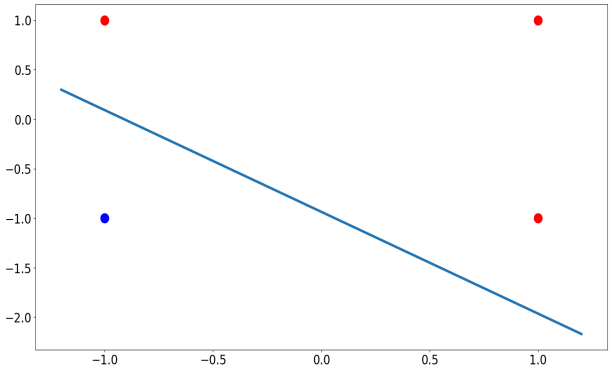
Primer Trabajo Práctico

Sistemas Adaptativos: redes neuronales

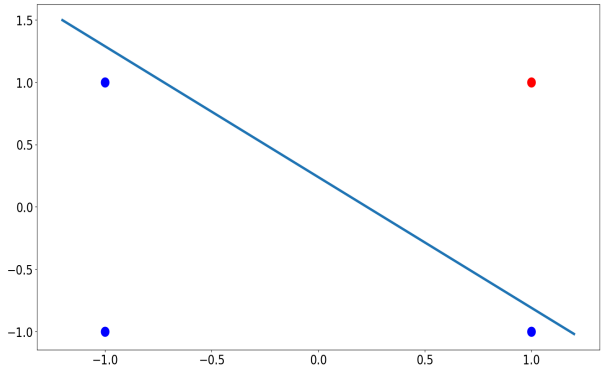
David Alejandro Jorge Tasé

Ejercicio 1

Implemente un perceptrón simple que aprenda la función lógica AND de 2 y de 4 entradas. Lo mismo para la función lógica OR. Para el caso de 2 dimensiones, grafique la recta discriminadora y todos los vectores de entrada de la red.



