Práctica #3 Red perceptrón (método gráfico y con aprendizaje)

Jorge Gómez Reus

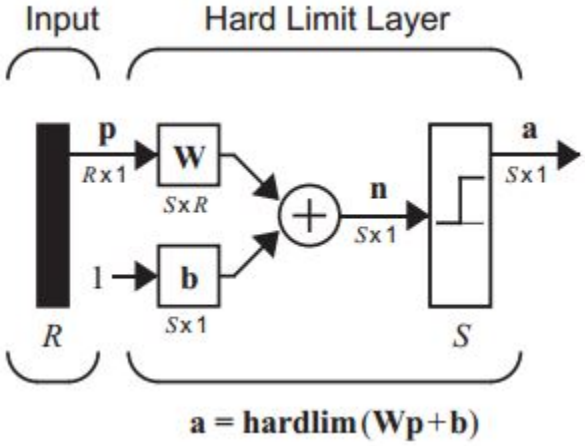
Índice

1. Introducción

El perceptrón es considerada la unidad básica para generar redes neuronales, esta puede clasificar vectores linealmente en dos categorías. El objetivo del perceptrón es acercar a los vectores de entrada al patrón más cercano.

1.1. Modelo

Figura 1: Modelo



2. Diagrama de Flujo

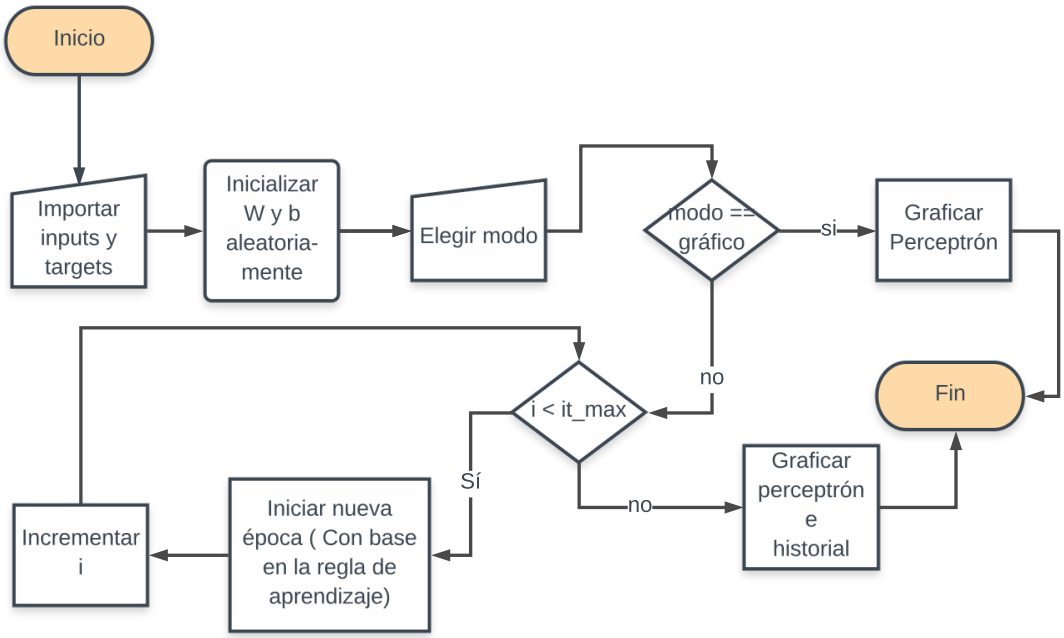


Figura 2: Diagrama de Flujo

3. Resultados

3.1. Ejemplo 1

3.1.1. Datos

$$p_1 = \begin{bmatrix} 2 \\ 2 \end{bmatrix}, t = 0$$

$$p_2 = \begin{bmatrix} 1 \\ -2 \end{bmatrix}, t = 1$$

$$p_3 = \begin{bmatrix} -2 \\ 2 \end{bmatrix}, t = 0$$

