Trabajo Práctico N° 3: Perceptrón simple y multicapa

Grupo 10

Implementación

Implementación

Perceptrón Simple

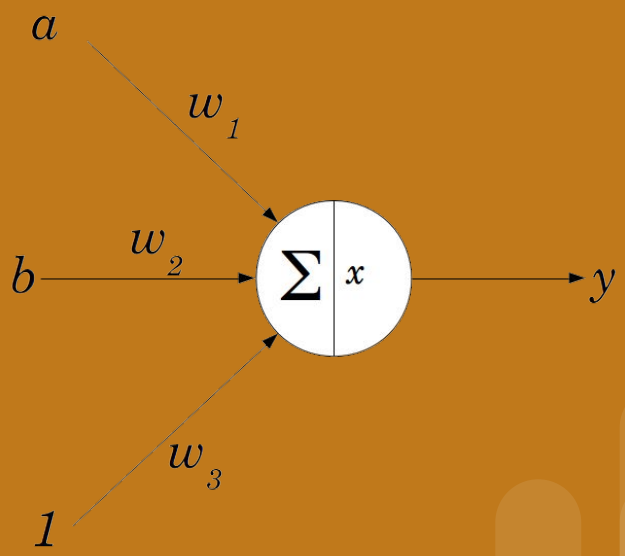
-Función de activación escalón

-Función de activación lineal

-Funciones de activación sigmoideas (tangente hiperbólica, logística)

-Normalización de datos

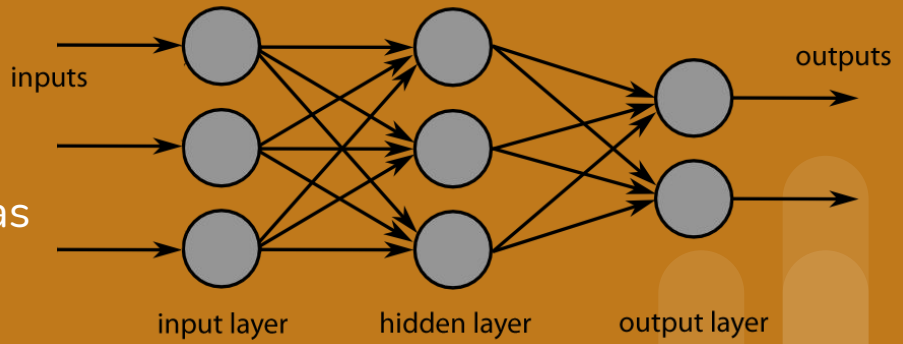
-Inicialización de pesos en 0



Implementación

Perceptrón Multicapa

- Función de activación lineal
- Funciones de activación sigmoideas
- Método de gradiente descendiente
- Variabilidad de capas y número de neuronas
- η adaptativo
- Inicialización de pesos entre -1 y 1



Aplicación

Aplicación - Ej 1

Función logística AND

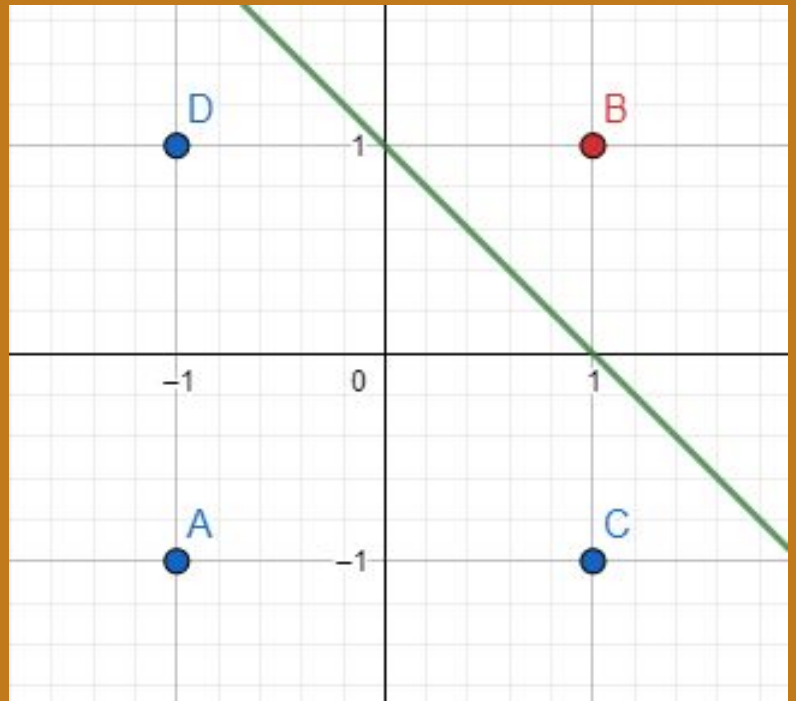
Entrada	Salida esperada
(-1,1)	-1
(1,-1)	-1
(-1,-1)	-1
(1,1)	1

