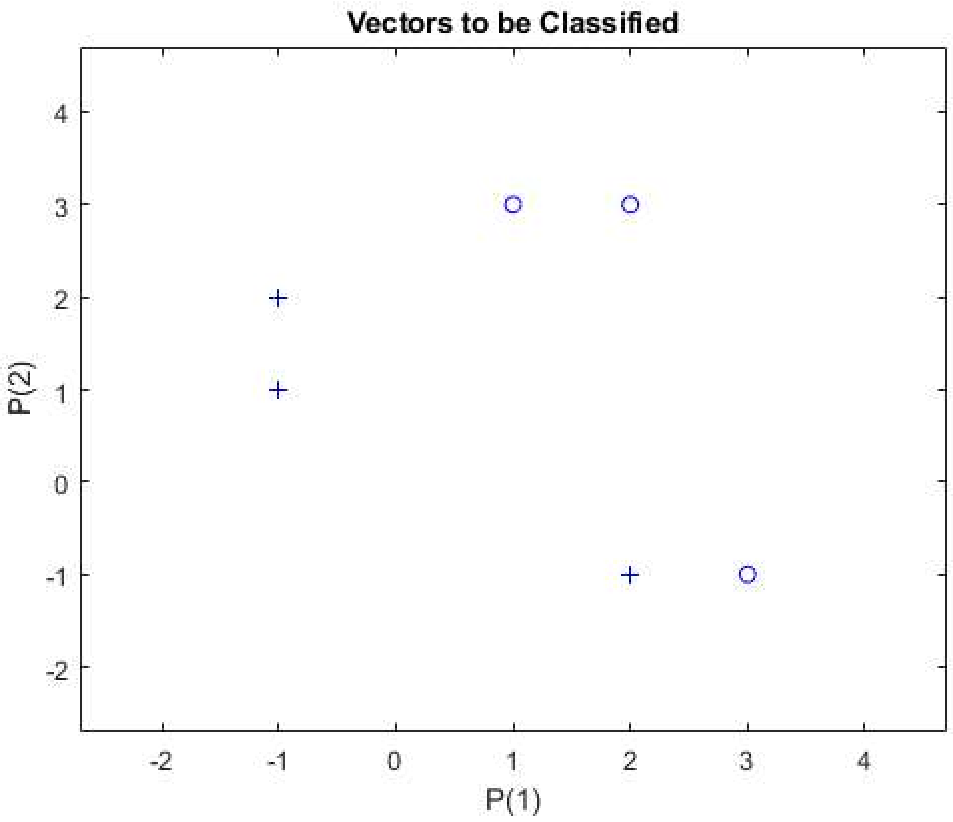
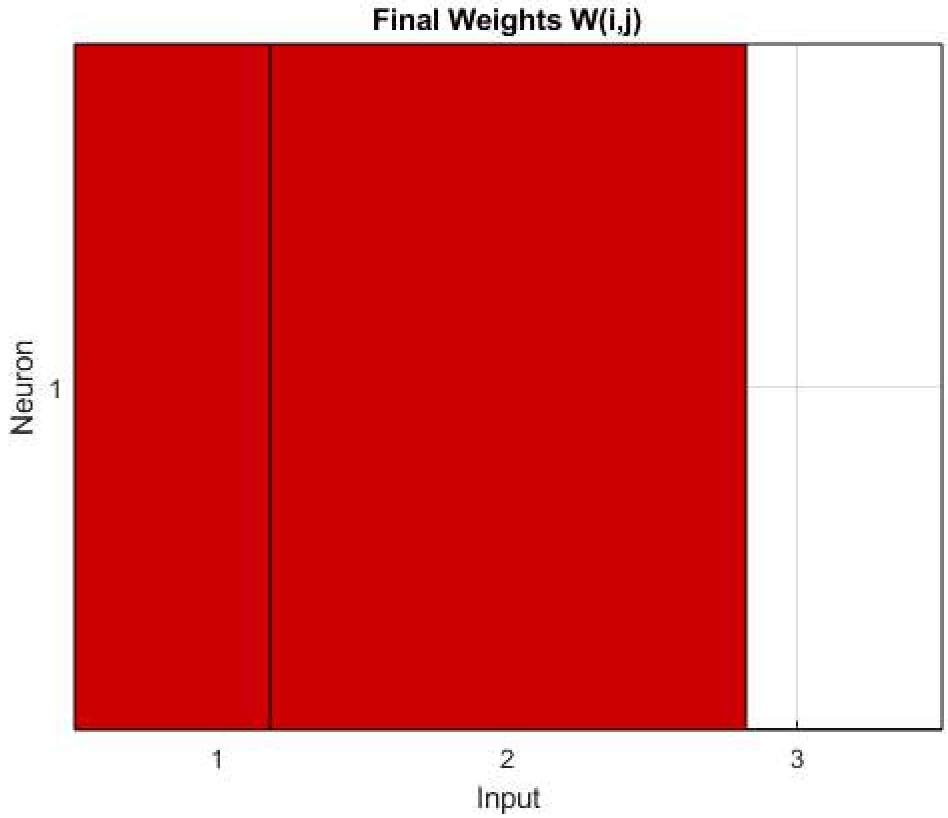
```matlab
%David Melanson
clf reset; clearvars; clc
pausetime = 0;
pauseflag = 1;
maxw = 10;
minw = 0;

%input vectors, bias
P = [
    -1.0 -1.0 2.0 1.0 2.0 3.0;
    1.0 2.0 -1.0 3.0 3.0 -1.0;
    1.0 1.0 1.0 1.0 1.0 1.0
    ];

%target vector
T = [1 1 1 -1 -1 -1];

%Initialize network
%===================================================
[R, Q] = size(P); [S, Q] = size(T);
W0 = zeros(S, R);
B0 = ones(S, 1);

figure(5)
plotpv(P(1:R-1,:), hardlim(T));

% Plot original values
%===================================================
pause(pausetime);
figure(1);
hintonw(W0, maxw, minw);
title('Original Weights W(i,j)');

% TRAIN THE NETWORK
%===================================================
% TRAINING PARAMETERS
disp_freq = 1;
max_epoch = 10;
% lr = 0.1;
% dr = lr/3;
% lp Learning Parameter
% lr Learning Rate
% dr Decay Rate
lr = 1;
lp.lr = lr;
lp.dr = 0;

W = W0;
B = B0;

for epoch = 1:max_epoch
    for q = 1:Q
        % PRESENTATION PHASE
        A = T(:, q);
        % LEARNING PHASE
        dW = learnhd(W, P(:, q), [], [], A, [], [], [], [], [], lp, []);
        W = W +dW;
        if pauseflag == 1
            pause(pausetime)
            figure(1)
        end
    end
    %end loop if solution found
    if (hardlims(W*P) == T)
        break
    end
    % DISPLAY PROGRESS
    if rem(epoch, disp_freq) == 0
        pause(pausetime)
        hintonw(W, maxw, minw)
