UNIVERSIDAD AUTÓNOMA DEL ESTADO DE MÉXICO

CENTRO UNIVERSITARIO UAEM ZUMPANGO

INGENIERÍA EN COMPUTACIÓN

Unidad de aprendizaje:
L41086 Redes Neuronales
Unidad de Competencia II

Tema:
El perceptrón

DR. ASDRÚBAL LÓPEZ CHAU

Información de la Unidad de Aprendizaje (UA)

En este material didáctico, se abordan los conocimientos de la **unidad de competencia II** de la UA L41086 Redes Neuronales.

Información de la Unidad de Aprendizaje (UA)

Propósito de la UA

Comprender y aplicar los principios y técnicas fundamentales de los modelos de redes neuronales comunes. Además de analizar y aplicar los principales métodos de aprendizaje en las redes neuronales y sus aplicaciones prácticas.

Información de la Unidad de Aprendizaje (UA)

Objetivo de la Unidad de Competencia I:

Comprender y analizar los modelos básicos de redes neuronales artificiales con conexión hacia adelante

Información de la Unidad de Aprendizaje (UA)

Conocimientos Unidad de Competencia II:

- El perceptrón (funciones de activación y aprendizaje)
- Problemas linealmente separables y no separables
- Aplicaciones del perceptrón
- Redes ADALINE y MADALINE (regla delta Widrow-Hoff y regla mínimo error cuadrático medio, ALC, aprendizaje)
- Aplicaciones de ADALINE

Algoritmo de entrenamiento del Perceptrón Simple

Enfoque de la explicación

- Se explicará el algoritmo de entrenamiento del perceptrón simple al mismo tiempo que se aplica a un ejemplo sencillo de dos entradas.

Entradas del algoritmo

- Vectores de entrada

$$X = \{x_i \in R^d, i = 1...N\}$$

- Vector de etiquetas

$$Y = \{y_i \in \{1, 0\}, i = 1...N\}$$

Entradas del algoritmo, ejemplo

$$X = \{x_i \in R^d, i = 1...N\}$$

X =

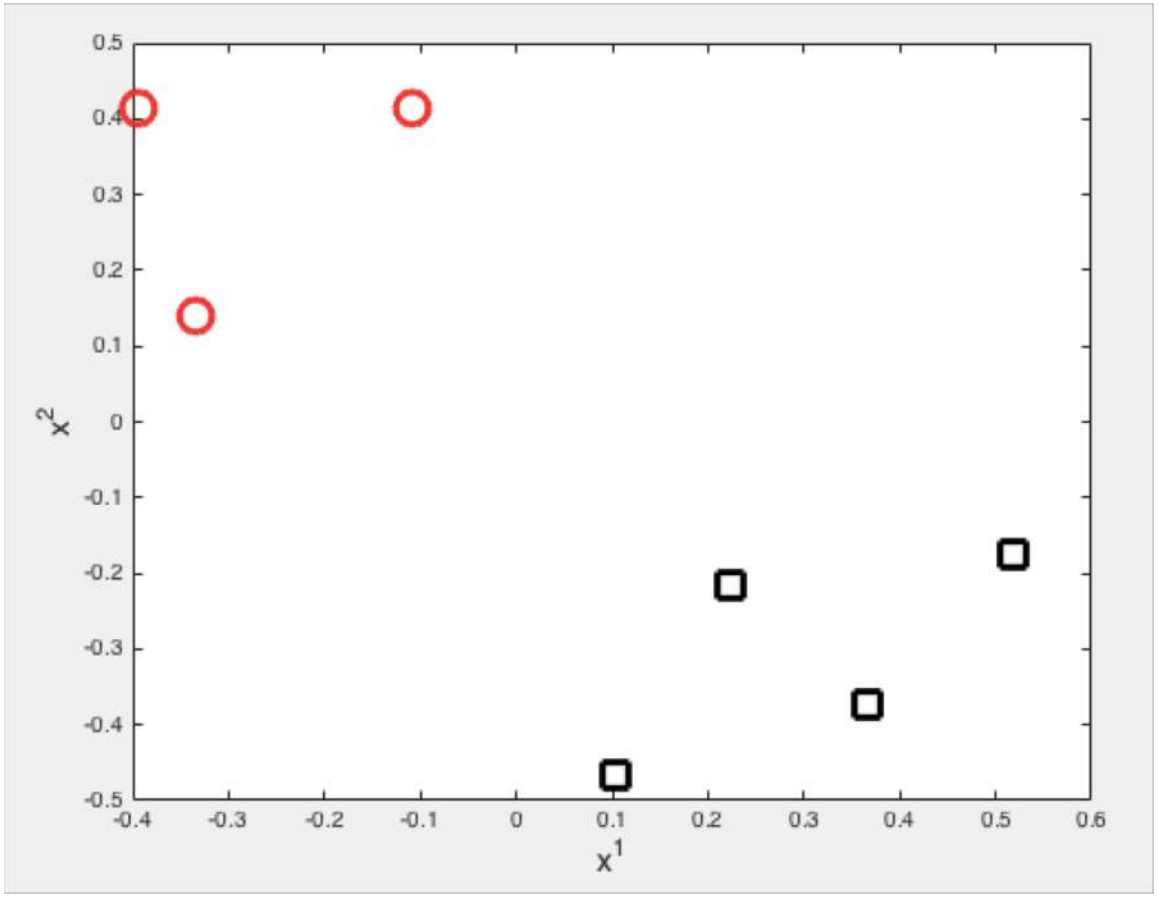
x^1	x^2
-0.1083	0.4140
-0.3940	0.4140
-0.3341	0.1399
0.5184	-0.1749
0.2235	-0.2157
0.3664	-0.3732
0.1037	-0.4665