- Utilizando $\eta = 0.01$
- Error 0 (calcula perfectamente el hiperplano)
- Logra separar las clases utilizando activación step
- Finalización en 2 epochs
- Pesos:
-

  w0: -0.02 w1: 0.02 w2: 0.02

Aplicación - Ej 1

Función logística AND



Utilizando los pesos para calcular el hiperplano:

w1 * x + w2 * y + w0 = 0

0.02x+0.02y-0.02=0

equivalente a

y = -x+1

Aplicación - Ej 1

Función logística XOR

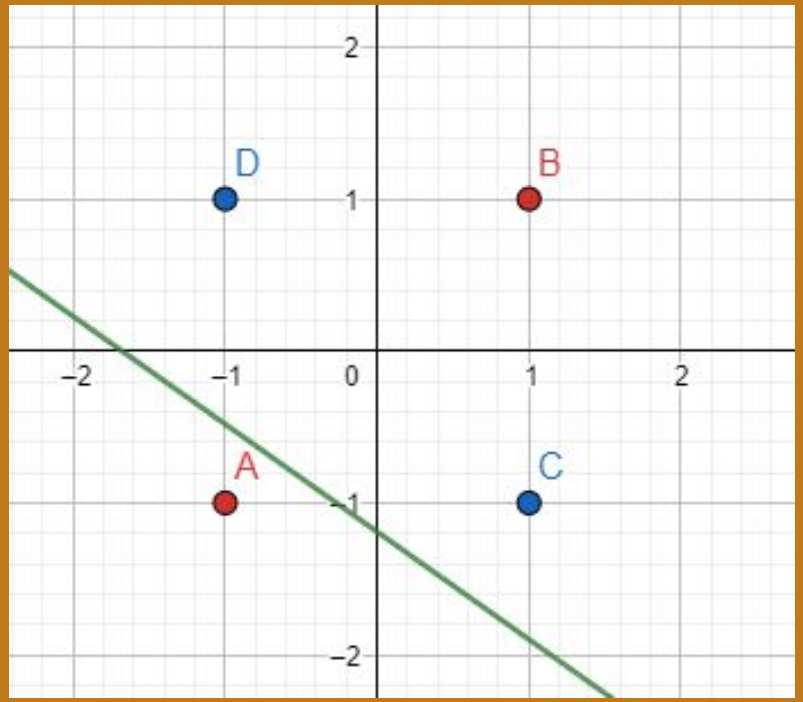
Entrada	Salida esperada
(-1,1)	1
(1,-1)	1
(-1,-1)	-1
(1,1)	-1

- Utilizando $\eta = 0.01$
- Error 2 (finaliza por máximo de 10000 epochs).
- No logra separar las clases adecuadamente.
- Pesos:

  w0: 0.03802 w1: 0.0226 w2: 0.03192

Aplicación - Ej 1

Función logística XOR



Utilizando los pesos para calcular el hiperplano:

