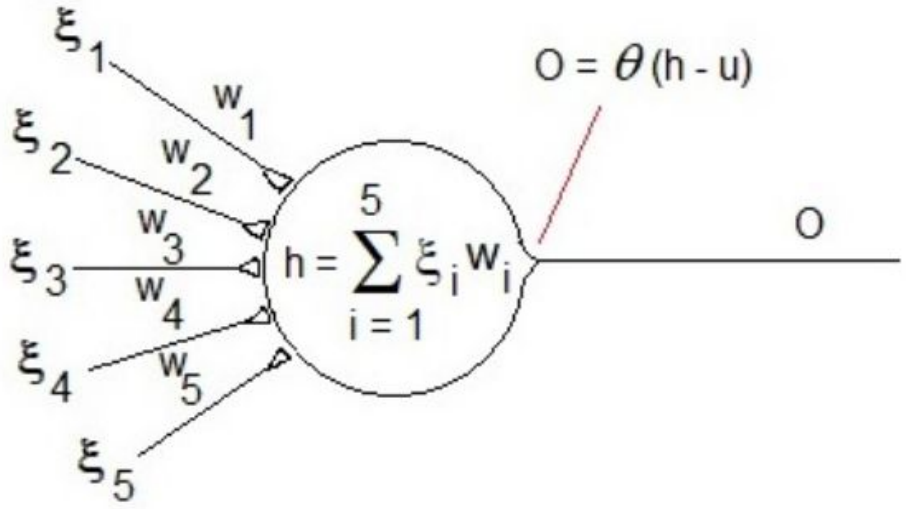
Trabajo práctico N°3: “Perceptrón simple y multicapa”

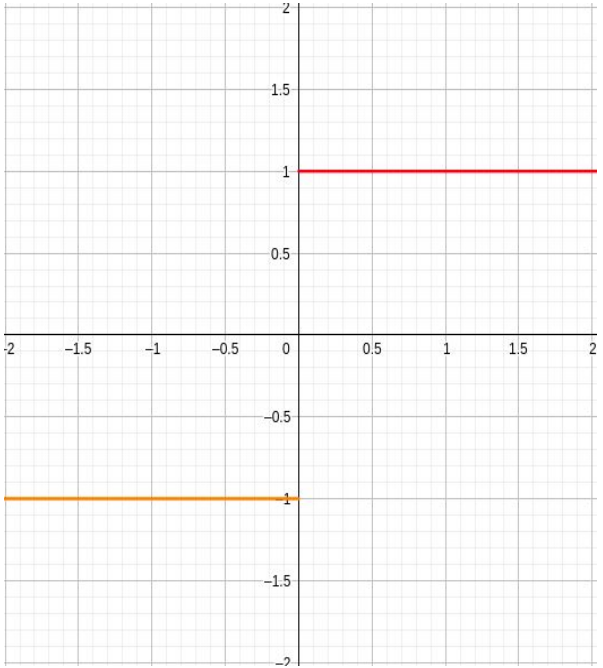
- Dizenhaus, Manuel - 61101
- De Simone, Franco - 61100
- Cornidez, Milagros - 61432

Ejercicio 1: Perceptrón simple con funciones “AND” y “XOR”

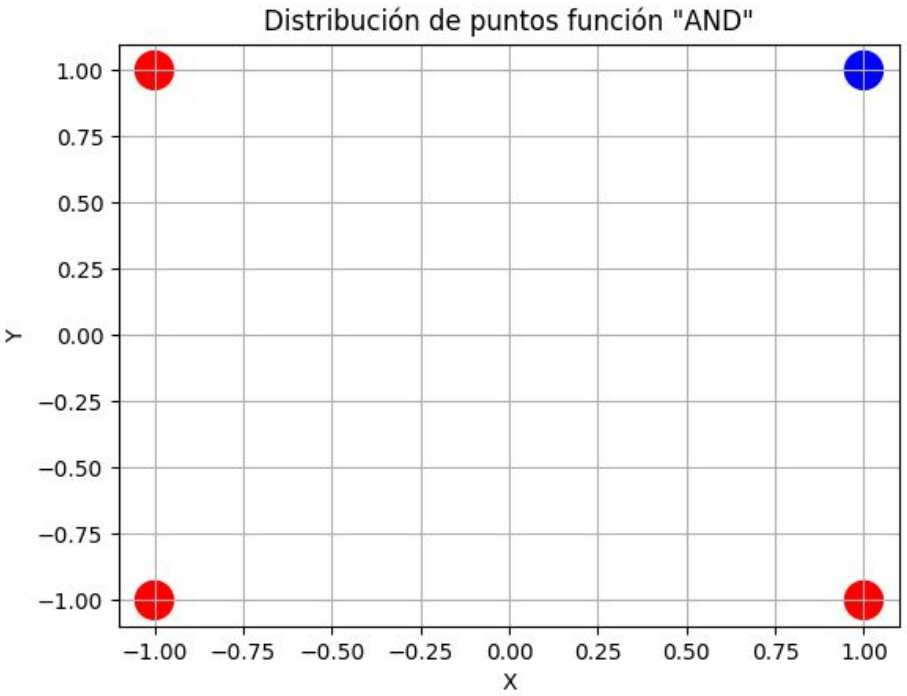
Perceptrón simple con función escalón

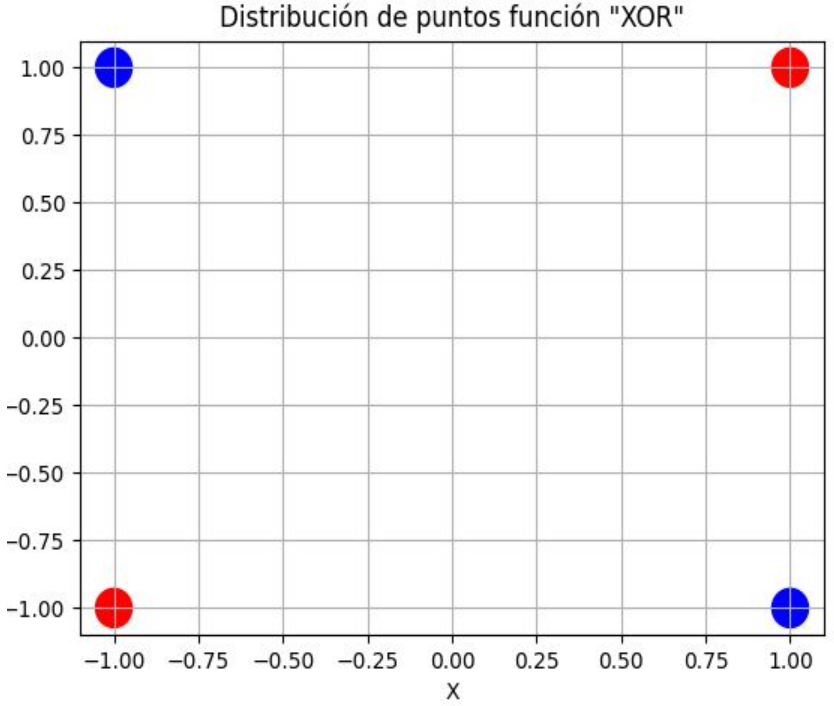
$$\Theta(x) = \begin{cases} 1 & x \geq 0 \\ -1 & x < 0 \end{cases}$$