        title('Weights W(i,j)');
    end
end

% PLOT FINAL VALUES
hintonw(W, maxw, minw);
title('Final Weights W(i,j)');
pause(pausetime);

% SUMMARIZE RESULTS
%===================================================
disp('With inputs of ');
P

disp('and weights of ');
W

disp('The network responds with outputs');
A = hardlims(W*P)
```

```
With inputs of

P =

    -1    -1     2     1     2     3
     1     2    -1     3     3    -1
     1     1     1     1     1     1

and weights of

W =

   -60   -30     0

The network responds with outputs

A =

     1     1    -1    -1    -1    -1
```

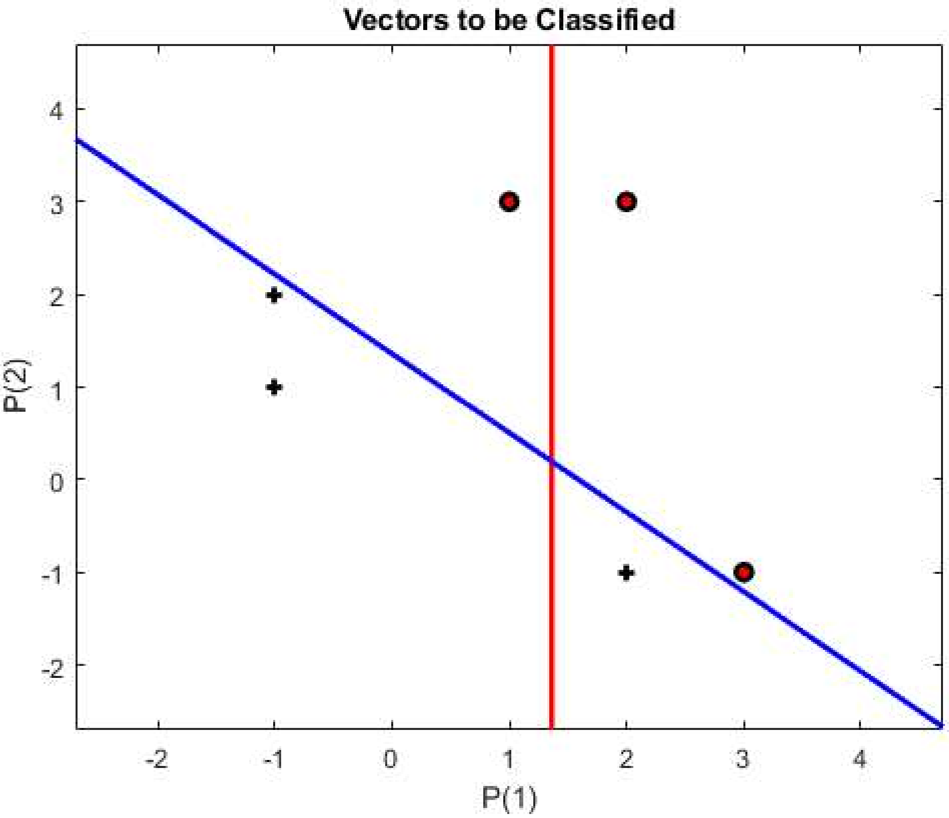




Published with MATLAB® R2022a

```
clear all; clc; cla; clf;
pause_flag = 0;
max_epoch = 200;

%Input vectors
P = [
    -1.0 -1.0 2.0 1.0 2.0 3.0;
    1.0 2.0 -1.0 3.0 3.0 -1.0;
    1.0 1.0 1.0 1.0 1.0 1.0
    ];

%Target vector
T = [1 1 1 0 0 0];

%Input layer
[R, Q] = size(P); [S, Q] = size(T);

%Initialize network parameters
figure(1);
plotpv(P(1:R-1,:), T);
Change_Marker

%Initialize weights randomly
W = rand(S,R);
Wp = W(:, 1:R-1);
Bp = W(:,R);

%display initial values
%The input vectors are replotted
plotpv(P(1:R-1,:), T);

plotpc(Wp, Bp);

watchon;
cla;
plotpv(P(1:R-1,:), T);

pause(3);
figure(1);
E=1;
linehandle = plotpc(Wp, Bp);
%disp('Hit something to continue');

%sum squared error performance function
epoch = 1;
while (sse(E) && (epoch <= max_epoch))
    Ai = hardlim(W*P);
    Ei = T-Ai;