$w1 * x + w2 * y + w0 = 0$

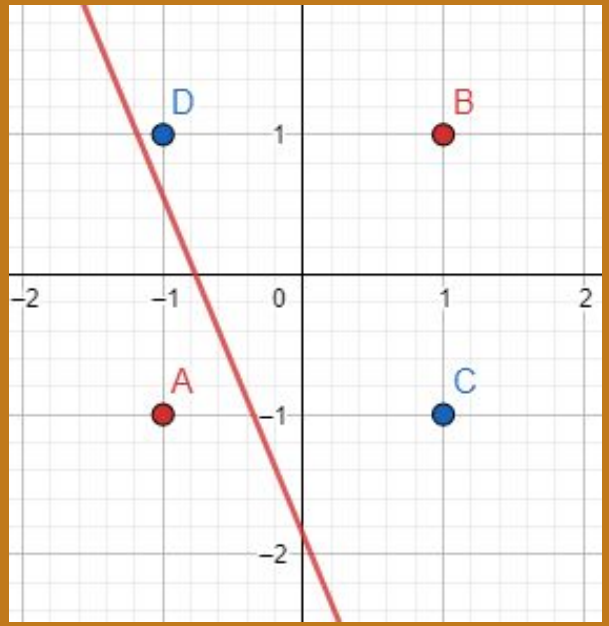
$0.0226x + 0.03192y + 0.03802 = 0$

No separa las clases correctamente

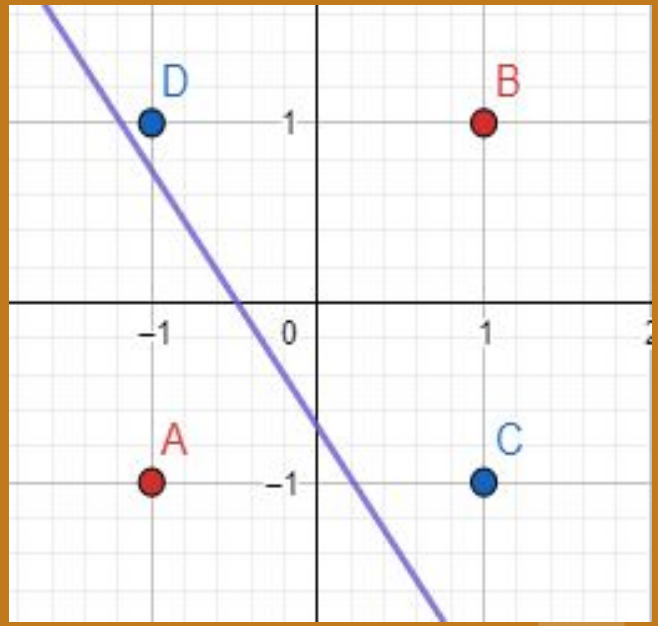
Aplicación - Ej 1

Función logística XOR

Con $\eta = 0.05$



Con $\eta = 0.1$



Aplicación - Ej 1

Función logística XOR - Explicación

Entrada	Salida esperada
(-1,1)	1
(1,-1)	1
(-1,-1)	-1
(1,1)	-1

El perceptrón simple con activación escalón resuelve problemas LINEALMENTE SEPARABLES. XOR no lo es.

i) -1*w1 + 1*w2 + w0 = 1

ii) 1*w1 - 1*w2 + w0 = 1

iii)-1*w1 - 1*w2 + w0 = -1

iv)1*w1 + 1*w2 + w0 = -1

i) + ii): 2*w0 = 1

ii) + iv) 2*w0 = -1 Absurdo!

Aplicación - Ej 2

Perceptrón simple lineal

Aproximación de función normalizando input y output en [0,1].

-Utilizando η = 0.01, epsilon de error 0.001.

-Función de activación f(x) = x.

-Corte por máximo de epochs 5000.

-Error mínimo: 0.95.