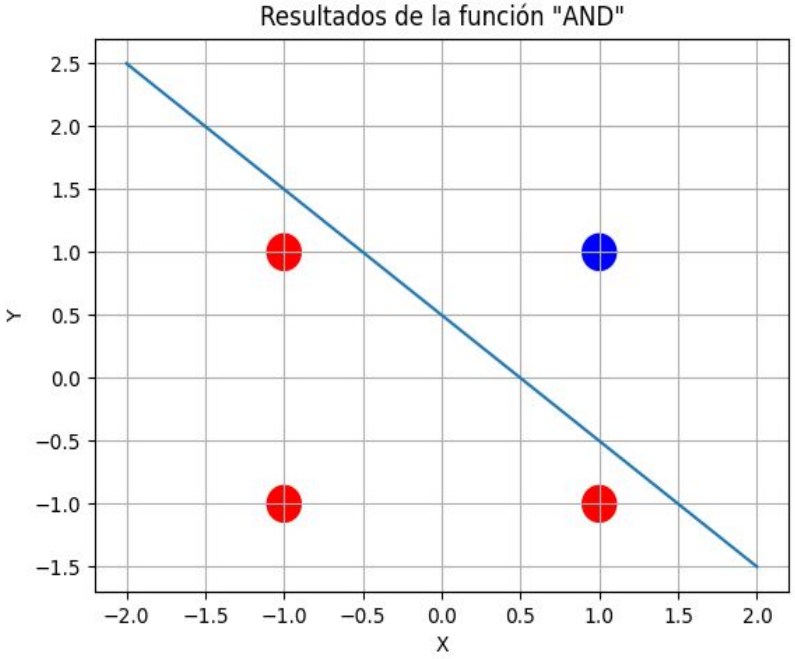
Análisis previo a aprendizaje

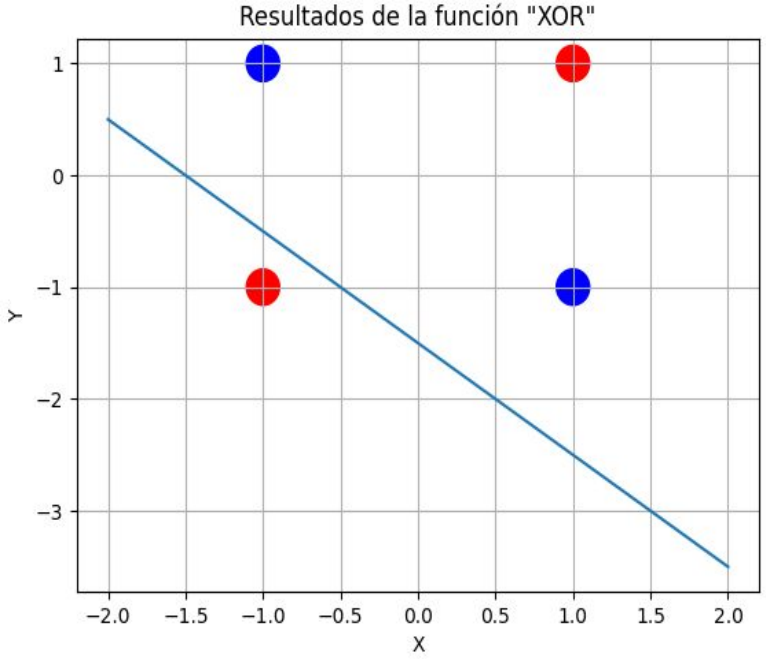




Resultados

¿Qué puede decir acerca de los problemas que puede resolver el perceptron simple escalón en relación a la resolución de los problemas que se le pidió que haga que el perceptron aprenda?

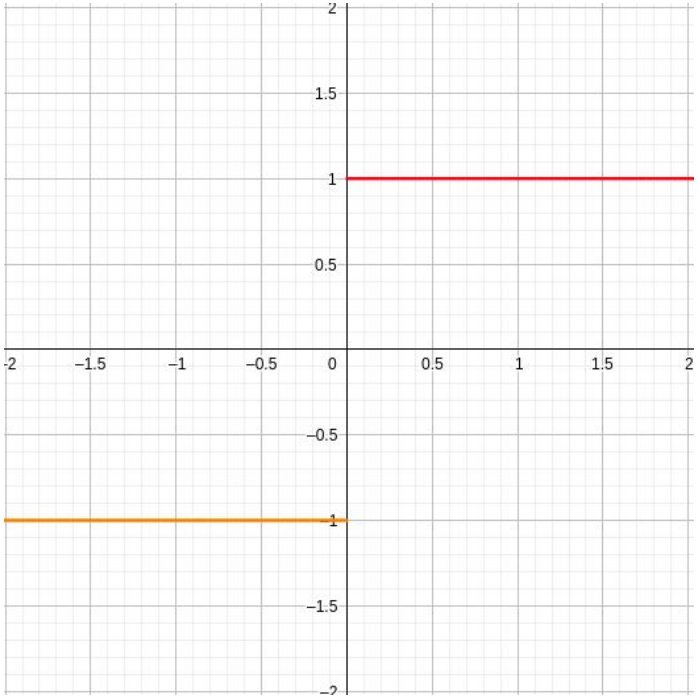




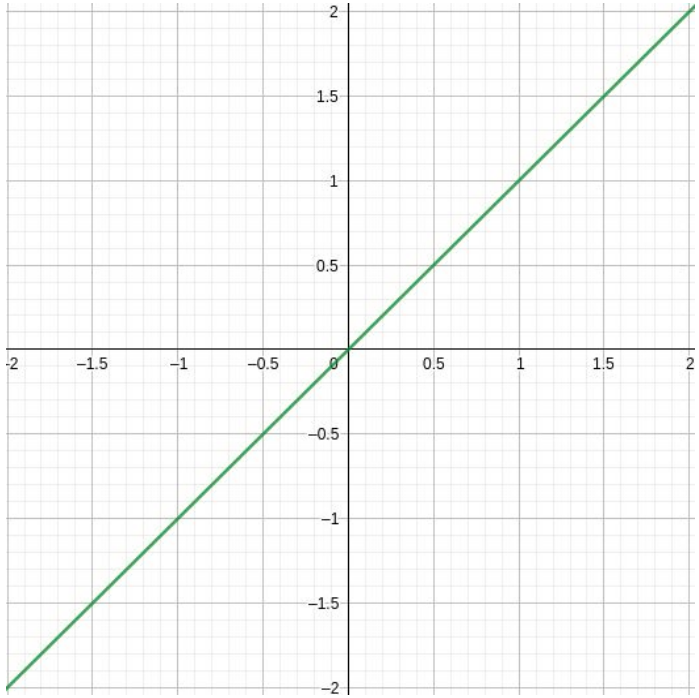
$$w_1x_1 + w_2x_2 + \theta = 0 \rightarrow x_2 = \frac{w_1x_1+\theta}{-w_2}$$