    dWq = learnp(W, P, [], [], [], [], Ei, [], [], [], [], []);
    W = W+dWq;
    Wp = W(:, R-1);
    Bp = W(:, R);
    linehandle = plotpc(Wp, Bp, linehandle);
    lines = findobj(gcf, 'Type', 'Line');
    Change_LineWidth
    Change_Marker
    drawnow;
    if(pause_flag == 1)
        pause(1);
    end
    A = hardlim(W*P);
    %error - Target minus calculated this epoch
    E = T-A;
    %disp(E)
    epoch = epoch +1;
end
watchoff;
disp('Target is ')
T
disp('Solution reached of ')
A
disp('With weights ')
W

testPoint = findobj(gca, 'Type', 'Line');
set(testPoint, 'Color', 'red');
hold on;
plotpv(P(1:R-1, :), T)
Wp = W(:, 1:R-1);
Bp = W(:, R);
plotpc(Wp, Bp);
Change_LineWidth
Change_Marker
hold off;
```

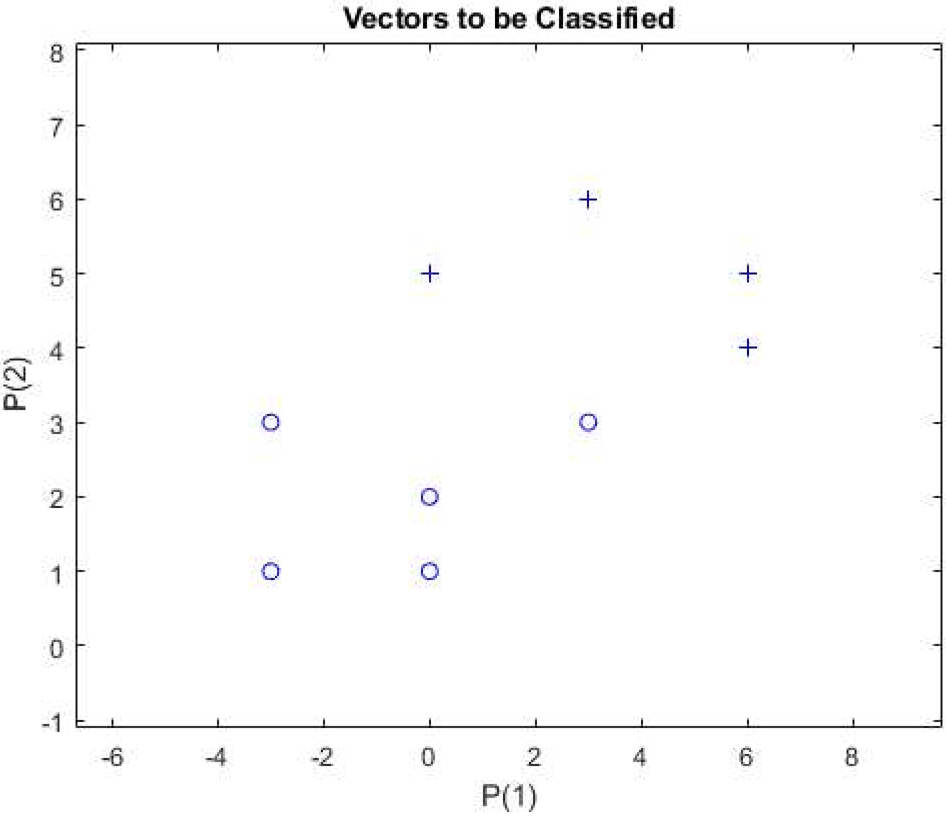
```
Target is

T =

     1     1     1     0     0     0

Solution reached of

A =

     1     1     1     0     0     0

With weights

W =

   -2.2463   -2.6196    3.5678
```

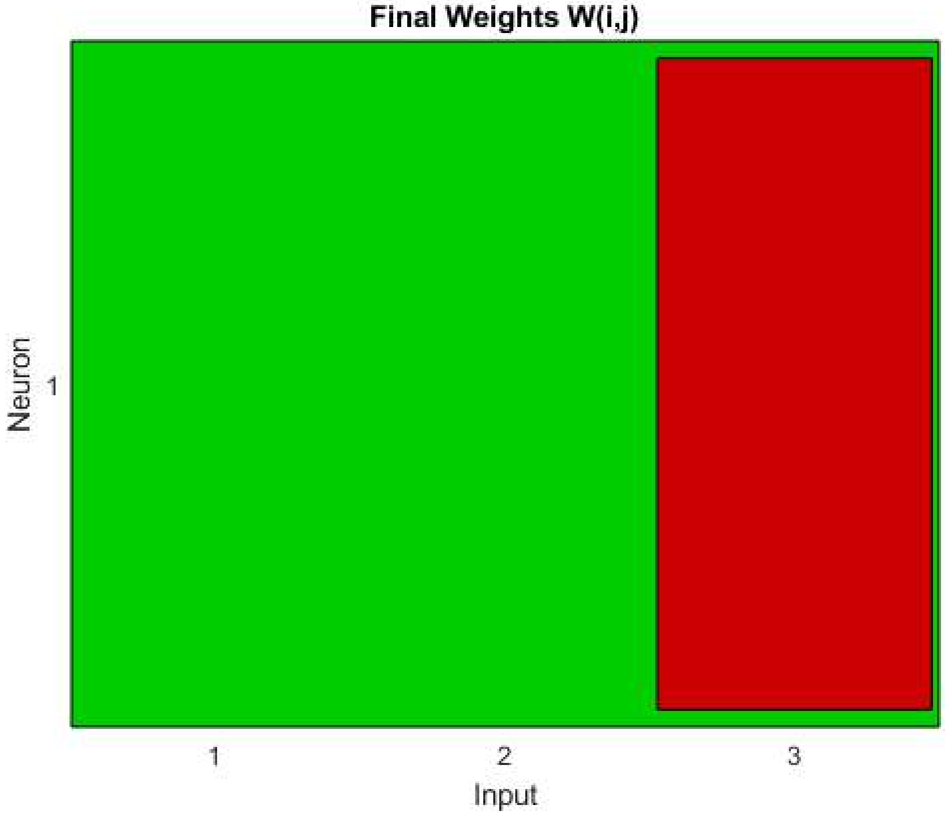


Published with MATLAB® R2022a

```
%David Melanson
clf reset; clearvars; clc
pausetime = 0;
pauseflag = 1;
maxw = 10;
minw = 0;

%Input vectors
P = [
    -3 -3 0 0 0 3 3 6 6;
    1 3 1 2 5 3 6 4 5;
    1 1 1 1 1 1 1 1 1
    ];

%Target vector
T = [-1 -1 -1 -1 1 -1 1 1 1];

%Initialize network
%===================================================
[R, Q] = size(P); [S, Q] = size(T);
W0 = zeros(S, R);
B0 = ones(S, 1);

figure(5)
plotpv(P(1:R-1,:), hardlim(T));