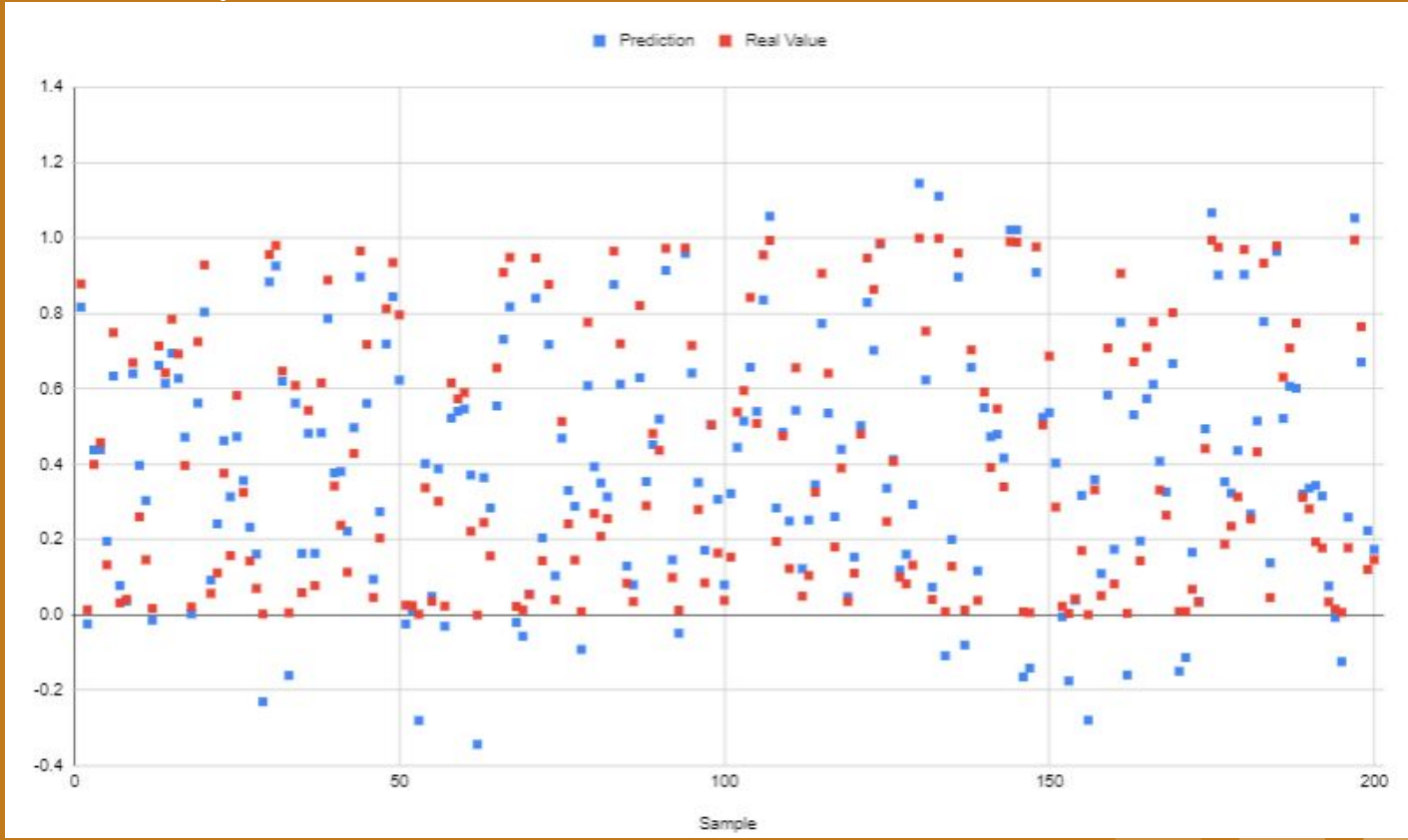
-Pesos:

w0: 0.41947 w1: 0.07020 w2: 0.05527 w3: 0.06918

Aplicación - Ej 2

Perceptrón simple lineal

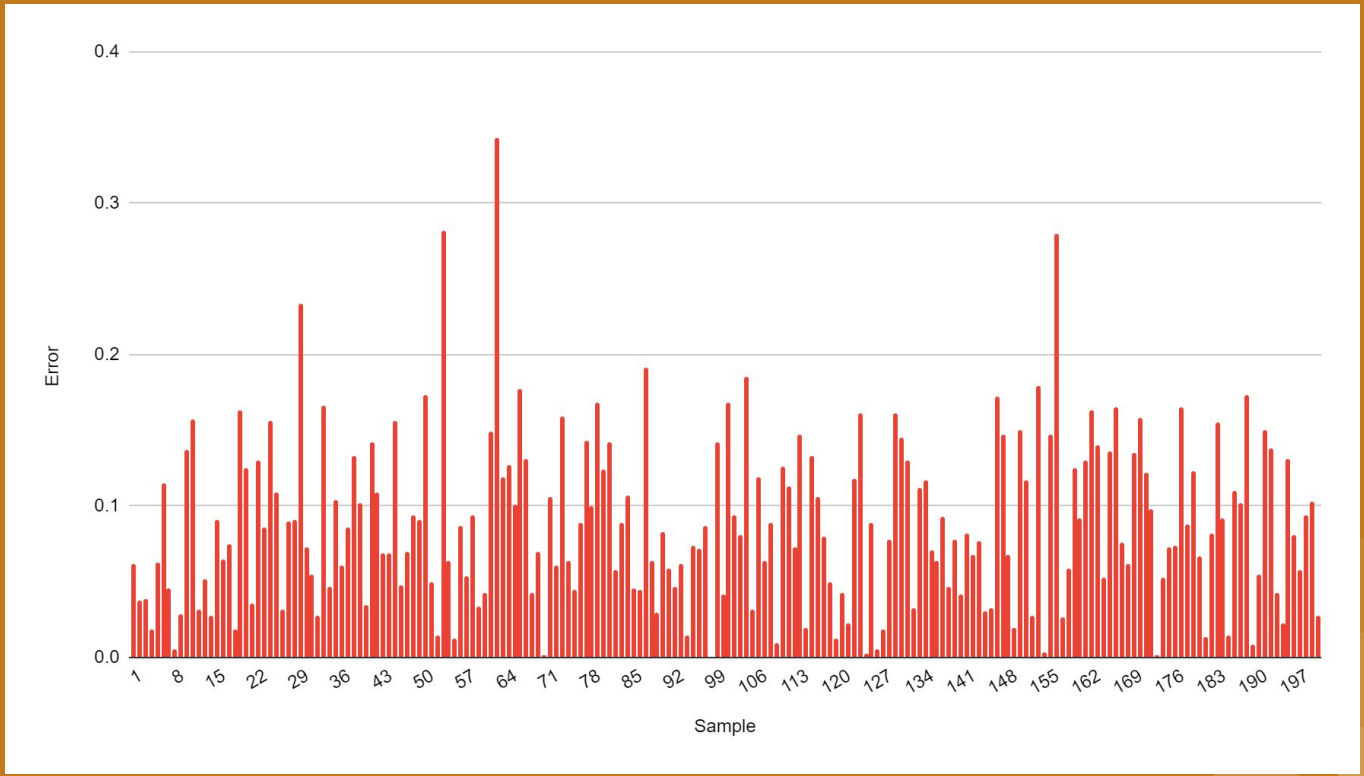
Función: 0.07020x + 0.05527y + 0.06918z + 0.41947 = 0



Aplicación - Ej 2

Perceptrón simple lineal

Errores locales



Aplicación - Ej 2

Perceptrón simple no lineal

Aproximación de función normalizando input y output en [0,1].

-Utilizando η = 0.01, epsilon de error 0.001.

-Función de activación $f(x,\beta) = 1/(1+\exp(-2\beta x))$ con $\beta=1$.

-Corte por alcance de epsilon de error en 21 epochs.

