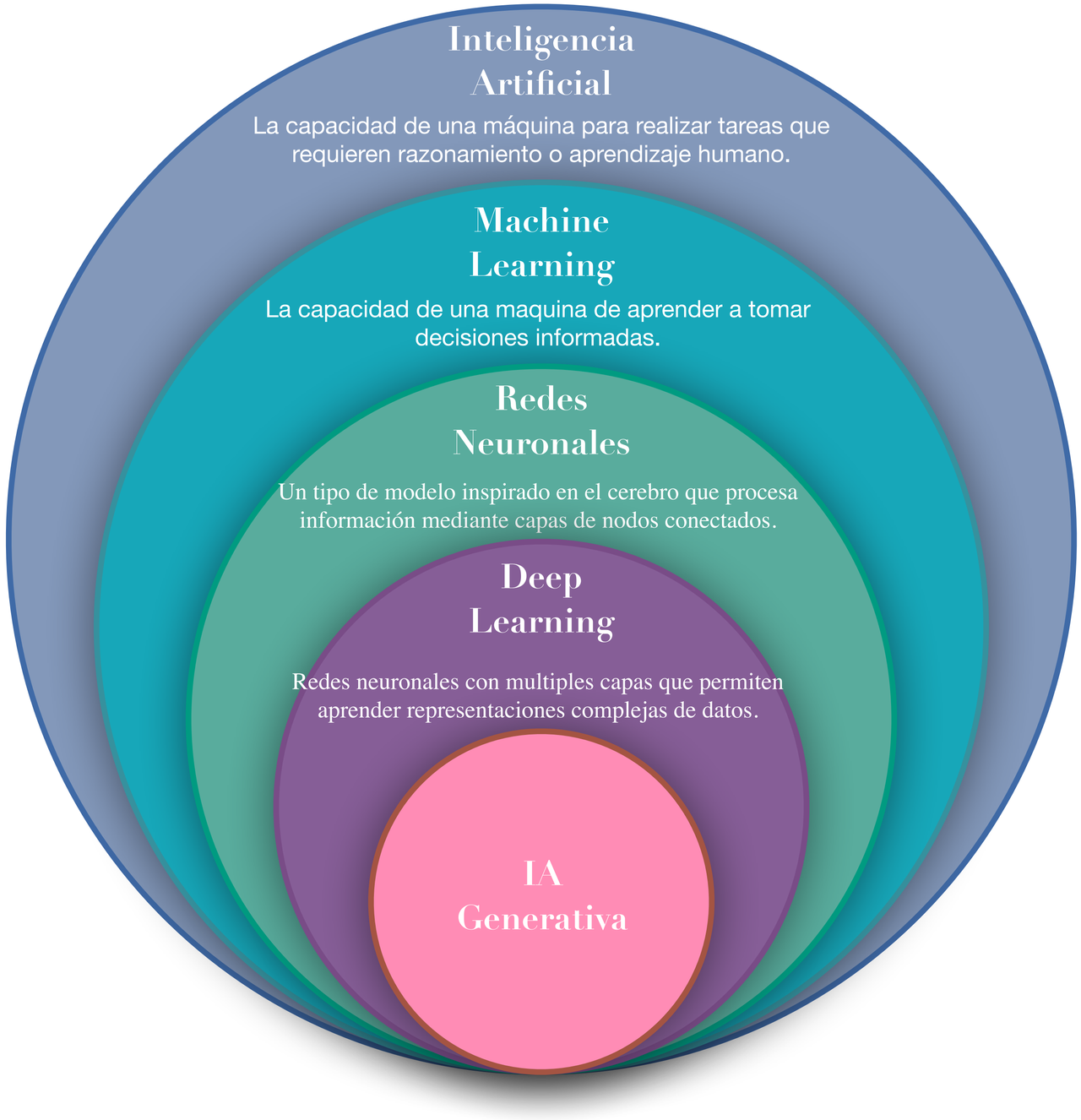
FCEN, UBA

Introducción a las Redes Profundas

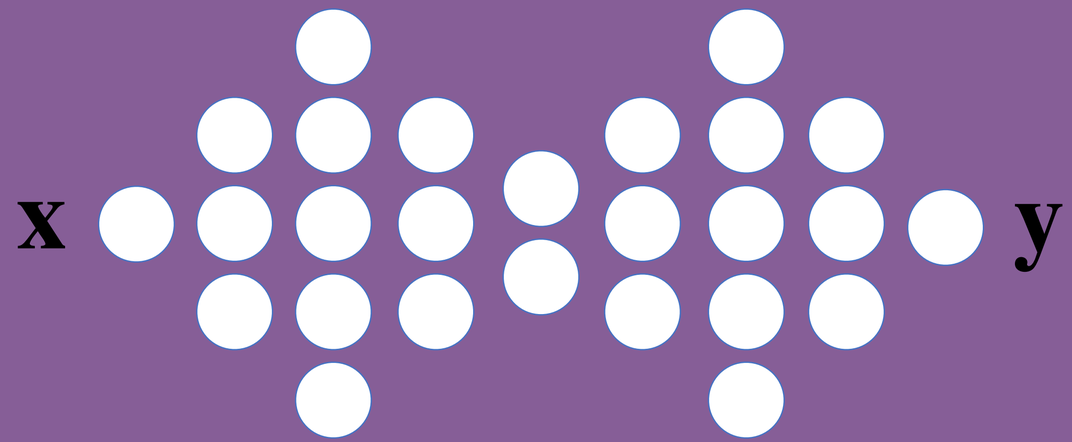
Cecilia Garraffo

Outline

- Descenso del Gradiente
- Estocasticidad para Funciones de Pérdida no Convexas
- Aprendizaje Adaptativo
- Teorema de Aproximación Universal
- Redes Neuronales Profundas
- Propuestas Para el Double Descent


Inteligencia Artificial
La capacidad de una máquina para realizar tareas que requieren razonamiento o aprendizaje humano.
Machine Learning
La capacidad de una maquina de aprender a tomar decisiones informadas.
Redes Neuronales
Un tipo de modelo inspirado en el cerebro que procesa información mediante capas de nodos conectados.
Deep Learning
Redes neuronales con multiples capas que permiten aprender representaciones complejas de datos.
IA Generativa

Deep
Learning

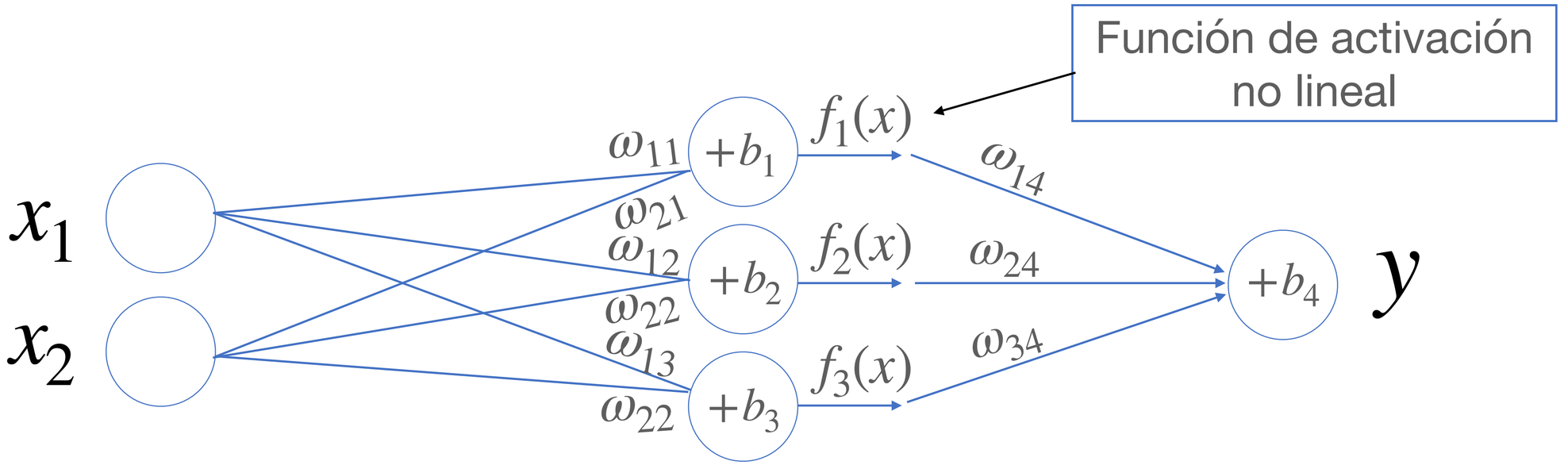


$$\mathbf{y} = \mathbf{F}[\mathbf{x}, \theta]$$

Recap

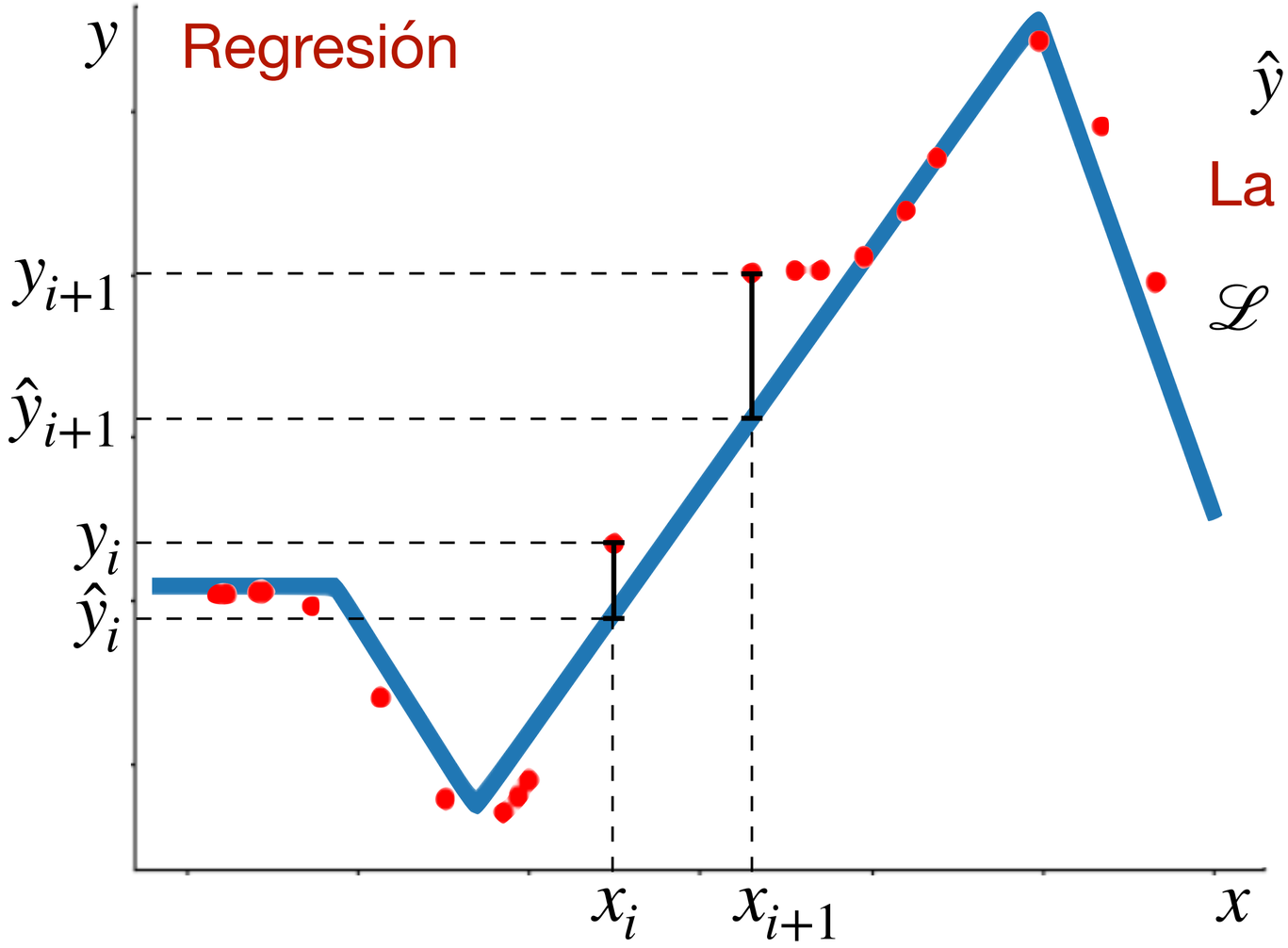
- Redes Neuronales Shallow
- Funciones de Activación
- Ajuste de Curvas con Redes Neuronales Shallow
- Funciones de Pérdida

Redes Neuronales: Shallow



$$y = \omega_{14} f_1(x_1\omega_{11} + x_2\omega_{21} + b_1) + \omega_{24} f_2(x_1\omega_{12} + x_2\omega_{22} + b_2) + \omega_{34} f_3(x_1\omega_{13} + x_2\omega_{23} + b_3) + b_4$$

Redes Neuronales: Función Pérdida



$$\hat{y} = f[x, \theta]$$

La función de pérdida $\mathscr{L}$

$$\mathscr{L} = \frac{1}{n}\sum_i (y_i - \hat{y}_i)^2 = MSE$$

Redes Neuronales: Función Pérdida

$$\mathscr{L}_{MSE} = \frac{1}{n}\sum_{i}(y_i - \hat{y}_i)^2$$

$$\mathscr{L}_{MAE} = \frac{1}{n}\sum_{i}|y_i - \hat{y}_i|$$

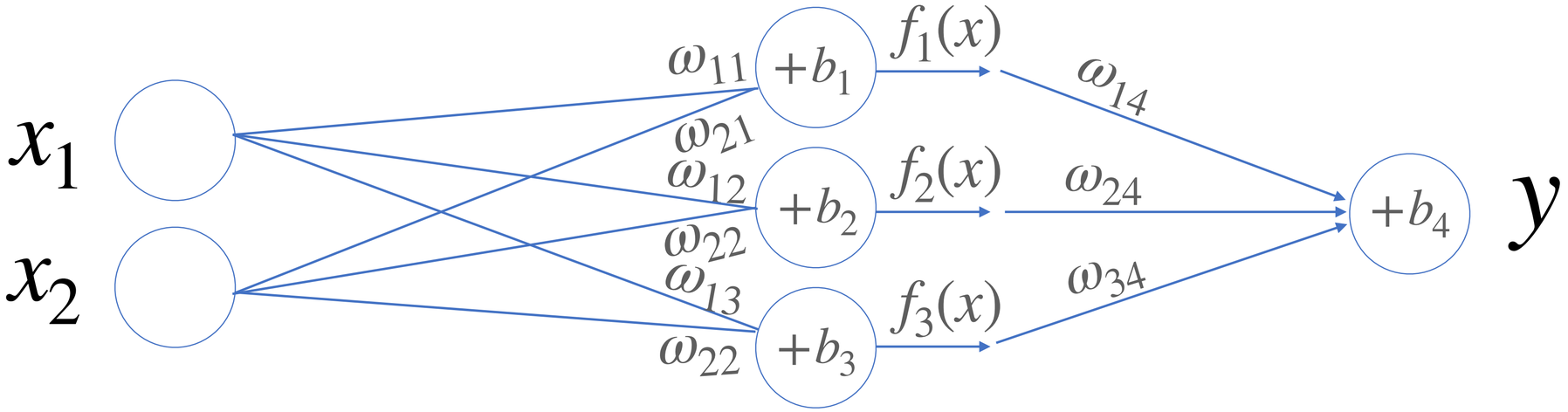
$$\mathscr{L}_{BCE} = -\left[y\log(\hat{y}) + (1-y)\log(1-\hat{y})\right]$$

$$\mathscr{L}_{CE} = -\sum_{k=1}^{K} y_k \log(\hat{y}_k)$$

• • •

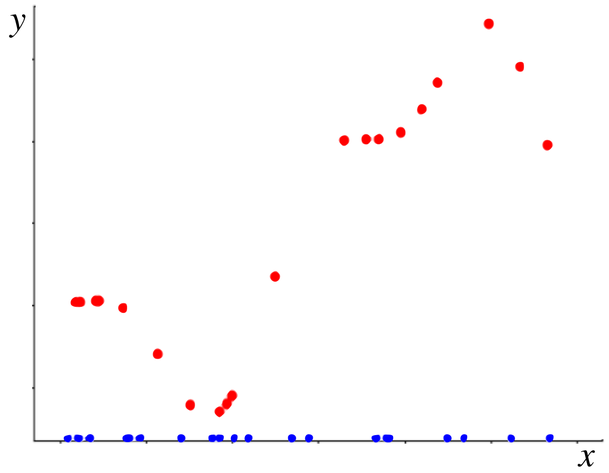
Redes Neuronales: Shallow

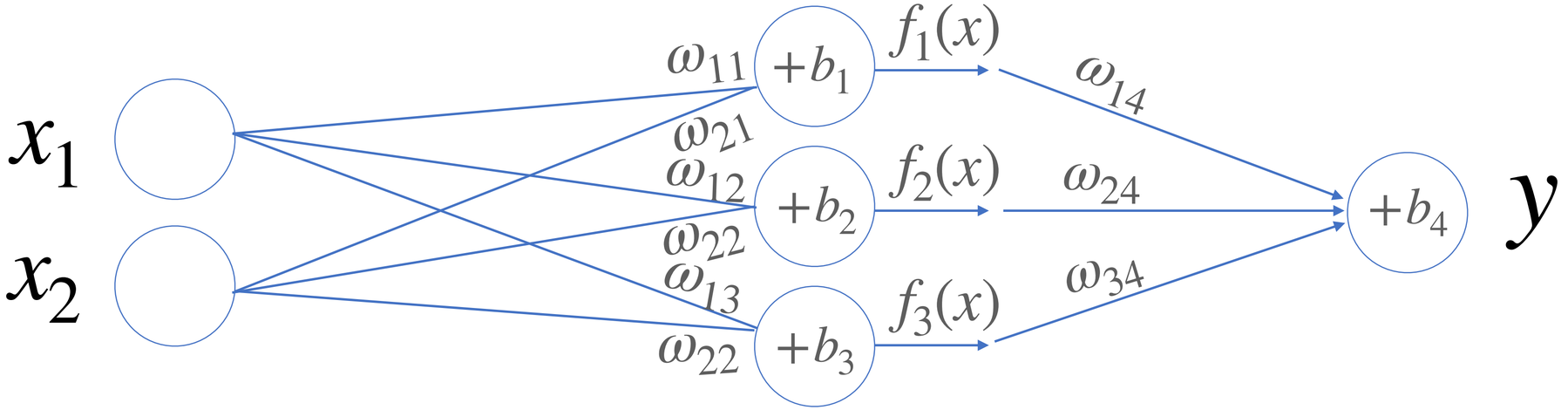
Optimizar los parámetros de la red: minimizar la función de pérdida



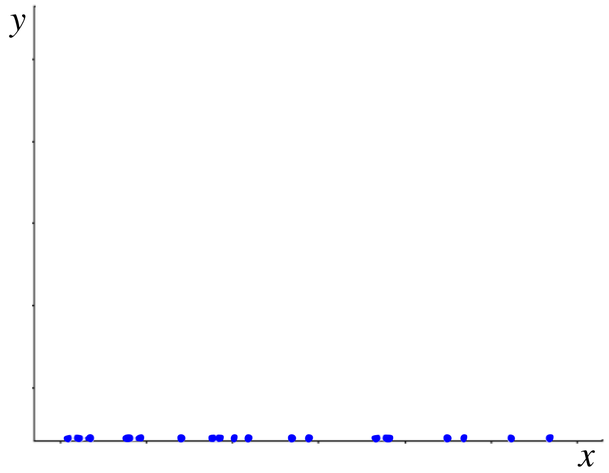
$$\mathscr{L}_{MSE} = \frac{1}{n}\sum_{i}(y_i - \hat{y}_i)^2$$

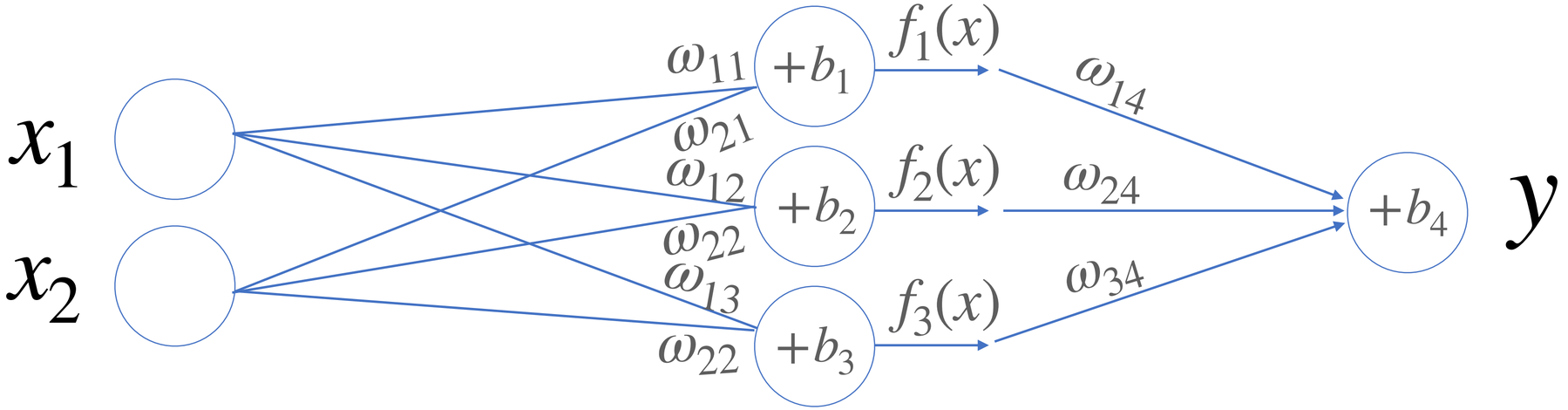
Redes Neuronales: Función Pérdida



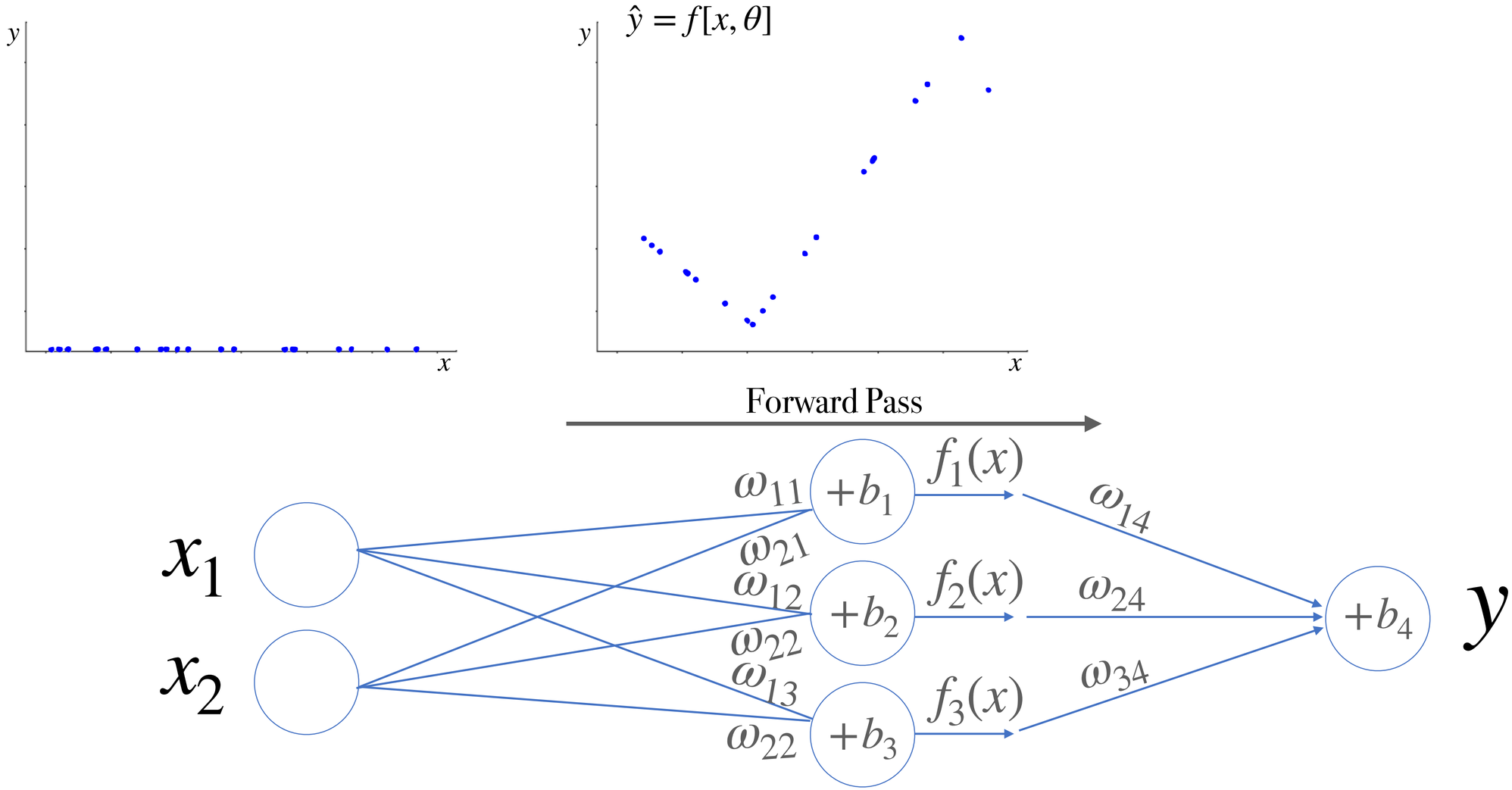


Redes Neuronales: Función Pérdida

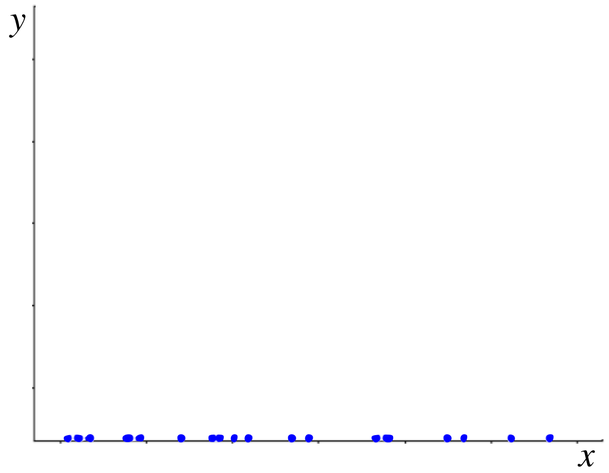


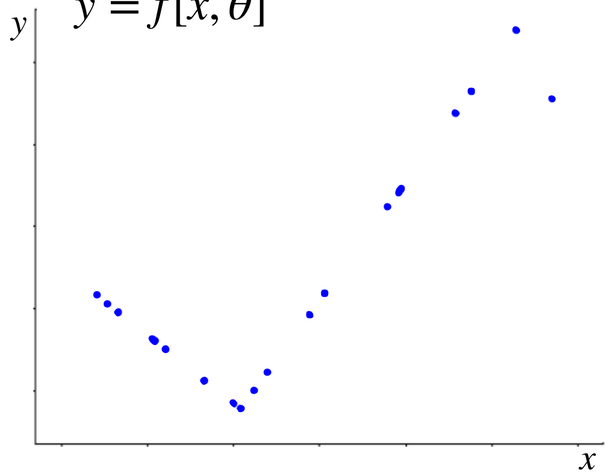


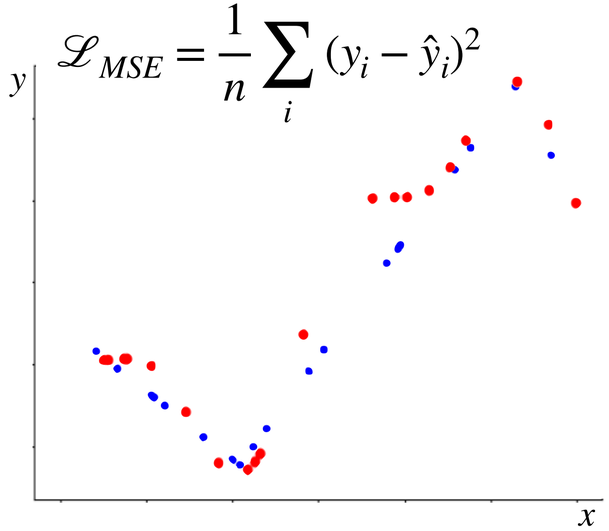
Redes Neuronales: Función Pérdida

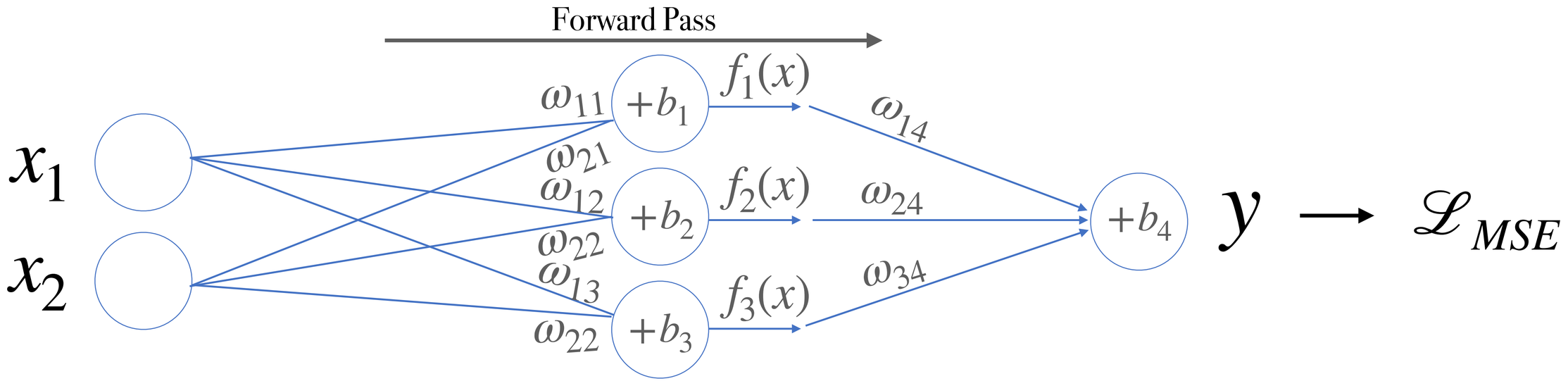


Redes Neuronales: Función Pérdida

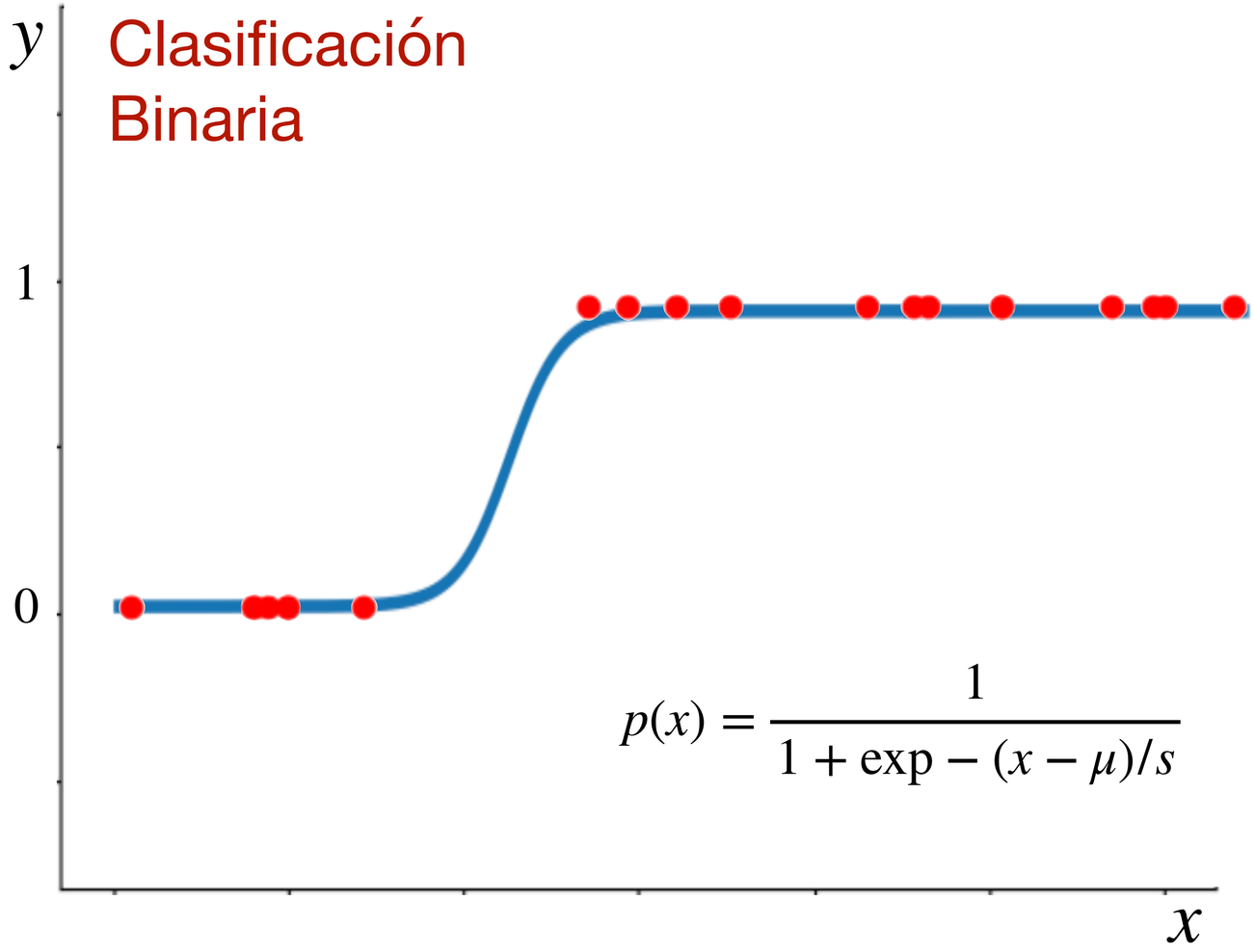








Redes Neuronales: Función Pérdida



$$p(x) = \frac{1}{1 + \exp - (x - \mu)/s}$$

$$\hat{y} = \begin{cases} 1 & p(x) \geq 0.5 \\ 0 & p(x) < 0.5 \end{cases}$$

Redes Neuronales: Clasificación

Métricas de Evaluación

	Detección de GW	**Ruido**
Hay GW	TP	FN
No hay GW	FP	TN

$$\text{Precisión} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

Redes Neuronales: Clasificación

$$\mathscr{L}_{BCE} = -\left[y \log(\sigma(x_k)) + (1-y)\log(1-\sigma(x_k)) \right]$$

$$\mathscr{L}_{CE} = -\sum_{k=1}^{K} y_k \log(\sigma(x_k))$$

Fin de Recap

Redes Neuronales: Clasificación

Matriz de Confusión

Real ↓ \ Predicho →	⭐ Estrella	🌌 Galaxia	🔴 Cuásar
⭐ **Estrella**	**80**	15	5
🌌 **Galaxia**	10	**70**	20
🔴 **Cuásar**	2	8	**90**

Redes Neuronales: Clasificación

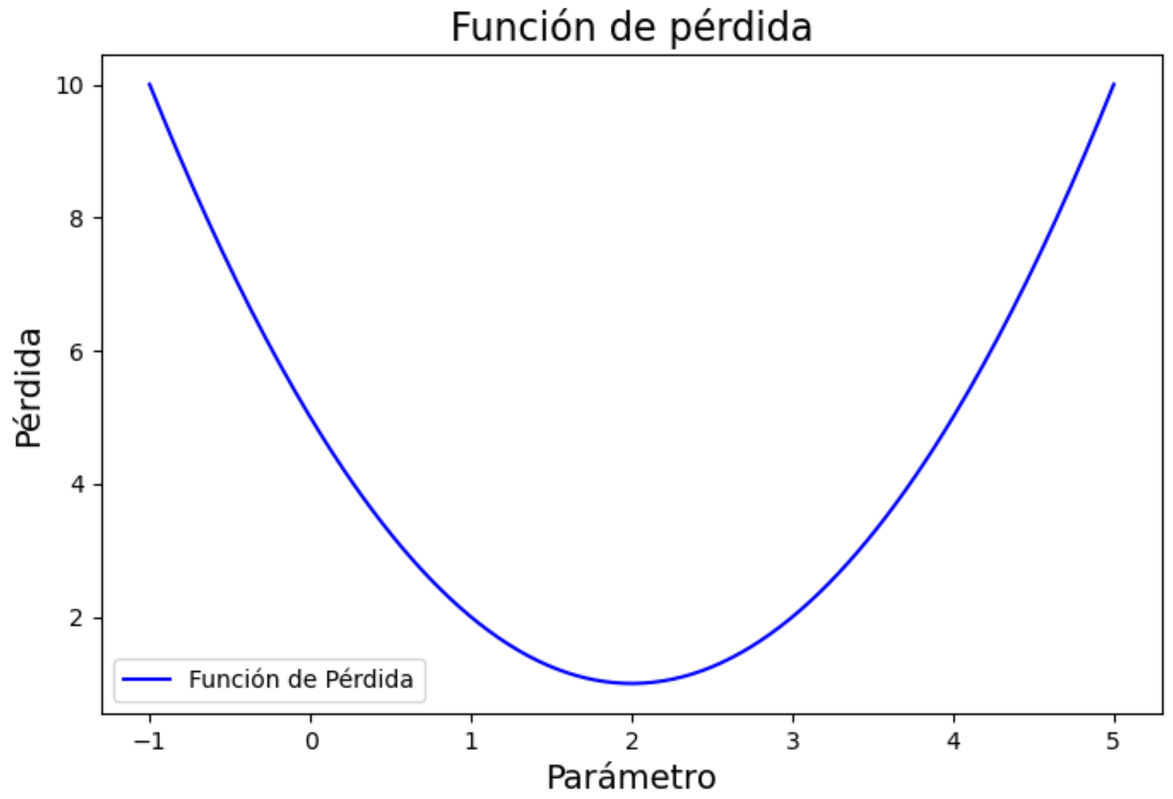
Matriz de Confusión

Real ↓ \ Predicho →	⭐ Estrella	🌌 Galaxia	🔴 Cuásar
⭐ **Estrella**	**80**	15	5
🌌 **Galaxia**	10	**70**	20
🔴 **Cuásar**	2	8	**90**

Quiz (10pts): Calcular la precisión y el recall de cada clase

Redes Neuronales: Descenso del Gradiente

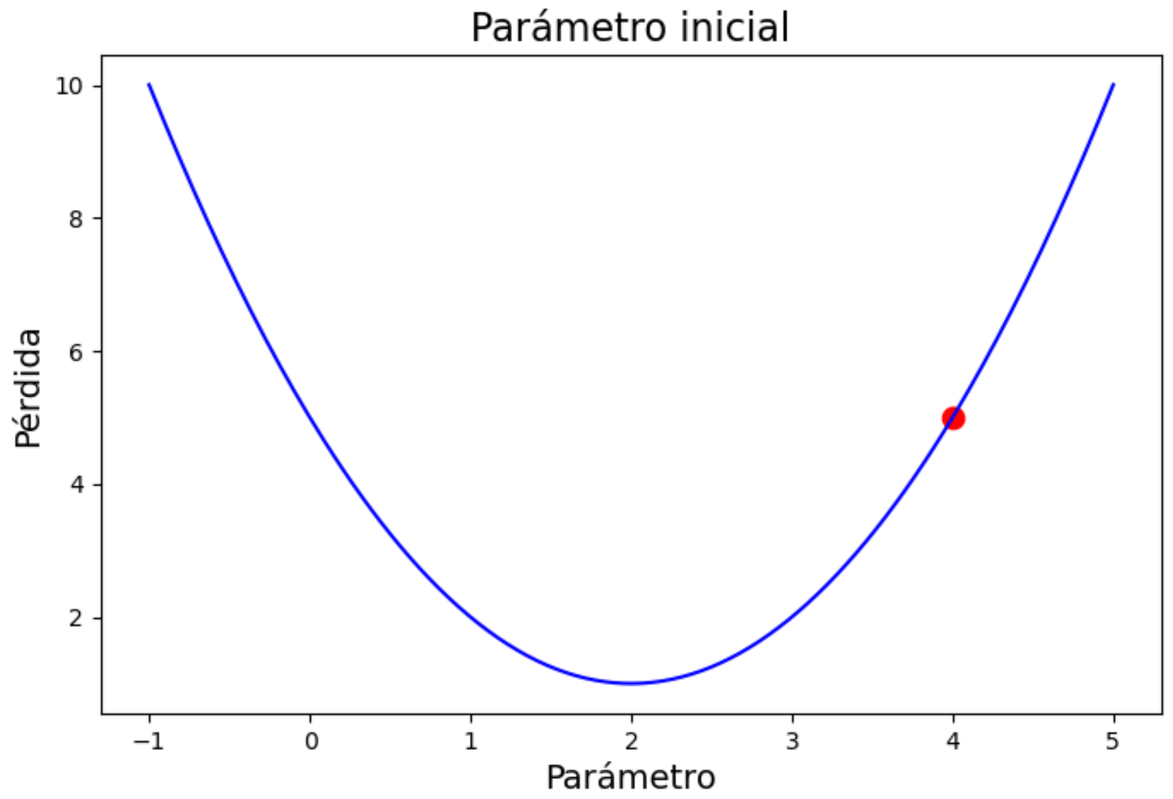
$$\mathscr{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}(x_i;\theta))^2$$



Redes Neuronales: Descenso del Gradiente

$$\mathscr{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}(x_i;\theta))^2$$

$$\mathscr{L}(\theta_{ini}) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}(x_i;\theta_{ini})\right)^2$$

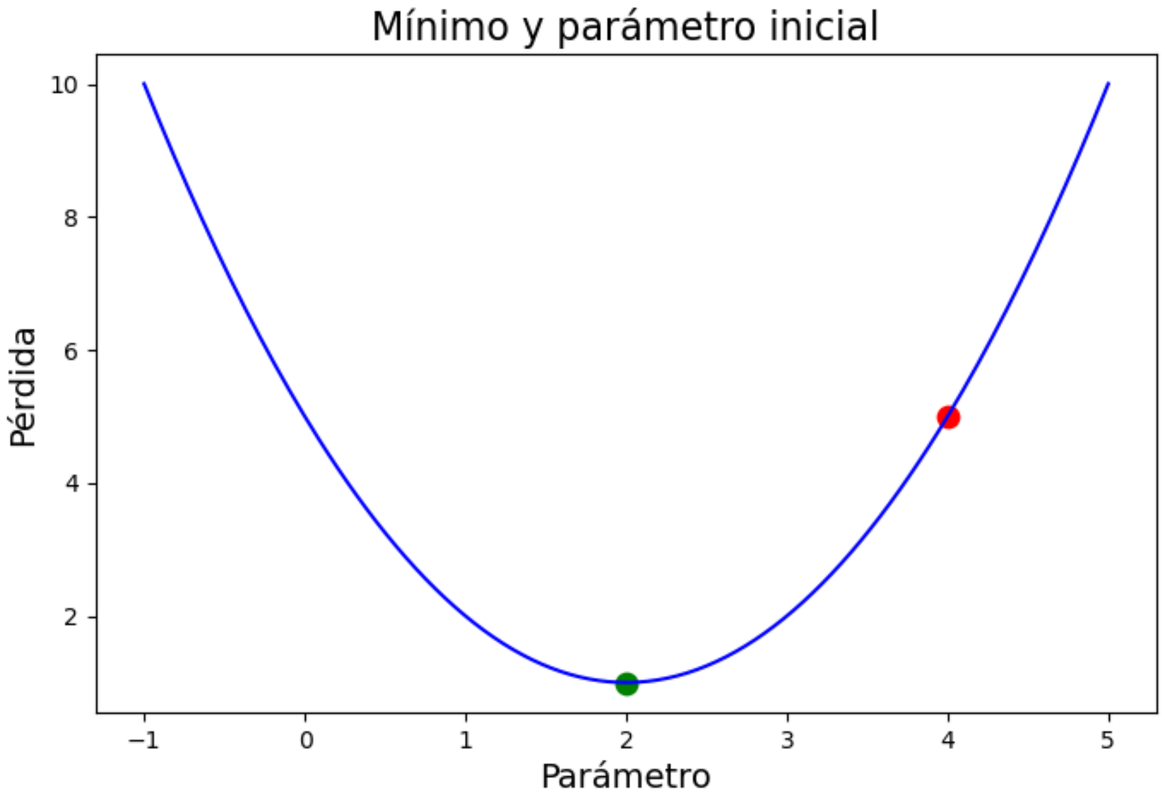


Redes Neuronales: Descenso del Gradiente

$$\mathscr{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}(x_i;\theta))^2$$

$$\mathscr{L}(\theta_{ini}) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}(x_i;\theta_{ini})\right)^2$$

$$\mathscr{L}(\theta_{min}) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}(x_i;\theta_{min})\right)^2$$



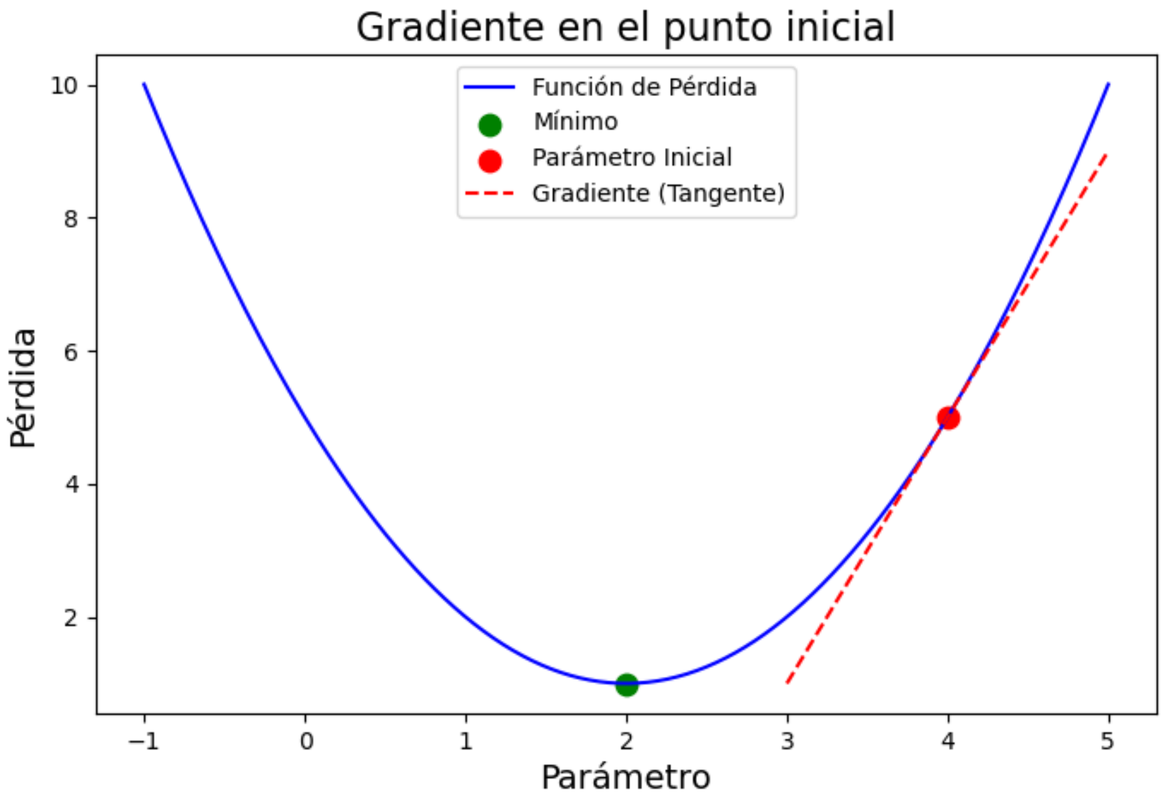
Redes Neuronales: Descenso del Gradiente

$$\mathscr{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}(x_i;\theta))^2$$

$$\mathscr{L}(\theta_{ini}) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}(x_i;\theta_{ini})\right)^2$$

$$\mathscr{L}(\theta_{min}) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}(x_i;\theta_{min})\right)^2$$

$$\nabla_\theta \mathscr{L}(\theta) = \frac{\partial \mathscr{L}(\theta)}{\partial \theta}$$