% Plot original values
%===================================================
pause(pausetime);
figure(1);
hintonw(W0, maxw, minw);
title('Original Weights W(i,j)');

% TRAIN THE NETWORK
%===================================================
% TRAINING PARAMETERS
disp_freq = 1;
max_epoch = 10;
% lr = 0.1;
% dr = lr/3;
% lp Learning Parameter
% lr Learning Rate
% dr Decay Rate
lr = 1;
lp.lr = lr;
lp.dr = 0;

W = W0;
B = B0;

for epoch = 1:max_epoch
    for q = 1:Q
        % PRESENTATION PHASE
        A = T(:, q);
        % LEARNING PHASE
        dW = learnhd(W, P(:, q), [], [], A, [], [], [], [], [], lp, []);
        W = W +dW;
        if pauseflag == 1
            pause(pausetime)
            figure(1)
        end
    end
    %end loop if solution found
    if (hardlims(W*P) == T)
        break
    end
    % DISPLAY PROGRESS
    if rem(epoch, disp_freq) == 0
        pause(pausetime)
        hintonw(W, maxw, minw)
        title('Weights W(i,j)');
    end
end

% PLOT FINAL VALUES
hintonw(W, maxw, minw);
title('Final Weights W(i,j)');
pause(pausetime);

% SUMMARIZE RESULTS
%===================================================
disp('With inputs of ');
P

disp('and weights of ');
W

disp('The network responds with outputs');
A = hardlims(W*P)
```

```
With inputs of

P =

    -3    -3     0     0     0     3     3     6     6
     1     3     1     2     5     3     6     4     5
     1     1     1     1     1     1     1     1     1

and weights of

