Red perceptron (método gráfico y con aprendizaje)

Alumno: *Osornio Gutiérrez Juan Damián*

Introducción.

A finales de 1950 Frank Rosenblatt y otros investigadores desarrollaron una clase de redes neuronales llamadas perceptrones. Las neuronas de estas redes eran similares a las de McCulloch y Pitts.

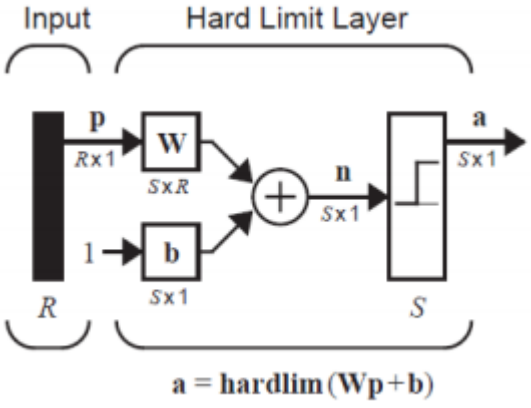
La contribución clave de Rosenblatt fue la introducción de una regla de aprendizaje para la formación de redes perceptrón para resolver problemas de reconocimiento de patrones. Demostró que su regla de aprendizaje siempre convergirá a los pesos correctos de la red, si existen pesos que solucionan el problema. El perceptrón pudo incluso aprender cuando se inicializaba con valores aleatorios de sus pesos y bias.

Por Reglas de Aprendizaje nos referimos a un procedimiento para modificar los pesos y bias de una red (también conocido como algoritmo de entrenamiento). El propósito de la Regla de Aprendizaje es entrenar la red para realizar alguna tarea. Existen varios tipos de reglas de aprendizaje de redes neuronales. Se dividen en tres categorías: Aprendizaje Supervisado, Aprendizaje No Supervisado y Aprendizaje por Reforzamiento.

EN el aprendizaje supervisado, la regla de aprendizaje cuenta con un conjunto de ejemplos (conjunto de entrenamiento) de comportamiento de la red.

