Table of Contents

ECE 532 - Fall 2017 - HW7

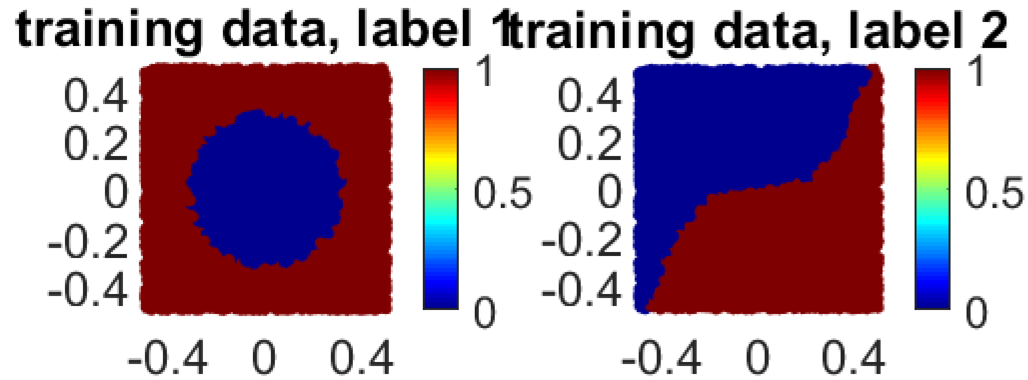
by Roumen Guha

```
clear
close all
```

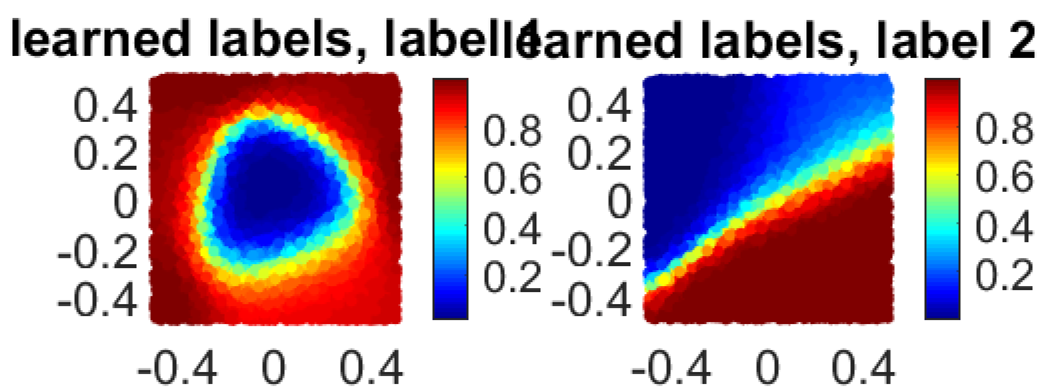
(1) Training Neural Networks.

```
p = 2;
n = 1e4;

% generate training data
X = rand(n, p) - 0.5;
Y1 = sum(X .^2, 2) > 0.1;
Y2 = 5*X(:,1).^3 > X(:,2);
Y = [Y1 Y2];

figure(1); clf;

subplot(121);
scatter(X(:,1), X(:,2), 20, Y1, 'filled');
title('training data, label 1');
axis image; colorbar; colormap jet; set(gca, 'fontsize', 18)

subplot(122);
scatter(X(:,1), X(:,2), 20, Y2, 'filled');
title('training data, label 2');
axis image; colorbar; colormap jet; set(gca, 'fontsize', 18)

Xb = [ones(n, 1) X];
q = size(Y, 2);
M = 3;
V = randn(M + 1, q);
W = randn(p + 1, M);

alpha = 0.1;

for epoch = 1:10
    ind = randperm(n);
    for i = ind
```