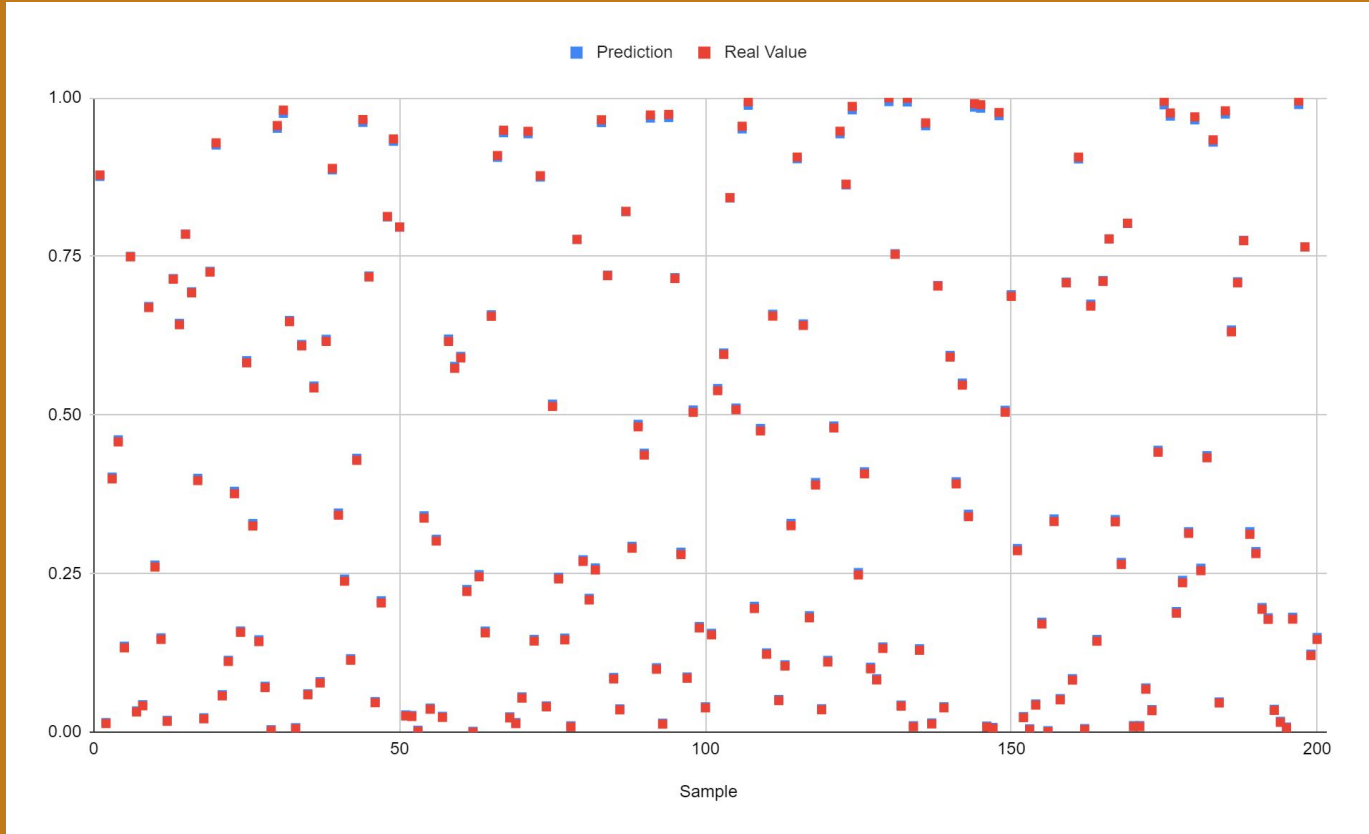
-Error mínimo: 8.53E-4.

-Pesos:

w0: -0.23911 w1: 0.25094 w2: 0.25108 w3: 0.25081

Aplicación - Ej 2 Perceptrón simple no lineal: aprendizaje

Función: 0.25094x + 0.25108y + 0.25081z - 0.23911 = 0



Aplicación - Ej 2

Perceptrón simple no lineal - Generalización

Training set	Testing set	Training error	Generalization error
30%	70%	9.82E-04	3.11265
50%	50%	9.54E-04	2.22432
75%	25%	8.84E-04	1.0721
80%	20%	9.46E-04	0.84481
Outputs <0.5	Outputs>0.5	9.00E-04	2.49887

Menor error cuadrático medio

Aplicación - Ej 3

Perceptrón multicapa - XOR

- Utilizando η inicial 0.1, con η adaptativo.
- Utilizando 3 neuronas como capa oculta.
- Utilizando β=1.5
- Epsilon de error 0.001
- Epochs 4717 (máx 10000)
- Error cuadrático medio mín 9.92E-4
- Logra separar correctamente las clases