$$Y = \{y_i \in \{1, 0\}, i = 1...N\}$$

Y =

1
1
1
0
0
0
0

Entradas del algoritmo, ejemplo



X		Y
x^1	x^2	
-0.1083	0.4140	1.0000
-0.3940	0.4140	1.0000
-0.3341	0.1399	1.0000
0.5184	-0.1749	0
0.2235	-0.2157	0
0.3664	-0.3732	0
0.1037	-0.4665	0

Figura 1. Vectores de entrada

Perceptrón de ejemplo

El perceptrón que permite clasificar las instancias es el mostrado en la Figura 2.

Es necesario determinar los valores del vector w:

$$\omega = \begin{pmatrix} \omega_o \\ \omega_1 \\ \omega_2 \end{pmatrix}$$

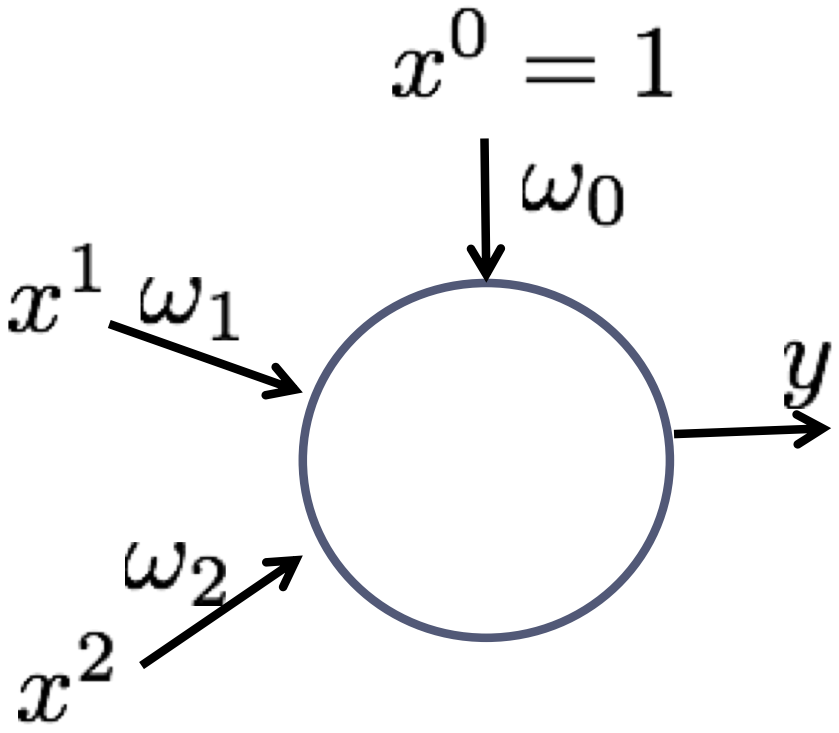
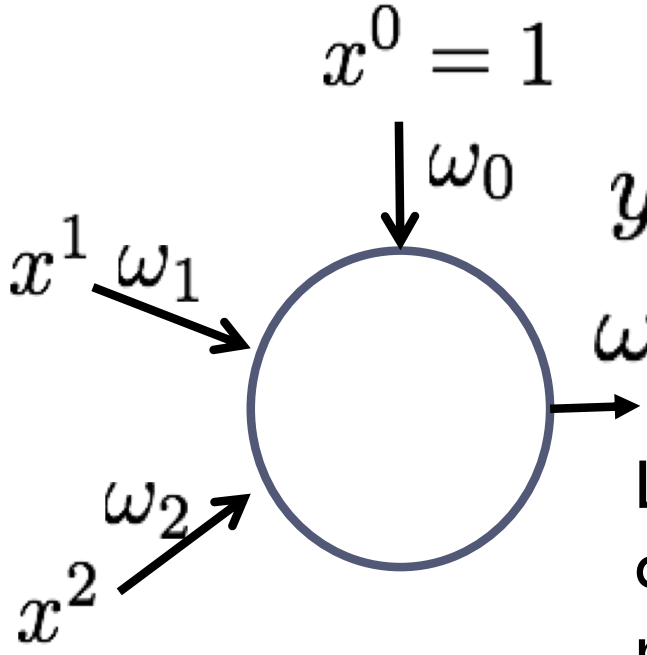