W =

   180   100   -10

The network responds with outputs

A =

    -1    -1     1     1     1     1     1     1     1
```

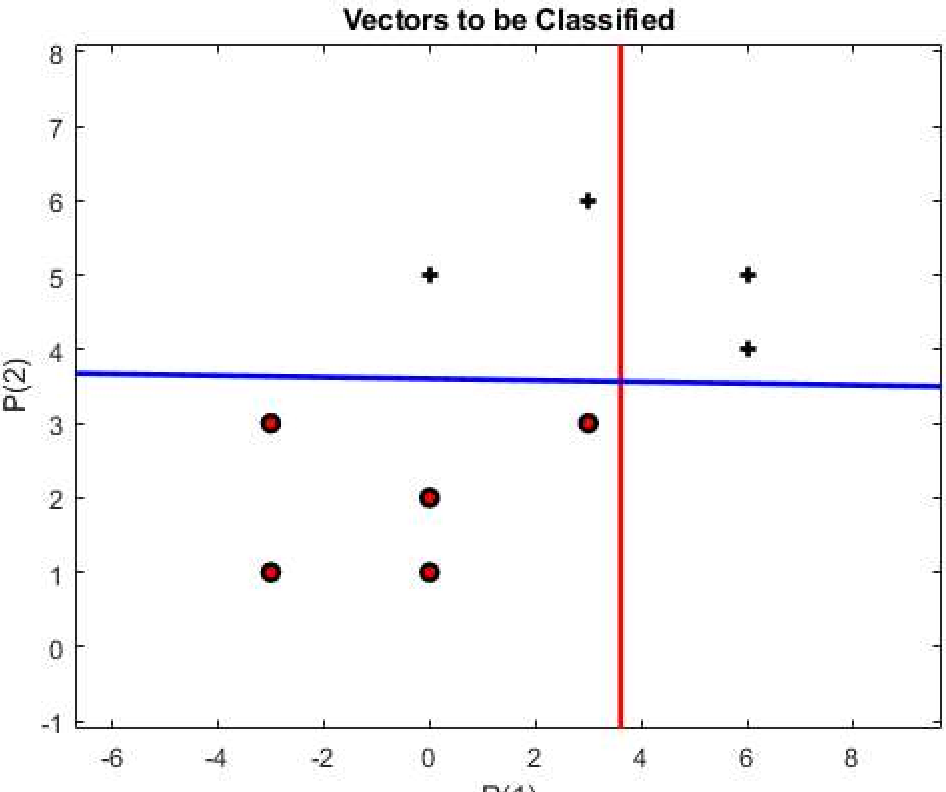




Published with MATLAB® R2022a

```
clear all; clc; cla; clf;
pause_flag = 0;
max_epoch = 200;

%Input vectors
P = [
    -3 -3 0 0 0 3 3 6 6;
    1 3 1 2 5 3 6 4 5;
    1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
    ];

%Target vector
T = [0 0 0 0 1 0 1 1 1];

%Input layer
[R, Q] = size(P); [S, Q] = size(T);

%Initialize network parameters
figure(1);
plotpv(P(1:R-1,:), T);
Change_Marker

%Initialize weights randomly
W = rand(S,R);
Wp = W(:, 1:R-1);
Bp = W(:,R);

%display initial values
%The input vectors are replotted
plotpv(P(1:R-1,:), T);

plotpc(Wp, Bp);

watchon;
cla;
plotpv(P(1:R-1,:), T);

pause(3);
figure(1);
E=1;
linehandle = plotpc(Wp, Bp);
%disp('Hit something to continue');

%sum squared error performance function
epoch = 1;
while (sse(E) && (epoch <= max_epoch))
    Ai = hardlim(W*P);
    Ei = T-Ai;
    dWq = learnp(W, P, [], [], [], [], Ei, [], [], [], [], []);
    W = W+dWq;
    Wp = W(:, R-1);
    Bp = W(:, R);
    linehandle = plotpc(Wp, Bp, linehandle);
    lines = findobj(gcf, 'Type', 'Line');
    Change_LineWidth
    Change_Marker
    drawnow;
    if(pause_flag == 1)
        pause(1);
    end
    A = hardlim(W*P);
    %error - Target minus calculated this epoch
    E = T-A;
    %disp(E)
    epoch = epoch +1;
end
watchoff;
disp('Target is ')