$$\{P_1, t_{1,}\}, \{P_2, t_{2,}\}, \dots, \{P_q, t_{q,}\}$$

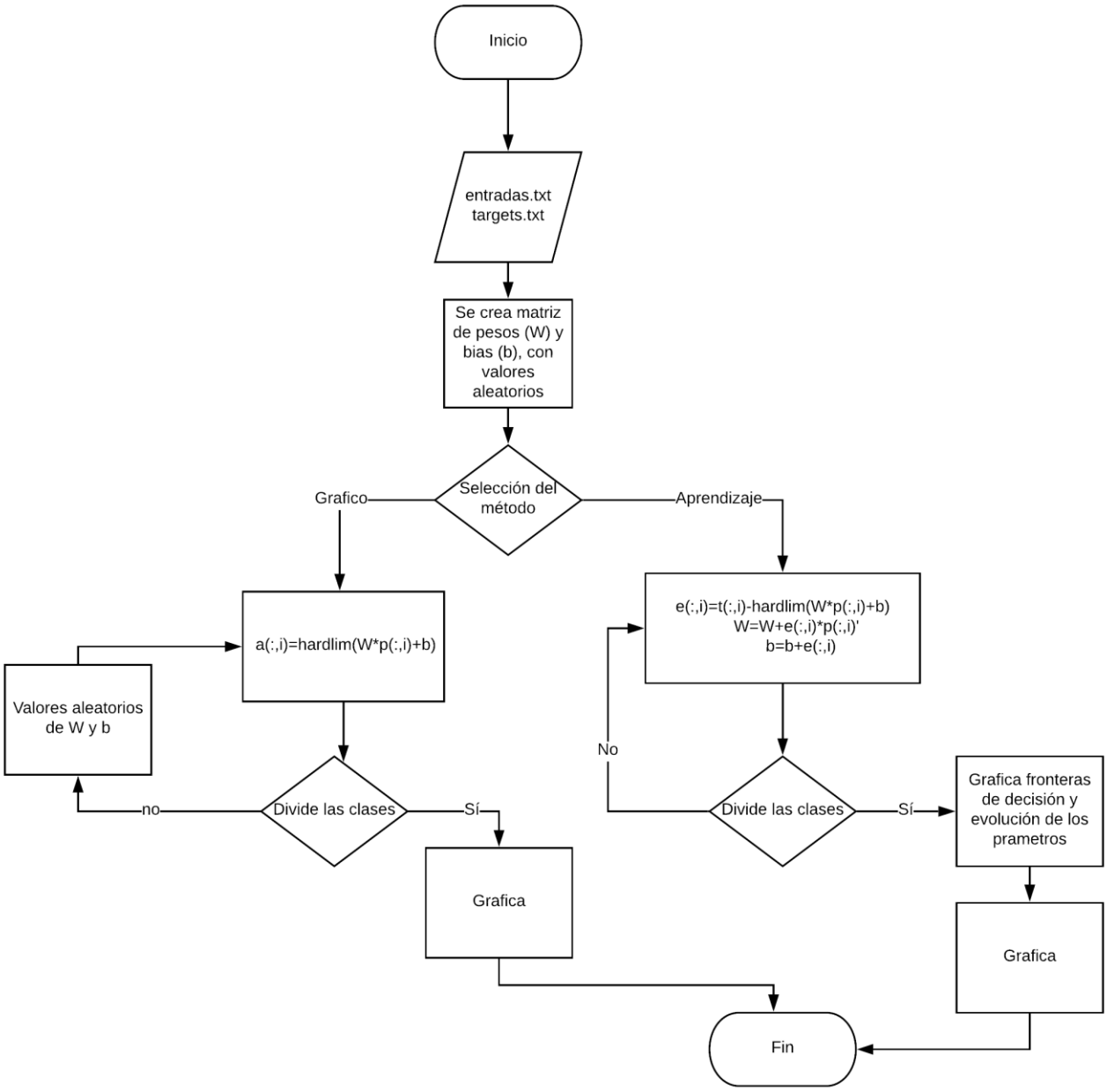
Donde P_q es una entrada a la red y t_q corresponde a la salida correcta (objetivo), Como las entradas se aplican a la red, la salida de la red se compara con los objetivos. La regla de aprendizaje se utiliza para ajustar los pesos y bias de la red objetivos. La regla de aprendizaje del perceptrón cae en esta categoría de aprendizaje supervisado.



Planteamiento del problema.

- El programa recibe un archivo *.txt con el dataset (sólo se puede usar este formato). El usuario elige como resolverlo (método gráfico o aprendizaje).
- El programa da el valor final de los parámetros de perceptron que clasifica correctamente.
    - En ambos casos se debe presentar la gráfica con los datos y dibujar la frontera de decisión final. Adicionalmente, si se usa aprendizaje se debe presentar otra figura con la evolución de los parámetros de la RNA.

Diagrama de flujo.



Resultado.

Se reciben dos archivos .txt con el dataset, uno para los vectores de entrada y otro archivo para los targets.

