

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

Jorge Gómez Reus

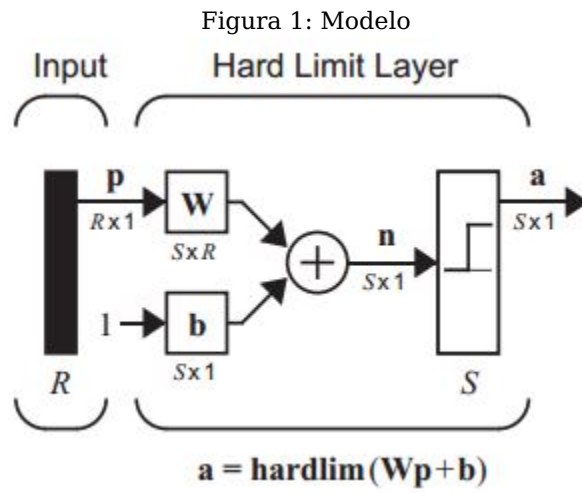
Índice

1. Introducción	1
1.1. Modelo	2
2. Diagrama de Flujo	3
3. Resultados	4
3.1. Ejemplo 1	4
3.1.1. Datos	4
3.1.2. Resultado	4
3.2. Ejemplo 2	6
3.2.1. Datos	6
3.2.2. Resultado	6
3.3. Ejemplo 3	9
3.3.1. Datos	9
3.3.2. Resultado	9
4. Discusión de Resultados	12
5. Conclusiones	12
6. Referencias	12
7. Apéndice	12