Ejercicio 2: Perceptrón simple lineal y no lineal

Comparación de funciones de activación

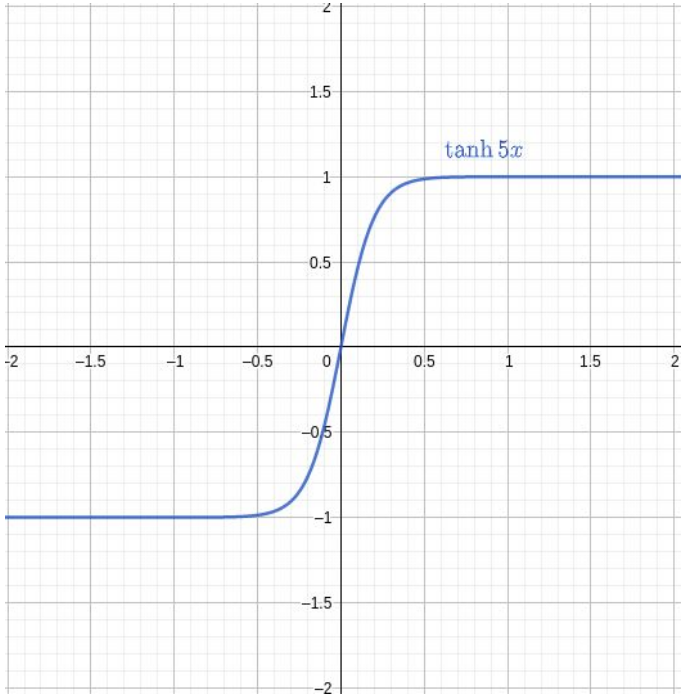


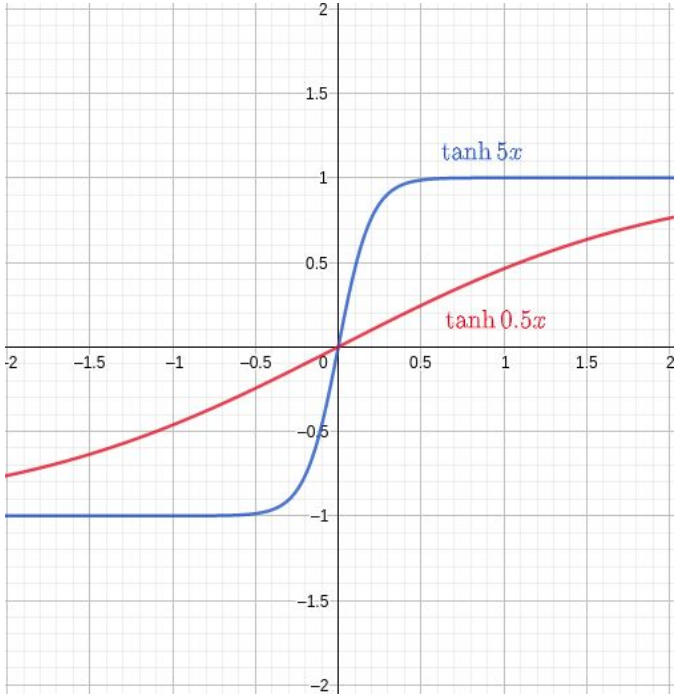
$$\Theta(x) = \begin{cases} 1 & x \geq 0 \\ -1 & x < 0 \end{cases}$$



$$\Theta(x) = x$$

Comparación de funciones de activación





$$\Theta(x) = tanh(\beta x) \quad / \quad \Theta(x) = \frac{1}{1+e^{-2\beta x}}$$

Entonces... ¿Cómo está definido el problema?

Datos de entrada

4.4793	-4.0765	4.4558
-4.1793	-4.9218	1.7664
-3.9429	-0.7689	4.8830
-3.5796	1.5557	2.6683
-3.3354	2.2292	-1.6330
1.2096	0.3121	1.6238
0.7371	-3.9118	-2.5583
-4.4792	1.3177	-2.0449
4.3120	-3.7350	1.8018
2.2866	-3.6570	0.2785
2.3784	-4.0141	-0.8841
-4.3660	-3.5797	1.0264
3.6044	-3.3175	2.5052
4.3441	-3.0375	0.8353

Salida esperada

87.3174
1.5257
39.7859
45.5674
13.3589
74.5119
3.3358
4.2974
66.5833
26.0005
14.6809
1.8713
71.0139
63.8979

Como vemos, el problema está definido en el mundo de los **reales**.

Sin embargo, las funciones de activación para el perceptrón **no lineal** trabajan en un dominio acotado:

[-1, 1] para Tanh y [0,1] para Logistic

¿Qué resultados podemos esperar?

Recordemos que la función de error, se define:

$$E(w) = \frac{1}{2} \sum_{\mu=1}^{p} (\zeta^{\mu} - O^{\mu})^2$$
$$= \frac{1}{2} \sum_{\mu=1}^{p} (\zeta^{\mu} - \sum_{i=0}^{N} w_i \xi_i^{\mu})^2$$

Es decir, hace una sumatoria del error individual de **todos los datos de entrada** para un w dado.

Para este problema, p = 200.

Resultados

Comenzamos corriendo nuestros tres perceptrones (Lineal, No Lineal-Tanh y No Lineal-Logistic) con los mismos parámetros (n = 0.01, en este caso).

- Lineal: w = [5.46630434 5.8458933 7.33922599 43.41572817],

  Error = 9276.726667578221

- No Lineal-Tanh: w = [2.55258424 2.1226252 2.80237545 11.82953935]

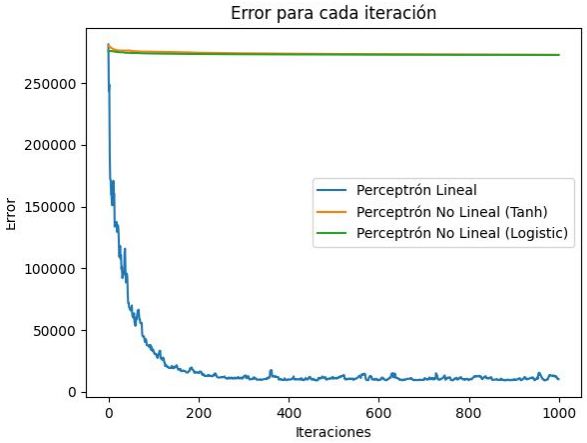
  Error = 272833.54961020884

- No Lineal-Logistic: w = [2.50580911 2.52596159 3.34135418 7.74599989]

