Perceptrón Simple

Sistemas de Inteligencia Artificial

Primer Cuatrimestre 2023

Rodrigo Ramele
Eugenia Piñeiro
Alan Pierri
Santiago Reyes
Marina Fuster
Luciano Bianchi

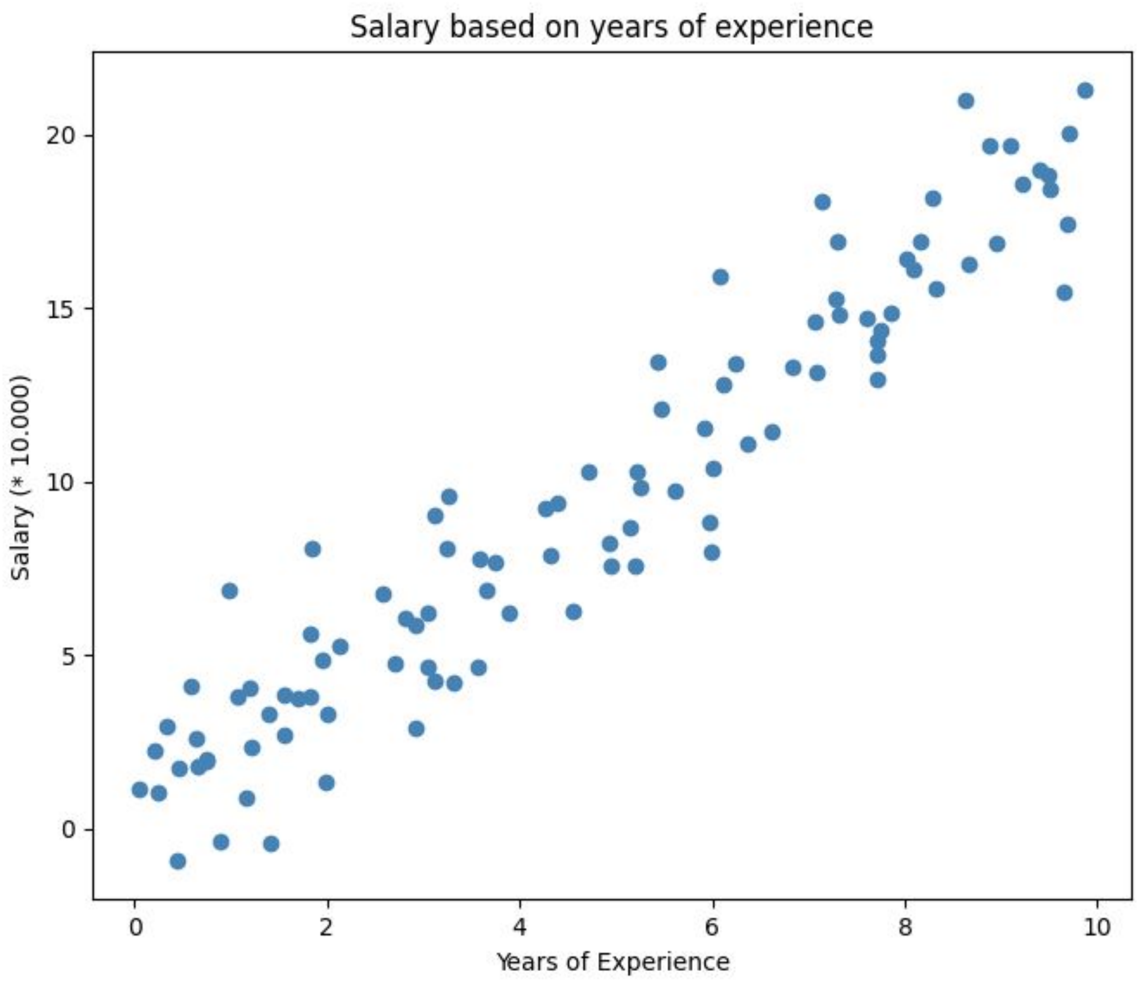
RESUMEN DE LA CLASE ANTERIOR

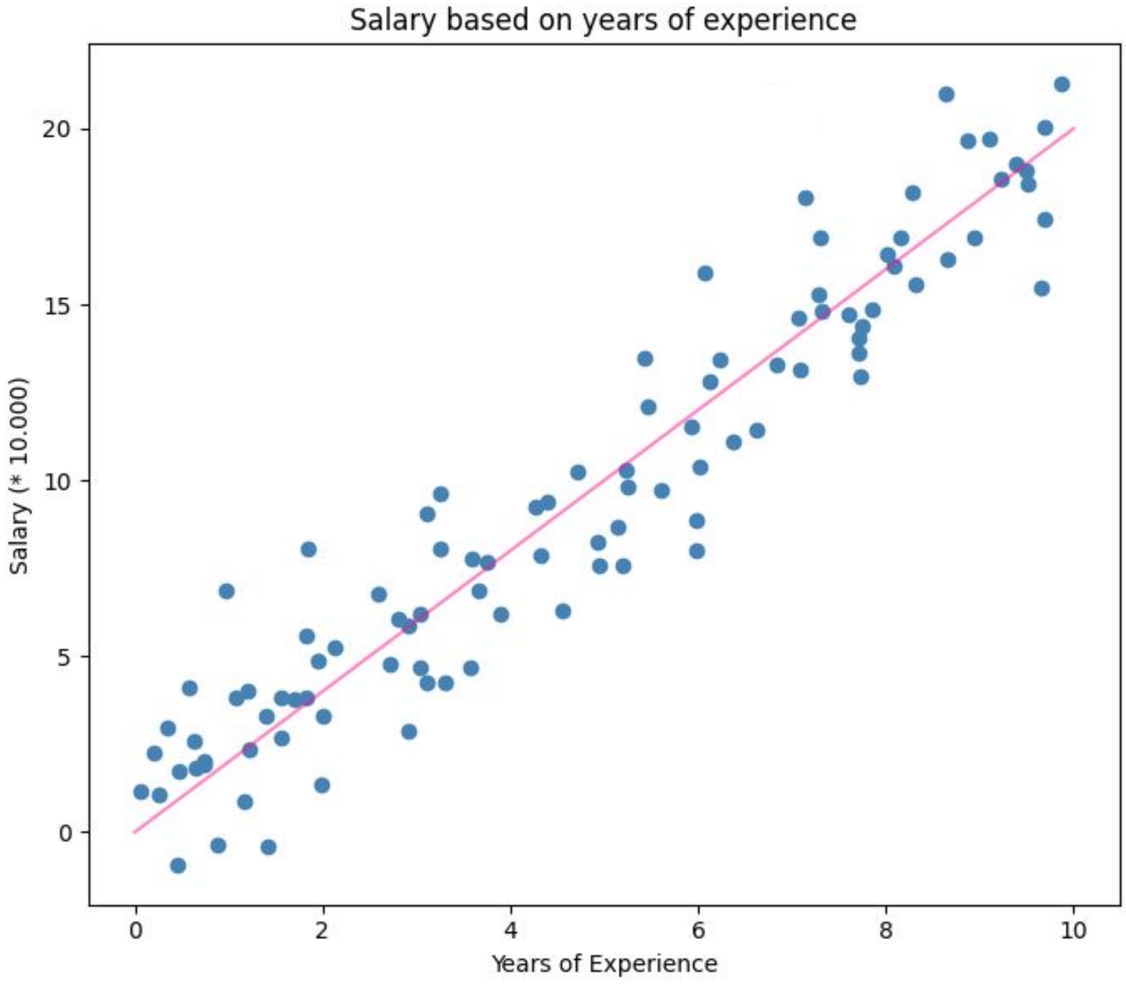
- McCulloch y Pitts sientan las bases del modelo de neurona que se utiliza en el área de redes neuronales. Este modelo se denomina **Perceptrón**.