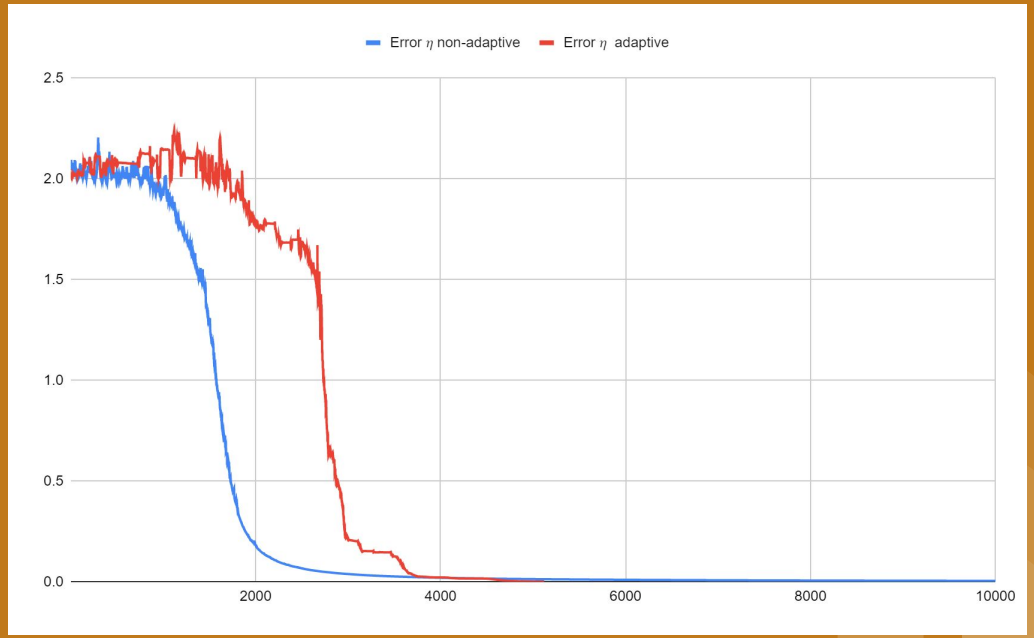
Input	Salida esperada	Predicción (> 0 clase 1, < 0 clase -1)
(-1, 1)	1	0.9809 [1]
(1 ,-1)	1	0.9796 [1]
(-1, -1)	-1	-0.9766 [-1]
(1, 1)	-1	-0.9741 [-1]

Aplicación - Ej 3

Perceptrón multicapa - XOR: Análisis η adaptativo

Además del método de gradiente decreciente, se consideró un ajuste del valor de η dependiendo del cambio en el error por epoch.

- Err $\uparrow$ $\rightarrow$ $\eta \downarrow$
- Err $\downarrow$ $\rightarrow$ $\eta \uparrow$



Aplicación - Ej 3

Perceptrón multicapa - XOR: Análisis cambio β

Utilizando la función sigmoidea tangente hiperbólica f(β,x) = tanh(β*x)

β	Err. cuadrático medio mín.
0.5	0.07475
1	0.01298
1.5	9.92E-4
5	1.0016

Aplicación - Ej 3

Perceptrón multicapa - Números pares

> 0 clase par, < 0 clase impar

Capacidad de predicción

- Utilizando η inicial 0.1, con η adaptativo.
- Utilizando 3 neuronas como capa oculta.
- Utilizando β=1.5
- Epsilon de error 0.001
- Epochs 2977 (máx 10000)
- Error cuadrático medio mín 9.99E-4
- Logra separar correctamente las clases

Número	Esperado	Obtenido
0	Par	0.984 [Par]
1	Impar	-0.987 [Impar]
2	Par	0.985 [Par]
3	Impar	-0.981 [Impar]
4	Par	0.987 [Par]
5	Impar	-0.995 [Impar]
...	...	...

Aplicación - Ej 3

Perceptrón multicapa - Números pares

Capacidad de generalización

- Se evaluó la capacidad de generalizar tomando sets de distintos tamaños, seleccionando los dígitos al azar.

- Se generó la matriz de confusión para los outputs de testeo.