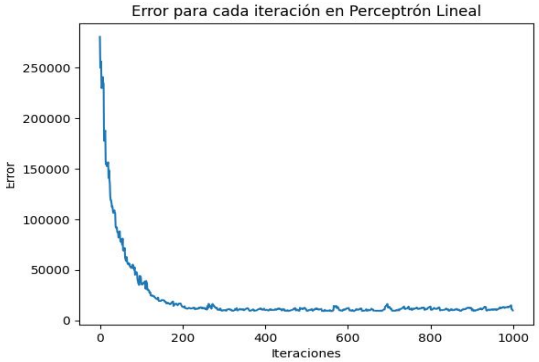
  Error = 272742.2204678929

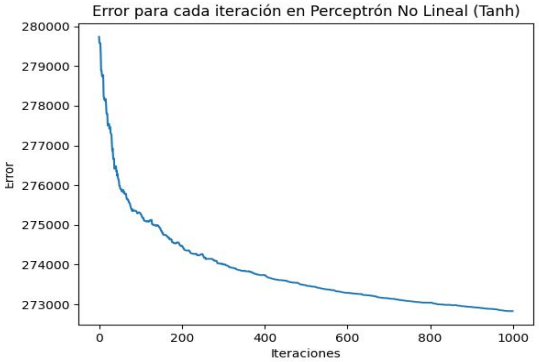
Resultados

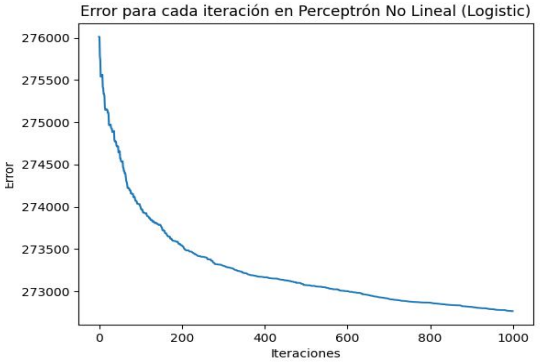
Error vs Iteraciones



n = 0.01

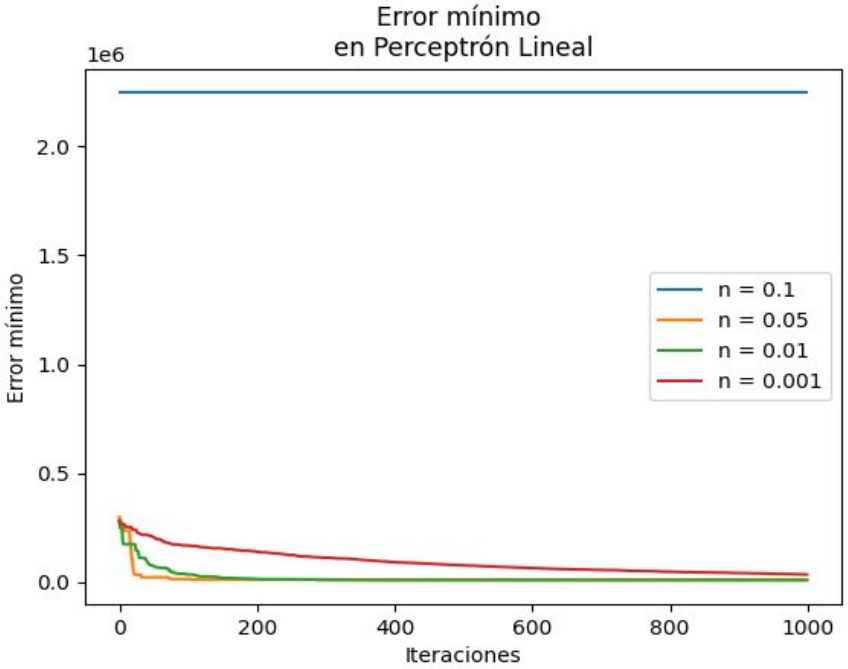


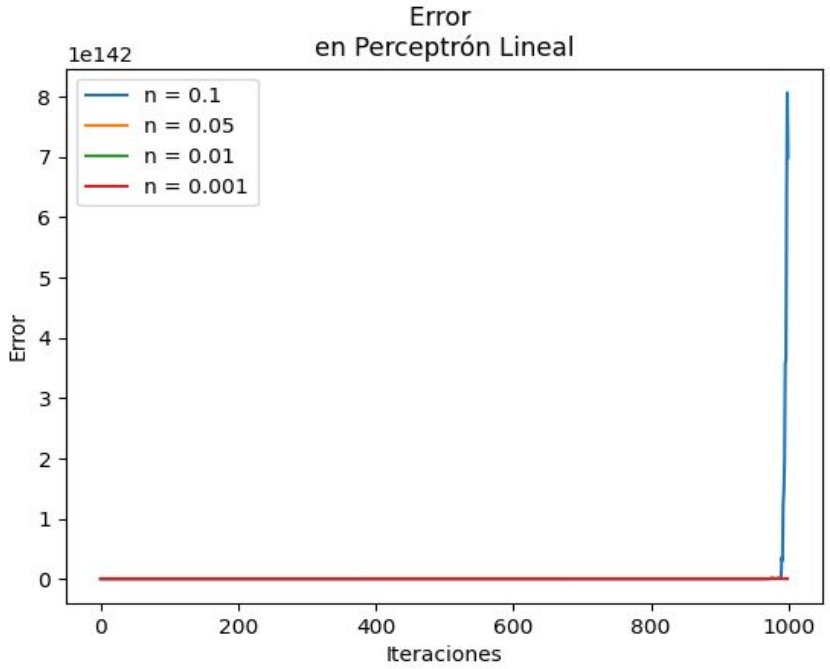




Resultados

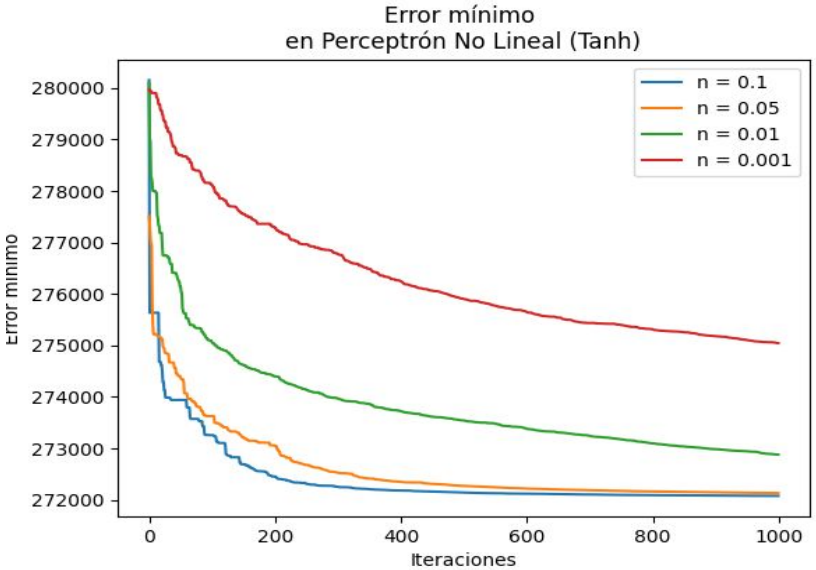
Perceptrón Lineal

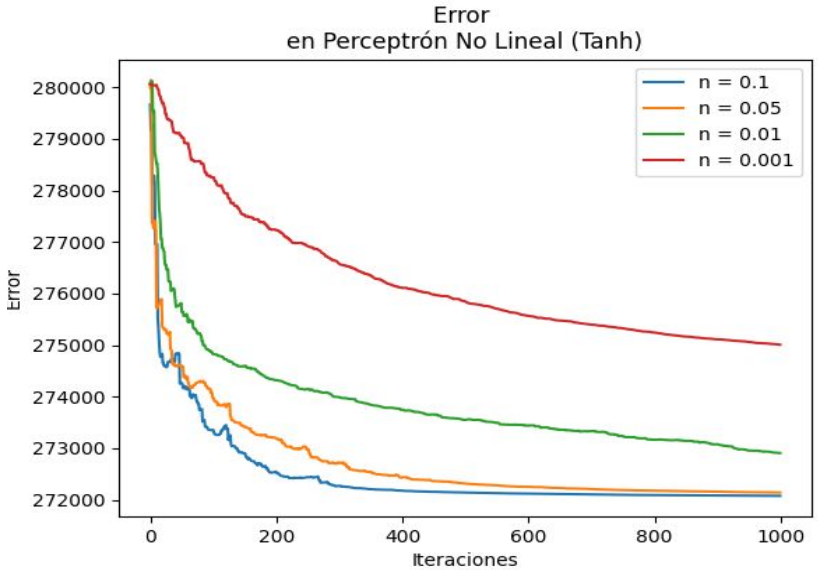




Resultados

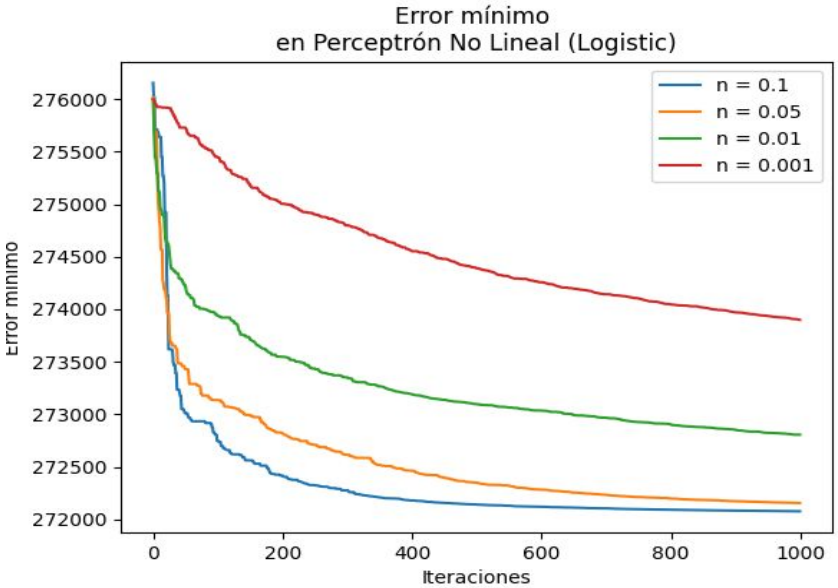
Perceptrón No Lineal (Tanh)

