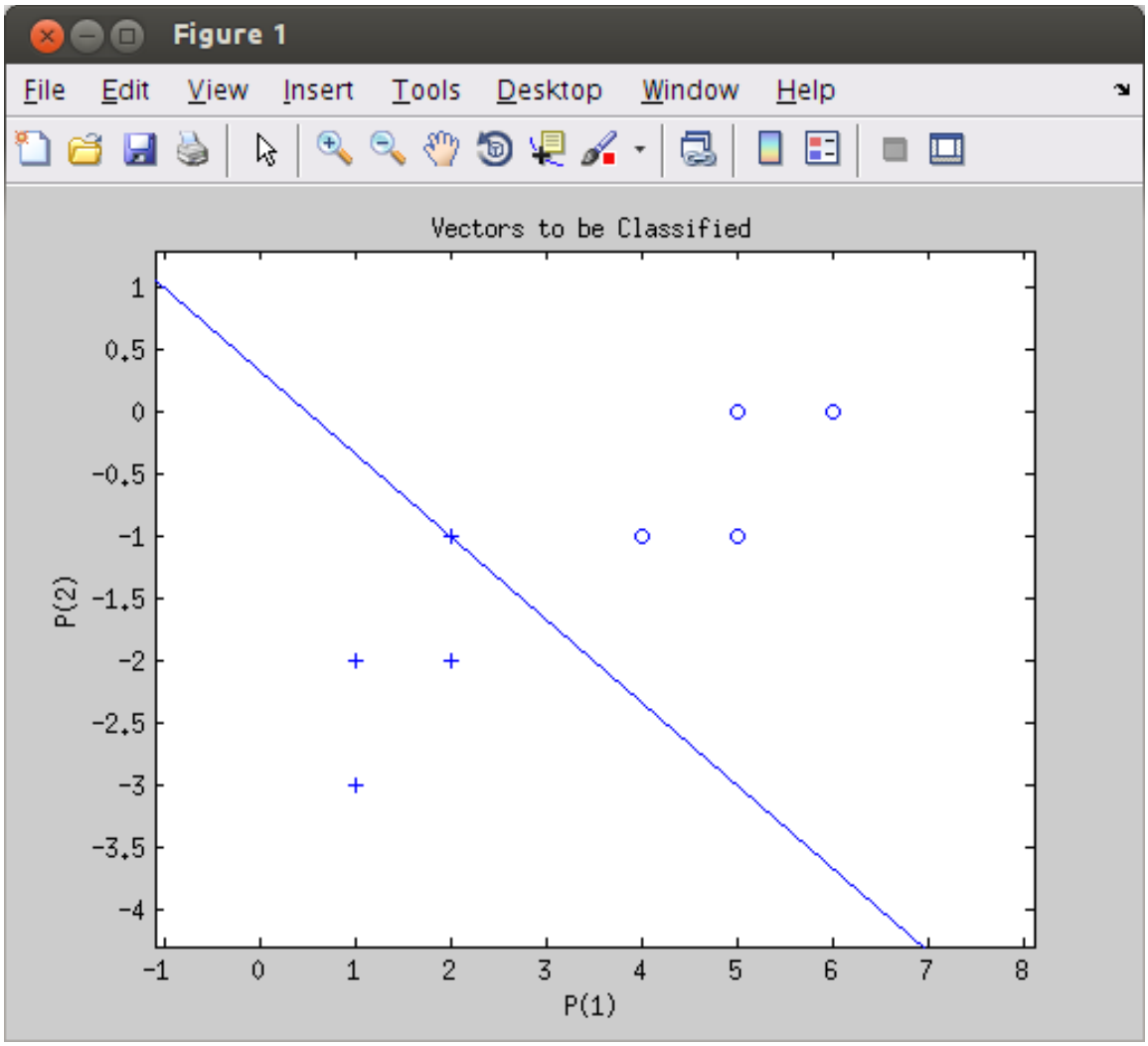
OutPut :

```
y =
    0     0     0     0     1     1     1     1
```

Plots :




Neural Network Training (nntraintool)
Neural Network
Hard Limit
Input
W
b
Output
2
1
1
Algorithms
Data Division: Training Only (dividetrain)
Training: Cyclical Weight/Bias Rule (trainc)
Performance: Mean Absolute Error (mae)
Derivative: Default (defaultderiv)
Progress
Epoch: 1 iterations 1000
Time: 0:00:01
Performance: 0.500 0.00 0.00
Plots
Performance (plotperform)
Training State (plottrainstate)
Confusion (plotconfusion)
Plot Interval: 1 epochs
Performance goal met.
Stop Training
Cancel



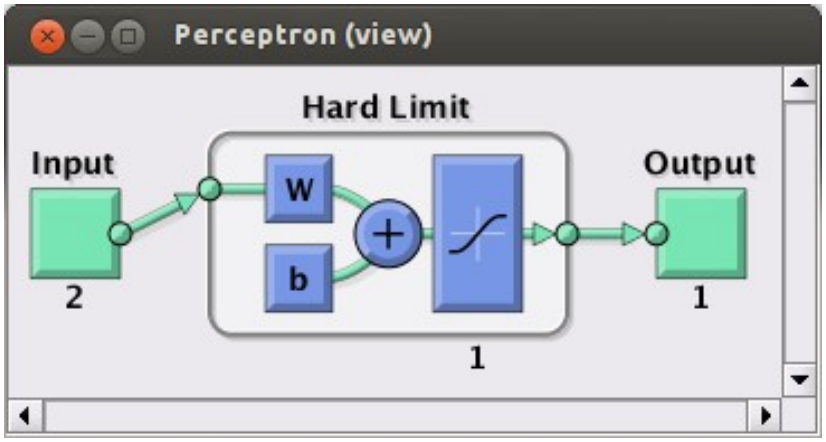
Solution 1 b :

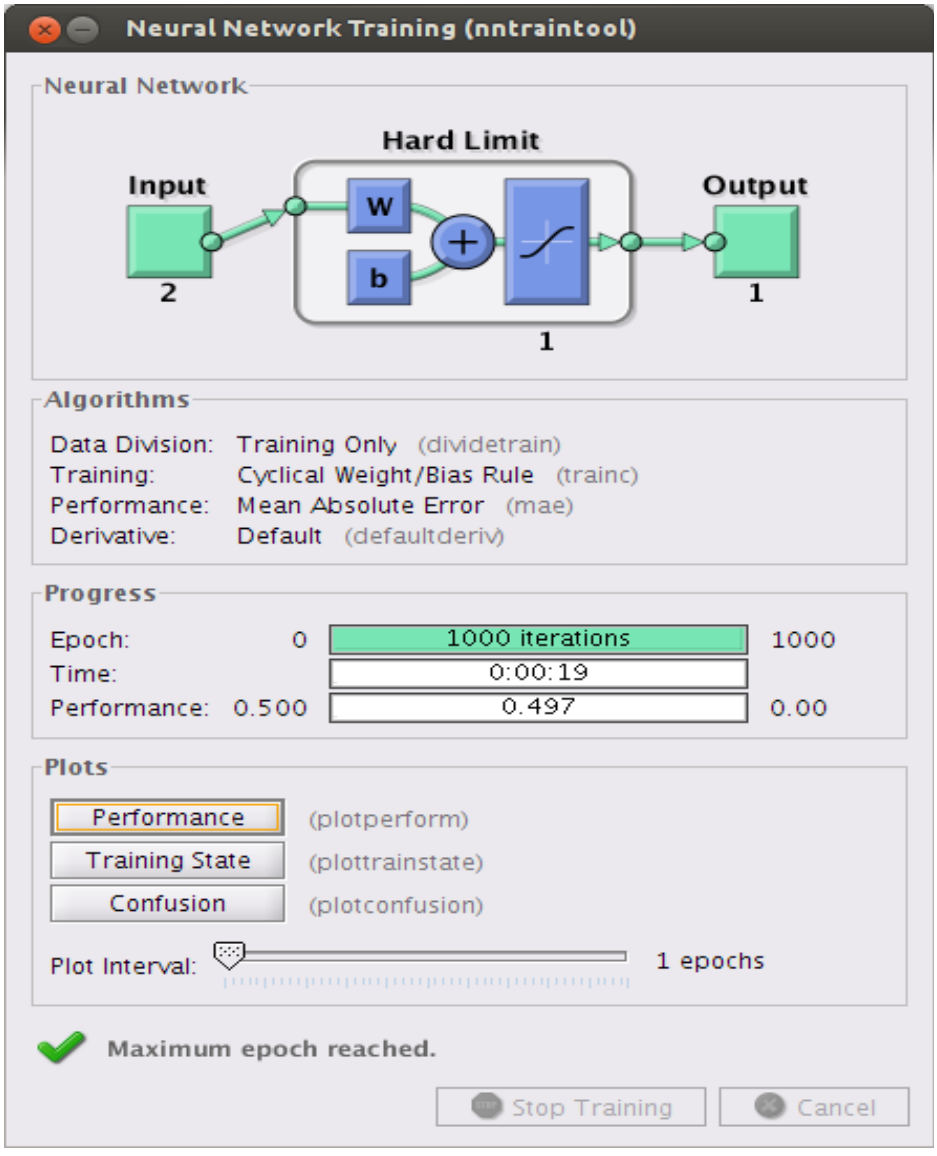
Code :