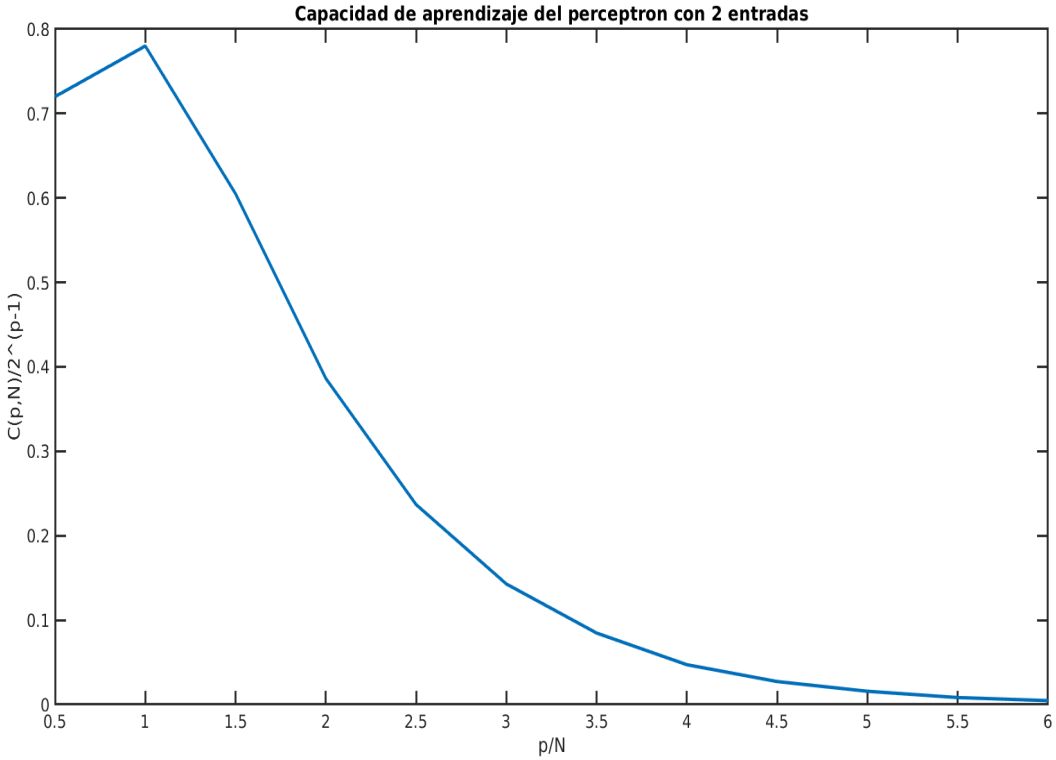
OR de dos entradas

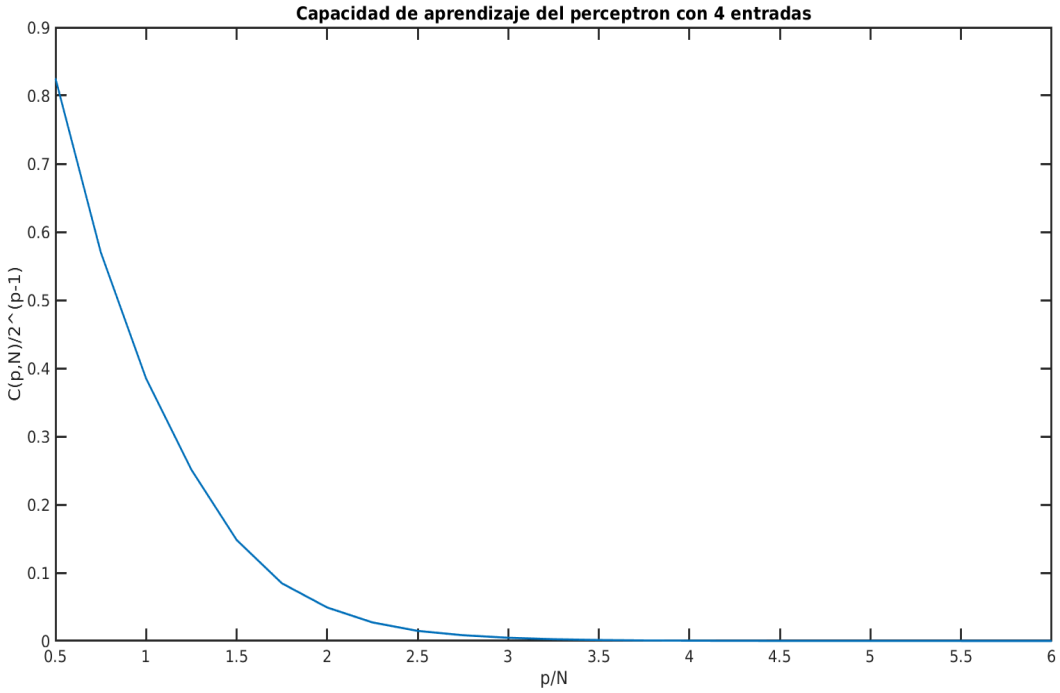


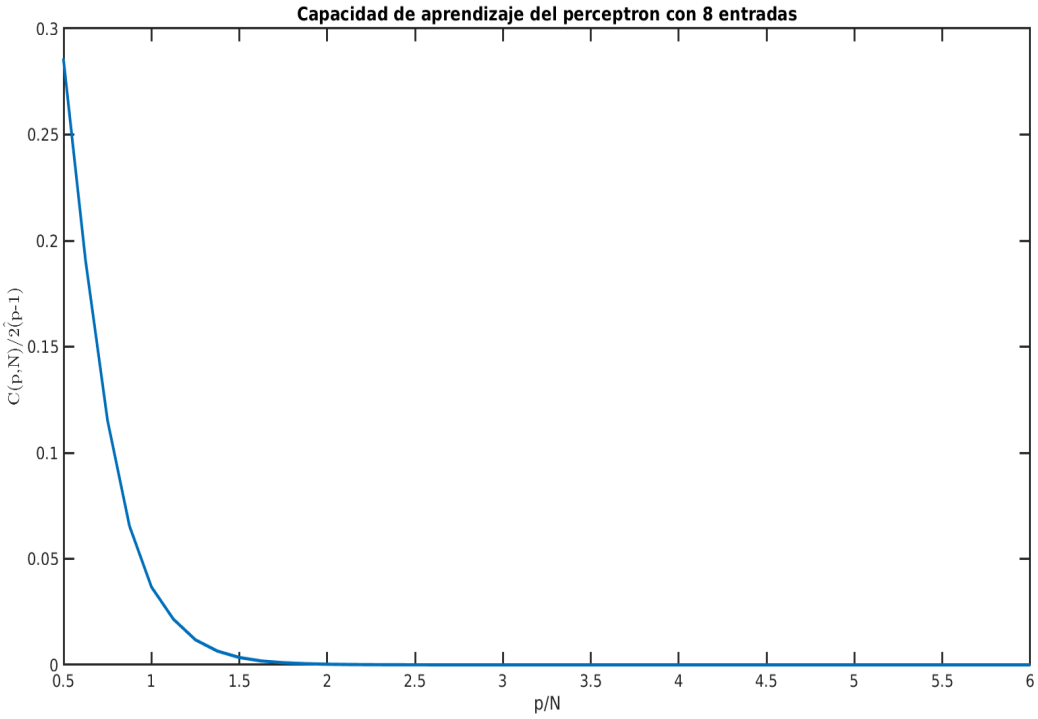
AND de dos entradas

Ejercicio 2

Determine numéricamente cómo varía la capacidad del perceptrón simple en función del número de patrones enseñados.


Capacidad de aprendizaje del perceptron con 2 entradas
C(p,N)/2^(p-1)
p/N
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
5.5
6


Capacidad de aprendizaje del perceptron con 4 entradas
C(p,N)/2^(p-1)
p/N
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
5.5
6



En los gráficos mostrados se define a p como la cantidad de patrones y N la cantidad de de entradas. El gráfico se construyó utilizando un perceptrón simple para clasificar si en una serie de números dados (patrón) todos son positivos (la salida debe ser 1) o negativos (la salida debe ser -1), la cantidad de patrones se aumentó progresivamente hasta 100 y para cada cantidad específica el proceso se repitió 100 veces calculando la capacidad de aprendizaje en cada una como el número de aciertos entre el número de patrones introducidos y luego promediando dichos resultados.

Se puede ver que a mayor cantidad de entradas más pronunciado se hace el decaimiento en la capacidad de aprendizaje.