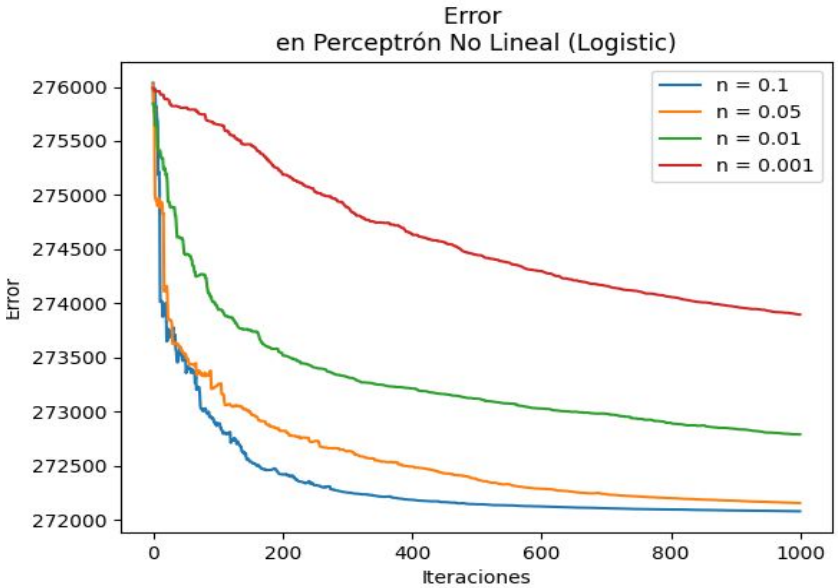




Resultados

Perceptrón No Lineal (Logistic)

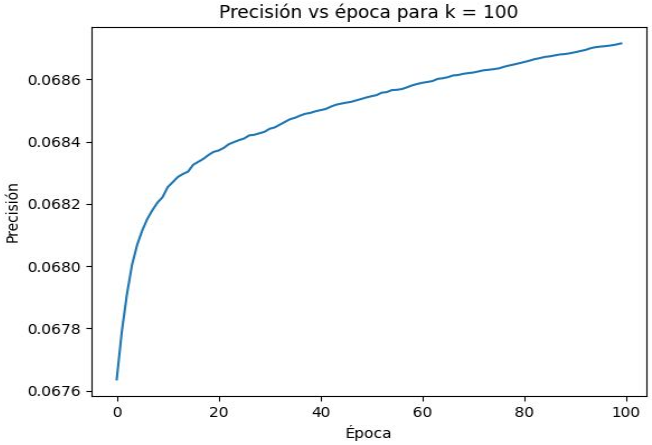


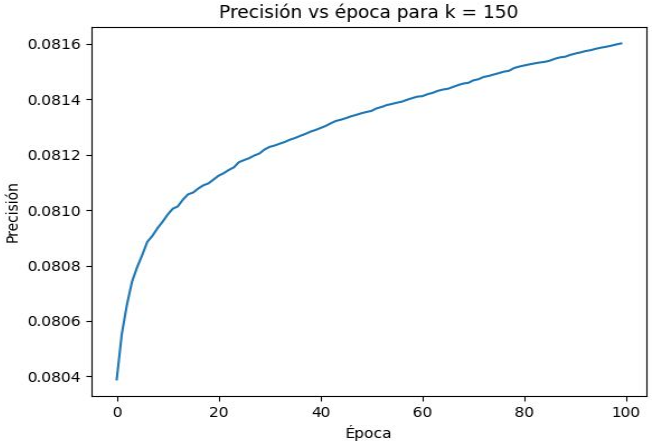


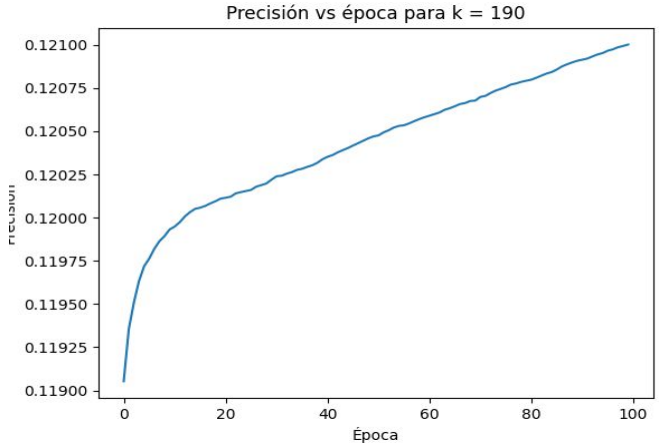
Una buena forma de medir la **capacidad de generalización** de un perceptrón es dividir el conjunto de entrada en dos nuevos conjuntos: **Conjunto de Entrenamiento** y **Conjunto de Prueba (Test)**.

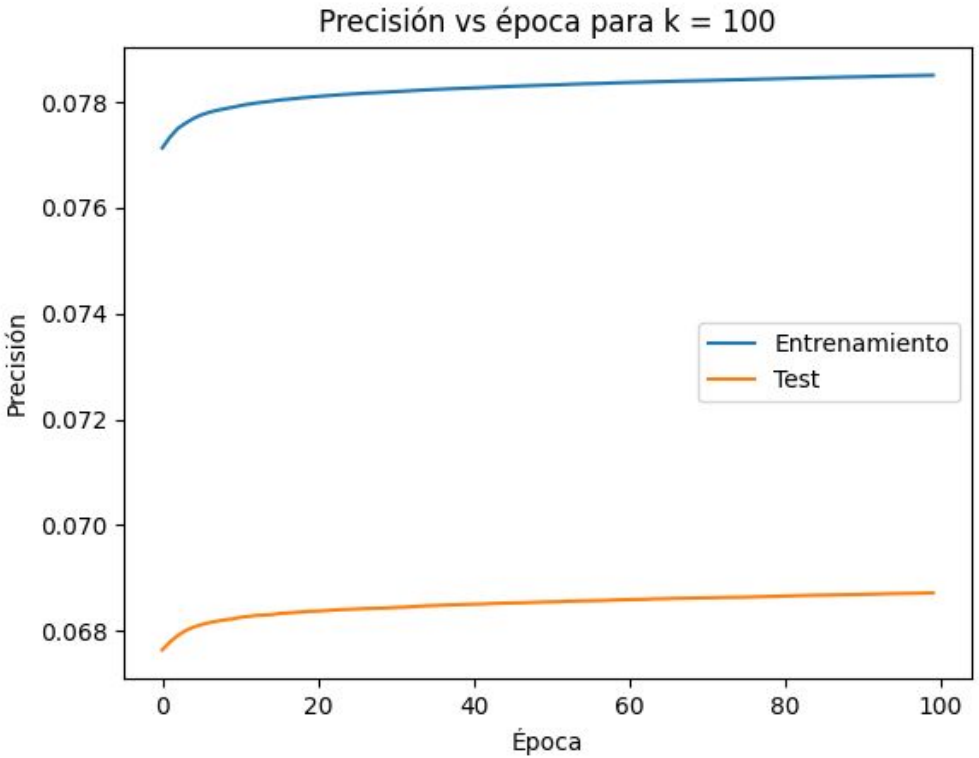
- Se “entrena” la red con el Conjunto de Entrenamiento (de *k* elementos), generando un vector *w* de pesos óptimo para ese conjunto.