Redes Neuronales: Descenso del Gradiente

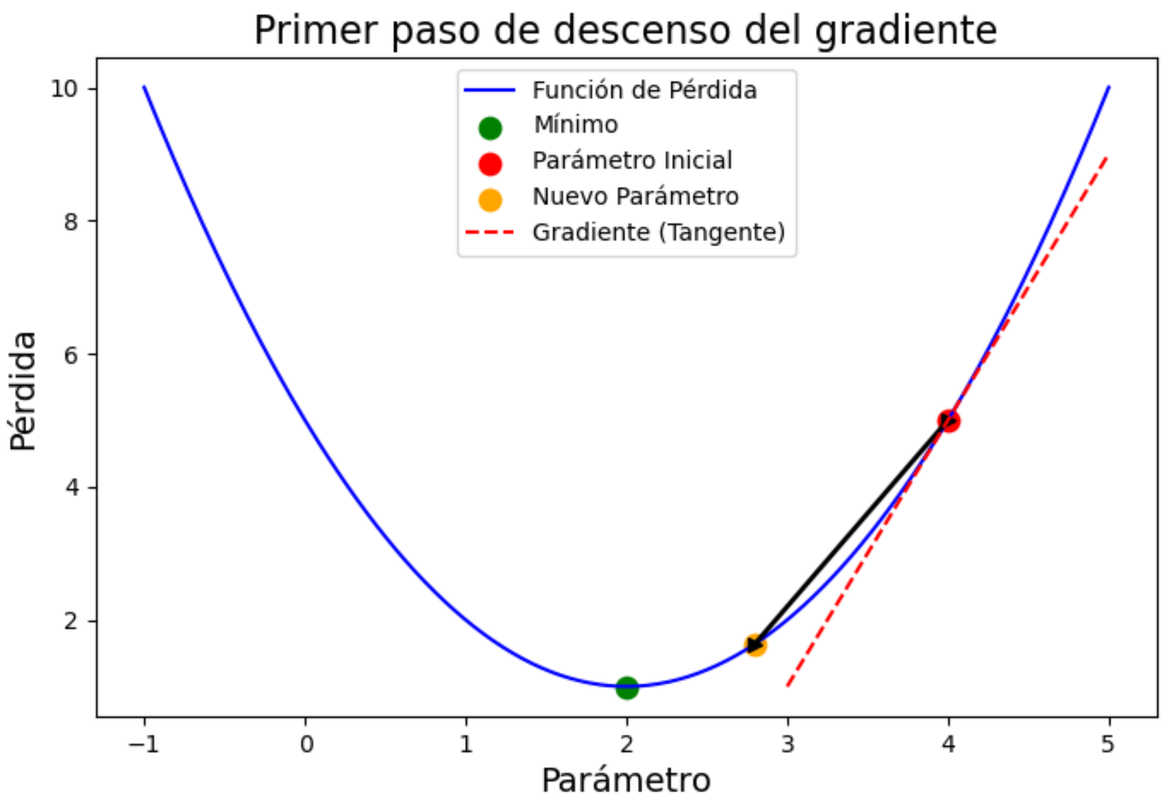
$$\mathscr{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}(x_i;\theta))^2$$

$$\mathscr{L}(\theta_{ini}) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}(x_i;\theta_{ini})\right)^2$$

$$\mathscr{L}(\theta_{min}) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}(x_i;\theta_{min})\right)^2$$

$$\nabla_\theta \mathscr{L}(\theta) = \frac{\partial \mathscr{L}(\theta)}{\partial \theta}$$

$$\theta_{new} = \theta_{old} - \eta \nabla_\theta \mathscr{L}(\theta_{old})$$



Redes Neuronales: Descenso del Gradiente

$$\mathscr{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}(x_i;\theta))^2$$

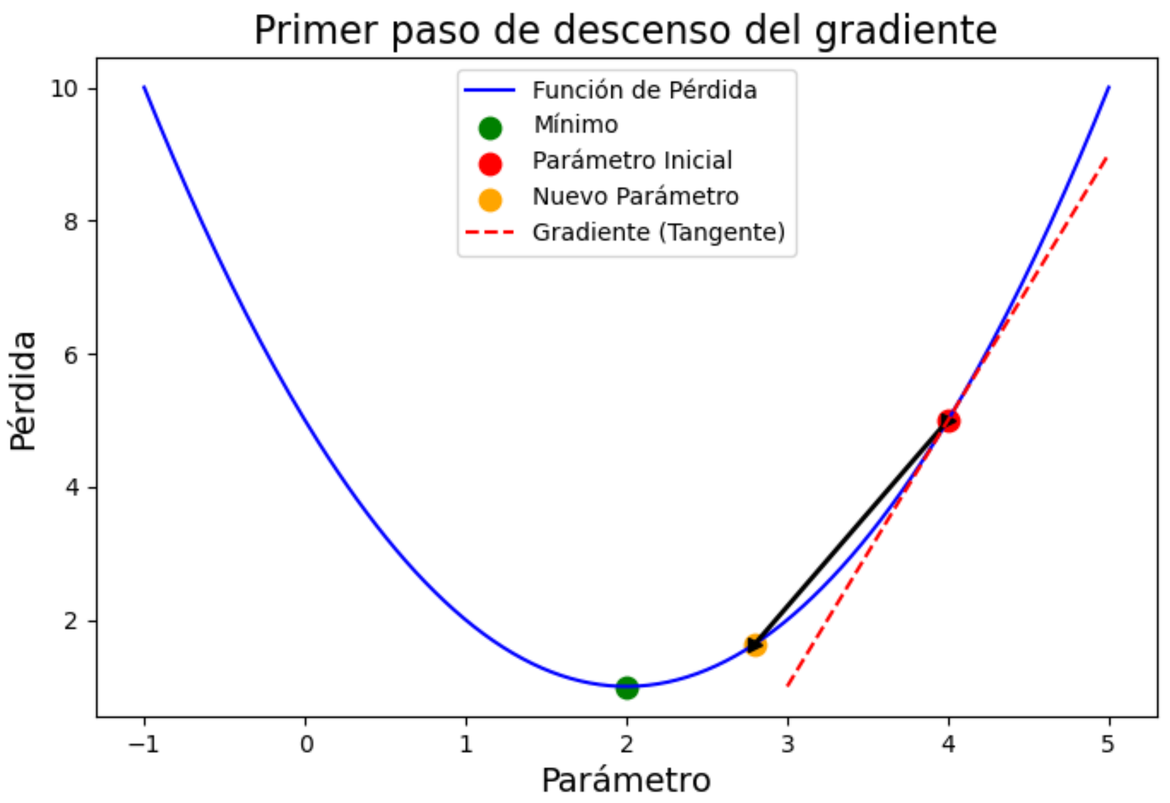
$$\mathscr{L}(\theta_{ini}) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}(x_i;\theta_{ini})\right)^2$$

$$\mathscr{L}(\theta_{min}) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}(x_i;\theta_{min})\right)^2$$

$$\nabla_\theta \mathscr{L}(\theta) = \frac{\partial \mathscr{L}(\theta)}{\partial \theta}$$

$$\theta_{new} = \theta_{old} - \eta \nabla_\theta \mathscr{L}(\theta_{old})$$

$\theta_{ini} = = \theta_{new}$



Redes Neuronales: Descenso del Gradiente

$$\mathscr{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}(x_i;\theta))^2$$

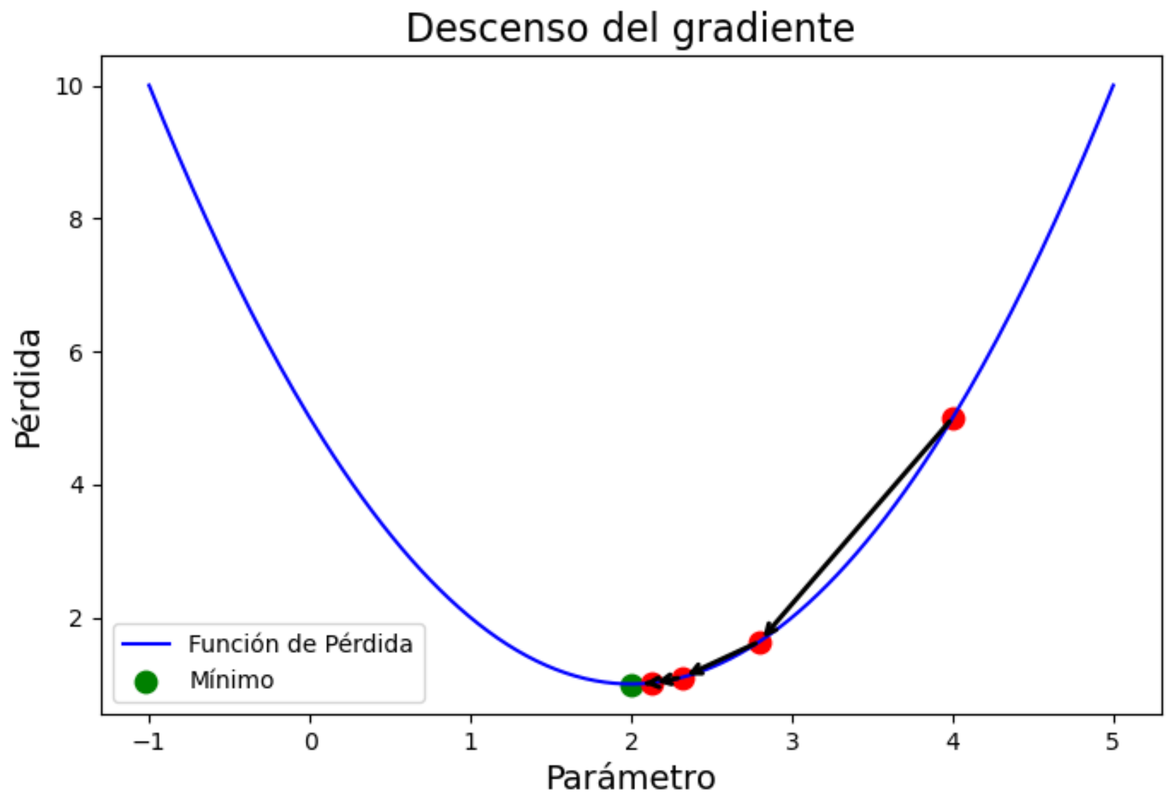
$$\mathscr{L}(\theta_{ini}) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}(x_i;\theta_{ini})\right)^2$$

$$\mathscr{L}(\theta_{min}) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}(x_i;\theta_{min})\right)^2$$

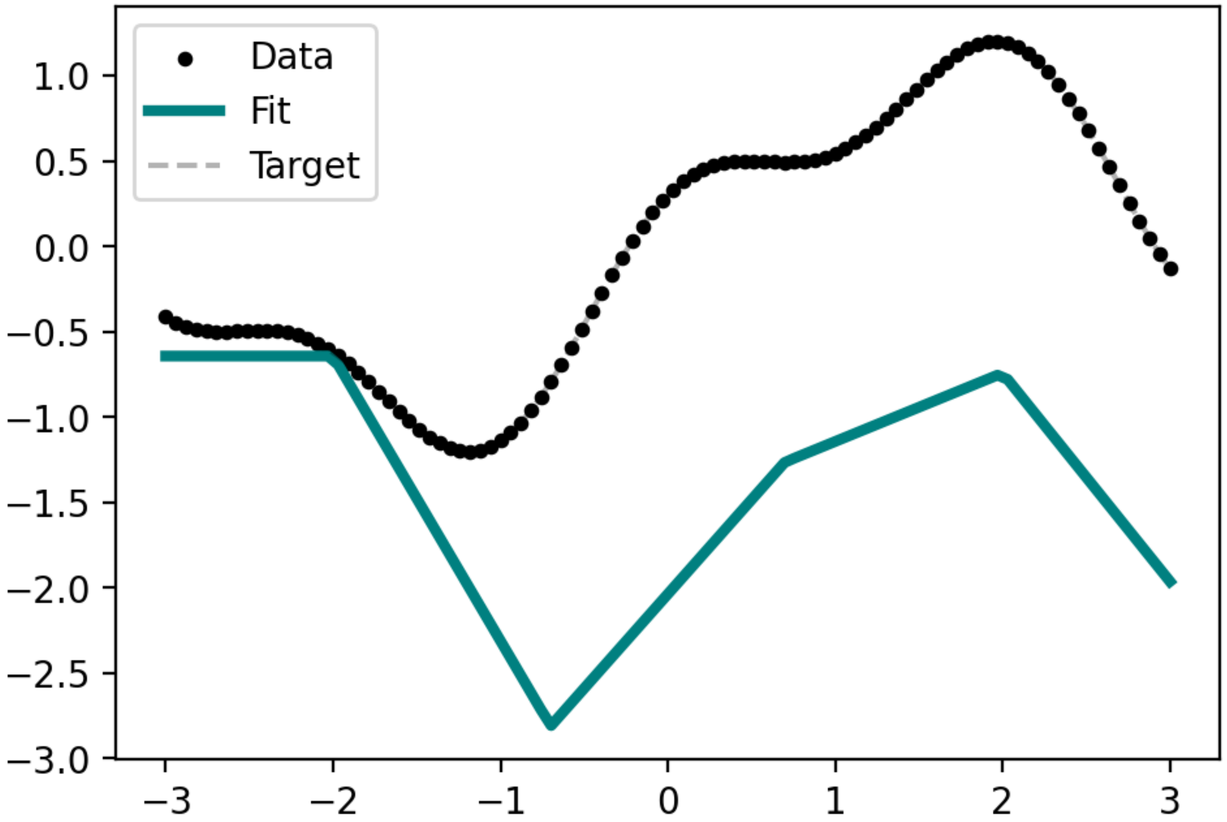
$$\nabla_\theta \mathscr{L}(\theta) = \frac{\partial \mathscr{L}(\theta)}{\partial \theta}$$

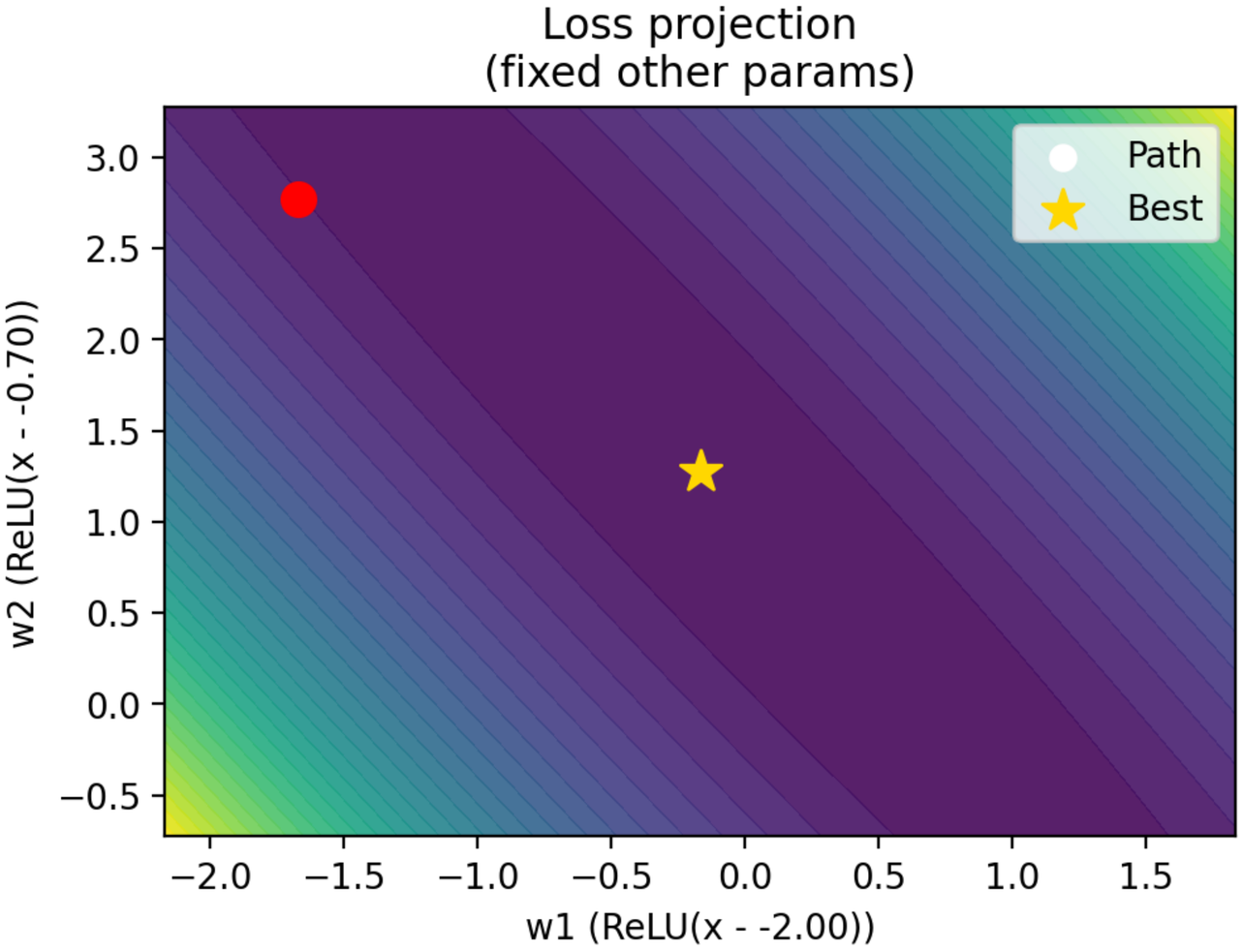
$$\theta_{new} = \theta_{old} - \eta \nabla_\theta \mathscr{L}(\theta_{old})$$