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



2. Diagrama de Flujo

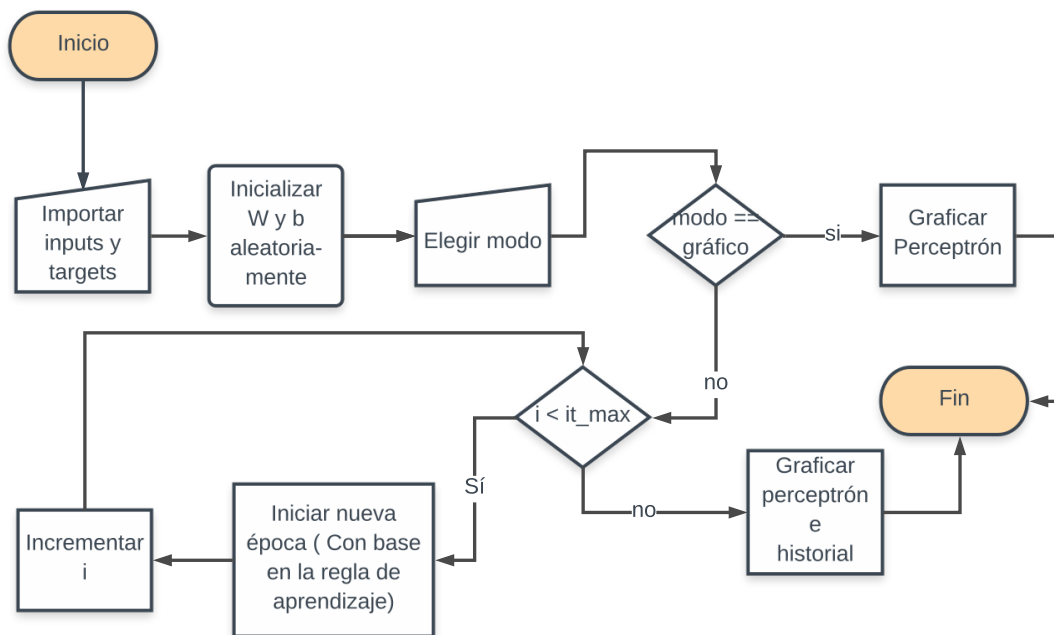


Figura 2: Diagrama de Flujo

3. Resultados

3.1. Ejemplo 1

3.1.1. Datos

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

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

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

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

$$W(0) = [0,4121 \quad -0,9363]$$

$$b = 0,2769$$

3.1.2. Resultado

W = 0.4121 -0.9363

b = 0.2769

Elija un modo: 1→Gráfico , 2→Regla de Aprendizaje

>>> 1

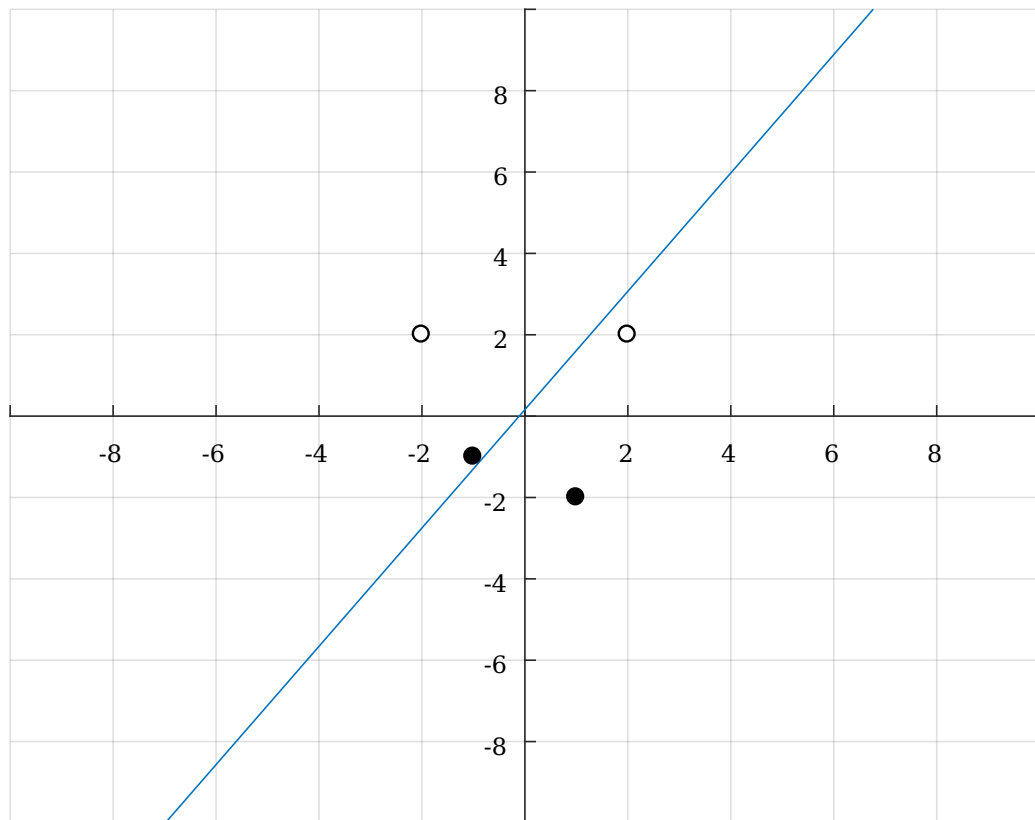


Figura 3: Gráfica del perceptrón 1

3.2. Ejemplo 2

3.2.1. Datos

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

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

$$W(0) = [-0,8946 \quad 0,4757]$$

$$b = 0,2691$$

3.2.2. Resultado

W = -0.8946 0.4757

b = 0.2691