- Se utiliza dicho vector *w* con los elementos del Conjunto de Prueba, y se evalúa su efectividad.


Precisión vs época para k = 100
Precisión
Época
0.0676
0.0678
0.0680
0.0682
0.0684
0.0686
0
20
40
60
80
100


Precisión vs época para k = 150
Precisión
Época
0.0804
0.0806
0.0808
0.0810
0.0812
0.0814
0.0816
0
20
40
60
80
100


Precisión vs época para k = 190
Precisión
Época
0.11900
0.11925
0.11950
0.11975
0.12000
0.12025
0.12050
0.12075
0.12100
0
20
40
60
80
100


Precisión vs época para k = 100
0.078
0.076
0.074
0.072
0.070
0.068
Precisión
0
20
40
60
80
100
Época
Entrenamiento
Test

¿Cómo elijo el mejor conjunto de entrenamiento?

Validación Cruzada (K-fold cross validation)

- Se divide el conjunto de entrada en k subconjuntos de igual tamaño

- Se toma uno de los k conjuntos para que sea el Conjunto Test, mientras que los restantes conforman el Conjunto de Entrenamiento

- Se entrena y se testea, eligiendo el Conjunto de Entrenamiento que de mejores resultados para el Conjunto Test -> **mejor capacidad de generalización**

Tomamos k = 10, obteniendo 10 subconjuntos de 20 elementos.

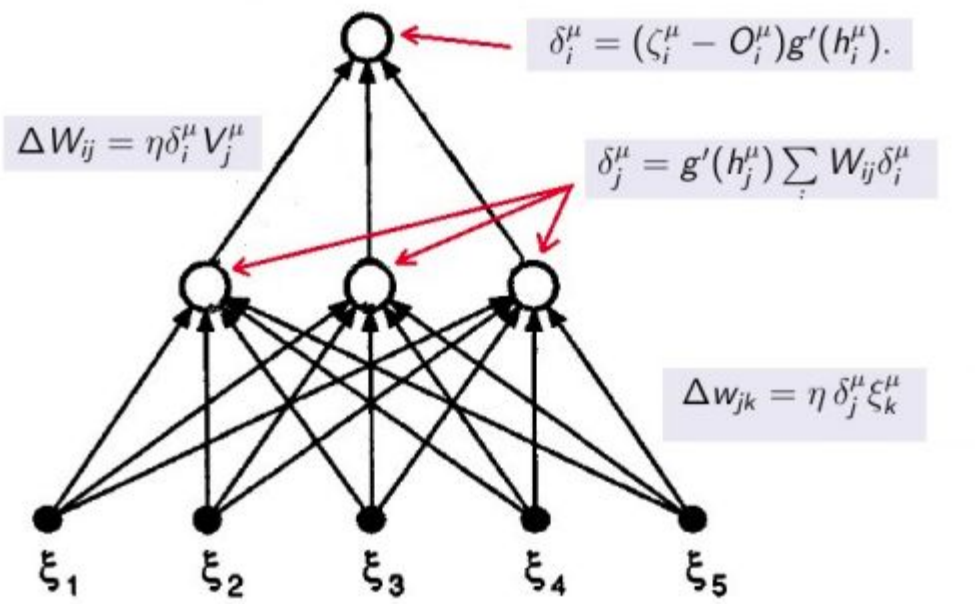
Cada uno de los K_i subconjuntos hace de Conjunto Test en una iteración.

Tomando el conjunto k2 como Conjunto Test, los 9 subconjuntos restantes conforman el mejor Conjunto de Entrenamiento.

i	accuracy
1	0,0705
2	0,0839
3	0,0794
4	0,0823
5	0,0744
6	0,0709
7	0,0552
8	0,0831
9	0,0634
10	0,0751

Ejercicio 3:
Perceptrón multicapa

Perceptrón multicapa



Retropropagación entre capa m y m-1

$$\Delta W_{ij} = \eta \sum_{\mu} \delta_i^\mu V_j^\mu$$

Con i siendo la capa m, j la capa m-1, δ el error en la capa i, y V la activación de la unidad en la capa j

Ejercicio 3.1: “Función XOR con perceptrón multicapa”

Función XOR

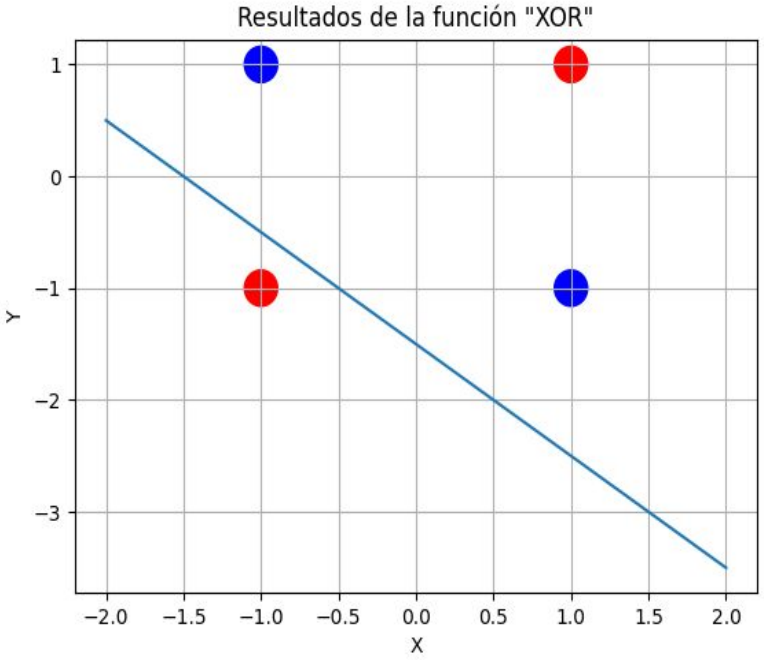
Como vimos en el ejercicio 1, la función XOR no es linealmente separable.