- El modelo de McCulloch y Pitts permite resolver problemas linealmente separables.

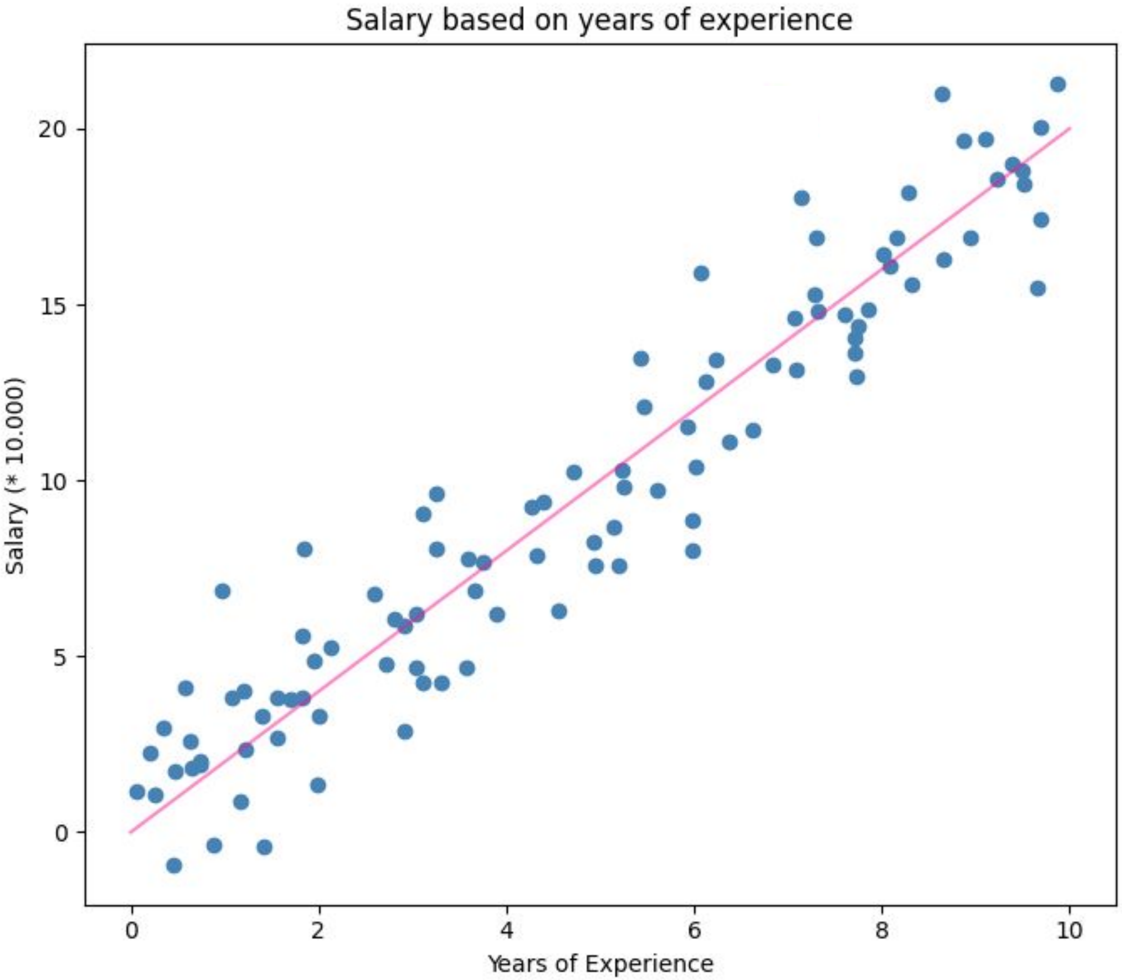
- Rosenblatt provee el mecanismo que permite obtener los pesos del perceptrón de manera iterativa

- No es lo mismo aprendizaje que generalización


Salary based on years of experience
Salary (* 10.000)
Years of Experience
0
5
10
15
20
0
2
4
6
8
10

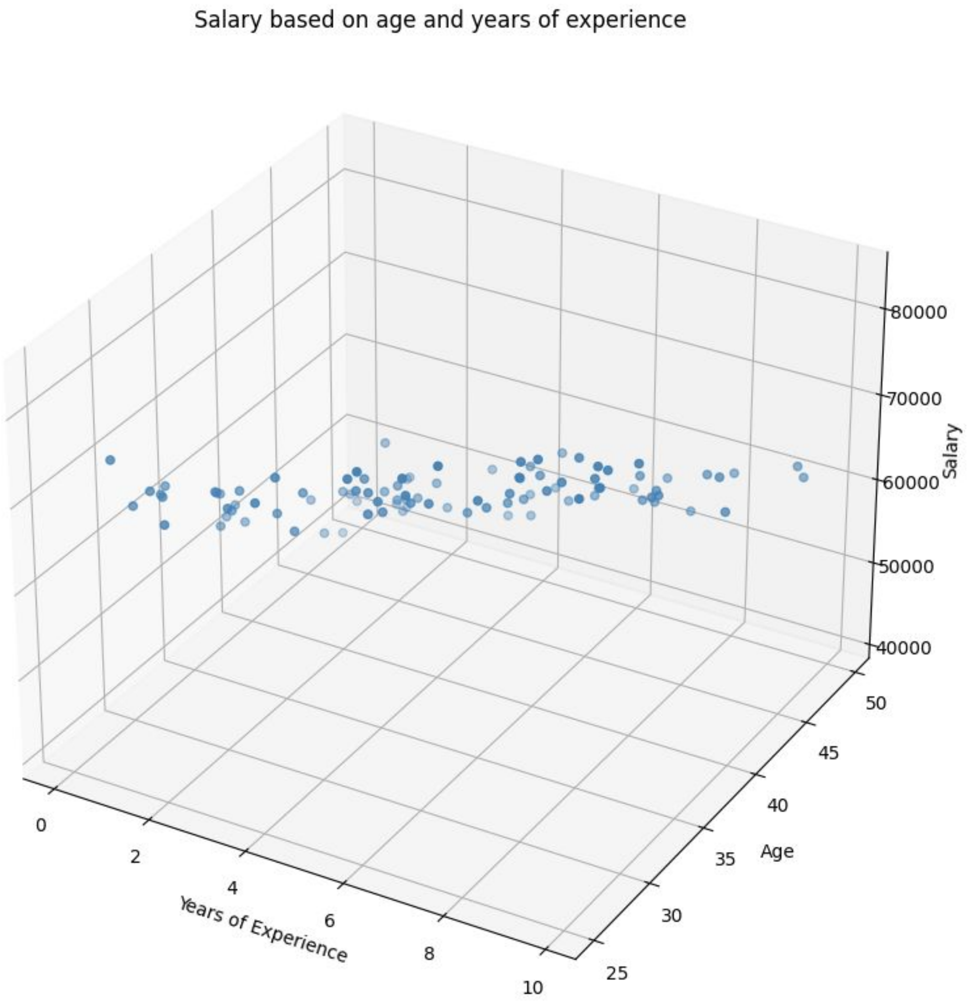



```
salary(years_of_experience):
    return a * years_of_experience + b
```