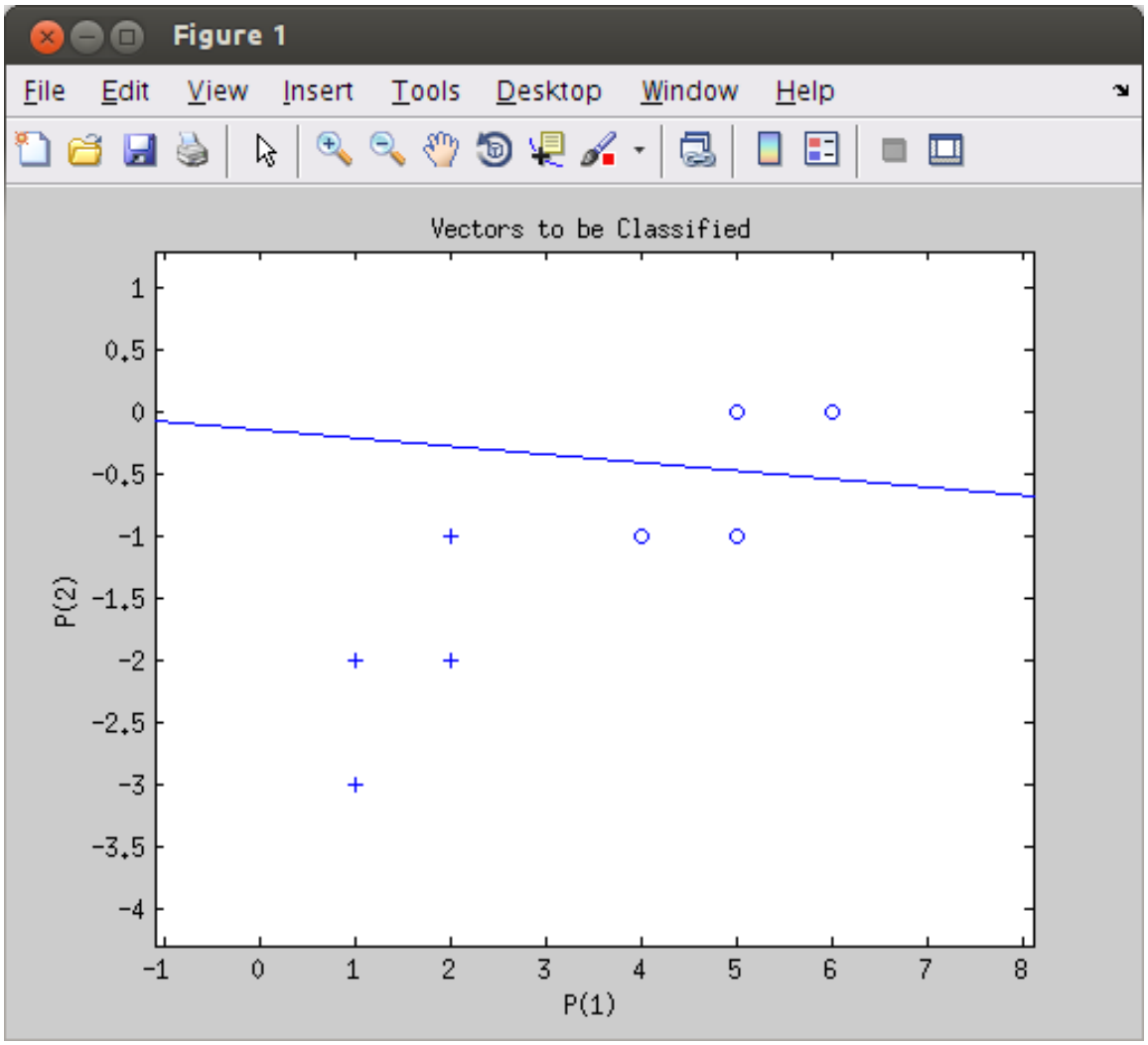
```
close all,
clear all,
clc,
format compact
% number of samples of each class
% define inputs and outputs
x = [5 4 6 5 2 1 2 1
    0 -1 0 -1 -1 -2 -2 -3]; % inputs
y = [0 0 0 0 1 1 1 1 ];
% outputs
% Plot input samples with PLOTPV (Plot perceptron input/target vectors)
figure(1)
plotpv(x,y);
net = perceptron;
net.layers{1}.transferFcn = 'tansig';
net = train(net,x,y);
view(net);
figure(1)
plotpc(net.IW{1},net.b{1});
y = net(x)
```

Output :y =
  Columns 1 through 6
   -0.9889   0.9969   -0.9946   0.9936   0.9993   1.0000
  Columns 7 through 8
    1.0000   1.0000

Plots :







Question 2:

Consider the dynamical system

a)Approximate the function f(x) using an MLP neural network and plot the function and the estimation on the same graph.
b)Simulate the system response for exact f(x) and the approximation. Use different initial conditions. Compare the results.