- Se analizó la precisión del perceptrón.

Aplicación - Ej 3

Perceptrón multicapa - Números pares

Capacidad de generalización: Matriz de confusión

Tr:8 Test:2	Par	Impar
Exp:Par	0	1
Exp:Impar	0	1

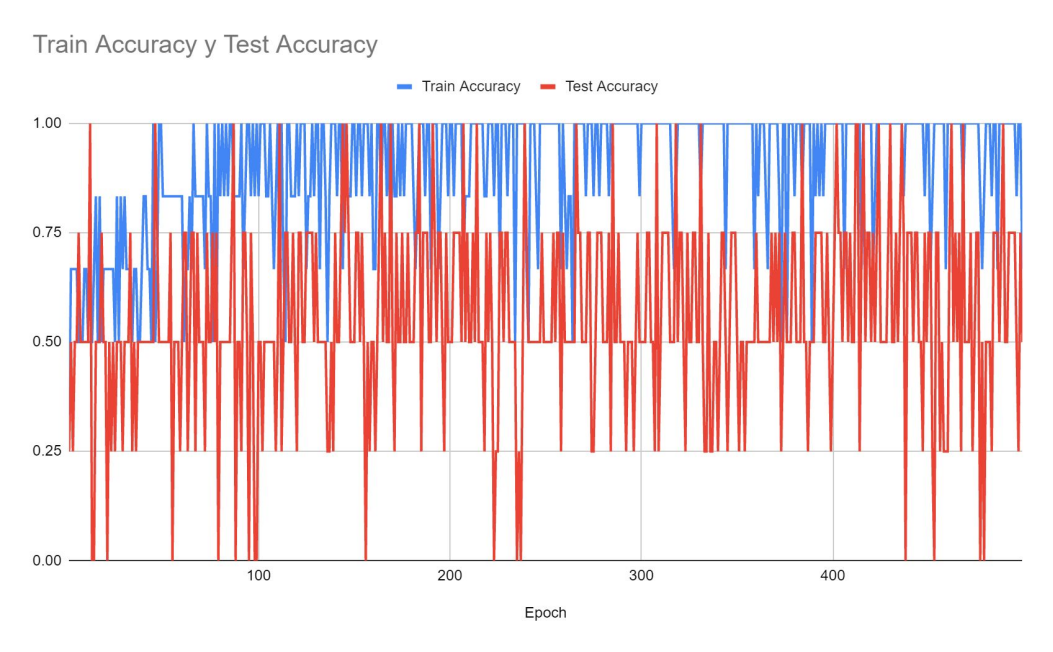
Tr:6 Test:4	Par	Impar
Exp:Par	2	1
Exp:Impar	1	0

Tr:5 Test:5	Par	Impar
Exp:Par	0	2
Exp:Impar	1	2

Aplicación - Ej 3

Perceptrón multicapa - Números pares

Capacidad de generalización: Accuracy



- Experimento realizado con # Training Set:6, #Testing Set: 4
- Como es de esperar, Training accuracy > Testing accuracy
- Testing accuracy es irregular. No generaliza correctamente.

Conclusiones

Conclusiones

- El conjunto de entrenamiento debería tener la mayor variedad posible, ya que esto afecta en el error general del perceptrón. Categorizar los inputs que se usarán para entrenar con algún criterio afecta negativamente al perceptrón.
- Elegir correctamente la función de activación y los parámetros (β, η) influye en la capacidad del perceptrón.
- Realizar experimentaciones para hallar los mejores parámetros da buenos resultados (combinaciones de función de activación y β, con distintos valores de η)
- η adaptativo optimiza la cantidad de epochs necesarios para un obtener un perceptrón con el mismo error.
- η adaptativo además puede alcanzar un error menor.
- En el caso del ejercicio 3 el perceptrón no puede generalizar a partir de los datos de entrada. Posiblemente, porque el input es una “imagen” y no logra “traducirla” a un número.
- Utilizar métricas como la accuracy permite evaluar el poder del perceptrón, en términos de training y de generalización.
- El método de gradiente descendiente es indispensable para alcanzar resultados válidos.