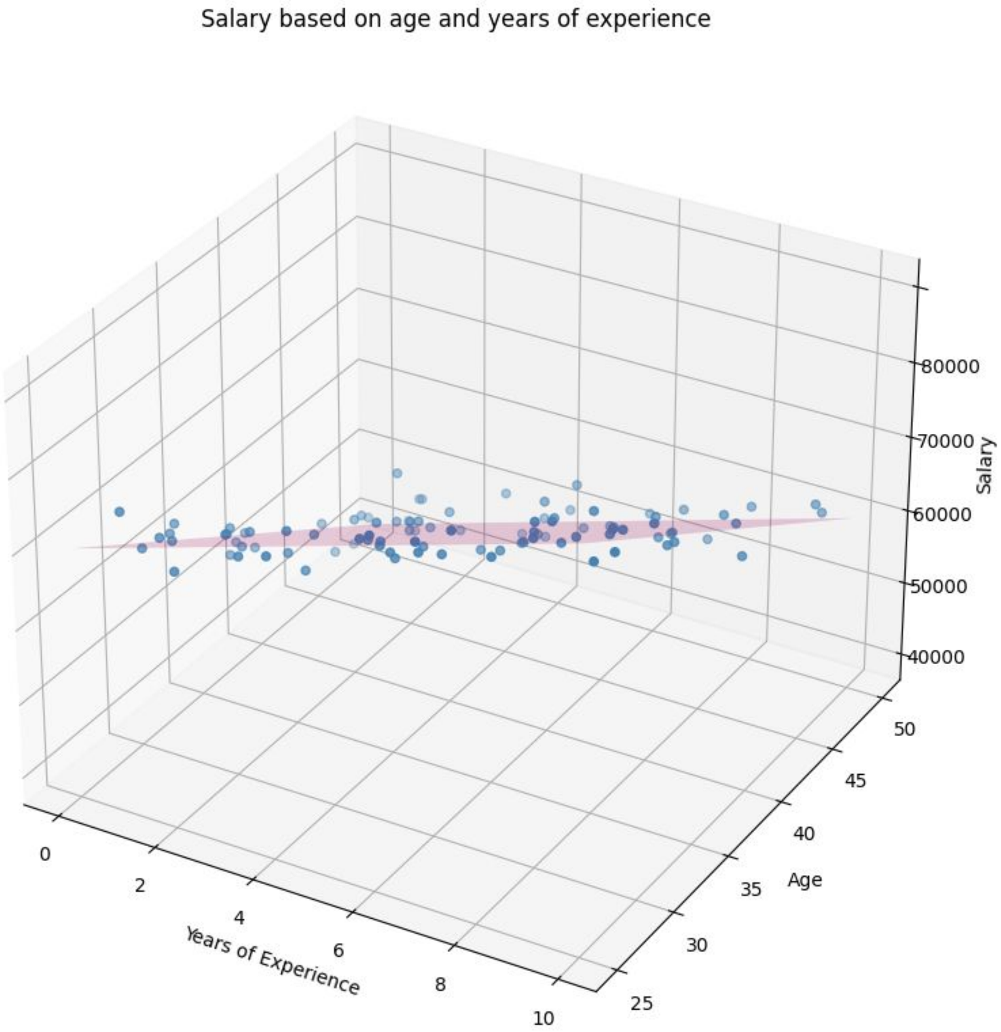


salary(years_of_experience):
    return a * years_of_experience + b

salary(x1):
    return $w_1 * x_1 - w_0$

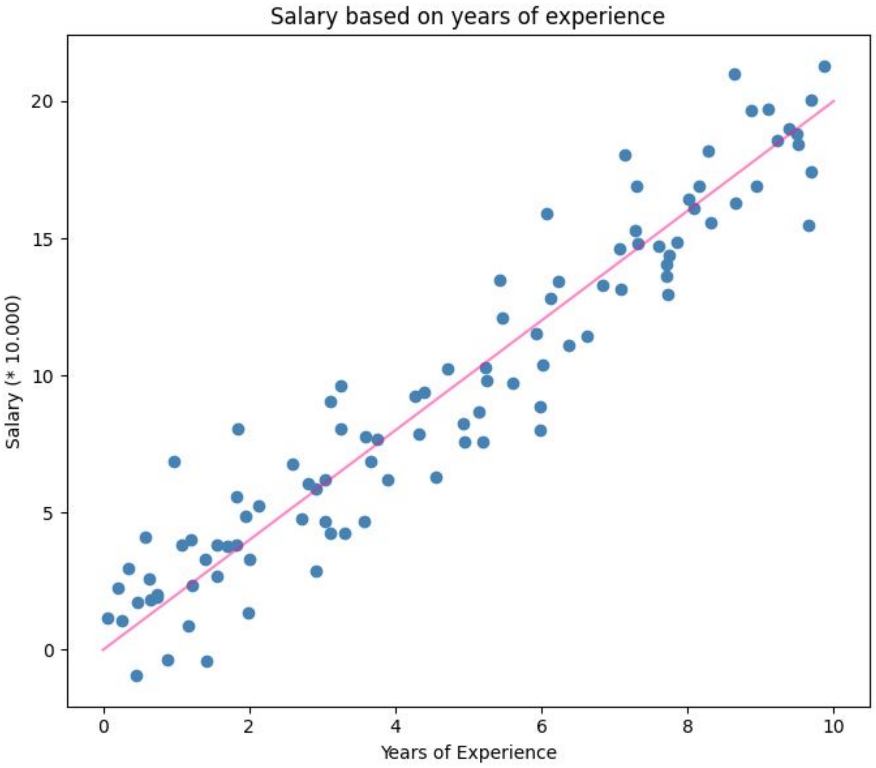


$$\text{salary}(x_1, x_2): \quad \text{return } w_1 * x_1 + w_2 * x_2 - w_0$$



salary(x_1, x_2):
    return $w_1 * x_1 + w_2 * x_2 - w_0$

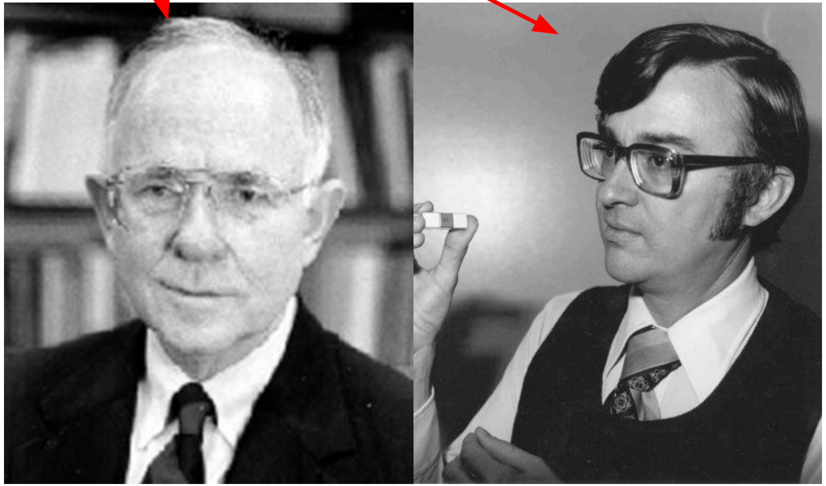
¿CÓMO RESUELVO EL PROBLEMA?