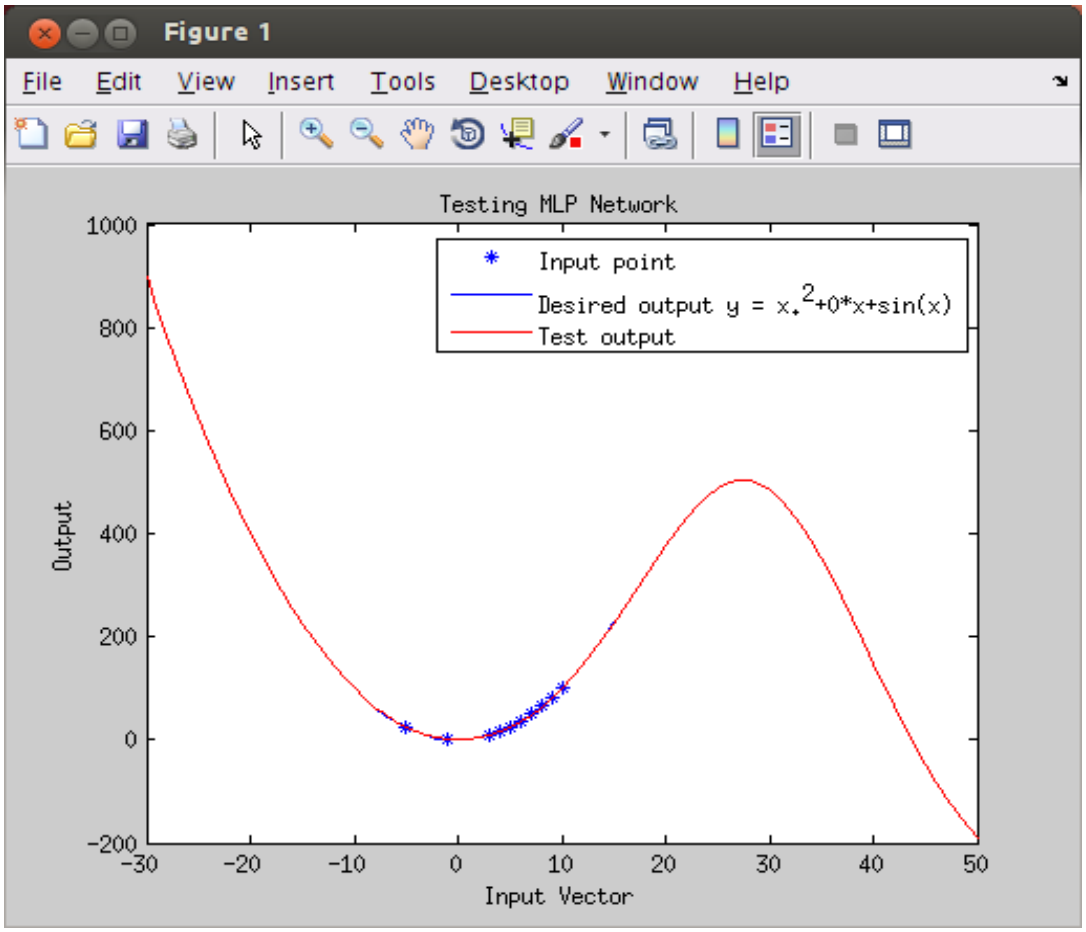
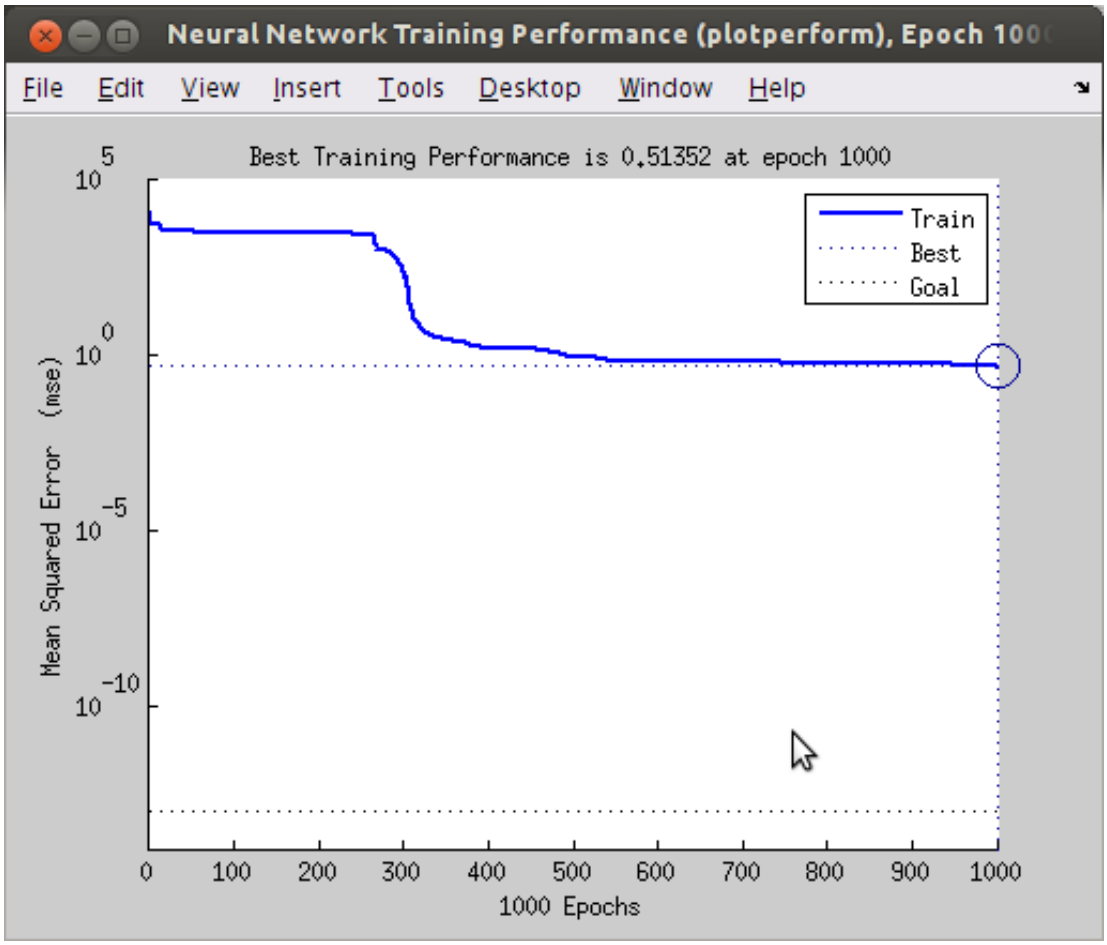
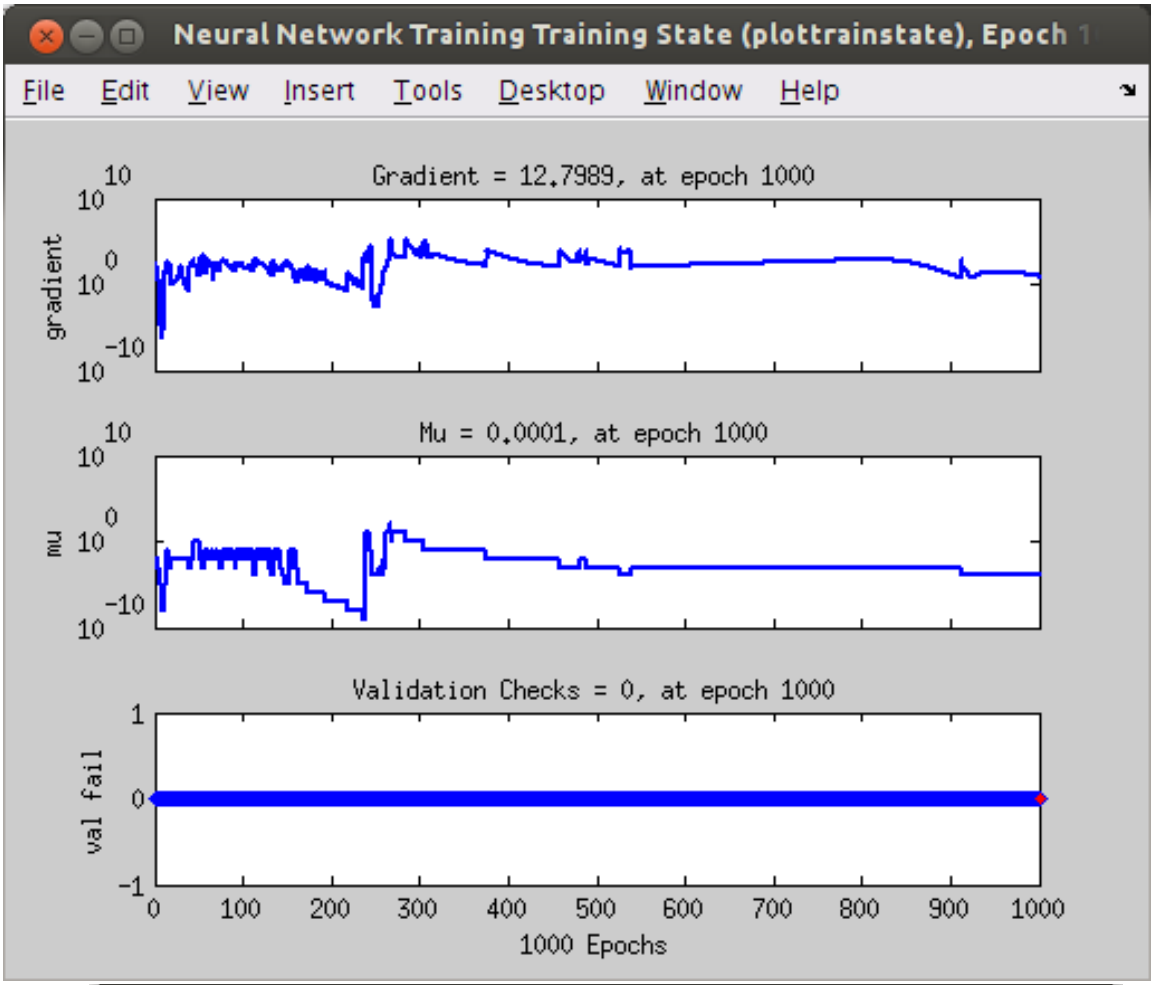
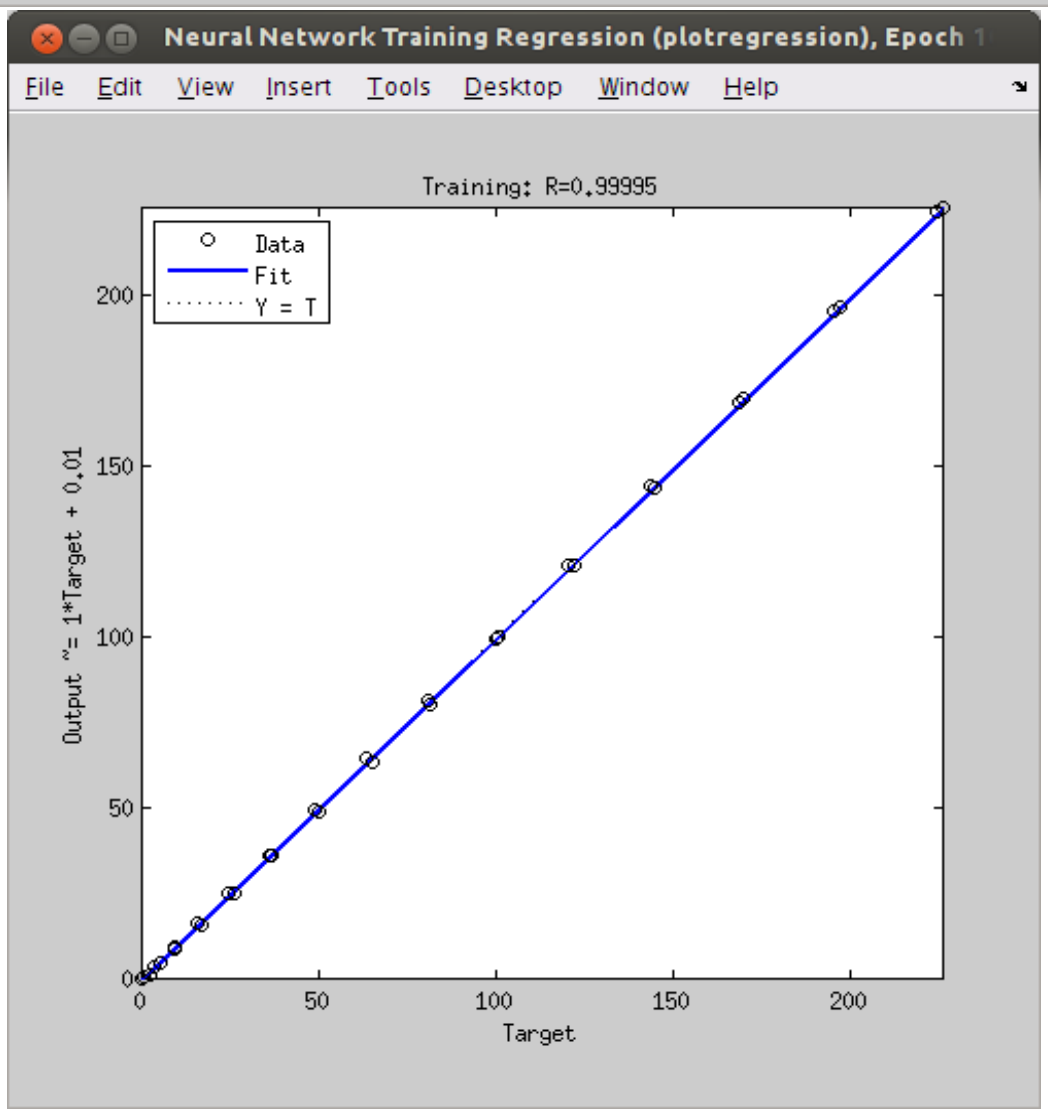