Figura 2. Perceptrón simple de dos entradas

Respuesta del perceptrón

- Para calcular la respuesta cada entrada del perceptrón se multiplica por el correspondiente peso sináptico:



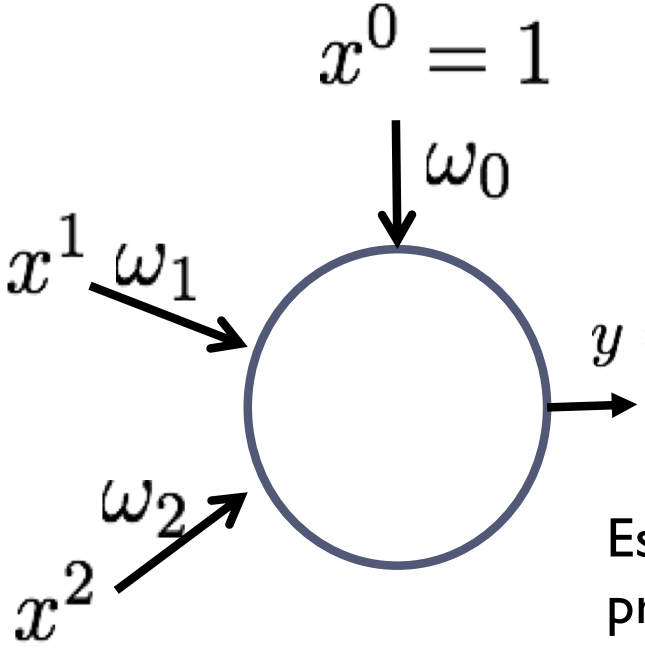
$$y = \omega^T x_i$$

$$\omega^T x_i = \omega_0 x_0 + \omega_1 x_1 + \omega_2 x_2$$

La respuesta del perceptrón se calcula de acuerdo a la siguiente regla:

$$y = \begin{cases} 1 \text{ si } \omega^T x_i \geq 0 \\ 0 \text{ otro caso} \end{cases}$$

Respuesta del perceptrón



$$y = \begin{cases} 1 \text{ si } \omega^T x_i \geq 0 \\ 0 \text{ otro caso} \end{cases}$$

Esta respuesta del perceptrón es usada para predecir la clase o categoría del vector que se le presente en las entradas.

Respuesta del perceptrón

Para observar las regiones del espacio en las que el perceptrón asigna cada clase, generaremos 100 vectores de manera pseudo aleatoria.

Presentaremos de color rojo los vectores para los cuales la salida y=1 y de color azul los vectores para los cuales la salida y es diferente a 1.