Necesito encontrar los valores de ***w*** que me permitan encontrar un hiperplano que ajuste lo mejor posible al conjunto de datos

$$salario = \sum_{i=1}^{n} x_i . w_i - w_0$$

Widrow-Hoff - 1960



ADALINE: ADAptive LINear Element (o perceptrón simple ***lineal***)

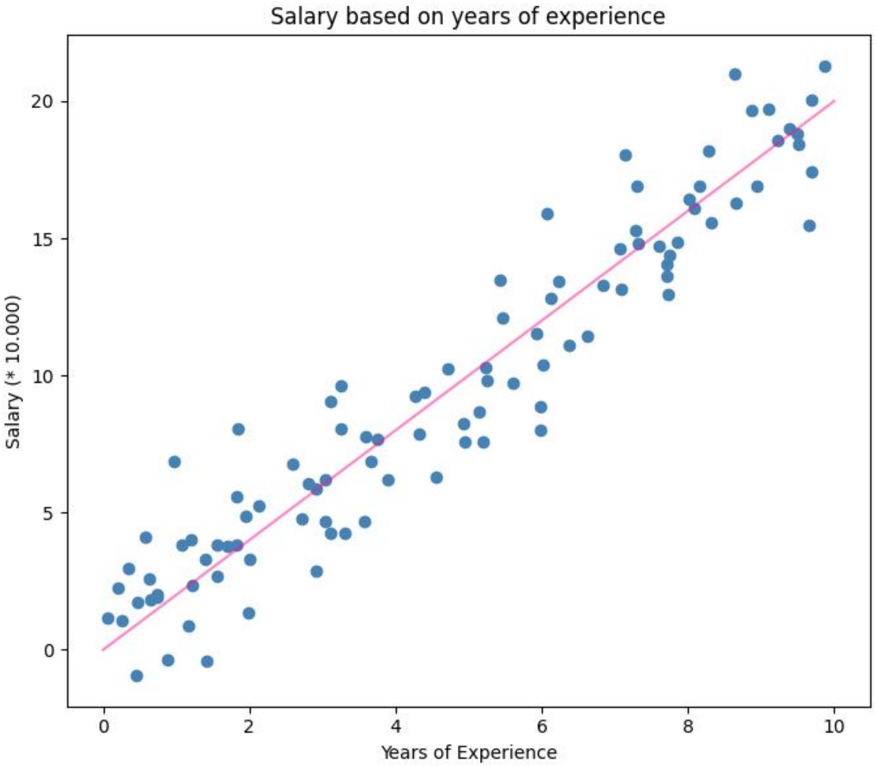
Cambiamos la función de activación por la identidad:

$$O(x) = \sum_{i=1}^{n} x_i \cdot w_i - w_0$$

La salida del perceptrón ya no está confinada a ser binaria: toma valores en los reales

Paper: https://www-isl.stanford.edu/~widrow/papers/c1960adaptiveswitching.pdf

¿CÓMO RESUELVO EL PROBLEMA?



Necesito encontrar los valores de ***w*** que me permitan encontrar un hiperplano que ajuste lo mejor posible al conjunto de datos

$$O(x) = \sum_{i=1}^{n} x_i . w_i - w_0$$

APRENDIZAJE PARA EL PERCEPTRÓN LINEAL

$$O(x) = \sum_{i=1}^{n} x_i . w_i - w_0$$

Rosenblatt: Cada vez que la neurona recibe un estímulo, los pesos sinápticos pueden actualizarse (proceso iterativo):

$$w^{nuevo} = w^{anterior} + \Delta w$$

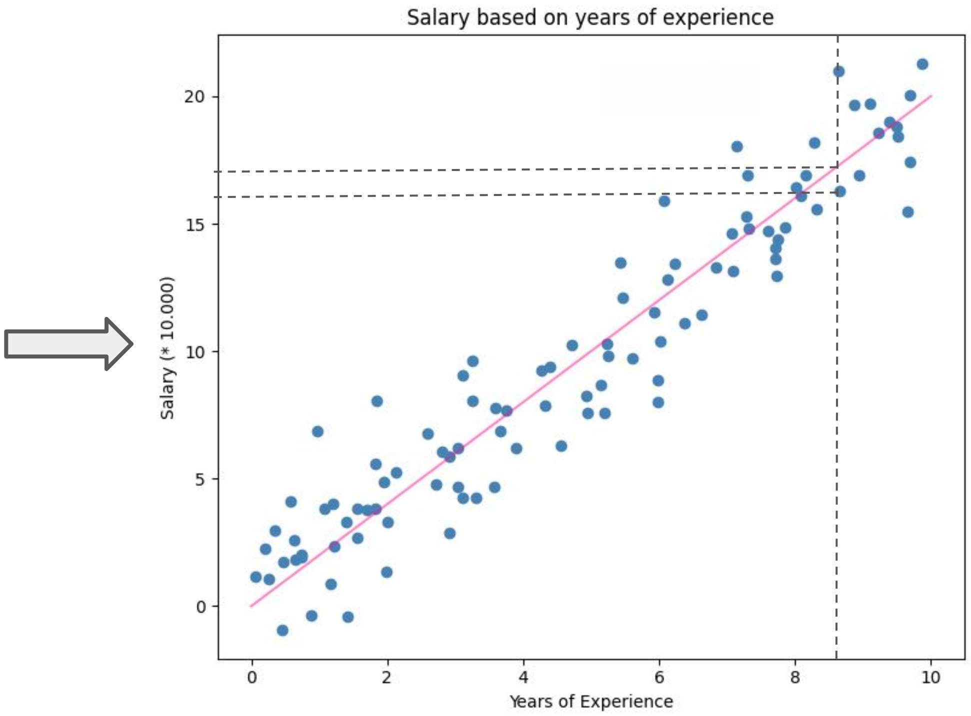
APRENDIZAJE PARA EL PERCEPTRÓN LINEAL

$$O(x) = \sum_{i=1}^{n} x_i . w_i - w_0$$

Rosenblatt: Cada vez que la neurona recibe un estímulo, los pesos sinápticos pueden actualizarse (proceso iterativo):

$$w^{nuevo} = w^{anterior} + \boxed{\Delta w}\ ?$$

¿CÓMO DEFINIMOS QUE EL PERCEPTRÓN SE EQUIVOCA?



¿CÓMO DEFINIMOS QUE EL PERCEPTRÓN SE EQUIVOCA?