Figure 1
File Edit View Insert Tools Desktop Window Help
Testing MLP Network
Input point
Desired output y = x.^2+0*x+sin(x)
Test output
Output
Input Vector
1000
800
600
400
200
0
-200
-30
-20
-10
0
10
20
30
40
50


Neural Network Training Performance (plotperform), Epoch 100
File Edit View Insert Tools Desktop Window Help
Best Training Performance is 0.51352 at epoch 1000
Train
Best
Goal
Mean Squared Error (mse)
1000 Epochs
10^5
10^0
10^-5
10^-10
0
100
200
300
400
500
600
700
800
900
1000


Neural Network Training Training State (plottrainstate), Epoch 1
File Edit View Insert Tools Desktop Window Help
Gradient = 12.7989, at epoch 1000
gradient
Mu = 0.0001, at epoch 1000
mu
Validation Checks = 0, at epoch 1000
val fail
1000 Epochs


Neural Network Training Regression (plotregression), Epoch 1
File Edit View Insert Tools Desktop Window Help
Training: R=0.99995
Data
Fit
Y = T
Output ~= 1*Target + 0.01
Target

Output :

```
ans =Neural Network
            name: 'Custom Neural Network'
       efficiency: .cacheDelayedInputs, .flattenTime,
                   .memoryReduction
         userdata: (your custom info)
   dimensions:
        numInputs: 1
        numLayers: 4
       numOutputs: 1
   numInputDelays: 0
   numLayerDelays: 0
 numFeedbackDelays: 0
 numWeightElements: 76
       sampleTime: 1
   connections:
      biasConnect: [1; 1; 1; 1]
     inputConnect: [1; 0; 0; 0]
     layerConnect: [4x4 boolean]
    outputConnect: [0 0 0 1]
   subobjects:
          inputs: {1x1 cell array of 1 input}
          layers: {4x1 cell array of 4 layers}
         outputs: {1x4 cell array of 1 output}
          biases: {4x1 cell array of 4 biases}
     inputWeights: {4x1 cell array of 1 weight}
     layerWeights: {4x4 cell array of 3 weights}
```

Question 3:

Use a multilayer perceptron to classify the data set into two classes and Compute the false-positive rate and false-negative rate of your classification results.

Code:

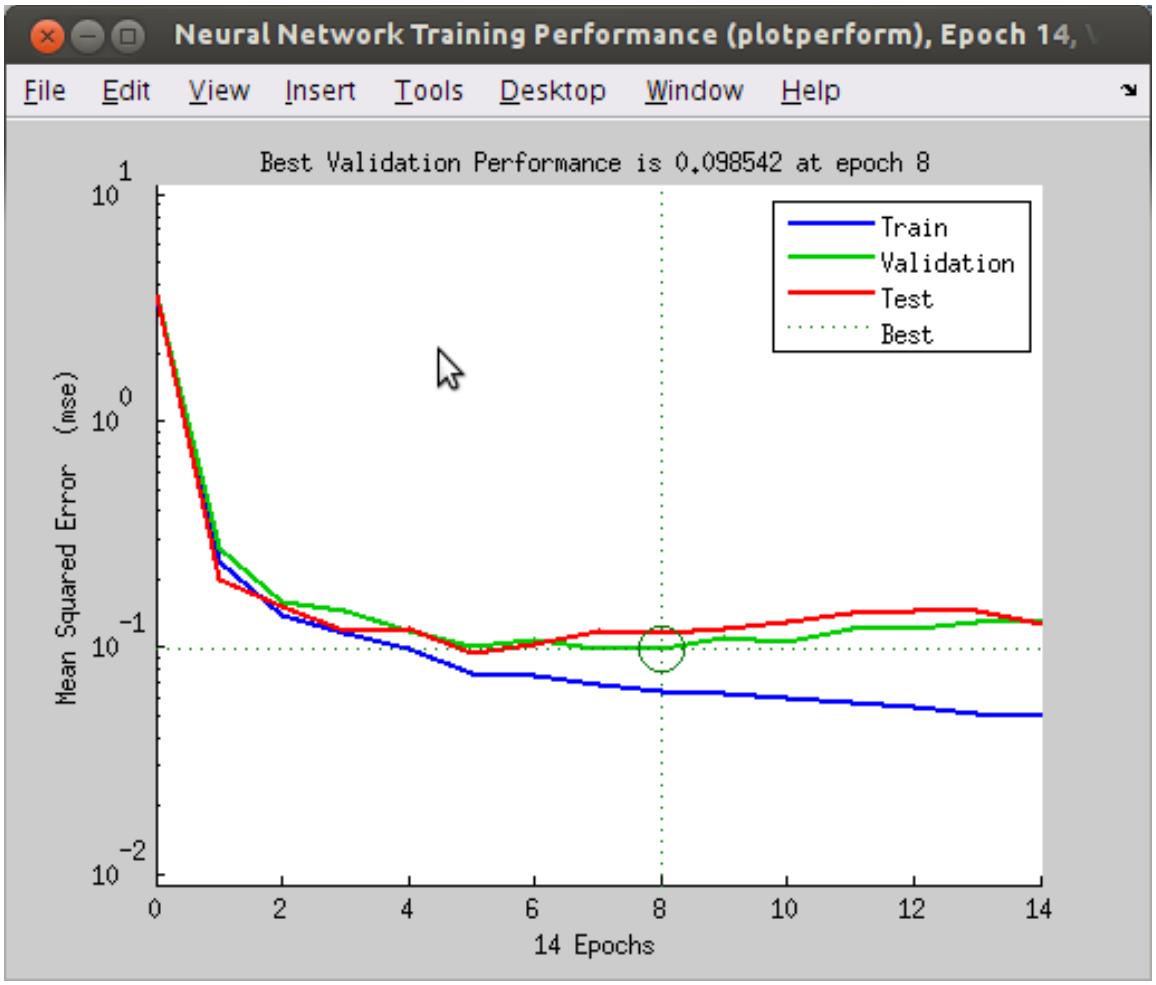
```
clc;
clear all;
close all;
%data classification using MLP
data=open('breastcancer.mat');
Input=data.data(:,2:10)'; %Input matrix
Target=data.data(:,11)'; %Target values
benign=0;
mali=0;
for i=1:length(Target)
    if Target(i)==2
        Target(i)=1; %Benign cell
        benign=benign+1;
    else
        Target(i)=-1; %Malignant cell
        malignant=malignant+1;
    end
end
nnet=feedforwardnet(10);
nnet.divideParam.trainRatio = 70/100;
nnet.divideParam.valRatio = 15/100;
nnet.divideParam.testRatio = 15/100;
```