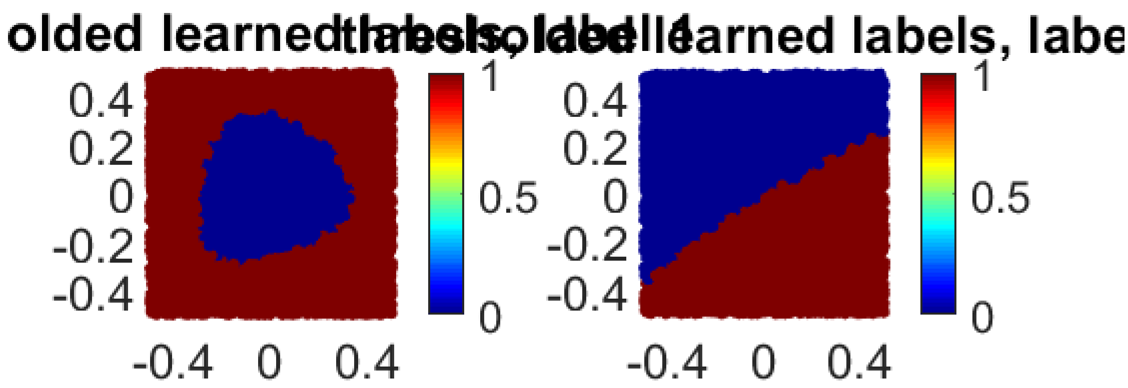
```
        % forward prop
        H = logsig([1 Xb(i ,:)*W]); % 1 x M+1
        Yhat = logsig(H*V); % 1 x q

        % backprop
        delta = (Yhat - Y(i,:)) .* Yhat .* (1 - Yhat); % 1 x q
        Vnew = V - alpha * H' * delta ;

        gamma = (delta * V(2:end,:)') .* H(2:end) .* (1 - H(2:end)); %
 1 x M
        Wnew = W - alpha * Xb(i,:)' * gamma;

        V = Vnew;
        W = Wnew;
    end
end

% final predicted labels
H = logsig([ones(n,1) Xb*W]); % n x M+1
Yhat = logsig(H*V); % n x q

figure(2); clf;

subplot(121); scatter(X(:,1), X(:,2), 20, Yhat(:,1), 'filled');
title('learned labels, label 1');
axis image; colorbar; colormap jet; set(gca, 'fontsize', 18)

subplot(122); scatter(X(:,1), X(:,2), 20, Yhat(:,2), 'filled');
title('learned labels, label 2');
axis image; colorbar; colormap jet; set(gca, 'fontsize', 18)

figure(3); clf;

subplot(121); scatter(X(:,1), X(:,2), 20, 1*(Yhat(:,1) >
 0.5), 'filled');
title('thresholded learned labels, label 1');
axis image; colorbar; colormap jet; set(gca, 'fontsize', 18)

subplot(122); scatter(X(:,1), X(:,2), 20, 1*(Yhat(:,2) >
 0.5), 'filled');
title ('thresholded learned labels, label 2');
axis image; colorbar; colormap jet; set(gca, 'fontsize', 18)
```



training data, label 1
training data, label 2
0.4
0.2
0
-0.2
-0.4
-0.4
0
0.4
1
0.5
0



learned labels, label 1
learned labels, label 2
0.4
0.2
0
-0.2
-0.4
-0.4
0
0.4
0.8
0.6
0.4
0.2



(a) Run the code. How does it perform?

Are the learned labels close to the original lables?

```
% Mediocre at best. It's great that it was able to perform this well seeing
% as it came from random data.
```

(b) Why do we use Xb instead of X? What if we use X instead?

```
% We use Xb to create an offset in the thesholded learned labels for feature 1.
% If we use X instead, the range of intensities for feature 1 is the same as
% that of feature 2. But if we use Xb, the intensities are in the correct range,
% as they are for the original data.
```

(c) Explain the use of the "2"s in the expression for gamma.

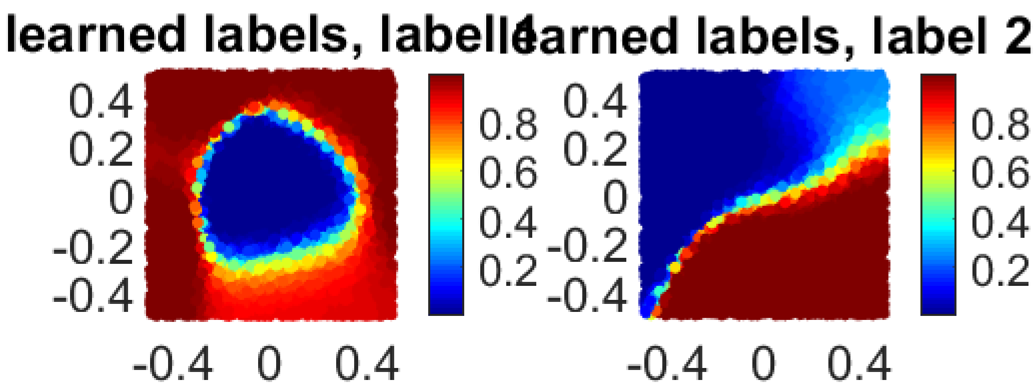
```
% We exclude the row of ones because they aren't used to calculate the
 weights,
% they simply serve to correct the range of outputs.
```