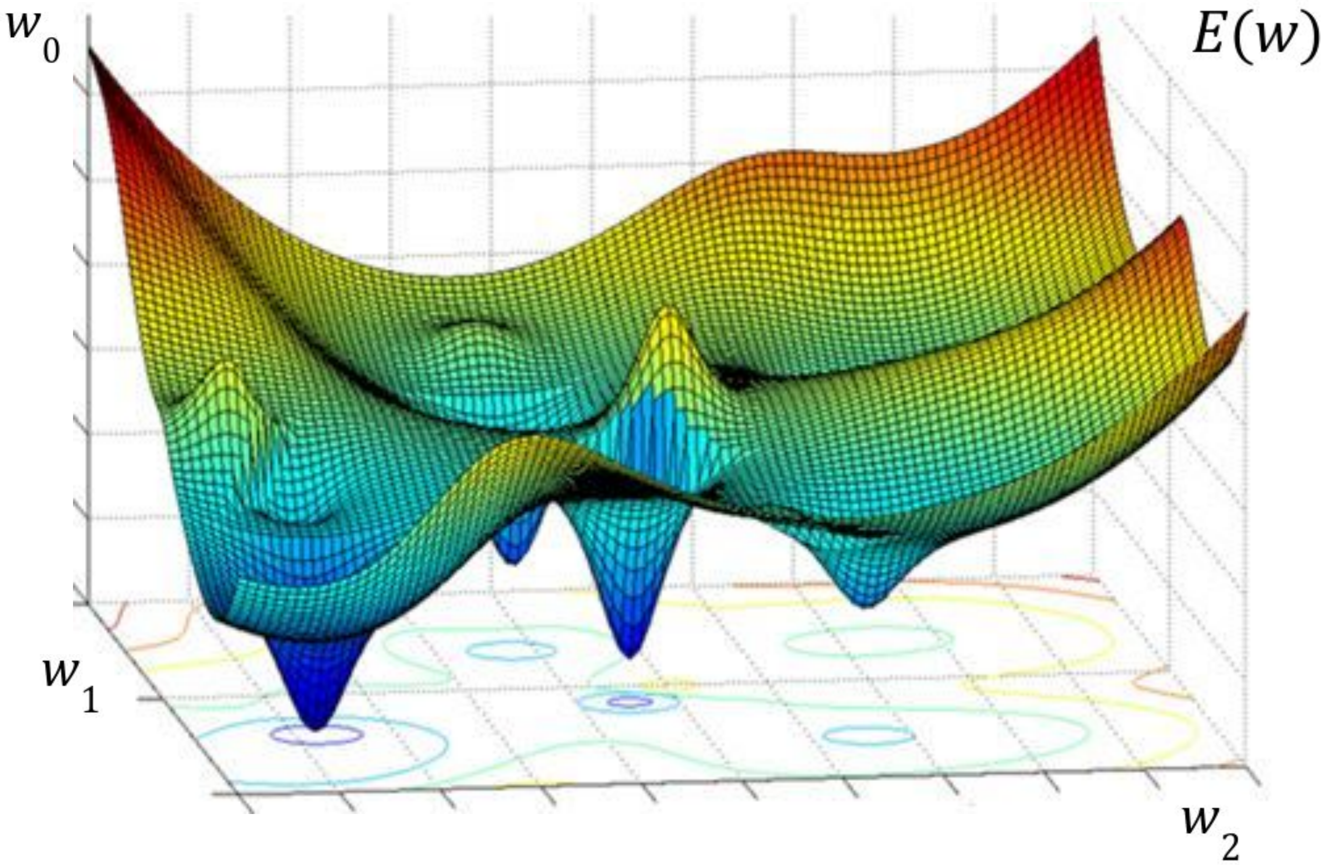
Podemos usar la siguiente función de error o costo:

$$E(O) = \frac{1}{2}(\zeta^{\mu} - O^{\mu})^2$$

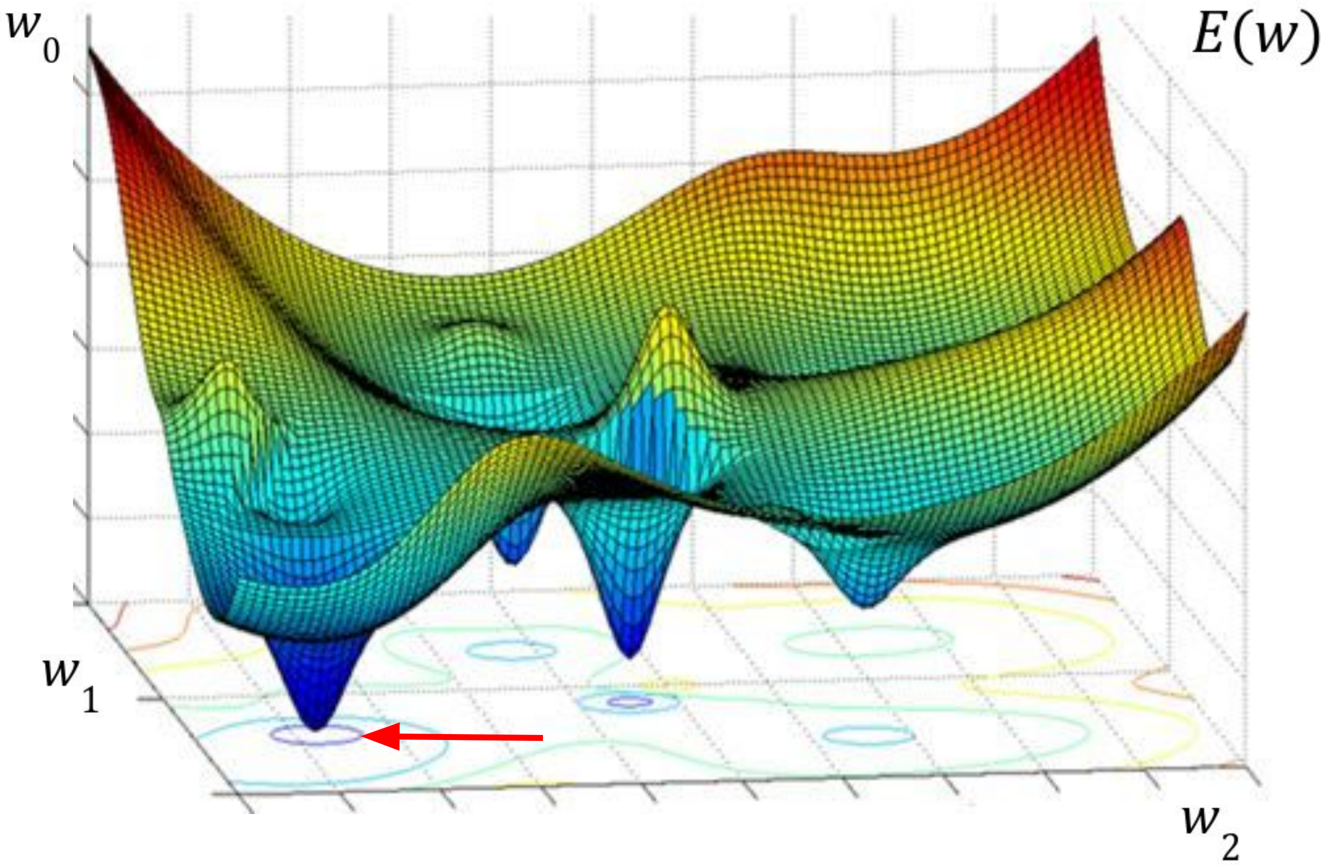
La salida del perceptrón depende, a su vez, de los pesos sinápticos.

$$E(w) = \frac{1}{2}(\zeta^{\mu} - \sum_{i=0}^{n} x_i^{\mu} . w_i)^2$$

¿QUÉ FORMA TIENE LA FUNCIÓN DE COSTO?