$$p_4 = \begin{bmatrix} -1 \\ -1 \end{bmatrix}, t = 1$$

$$W(0) = \begin{bmatrix} 0{,}4121 & -0{,}9363 \end{bmatrix}$$

$$b = 0{,}2769$$

3.1.2. Resultado

```
W = 0.4121     −0.9363
b = 0.2769
Elija un modo: 1–>Gráfico , 2–>Regla de Aprendizaje
>>> 1
```

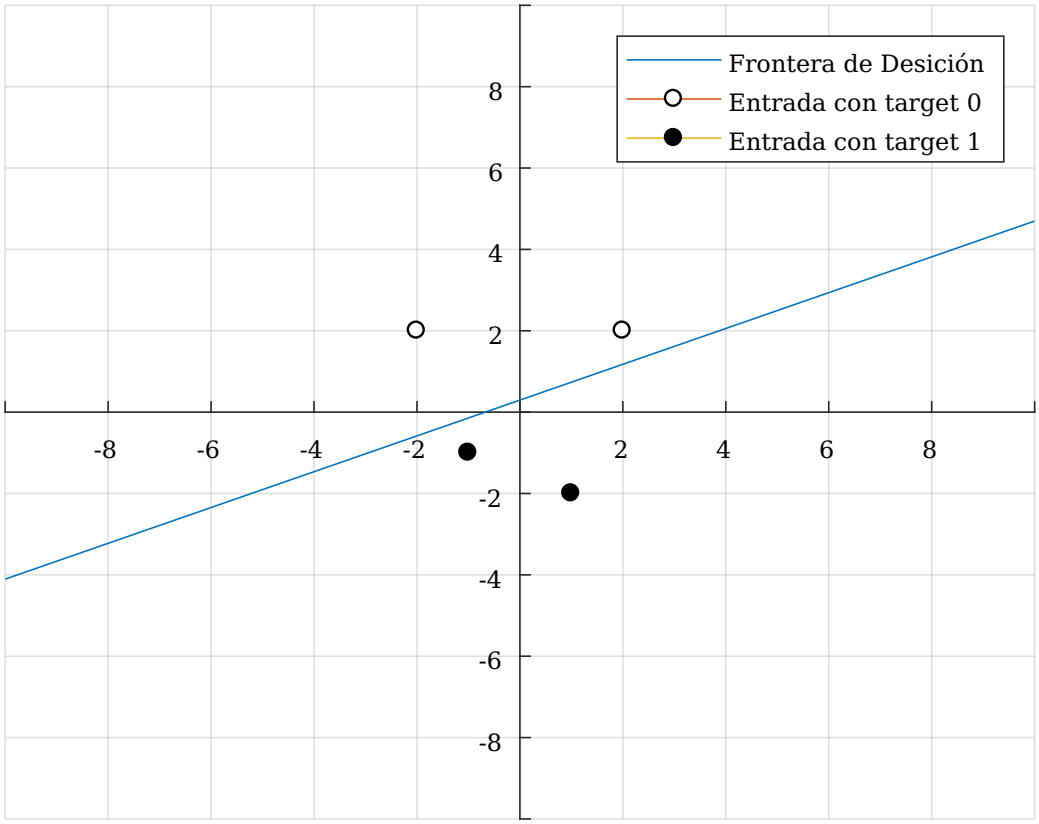


Figura 3: Gráfica del perceptrón 1

3.2. Ejemplo 2

3.2.1. Datos

$$p_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, t = 0$$

$$p_2 = \begin{bmatrix} -1 \\ -1 \end{bmatrix}, t = 1$$

$$W(0) = \begin{bmatrix} -0{,}9077 & -0{,}8057 \end{bmatrix}$$

$$b = 0{,}8235$$

3.2.2. Resultado

```
W = −0.9077    −0.8057
b = 0.8235
Elija un modo: 1–>Gráfico, 2–>Regla de Aprendizaje
>>> 2
W = −0.9077    −0.8057
b = 0.8235
```