Ejercicio 3

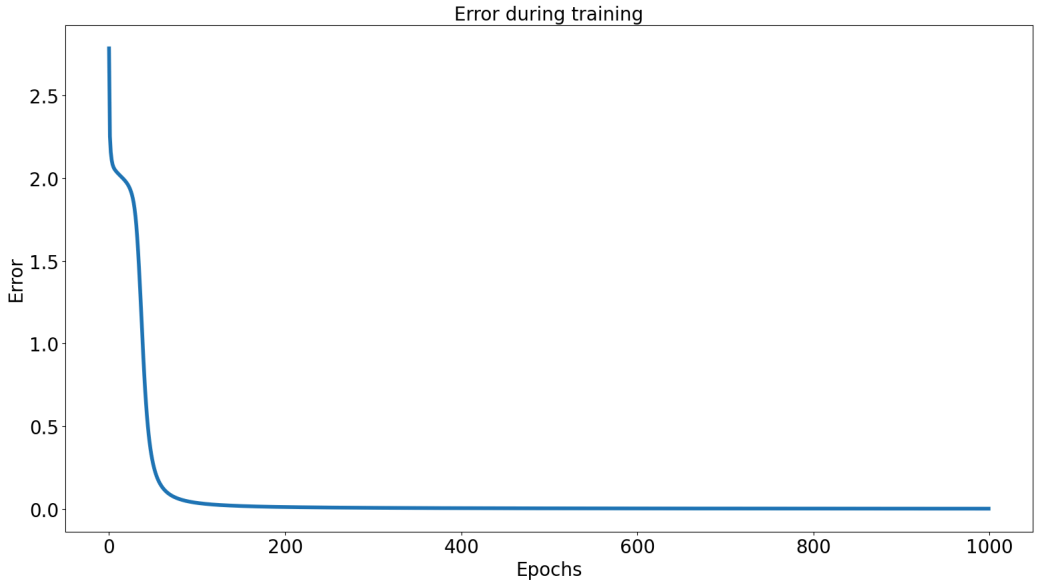
a) Implemente un perceptrón multicapa que aprenda la función lógica XOR de 2 y de 4 entradas (utilizando el algoritmo Backpropagation).

Configuración utilizada

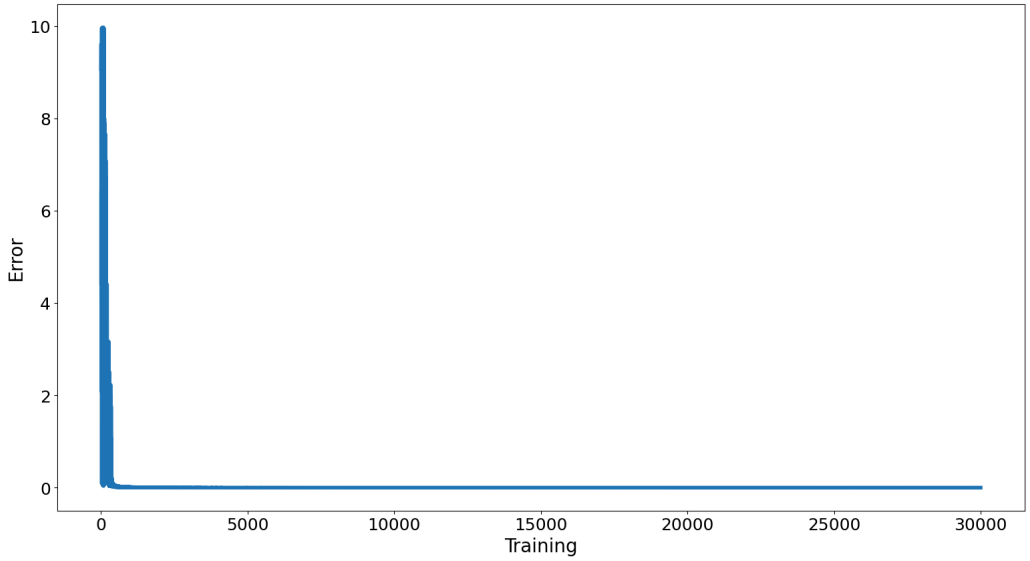
XOR de 2 entradas: 2 capas, 4 neuronas en la primera y 1 a la salida.

XOR de 4 entradas: 2 capas, 12 neuronas en la primera y 1 a la salida.

b) Muestre cómo evoluciona el error durante el entrenamiento.



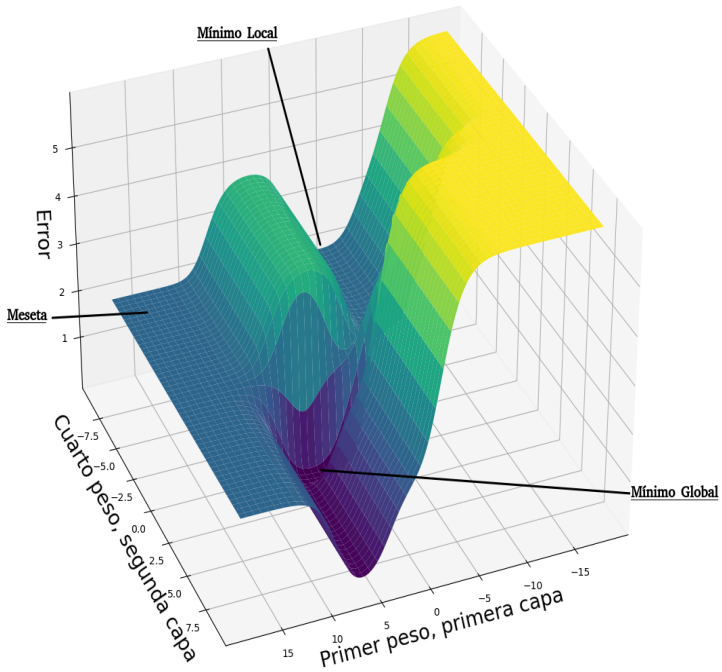
XOR de 2 entradas



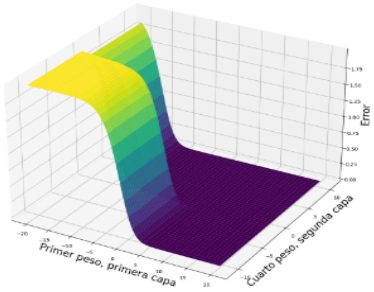
XOR de 4 entradas. El entrenamiento se realiza con minibatch de 8 patrones (de 16 posibles) permutados de forma aleatoria.