$\theta_{ini} = = \theta_{new}$

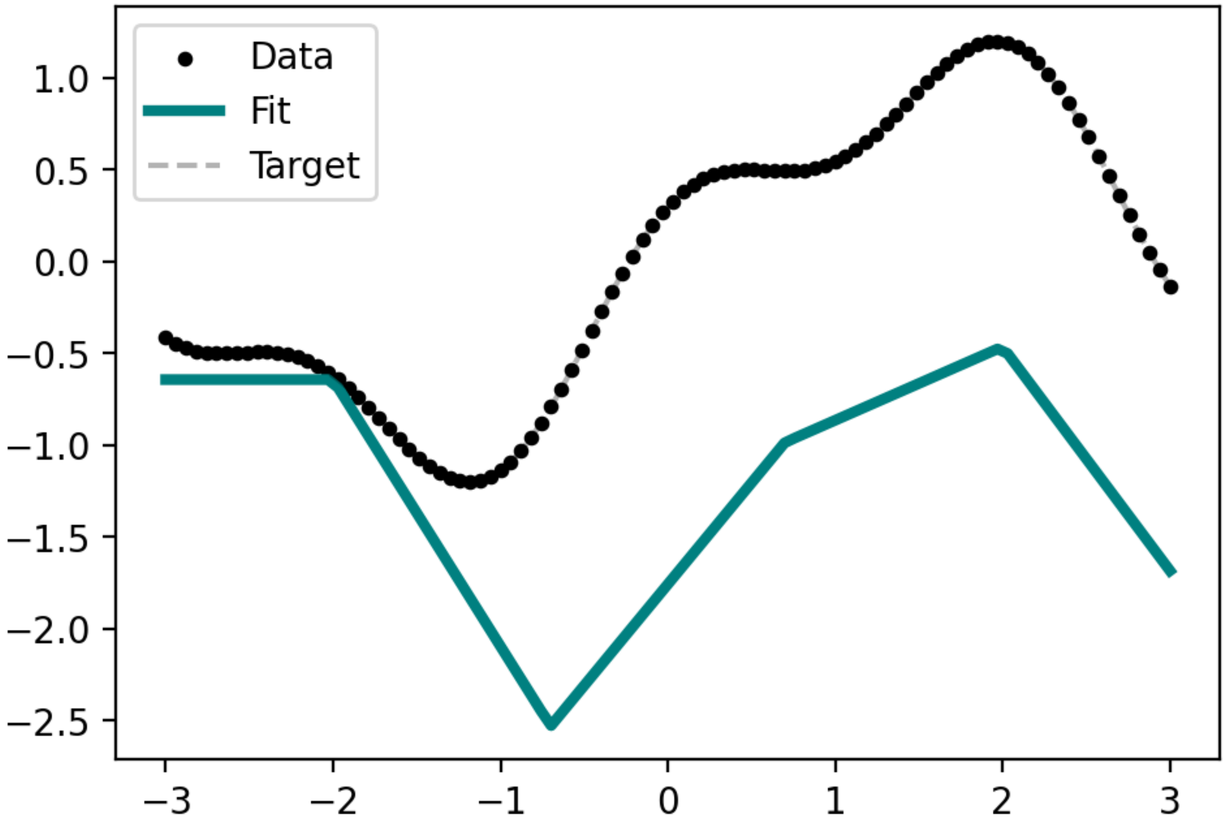


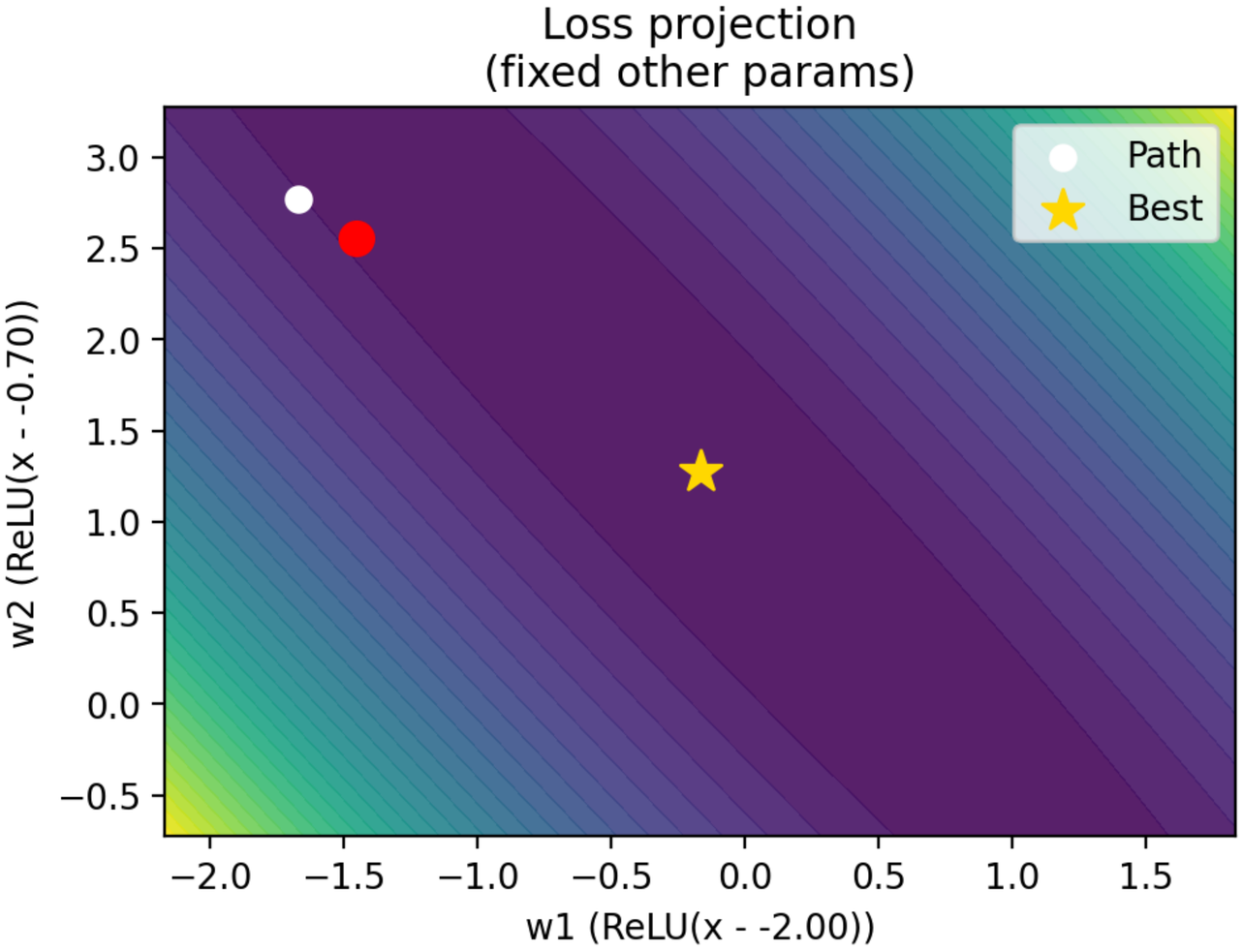
Redes Neuronales: Entrenamiento



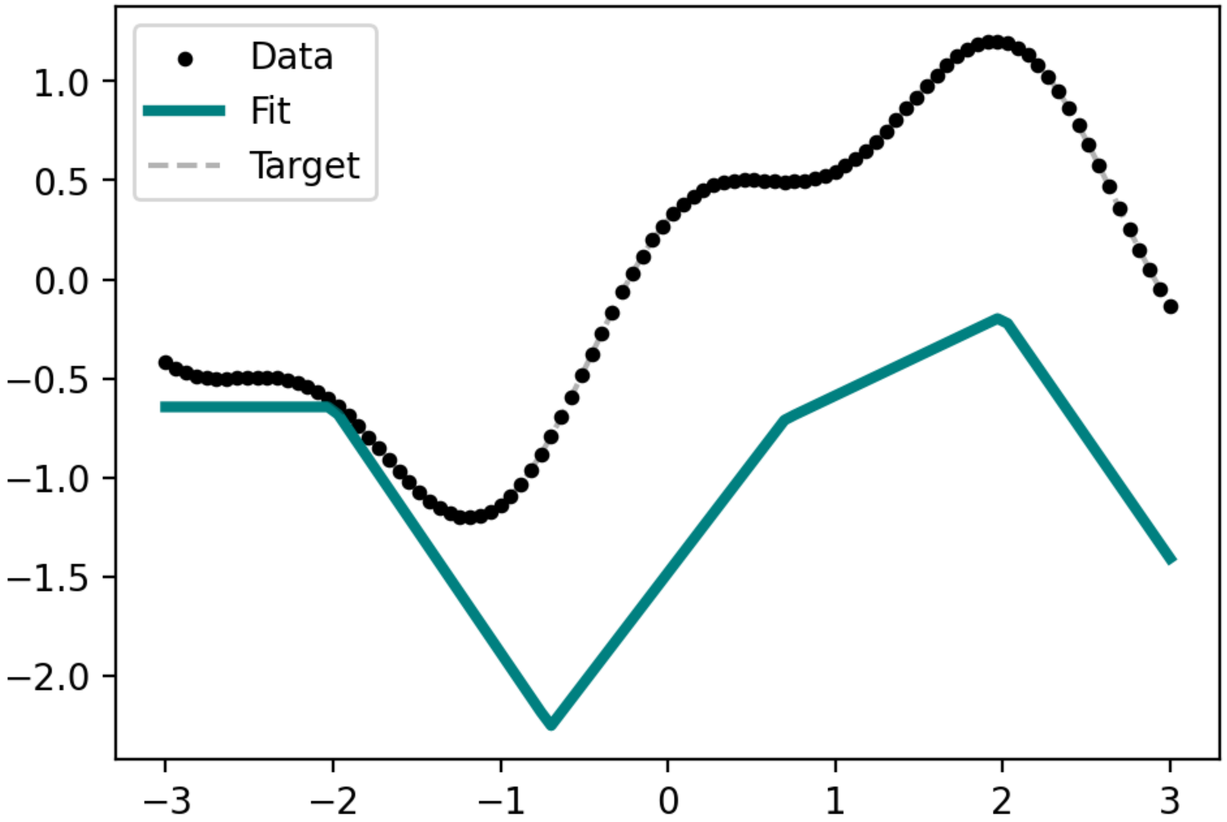


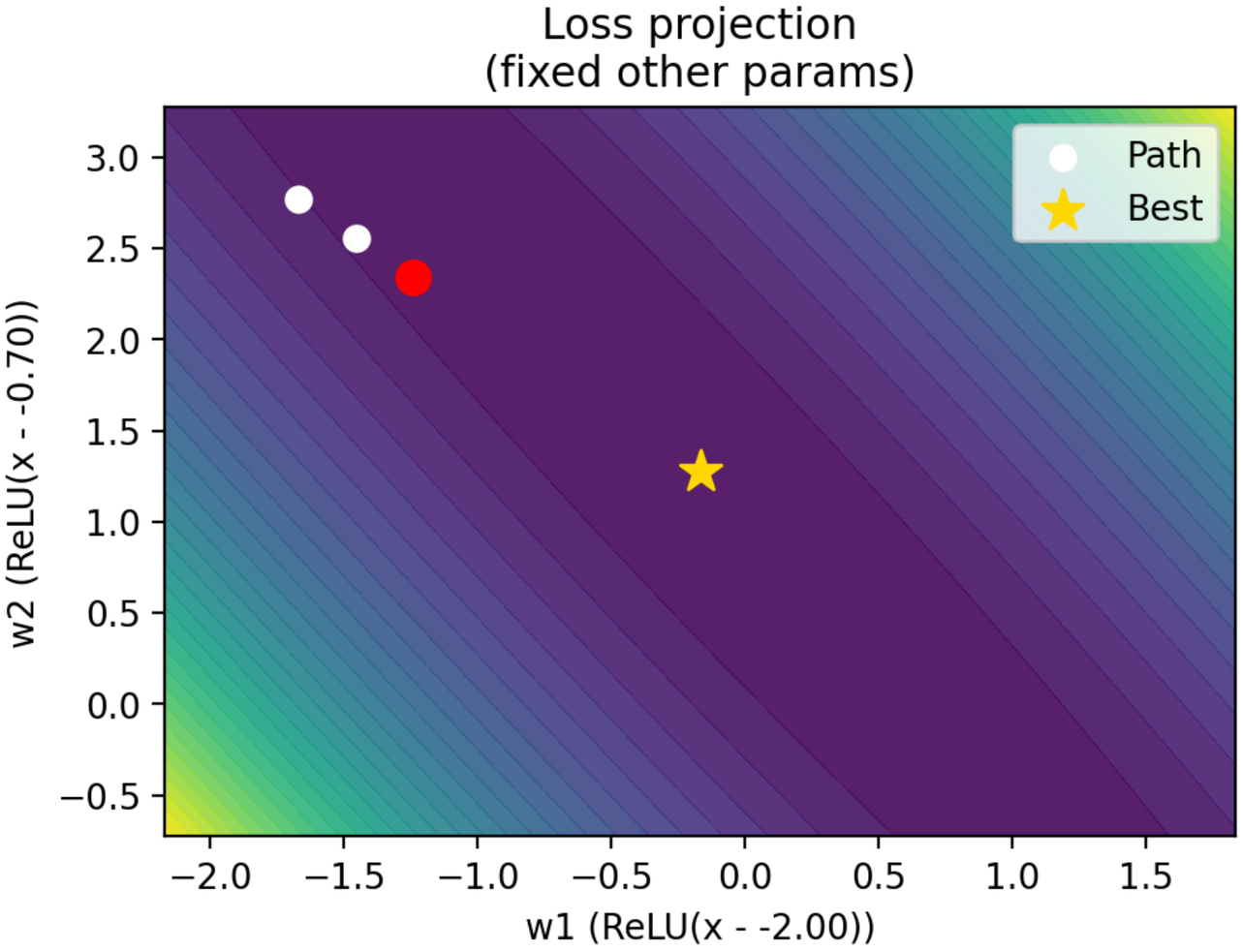
Redes Neuronales: Entrenamiento



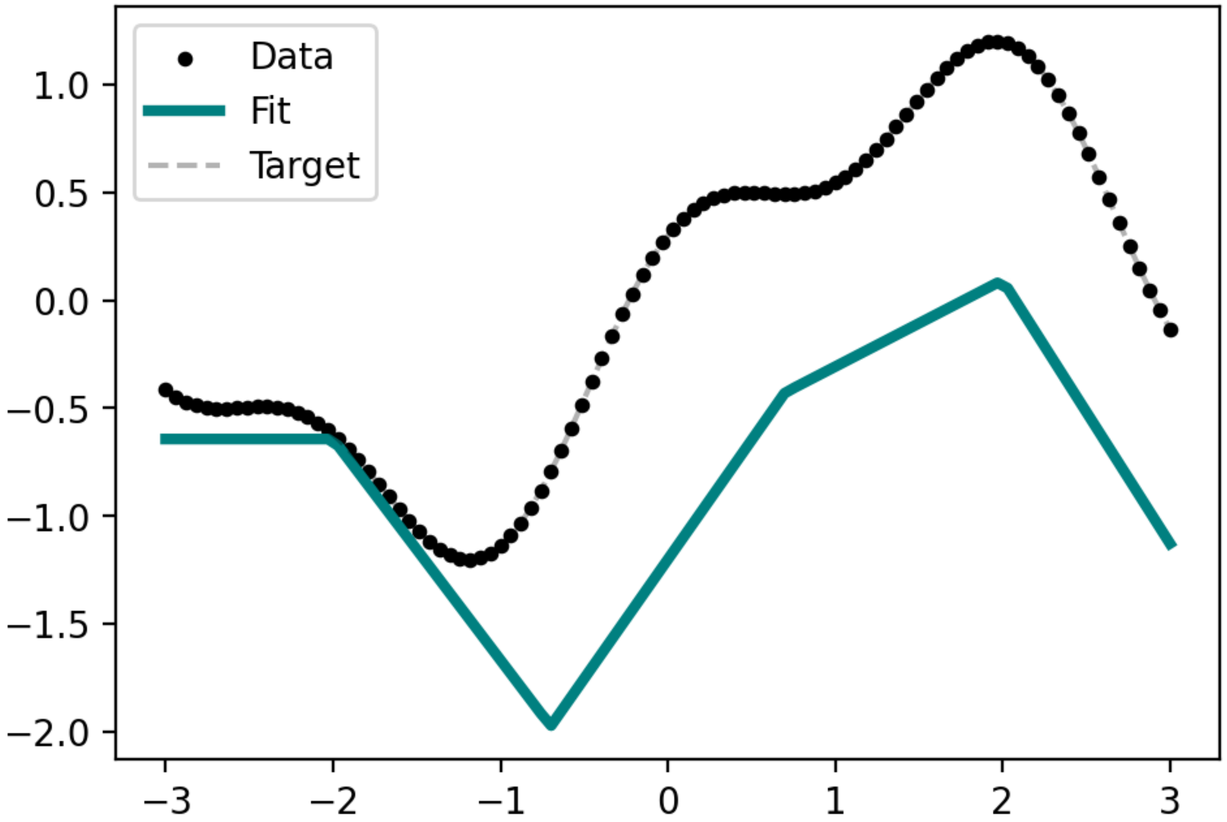


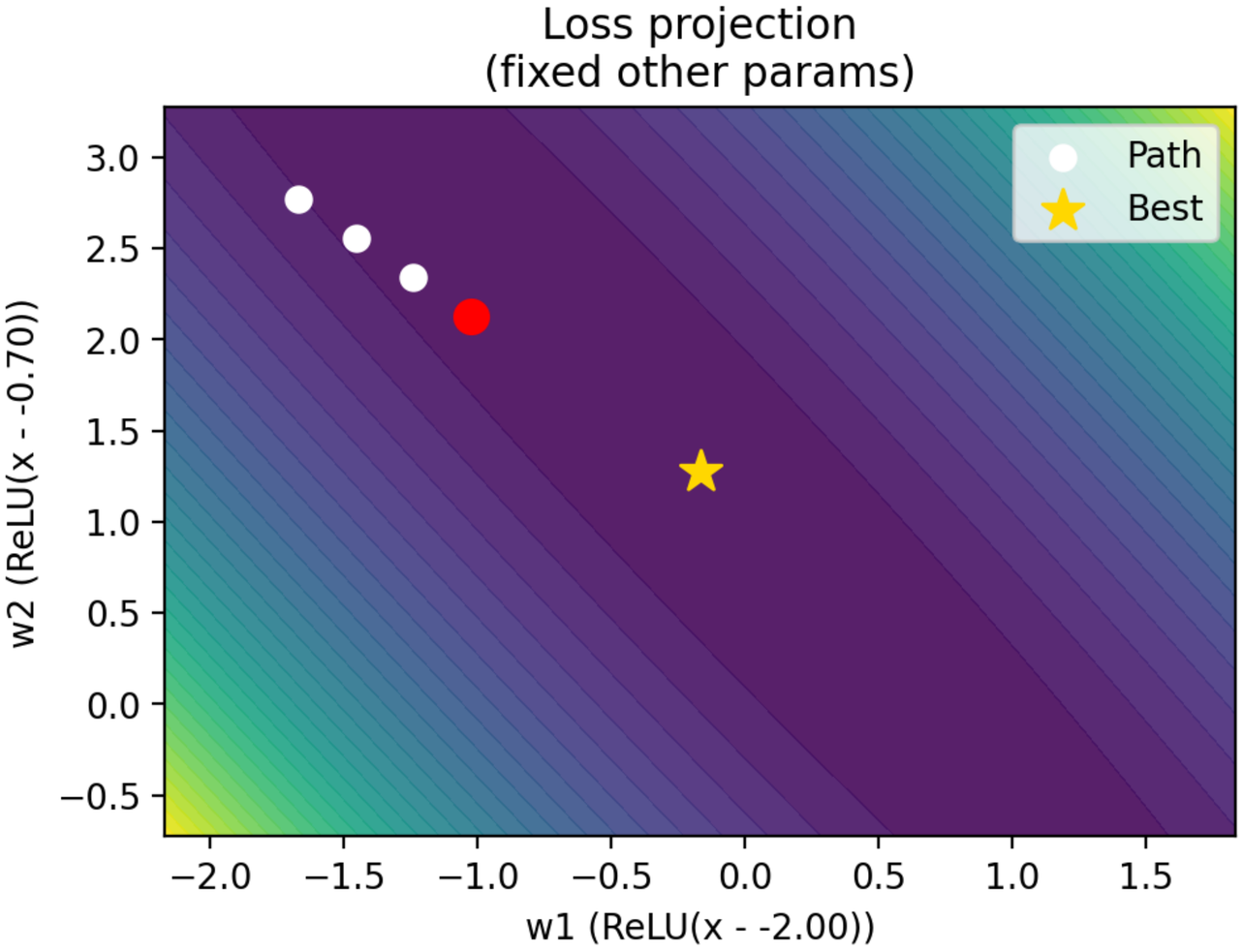
Redes Neuronales: Entrenamiento



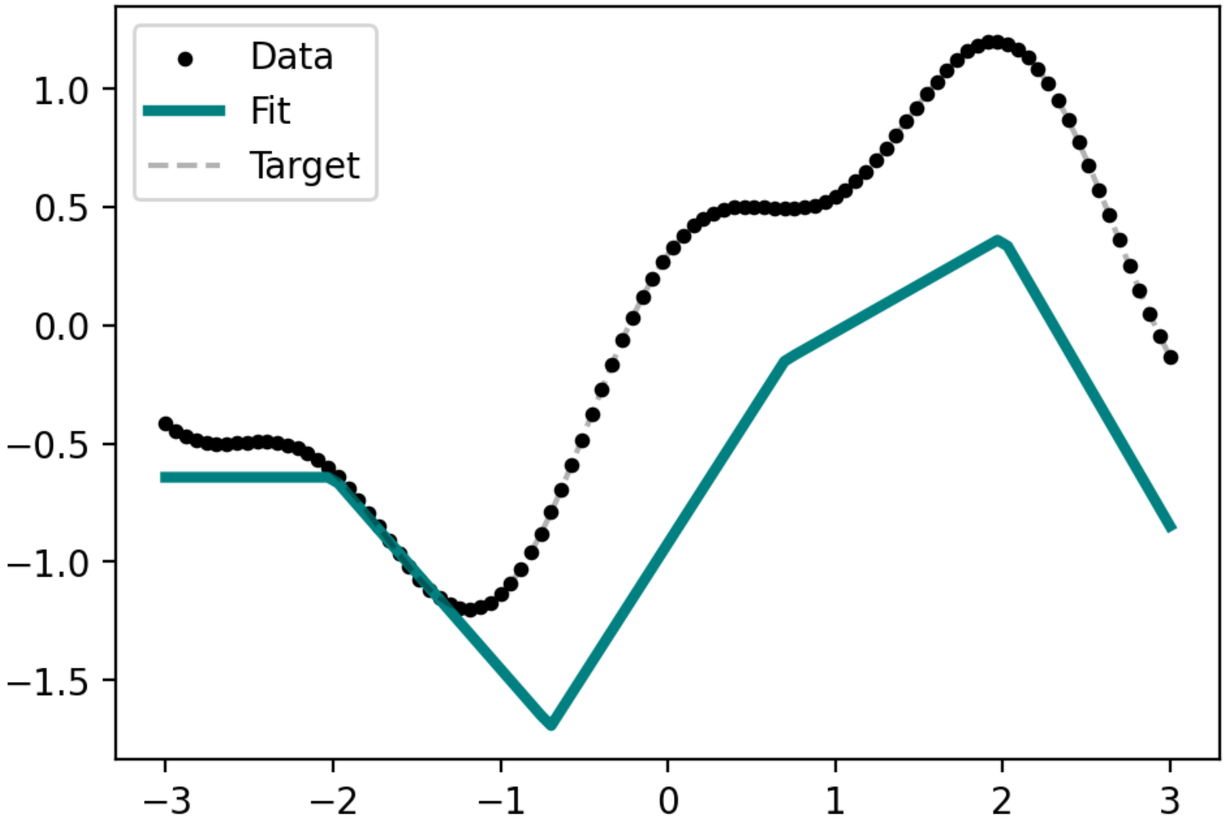


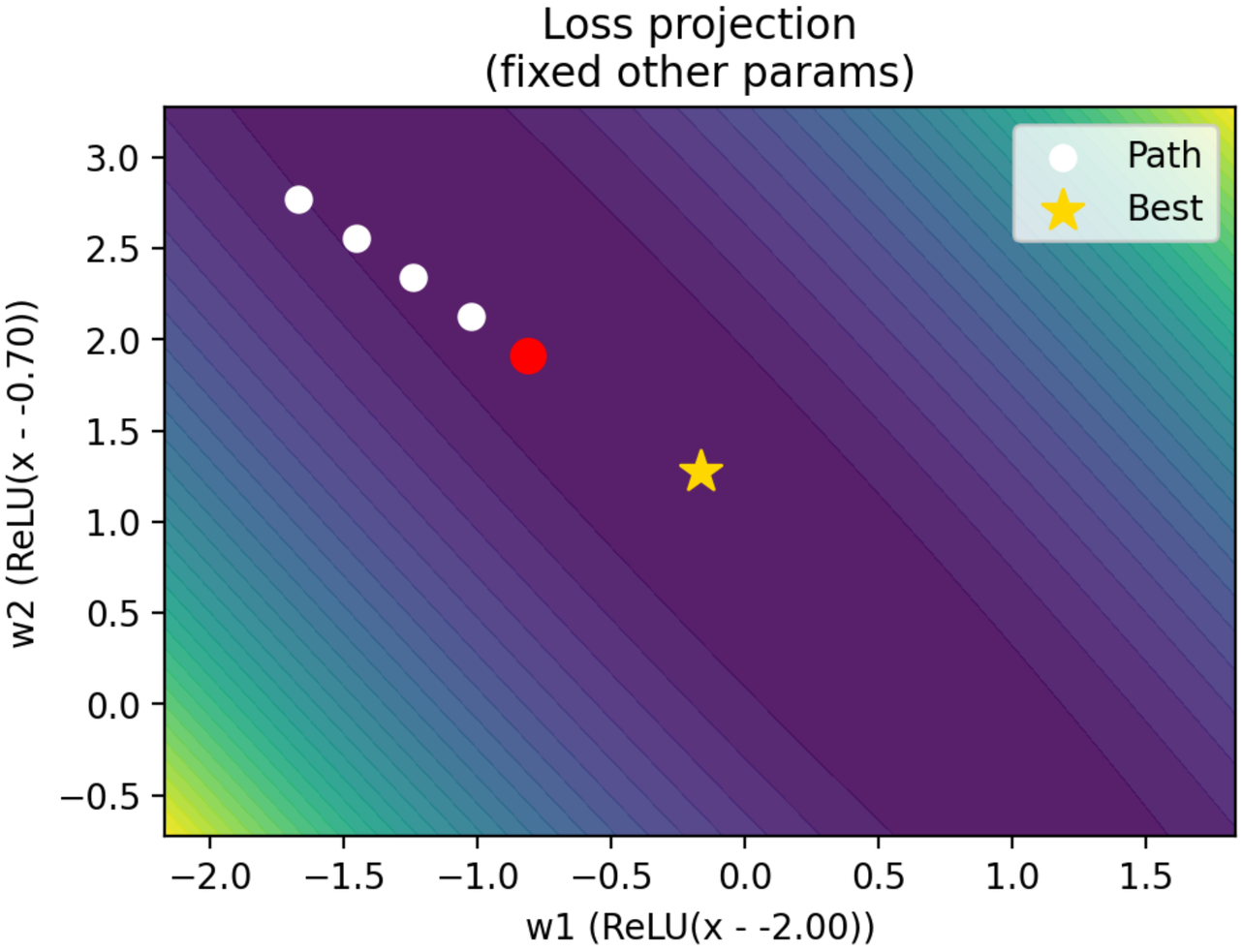
Redes Neuronales: Entrenamiento



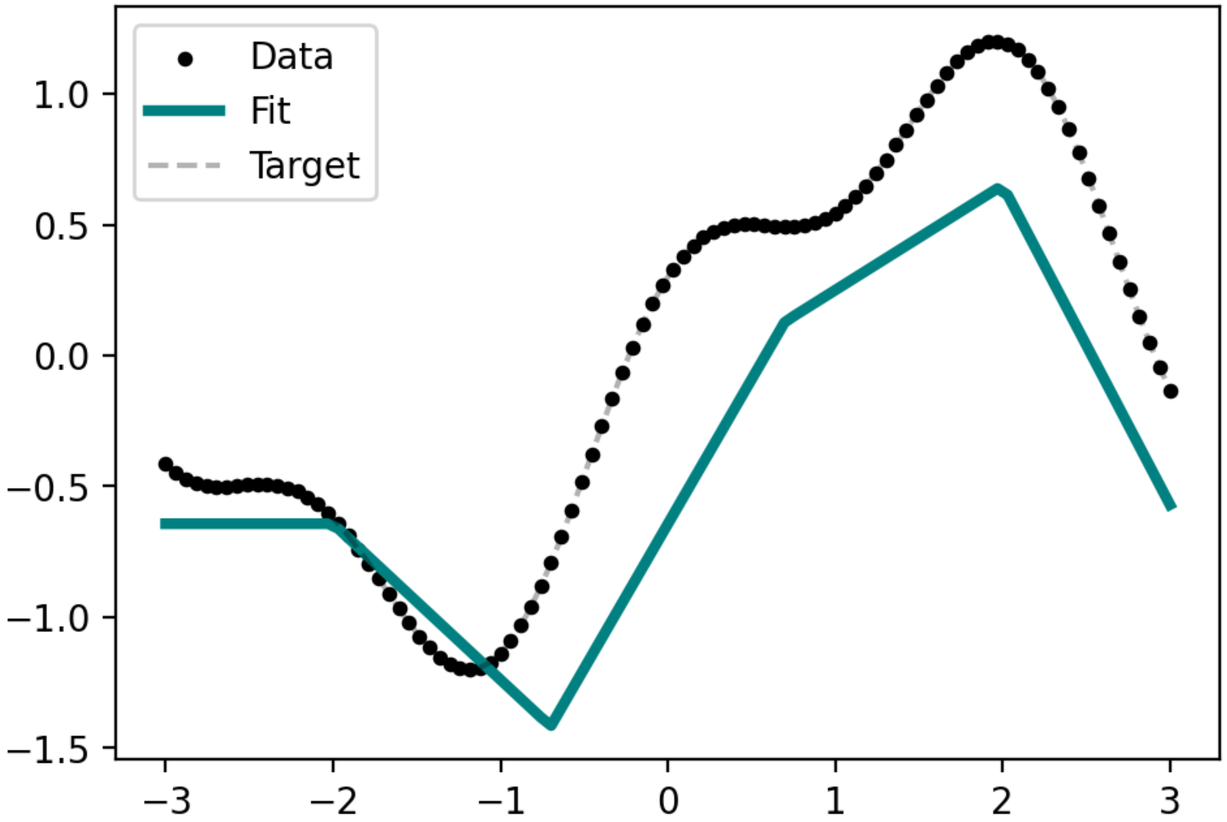


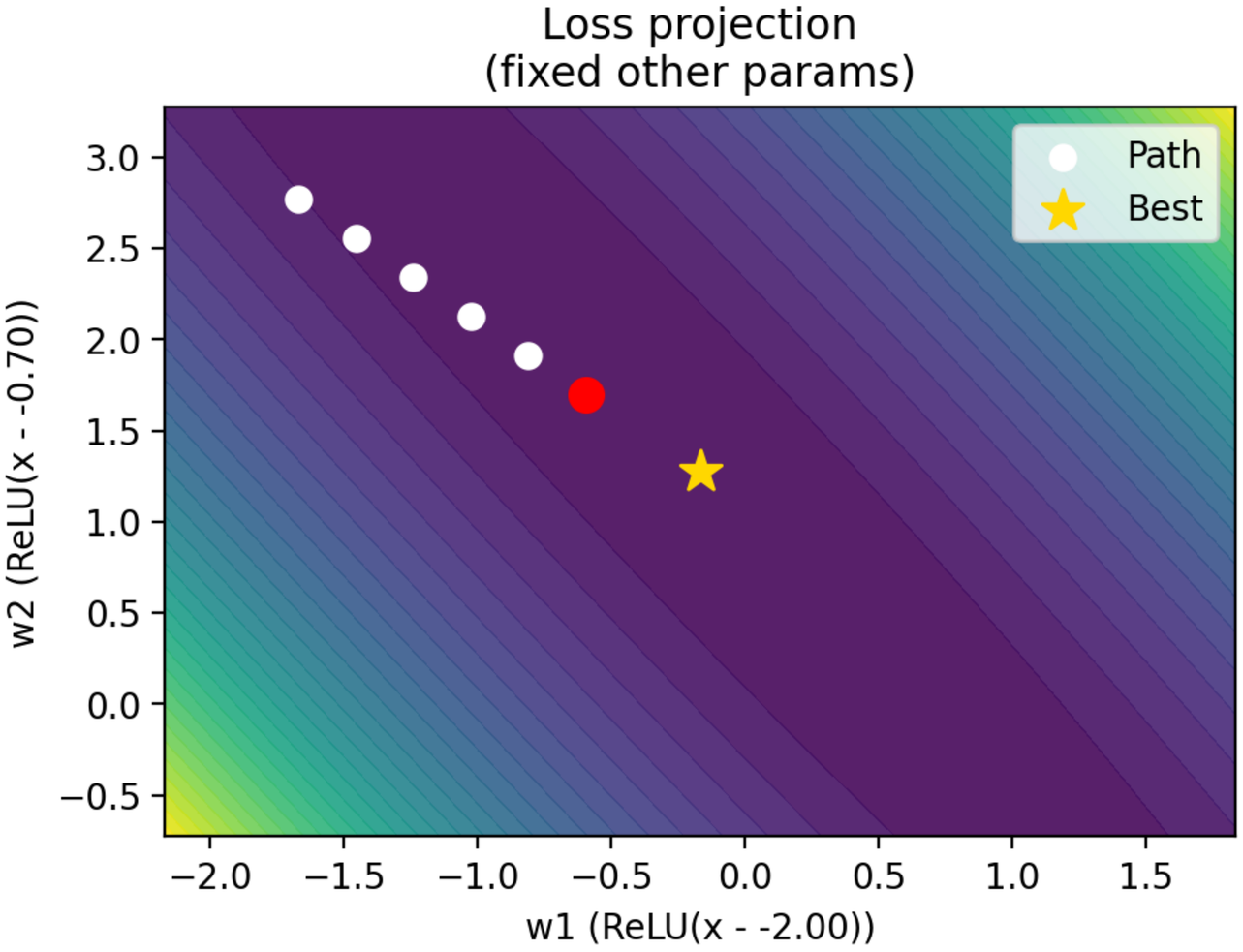
Redes Neuronales: Entrenamiento



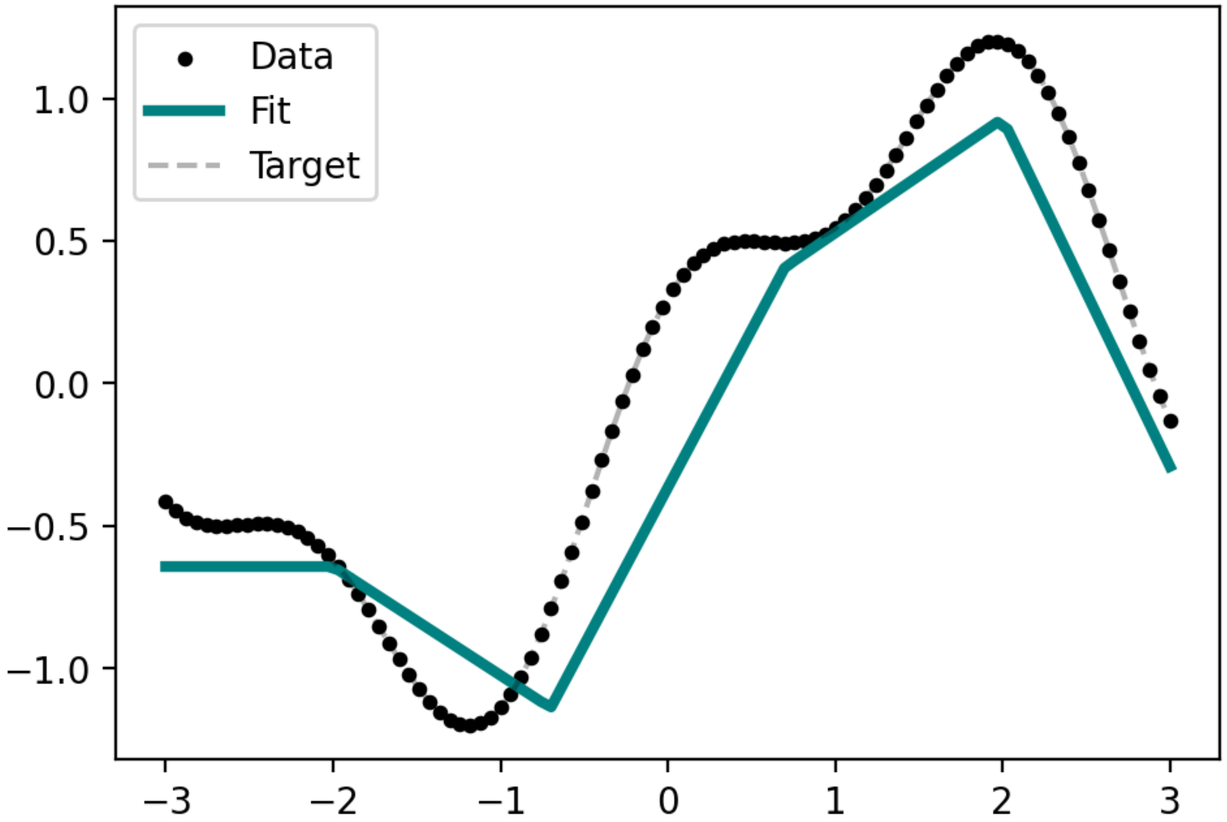


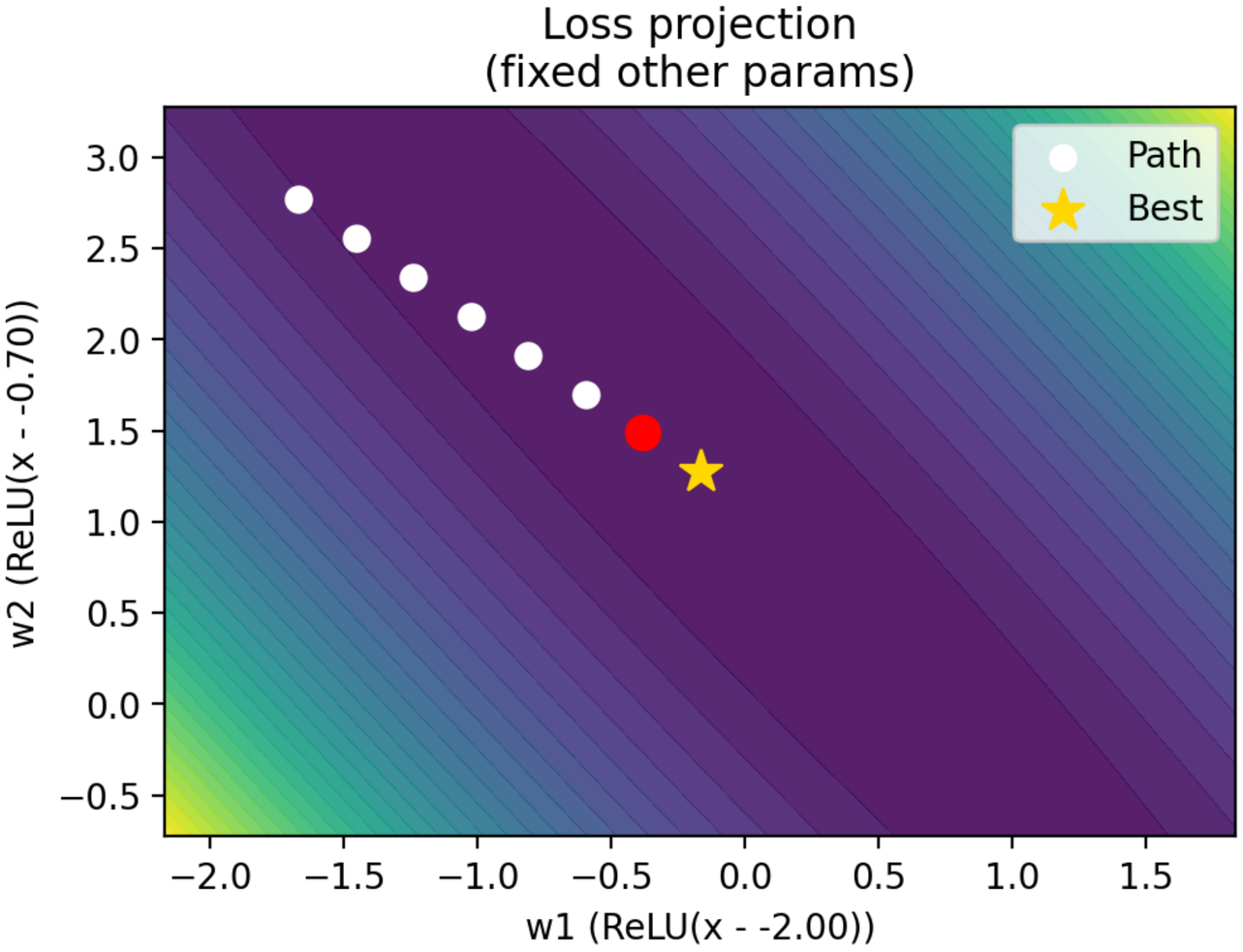
Redes Neuronales: Entrenamiento



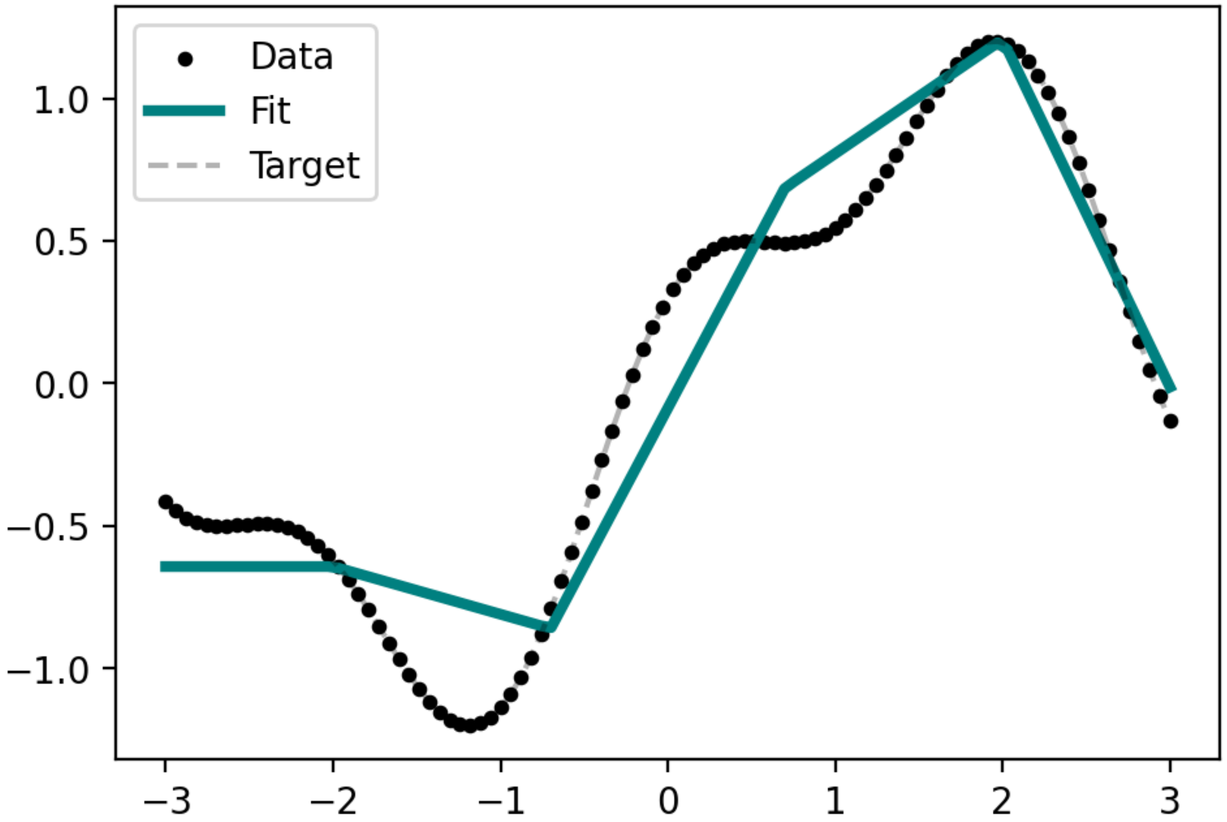


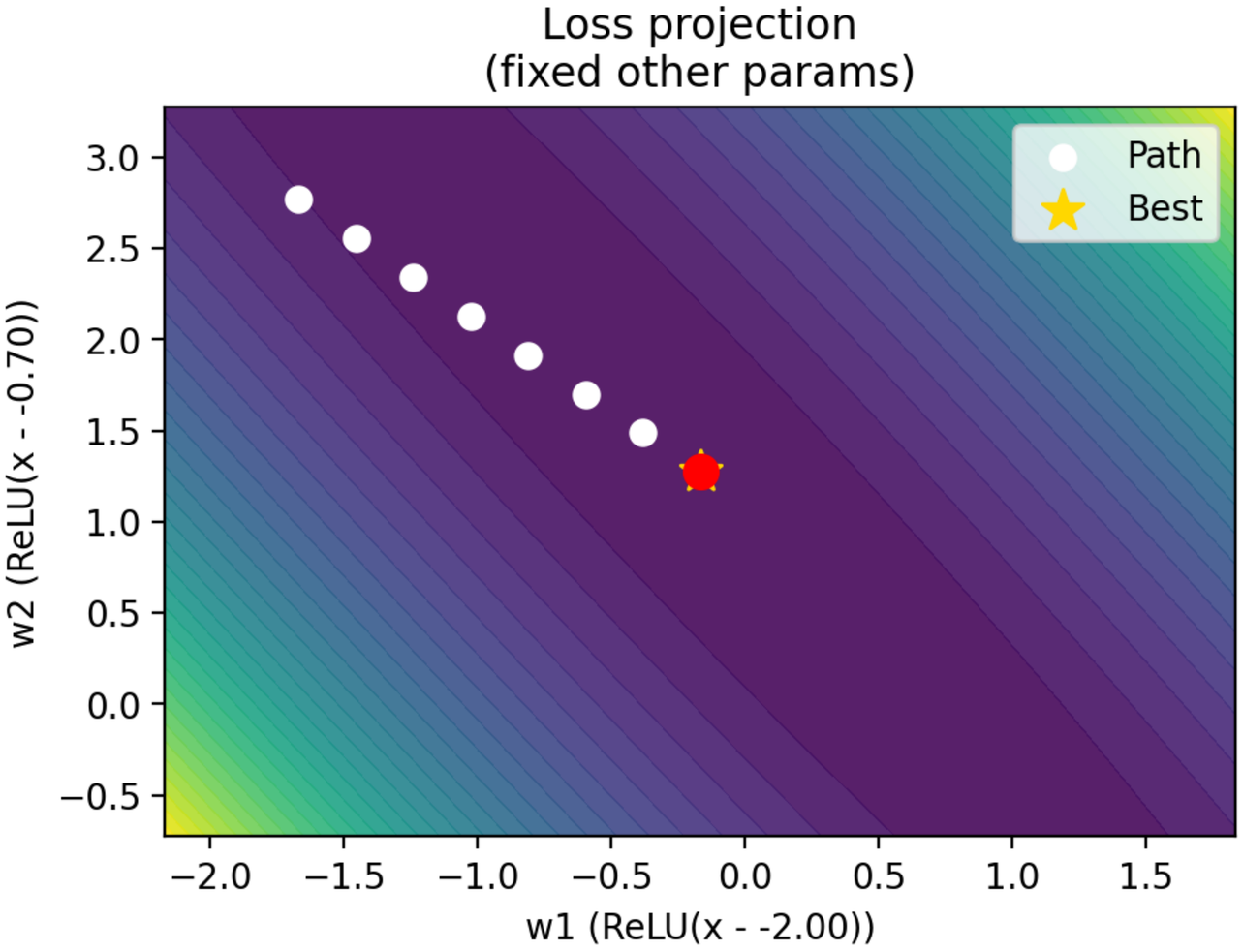
Redes Neuronales: Entrenamiento





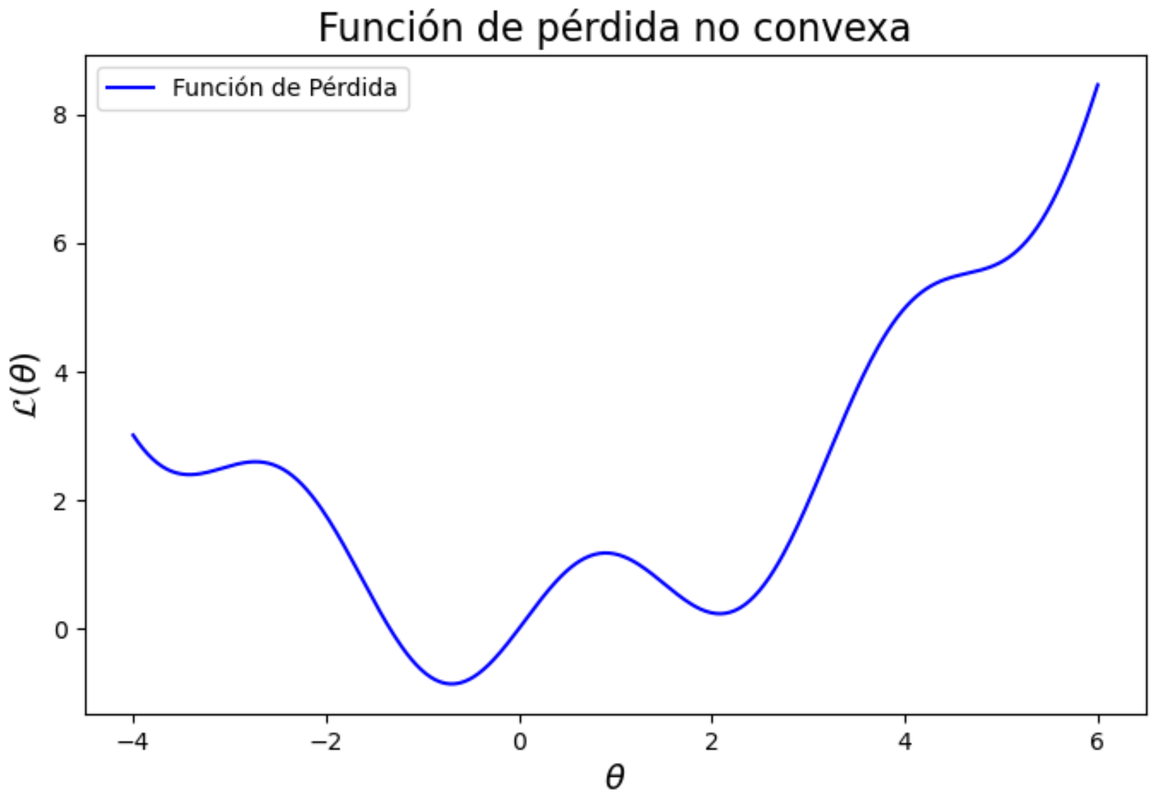
Redes Neuronales: Entrenamiento





Redes Neuronales: Descenso del Gradiente

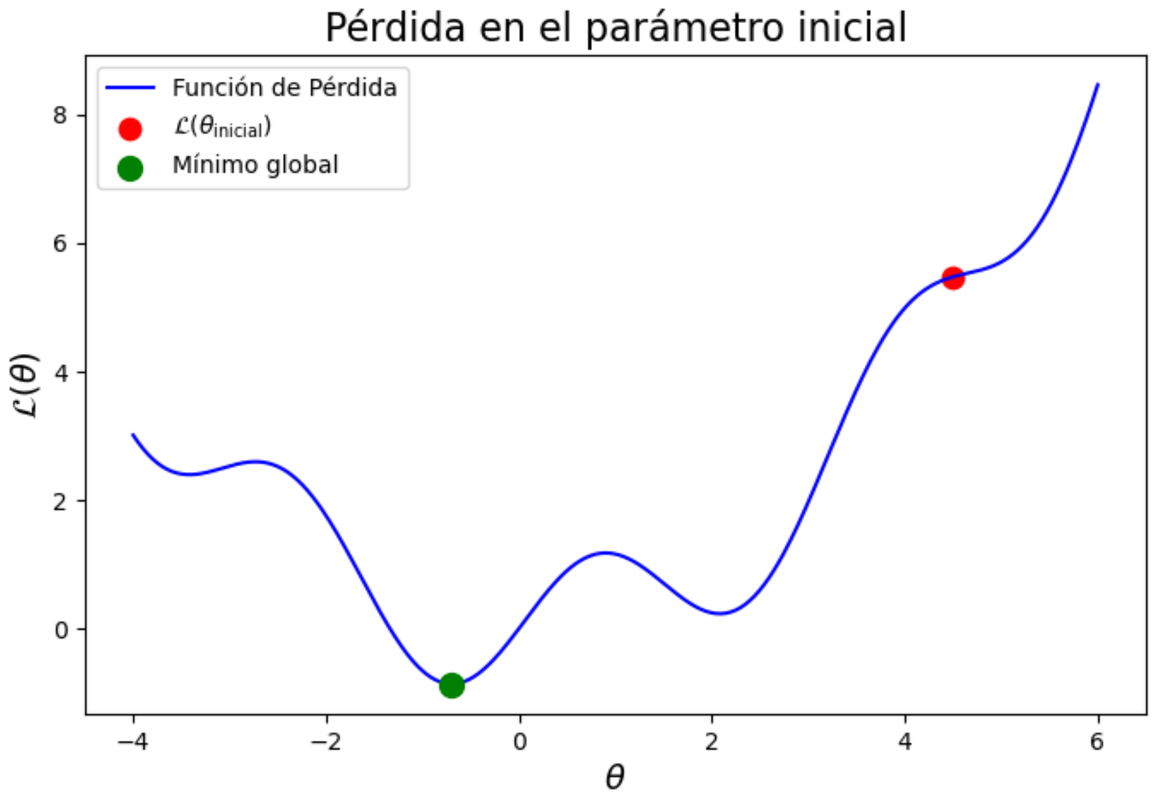
$$\mathscr{L}(\theta) = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}(x_i; \theta))^2$$



Redes Neuronales: Descenso del Gradiente

$$\mathscr{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}(x_i;\theta))^2$$

$$\mathscr{L}(\theta_{ini}) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}(x_i;\theta_{ini})\right)^2$$



Redes Neuronales: Función de Pérdida no Convexa

$$\mathscr{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}(x_i;\theta))^2$$

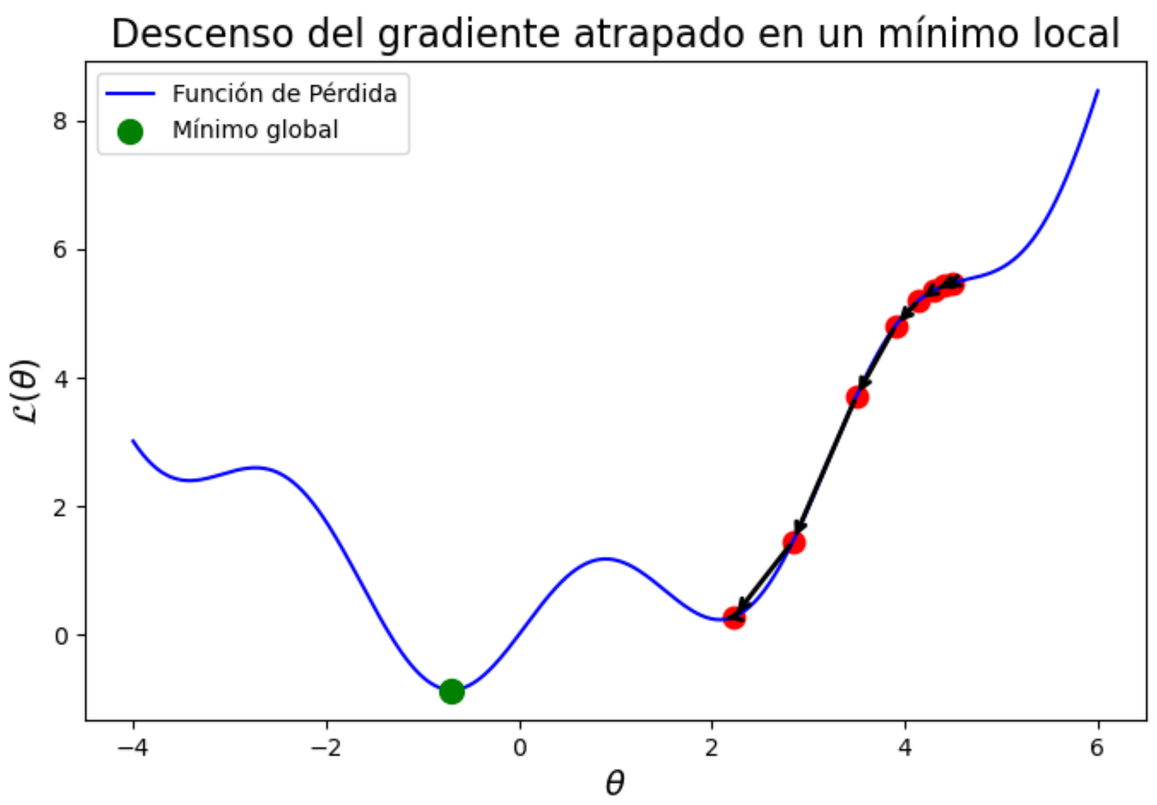
$$\mathscr{L}(\theta_{ini}) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}(x_i;\theta_{ini})\right)^2$$

$$\mathscr{L}(\theta_{min}) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}(x_i;\theta_{min})\right)^2$$

$$\nabla_\theta \mathscr{L}(\theta) = \frac{\partial \mathscr{L}(\theta)}{\partial \theta}$$

$$\theta_{new} = \theta_{old} - \eta \nabla_\theta \mathscr{L}(\theta_{old})$$

$\theta_{ini} = = \theta_{new}$



Redes Neuronales: Estocasticidad

$$\mathscr{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}(x_i;\theta))^2$$

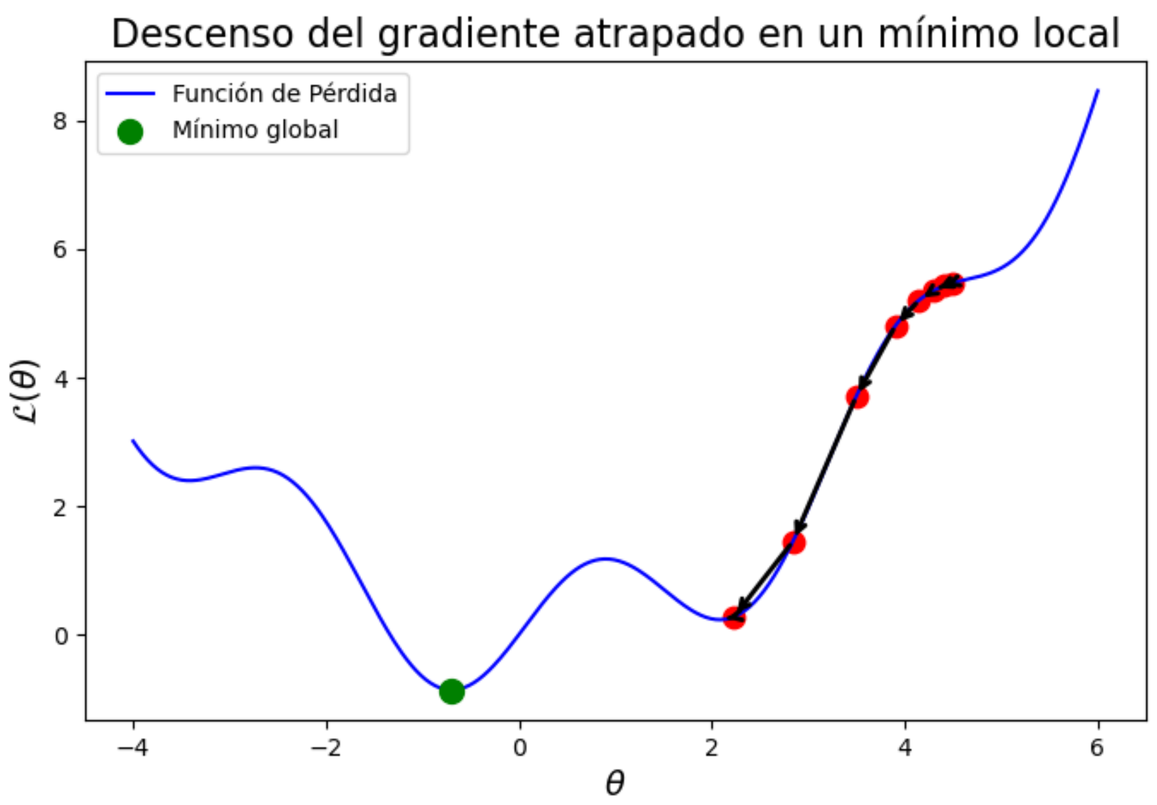
$$\mathscr{L}(\theta_{ini}) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}(x_i;\theta_{ini})\right)^2$$

$$\mathscr{L}(\theta_{min}) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}(x_i;\theta_{min})\right)^2$$

$$\nabla_\theta \mathscr{L}(\theta) = \frac{\partial \mathscr{L}(\theta)}{\partial \theta}$$

$$\theta_{new} = \theta_{old} - \eta \nabla_\theta \mathscr{L}(\theta_{old})$$

$\theta_{ini} = = \theta_{new}$



Redes Neuronales: Estocasticidad

$$\mathscr{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}(x_i;\theta))^2$$

$$\mathscr{L}(\theta_{ini}) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}(x_i;\theta_{ini})\right)^2$$

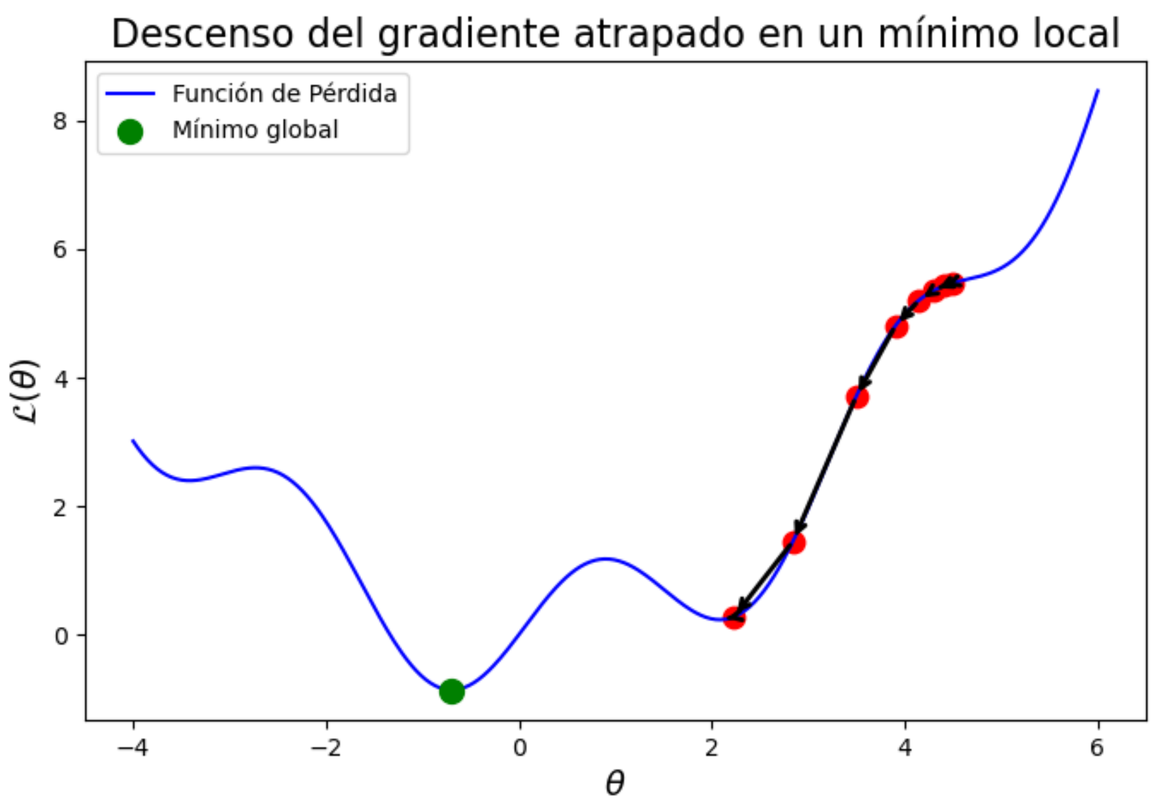
$$\mathscr{L}(\theta_{min}) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}(x_i;\theta_{min})\right)^2$$

$\theta_{ini} = = \theta_{new}$

$$\nabla_\theta \mathscr{L}(\theta) = \frac{\partial \mathscr{L}(\theta)}{\partial \theta}$$

$$\theta_{new} = \theta_{old} - \eta \nabla_\theta \mathscr{L}(\theta_{old})$$

$$\theta_{new} = \theta_{old} - \eta \nabla_\theta \mathscr{L}(\theta_{old}) + \mathcal{N}(0,\sigma^2)$$



Redes Neuronales: Estocasticidad

$$\mathscr{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}(x_i;\theta))^2$$

$$\mathscr{L}(\theta_{ini}) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}(x_i;\theta_{ini})\right)^2$$

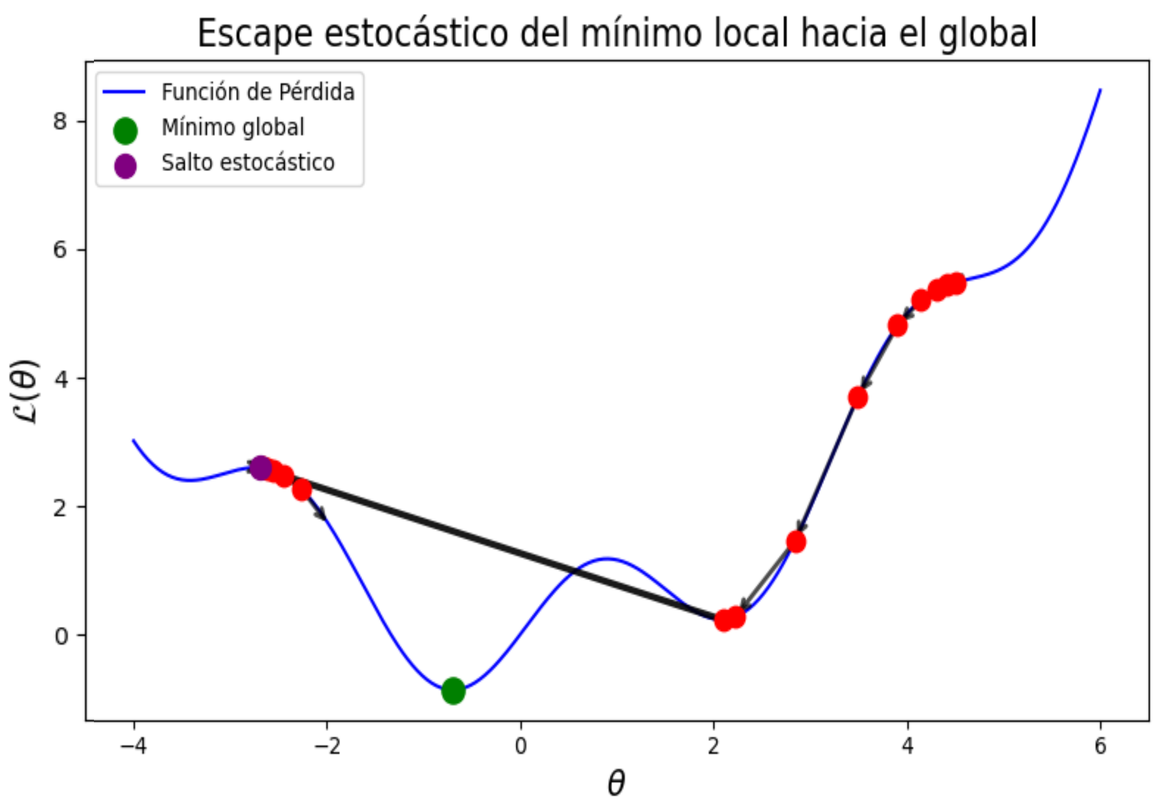
$$\mathscr{L}(\theta_{min}) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}(x_i;\theta_{min})\right)^2$$

$\theta_{ini} = = \theta_{new}$

$$\nabla_\theta \mathscr{L}(\theta) = \frac{\partial \mathscr{L}(\theta)}{\partial \theta}$$

$$\theta_{new} = \theta_{old} - \eta \nabla_\theta \mathscr{L}(\theta_{old})$$

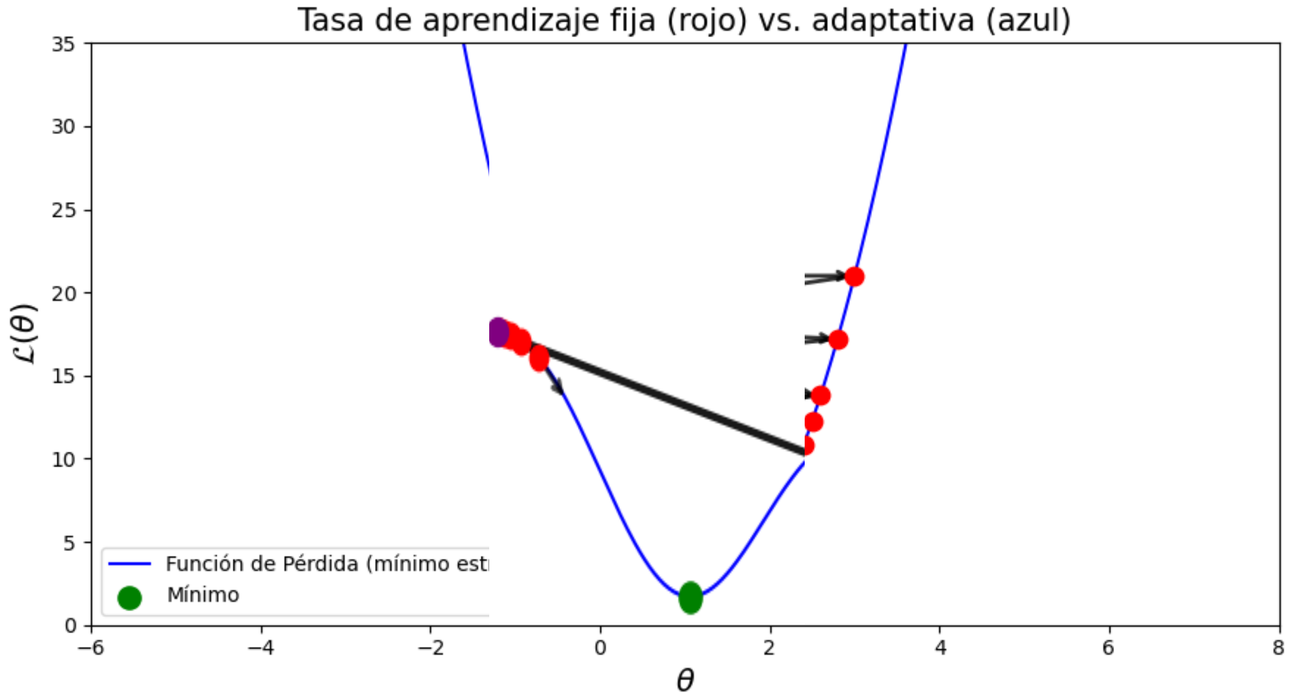
$$\theta_{new} = \theta_{old} - \eta \nabla_\theta \mathscr{L}(\theta_{old}) + \mathcal{N}(0,\sigma^2)$$



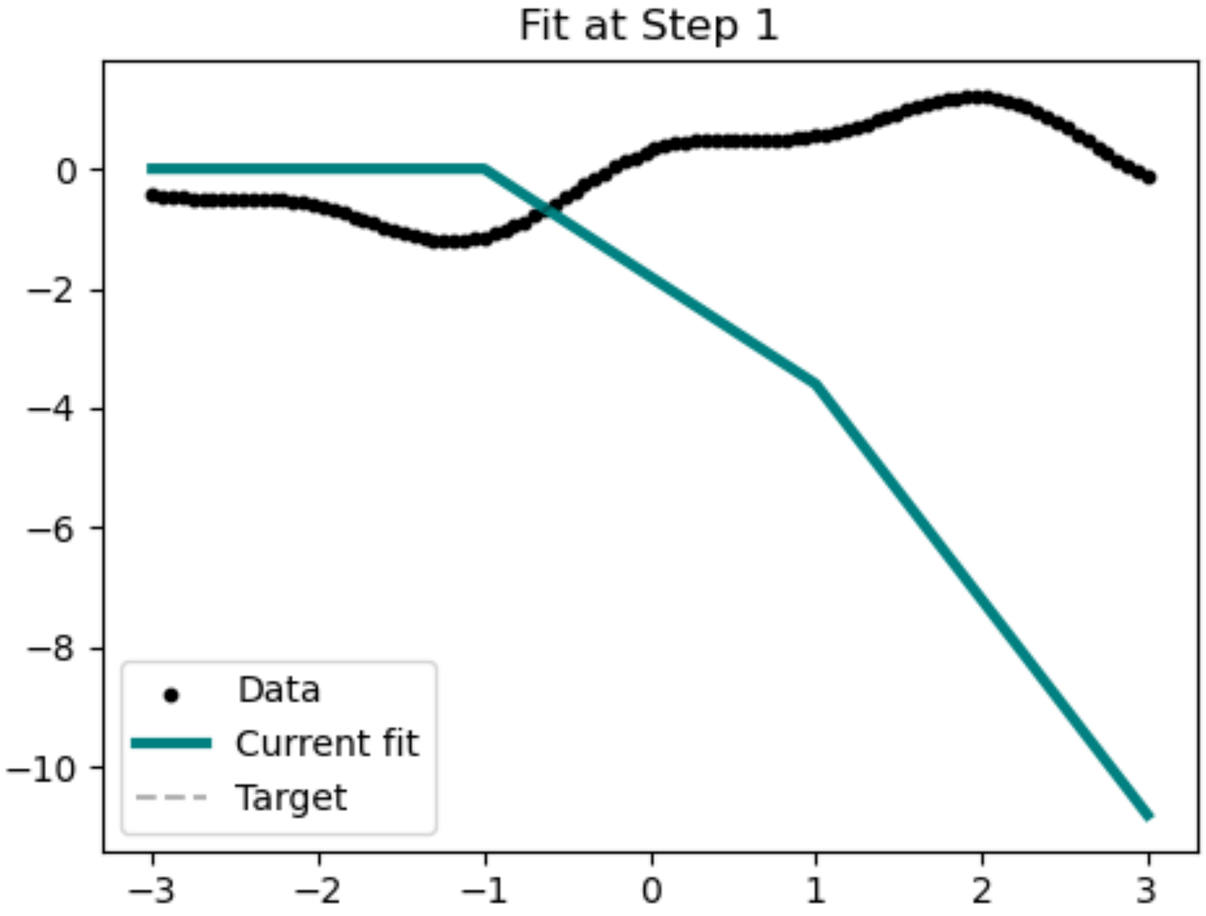
Redes Neuronales: Aprendizaje Adaptativo

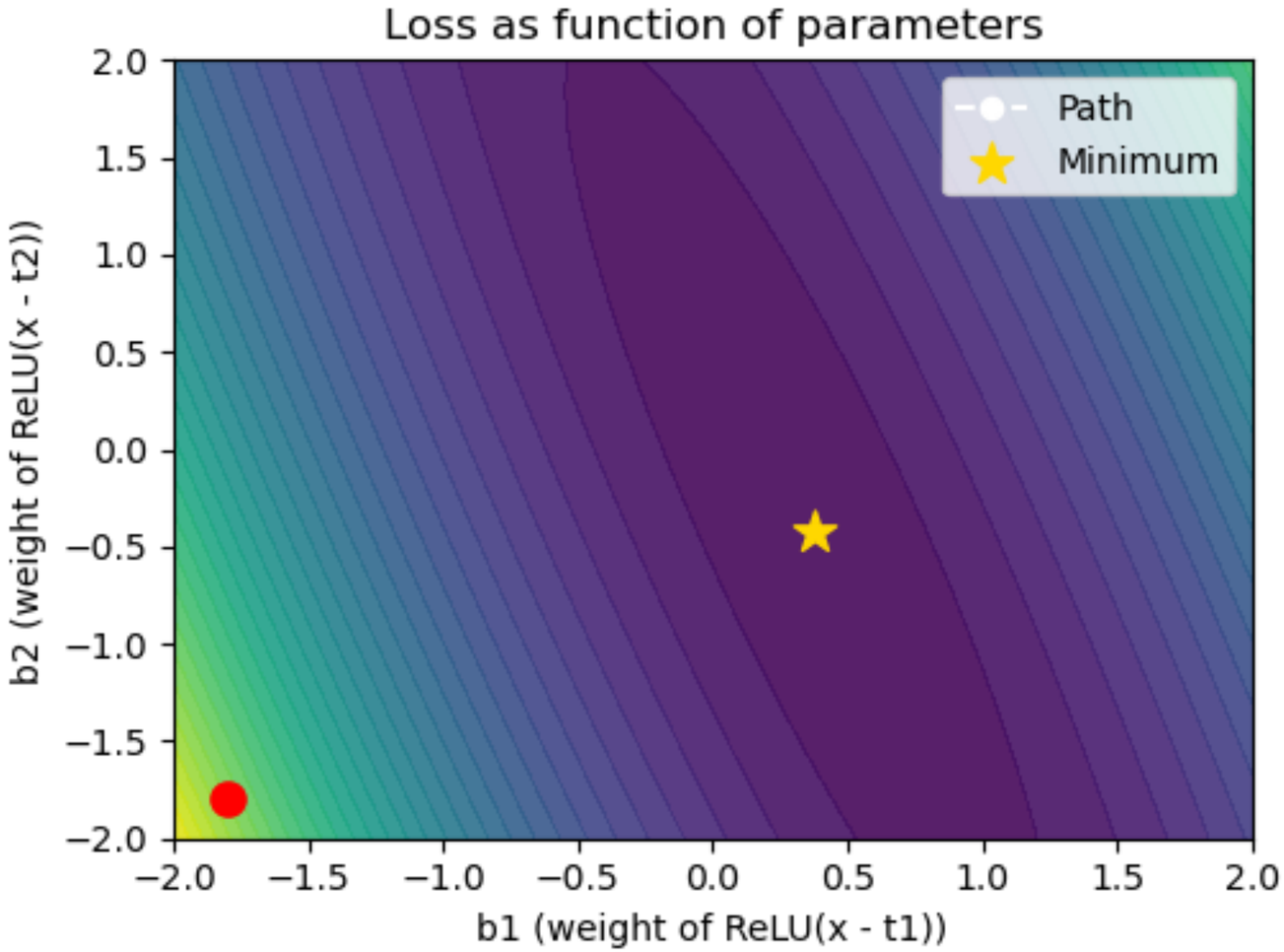
$$\theta_{t+1} = \theta_t - \eta \nabla_\theta \mathcal{L}(\theta_t)$$

η : tasa de aprendizaje fija
(puede causar oscilaciones si es muy grande)