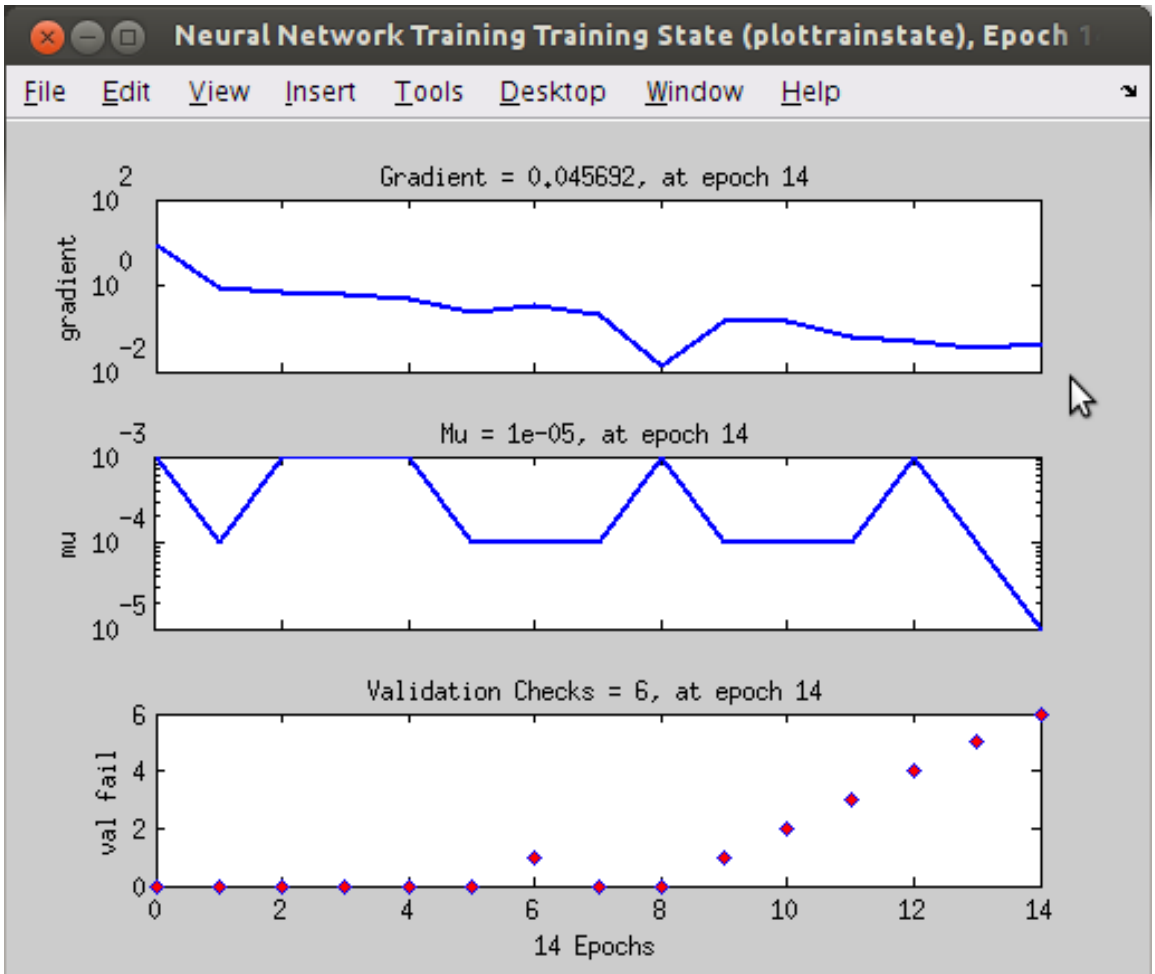
```
nnet=train(nnet,Input,Target);
output=hardlims(sim(nnet,Input));
wrongMalign=0;
wrongBenign=0;
for i=1:length(Target)
    if Target(i)==1 && output(i)==-1
        wrongMalign=wrongMalign+1;
    end
    if Target(i)==-1 && output(i)==1
        wrongBenign=wrongBenign+1;
    end
end
FalsePositive=wrongMalign/benign
FalseNegative=wrongBenign/malignant
```

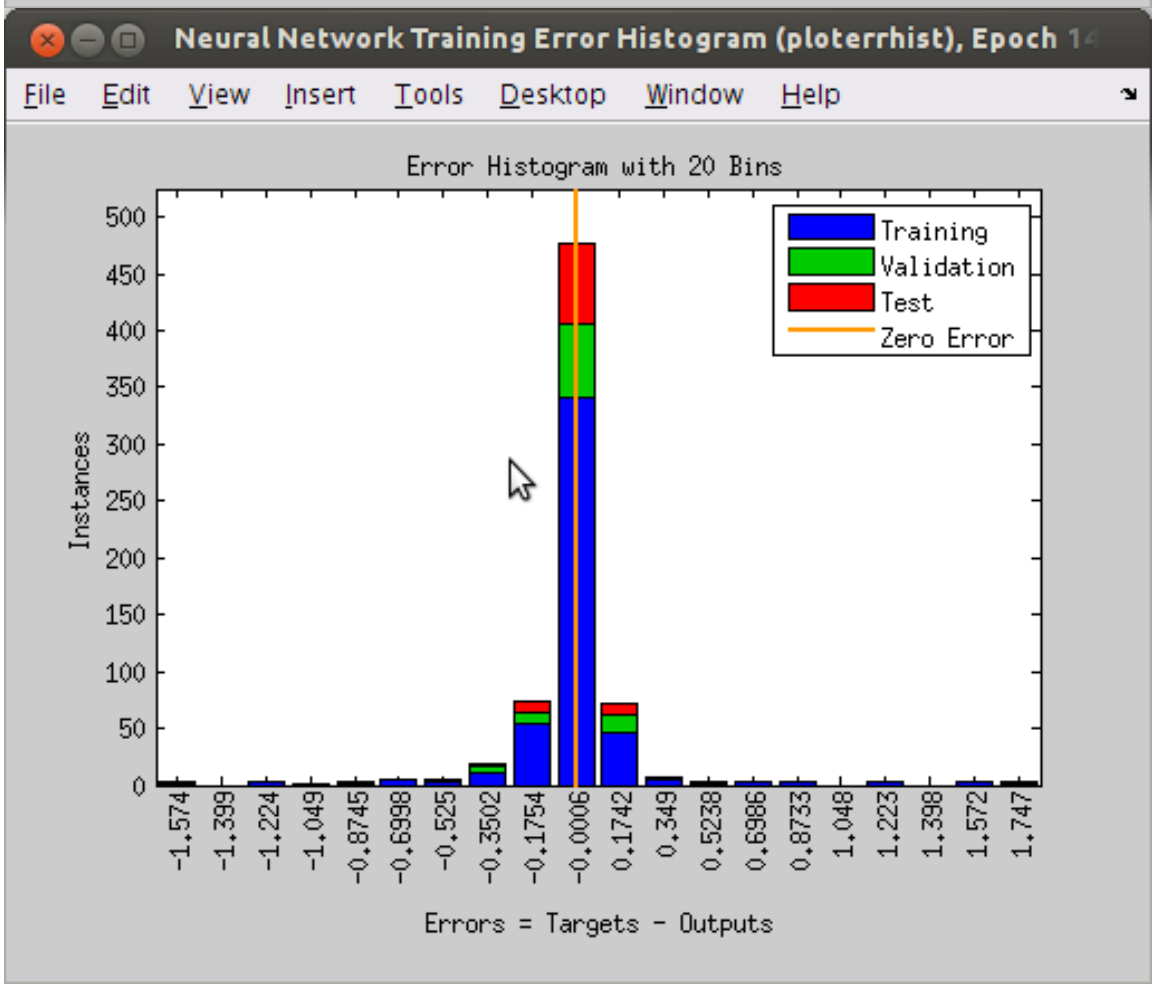
Output :