c) Para una red entrenada en la función XOR de dos entradas, muestre cómo varía el error en función del cambio en dos pesos de la red. De ejemplos de mínimos locales y mesetas.

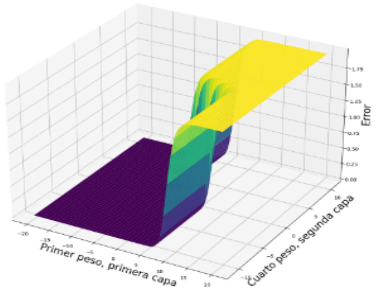
Se varió el primer peso de la primera capa (w_{11}) en el intervalo de $[-10w_{11}, 10w_{11})$, y el cuarto peso de la segunda capa (w_{24}) en el intervalo de $[-5w_{24}, 5w_{24})$



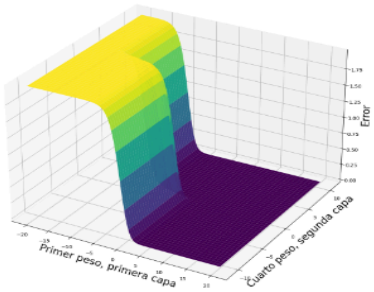
d) Idem (c) pero mostrando el error para cada patrón de entrada por separado.



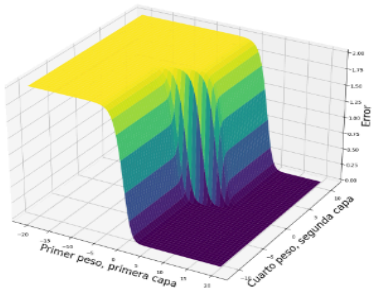
Patrón [0 0]



Patrón [0 1]



Patrón [1 0]



Patrón [1 1]

Ejercicio 4

a) Implemente una red con aprendizaje Backpropagation que aprenda la siguiente función:

$$f_{(x,y,z)} = sen(x) + cos(y) + z$$

donde: x e y $\in$ [0, 2 π] y z $\in$[− 1, 1]. Para ello construya un conjunto de datos de entrenamiento y un conjunto de evaluación. Muestre el error en función de las épocas de entrenamiento.

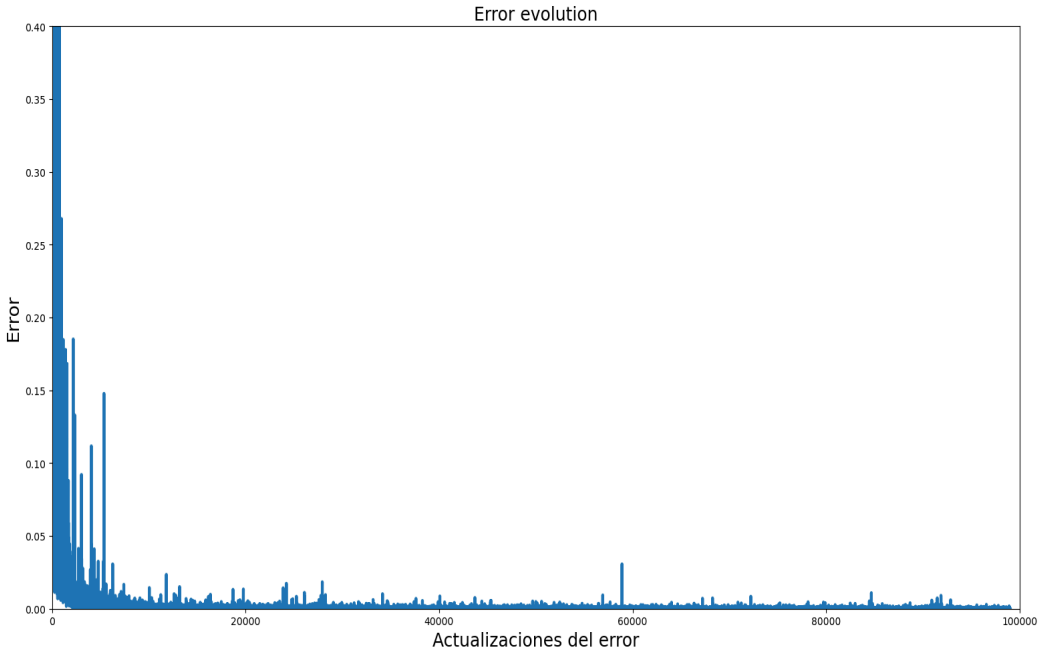
Configuración inicial utilizada: 3 capas, 8 en la primera, 6 en la segunda, y una en la tercera. Factor de aprendizaje 0.1, y minibatch de 2 patrones con permutaciones aleatorias en cada época.

Para entrenar el perceptrón se generaron 1000 muestras y para el test de comprobación 200.