Elija un modo: 1→Gráfico , 2→Regla de Aprendizaje

>>> 2

W = -0.2552 -0.6038

b = 0.4897

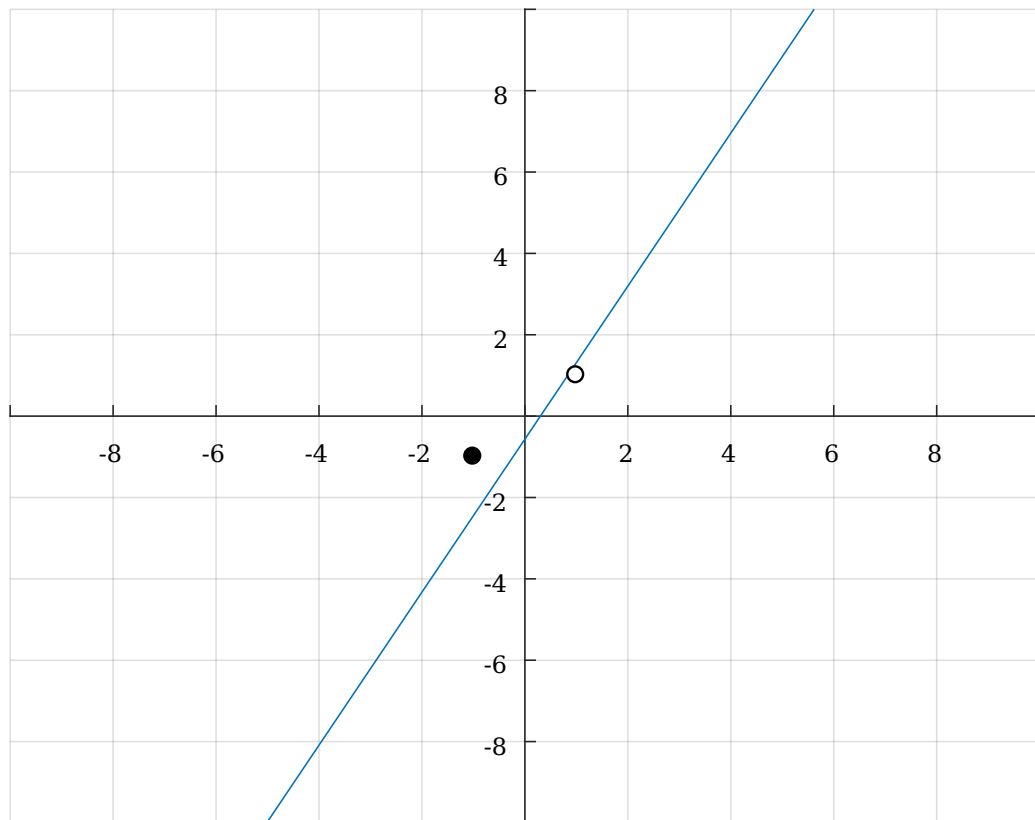


Figura 4: Gráfica del perceptrón 2

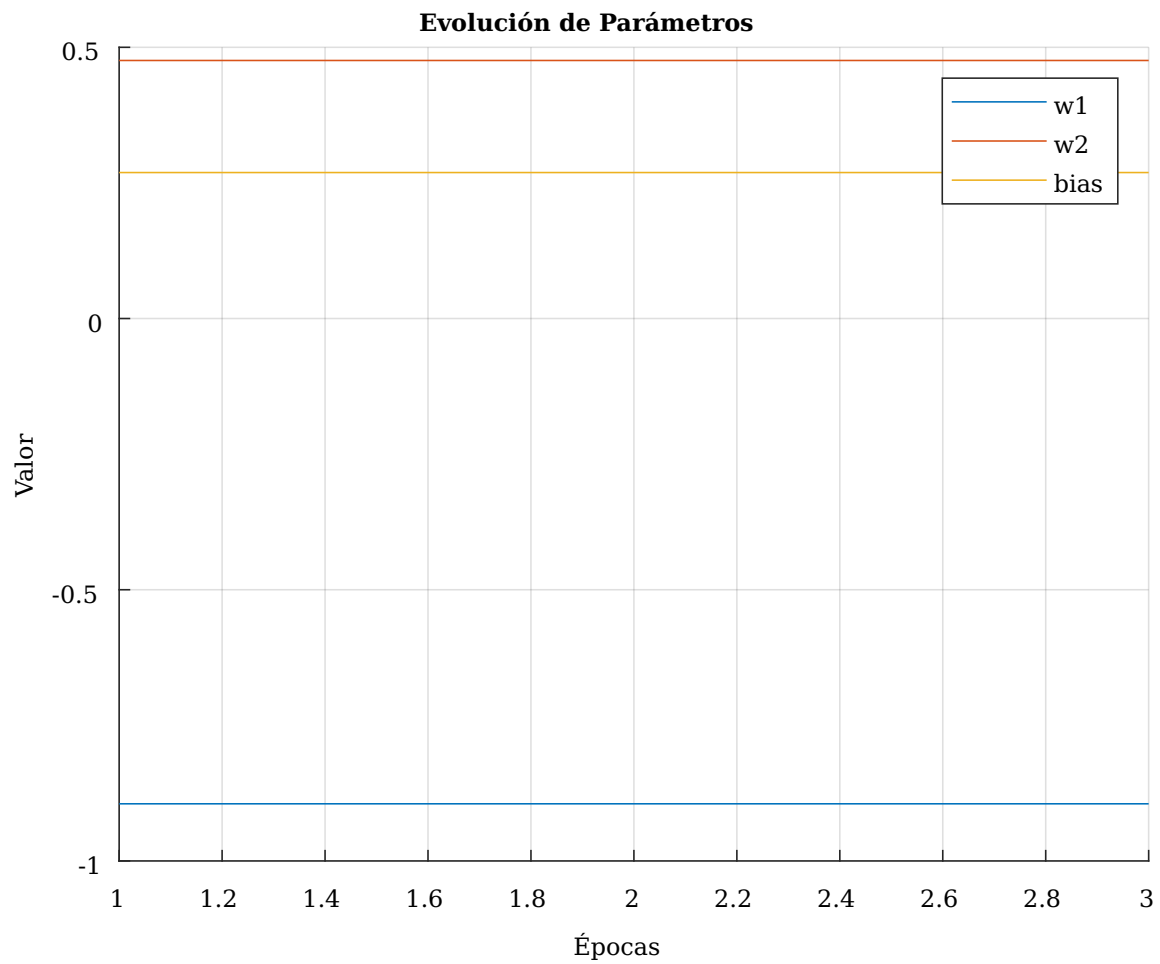


Figura 5: Gráfica de evolución de datos del perceptrón 2

3.3. Ejemplo 3

3.3.1. Datos

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

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

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

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

$$W(0) = [0,4022 \quad 0,3327]$$

$$b = 0,5391$$

3.3.2. Resultado

W = 0.4022 0.3327

b = 0.5391

Elija un modo: 1→Gráfico , 2→Regla de Aprendizaje

>>> 2

W = -1.5978 2.3327

b = 2.5391

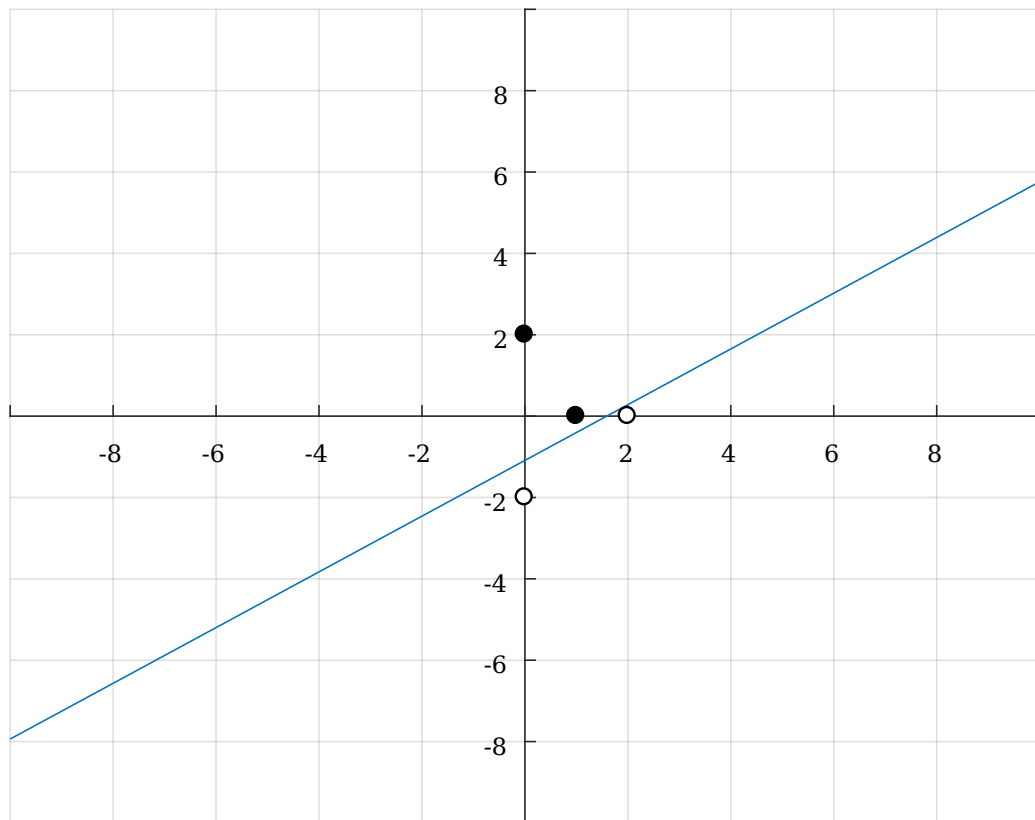


Figura 6: Gráfica del perceptrón 3

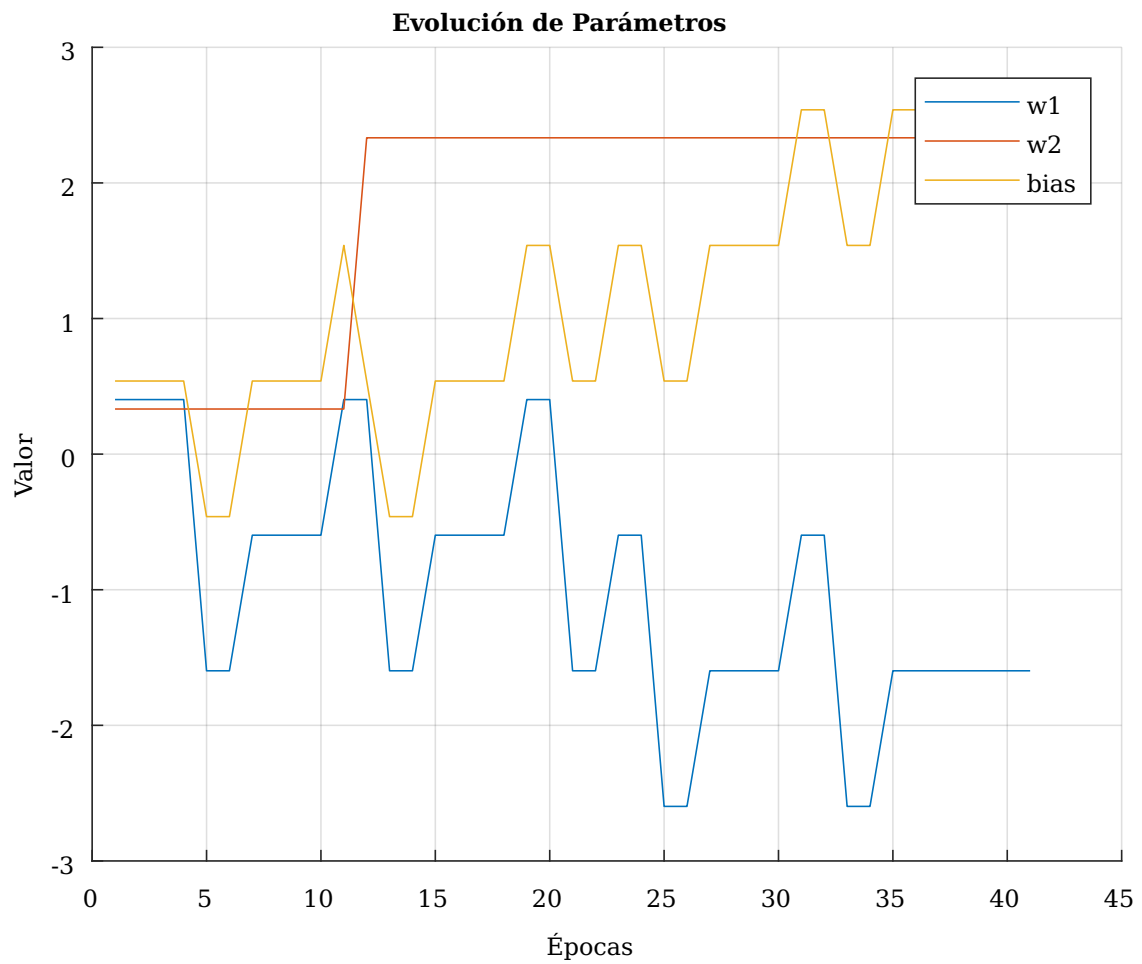


Figura 7: Gráfica de evolución de datos del perceptrón 3