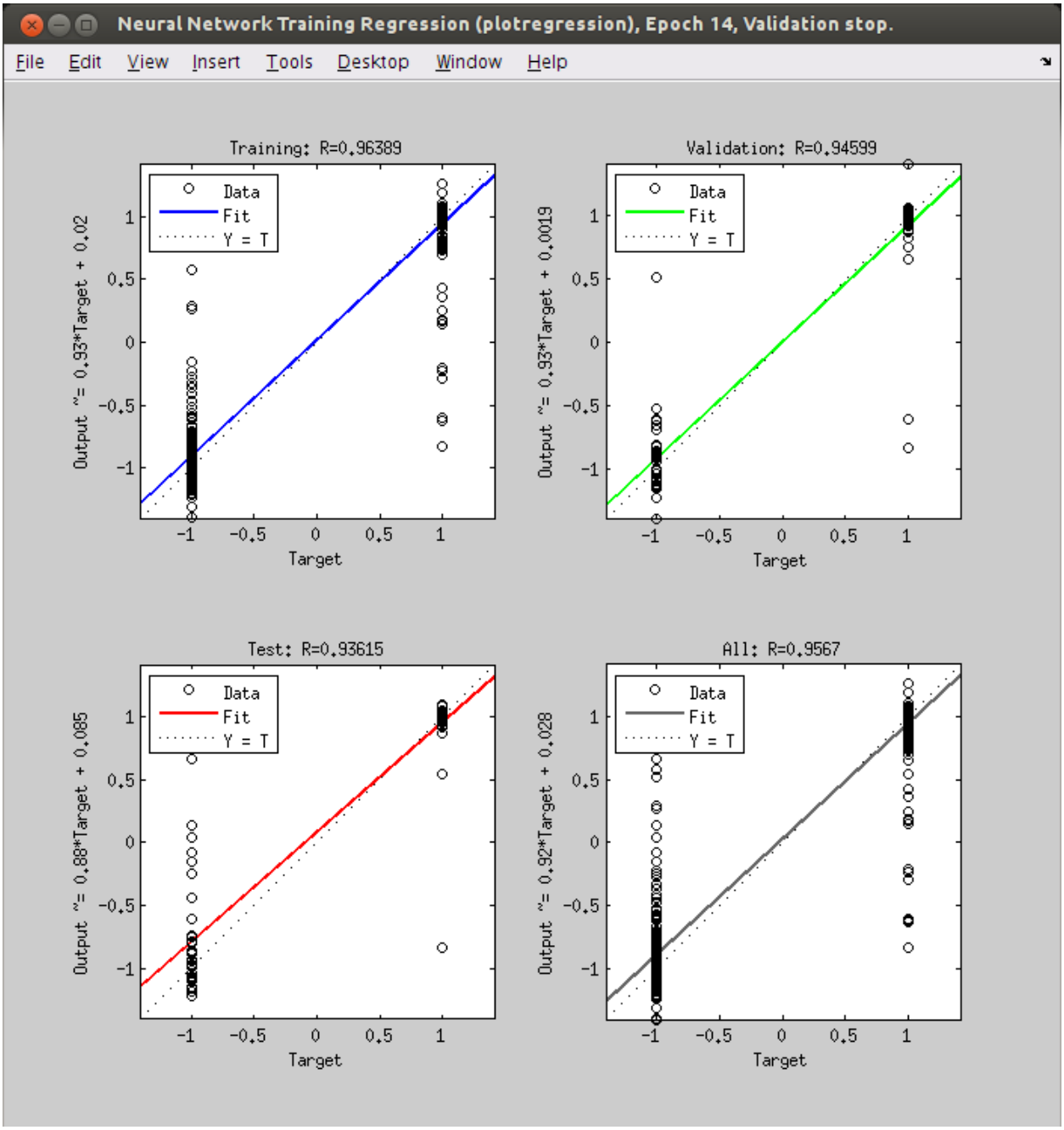
```
FalsePositive =
   0.0203
FalseNegative =
   0.0293
```

Plot:



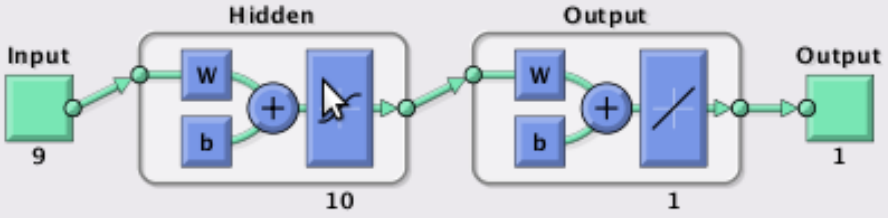

Neural Network Training Training State (plottrainstate), Epoch 1
File Edit View Insert Tools Desktop Window Help
Gradient = 0.045692, at epoch 14
gradient
Mu = 1e-05, at epoch 14
mu
Validation Checks = 6, at epoch 14
val fail
14 Epochs


Neural Network Training Error Histogram (ploterrhist), Epoch 14
File Edit View Insert Tools Desktop Window Help
Error Histogram with 20 Bins
Training
Validation
Test
Zero Error
Instances
Errors = Targets - Outputs


Neural Network Training Regression (plotregression), Epoch 14, Validation stop.
File Edit View Insert Tools Desktop Window Help
Training: R=0.96389
Output ~= 0.93*Target + 0.02
Data
Fit
Y = T
Target
Validation: R=0.94599
Output ~= 0.93*Target + 0.0019
Test: R=0.93615
Output ~= 0.88*Target + 0.085
All: R=0.9567
Output ~= 0.92*Target + 0.028

Neural Network Training (nntraintool)

Neural Network



Algorithms

Data Division:	Random	(dividerand)
Training:	Levenberg-Marquardt	(trainlm)
Performance:	Mean Squared Error	(mse)
Derivative:	Default	(defaultderiv)

Progress

Epoch:	0	14 iterations	1000
Time:		0:00:00	
Performance:	3.49	0.0513	0.00
Gradient:	8.90	0.0457	1.00e-07
Mu:	0.00100	1.00e-05	1.00e+10
Validation Checks:	0	6	6

Plots

Performance	(plotperform)
Training State	(plottrainstate)
Error Histogram	(ploterrhist)
Regression	(plotregression)