Supongamos que

$$\omega = \begin{pmatrix} -0.2661 \\ 0.5943 \\ -0.9763 \end{pmatrix}$$

La respuesta del perceptrón es

$$y = \begin{cases} 1 \text{ si } \omega^T x_i \geq 0 \\ 0 \text{ otro caso} \end{cases}$$

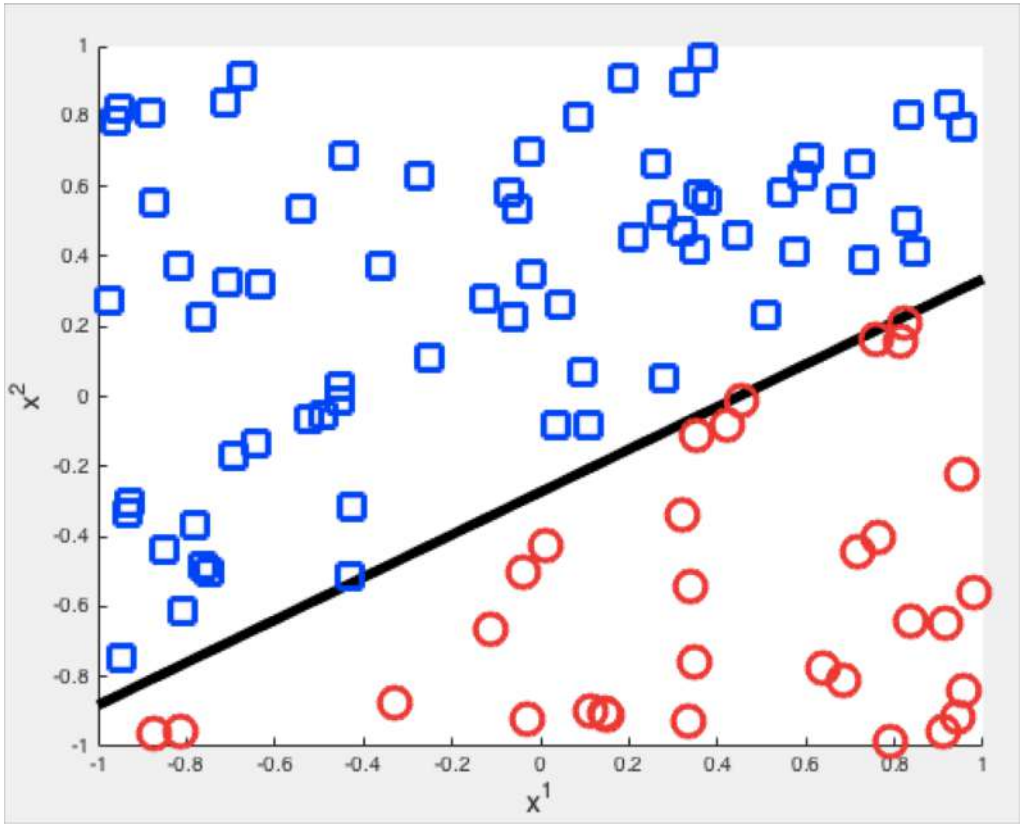
Respuesta del perceptrón

Supongamos que

$$\omega = \begin{pmatrix} -0.2661 \\ 0.5943 \\ -0.9763 \end{pmatrix}$$

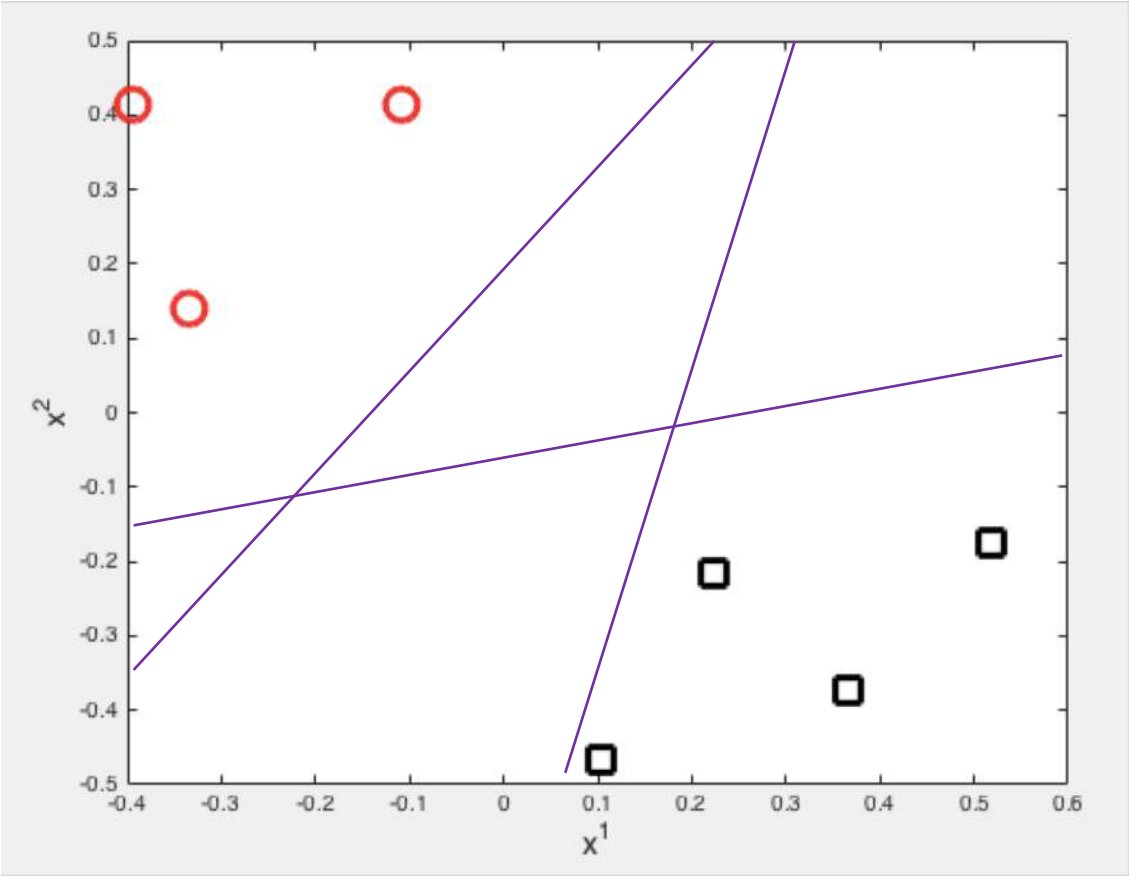
Respuesta del perceptrón

$$y = \begin{cases} 1 \text{ si } \omega^T x_i \geq 0 & \text{Rojo} \\ 0 \text{ otro caso} & \text{Azul} \end{cases}$$



Si generamos infinitos vectores, se observará una frontera de decisión que separa los vectores de un tipo y el otro

Fronteras de decisión



$$\omega = \begin{pmatrix} \omega_o \\ \omega_1 \\ \omega_2 \end{pmatrix}$$

Dependiendo del valor de los componentes de W, será la frontera de decisión que se genere. La Figura 3 muestra algunas de las infinitas fronteras posibles.

Figura 3. Algunas fronteras de decisión

Inicio del algoritmo

- Agregar un vector de 1's a la matriz X

X=

1.0000	−0.1083	0.4140
1.0000	−0.3940	0.4140
1.0000	−0.3341	0.1399
1.0000	0.5184	−0.1749
1.0000	0.2235	−0.2157
1.0000	0.3664	−0.3732
1.0000	0.1037	−0.4665

- Generar un vector inicial de pesos sinápticos, cada componente puede ser cero o algún valor aleatorio

$$\omega = \{w_i, i = 1, ...d + 1\}$$

NOTA: Generalmente, el valor inicial de W produce una frontera de decisión incorrecta.

Inicio

NOTA: Generalmente, el valor inicial de W produce una frontera de decisión incorrecta.

Ejemplo:

$$W = \{\omega_i, i = 1..d+1\}$$

```
W =

    0.9042
    0.4980
    0.5135
```

*Valores generados de manera pseudoaleatoria

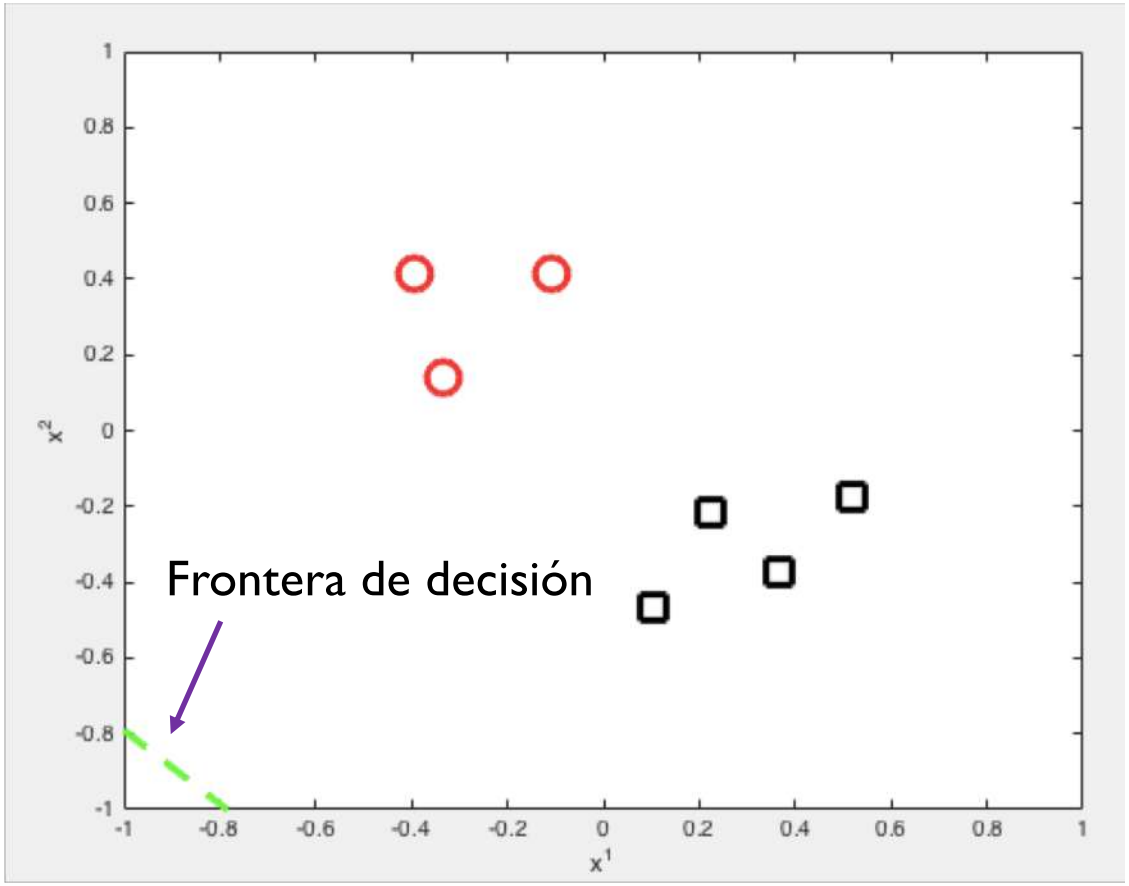


Figura 4. Frontera de decisión generada por el vector W

Ciclo principal del algoritmo

Algorithm 1: Entrenamiento Perceptrón

Input: ω inicial, X, Y

Output: Neurona entrenada

```
1 foreach x_i ∈ X do
2     if ω^T x_i ≥ 0 then
3         y=1
4     else
5         y=0
6     ω = ω + (y_i − y)x_i
```

Ciclo principal del algoritmo, explicación

Algorithm 1: Entrenamiento Perceptrón

Input: ω inicial, X, Y

Output: Neurona entrenada

```
1 foreach x_i ∈ X do
2     if ω^T x_i ≥ 0 then
3         y=1
4     else
5         y=0
6     ω = ω + (y_i − y)x_i
```

Todos los vectores son probados

Calcula la respuesta y determina el valor de la salida (predicción)

Actualiza el vector de pesos sinápticos

Actualización de los pesos sinápticos

En la expresión:

$$\omega = \omega + (y_i - y)x_i$$

El vector w se actualiza de acuerdo a lo siguiente:

$$y_i - y = \begin{cases} 0, \text{la predicción es correcta para vector } x_i \\ 1, \text{la predicción es incorrecta para vector } x_i \\ -1, \text{la predicción es incorrecta para vector } x_i \end{cases}$$

Actualización de los pesos sinápticos

En la expresión:

$$\omega = \omega + (y_i - y)x_i$$

$$y_i - y = \begin{cases} 0, \text{No actualizar } \omega \\ 1, \text{Actualizar } \omega \text{ en dirección de } x_i \\ -1, \text{Actualizar } \omega \text{ en dirección contraria a } x_i \end{cases}$$

Algoritmo completo

- El ciclo principal se repite N veces

Input: ω inicial, X, Y, N: Número de iteraciones
Output: Neurona entrenada

```
1 for j=1:N do
2     foreach x_i ∈ X do
3         if ω^T x_i ≥ 0 then
4             y=1
5         else
6             y=0
7         ω = ω + (y_i − y)x_i
```