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, hizo 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/@thomascourtz/calculate-the-decision-boundary-of-a-single-perceptron-visualizing>

7. Apéndice

Listing 1: Código

```
1 % Datos ingresados por el usuario
2 inputs = importdata('inputs.txt');
3 targets = importdata('targets.txt');
4 max_it = 10;
5 % merged the matrixes
6 total_matrix = [ inputs targets];
7 max_random_range = 1;
8 min_random_range = -1;
9 % Weight and bias initialization
10 W = rand(1, size(inputs, 2))*(2*max_random_range) + min_random_range
11 b = rand
12 Wevo = [];
13 bevo = [];
14 % For plotting the evolution of the parameters
15 Wevo = [Wevo; W];
16 bevo = [bevo; b];
17 mode = input('Elija un modo: 1->Gráfico, 2->Regla de Aprendizaje\n', 's');
18 if(mode=='1')
19     if (size(inputs, 2) == 2)
20         plotPerceptron(total_matrix, W, b);
21     else
22         fprintf("Solo impresiones en 2 dimensiones soportada");
```

```

23 end
24 elseif(mode=='2')
25     % For convergence checking
26     Waux = W;
27     baux = b;
28     % Begin the iterations
29     for i = 1:max_it
30         for row = total_matrix.'
31             % Array Indexing
32             p = row(1:size(inputs, 2));
33             target = row(size(inputs, 2) + 1);
34             a = hardlim(W*p + b);
35             % Calculate the error
36             e = target - a;
37             % Convergence Checking
38             Waux = W;
39             baux = b;
40             % Weight update
41             W = W + e*p';
42             % Bias update
43             b = b + e;
44             % Save the values
45             Wevo = [Wevo; W];
46             bevo = [bevo; b];
47         end
48     end
49     W
50     b
51     plotHistory(Wevo, bevo);
52     if (size(inputs, 2) == 2)
53         plotPerceptron(total_matrix, W, b);
54     else
55         fprintf("Solo impresiones en 2 dimensiones soportada");
56     end
57 else
58     fprintf("Opción no reconocida\n");
59 end
60
61 function h = circle(x ,y, r, color)
62     hold on
63     h = plot(x, y, '-o', ...
64         'MarkerSize', r, ...
65         'MarkerEdgeColor', 'black',...
66         'MarkerFaceColor', color);
67     hold off
68 end
69
70 function h = plotPerceptron(matrix, W, b)
71     % Plot the perceptron desicion boundary and the inputs

```

```

72     figure
73     ax = gca;                                % gets the current axes
74     ax.XAxisLocation = 'origin';             % sets them to zero
75     ax.YAxisLocation = 'origin';
76     hold on
77     grid on
78     % plot the decision 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 Decisión', 'Entrada con target 0', 'Entrada con target 1')
98     ;
99 end
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

```