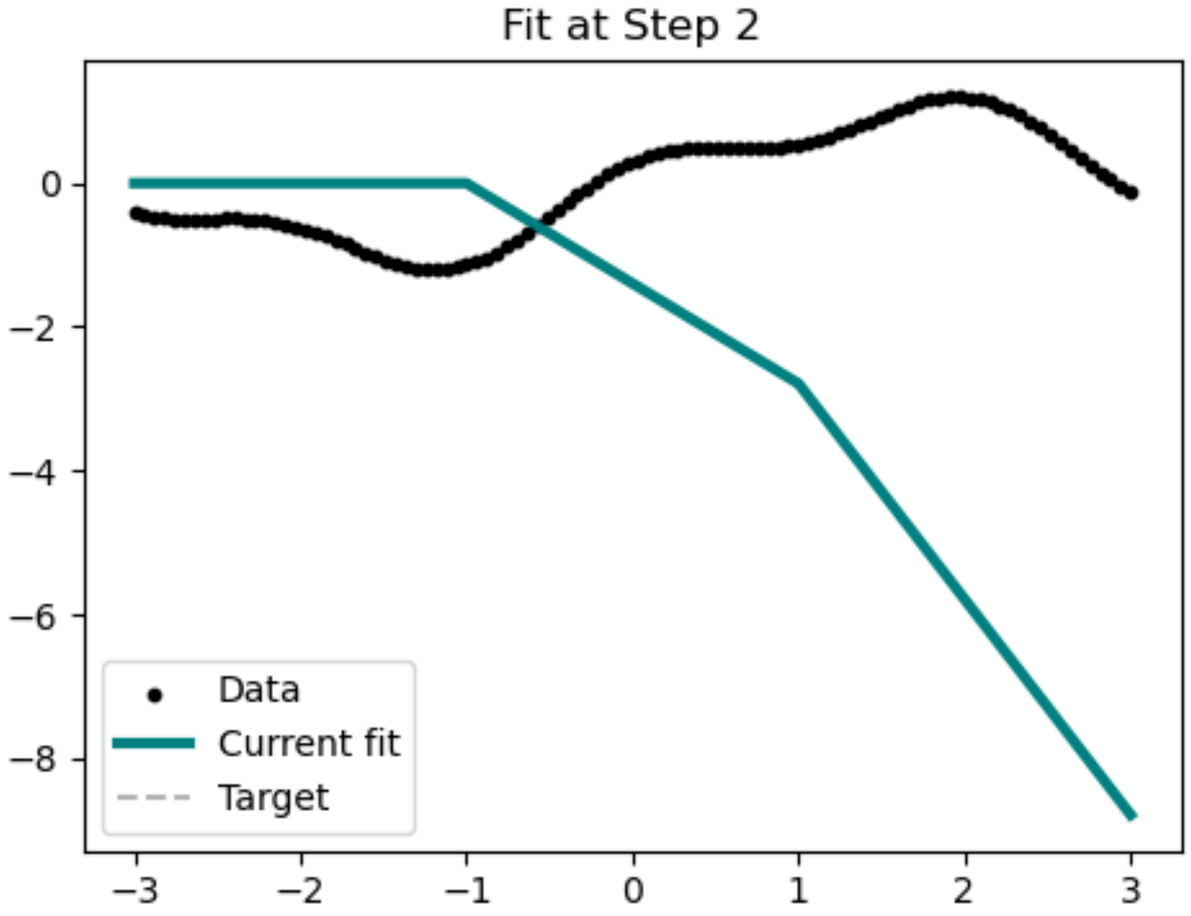


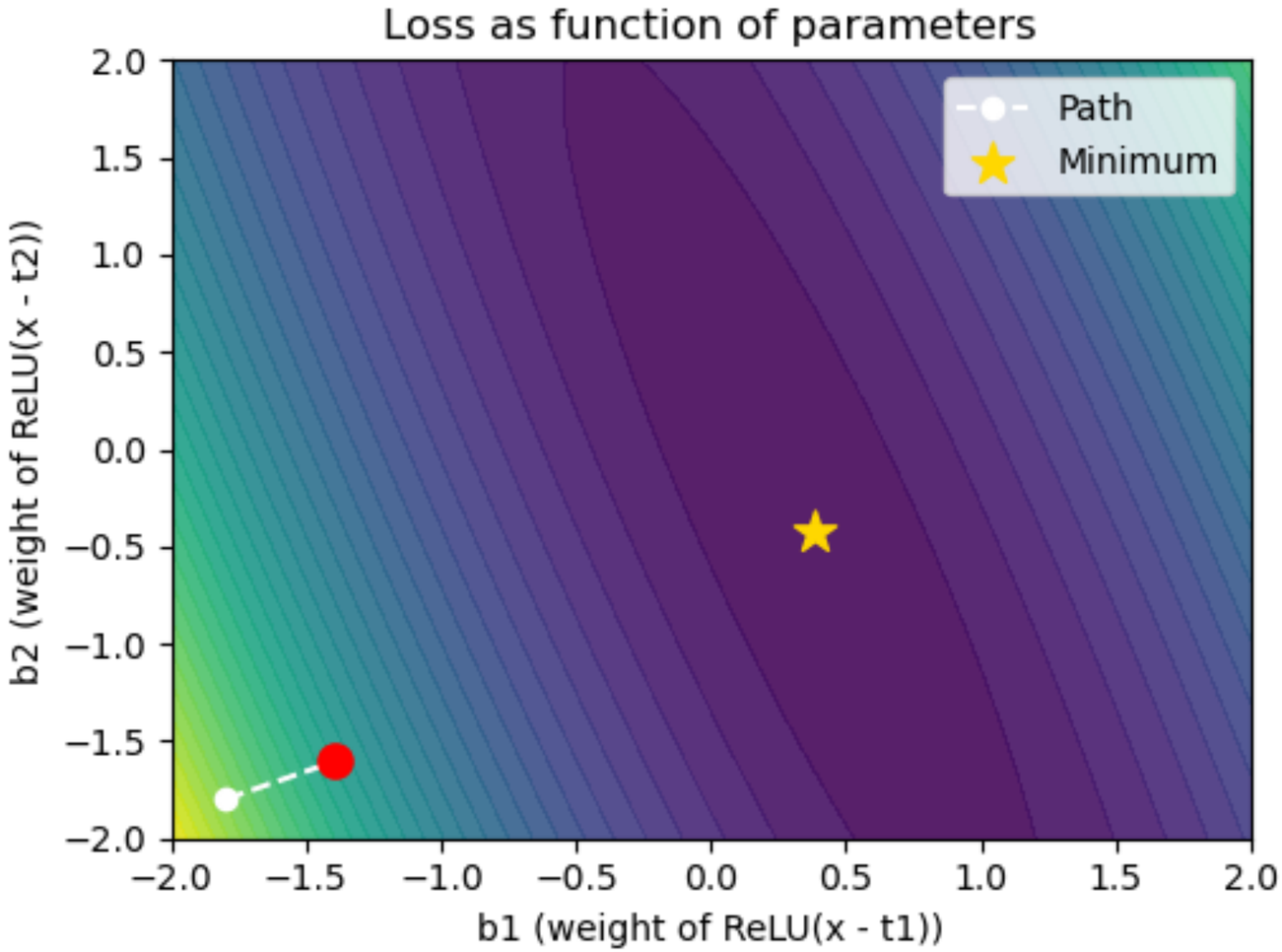
Redes Neuronales: Entrenamiento



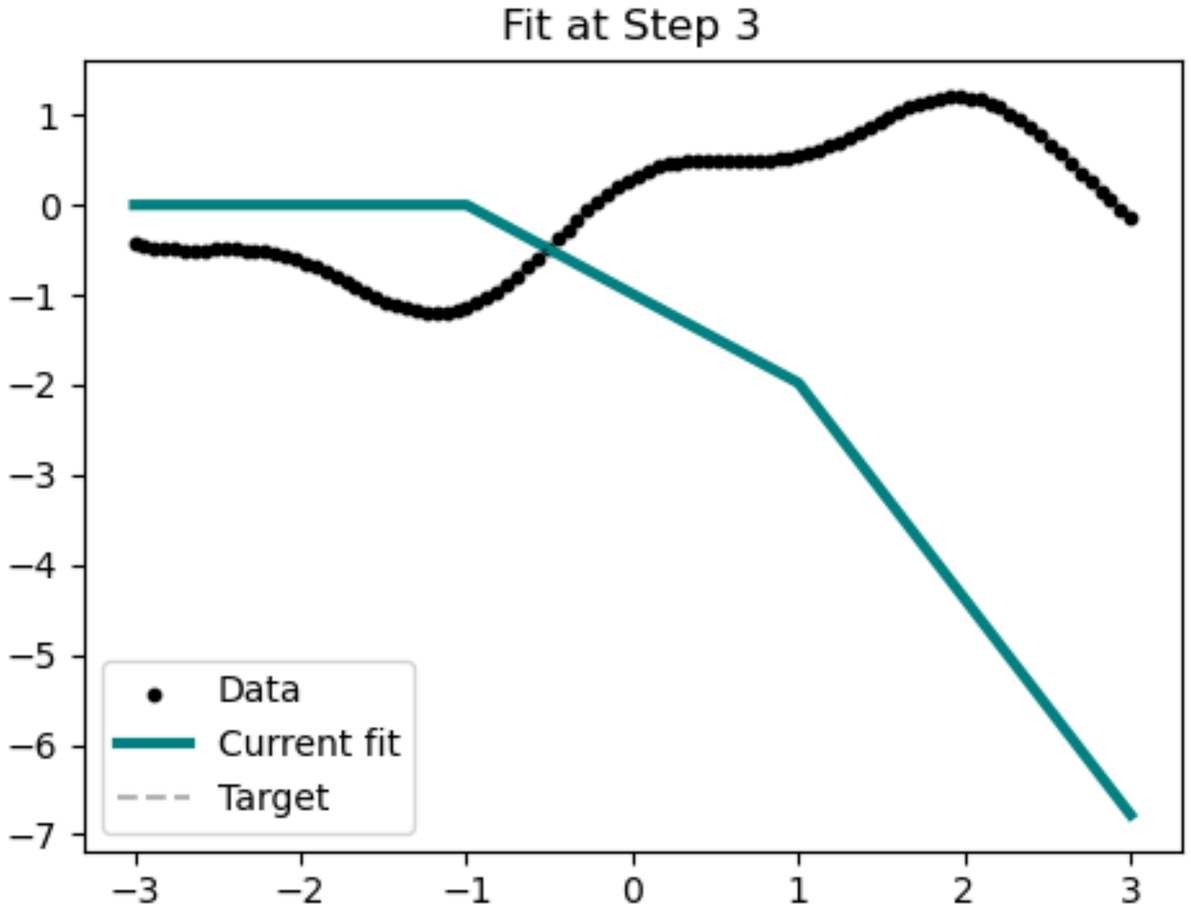


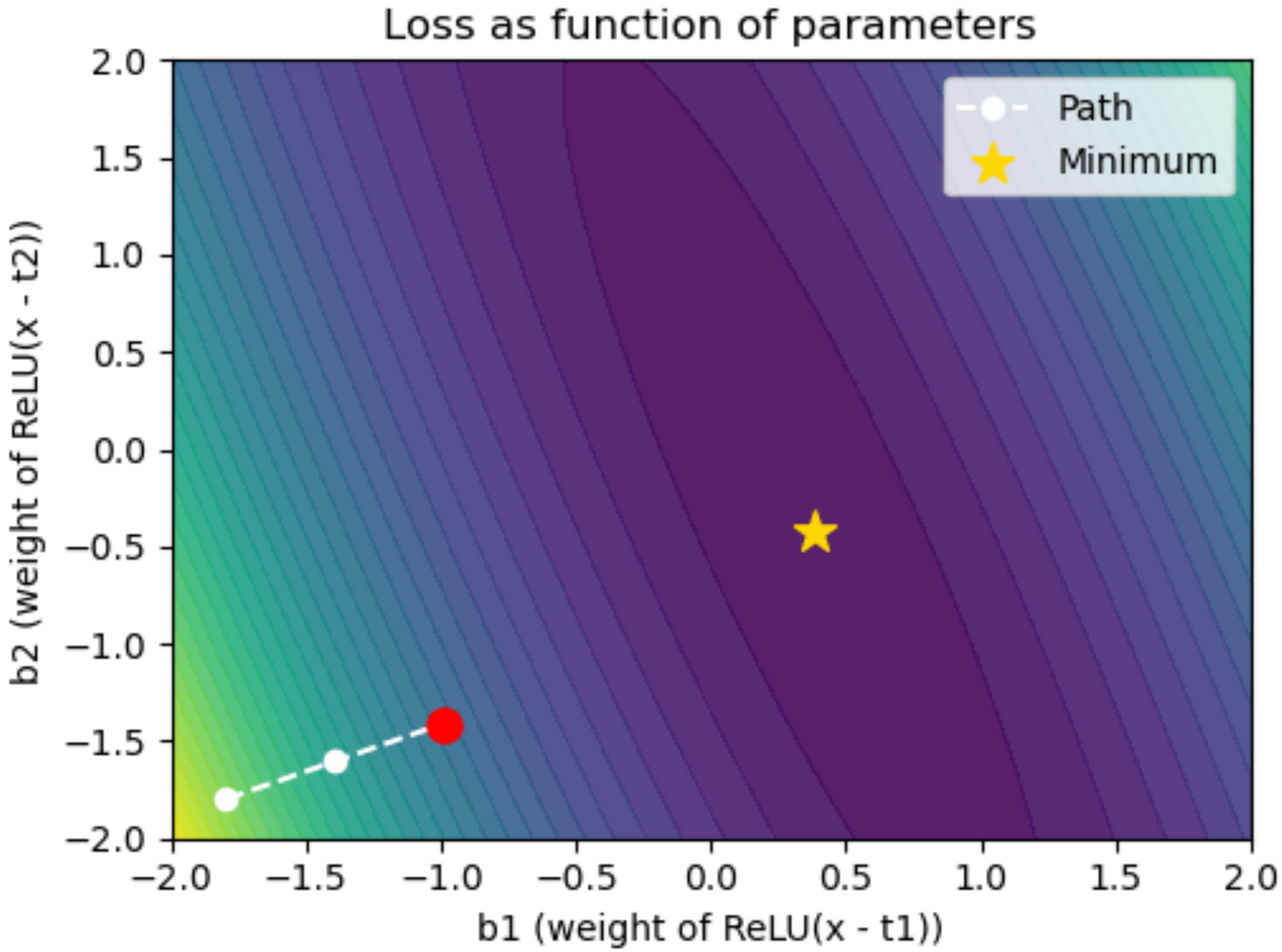
Redes Neuronales: Entrenamiento



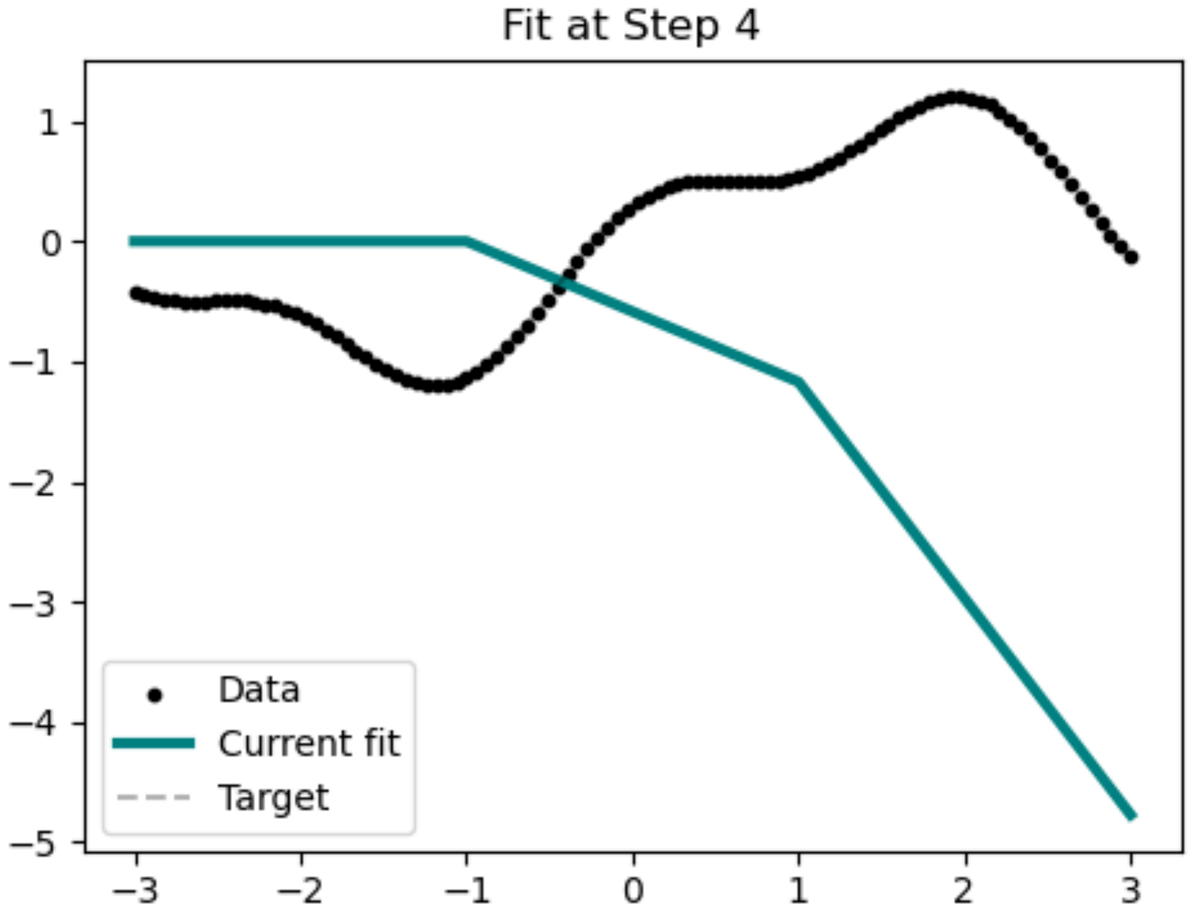


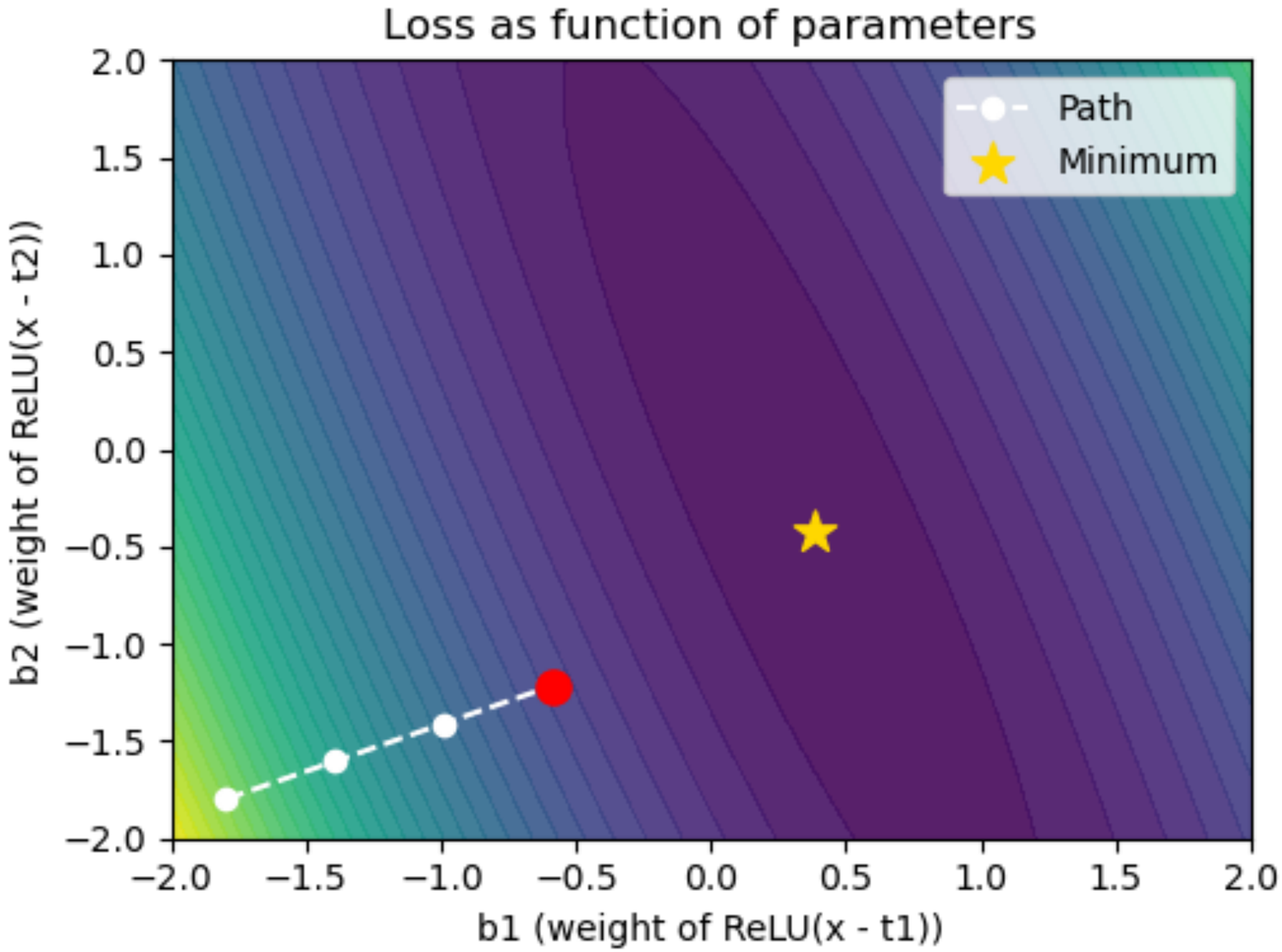
Redes Neuronales: Entrenamiento



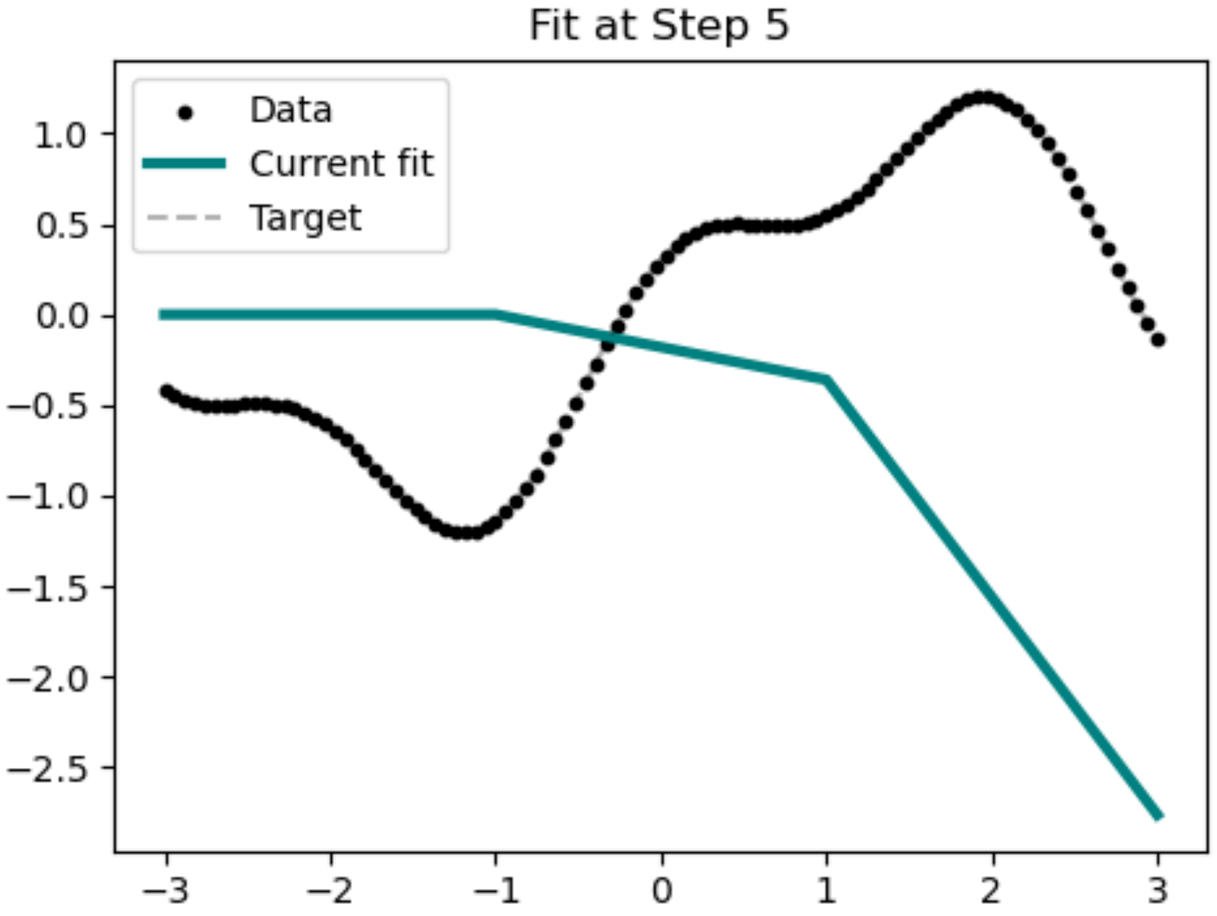


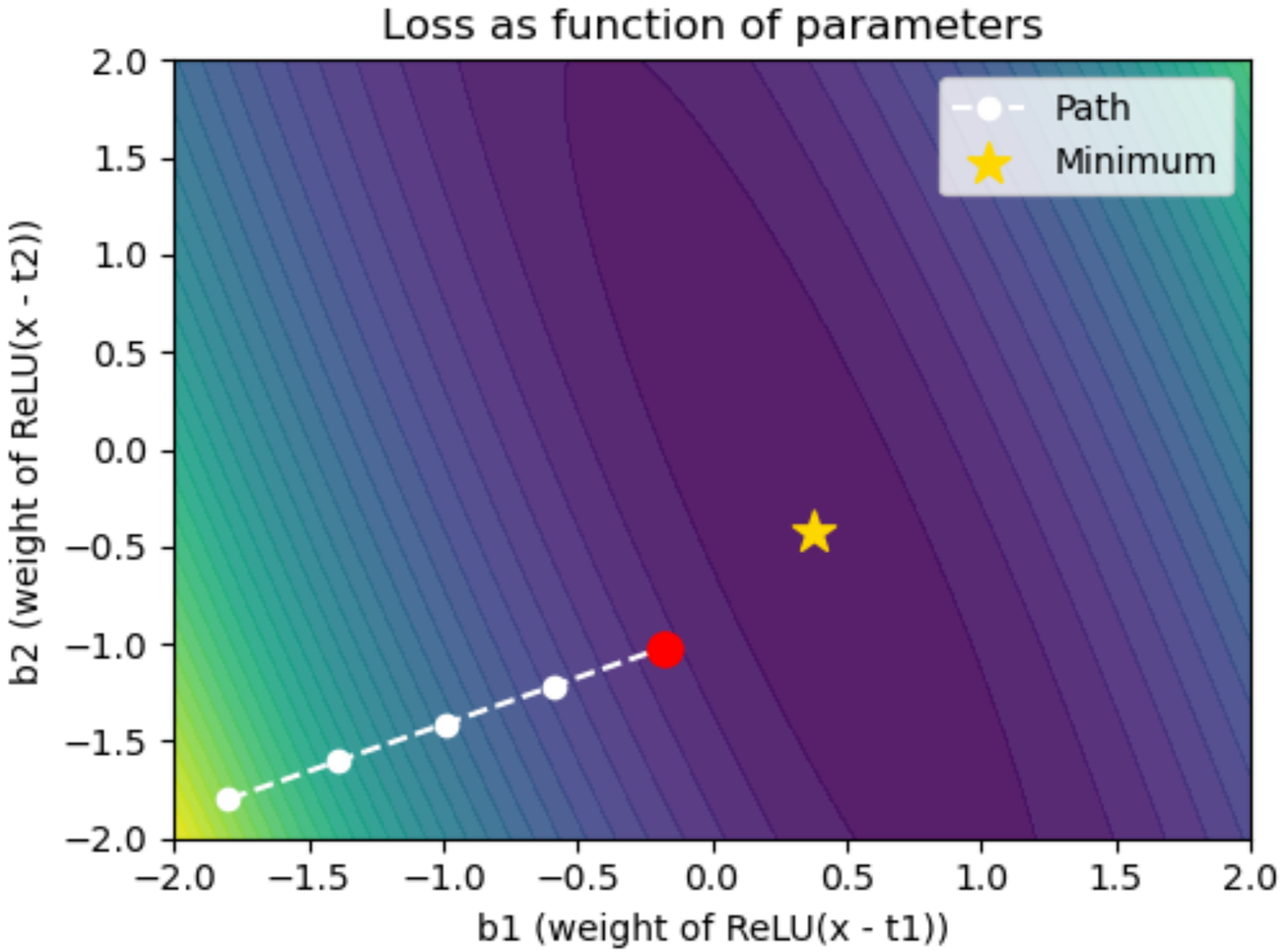
Redes Neuronales: Entrenamiento



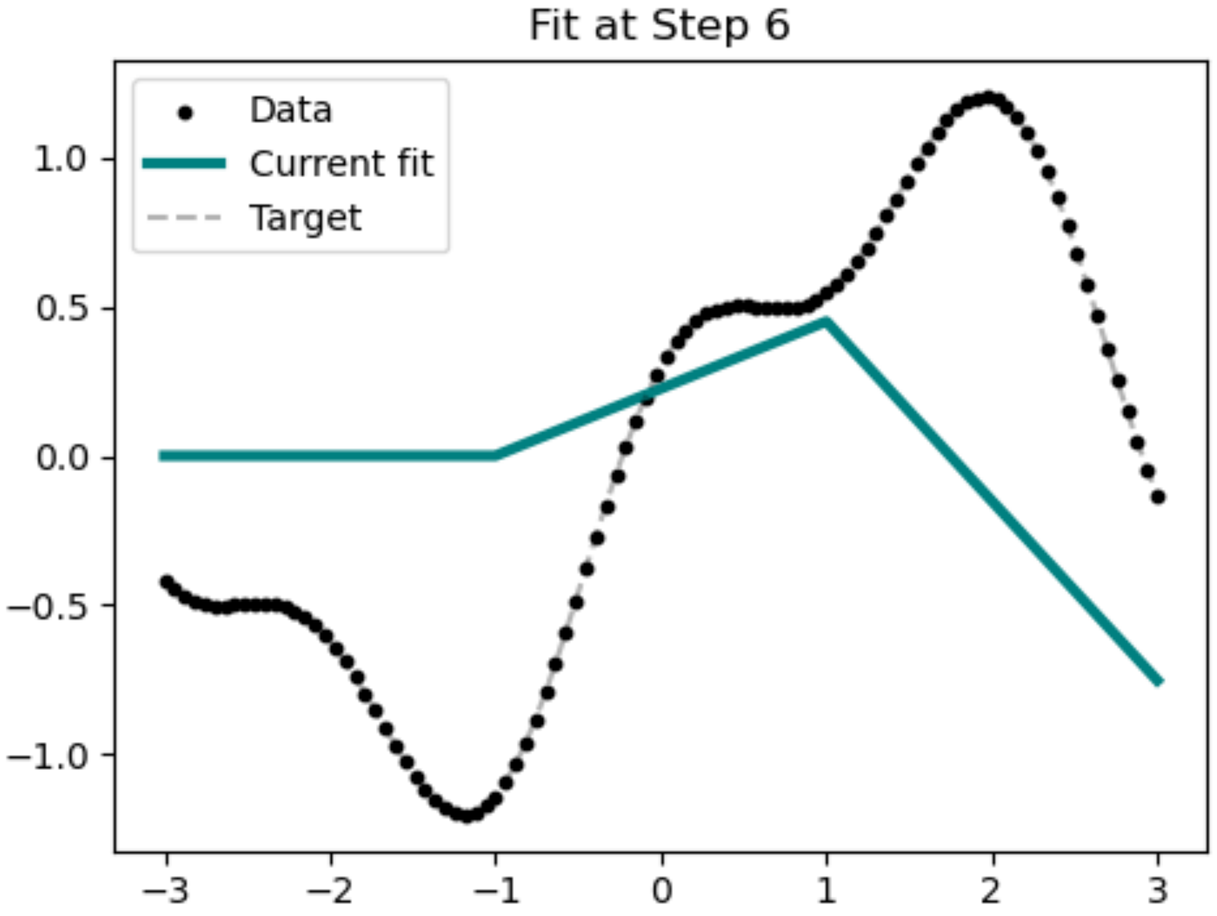


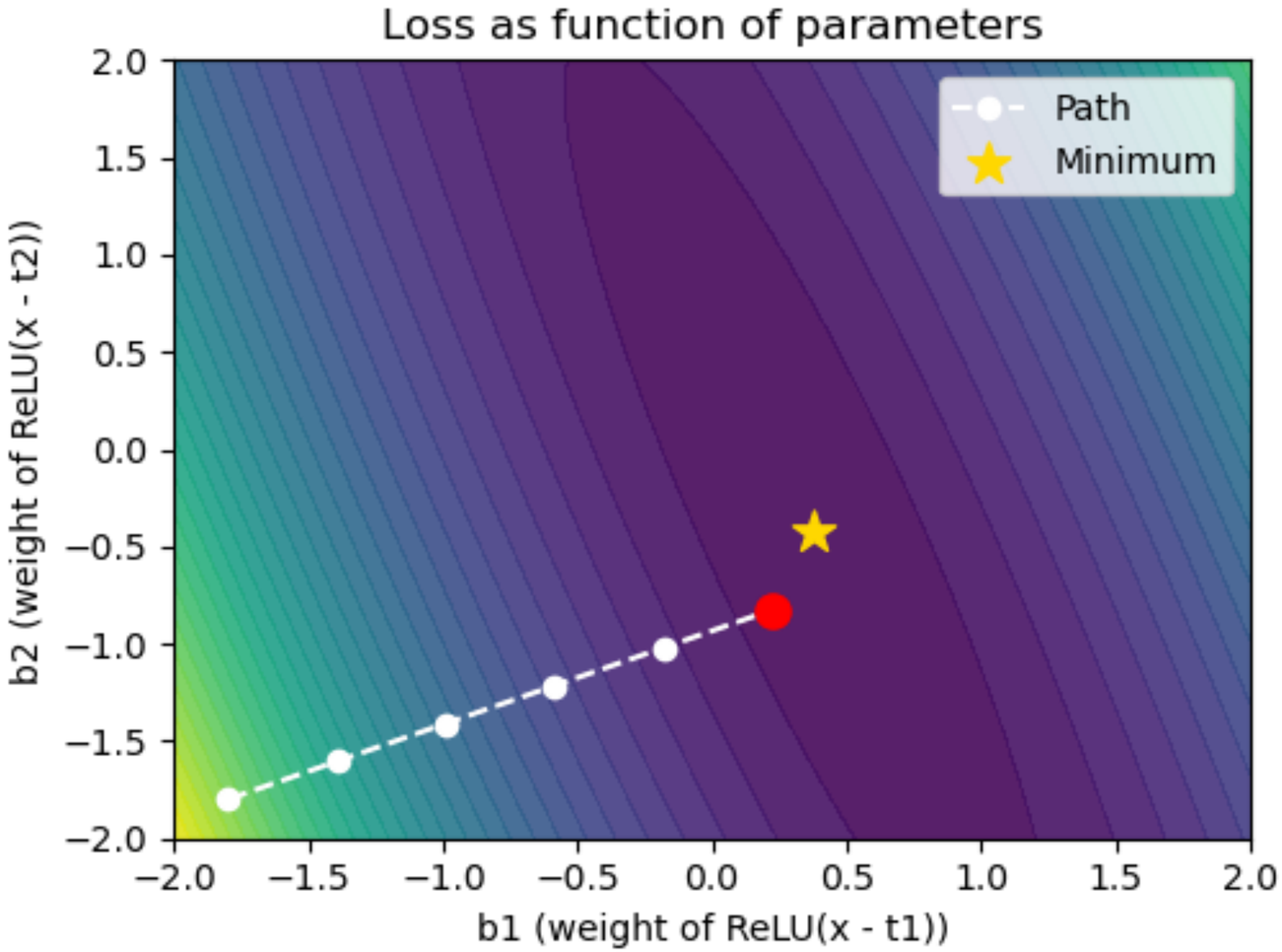
Redes Neuronales: Entrenamiento



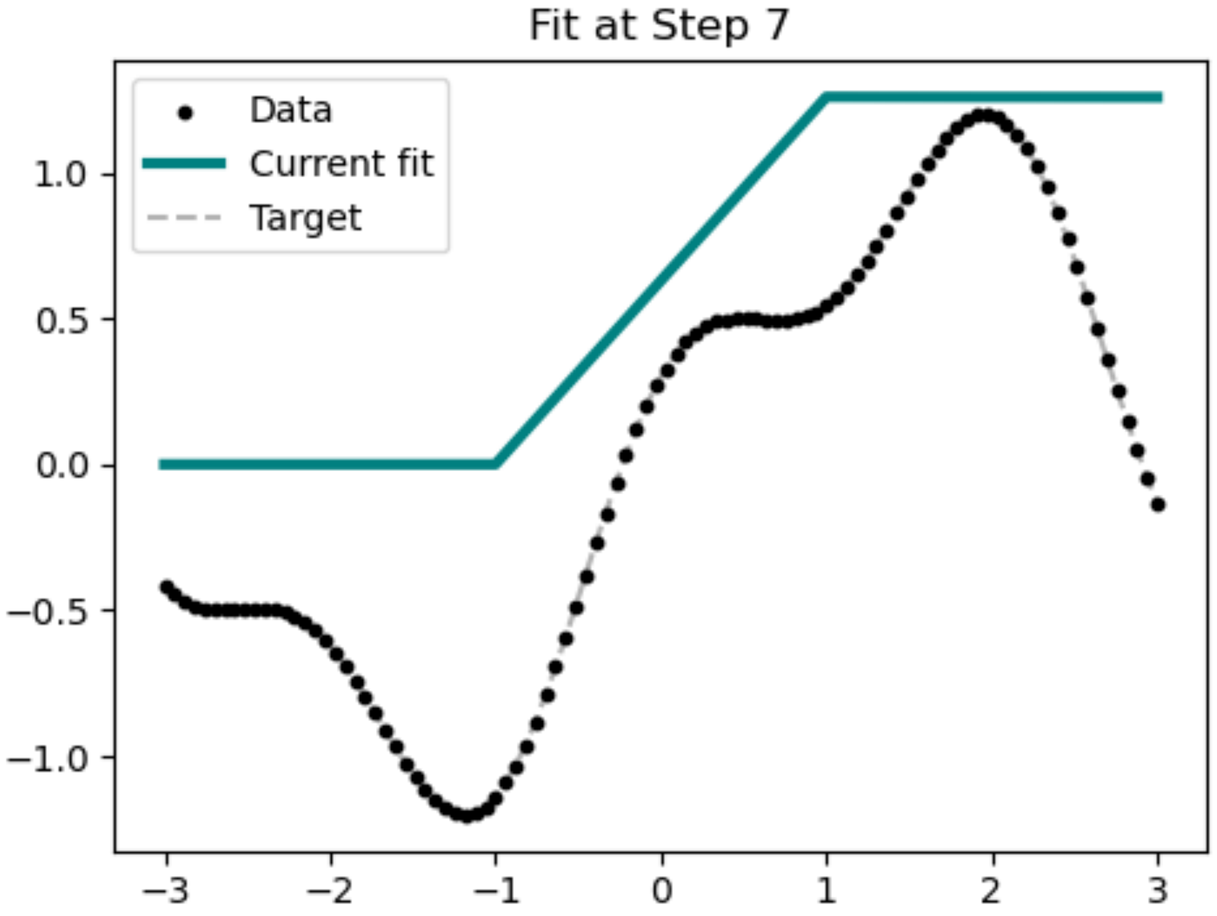


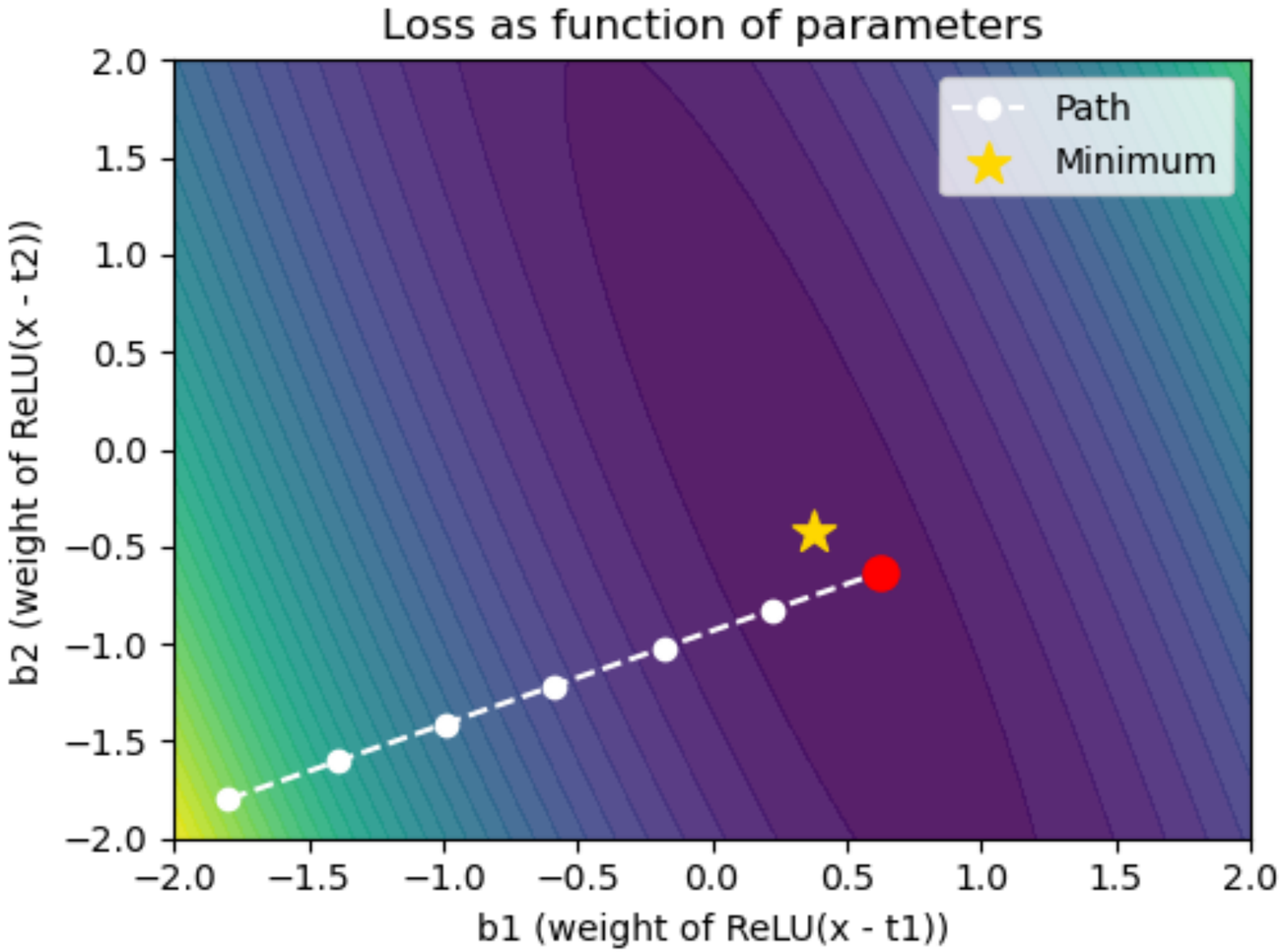
Redes Neuronales: Entrenamiento