¿QUÉ FORMA TIENE LA FUNCIÓN DE COSTO?



¿CÓMO “***APRENDE***” EL PERCEPTRÓN CON ESTA FUNCIÓN DE COSTO?

$$E(w) = \frac{1}{2}\left(\zeta^{\mu} - \sum_{i=0}^{n} x_i^{\mu} . w_i\right)^2$$

Fórmula de actualización de los pesos al evaluar un dato de entrada:

$$w^{nuevo} = w^{anterior} + \Delta w$$

$$\Delta w = - \eta \frac{\partial E}{\partial w}$$

DESARROLLO DE LA EXPRESIÓN PARA Δw

$$\frac{\partial E}{\partial w_i} = \frac{\partial\left(\frac{1}{2}\left(\zeta - \theta\left(\sum_{i=0}^{n} x_i . w_i\right)\right)^2\right)}{\partial w_i}$$

DESARROLLO DE LA EXPRESIÓN PARA Δ*w*

$$\frac{\partial E}{\partial w_i} = \frac{\partial\left(\frac{1}{2}\left(\zeta - \theta\left(\sum_{i=0}^{n} x_i . w_i\right)\right)^2\right)}{\partial w_i} = \left(\zeta - \theta\left(\sum_{i=0}^{n} x_i . w_i\right)\right)(-1)\theta'\left(\sum_{i=0}^{n} x_i . w_i\right)x_i$$

DESARROLLO DE LA EXPRESIÓN PARA Δ*w*

$$O = \theta(\sum_{i=0}^{n} x_i . w_i)$$

$$h = \sum_{i=0}^{n} x_i . w_i$$

$$\frac{\partial E}{\partial w_i} = \frac{\partial(\frac{1}{2}(\zeta - \theta(\sum_{i=0}^{n} x_i . w_i))^2)}{\partial w_i} = \boxed{(\zeta - \theta(\sum_{i=0}^{n} x_i . w_i))}(-1)\theta'(\boxed{\sum_{i=0}^{n} x_i . w_i})x_i$$

DESARROLLO DE LA EXPRESIÓN PARA Δ*w*

$$\frac{\partial E}{\partial w_i} = \frac{\partial\left(\frac{1}{2}\left(\zeta - \theta\left(\sum_{i=0}^{n} x_i . w_i\right)\right)^2\right)}{\partial w_i} = (\zeta - \theta(\sum_{i=0}^{n} x_i . w_i))(-1)\theta'(\sum_{i=0}^{n} x_i . w_i)x_i$$

$$= (\zeta - O)(-1)\theta'(h)x_i = \boxed{-(\zeta - O)\theta'(h)x_i}$$

DESARROLLO DE LA EXPRESIÓN PARA Δ*w*

$$\frac{\partial E}{\partial w_i} = \frac{\partial\left(\frac{1}{2}\left(\zeta - \theta\left(\sum_{i=0}^{n} x_i . w_i\right)\right)^2\right)}{\partial w_i} = \left(\zeta - \theta\left(\sum_{i=0}^{n} x_i . w_i\right)\right)(-1)\theta'\left(\sum_{i=0}^{n} x_i . w_i\right)x_i$$

$$= (\zeta - O)(-1)\theta'(h)x_i = \boxed{-(\zeta - O)\theta'(h)x_i}$$

Para el perceptrón lineal, Θ es la identidad (la derivada es 1)

$$\Delta w = -\eta\frac{\partial E}{\partial w} = \eta(\zeta^{\mu} - O^{\mu})x^{\mu}$$

¿CÓMO "***APRENDE***" EL PERCEPTRÓN LINEAL?

$$E(w) = \frac{1}{2}\left(\zeta^{\mu} - \sum_{i=0}^{n} x_i^{\mu} . w_i\right)^2$$

Fórmula de actualización de los pesos al evaluar un dato de entrada **para el perceptrón lineal**:

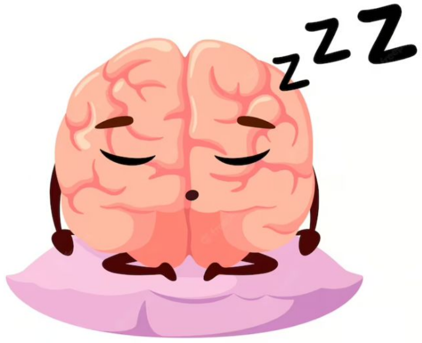
$$w^{nuevo} = w^{anterior} + \Delta w$$

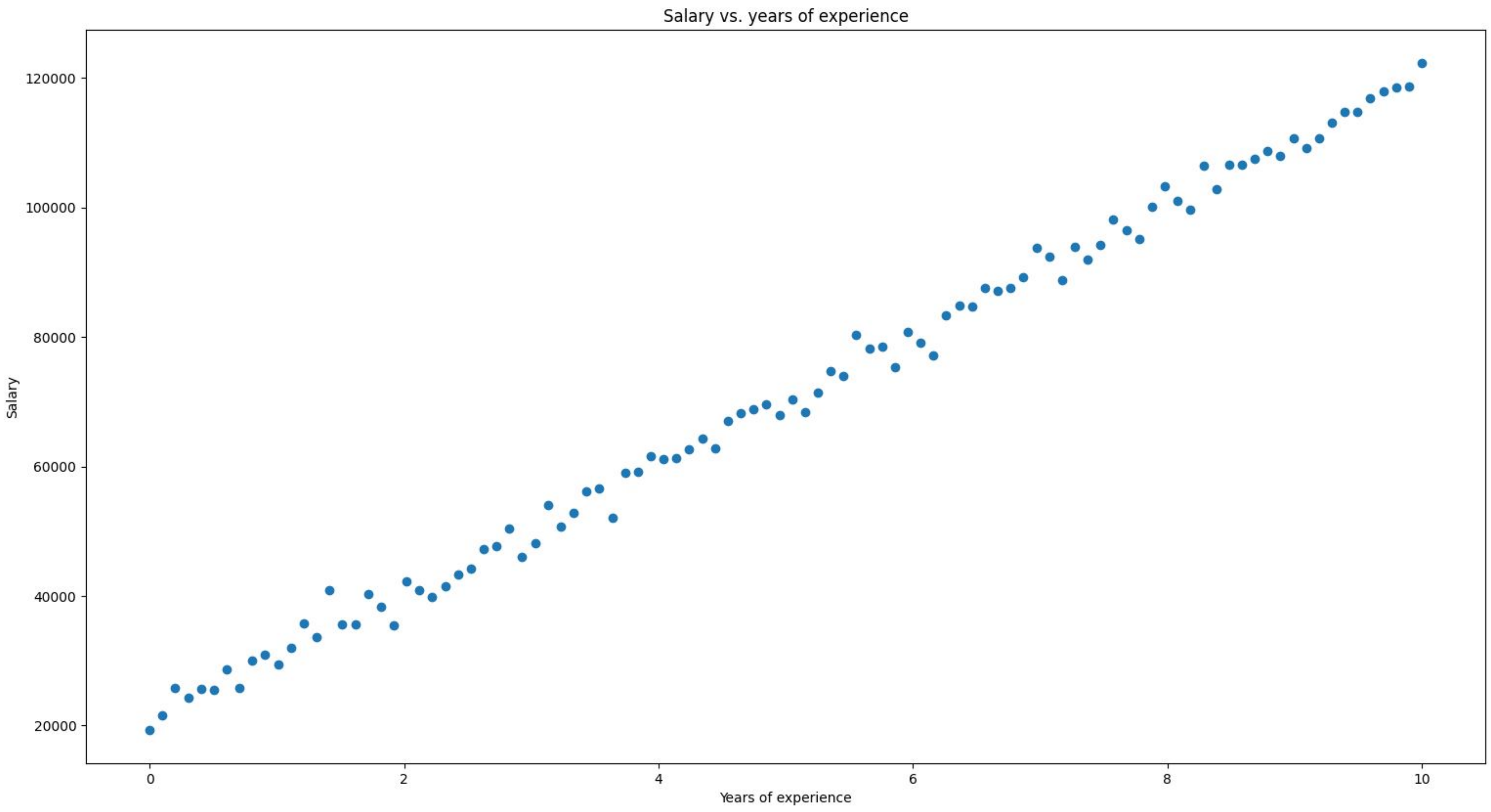
$$\Delta w = -\eta \frac{\partial E}{\partial w} = \eta(\zeta^{\mu} - O^{\mu})x^{\mu}$$