T
disp('Solution reached of ')
A
disp('With weights ')
W

testPoint = findobj(gca, 'Type', 'Line');
set(testPoint, 'Color', 'red');
hold on;
plotpv(P(1:R-1, :), T)
Wp = W(:, 1:R-1);
Bp = W(:, R);
plotpc(Wp, Bp);
Change_LineWidth
Change_Marker
hold off;
```

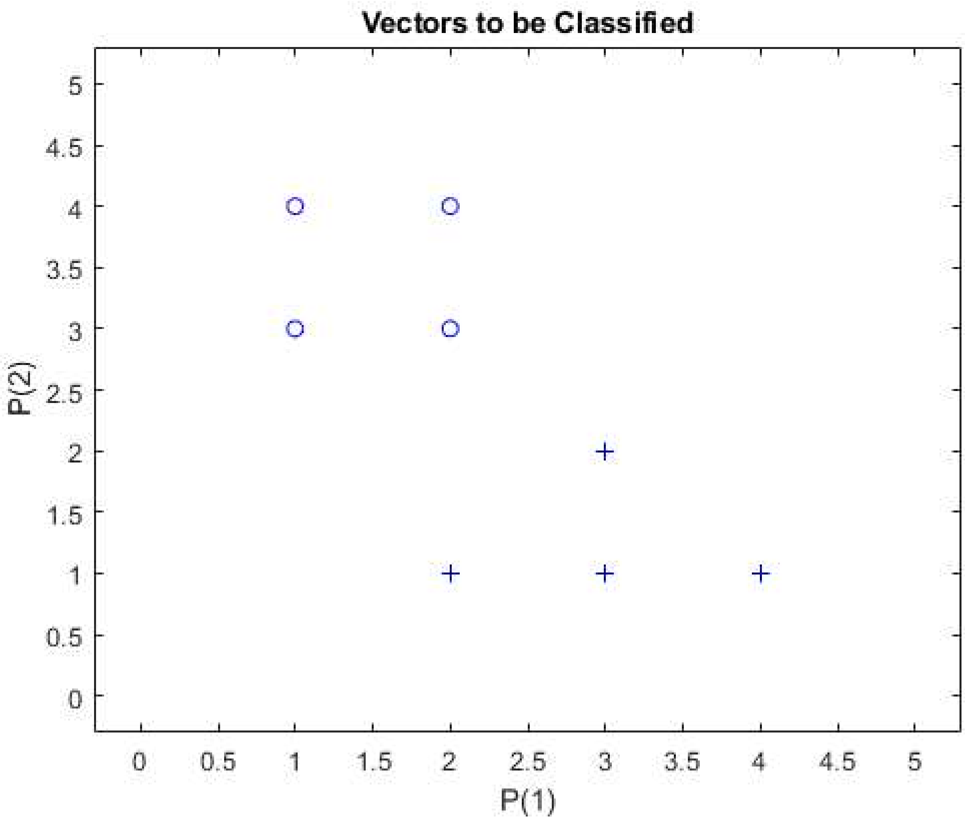
```
Target is

T =

     0     0     0     0     1     0     1     1     1

Solution reached of

A =

     0     0     0     0     1     0     1     1     1

With weights

W =

    0.0759    7.0540  -25.4692
```

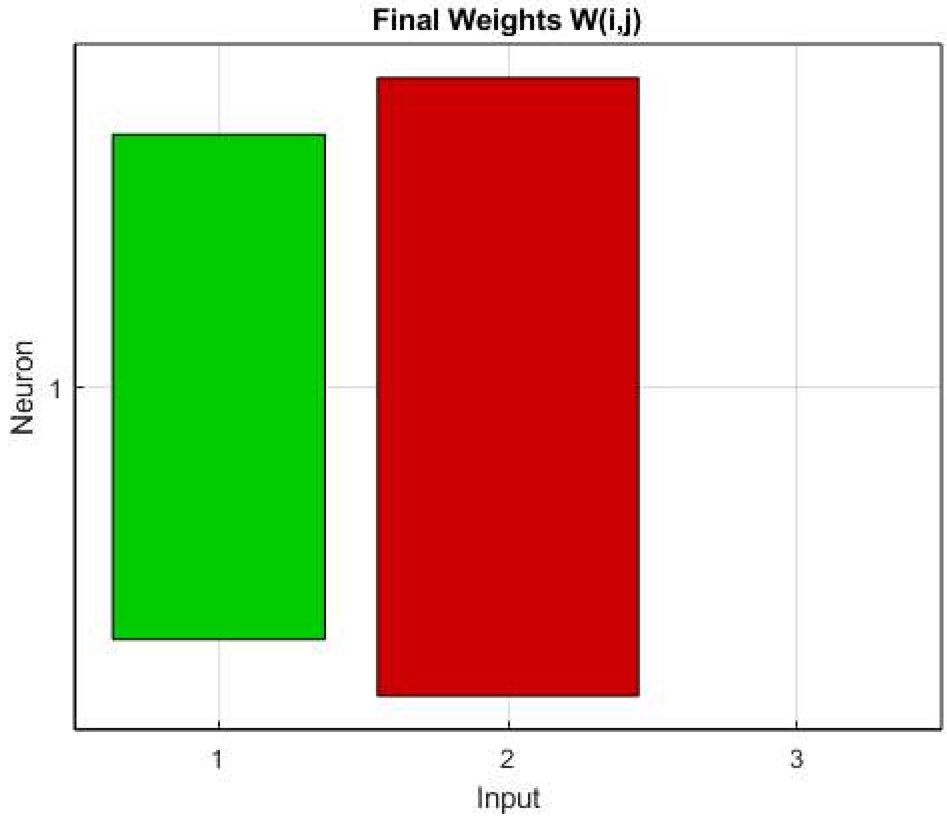


Published with MATLAB® R2022a

```
%David Melanson
clf reset; clearvars; clc
pausetime = 0;
pauseflag = 1;
maxw = 10;
minw = 0;

%Input vectors
P = [
    2 3 4 3 1 1 2 2;
    1 1 1 2 3 4 3 4;
    1 1 1 1 1 1 1 1
    ];

%Target vector
T = [1 1 1 1 -1 -1 -1 -1];

%Initialize network
%===================================================
[R, Q] = size(P); [S, Q] = size(T);
W0 = zeros(S, R);
B0 = ones(S, 1);

figure(5)
plotpv(P(1:R-1,:), hardlim(T));

% Plot original values
%===================================================
pause(pausetime);
figure(1);
hintonw(W0, maxw, minw);
title('Original Weights W(i,j)');

% TRAIN THE NETWORK
%===================================================
% TRAINING PARAMETERS
disp_freq = 1;
max_epoch = 10;
% lr = 0.1;
% dr = lr/3;
% lp Learning Parameter
% lr Learning Rate
% dr Decay Rate
lr = 1;
lp.lr = lr;
lp.dr = 0;

W = W0;
B = B0;

for epoch = 1:max_epoch