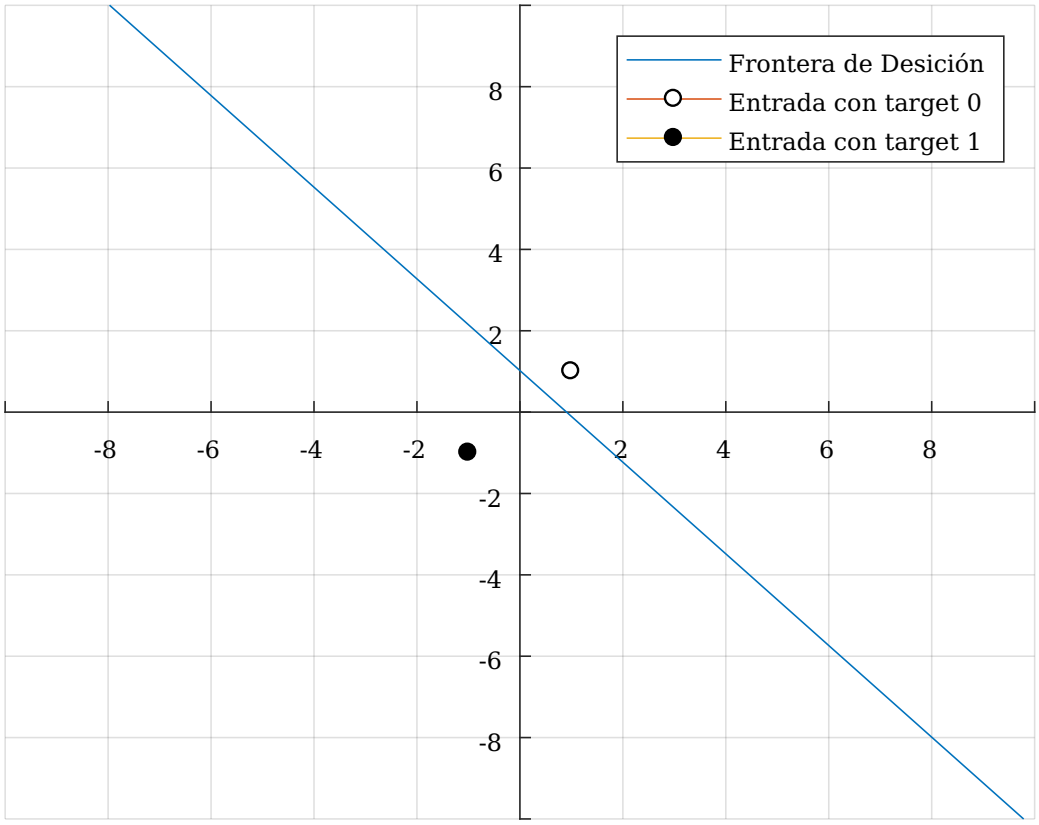


Figura 4: Gráfica del perceptrón 2

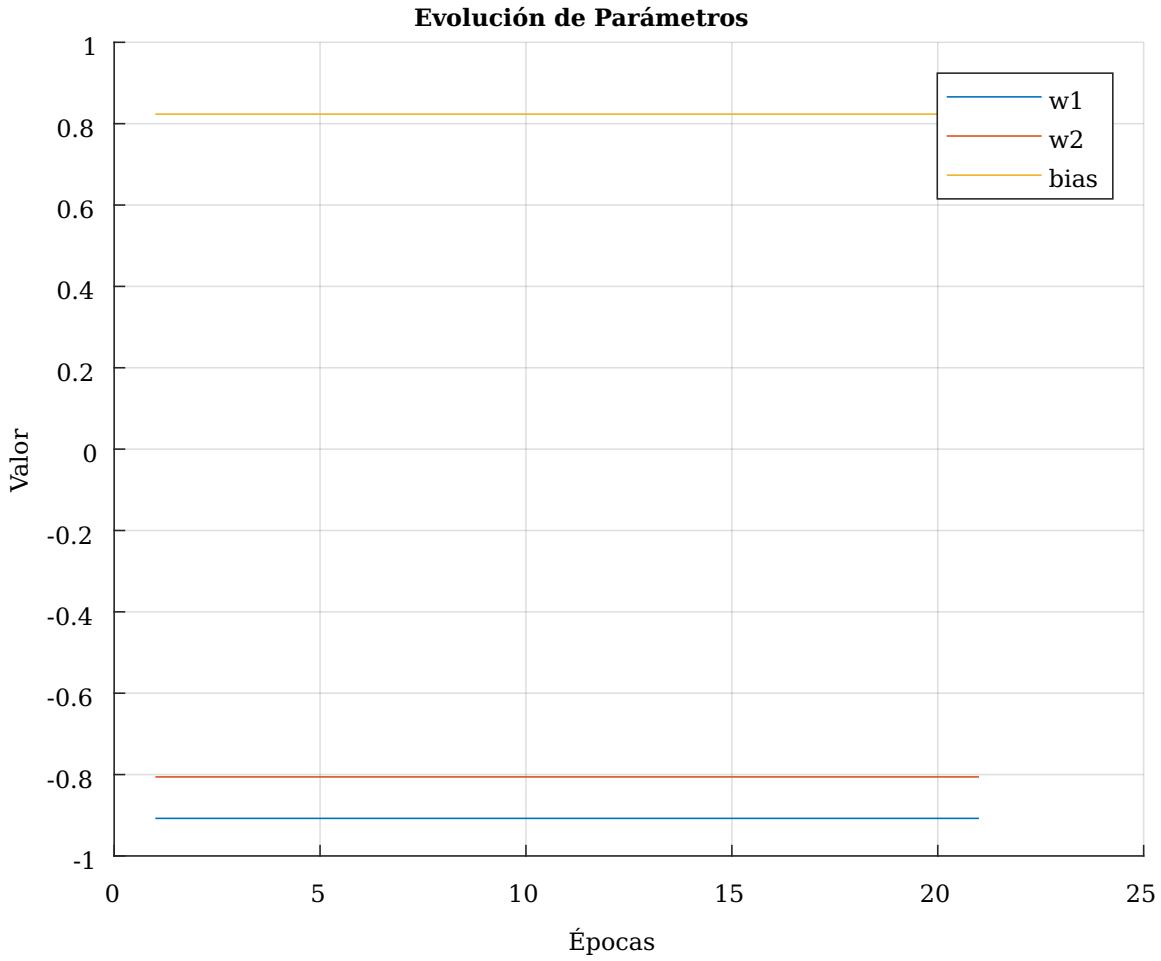


Figura 5: Gráfica de evolución de datos del perceptrón 2

3.3. Ejemplo 3

3.3.1. Datos

$$p_1 = \begin{bmatrix} 0 \\ 2 \end{bmatrix}, t = 1$$

$$p_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, t = 1$$

$$p_3 = \begin{bmatrix} 0 \\ -2 \end{bmatrix}, t = 0$$

$$p_2 = \begin{bmatrix} 2 \\ 0 \end{bmatrix}, t = 0$$

$$W(0) = \begin{bmatrix} -0{,}9311 & -0{,}1225 \end{bmatrix}$$

$$b = 1{,}3816$$

3.3.2. Resultado

```
W = −0.9311    −0.1225
b = 0.3816
Elija un modo: 1–>Gráfico , 2–>Regla de Aprendizaje
>>> 2
W = −0.9311     1.8775
b = 1.3816
```