Ejemplo

```
X =

   -0.1083    0.4140
   -0.3940    0.4140
   -0.3341    0.1399
    0.5184   -0.1749
    0.2235   -0.2157
    0.3664   -0.3732
    0.1037   -0.4665
```

```
Y =

     1
     1
     1
     0
     0
     0
     0
```

Ejemplo

Inicial

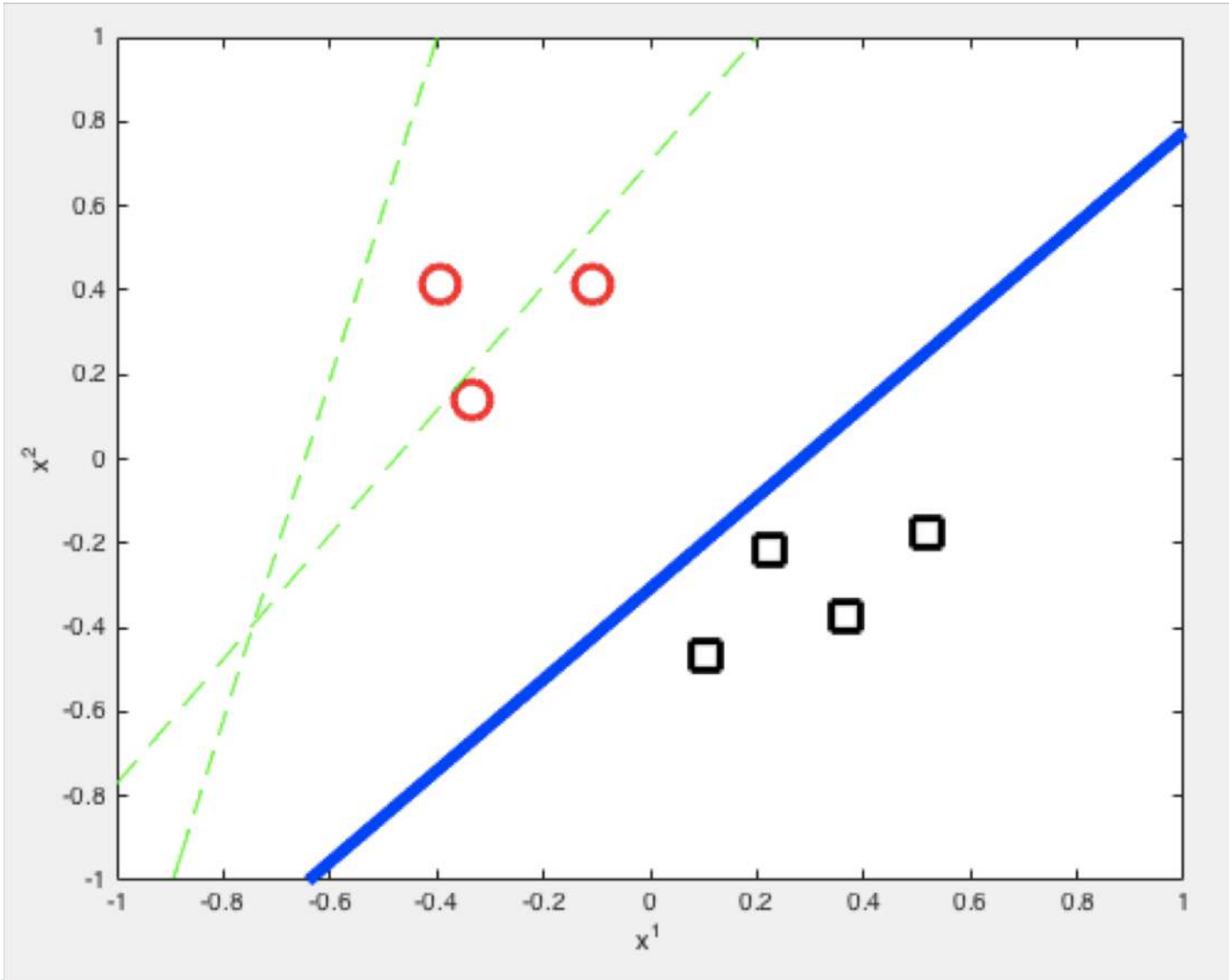
```
W =

    0.3912
   -0.4229
    0.0582
```