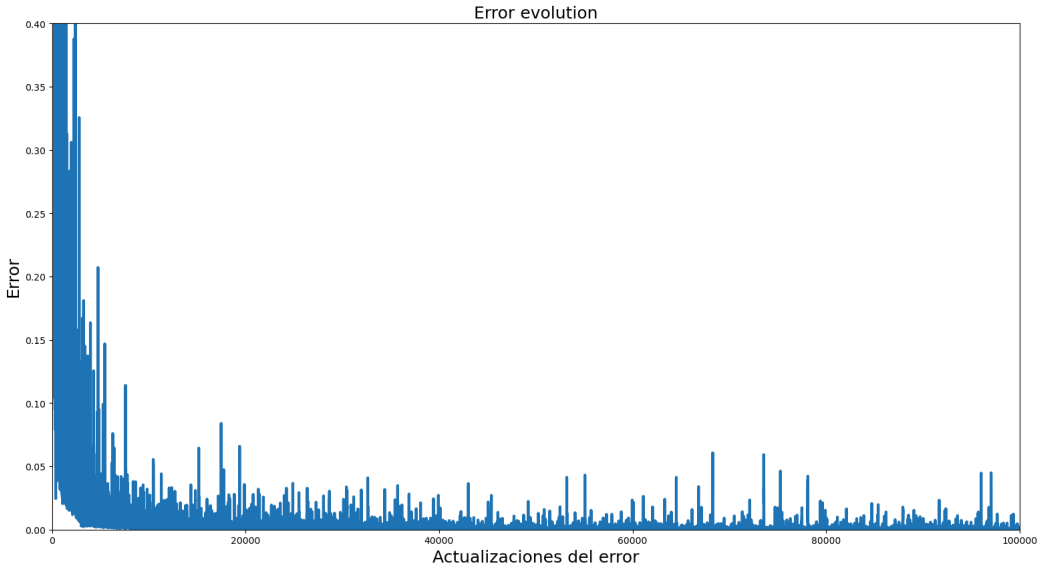
b) Muestre cómo evoluciona el error durante el entrenamiento.

c) Analice, mediante simulaciones, el efecto que tiene el tamaño de minibatch y la constante de aprendizaje en el número de iteraciones y el tiempo total de entrenamiento necesario para obtener un buen desempeño de la red.

Variando el tamaño del minibatch:



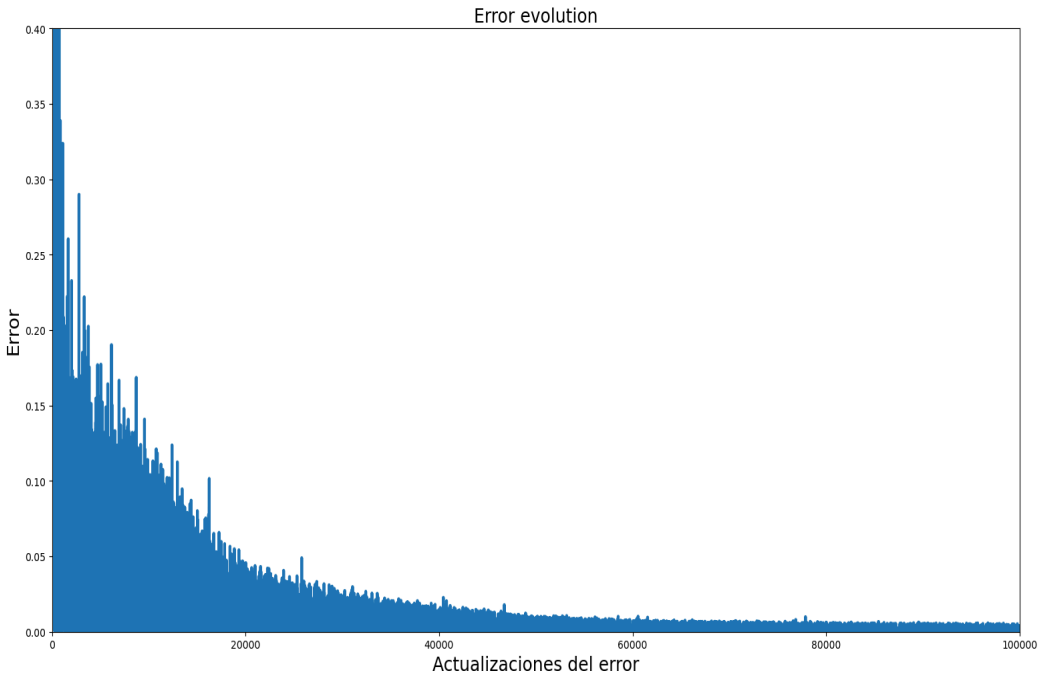
Minibatch de 10



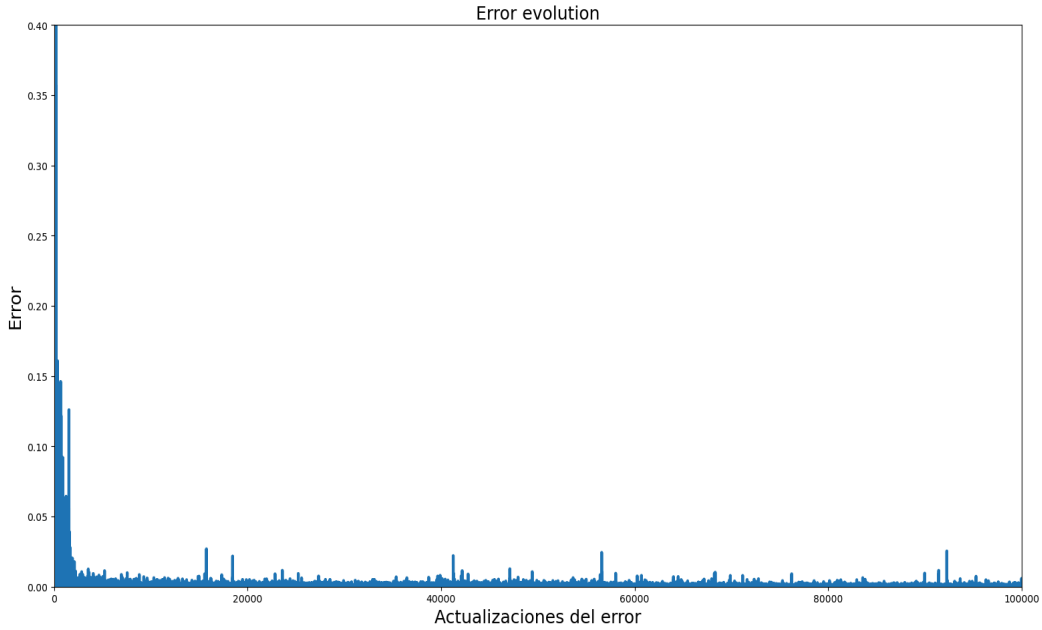
Minibatch de 20

Se puede ver cómo a medida que aumenta el tamaño del minibatch también se requiere aumentar la cantidad de veces que se actualiza el error para lograr que este baje hasta niveles razonables.

Variando el factor de aprendizaje (***η***):



Factor de aprendizaje 0.01



Factor de aprendizaje 0.8

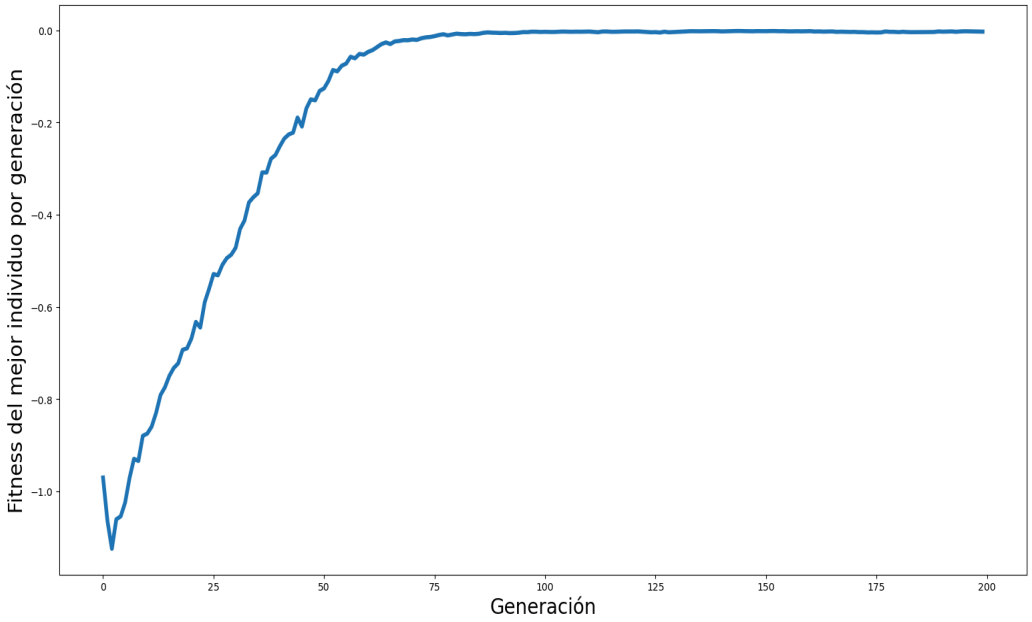
Para valores de pequeños de η = 0.01 el aprendizaje se hace considerablemente más lento y para η = 0.8 es significativamente más rápido sin embargo el aumento brusco en esta provoca oscilaciones en el proceso.

Ejercicio 5

a) Encontrar un perceptrón multicapa que resuelva una XOR de 2 entradas con un algoritmo genético. Graficar el fitness a lo largo del proceso de evolución.

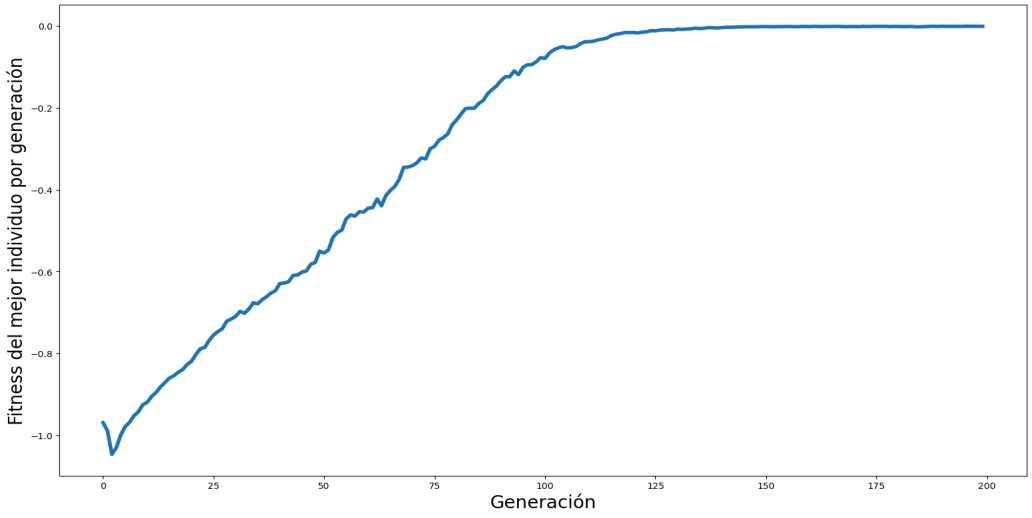
Configuración inicial utilizada: perceptrón de 2 capas, 4 neuronas en la primera y 1 en la segunda. Población de 100 perceptrones, constante de mutación 0.1 y probabilidad de cross-over 0.083. Se define el fitness como el inverso del error (