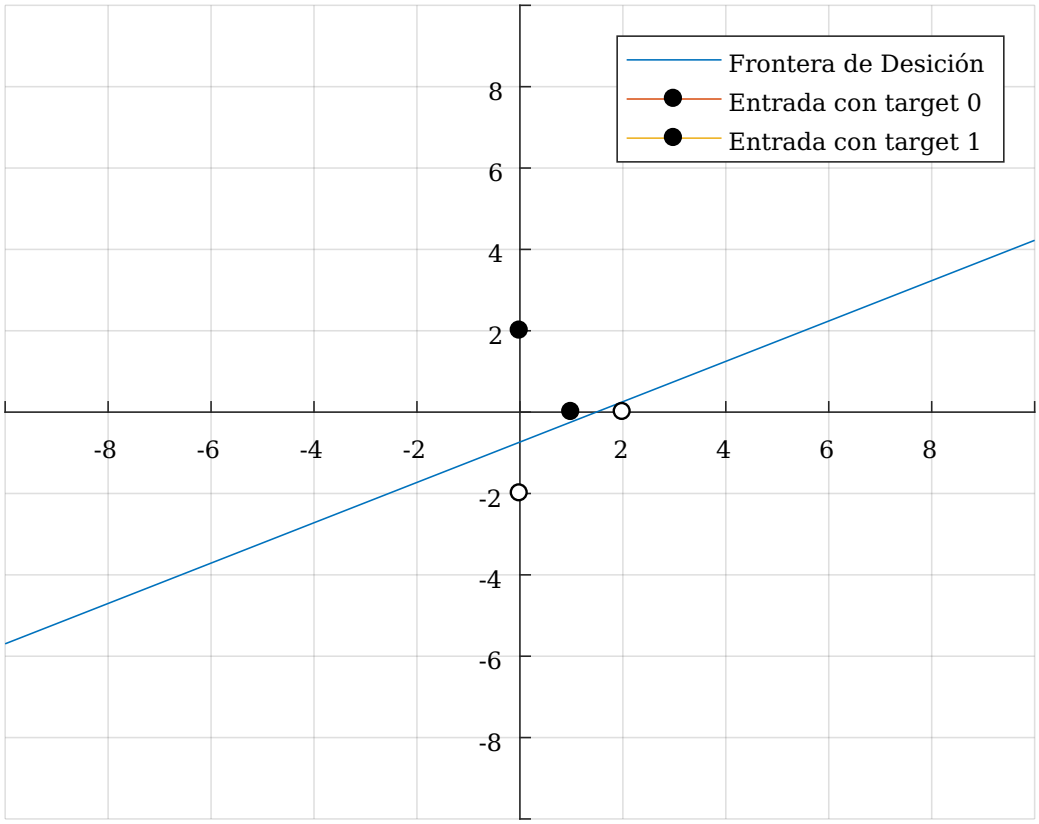


Figura 6: Gráfica del perceptrón 3

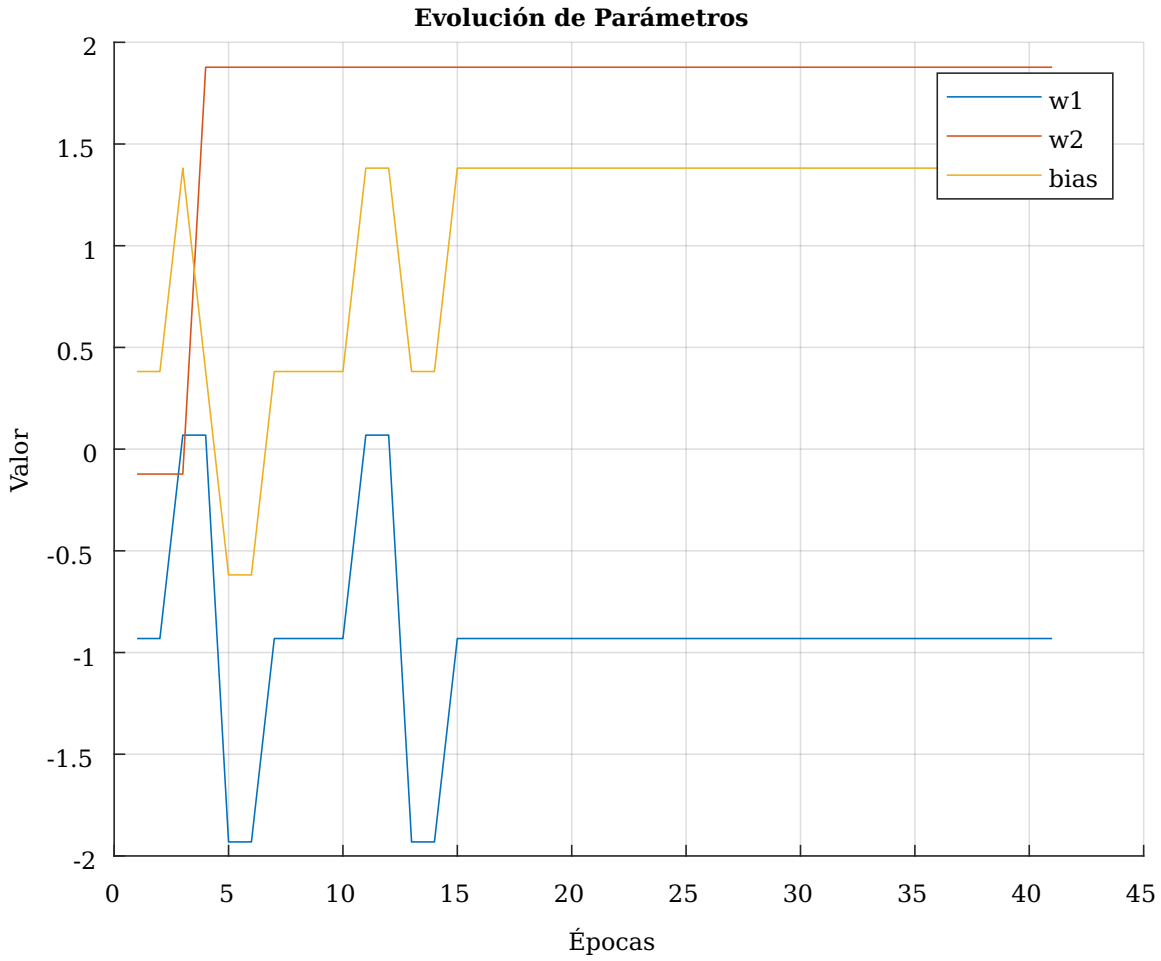


Figura 7: Gráfica de evolución de datos del perceptrón 3

4. Discusión de Resultados

Para cada uno de los resultados se muestra:

1. Los datos con los cuales fue realizado el ejemplo.
2. Los pesos y bias iniciales.
3. La gráfica del "historial" de la evolución de los parámetros del perceptrón
4. La gráfica de los vectores de entrada con su target y la frontera de desición final.