Final

```
W =

    0.3912
   -1.3814
    1.2769
```



Ejemplo

Inicial

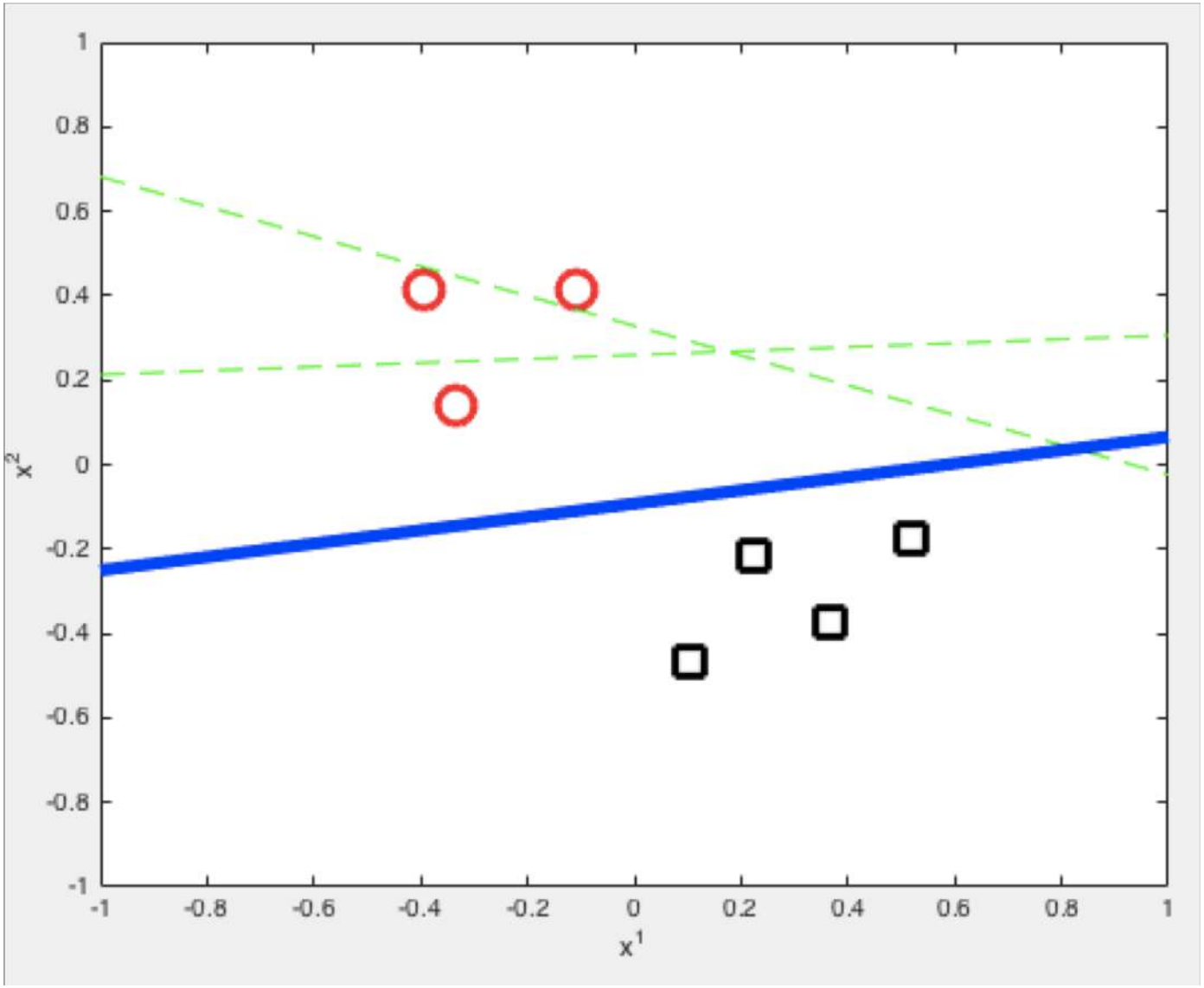
```
W =

    0.2717
    1.8577
    1.6851
```

Final

```
W =

    0.2717
   -0.4649
    2.9446
```



Graficando la frontera de decisión

Código completo en Matlab