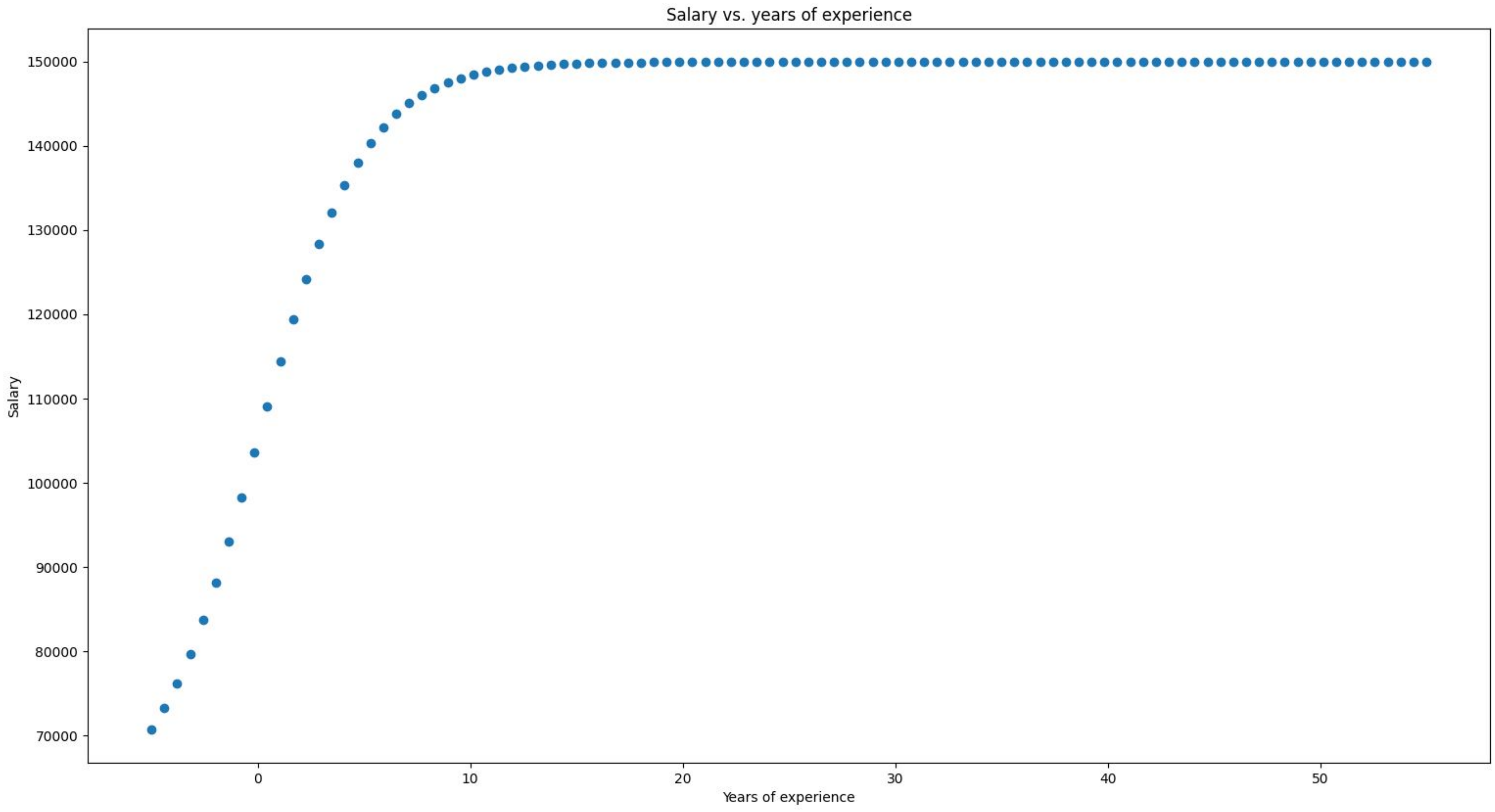
EL ALGORITMO PARA ADALINE ES IGUAL AL ANTERIOR

1. Inicializar los pesos sinápticos en valores aleatorios pequeños o cero $w = (w_0, w_1, \ldots w_n)$
2. Definir: tasa de aprendizaje, épocas.
3. Para cada elemento del conjunto de datos
    a. Calcular la salida de la neurona $O = \theta(\sum_{i=1}^{n} x_i . w_i - w_0)$
    b. Actualizar los pesos sinápticos $\boxed{\Delta w = - \eta \frac{\partial E}{\partial w}}$
4. Calcular el error del perceptrón para verificar si se alcanzó convergencia
    a. Si el perceptrón alcanzó convergencia, finalizar.

    $MSE = \frac{1}{p} \sum_{\mu=0}^{p-1} (\zeta^{\mu} - O^{\mu})^2$
5. Repetir 3 y 4 hasta alcanzar convergencia o hasta finalizar la cantidad de épocas




Salary vs. years of experience
Salary
Years of experience
20000
40000
60000
80000
100000
120000
0
2
4
6
8
10


Salary vs. years of experience
Salary
Years of experience
150000
140000
130000
120000
110000
100000
90000
80000
70000
0
10
20
30
40
50

PERCEPTRÓN SIMPLE NO LINEAL

Cambiamos la función de activación por una sigmoidea, ***tanh*** o ***logística***

$$O = \theta\left(\sum_{i=0}^{n} x_i . w_i\right)$$

$$\theta(x) = tanh(\beta x) \quad Im = (-1, 1)$$

$$\theta(x) = \frac{1}{1+exp^{-2\beta x}} \quad Im = (0, 1)$$

La fórmula de error se mantiene igual al perceptrón lineal:

$$E(O) = \frac{1}{2}(\zeta^{\mu} - O^{\mu})^{2}$$

¿Cómo "***aprende***"? ¡Igual que el anterior!