Input = [[-1, 1], [1, -1], [-1, -1], [1, 1]]

ExpectedOutput = [1, 1, -1, -1]

Siendo 1 = verdadero

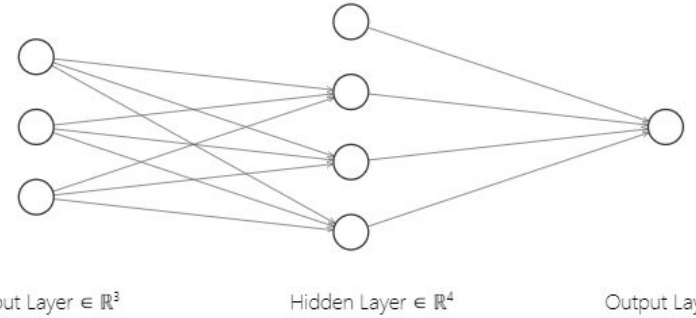
-1 = falso

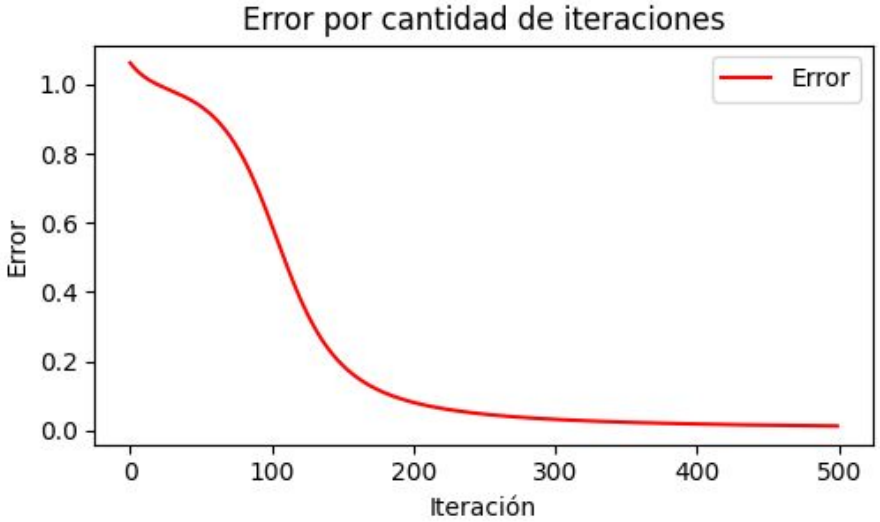


Resultados

$$\Theta(x) = tanh(\beta x)$$

η = 0.01, β = 0,5





En 500 iteraciones:

h = [0.87521761], expectedOut = 1, error = [0.01557065]

h = [0.87041819], expectedOut = 1, error = [0.01679144]

h = [-0.88584369], expectedOut = -1, error = [0.01303166]

h = [-0.90799768], expectedOut = -1, error = [0.00846443]

errorPromedio = [0.013484545]

Ejercicio 3.2: “Par o impar”

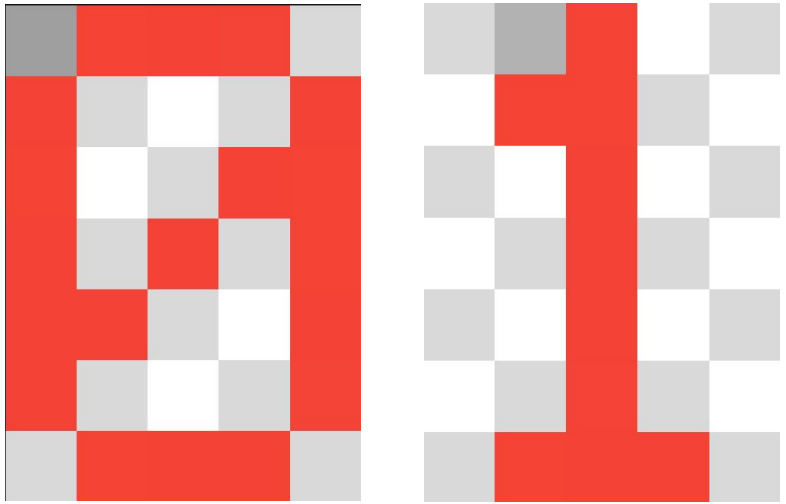
Par o impar

Input = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

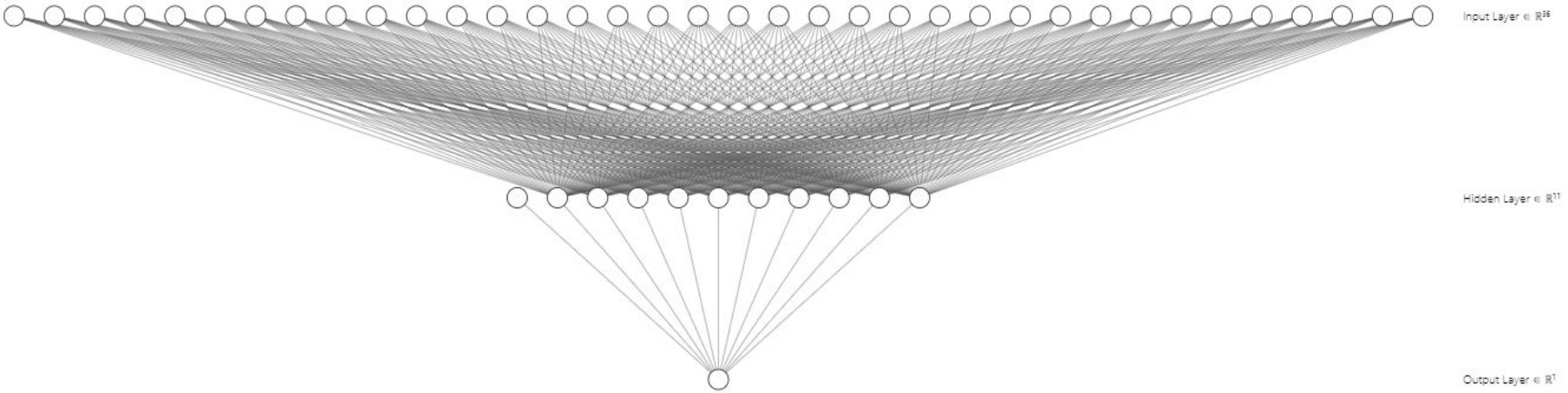
ExpectedOutput = [-1, 1, -1, 1, -1, 1, -1, 1, -1, 1]