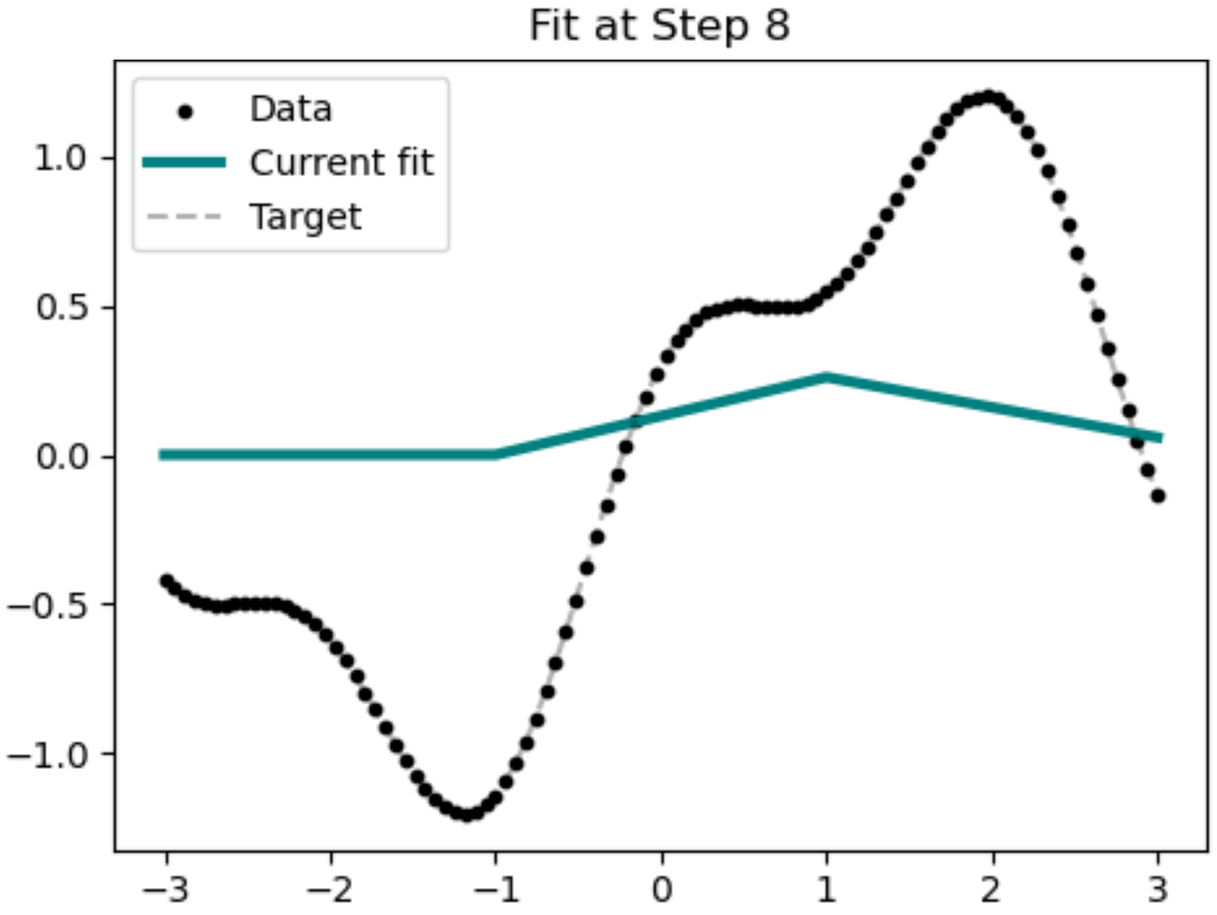


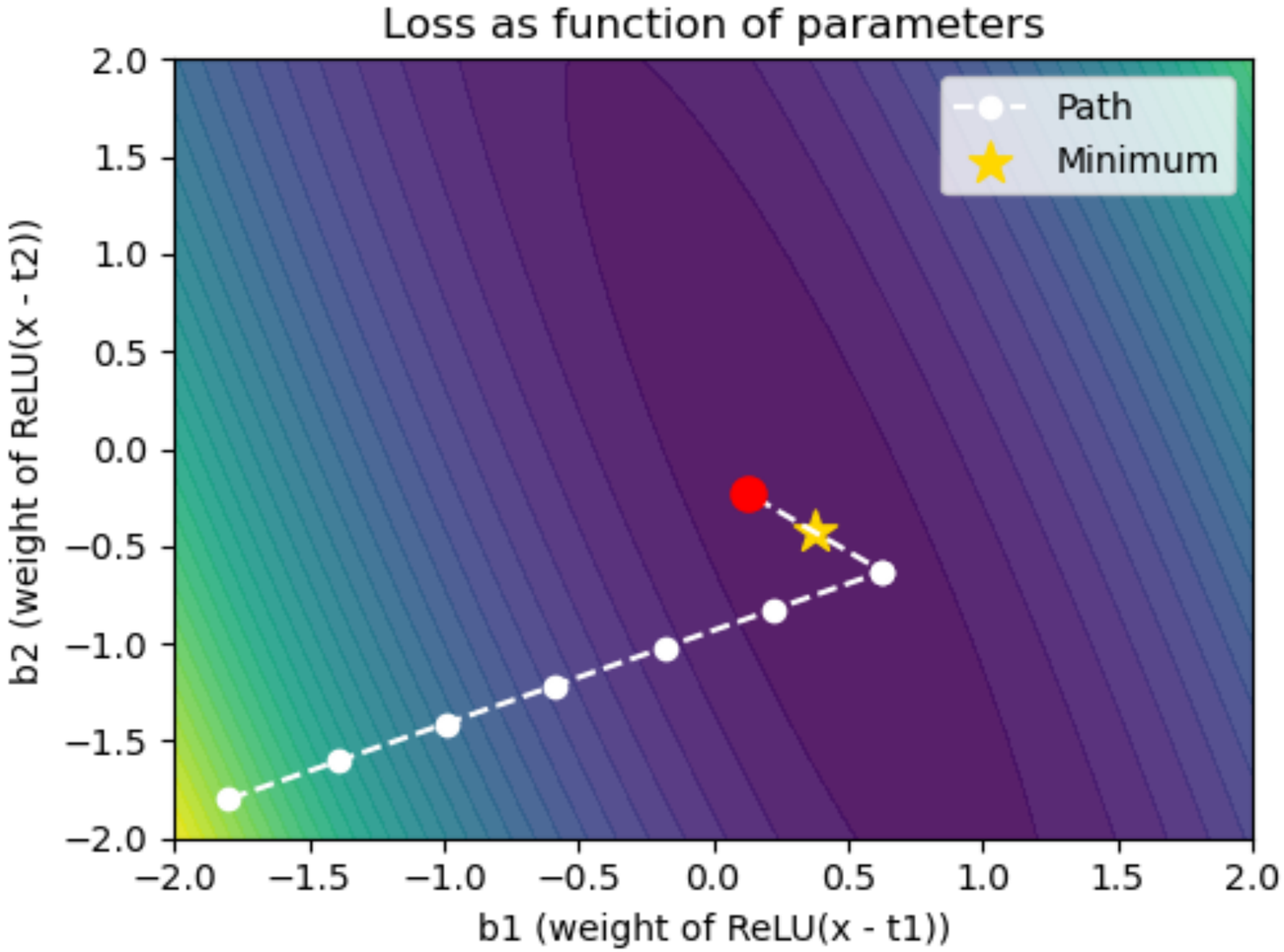
Redes Neuronales: Entrenamiento



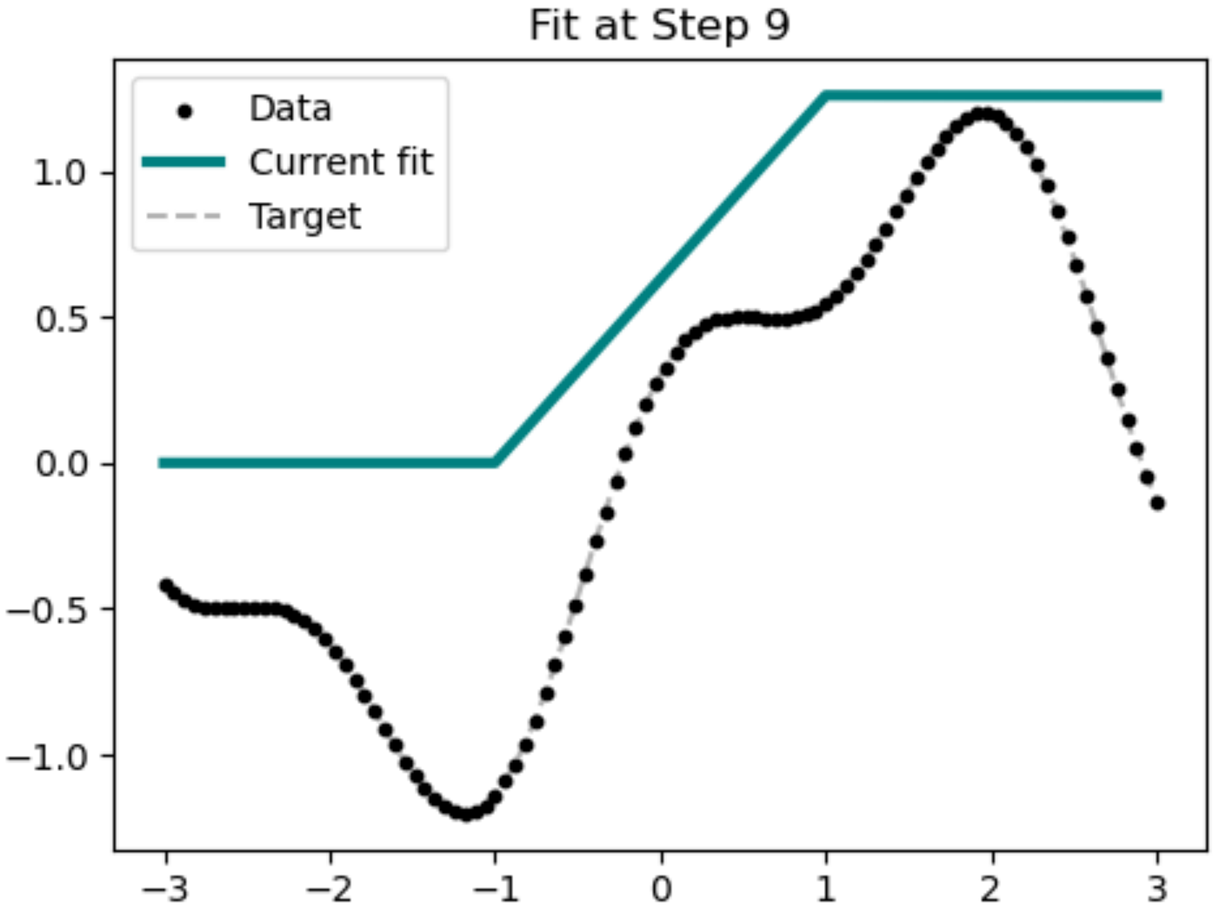


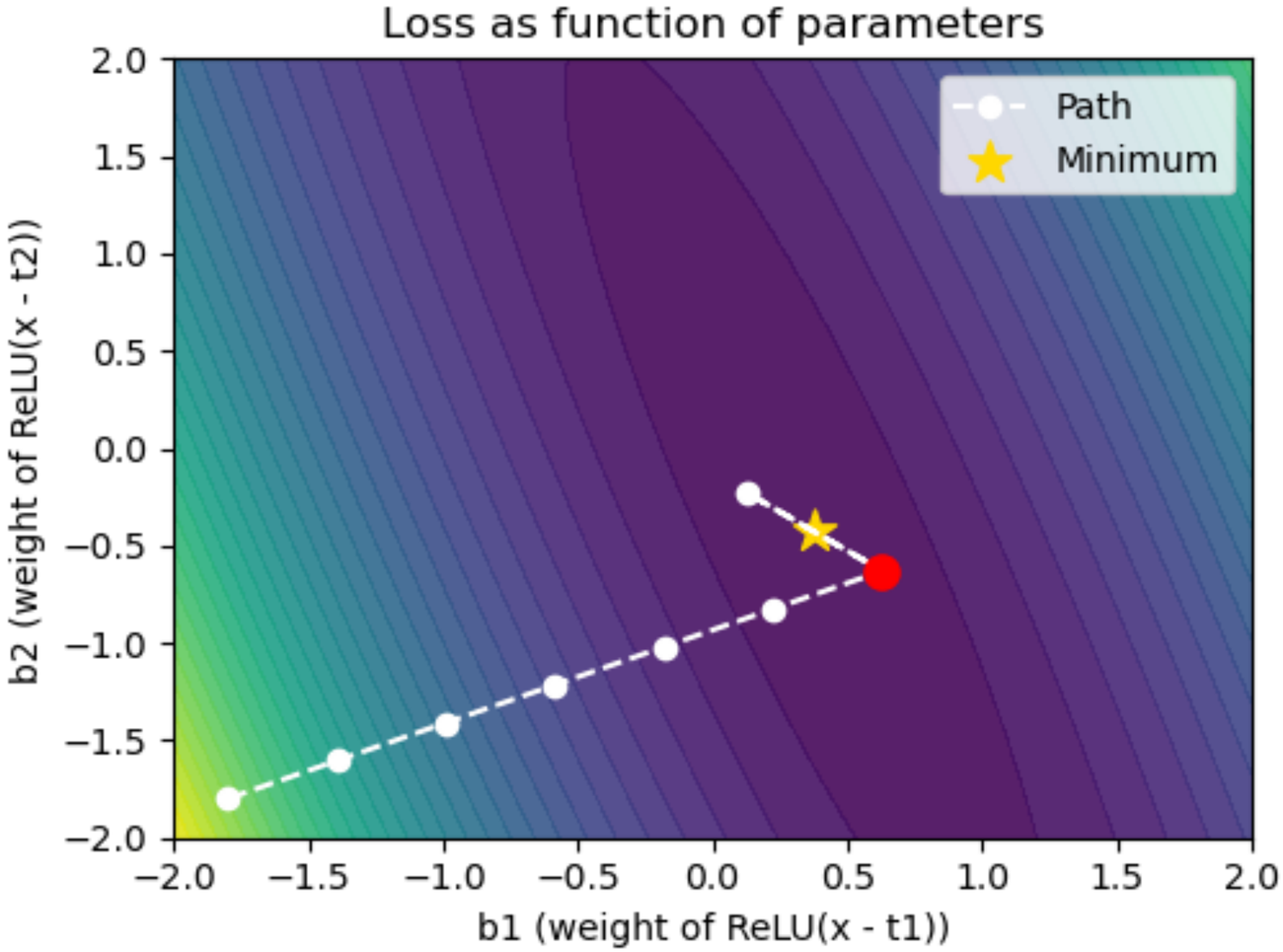
Redes Neuronales: Entrenamiento



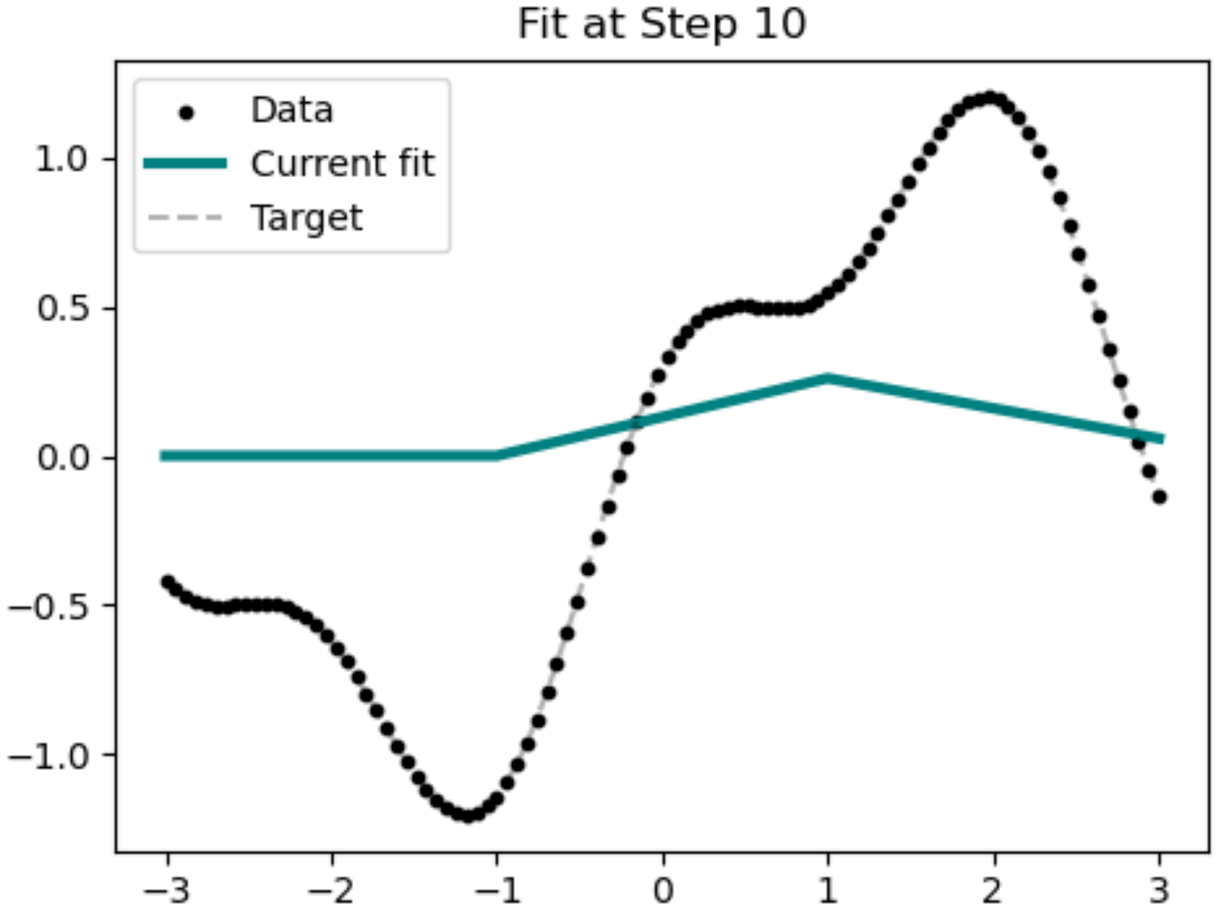


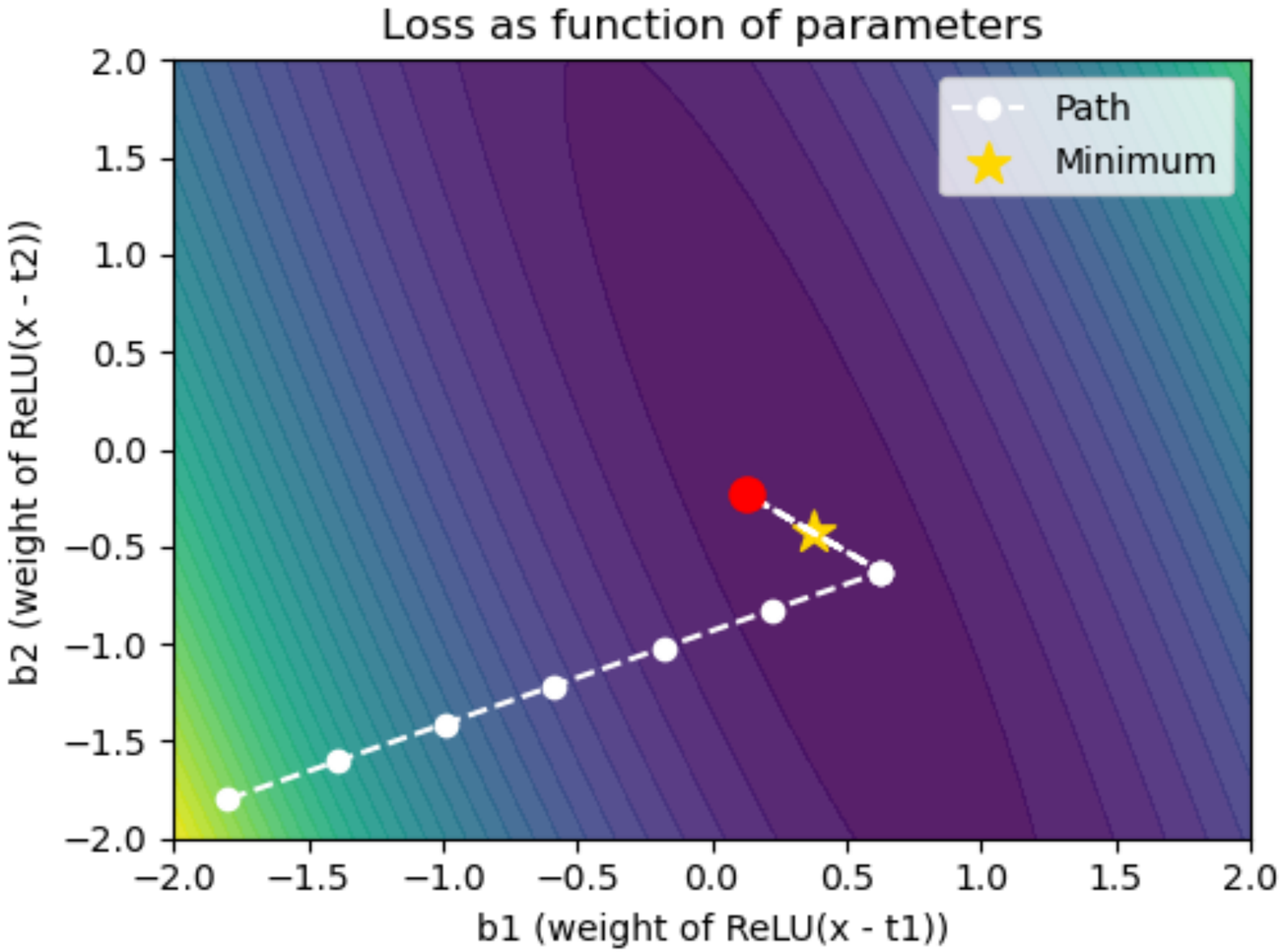
Redes Neuronales: Entrenamiento



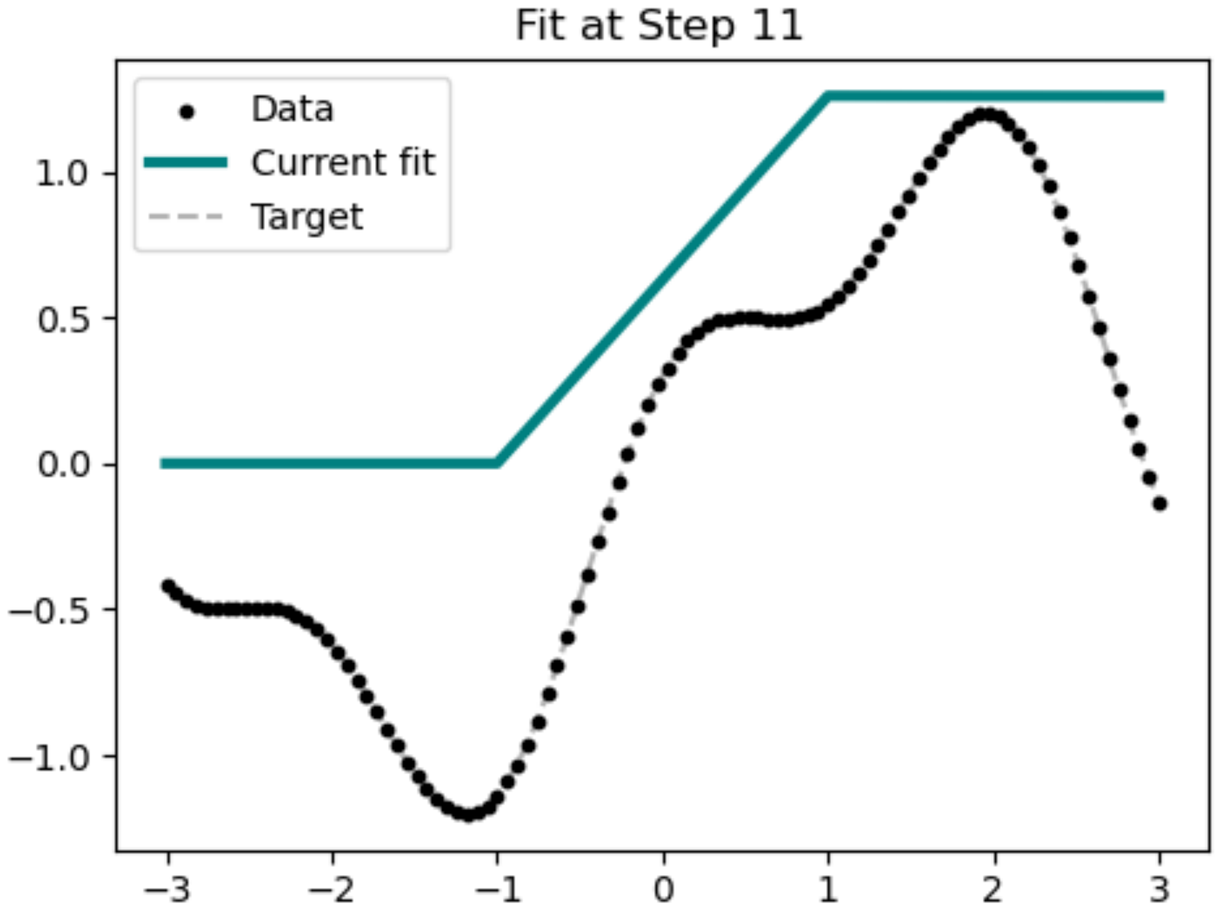


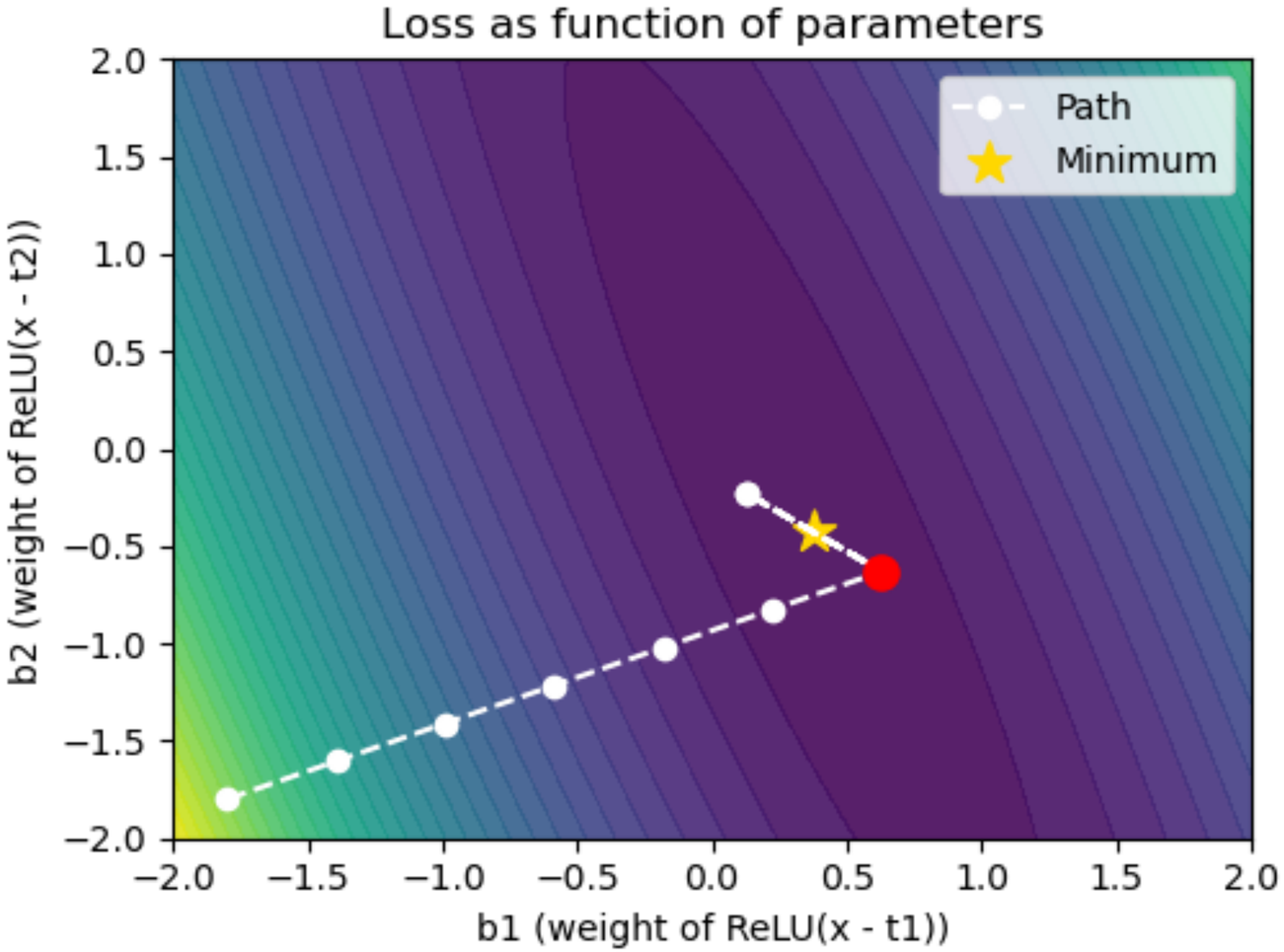
Redes Neuronales: Entrenamiento





Redes Neuronales: Entrenamiento

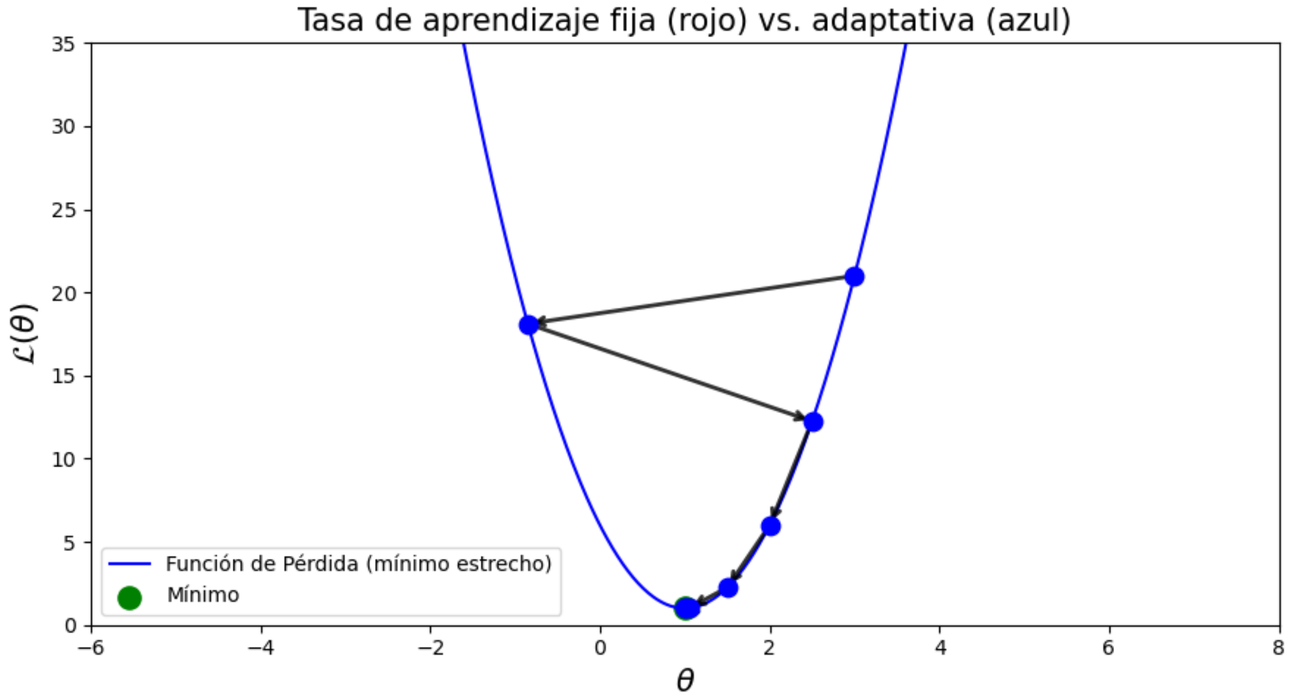




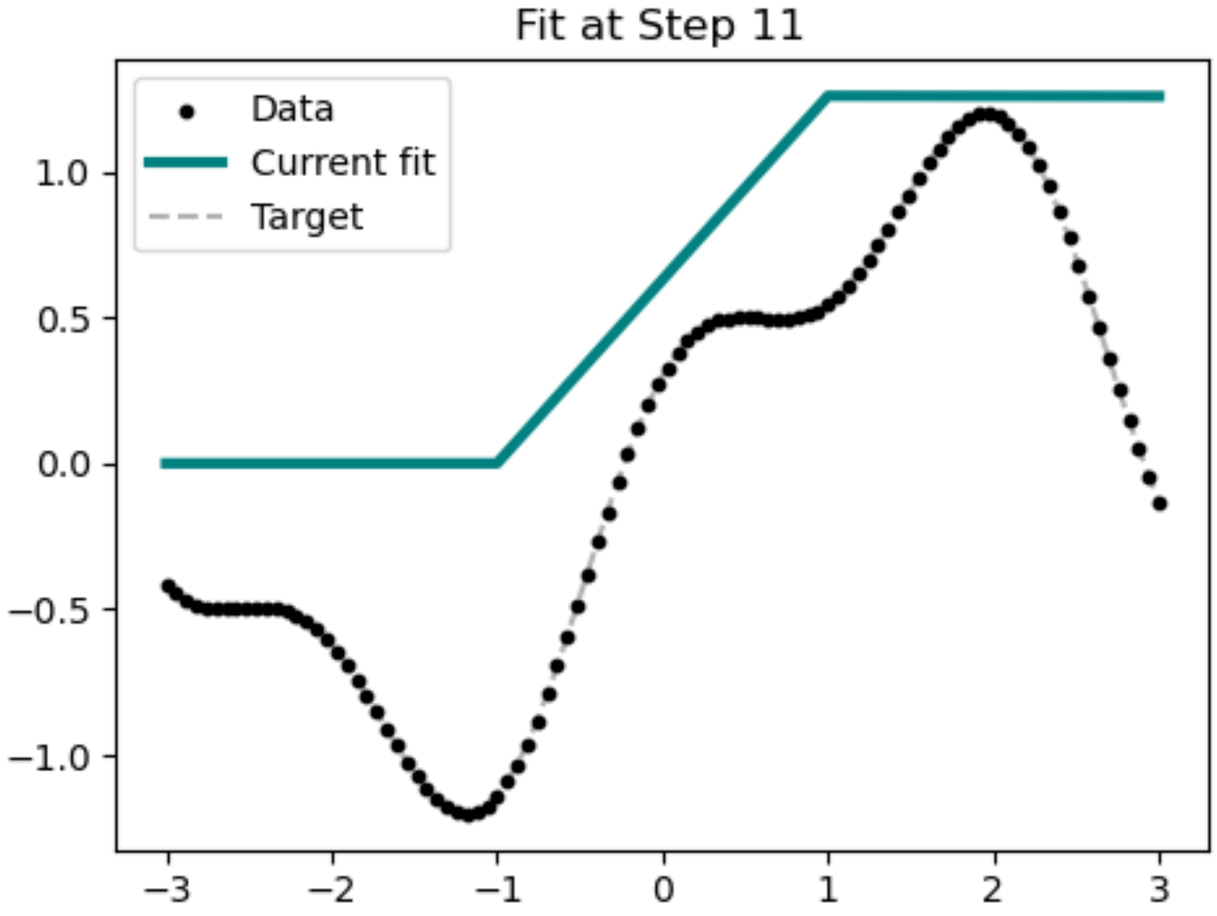
Redes Neuronales: Aprendizaje Adaptativo

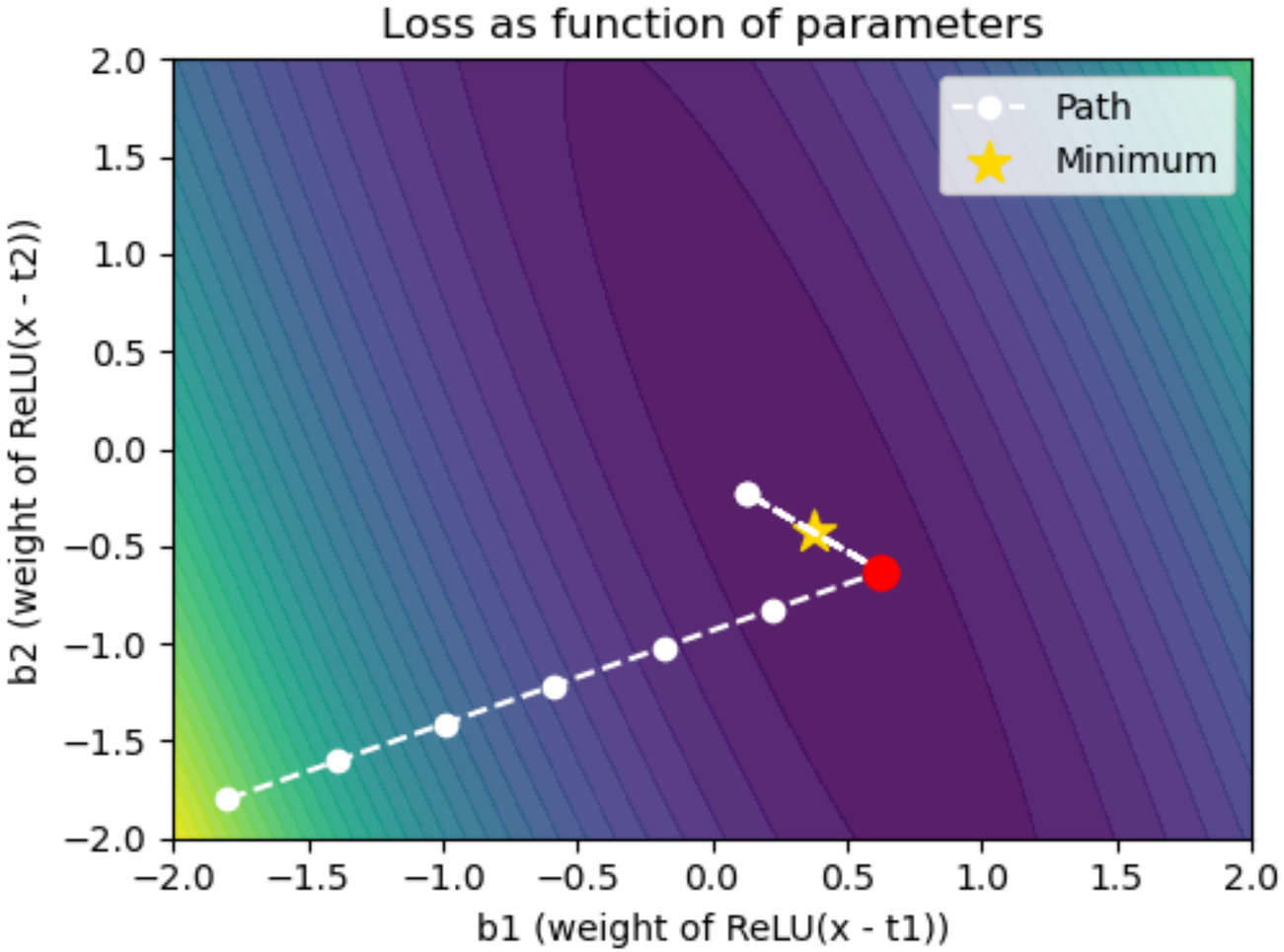
$$\theta_{t+1} = \theta_t - \eta_t \nabla_\theta \mathcal{L}(\theta_t)$$