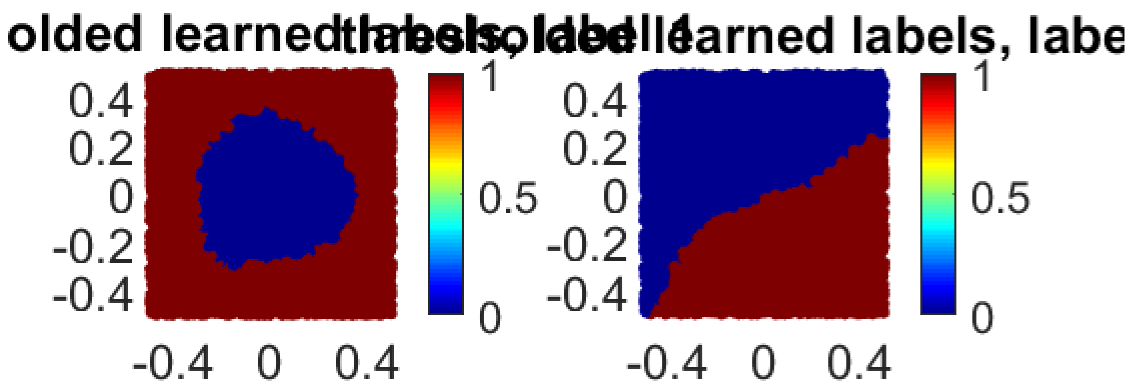
(d) Try increasing the number of epochs to 100.

What effect does this have?

```
for epoch = 1:100
    ind = randperm(n);
    for i = ind
        % forward prop
        H = logsig([1 Xb(i ,:)*W]); % 1 x M+1
        Yhat = logsig(H*V); % 1 x q

        % backprop
        delta = (Yhat - Y(i,:)) .* Yhat .* (1 - Yhat); % 1 x q
        Vnew = V - alpha * H' * delta ;

        gamma = (delta * V(2:end,:)') .* H(2:end) .* (1 - H(2:end)); %
 1 x M
        Wnew = W - alpha * Xb(i,:)' * gamma;

        V = Vnew;
        W = Wnew;
    end
end

% final predicted labels
H = logsig([ones(n,1) Xb*W]); % n x M+1
Yhat = logsig(H*V); % n x q

figure(2); clf;

subplot(121); scatter(X(:,1), X(:,2), 20, Yhat(:,1), 'filled');
title('learned labels, label 1');
axis image; colorbar; colormap jet; set(gca, 'fontsize', 18)

subplot(122); scatter(X(:,1), X(:,2), 20, Yhat(:,2), 'filled');
title('learned labels, label 2');
axis image; colorbar; colormap jet; set(gca, 'fontsize', 18)

figure(3); clf;

subplot(121); scatter(X(:,1), X(:,2), 20, 1*(Yhat(:,1) >
 0.5), 'filled');
title('thresholded learned labels, label 1');
axis image; colorbar; colormap jet; set(gca, 'fontsize', 18)
```

```
subplot(122); scatter(X(:,1), X(:,2), 20, 1*(Yhat(:,2) >
 0.5), 'filled');
title ('thresholded learned labels, label 2');
axis image; colorbar; colormap jet; set(gca, 'fontsize', 18)

% The thresholded learned label 2 didn't change at all, but the
 thresholded
% learned label 1 converged to the boundary of the circle from the
 positive
% side. It performs marginally better.
```





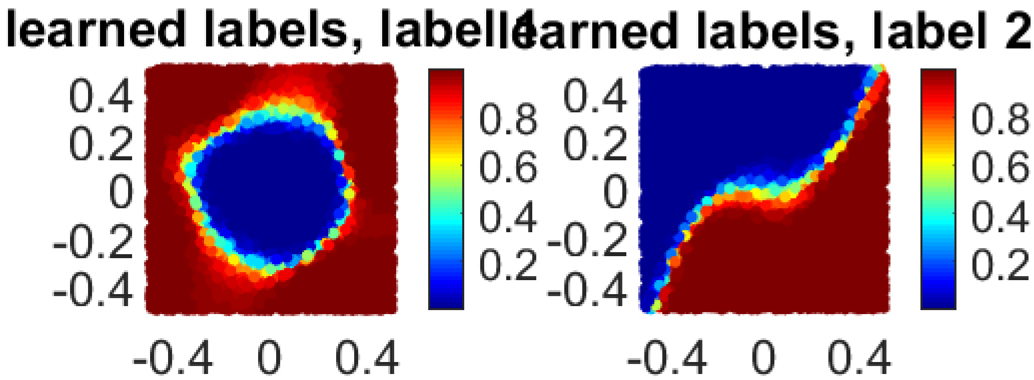
(e) Try increasing the number of hidden nodes to 3; what happens?