Siendo -1 = par

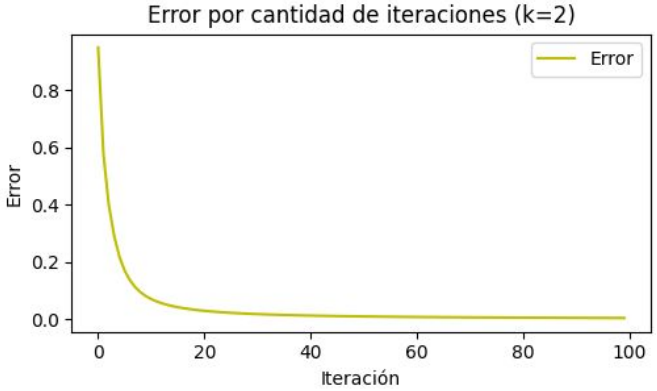
1 = impar

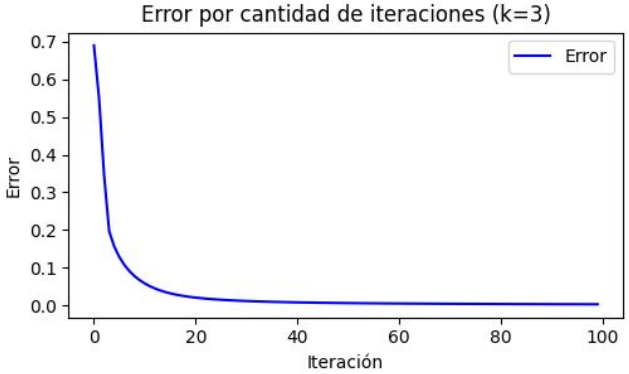


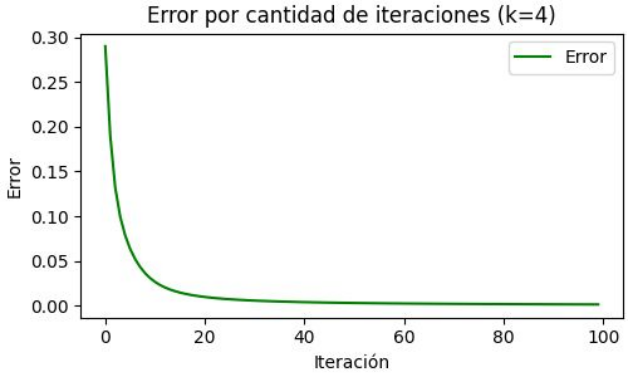
Arquitectura

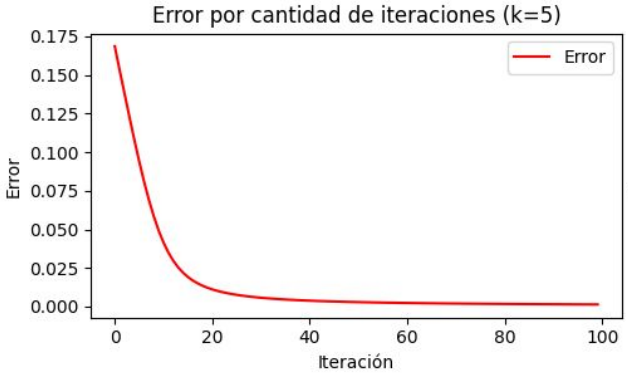


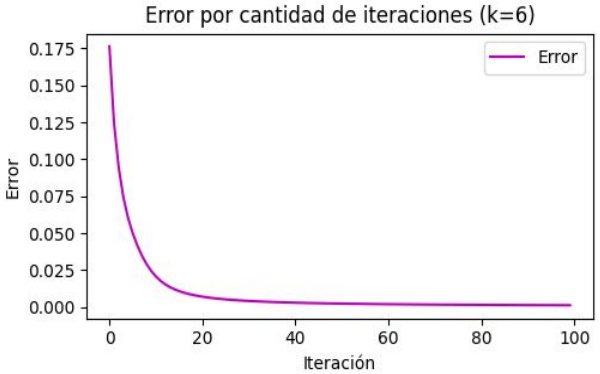
¿Qué podría decir acerca de la capacidad para generalizar de la red?



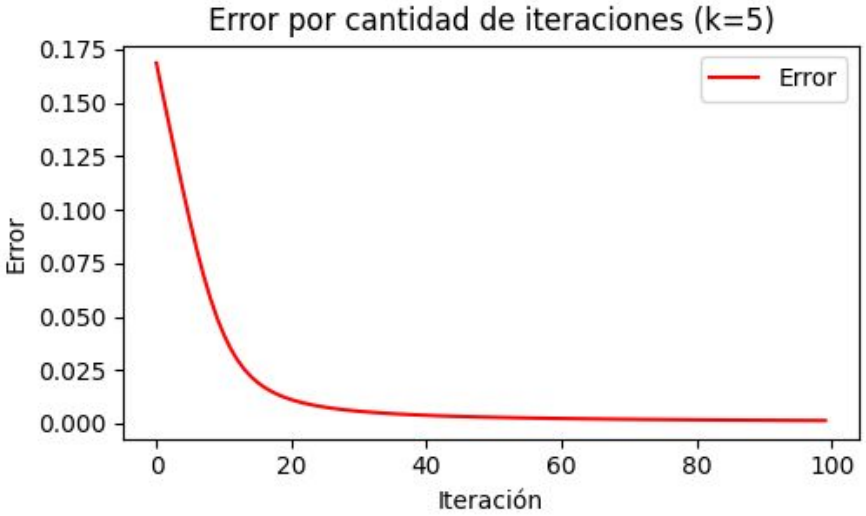








Resultados



h = [-0.9866189], expectedOut = -1, error = [0.00017905]

h = [0.98816169], expectedOut = 1, error = [0.00014015]

Error = 0.0014628076906760 29

Ejercicio 3.3:
“Números del 0 al 9”

Números del 0-9

Input = “[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]”

ExpectedOutput = [[1, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 1, 0, 0, 0, 0, 0, 0, 0, 0],

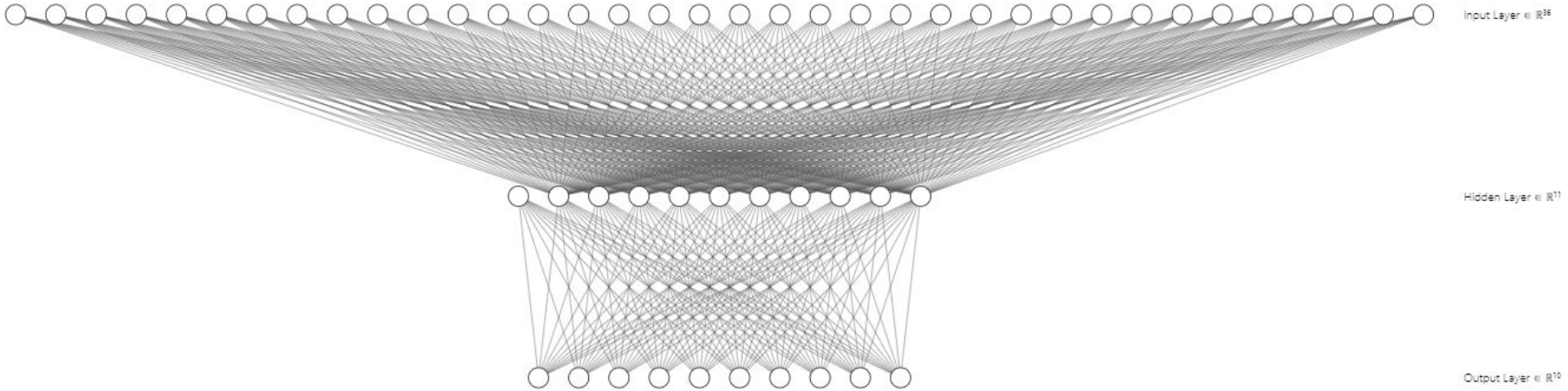
[0, 0, 1, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 1, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 1, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 1, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 1, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 1, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 1, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 1]]

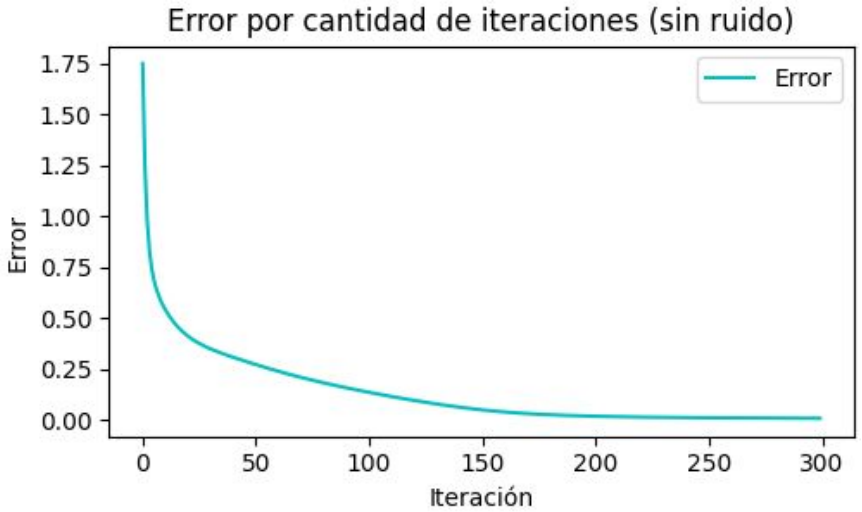
Arquitectura



Resultados

$$\Theta(x) = tanh(\beta x)$$

η = 0.01, β = 0,5



Estimado= [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

Error = 0.006199835529370979

Resultados con y sin ruido