5. Conclusiones

El perceptrón es la unidad básico de las redes neuronales que se usan hoy en día, su creador, Frank Rosenblatt, hizp una muy importante aportación para el campo de las redes neuronales artificiales. La práctica estuvo mucho más corta de lo que esperaba, la regla de aprendizaje es "magia".

6. Referencias

Martin T Hagan. Machine Learning, Neural Network Design (2nd Edition), 2014.
https://medium.com/@thomascountz/calculate-the-decision-boundary-of-a-single-perceptron-visualizing-

7. Apéndice

Listing 1: Código

```
% Datos ingresados por el usuario
inputs = importdata('inputs.txt');
targets = importdata('targets.txt');
max_it = 10;
% merged the matrixes
total_matrix = [ inputs targets];
max_random_range = 1;
min_random_range = -1;
% Weight and bias initialization
W = rand(1, size(inputs, 2))*(2*max_random_range) + min_random_range
b = rand
Wevo = [];
bevo = [];
% For plotting the evolution of the parameters
Wevo = [Wevo; W];
bevo = [bevo; b];
mode = input('Elija un modo: 1->Gráfico, 2->Regla de Aprendizaje\n', 's');
if(mode=='1')
        if (size(inputs, 2) == 2)
                plotPerceptron(total_matrix, W, b);
        else
                fprintf("Solo impresiones en 2 dimensiones soportada");
```

```
end
elseif(mode=='2')
        % For convergence checking
        Waux = W;
        baux = b;
        % Begin the iterations
        for i = 1:max_it
                for row = total_matrix.'
                        % Array Indexing
                        p = row(1:size(inputs, 2));
                        target = row(size(inputs, 2) + 1);
                        a = hardlim(W*p + b);
                        % Calculate the error
                        e = target - a;
                        % Convergence Checking
                        Waux = W;
                        baux = b;
                        % Weight update
                        W = W + e*p';
                        % Bias update
                        b = b + e;
                        % Save the values
                        Wevo = [Wevo; W];
                        bevo = [bevo; b];
                end
        end
        W
        b
        plotHistory(Wevo, bevo);
        if (size(inputs, 2) == 2)
                plotPerceptron(total_matrix, W, b);
        else
                fprintf("Solo impresiones en 2 dimensiones soportada");
        end
else
        fprintf("Opción no reconocida\n");
end

function h = circle(x ,y, r, color)
        hold on
        h = plot(x, y, '-o', ...
        'MarkerSize', r, ...
        'MarkerEdgeColor', 'black',...
        'MarkerFaceColor', color);
        hold off
end

function h = plotPerceptron(matrix, W, b)
        % Plot the perceptron desicion boundary and the inputs
```

```
72          figure
73          ax = gca;                       % gets the current axes
74          ax.XAxisLocation = 'origin';    % sets t1hem to zero
75          ax.YAxisLocation = 'origin';
76          hold on
77          grid on
78          % plot the desicion boundary
79          x = -10:10;
80          slope = -(b / W(2)) / (b / W(1));
81          intercept = -b / W(2);
82          y = slope * x + intercept;
83          plot(x, y);
84          ylim([-10 10])
85          xlim([-10 10])
86          r = 5;
87          for row = matrix.'
88                  p = row(1:size(matrix, 2));
89                  target = row(size(matrix, 2));
90                  % Plot the input
91                  if (target == 1)
92                          h = circle(p(1), p(2), r, 'black');
93                  else
94                          h = circle(p(1), p(2), r, 'white');
95                  end
96          end
97          legend('Frontera de Desición', 'Entrada con target 0', 'Entrada con target 1')
                ;
98  end
99
100 function plotHistory(Wevo, bevo)
101         % Plot the values
102         hold on
103         grid on
104         title('Evolución de Parámetros');
105         legends = [];
106         x = 1:size(Wevo, 1);
107         for i = 1:size(Wevo, 2)
108                 colW = Wevo(:, i);
109                 plot(x, colW);
110                 legends = [legends, sprintf("w%d", i)];
111         end
112         plot(x, bevo);
113         legends = [legends, "bias"];
114         legends = mat2cell(legends,1, ones(1,numel(legends)));
115         legend(legends{:});
116         xlabel('Épocas')
117         ylabel('Valor')
118         hold off
119 end
```