$$- E = - \frac{1}{2} \sum_{i=1}^{n} (Y_{deseado} - Y_{obtenido})^2)$$

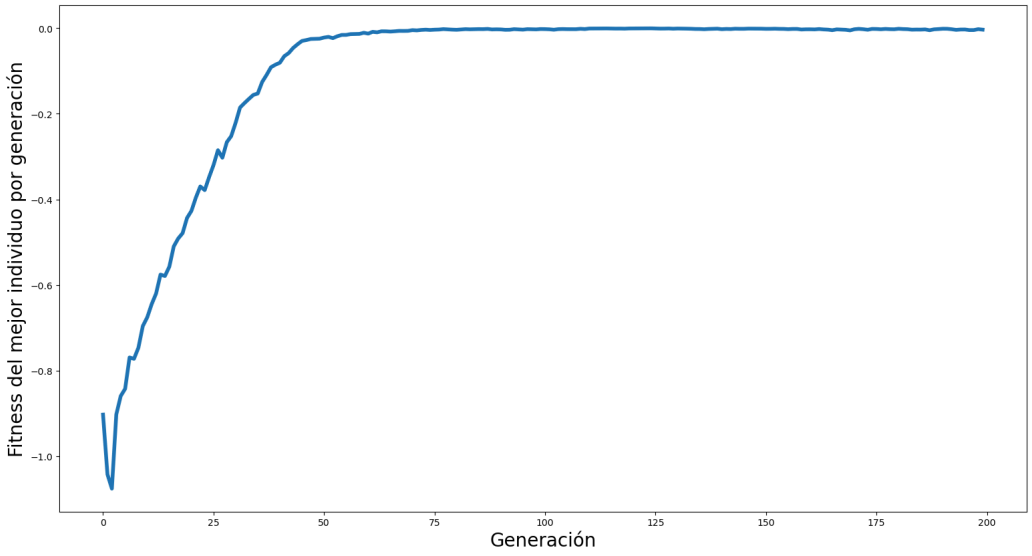


Fitness del mejor individuo en cada generación

Variando la constante de mutación:



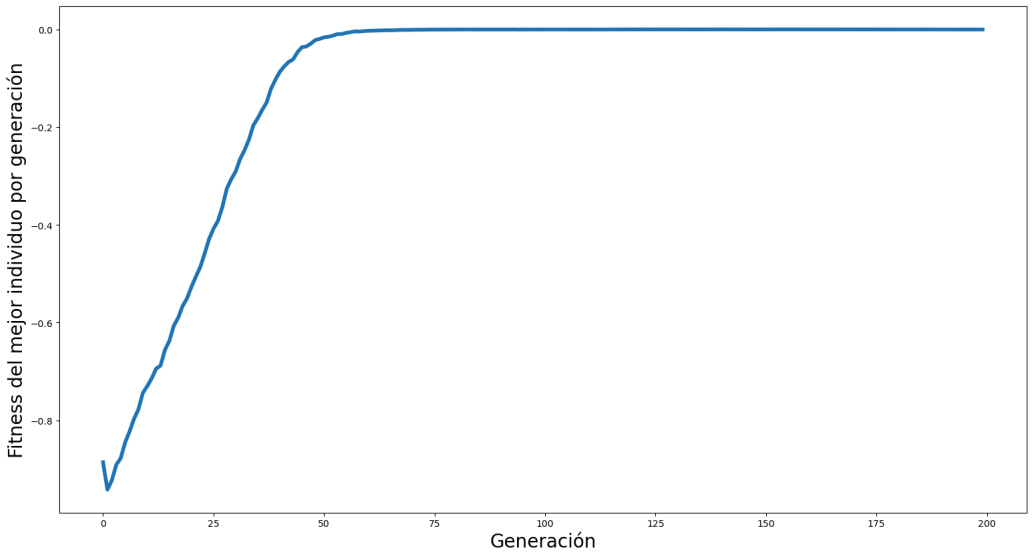
Constante de mutación 0.001



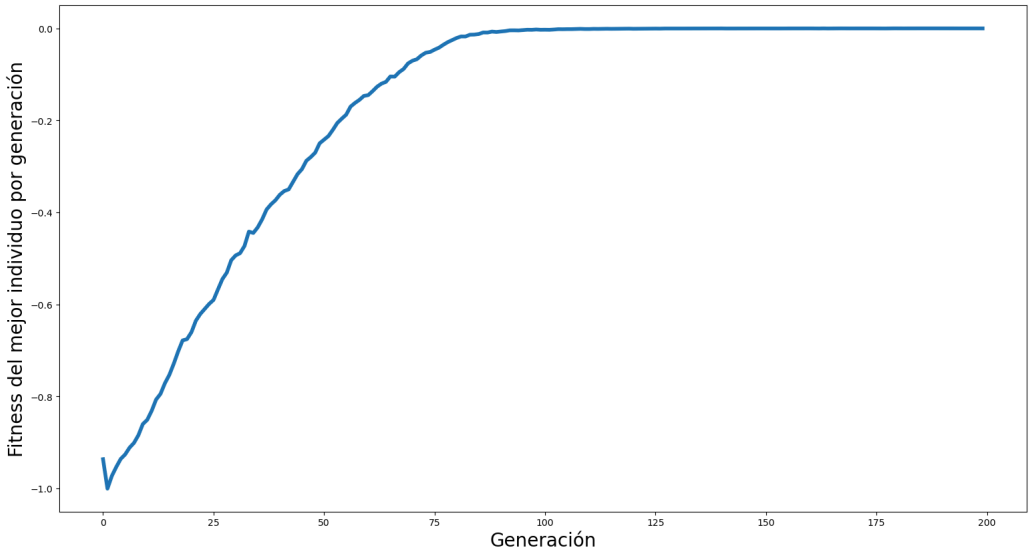
Constante de mutación 0.9

No se observa un gran cambio pero al parecer una baja constante de mutación desacelera el proceso evolución.

Variando la probabilidad de cross-over:



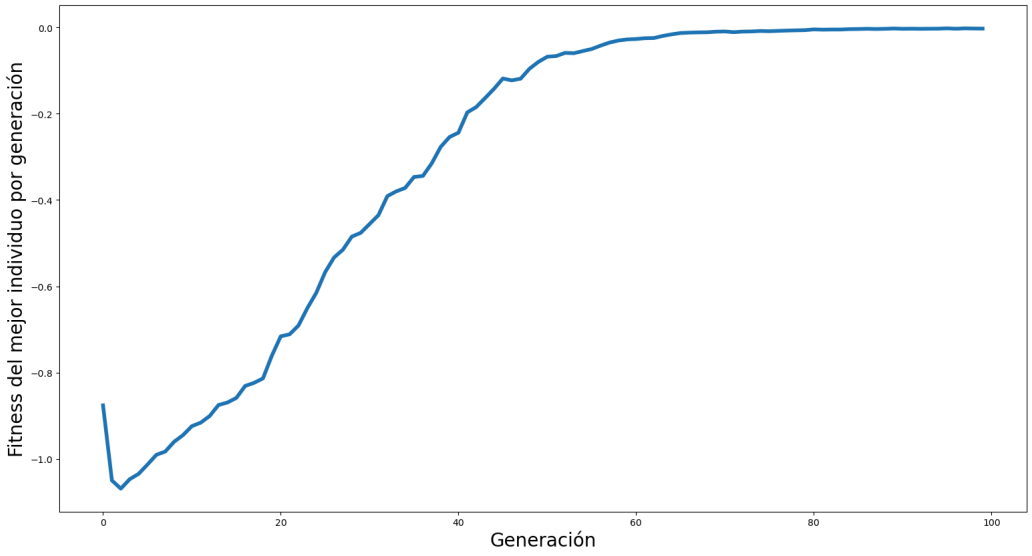
Probabilidad de cross-over 0.083



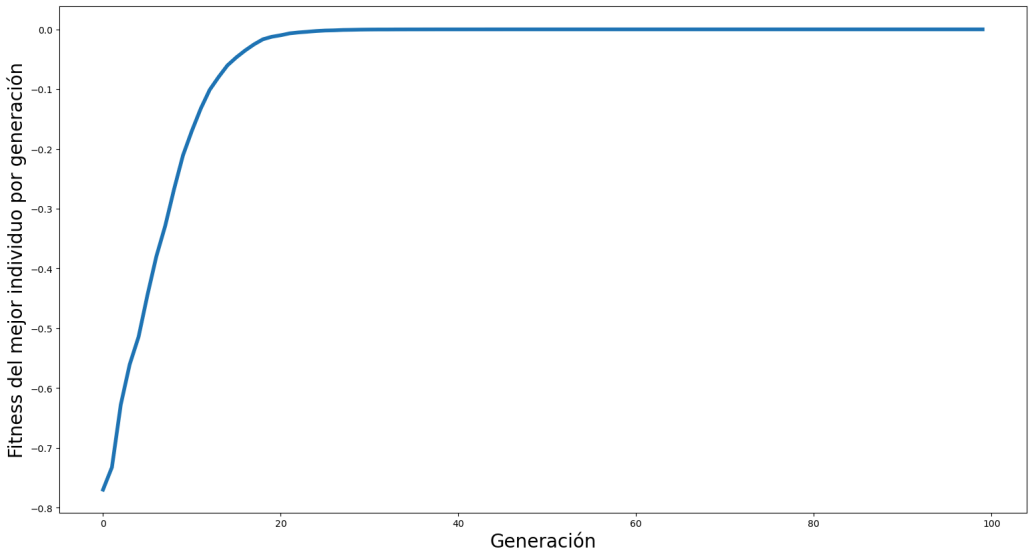
Probabilidad de cross-over 0.5

Acá aparece el caso contrario, una baja probabilidad de cross-over favorece el proceso de evolución ya que se llega a un fitness relativamente alto en menor cantidad de generaciones.

Variando el tamaño de la población:



Población de 50 perceptrones



Población de 5000 perceptrones

Acá se hace evidente que a mayor cantidad de individuos la evolución se hará más rápida y estable.