    for q = 1:Q
        % PRESENTATION PHASE
        A = T(:, q);
        % LEARNING PHASE
        dW = learnhd(W, P(:, q), [], [], A, [], [], [], [], [], lp, []);
        W = W +dW;
        if pauseflag == 1
            pause(pausetime)
            figure(1)
        end
    end
    %end loop if solution found
    if (hardlims(W*P) == T)
        break
    end
    % DISPLAY PROGRESS
    if rem(epoch, disp_freq) == 0
        pause(pausetime)
        hintonw(W, maxw, minw)
        title('Weights W(i,j)');
    end
end

% PLOT FINAL VALUES
hintonw(W, maxw, minw);
title('Final Weights W(i,j)');
pause(pausetime);

% SUMMARIZE RESULTS
%====================================================
disp('With inputs of ');
P

disp('and weights of ');
W

disp('The network responds with outputs');
A = hardlims(W*P)
```

```
With inputs of

P =

     2     3     4     3     1     1     2     2
     1     1     1     2     3     4     3     4
     1     1     1     1     1     1     1     1

and weights of

W =

     6    -9     0

The network responds with outputs

A =

     1     1     1     1    -1    -1    -1    -1
```

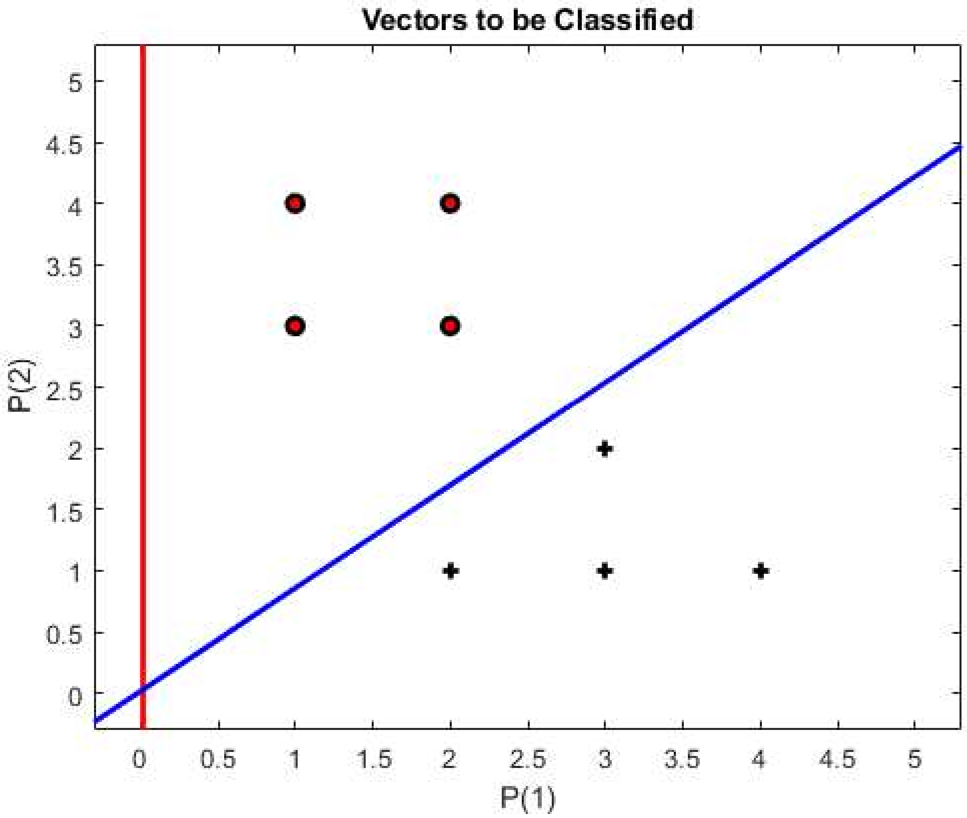




Published with MATLAB® R2022a

```
clear all; clc; cla; clf;
pause_flag = 0;
max_epoch = 200;

%Input vectors
P = [
    2 3 4 3 1 1 2 2;
    1 1 1 2 3 4 3 4;
    1 1 1 1 1 1 1 1
    ];

%Target vector
T = [1 1 1 1 0 0 0 0];

%Input layer
[R, Q] = size(P); [S, Q] = size(T);