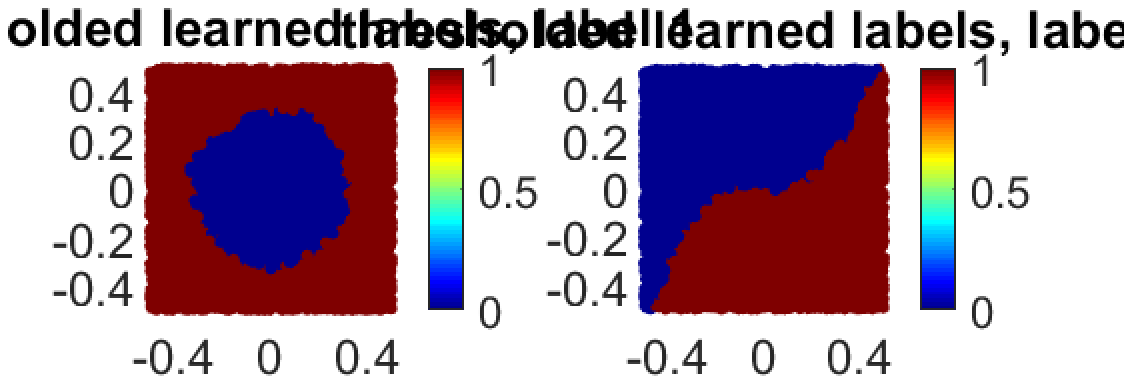
What happens if you use 4 hidden nodes? Can you explain why four hidden nodes performs so much differently from two?

```
M = 4;
V = randn(M + 1, q);
W = randn(p + 1, M);

alpha = 0.1;

for epoch = 1:100
    ind = randperm(n);
    for i = ind
        % forward prop
        H = logsig([1 Xb(i ,:)*W]); % 1 x M+1
        Yhat = logsig(H*V); % 1 x q

        % backprop
        delta = (Yhat - Y(i,:)) .* Yhat .* (1 - Yhat); % 1 x q
        Vnew = V - alpha * H' * delta ;

        gamma = (delta * V(2:end,:)') .* H(2:end) .* (1 - H(2:end)); %
 1 x M
```

```
        Wnew = W - alpha * Xb(i,:)' * gamma;

        V = Vnew;
        W = Wnew;
    end
end

% final predicted labels
H = logsig([ones(n,1) Xb*W]); % n x M+1
Yhat = logsig(H*V); % n x q

figure(2); clf;

subplot(121); scatter(X(:,1), X(:,2), 20, Yhat(:,1), 'filled');
title('learned labels, label 1');
axis image; colorbar; colormap jet; set(gca, 'fontsize', 18)

subplot(122); scatter(X(:,1), X(:,2), 20, Yhat(:,2), 'filled');
title('learned labels, label 2');
axis image; colorbar; colormap jet; set(gca, 'fontsize', 18)

figure(3); clf;

subplot(121); scatter(X(:,1), X(:,2), 20, (Yhat(:,1) >
 0.5), 'filled');
title('thresholded learned labels, label 1');
axis image; colorbar; colormap jet; set(gca, 'fontsize', 18)

subplot(122); scatter(X(:,1), X(:,2), 20, (Yhat(:,2) >
 0.5), 'filled');
title ('thresholded learned labels, label 2');
axis image; colorbar; colormap jet; set(gca, 'fontsize', 18)

% When we use 3 nodes and 10 epochs, the thresholded learned label 1
 converges
% to the true shape very closely, much better than 2 hidden notes. The
% thresholded learned label 2 is less linear than with 2 nodes, but
 still
% isn't quite there: it looks quadratic in its curvature, but we need
 it to
% look cubic.

% When we use 4 nodes and 10 epochs, the thresholded learned label 2
% resembles the original label 2 better than with 3 nodes, but it
 actually
% gains nothing in resemblance to its thresholded learned label 1
 versus
% the original label, except it is different.

% Having 4 hidden nodes gives more variability in terms of the output
 that
% is achievable. It allows more "space" for the optimization to take
 place.
```

```
% In fact, consider 4 hidden nodes and 100 epochs, as shown above.
 Both
% thresholded learned labels 1 and 2 resemble the original training
 labels
% very closely, that they can barely be distinguished by eye.
```





Published with MATLAB® R2017a