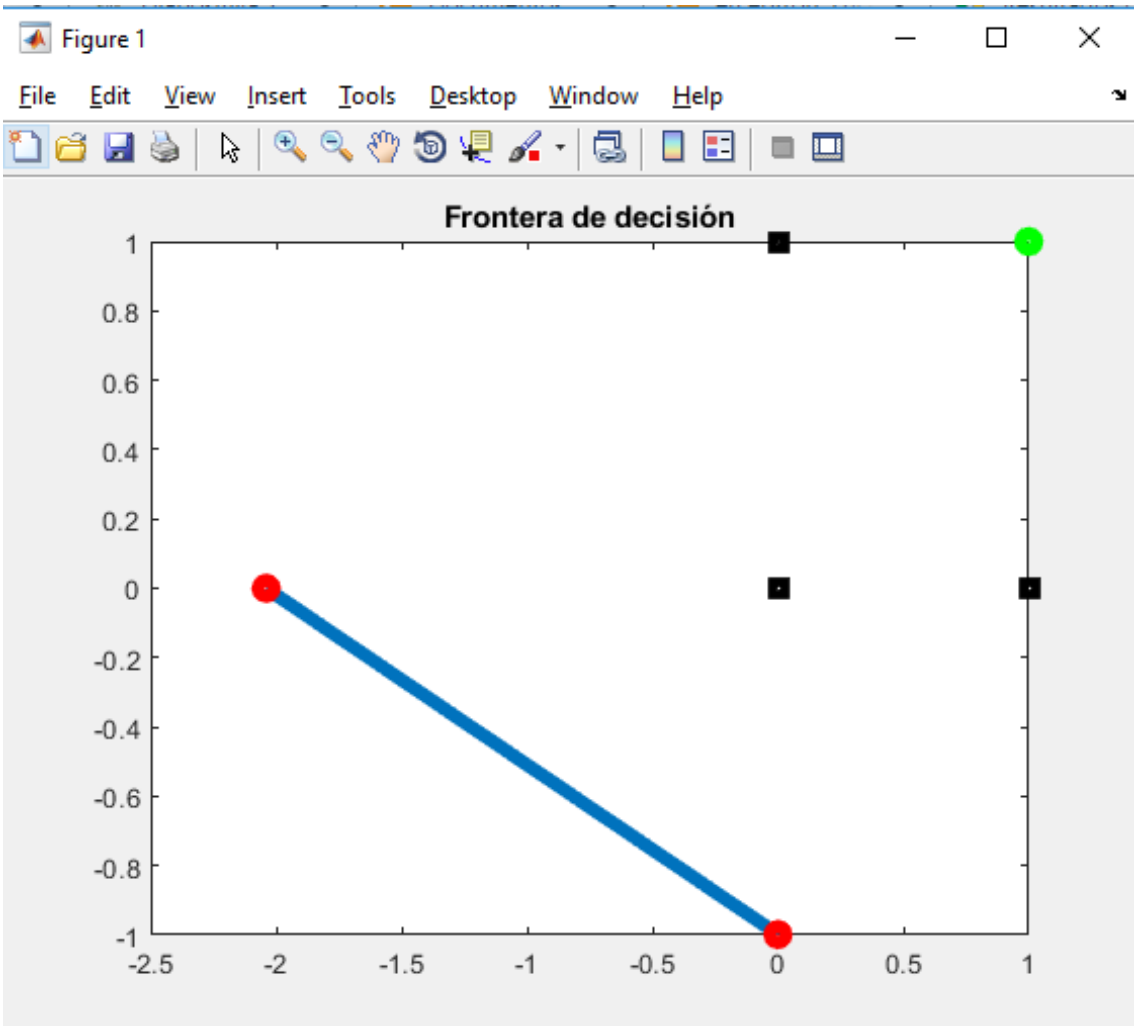
Para los vectores de entrada:

P1	P2	P3	P4
0	0	1	1
0	1	0	1

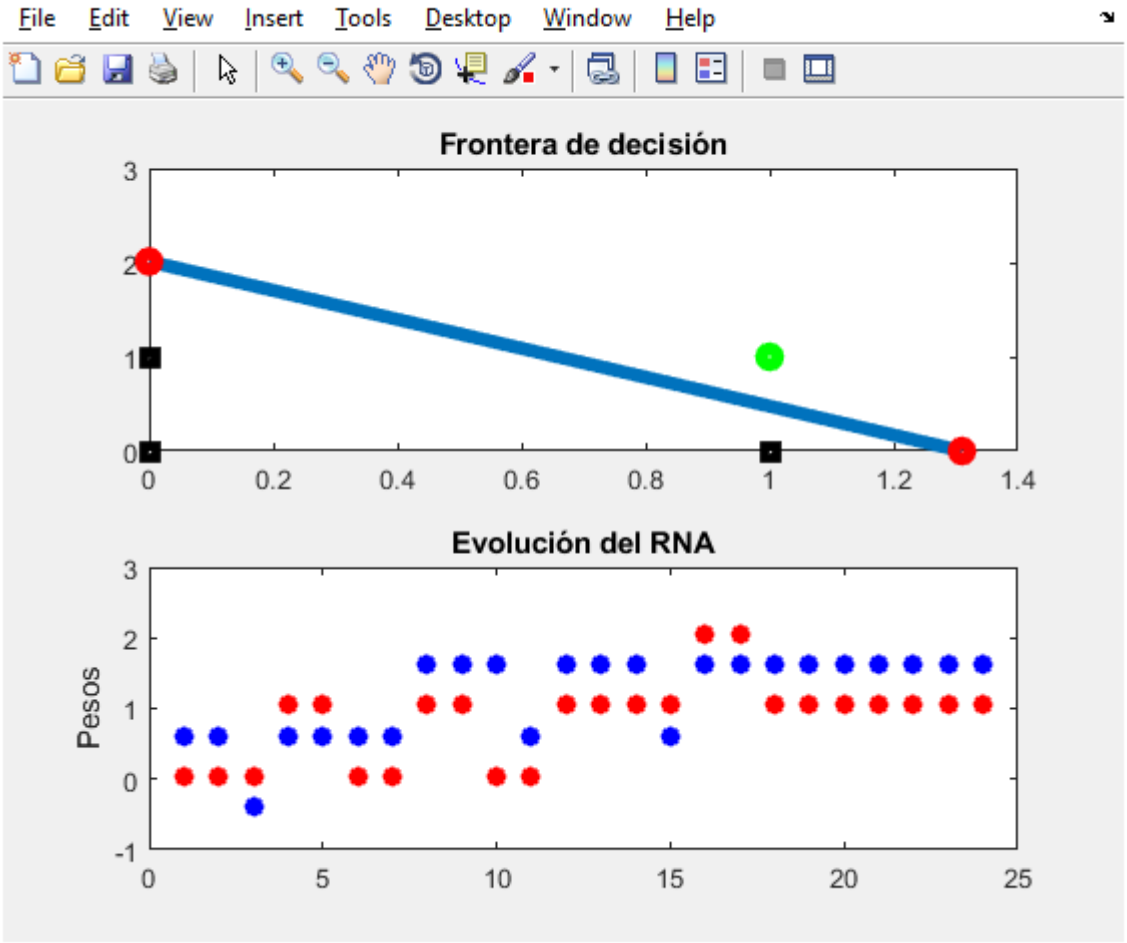
Para los targets:

t1	t2	t3	t4
0	0	0	1

Elegimos el método,"1" para método gráfico y "2" para aprendizaje. En este caso, método gráfico y el programa no arroja:



Volvemos a correr el programa, pero ahora seleccionamos aprendizaje:



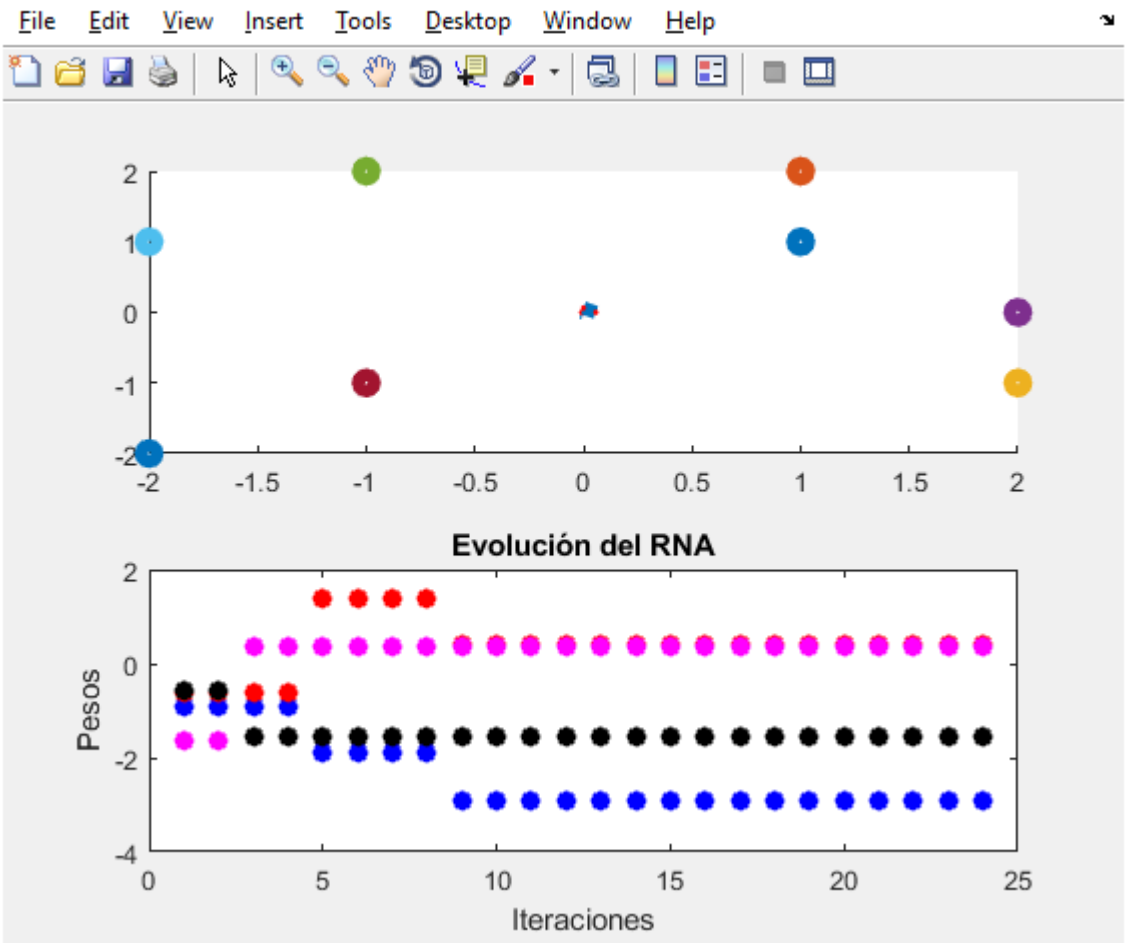
Como se puede observar se separan de manera correcta y también nos muestra la evolución de los pesos del RNA.