η_t : tasa adaptativa que disminuye con los pasos para asegurar convergencia

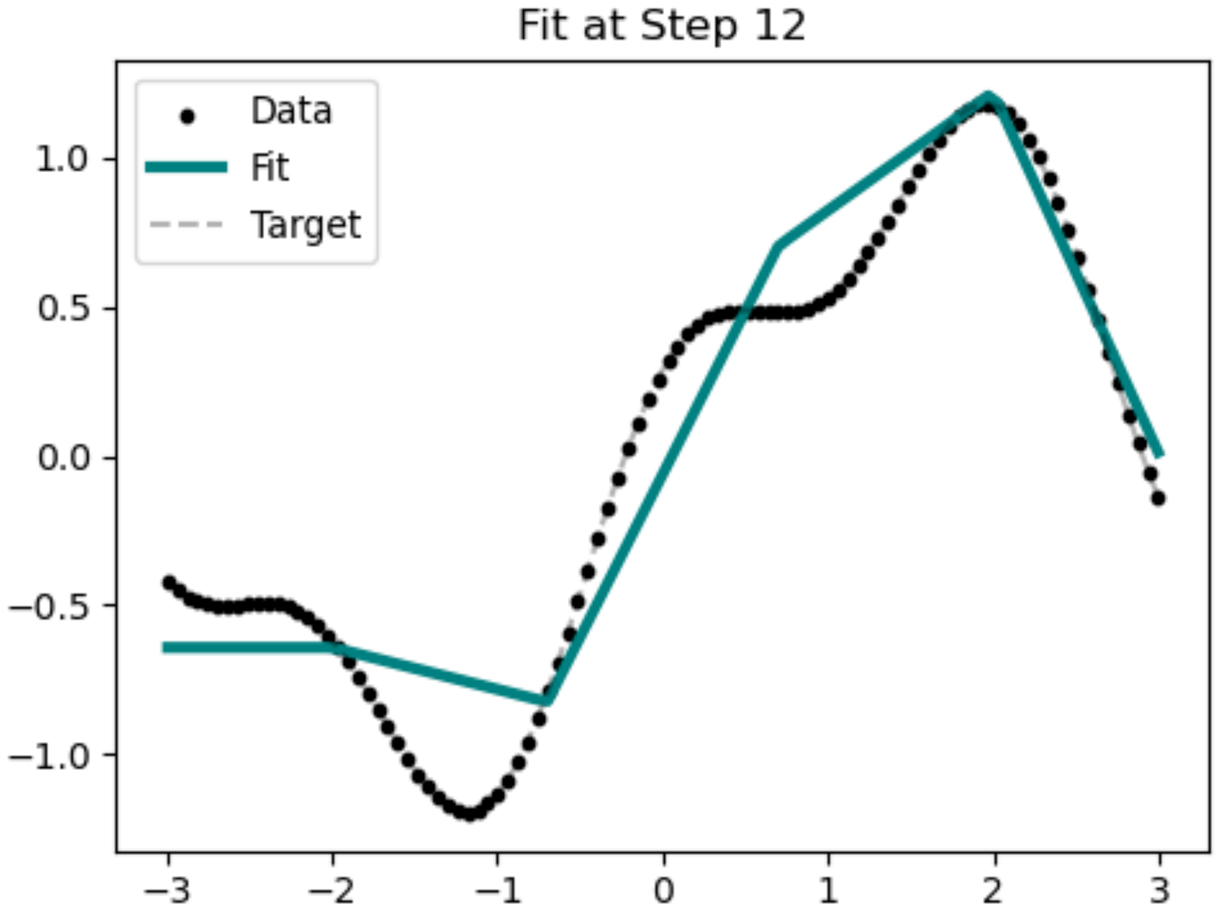


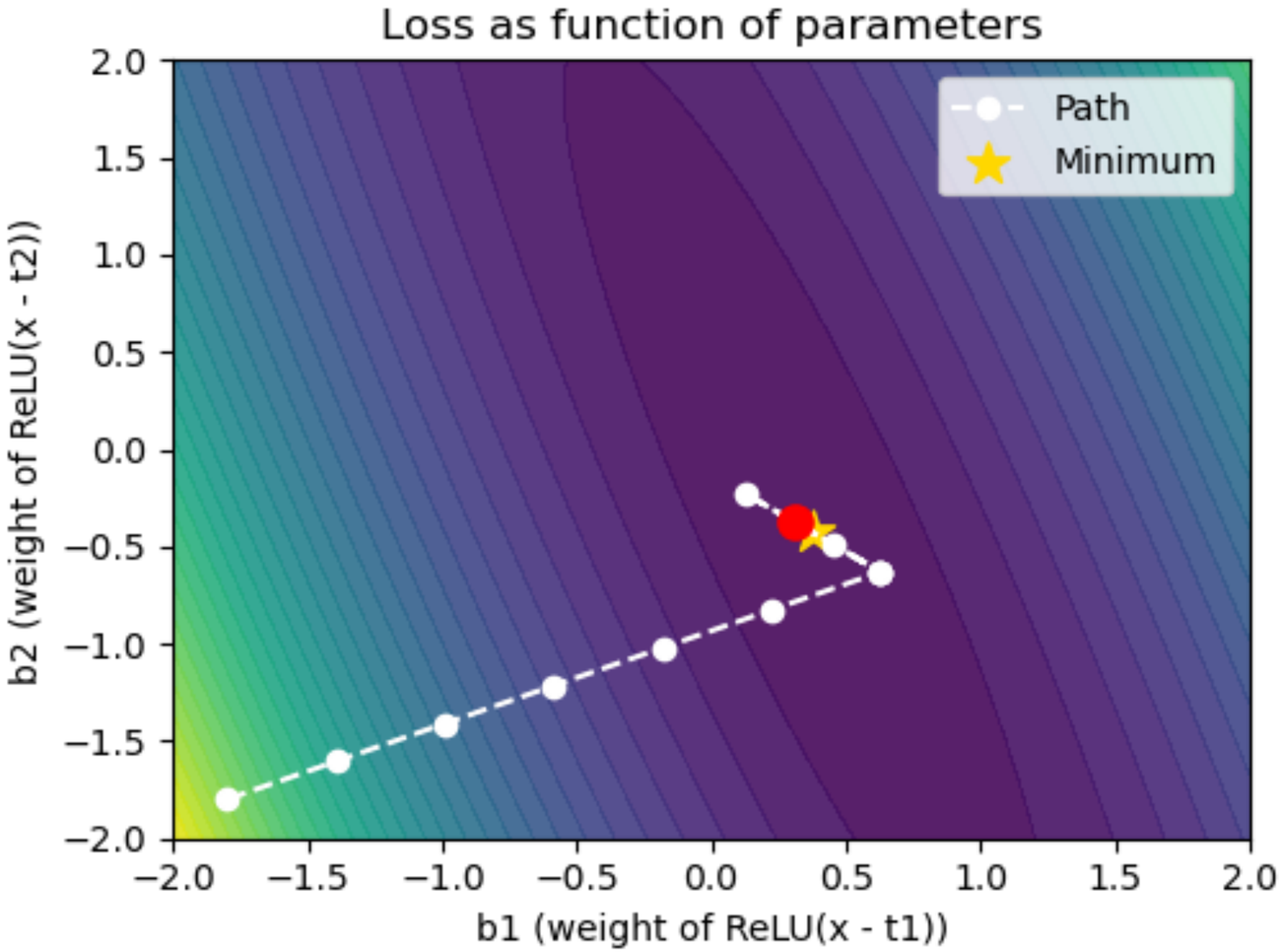
Redes Neuronales: Entrenamiento



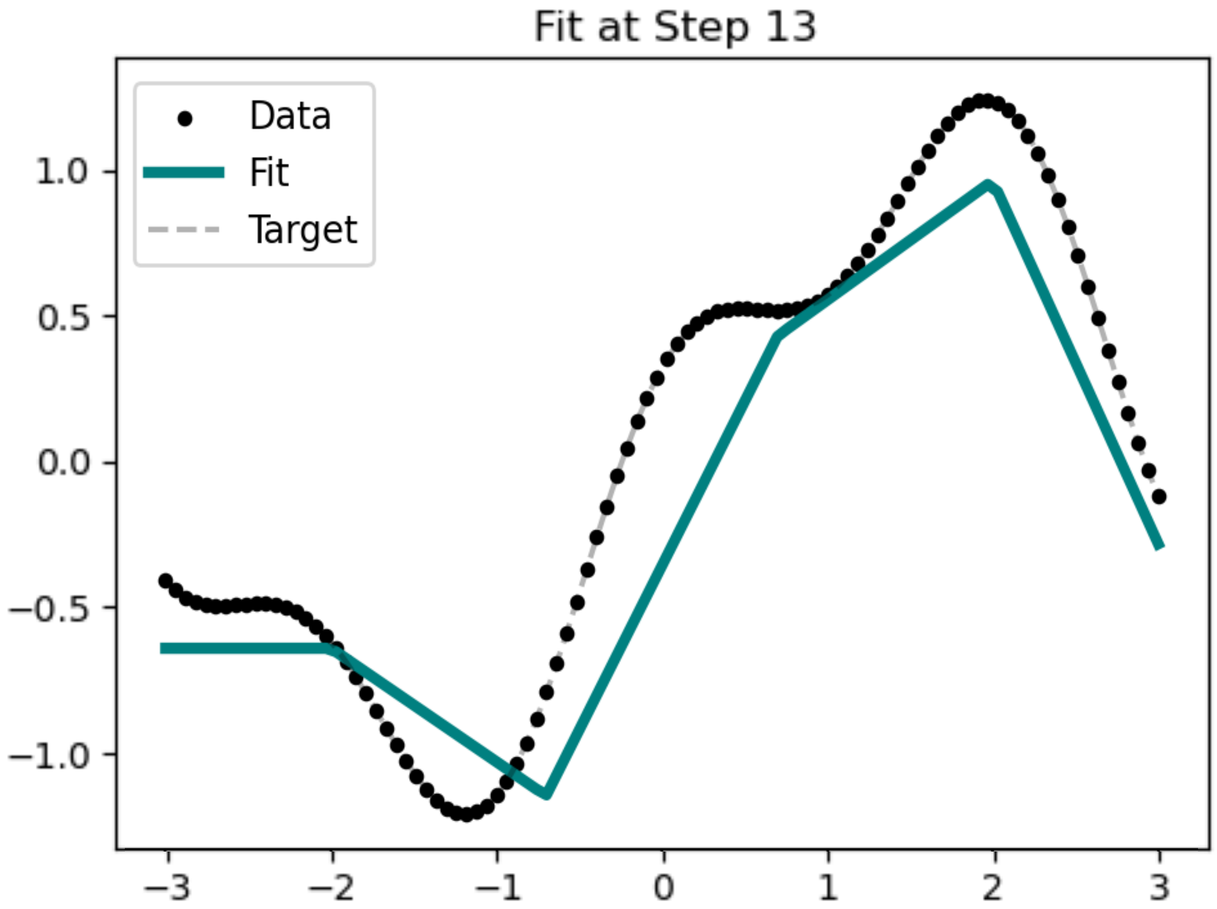


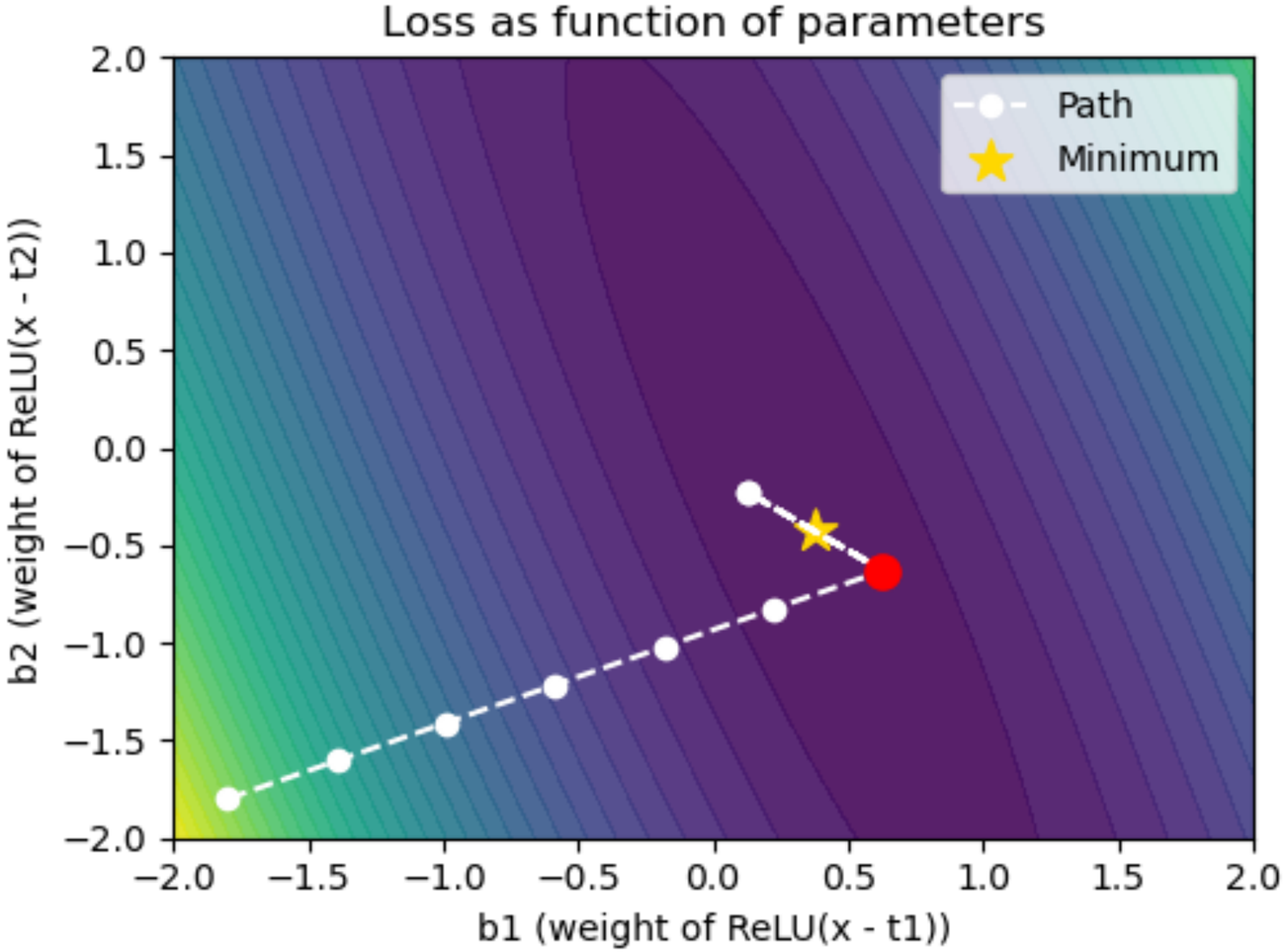
Redes Neuronales: Entrenamiento



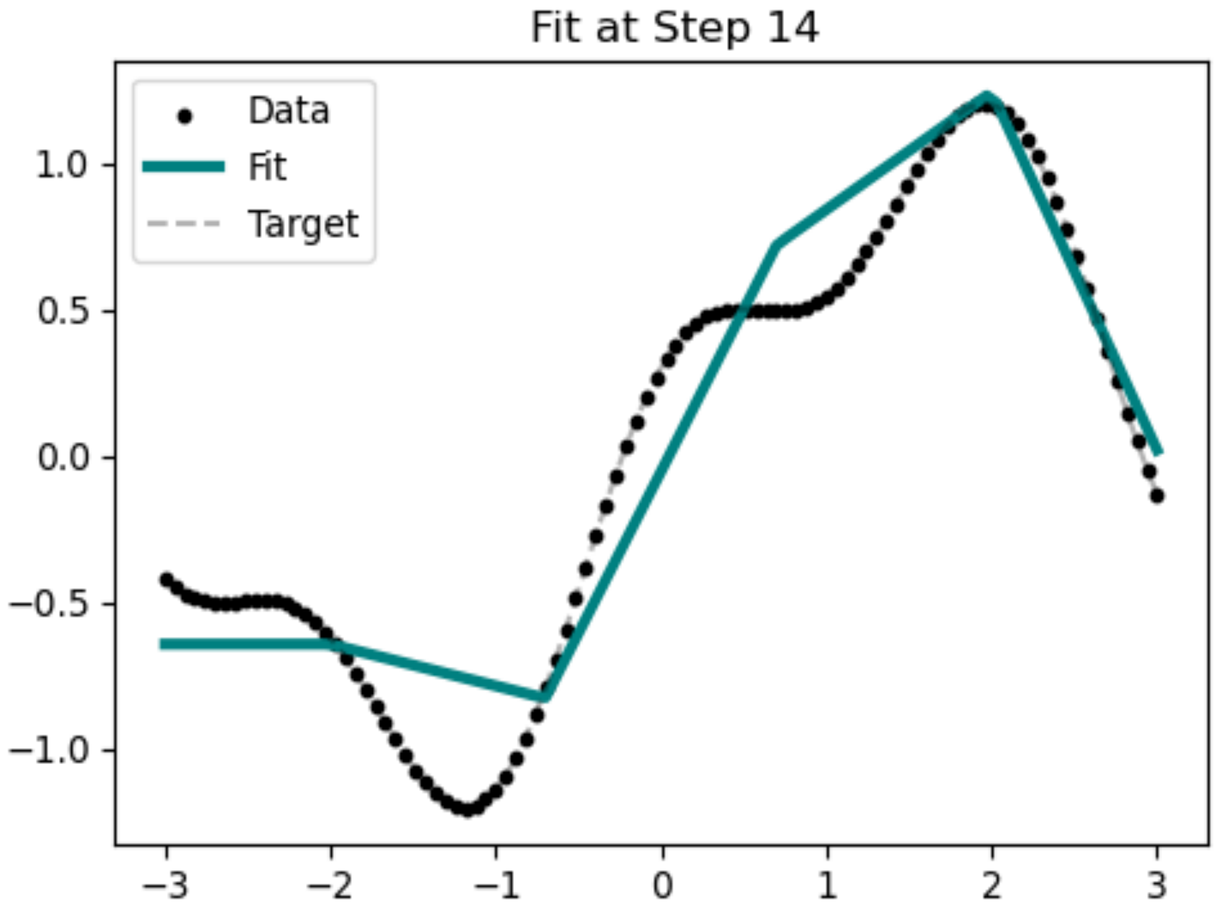


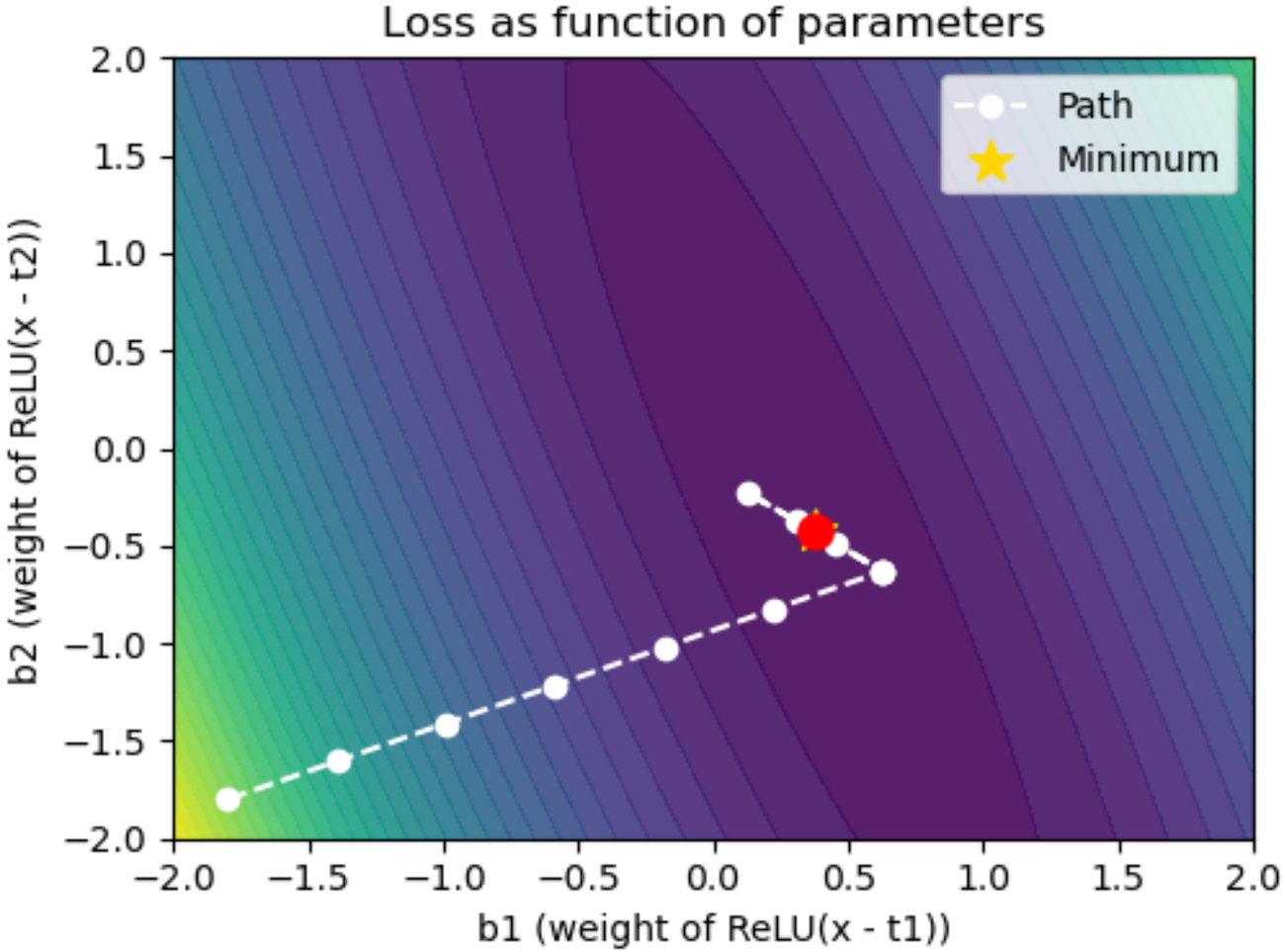
Redes Neuronales: Entrenamiento



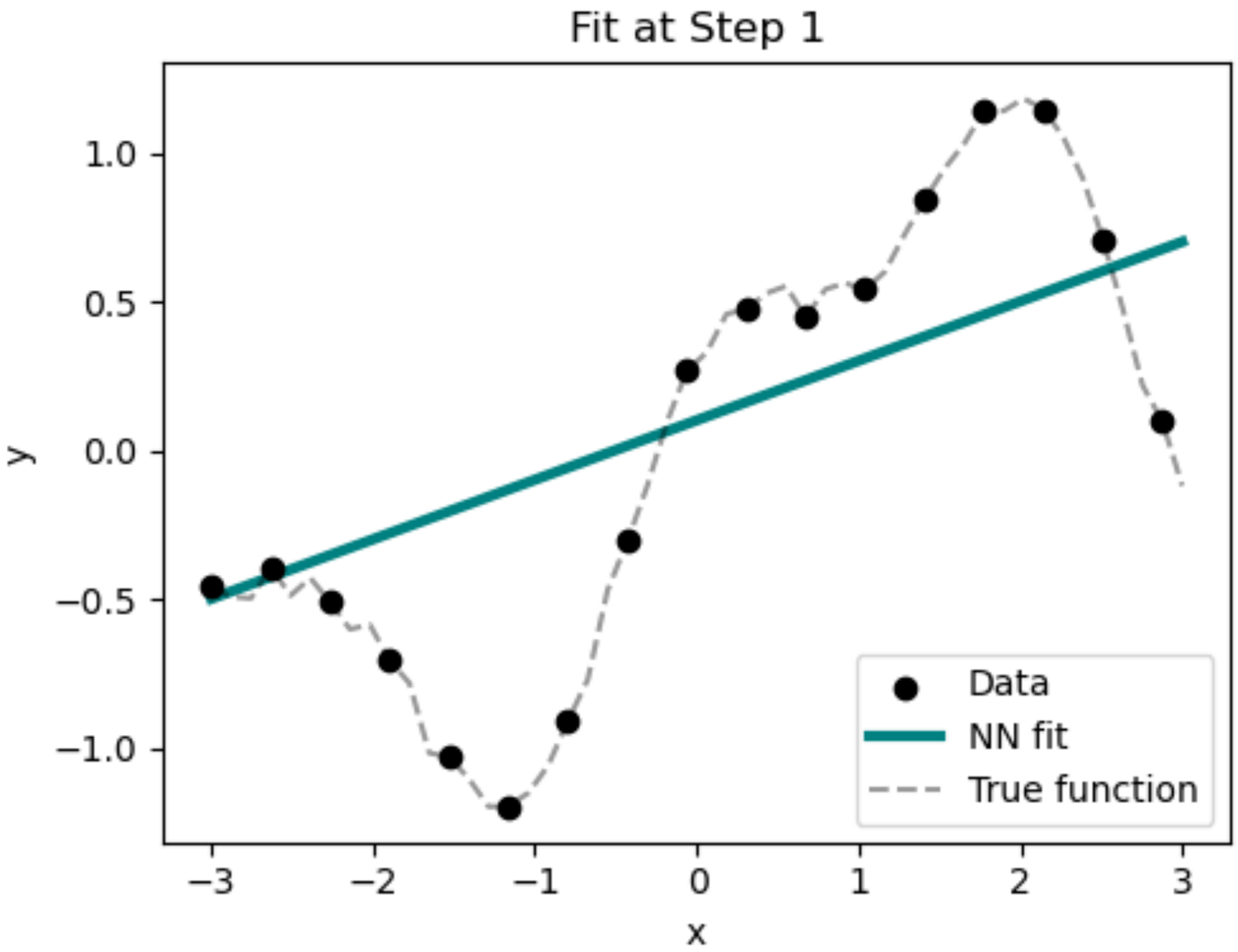


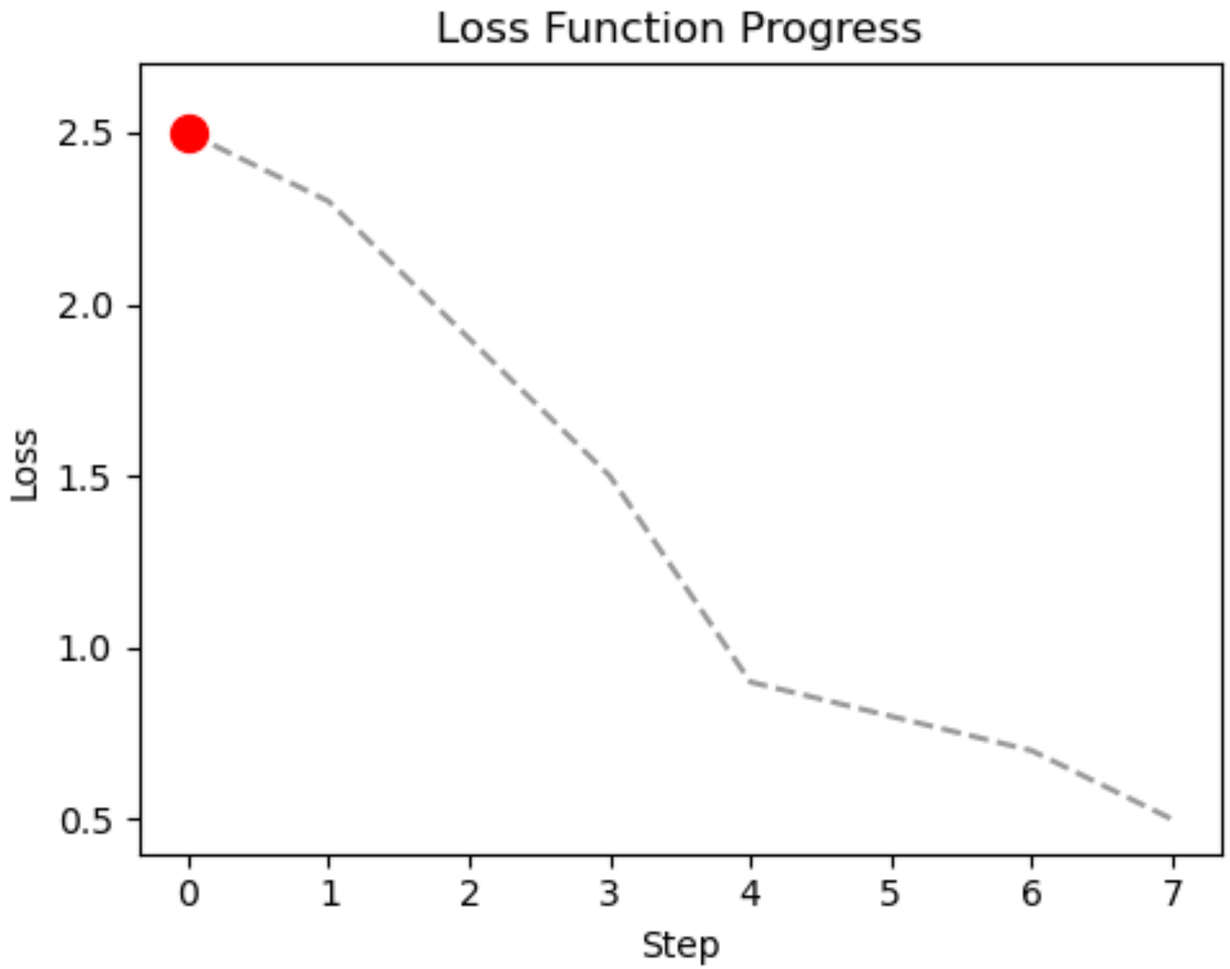
Redes Neuronales: Entrenamiento



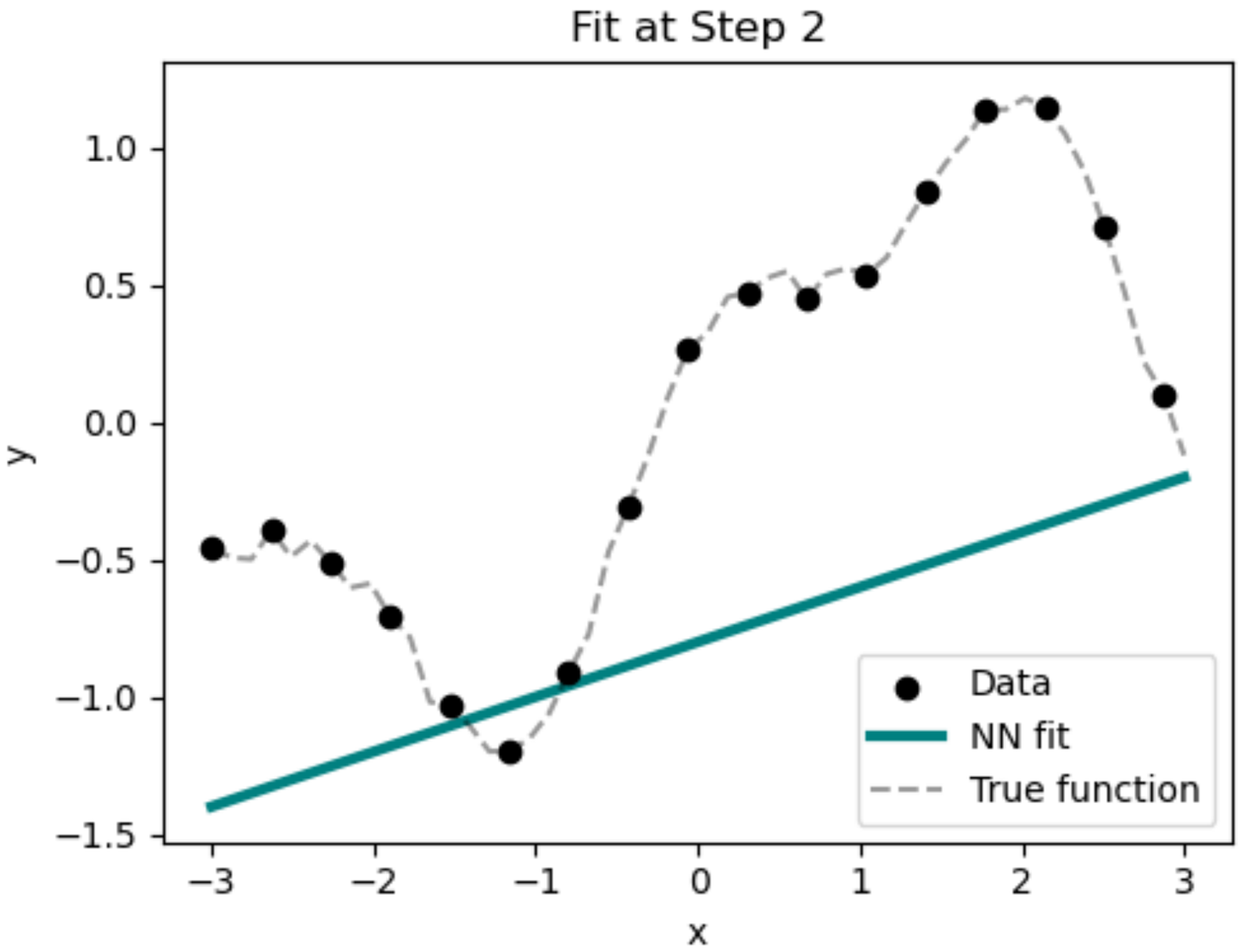


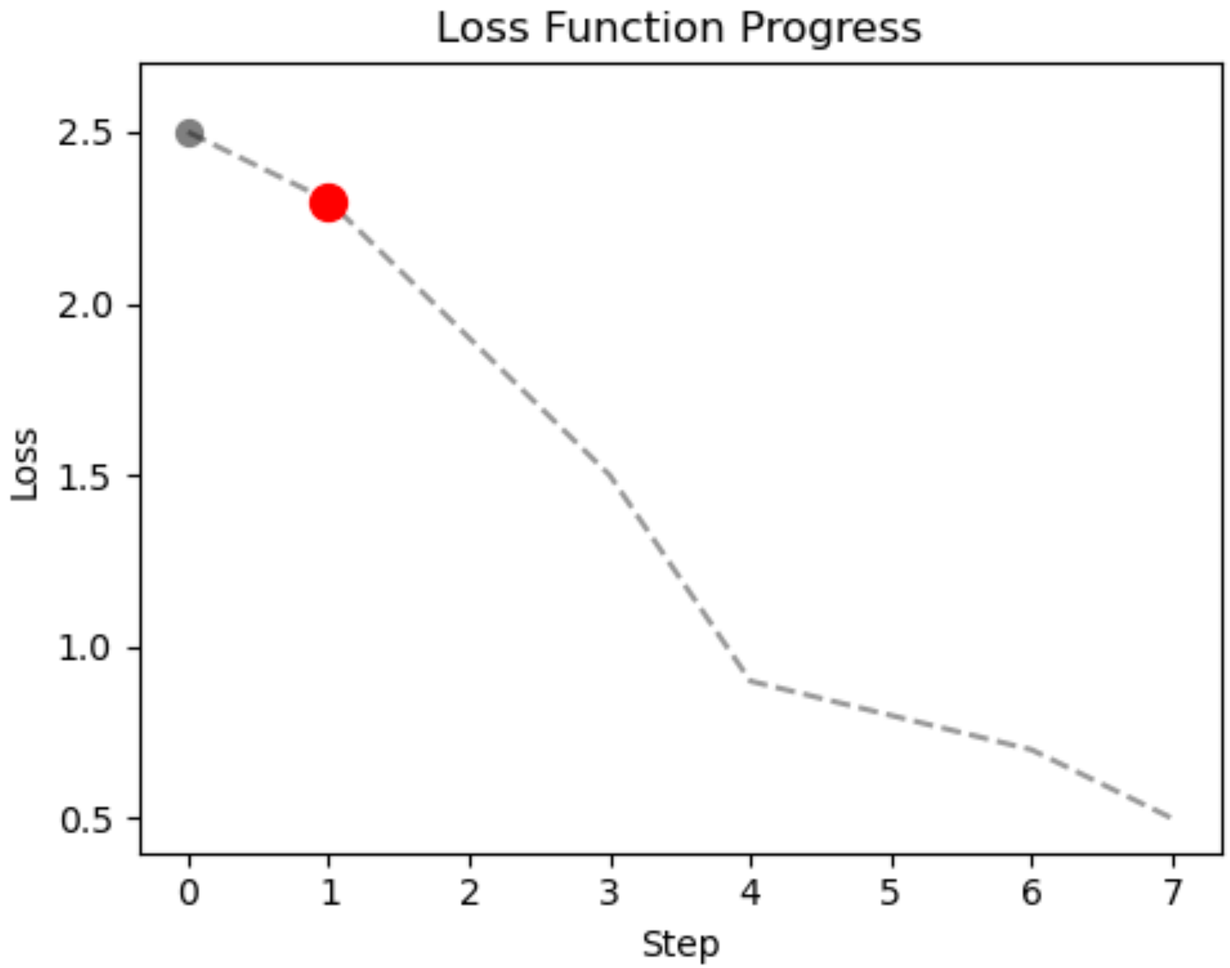
Redes Neuronales: Entrenamiento



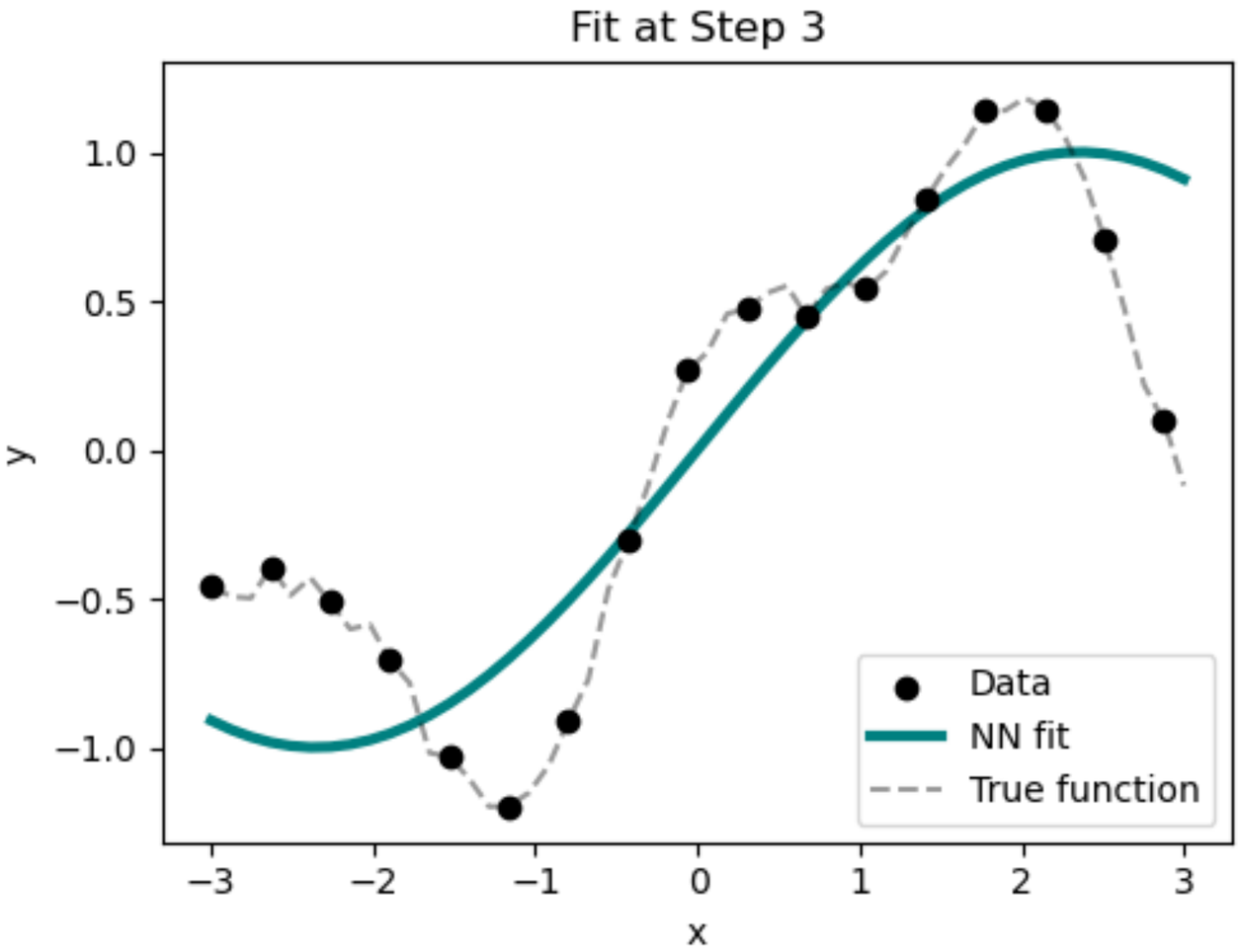


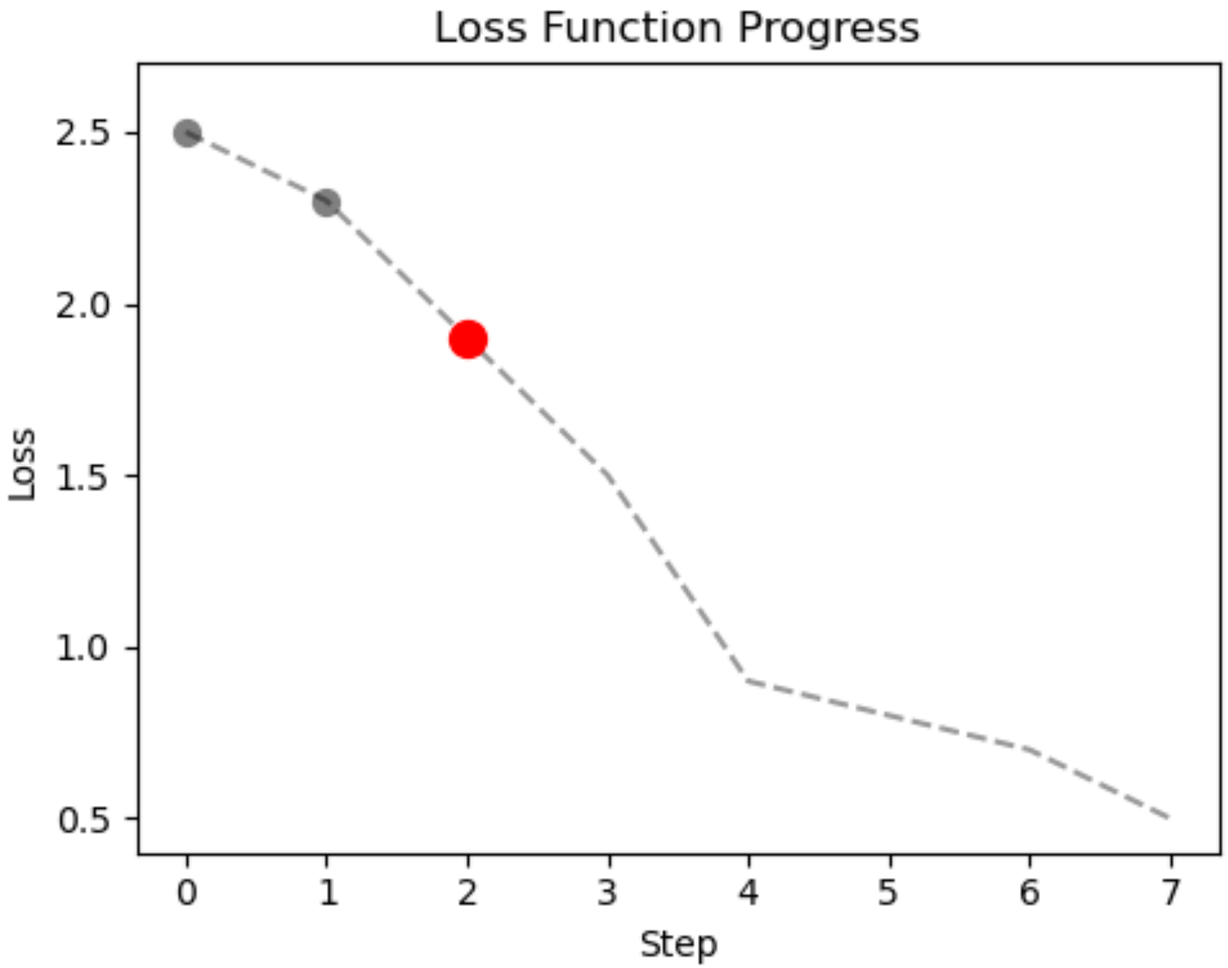
Redes Neuronales: Entrenamiento



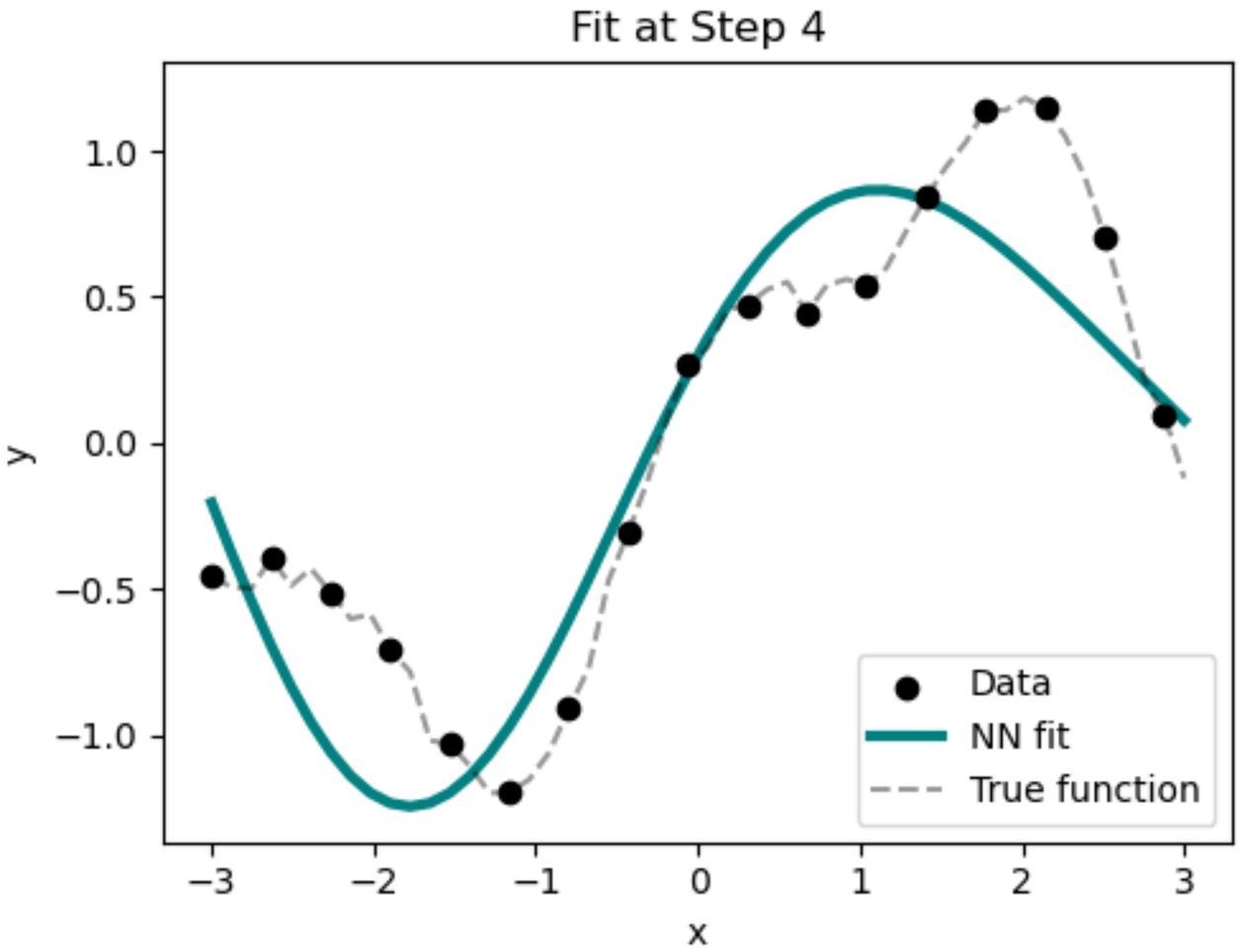


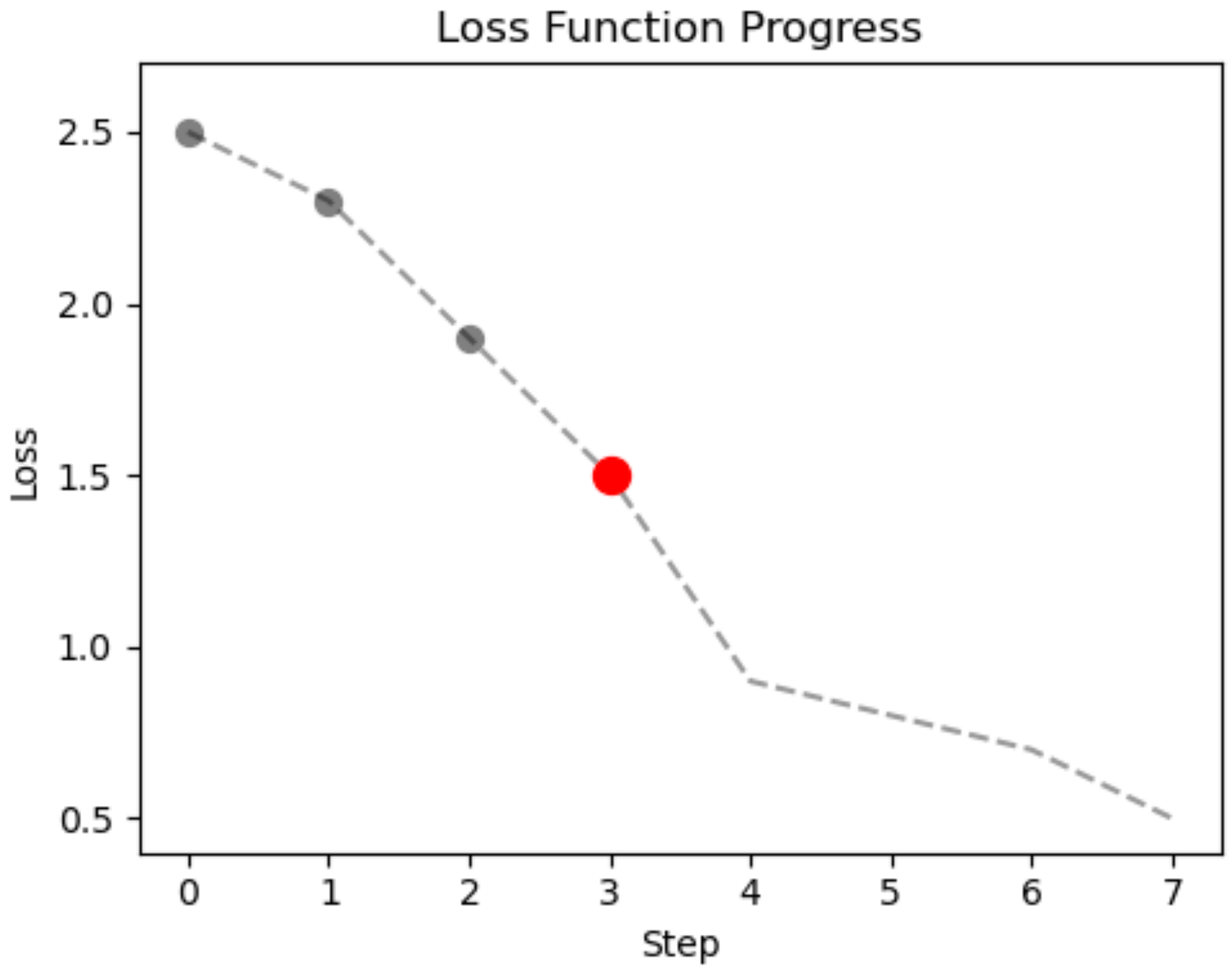
Redes Neuronales: Entrenamiento



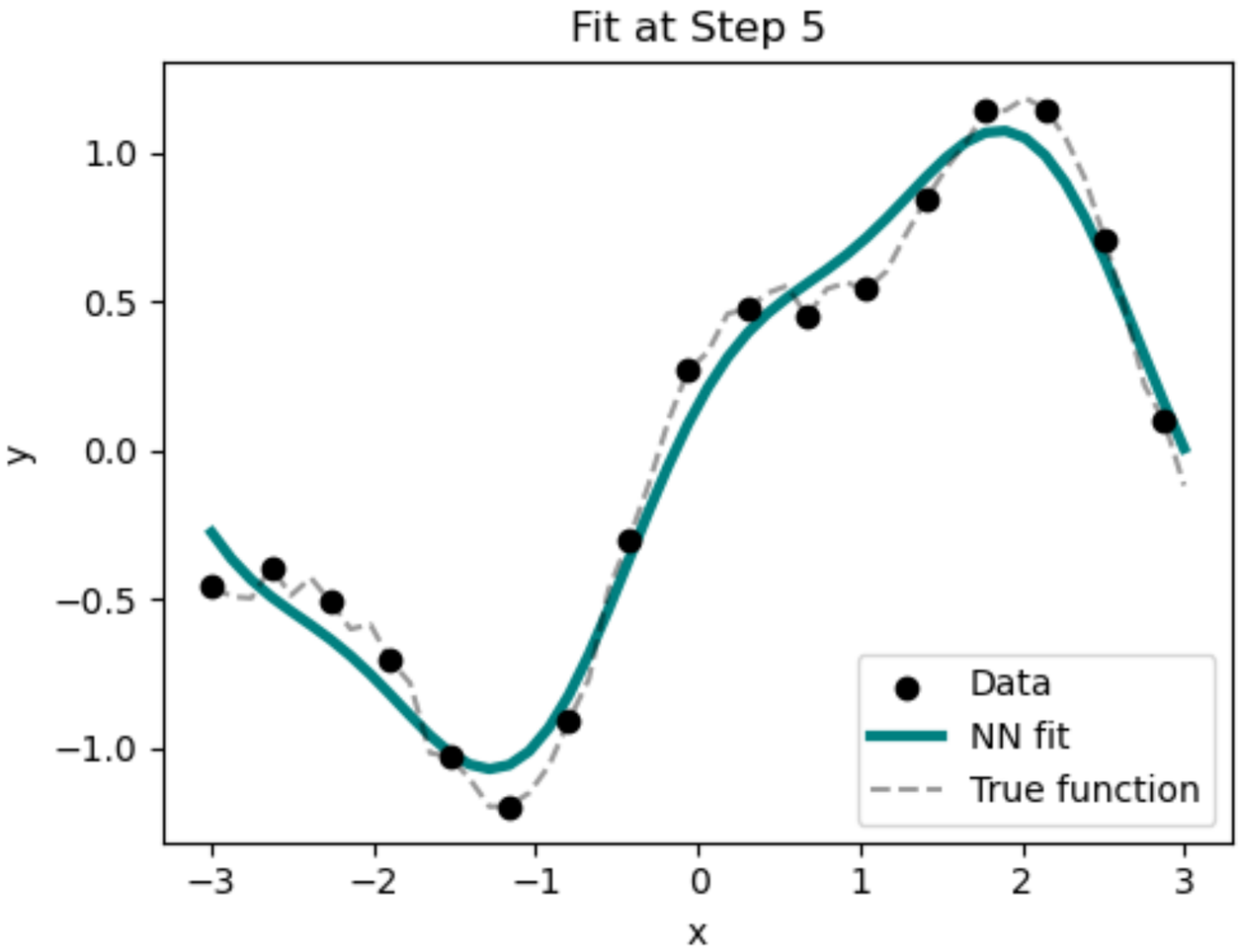


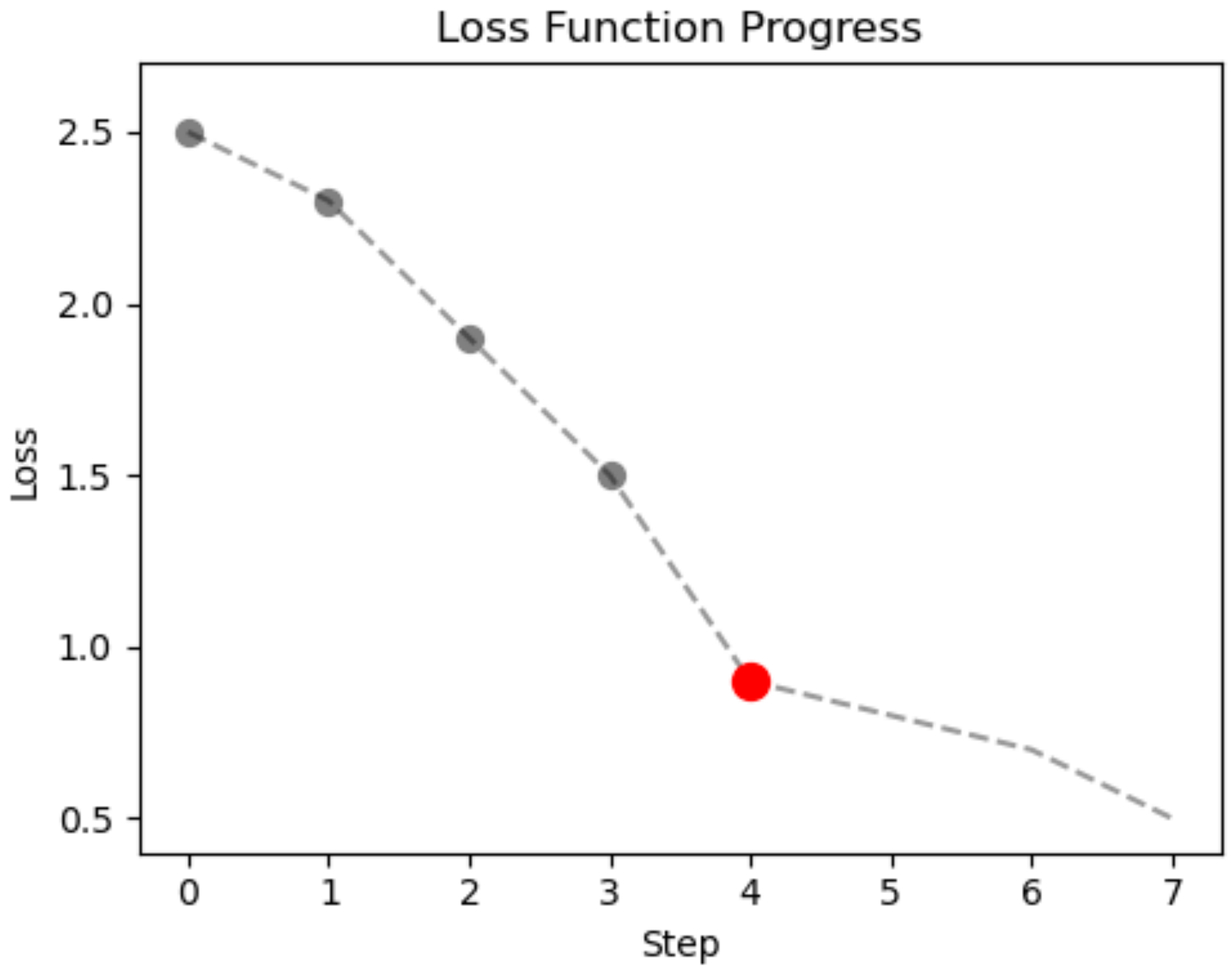
Redes Neuronales: Entrenamiento



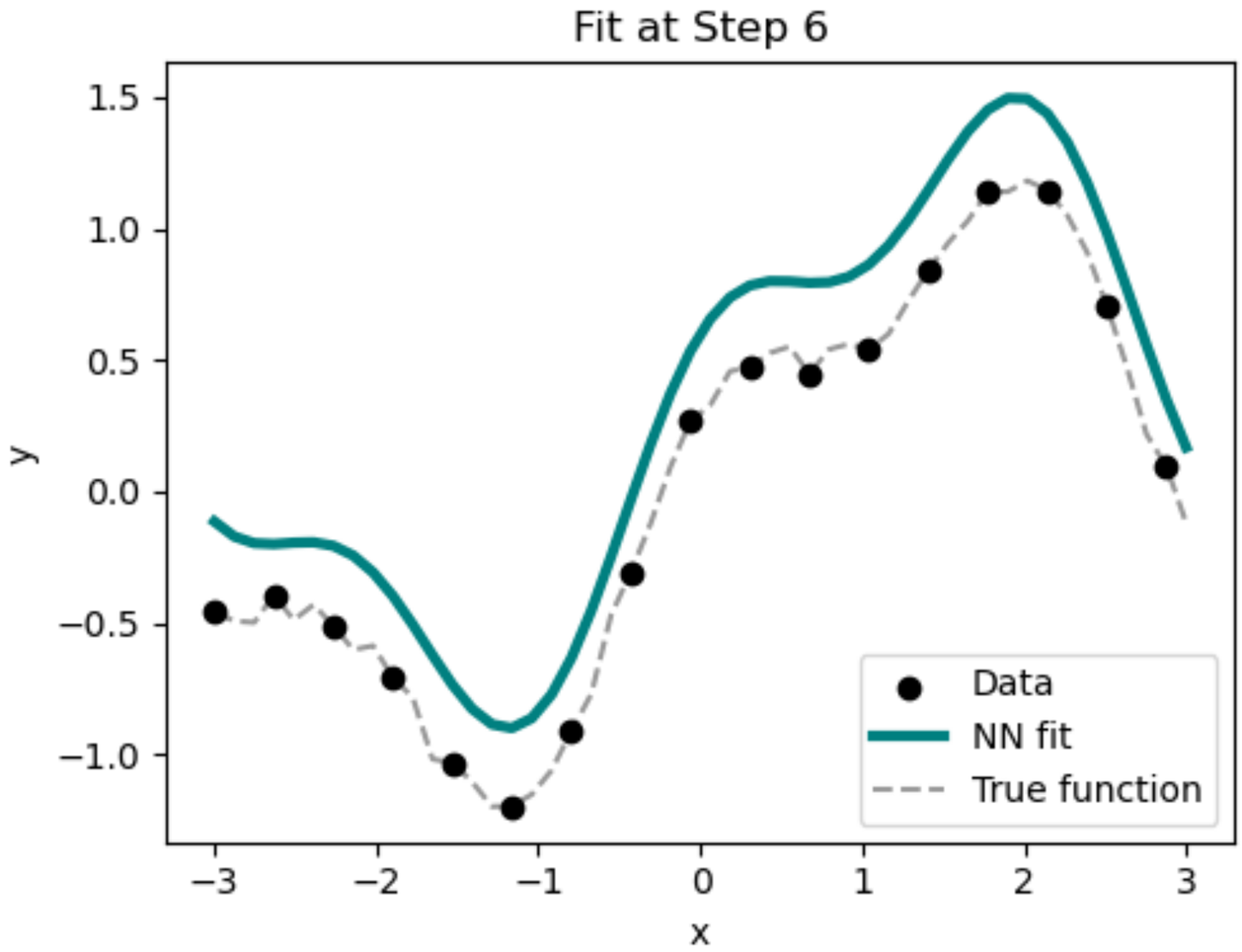


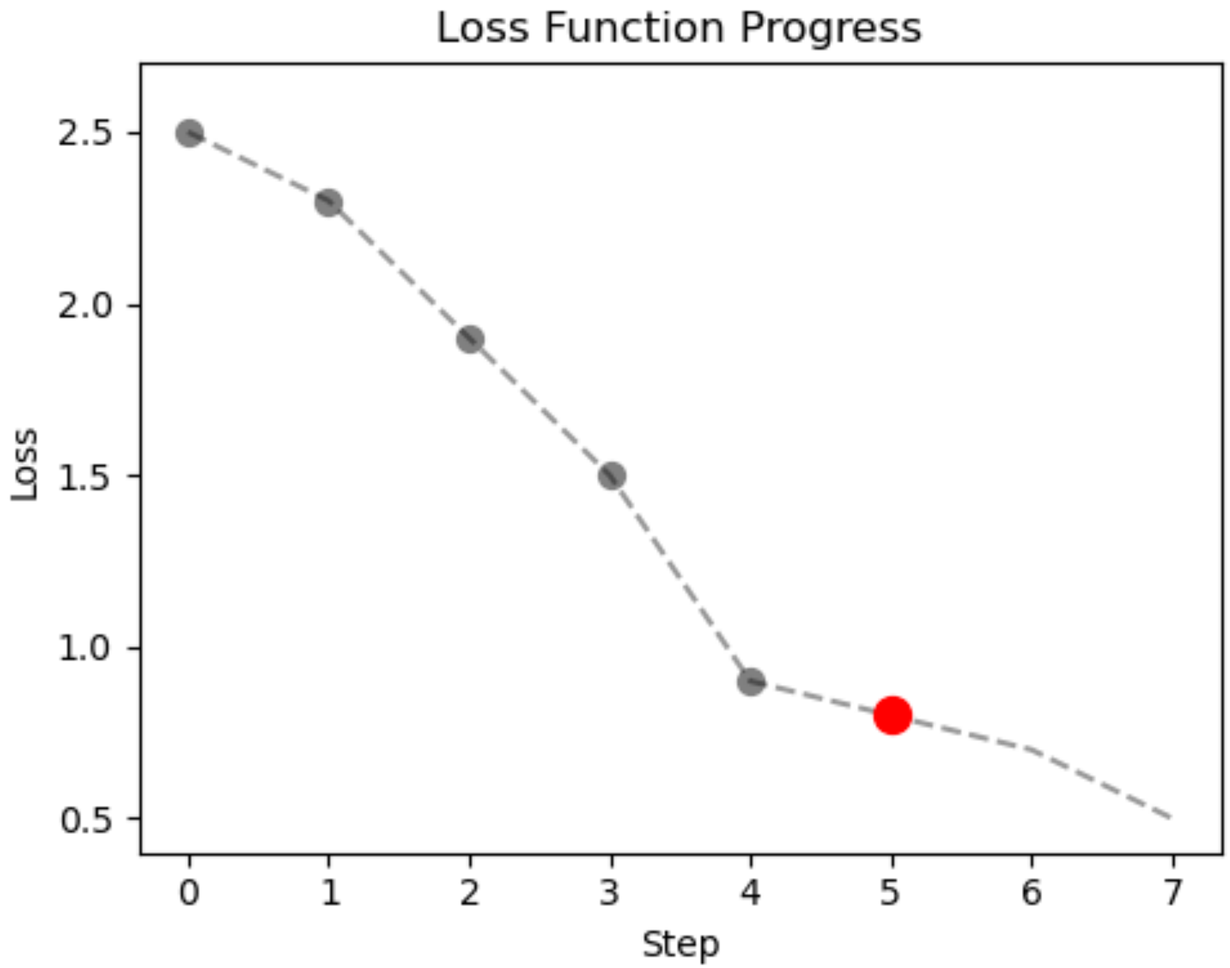
Redes Neuronales: Entrenamiento



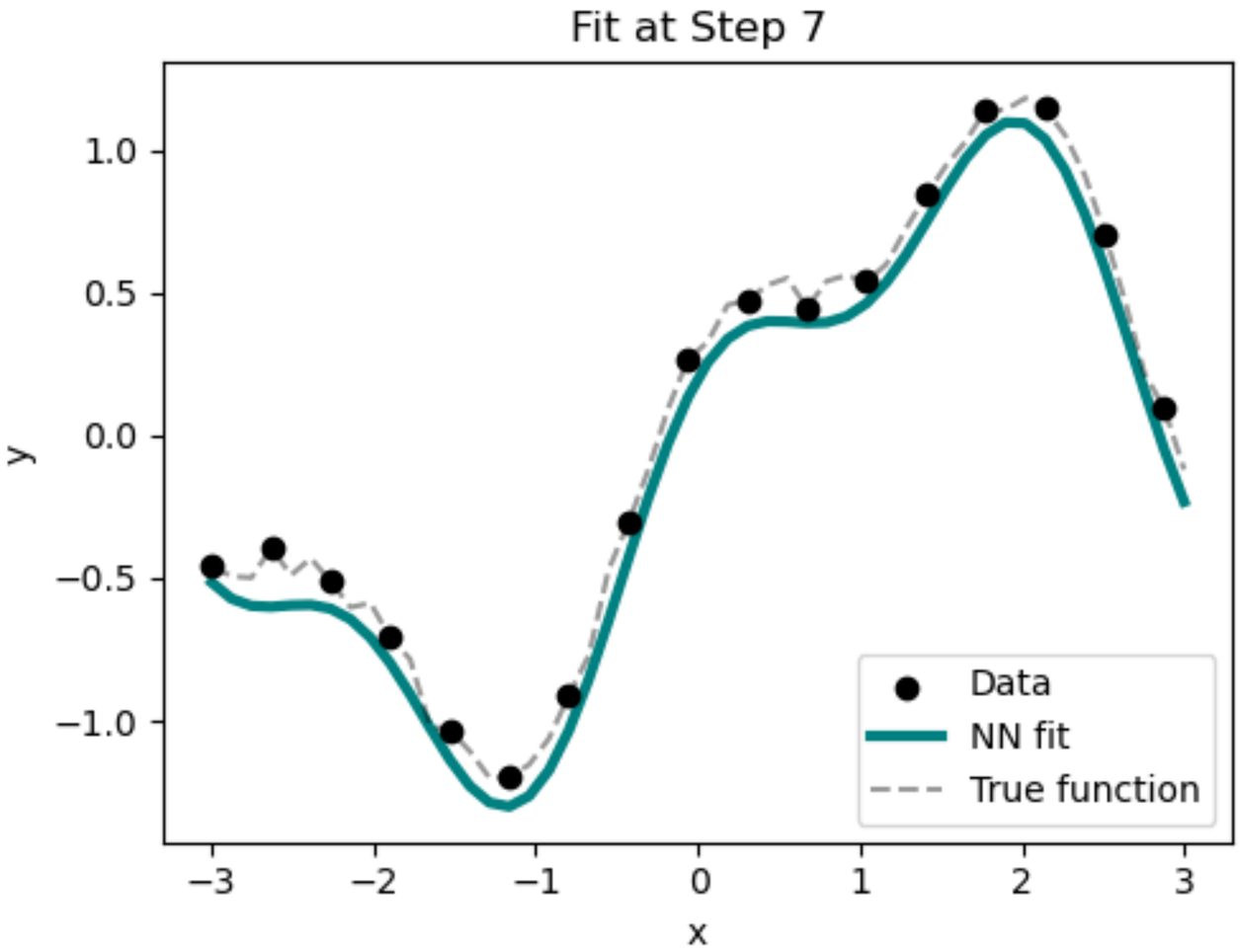


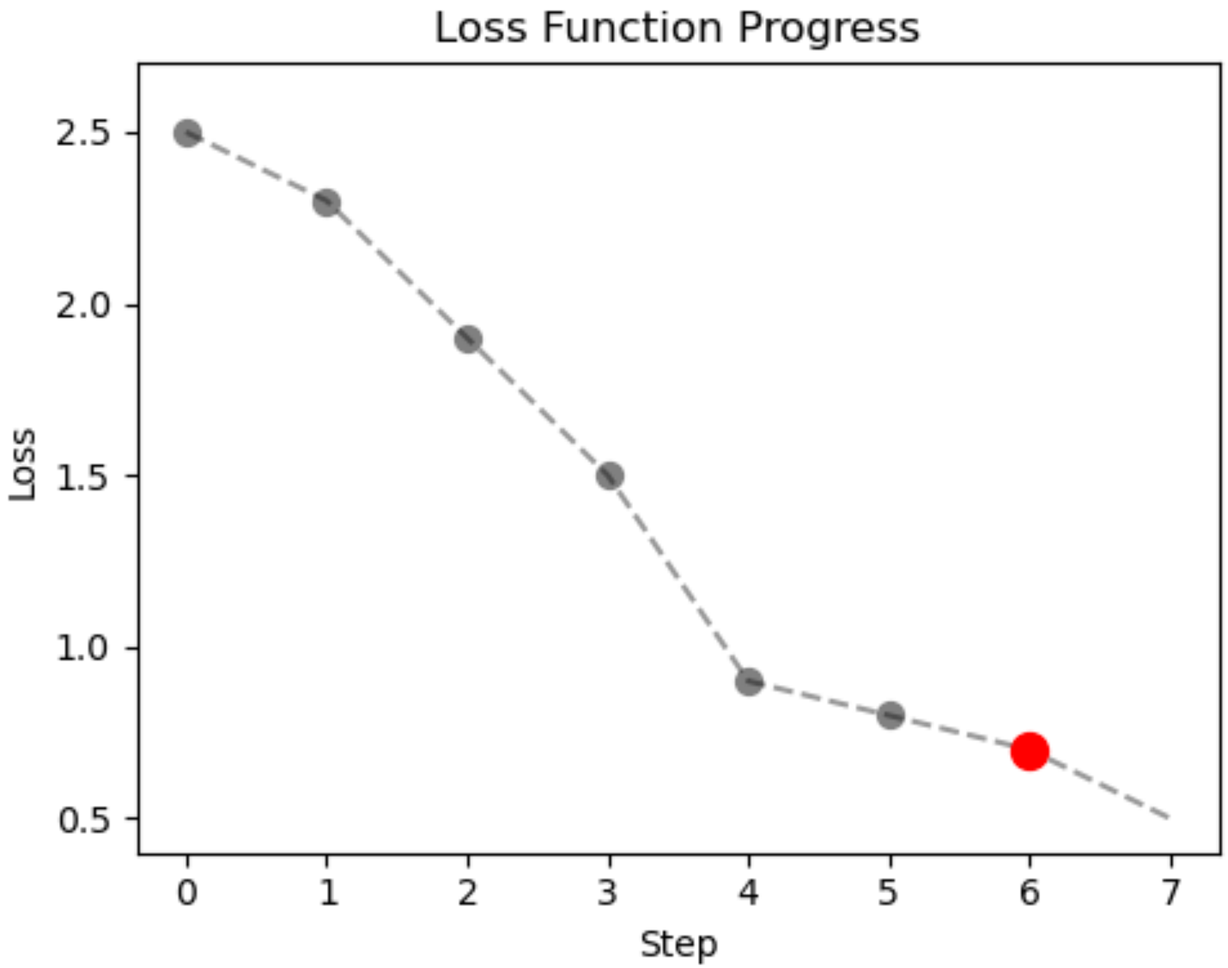
Redes Neuronales: Entrenamiento



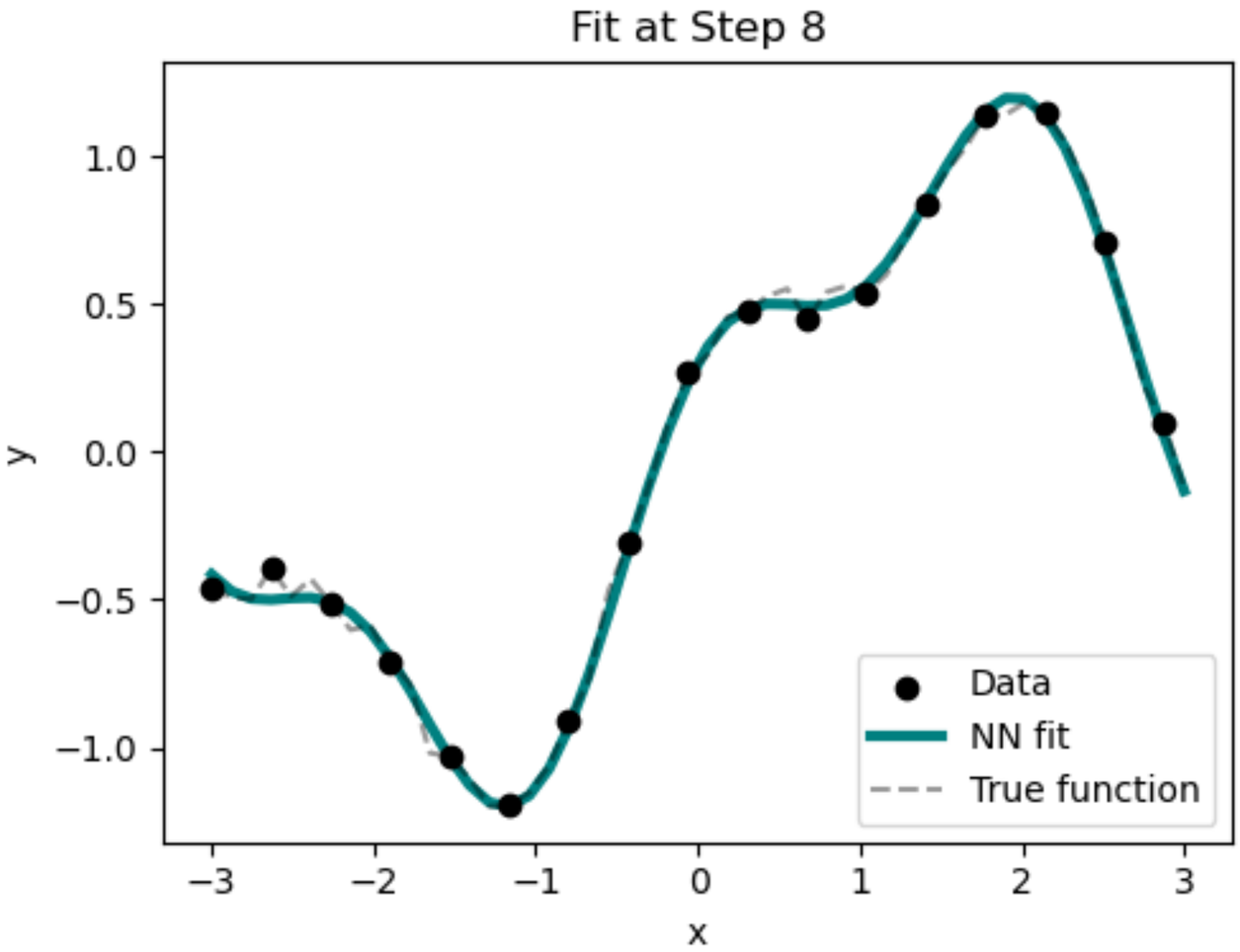


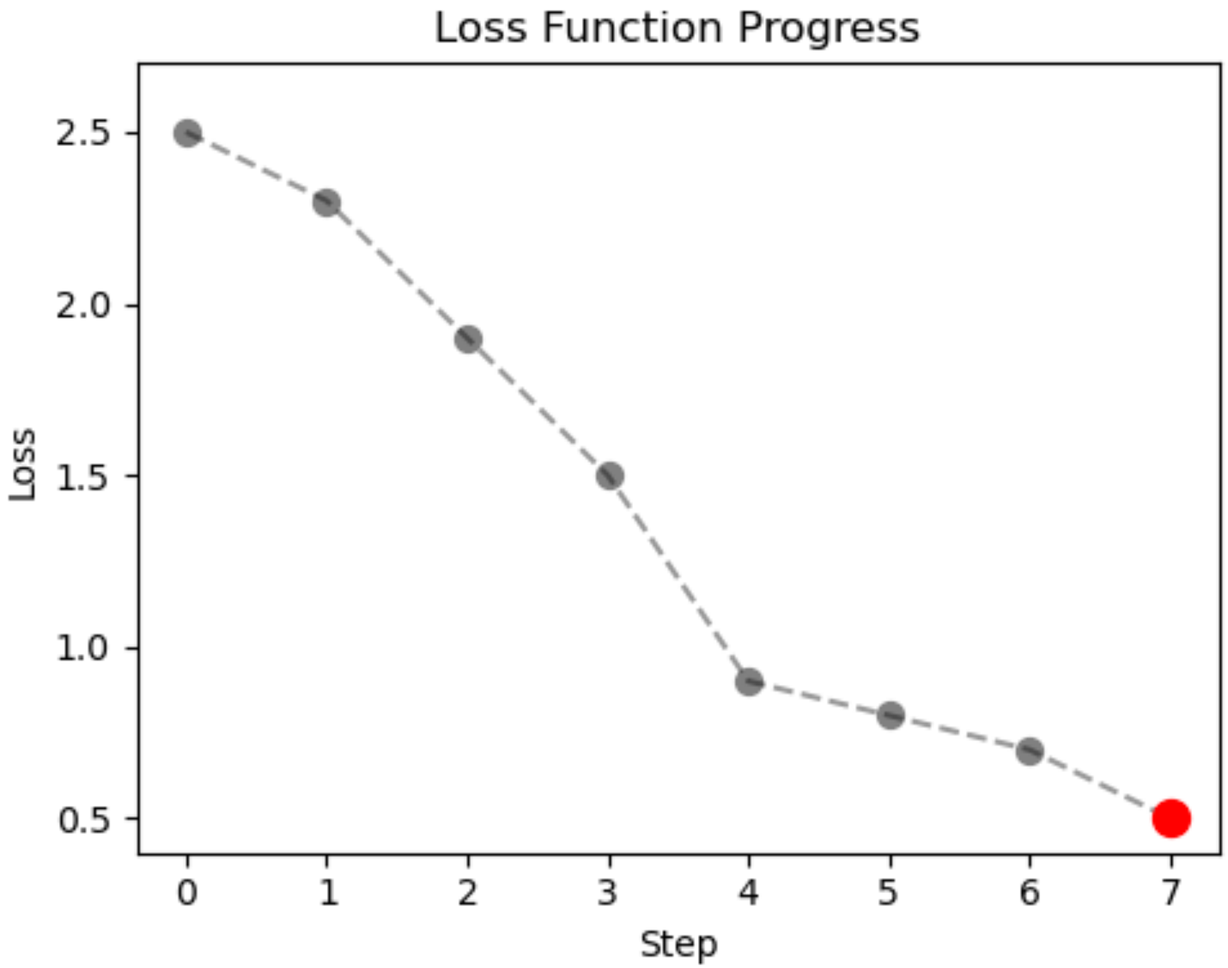
Redes Neuronales: Entrenamiento



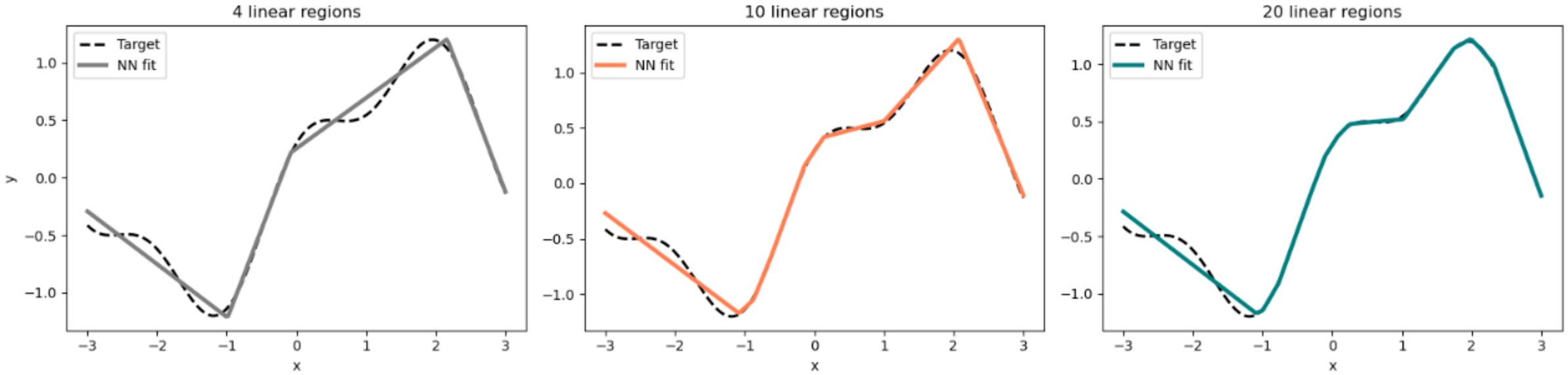


Redes Neuronales: Entrenamiento



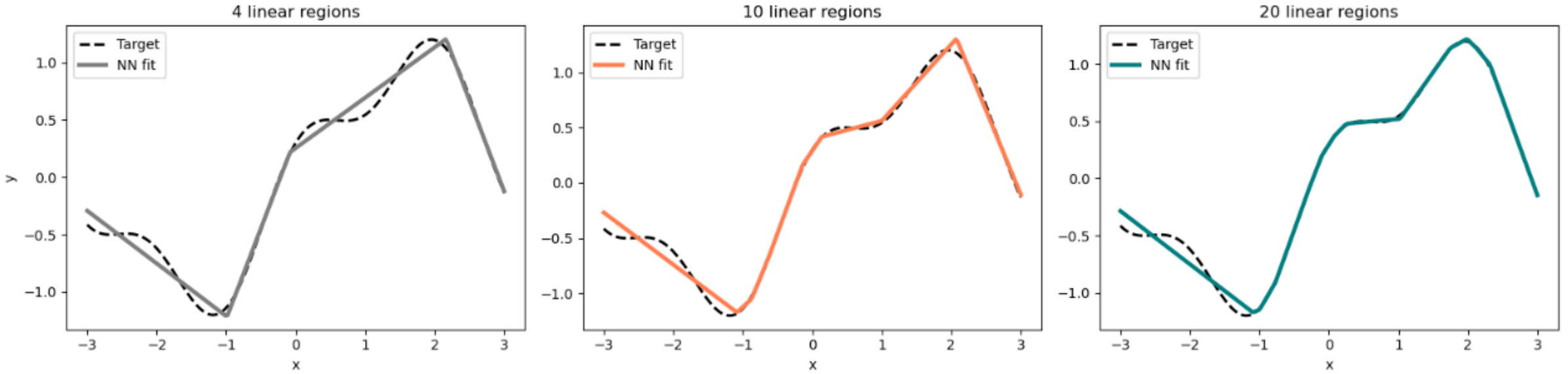


Redes Neuronales: Convergencia



Cuantas Neuronas Necesitamos?

Redes Neuronales: UAT



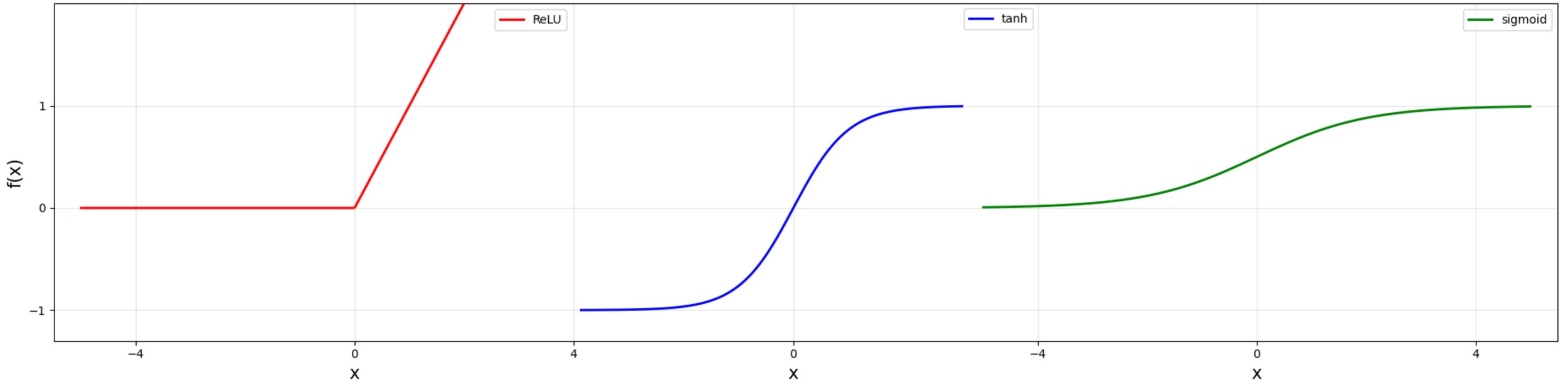
Universal Approximation Theorem

Una red neuronal de una sola capa oculta con un número finito de neuronas y una función de activación adecuada (como ReLU o sigmoide) puede aproximar cualquier función continua definida en un intervalo compacto de R^n con la precisión que se desee.

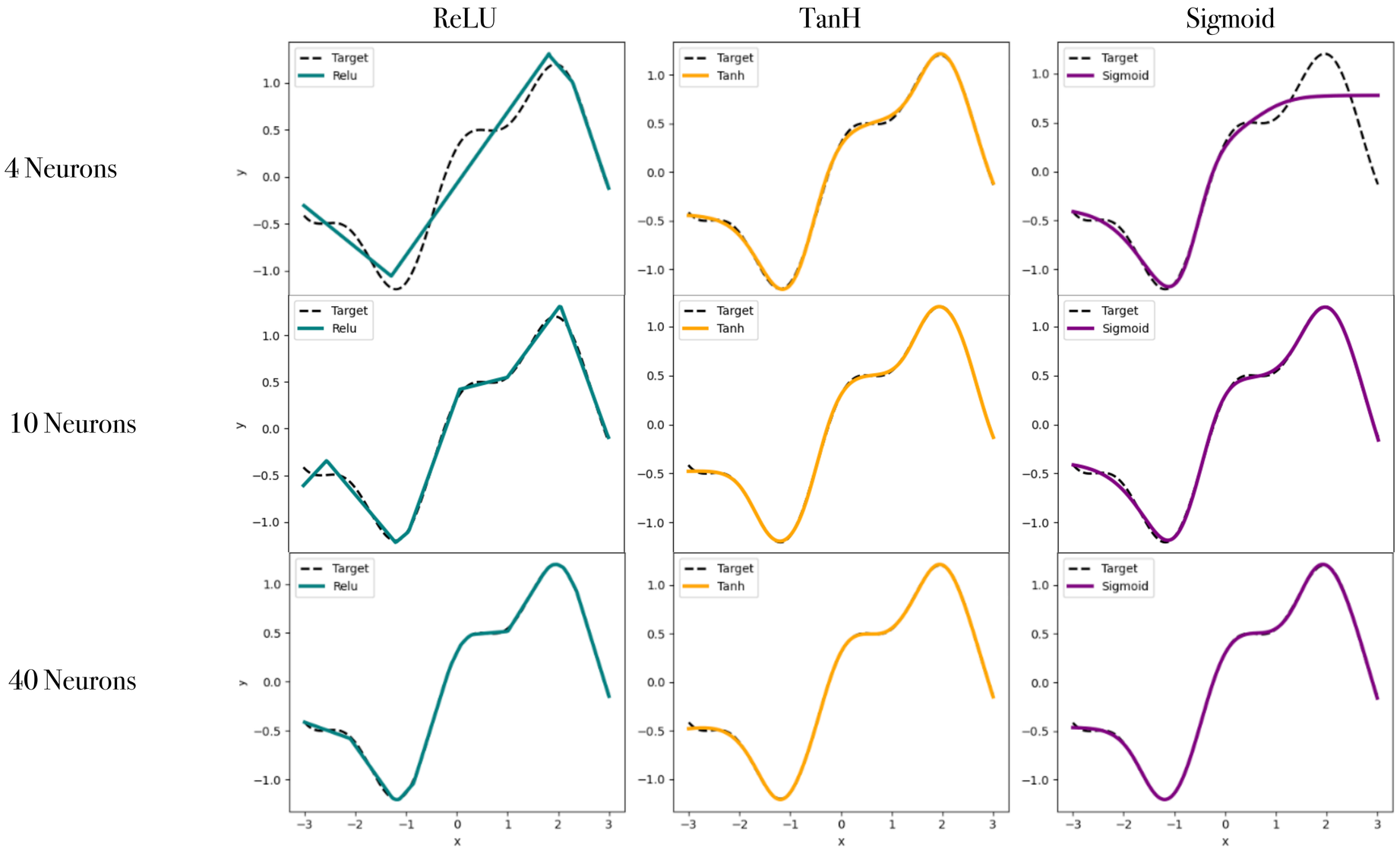
Nota: Se dice “función de activación adecuada” porque el teorema requiere que la activación sea no lineal, acotada o creciente, como la sigmoide, tanh o ReLU (no cualquier función arbitraria)

Redes Neuronales: Función de Activación

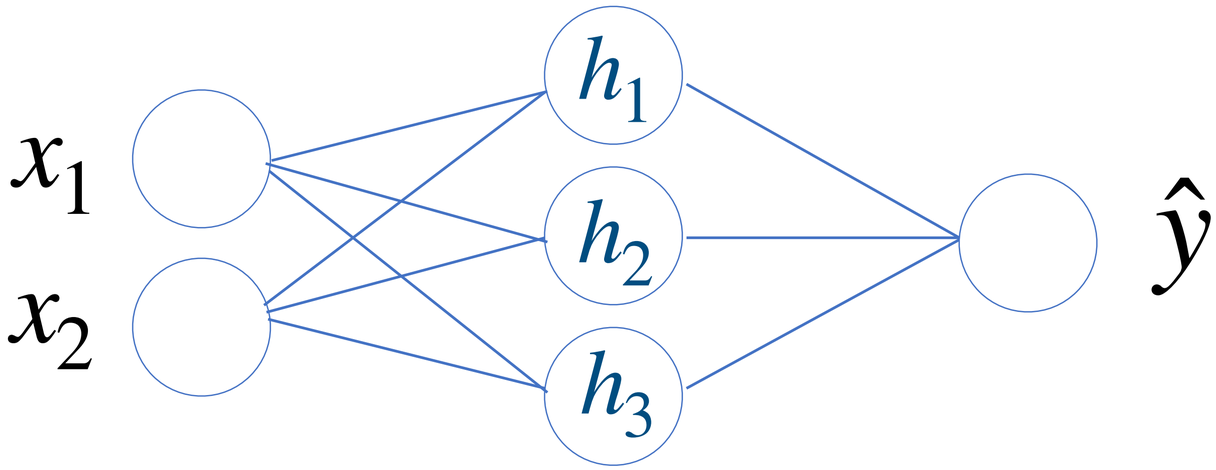