Plot Interval: 1 epochs

Opening Regression Plot

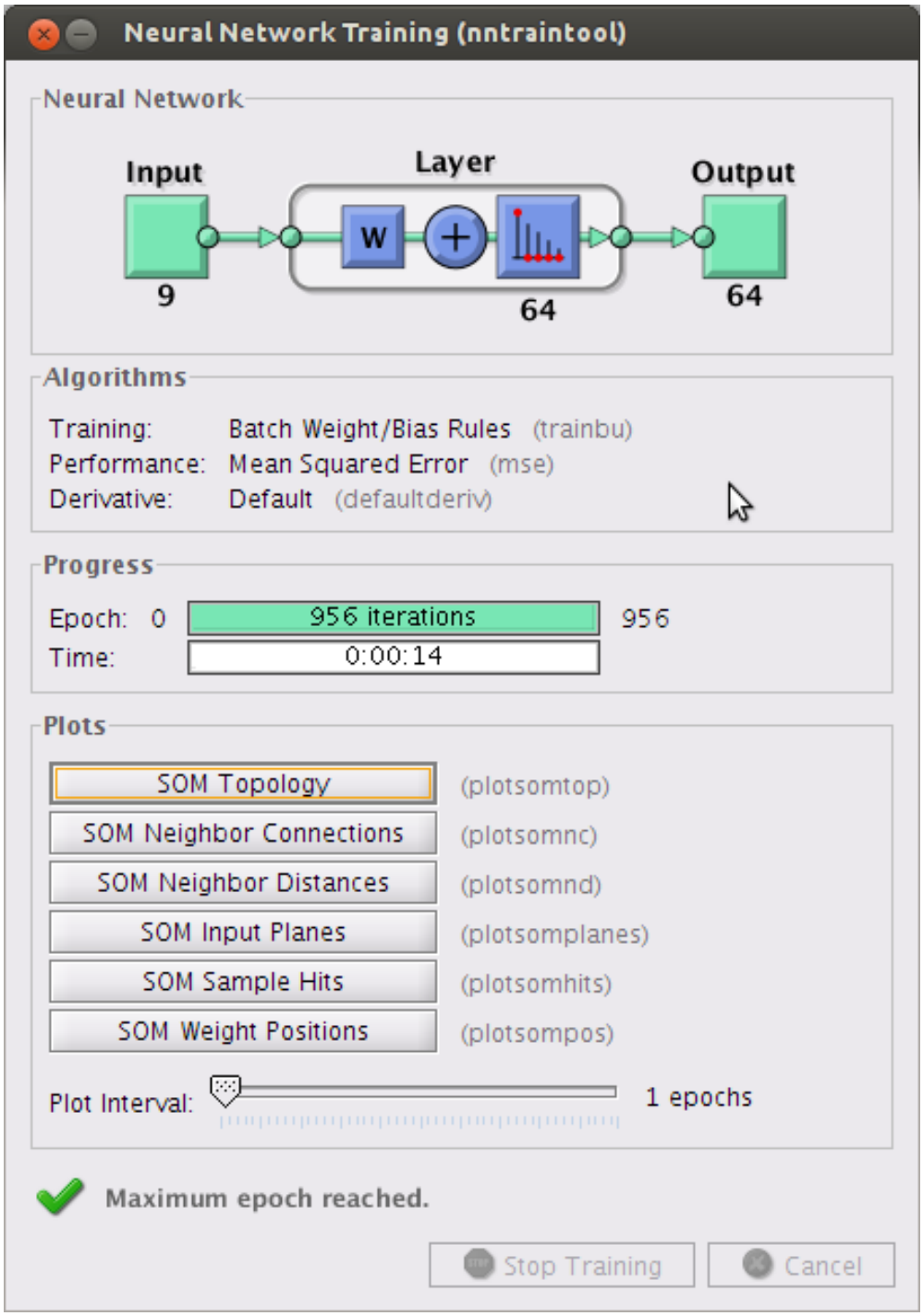
Stop Training | Cancel

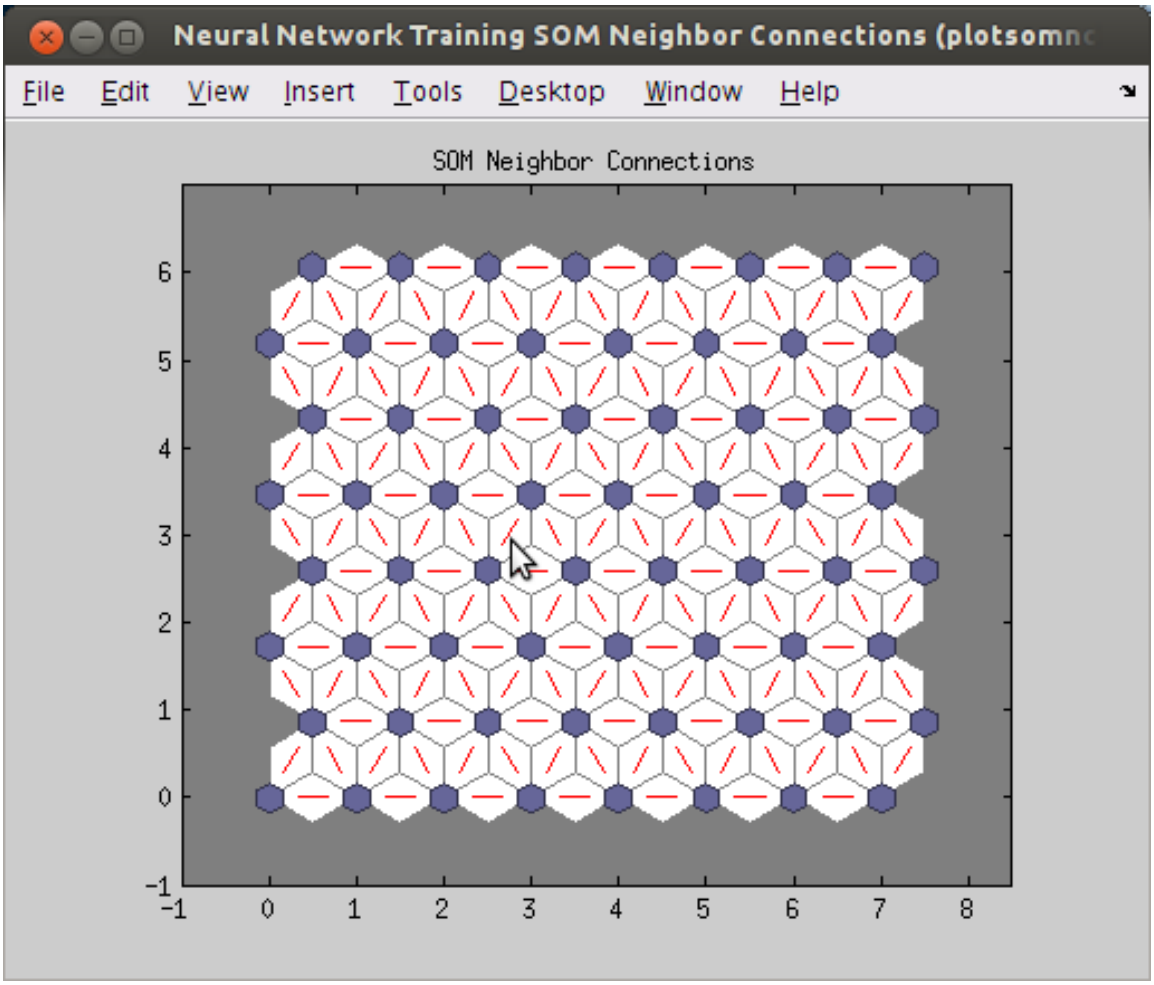
Solution 3b:

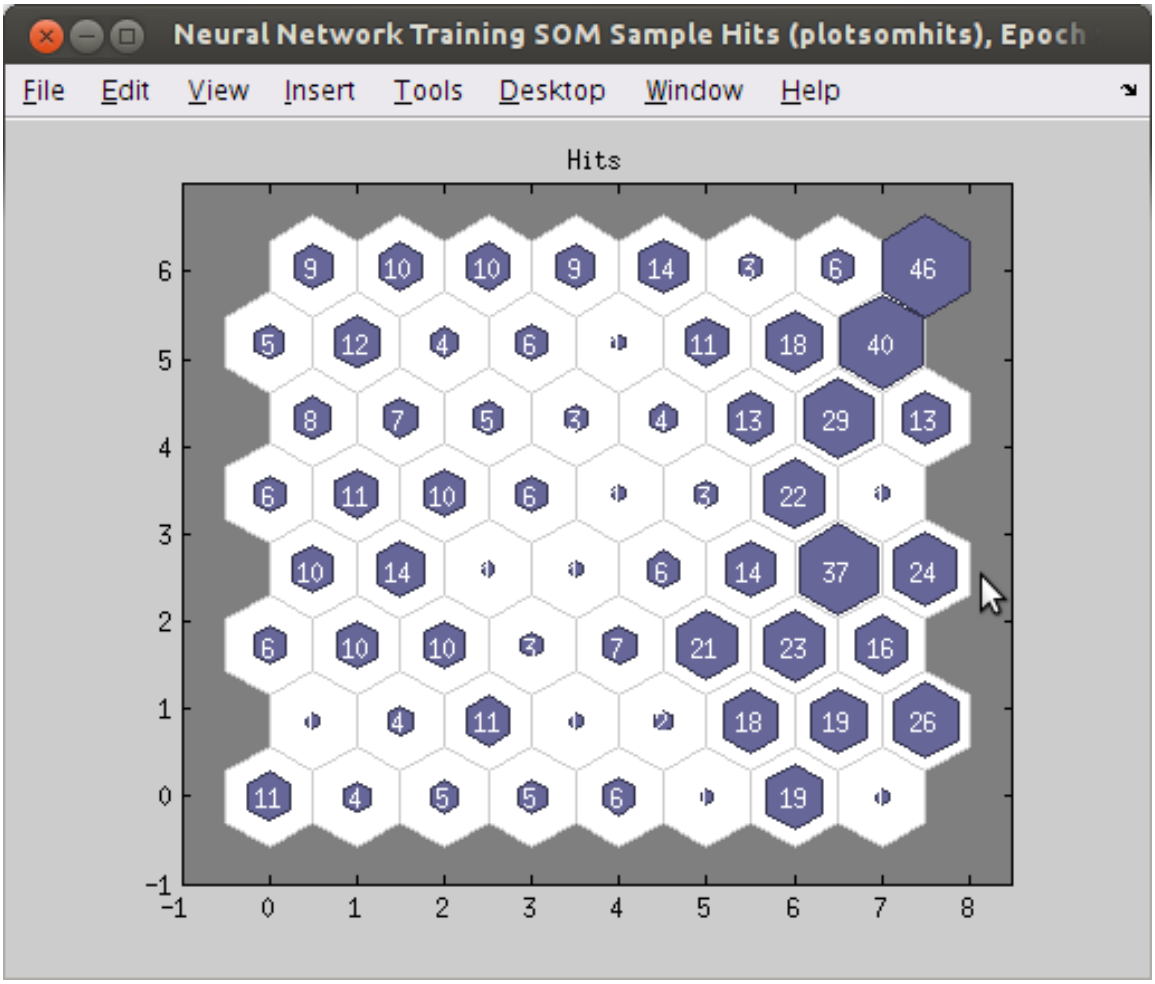
Code :