Otro ejemplo de regla de aprendizaje, ahora se ingresarán las siguientes matrices:

P1	P2	P3	P4	P5	P6	P7	P8
1	1	2	2	-1	-2	-1	-2
1	2	-1	0	2	1	-1	-2

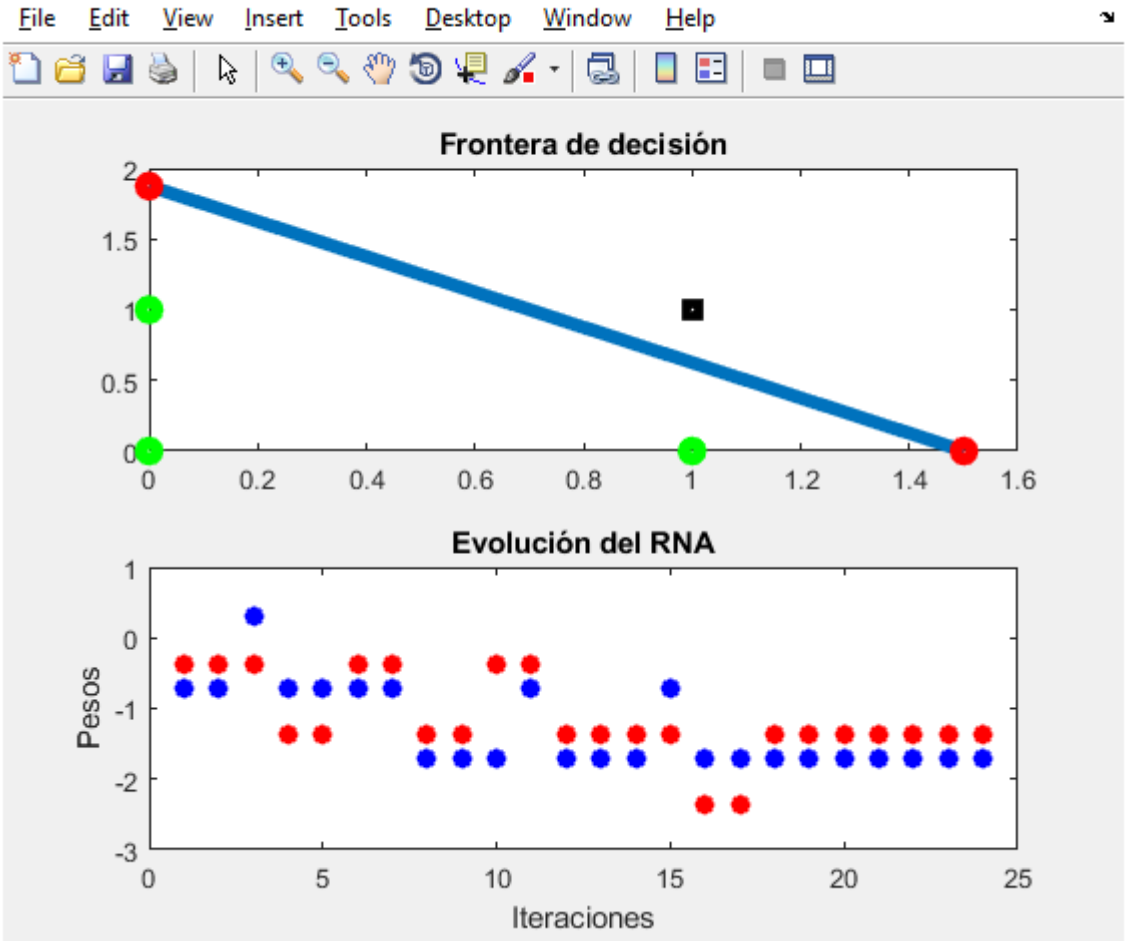
t1	t2	t3	t4	t5	t6	t7	t8
0	0	0	0	1	1	1	1
0	0	1	1	0	0	1	1