Otras funciones de activación



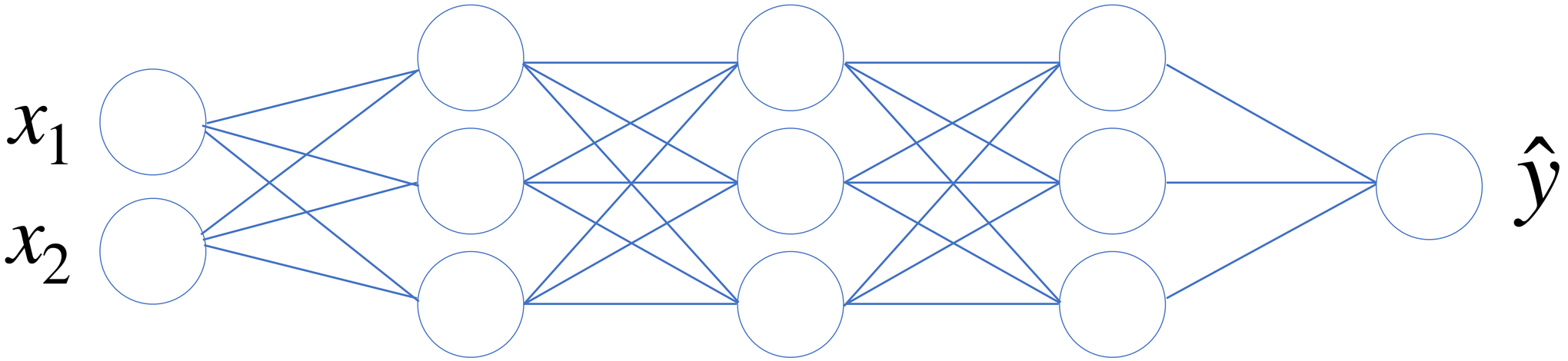
Redes Neuronales: Activation Functions



Redes Neuronales: Profundas



Redes Neuronales: Profundas



$a^{[l]}$ = activación de la capa l

$W^{[l]}$ = pesos de la capa l

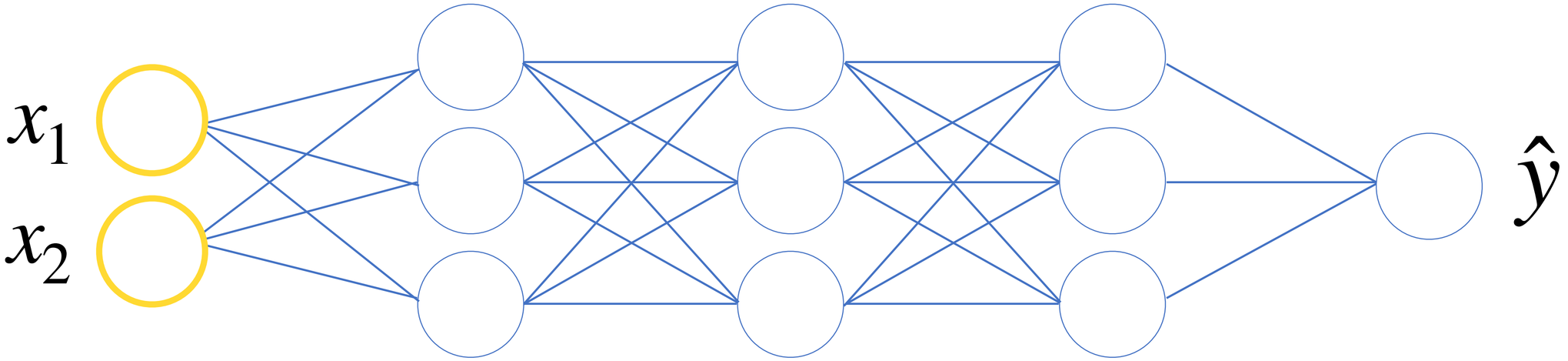
$b^{[l]}$ = sesgos (biases) de la capa l

σ = función de activación (por ejemplo: sigmoide, tanh, ReLU)

Redes Neuronales: Profundas

Forward Pass

$$a^{[l]} = \sigma\left(W^{[l]}a^{[l-1]} + b^{[l]}\right)$$



$a^{[l]}$ = activación de la capa l

$W^{[l]}$ = pesos de la capa l

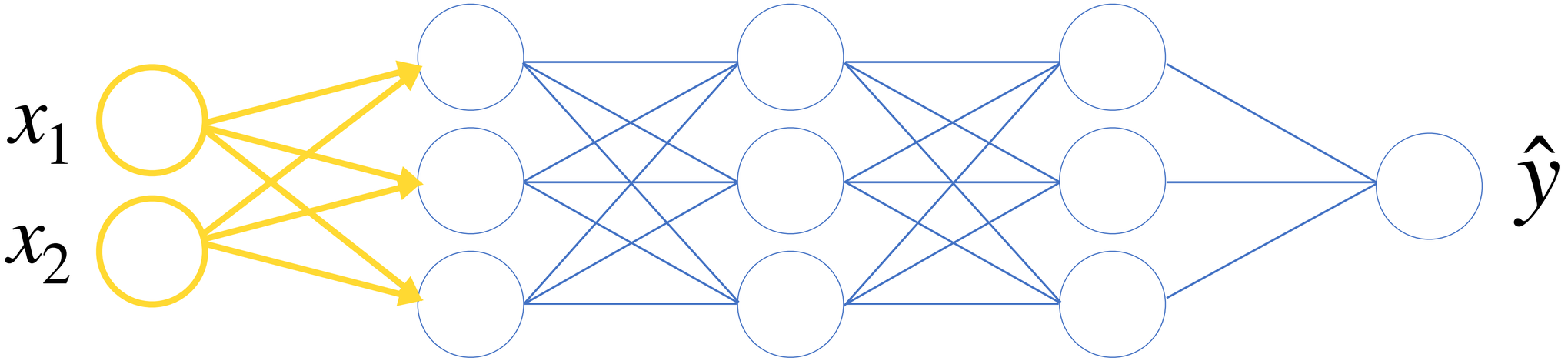
$b^{[l]}$ = sesgos (biases) de la capa l

σ = función de activación (por ejemplo: sigmoide, tanh, ReLU)

Redes Neuronales: Profundas

Forward Pass

$$a^{[l]} = \sigma\left(W^{[l]}a^{[l-1]} + b^{[l]}\right)$$



$a^{[l]}$ = activación de la capa l

$W^{[l]}$ = pesos de la capa l

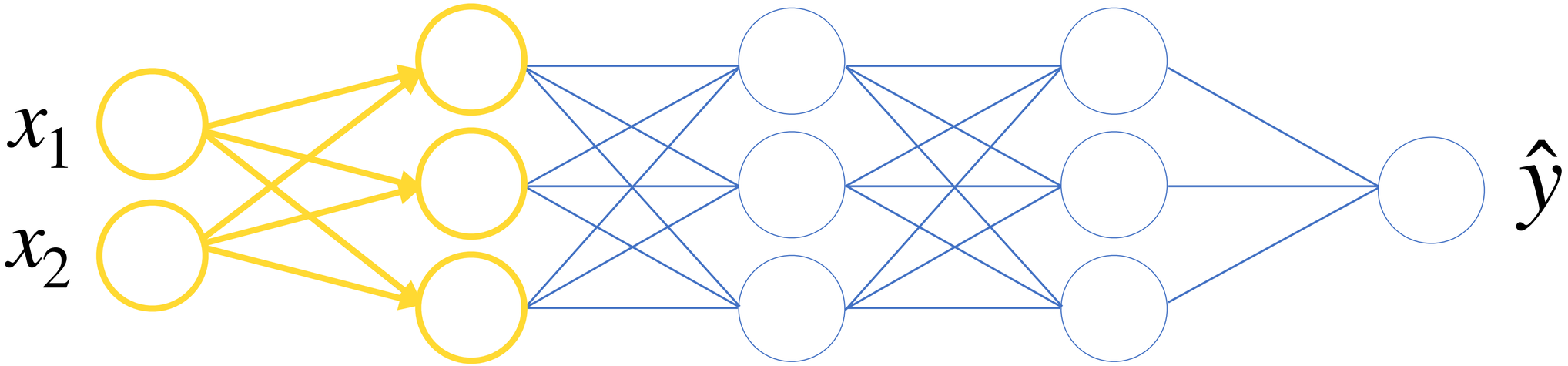
$b^{[l]}$ = sesgos (biases) de la capa l

σ = función de activación (por ejemplo: sigmoide, tanh, ReLU)

Redes Neuronales: Profundas

Forward Pass

$$a^{[l]} = \sigma\left(W^{[l]}a^{[l-1]} + b^{[l]}\right)$$



$a^{[l]}$ = activación de la capa l

$W^{[l]}$ = pesos de la capa l

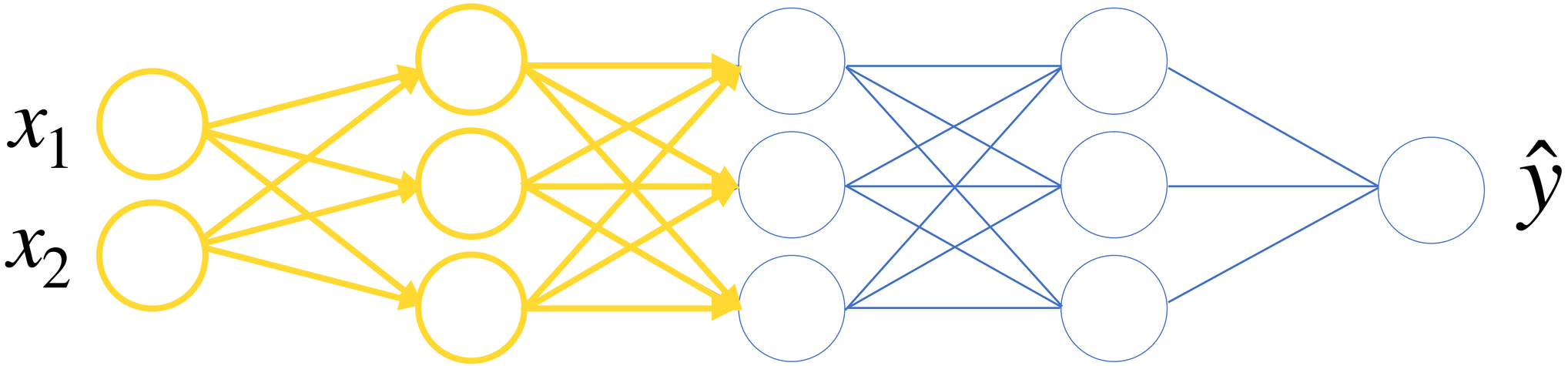
$b^{[l]}$ = sesgos (biases) de la capa l

σ = función de activación (por ejemplo: sigmoide, tanh, ReLU)

Redes Neuronales: Profundas

Forward Pass

$$a^{[l]} = \sigma\left(W^{[l]}a^{[l-1]} + b^{[l]}\right)$$



$a^{[l]}$ = activación de la capa l

$W^{[l]}$ = pesos de la capa l

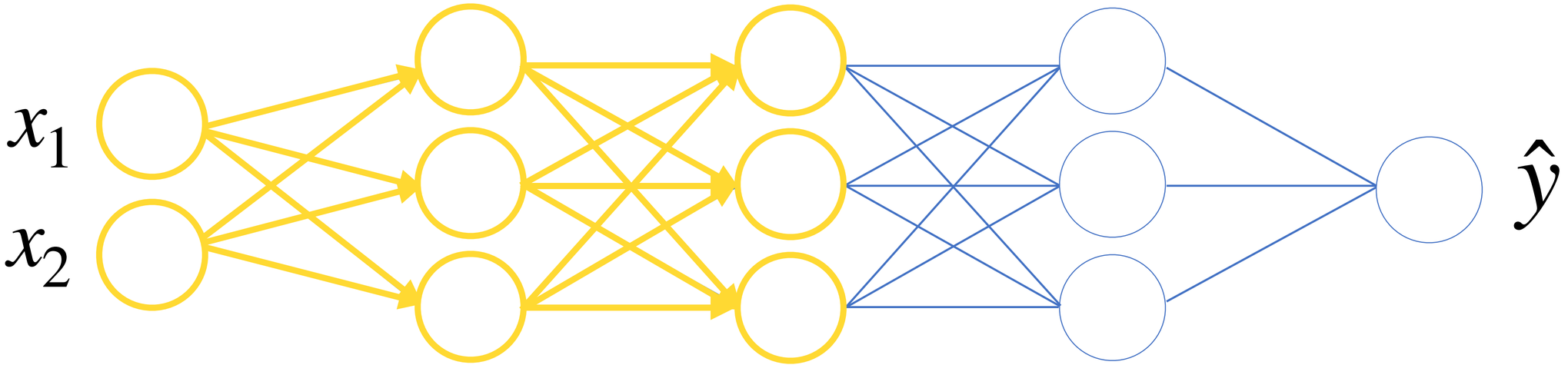
$b^{[l]}$ = sesgos (biases) de la capa l

σ = función de activación (por ejemplo: sigmoide, tanh, ReLU)

Redes Neuronales: Profundas

Forward Pass

$$a^{[l]} = \sigma\left(W^{[l]}a^{[l-1]} + b^{[l]}\right)$$



$a^{[l]}$ = activación de la capa l

$W^{[l]}$ = pesos de la capa l

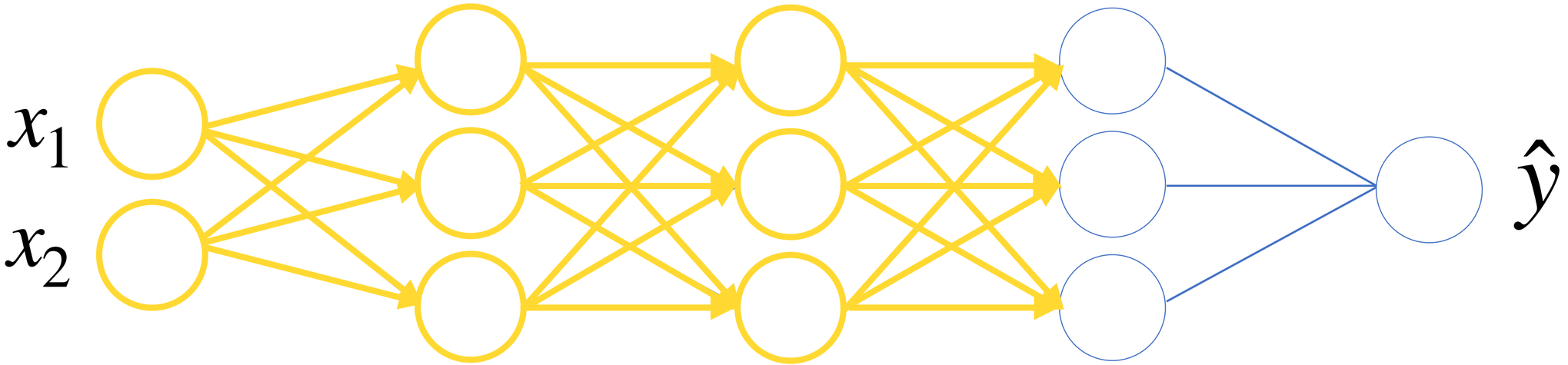
$b^{[l]}$ = sesgos (biases) de la capa l

σ = función de activación (por ejemplo: sigmoide, tanh, ReLU)

Redes Neuronales: Profundas

Forward Pass

$$a^{[l]} = \sigma\left(W^{[l]}a^{[l-1]} + b^{[l]}\right)$$



$a^{[l]}$ = activación de la capa l

$W^{[l]}$ = pesos de la capa l

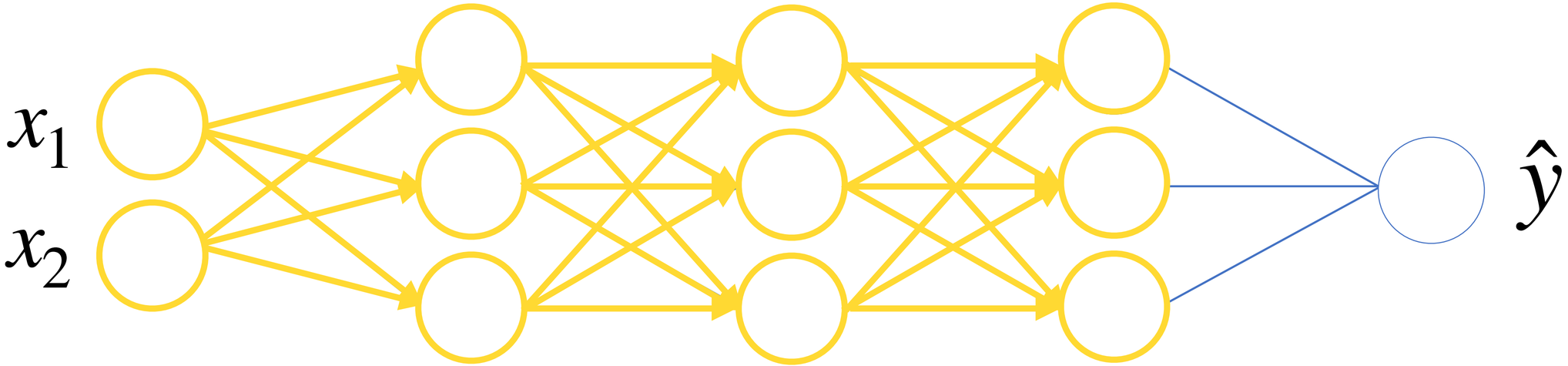
$b^{[l]}$ = sesgos (biases) de la capa l