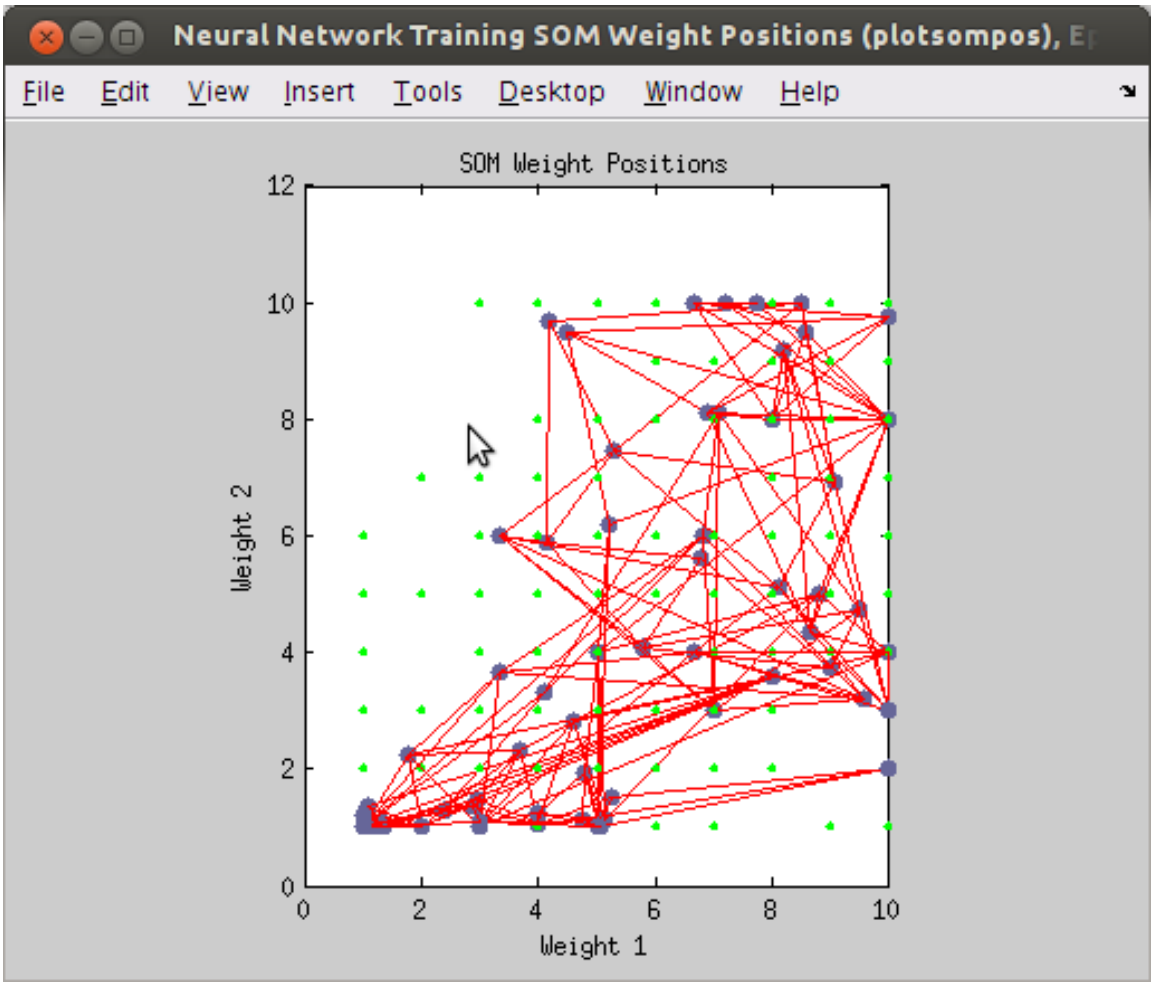
```
clc;
clear all;
close all;
%Classification using Self-Organizing Map
data=open('breastcancer.mat');
P=data.data(:,2:10)'; %Input matrix
T=data.data(:,11)'; %Target values
benign=0;
malignant=0;
for i=1:length(T)
    if T(i)==2
        %T(i)=1; %Benign cell
        benign=benign+1;
    else
        %T(i)=0; %Malignant cell
        malignant=malignant+1;
    end
end
SOM=selforgmap([8 8],478);
SOM=train(SOM,P);
output=SOM(P);
classes=vec2ind(output);
```

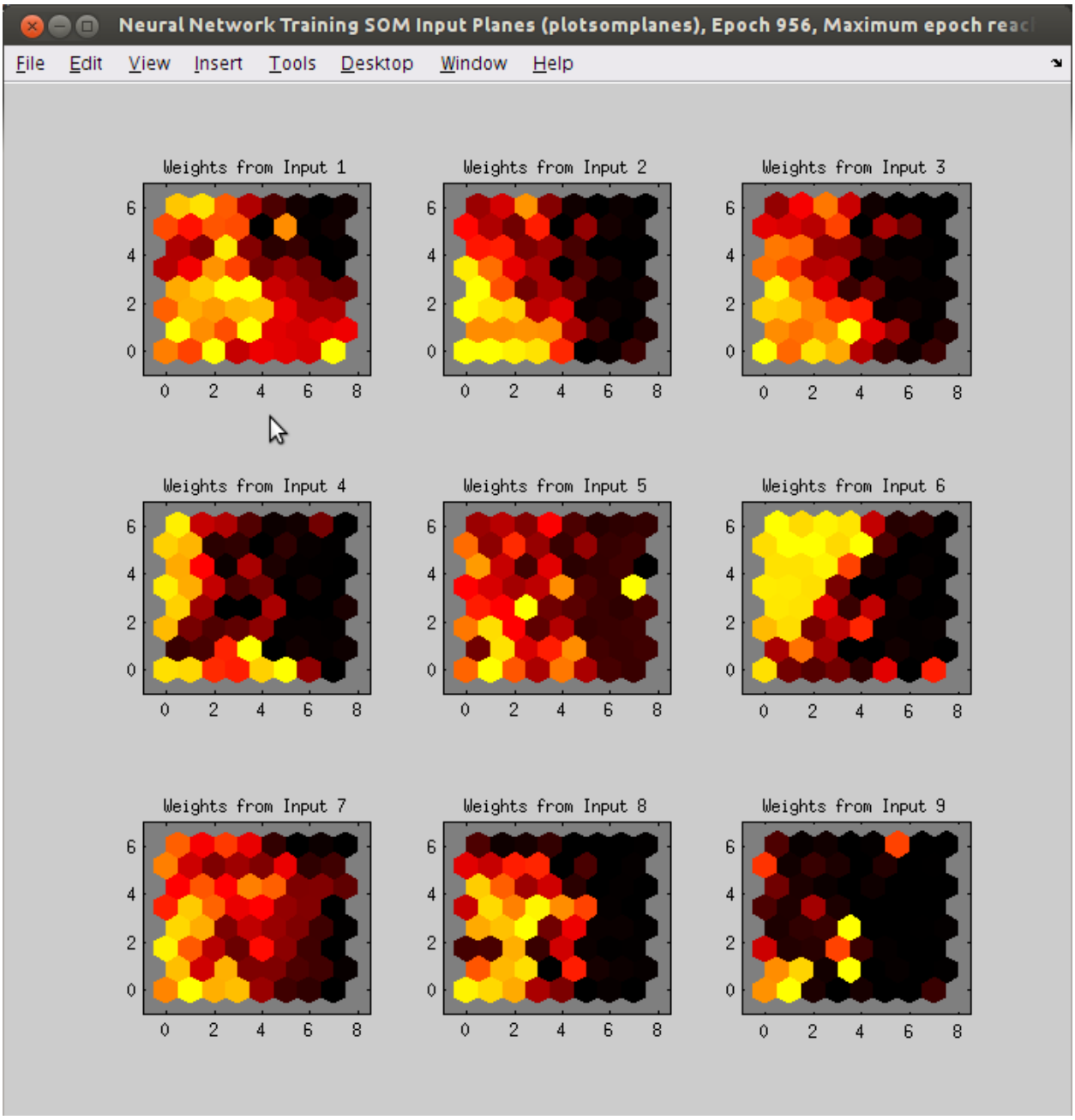
Output:


Neural Network Training (nntraintool)
Neural Network
Input
Layer
Output
W
9
64
64
Algorithms
Training: Batch Weight/Bias Rules (trainbu)
Performance: Mean Squared Error (mse)
Derivative: Default (defaultderiv)
Progress
Epoch: 0
956 iterations
956
Time:
0:00:14
Plots
SOM Topology
(plotsomtop)
SOM Neighbor Connections
(plotsomnc)
SOM Neighbor Distances
(plotsomnd)
SOM Input Planes
(plotsomplanes)
SOM Sample Hits
(plotsomhits)
SOM Weight Positions
(plotsompos)
Plot Interval:
1 epochs
Maximum epoch reached.
Stop Training
Cancel


Neural Network Training SOM Neighbor Connections (plotsomnc
File Edit View Insert Tools Desktop Window Help
SOM Neighbor Connections
6
5
4
3
2
1
0
-1
-1 0 1 2 3 4 5 6 7 8


Neural Network Training SOM Sample Hits (plotsomhits), Epoch
File Edit View Insert Tools Desktop Window Help
Hits
9 10 10 9 14 3 6 46
5 12 4 6 11 18 40
8 7 5 3 4 13 29 13
6 11 10 6 3 22
10 14 6 14 37 24
6 10 10 3 7 21 23 16
4 11 2 18 19 26
11 4 5 5 6 19
6
5
4
3
2
1
0
-1
-1 0 1 2 3 4 5 6 7 8


Neural Network Training SOM Weight Positions (plotsompos), E
File Edit View Insert Tools Desktop Window Help
SOM Weight Positions
Weight 2
Weight 1
0
2
4
6
8
10
12


Neural Network Training SOM Input Planes (plotsomplanes), Epoch 956, Maximum epoch reac
File Edit View Insert Tools Desktop Window Help
Weights from Input 1
Weights from Input 2
Weights from Input 3
Weights from Input 4
Weights from Input 5
Weights from Input 6
Weights from Input 7
Weights from Input 8
Weights from Input 9
0 2 4 6 8
0 2 4 6