Cambiamos la función de activación por una sigmoidea, ***tanh*** o ***logística***

$$O = \theta(\sum_{i=0}^{n} x_i . w_i)$$

Fórmula de actualización de los pesos al evaluar un dato de entrada:

$$w^{nuevo} = w^{anterior} + \Delta w$$

$$\Delta w = -\eta \frac{\partial E}{\partial w} = \eta(\zeta^{\mu} - O^{\mu})\theta'(h)x^{\mu}$$

Funciones sigmoideas

Tangente Hiperbólica

Parámetro ***beta*** que cambia la forma:

$$\theta(h) = tanh(\beta h)$$

$$\theta'(h) = \beta(1 - \theta^2(h))$$

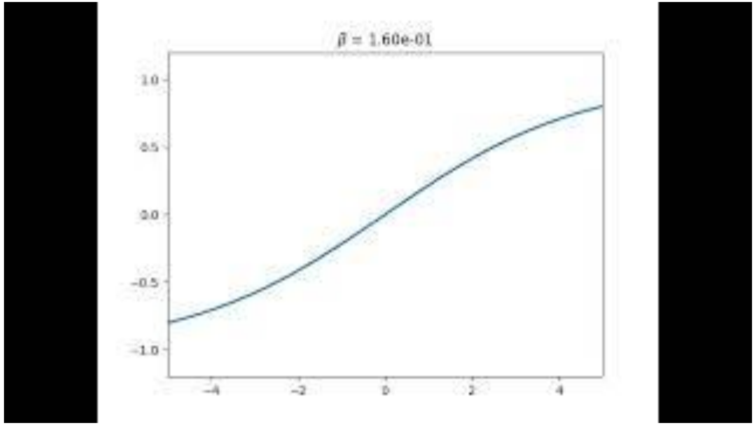
Función Logística

Parámetro ***beta*** que cambia la forma:

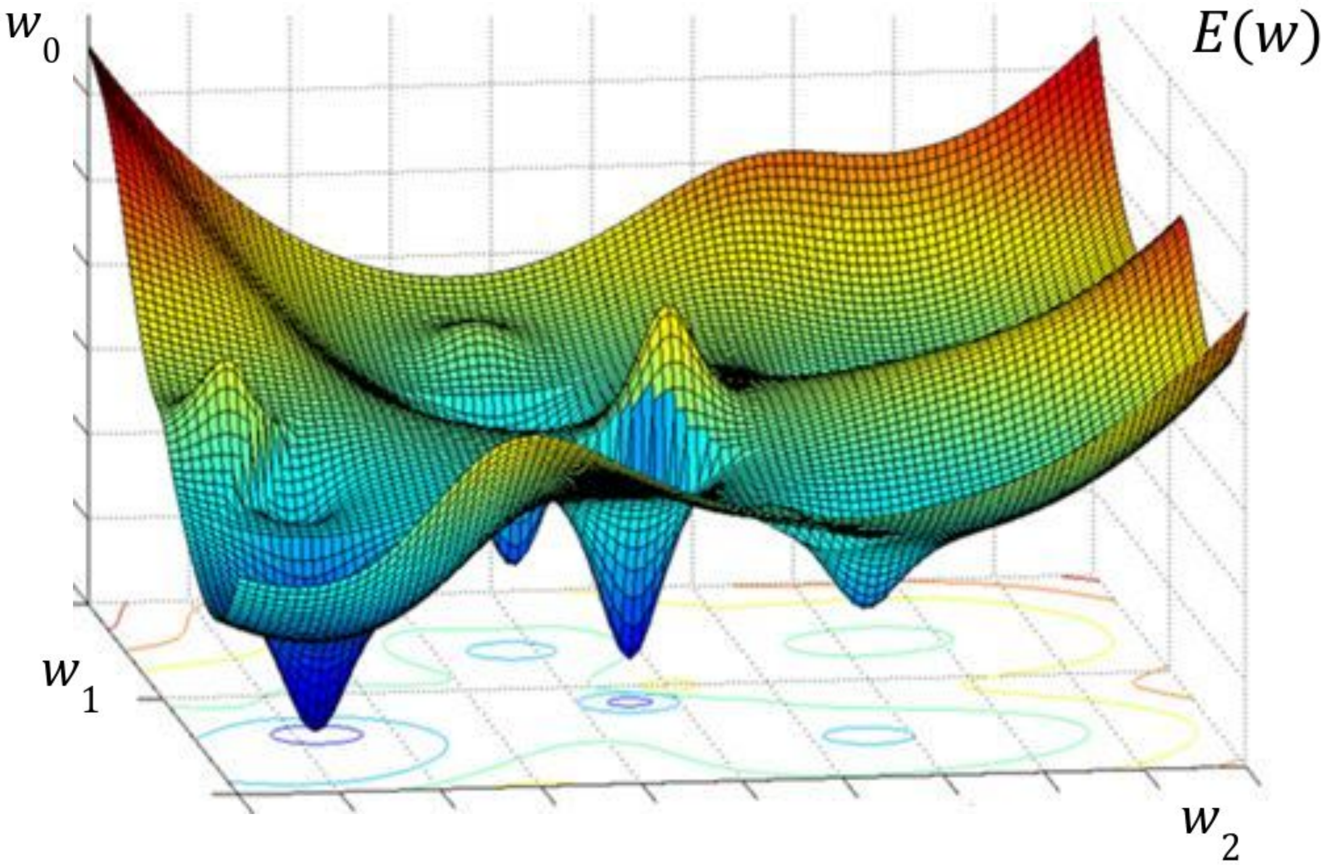
$$\theta(h) = \frac{1}{1+exp^{-2\beta h}}$$

$$\theta'(h) = 2\beta\theta(h)(1 - g(h))$$

[Variación de tanh de acuerdo a beta]



OBSERVACIÓN: LA IMPORTANCIA DE η



¿EXISTEN OTROS MÉTODOS PARA RESOLVER ESTOS PROBLEMAS?

- Regresión Lineal (Least Mean Squares)
- Regresión Logística
- SVM
- …

RESUMEN

- Los perceptrones simples lineal y no lineal extienden la utilidad del perceptrón simple escalón
- Si las funciones de activación son derivables, encontramos cómo actualizar los pesos usando el algoritmo de gradiente descendente.
- El ajuste del parámetro tasa de aprendizaje impacta fuertemente en la convergencia hacia los pesos que deseamos (aquellos que minimizan la función de costo)

Lectura recomendada: https://towardsdatascience.com/why-we-need-bias-in-neural-networks-db8f7e07cb98

ALGUNAS VARIACIONES POSIBLES

Variable	Posibles Valores
Función de activación (Θ)	Escalón, Identidad, Sigmoidea
Tasa de aprendizaje (η)	~0.1
Actualización de los pesos	Batch/Online
Error del perceptrón (función de costo)	Accuracy, suma valores absolutos, MSE
Épocas	
Método de optimización	Gradiente Descendente
Parámetros de función de activación	□ de tanh o de log
Técnica para separar en entrenamiento y testeo	Ejemplo: 80%-20%