NOTA: No es necesario indicarle manualmente al programa a que clase pertenece cada vector de entrada.



En este caso, logra separar las clases, pero no se aprecia en la gráfica. Para confirmar que la neurona funciona el programa imprime una matriz que muestra los valores de error durante la evolución.

Otro ejemplo, simulando una compuerta OP de dos entradas:



Discusión de resutlados.

Como se observa en las gráficas es muy complicado tener resultados correctos con el método gráfico, aunque a simple vista sea sencillo separa las diferentes clases. Con aprendizaje la red funciona muy bien y separa las clases de manera correcta en pocas iteraciones.

Conclusiones.

Con esta práctica comprendí el comportamiento de la Red perceptron y sus reglas de aprendizaje vistas en clase, en cambio, el método grafico no resulto tan factible al momento de usarlo, aunque arrojo resultados correctos es muy complicado basar el aprendizaje de una red perceptron por la gran inexactitud.

Código:

```
clear;
clc;
p = double(dataset('File', 'matriz.txt'));
t = double(dataset('File' , 'targets.txt'));
[x,y] = size(p);
[x2,y2] = size(t);


fprintf('¿Cómo desea resolver, método grafico o aprendizaje?\n');
fprintf('1.Método grafico \n2.Aprendizaje\n')
D=input('');

if x2==1
if D==1
 a=zeros(1,y);

W=2*rand(1,x)-1;
b=2*rand(1)-1;

for intentos = 1:10
for i=1:y
    a(i)=hardlim(W*p(:,i)+b);
        if a(i)~=t(i)
            W=2*rand(1,x)-1;
            b=2*rand(1)-1;
            break
        end
end
    if a(:)==t(:)
        break
    end
end

for r = 1:y
    if t(r)==1

        plot(p(1,r),p(2,r),'go','LineWidth',5)
title('Frontera de decisión');
        hold on
    end
    if t(r)==0
         plot(p(1,r),p(2,r),'ks','LineWidth',5)
         hold on
    end
end

P1=[0 -b/W(1)];
P2=[-b/W(2) 0];
line(P1,P2, 'LineWidth',5)
hold on
plot(P1,P2,'ro','LineWidth',5)

end

if D==2
    j=1;
```

```
    q=1;
    e(1)=1;

W=2*rand(1,x)-1;
b=2*rand(1)-1;

for Epocas=1:50
for i=1:y
    a(i)=hardlim(W*p(:,i)+b);
    e(i)=t(i)-a(i);
    W=W+e(i)*p(:,i)';


     subplot(2,1,2)
     plot(j,W(1),'b*','LineWidth',5)
     title('Evolución del RNA');
     ylabel('Pesos');
     xlabel('Iteraciones');
     hold on
     plot(j,W(2),'r*','LineWidth',5)
     j=j+1;

    b=b+e(i);
end

    if e(:)==0
        break
    end
end

for r = 1:y
    if t(r)==1
        subplot(2,1,1)
        plot(p(1,r),p(2,r),'go','LineWidth',5)
        title('Frontera de decisión');
        hold on
    end
    if t(r)==0
        subplot(2,1,1)
         plot(p(1,r),p(2,r),'ks','LineWidth',5)
         hold on
    end
end

P1=[0 -b/W(1)];
P2=[-b/W(2) 0];
line(P1,P2, 'LineWidth',5)
hold on
plot(P1,P2,'ro','LineWidth',5)


end
end


if x2~=1

    if D==1

clases=0;
```

```
    W=2*rand(x)-1;
    b=2*rand(x,1)-1;

     for i2 = 1:y2
        if i2+1>y2
             break;
         end
     if t(:,i2)==t(:,i2+1)
         clases=clases+1;
         if i2>y2
             break;
         end
     end
    end
 S=log2(clases);
 W=2*rand(x)-1;
  b=2*rand(x,1)-1;
for Epocas = 1:10
for i=1:y

    a(:,i)=hardlim(W*p(:,i)+b);
    if dot(W(1,:),W(2,:))~=0
    W=2*rand(x)-1;
    b=2*rand(x,1)-1;
    end
    if dot(W(1,:),W(2,:))==0
break;
    end
end
end


W2=inv(W);
for linea=1:S
    P1=[0 (-b(linea)*W2(2,:))];
    P2=[(-b(linea)*W2(1,:)) 0];

    line(P1(1:2),P2(2:3), 'LineWidth',5)
    hold on
    plot(P1(1:2),P2(2:3),'r.','LineWidth',5)
    hold on
end
for punto=1:y

    plot(p(1,punto),p(2,punto),'o','LineWidth',5)
    title('Frontera de decisión');
    hold on
end
    end



    if D==2
    clases=0;
    j=1;
    q=1;



 for i2 = 1:y2
        if i2+1>y2
```

```
            break;
        end
    if t(:,i2)==t(:,i2+1)
        clases=clases+1;
        if i2>y2
            break;
        end
    end
end
S=log2(clases);


    W=2*rand(x)-1;
    b=2*rand(x,1)-1;
for Epocas=1:100
for i=1:y
     e(:,i)=t(:,i)-hardlim(W*p(:,i)+b);
      W=W+e(:,i)*p(:,i)';
        b=b+e(:,i);

     subplot(2,1,2)
     plot(j,W(1,1),'b*','LineWidth',5)
     title('Evolución del RNA');
     ylabel('Pesos');
     xlabel('Iteraciones');
     hold on
     plot(j,W(1,2),'r*','LineWidth',5)
     hold on
     plot(j,W(2,1),'m*','LineWidth',5)
     hold on
     plot(j,W(2,2),'k*','LineWidth',5)
     j=j+1;
end

if e(:,:)==0
    break;
end

end

W2=inv(W);

for linea=1:S
    P1=[0 (-b(linea)*W2(2,:))];
    P2=[(-b(linea)*W2(1,:)) 0];

    subplot(2,1,1)
    line(P1(1:2),P2(2:3), 'LineWidth',5)
    hold on
    plot(P1(1:2),P2(2:3),'r.','LineWidth',5)
end

for punto=1:y
    subplot(2,1,1)
    plot(p(1,punto),p(2,punto),'o','LineWidth',5)
    hold on
end
if e(:,:)==0
fprintf('Es posible que no se aprecie gráficamente las fronteras de decisión, así
que se muestra la matriz en la que se guardó los valores del error durante el
proceso');
```

```
fprintf('\ne: [') ;
fprintf(' %d ', e) ;
fprintf(']\n') ;
end
    end

end
```

Referencias.

Neural Network Design. [Martin T. Hagan, Howard B, Demuth, Mark Beale – PWS Publishing Company]

Redes neuronales

Algoritmos, aplicaciones y tecnicas de programación, James A. Freeman, David M. Skapura, ADDISON-WESLEY, 1993