%Initialize network parameters
figure(1);
plotpv(P(1:R-1,:), T);
Change_Marker

%Initialize weights randomly
W = rand(S,R);
Wp = W(:, 1:R-1);
Bp = W(:,R);

%display initial values
%The input vectors are replotted
plotpv(P(1:R-1,:), T);

plotpc(Wp, Bp);

watchon;
cla;
plotpv(P(1:R-1,:), T);

pause(3);
figure(1);
E=1;
linehandle = plotpc(Wp, Bp);
%disp('Hit something to continue');

%sum squared error performance function
epoch = 1;
while (sse(E) && (epoch <= max_epoch))
    Ai = hardlim(W*P);
    Ei = T-Ai;
    dWq = learnp(W, P, [], [], [], [], Ei, [], [], [], [], []);
    W = W+dWq;
    Wp = W(:, R-1);
    Bp = W(:, R);
    linehandle = plotpc(Wp, Bp, linehandle);
    lines = findobj(gcf, 'Type', 'Line');
    Change_LineWidth
    Change_Marker
    drawnow;
    if(pause_flag == 1)
        pause(1);
    end
    A = hardlim(W*P);
    %error - Target minus calculated this epoch
    E = T-A;
    %disp(E)
    epoch = epoch +1;
end
watchoff;
disp('Target is ')
T
disp('Solution reached of ')
A
disp('With weights ')
W

testPoint = findobj(gca, 'Type', 'Line');
set(testPoint, 'Color', 'red');
hold on;
plotpv(P(1:R-1, :), T)
Wp = W(:, 1:R-1);
Bp = W(:, R);
plotpc(Wp, Bp);
Change_LineWidth
Change_Marker
hold off;
```

```
Target is

T =

     1     1     1     1     0     0     0     0

Solution reached of

A =

     1     1     1     1     0     0     0     0

With weights

W =

    6.7792   -8.0660    0.1299
```



Published with MATLAB® R2022a