σ = función de activación (por ejemplo: sigmoide, tanh, ReLU)

Redes Neuronales: Profundas

Forward Pass

$$a^{[l]} = \sigma\left(W^{[l]}a^{[l-1]} + b^{[l]}\right)$$



$a^{[l]}$ = activación de la capa l

$W^{[l]}$ = pesos de la capa l

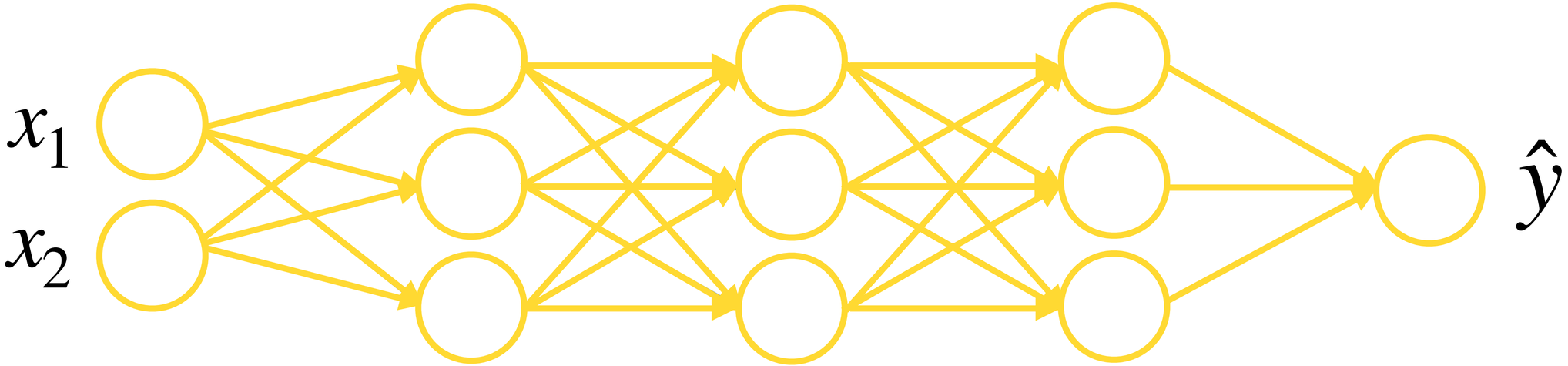
$b^{[l]}$ = sesgos (biases) de la capa l

σ = función de activación (por ejemplo: sigmoide, tanh, ReLU)

Redes Neuronales: Profundas

Forward Pass

$$a^{[l]} = \sigma\left(W^{[l]}a^{[l-1]} + b^{[l]}\right)$$



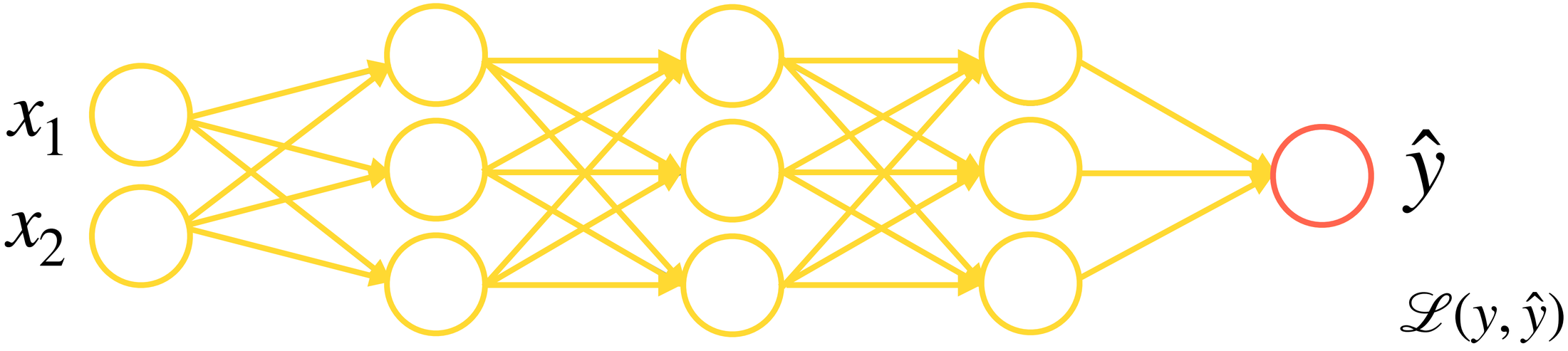
$a^{[l]}$ = activación de la capa l

$W^{[l]}$ = pesos de la capa l

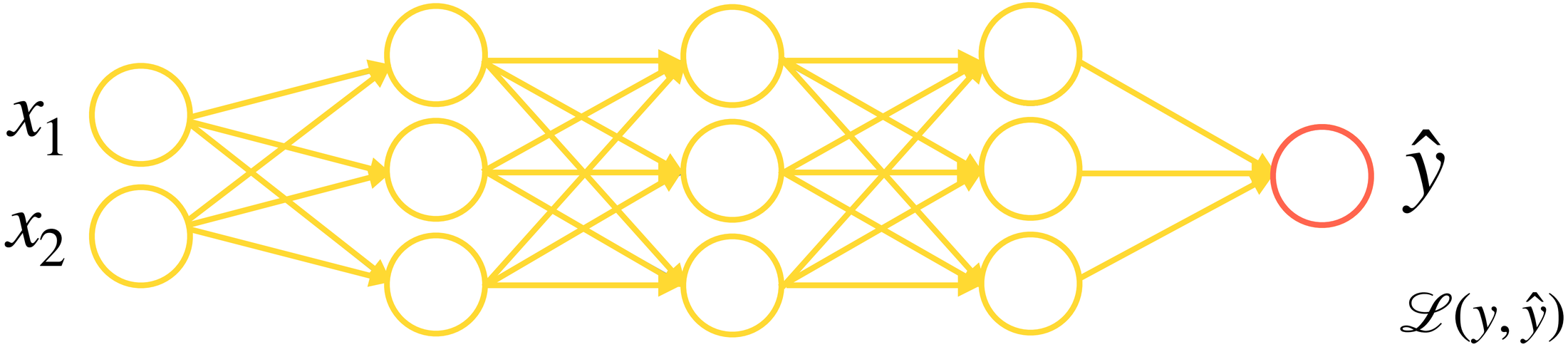
$b^{[l]}$ = sesgos (biases) de la capa l

σ = función de activación (por ejemplo: sigmoide, tanh, ReLU)

Redes Neuronales: Profundas

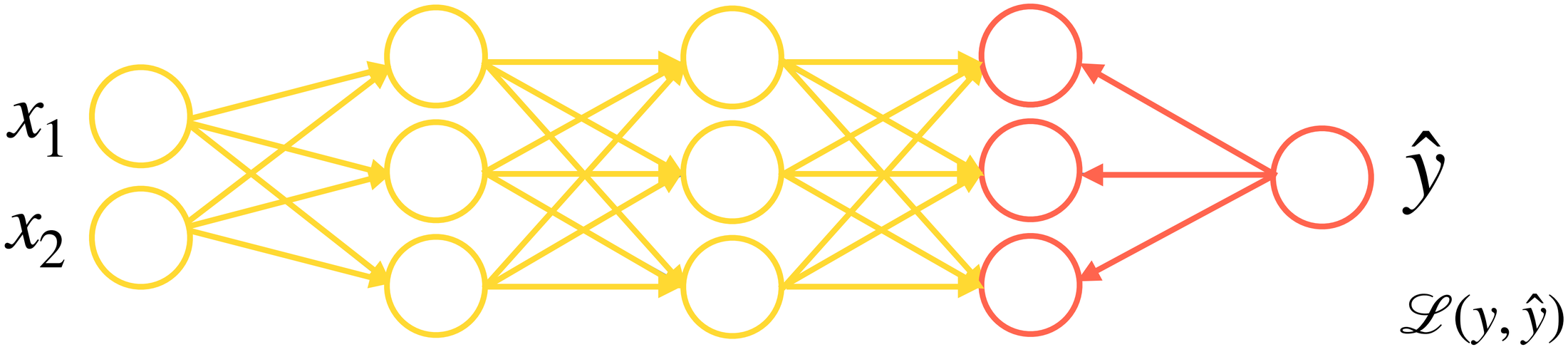


Redes Neuronales: Profundas



$$\nabla_\theta \mathscr{L}(\theta) = \frac{\partial \mathscr{L}(\theta)}{\partial \theta} \qquad \theta_{new} = \theta_{old} - \eta \nabla_\theta \mathscr{L}(\theta_{old})$$

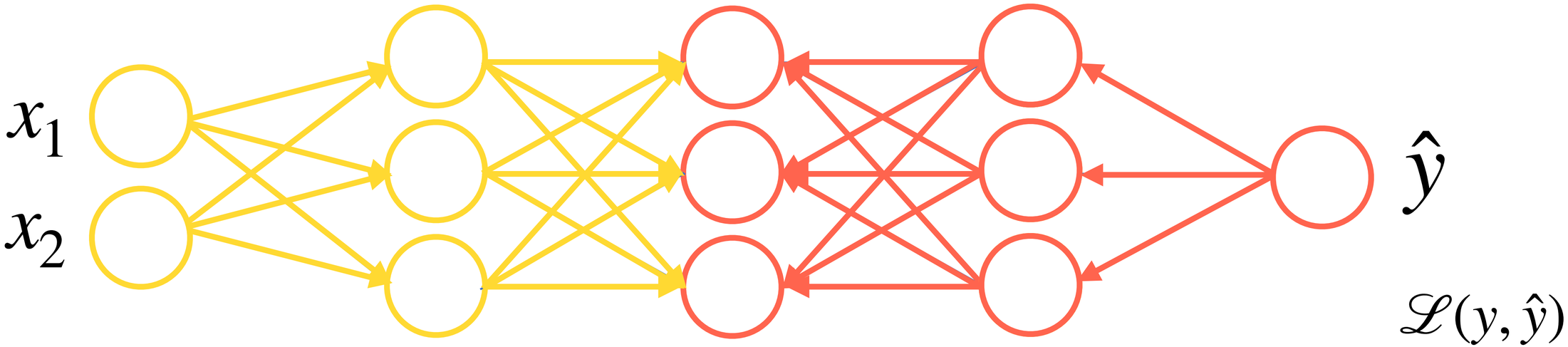
Redes Neuronales: Profundas



Back Propagate

$$\nabla_\theta \mathscr{L}(\theta) = \frac{\partial \mathscr{L}(\theta)}{\partial \theta} \qquad \theta_{new} = \theta_{old} - \eta \nabla_\theta \mathscr{L}(\theta_{old})$$

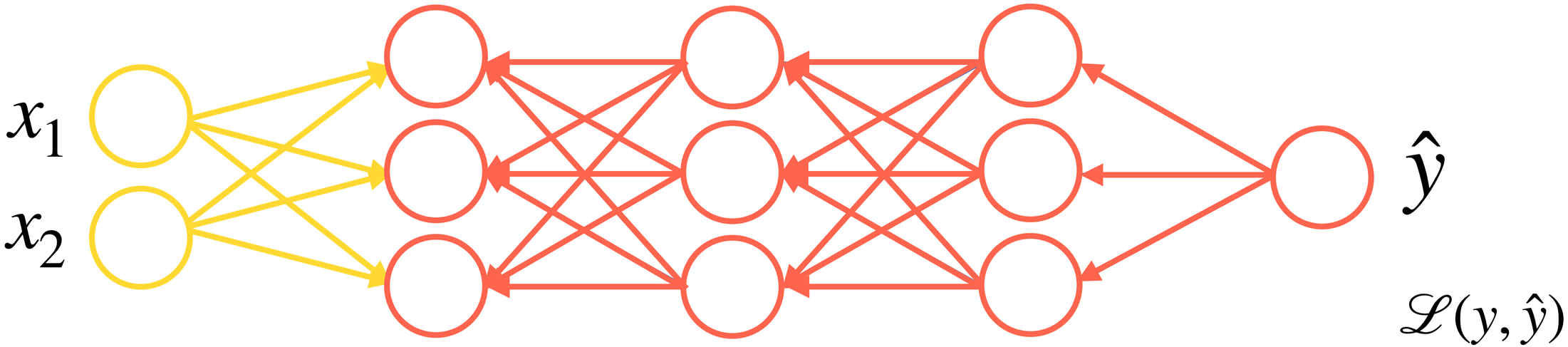
Redes Neuronales: Profundas



Back Propagate

$$\nabla_\theta \mathscr{L}(\theta) = \frac{\partial \mathscr{L}(\theta)}{\partial \theta} \qquad \theta_{new} = \theta_{old} - \eta \nabla_\theta \mathscr{L}(\theta_{old})$$

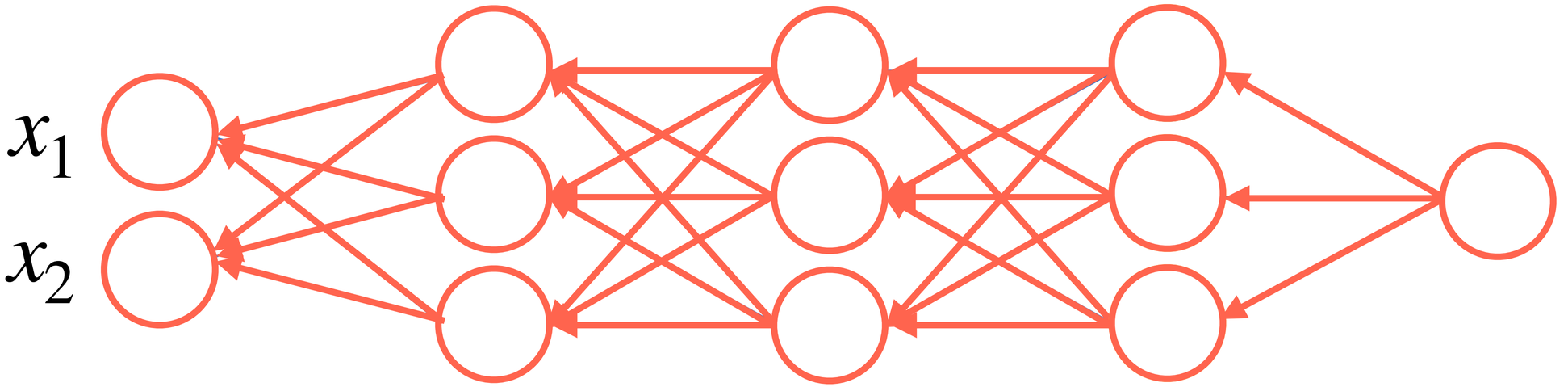
Redes Neuronales: Profundas



Back Propagate

$$\nabla_\theta \mathscr{L}(\theta) = \frac{\partial \mathscr{L}(\theta)}{\partial \theta} \qquad \theta_{new} = \theta_{old} - \eta \nabla_\theta \mathscr{L}(\theta_{old})$$

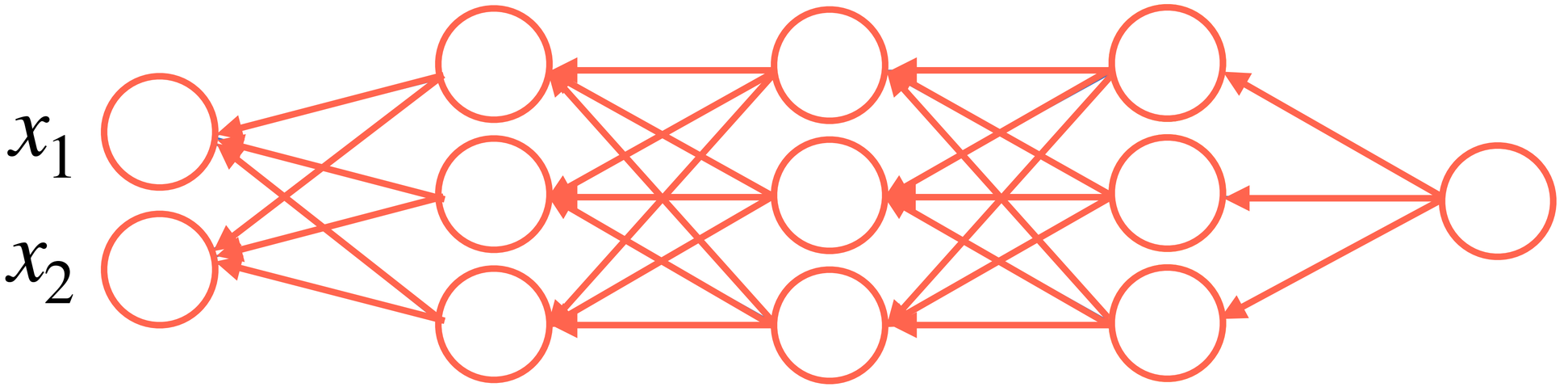
Redes Neuronales: Profundas



Back Propagate

$$\nabla_\theta \mathscr{L}(\theta) = \frac{\partial \mathscr{L}(\theta)}{\partial \theta} \qquad \theta_{new} = \theta_{old} - \eta \nabla_\theta \mathscr{L}(\theta_{old})$$

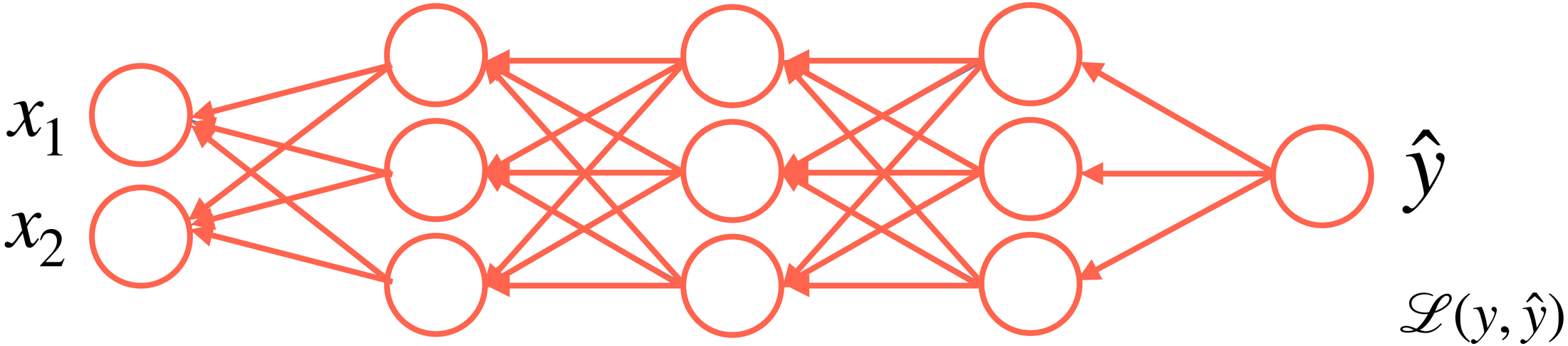
Redes Neuronales: Profundas



Back Propagate

$$\nabla_\theta \mathscr{L}(\theta) = \frac{\partial \mathscr{L}(\theta)}{\partial \theta} \qquad \theta_{new} = \theta_{old} - \eta \nabla_\theta \mathscr{L}(\theta_{old})$$

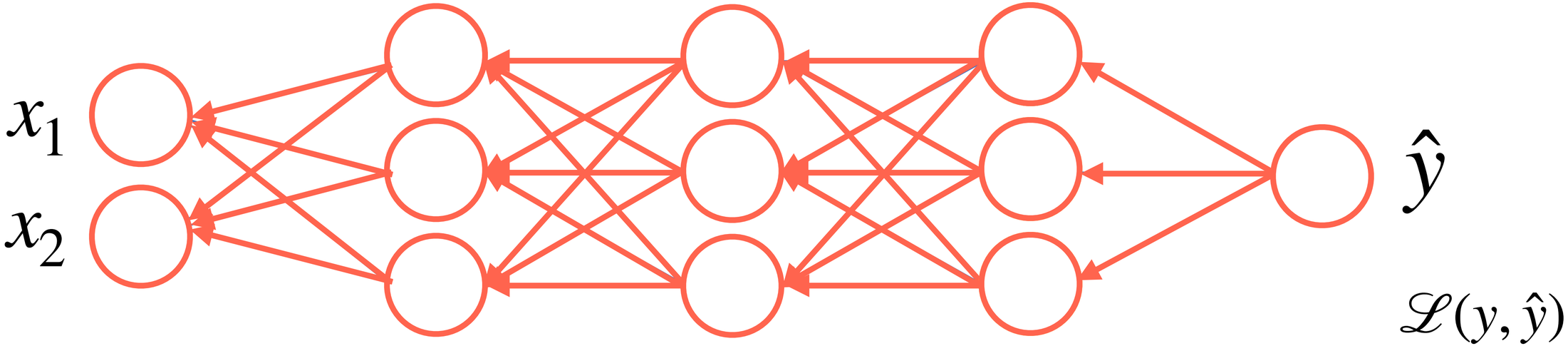
Redes Neuronales: Profundas



Back Propagate

$$\frac{\partial \mathscr{L}}{\partial \theta} = \frac{\partial \mathscr{L}}{\partial a_n} \cdot \frac{\partial a_n}{\partial a_{n-1}} \cdot \frac{\partial a_{n-1}}{\partial a_{n-2}} \cdots \frac{\partial a_0}{\partial \theta}$$

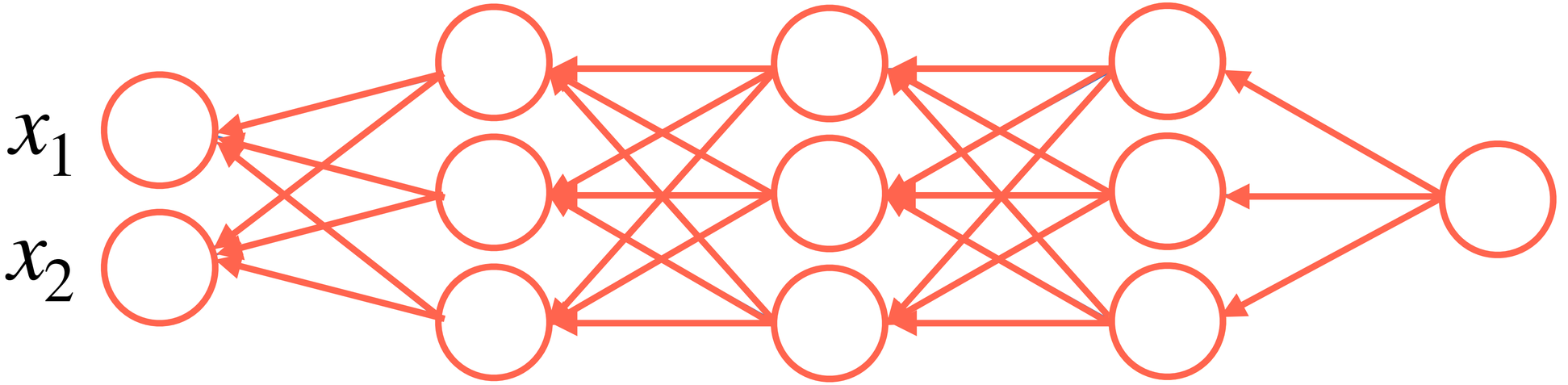
Redes Neuronales: Profundas



Back Propagate

$$\frac{\partial \mathscr{L}}{\partial \theta} = \frac{\partial \mathscr{L}}{\partial a_n} \prod_{k=1}^{n} \frac{\partial a_k}{\partial a_{k-1}} \frac{\partial a_0}{\partial \theta}$$

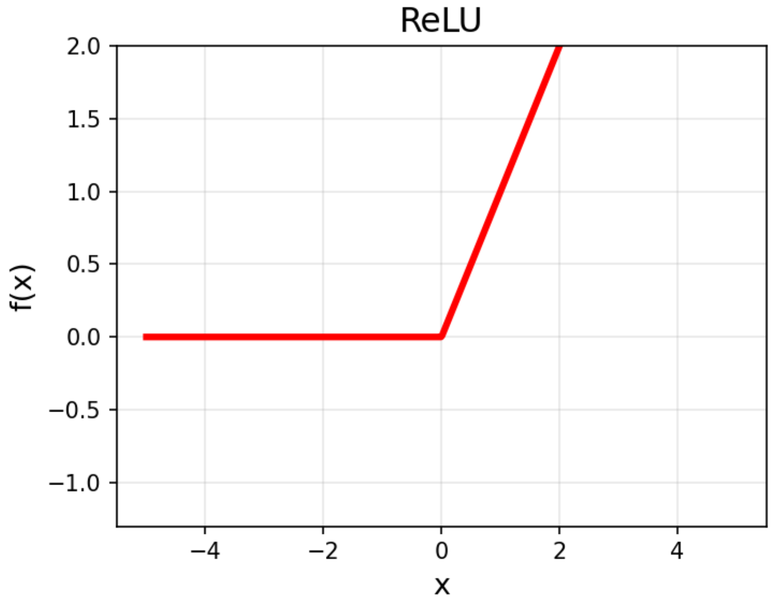
Redes Neuronales: Profundas

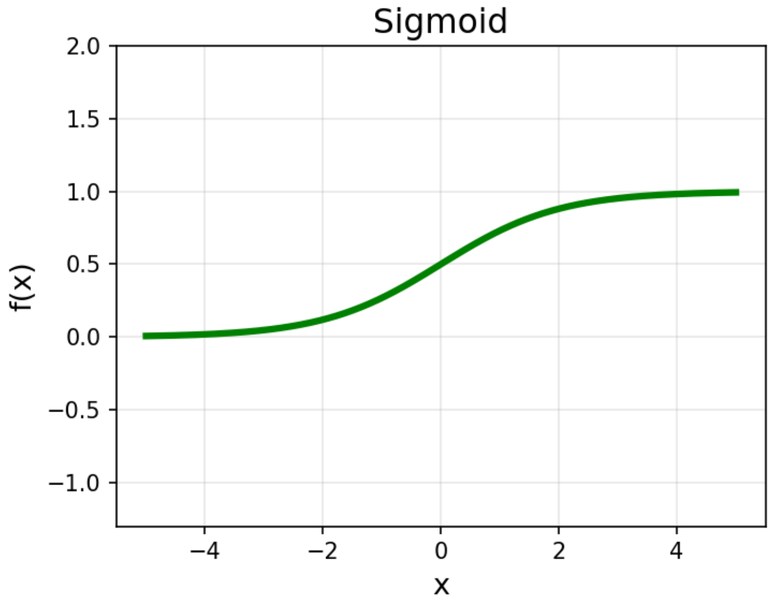


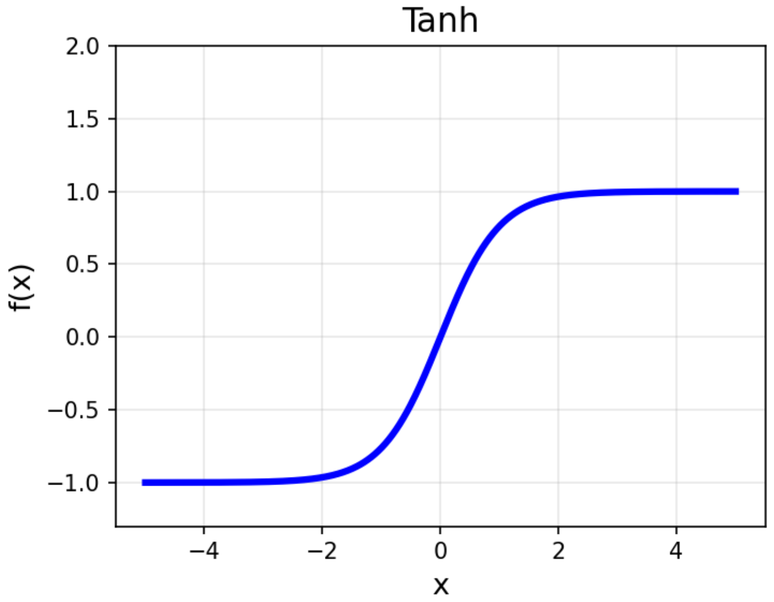
Back Propagate

$$\frac{\partial \mathscr{L}}{\partial \theta} = \frac{\partial \mathscr{L}}{\partial a_n} \prod_{k=1}^{n} \frac{\partial a_k}{\partial a_{k-1}} \frac{\partial a_0}{\partial \theta} \qquad \text{if} \quad \frac{\partial a_k}{\partial a_{k-1}} < 1$$

Redes Neuronales: Vanishing Gradients

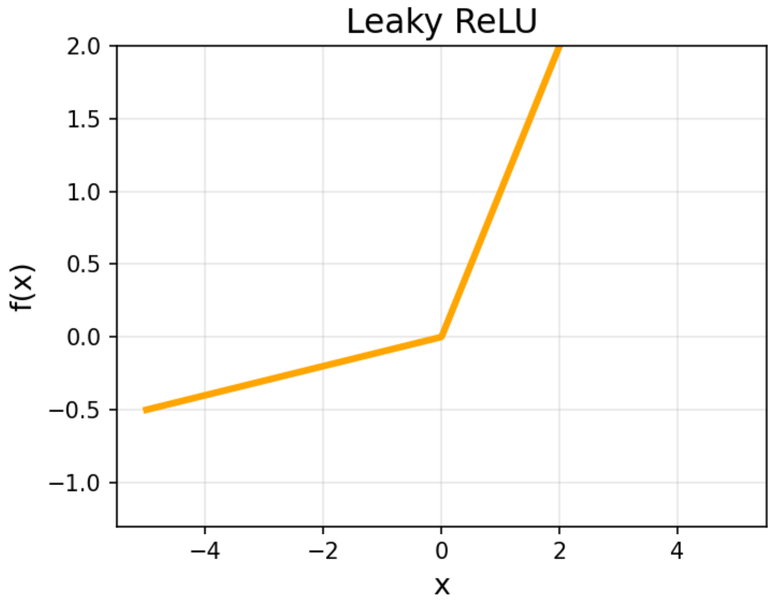


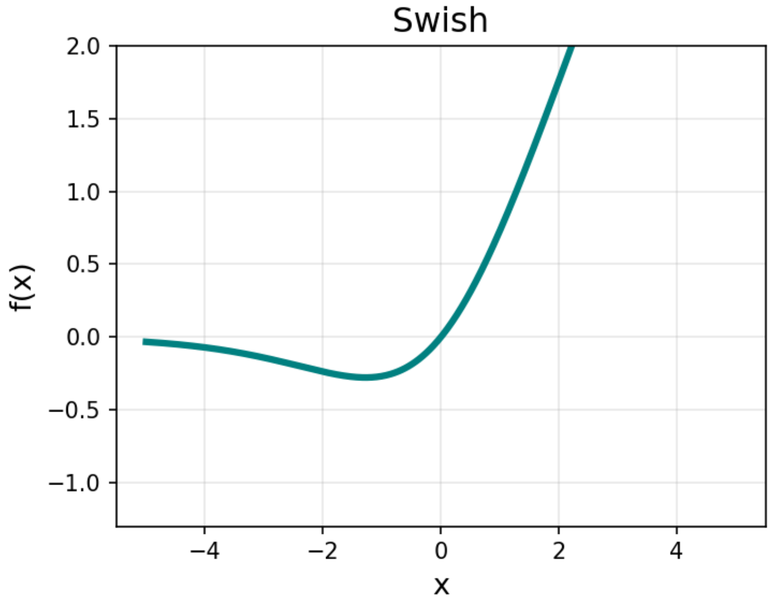


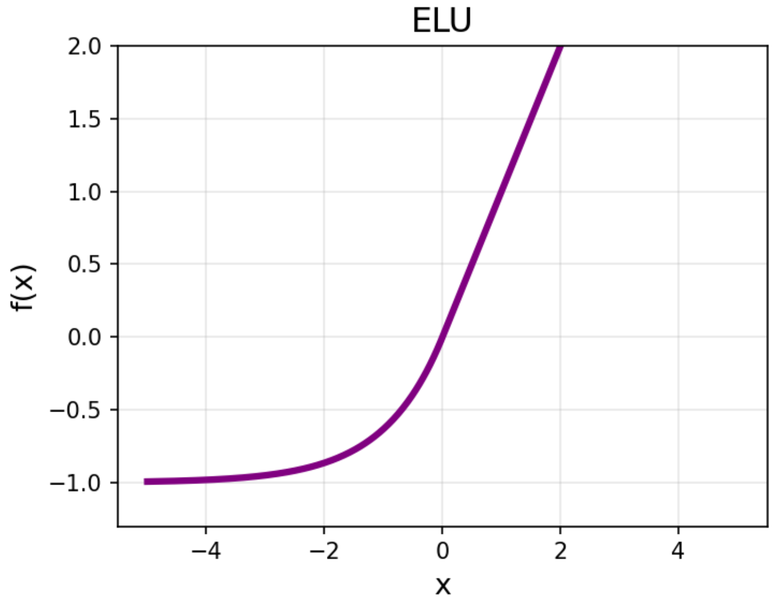


$$\text{Gradiente total} \approx \frac{\partial \mathscr{L}}{\partial \theta} = \frac{\partial \mathscr{L}}{\partial a_n} \prod_{k=1}^{n} \frac{\partial a_k}{\partial a_{k-1}} \approx (<1)^n$$

Redes Neuronales: Vanishing Gradients

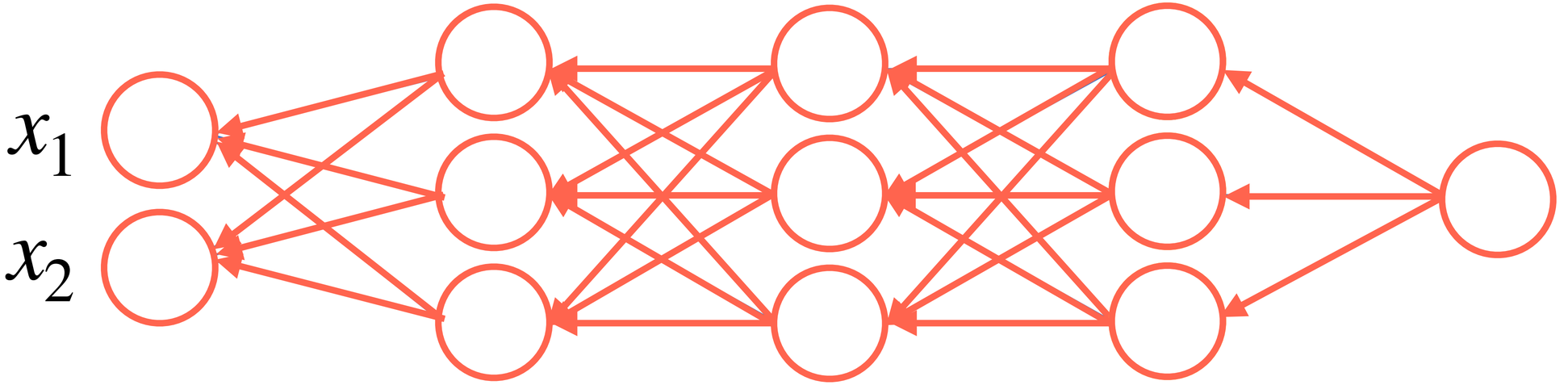






$$\text{Gradiente total} \approx \frac{\partial \mathscr{L}}{\partial \theta} = \frac{\partial \mathscr{L}}{\partial a_n} \prod_{k=1}^{n} \frac{\partial a_k}{\partial a_{k-1}} \approx (<1)^n$$

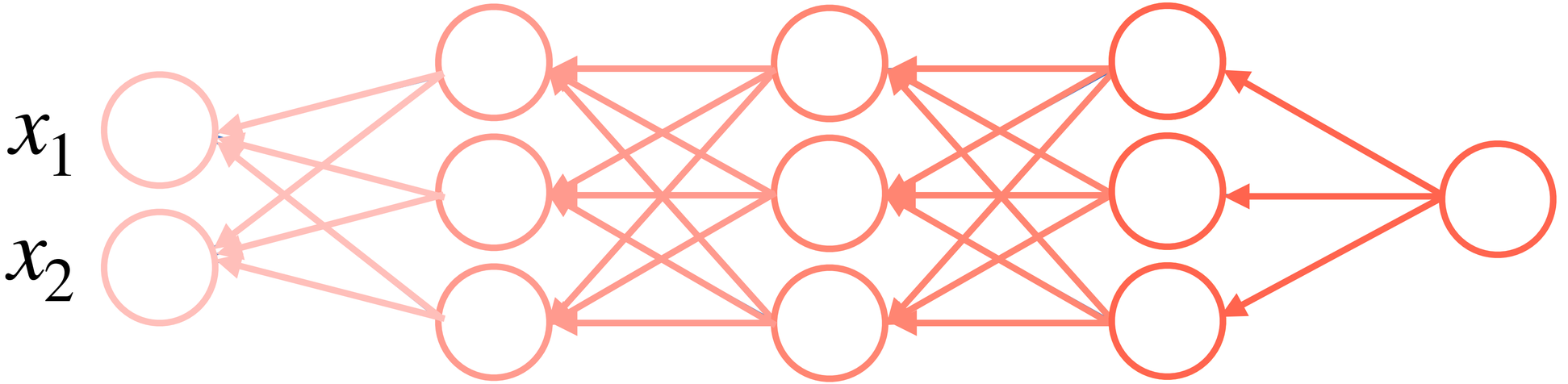
Redes Neuronales: Profundas



Back Propagate

$$\frac{\partial \mathscr{L}}{\partial \theta} = \frac{\partial \mathscr{L}}{\partial a_n} \prod_{k=1}^{n} \frac{\partial a_k}{\partial a_{k-1}} \frac{\partial a_0}{\partial \theta} \quad \text{if} \quad \frac{\partial a_k}{\partial a_{k-1}} < 1$$

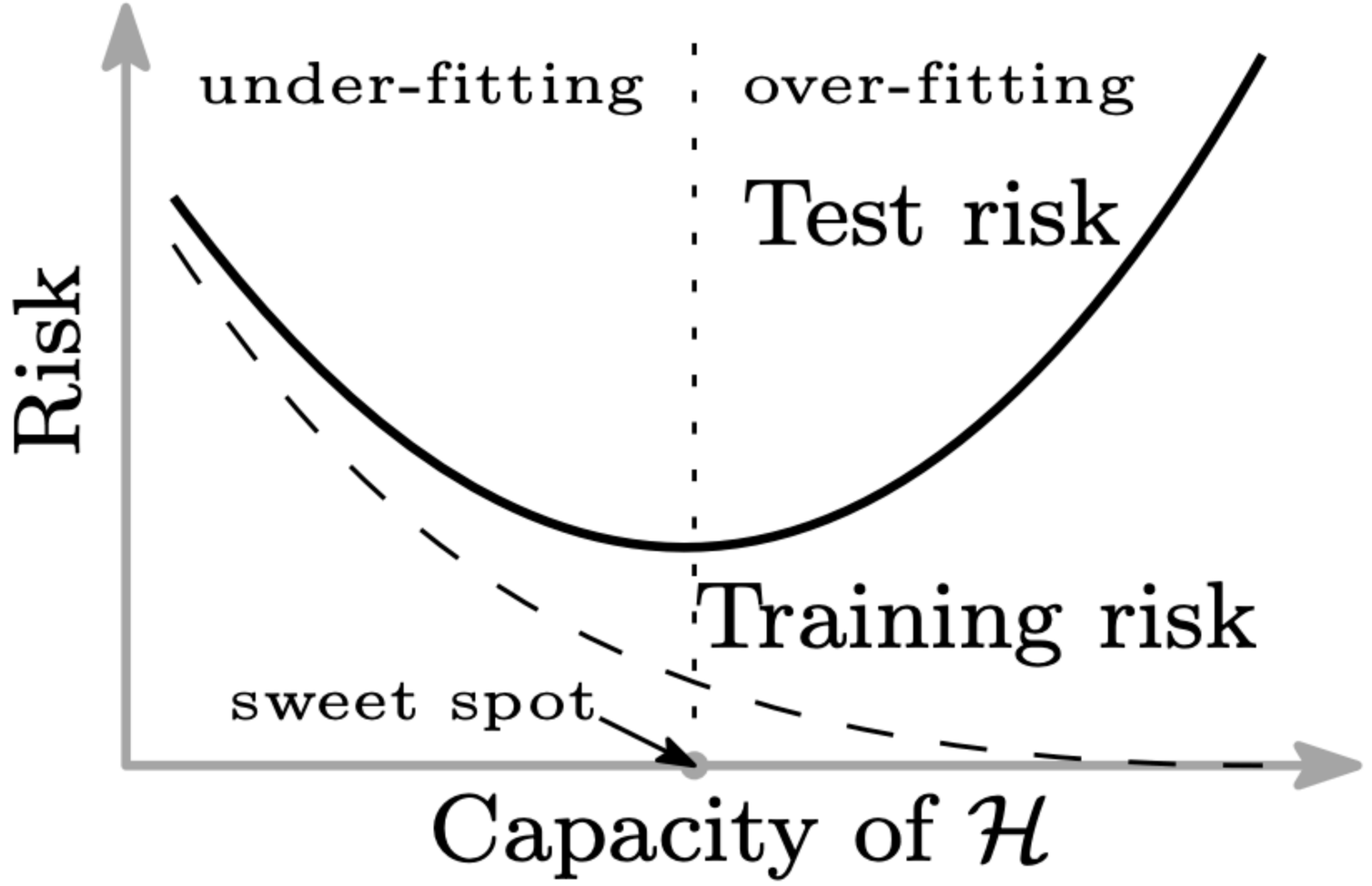
Redes Neuronales: Profundas



Back Propagate

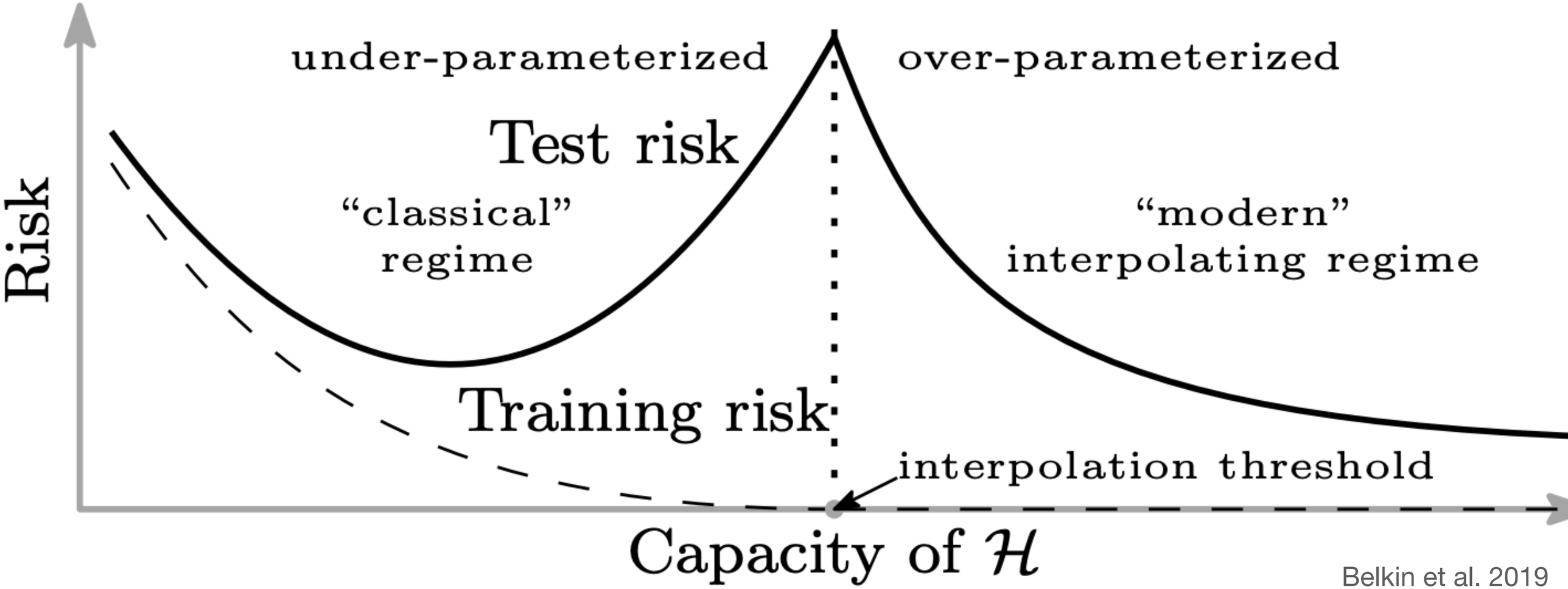
$$\frac{\partial \mathscr{L}}{\partial \theta} = \frac{\partial \mathscr{L}}{\partial a_n} \prod_{k=1}^{n} \frac{\partial a_k}{\partial a_{k-1}} \frac{\partial a_0}{\partial \theta} \quad << 1$$

Redes Neuronales: Trade Off Bias-Variance



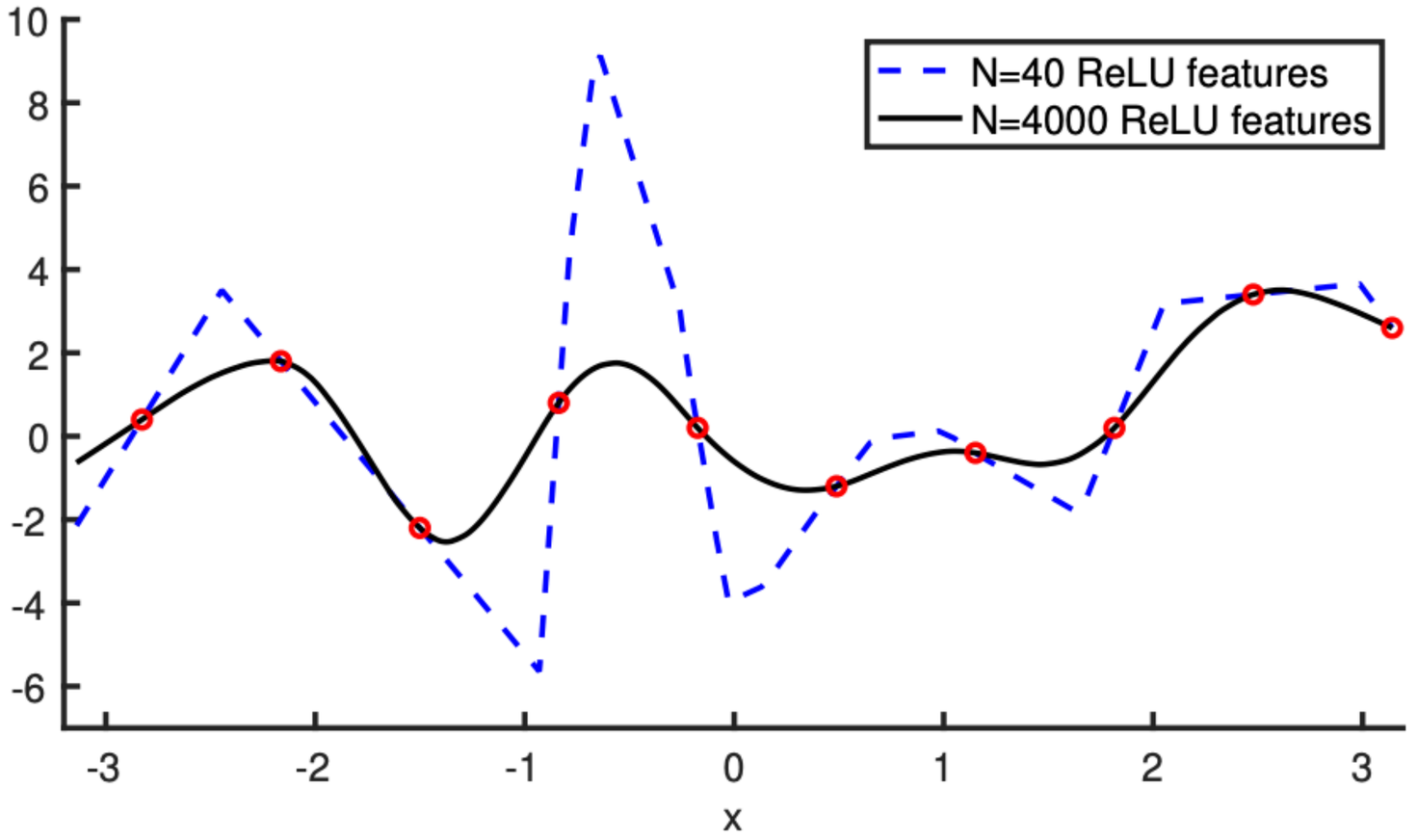
Belkin et al. 2019

Redes Neuronales: Trade Off Bias-Variance



Belkin et al. 2019

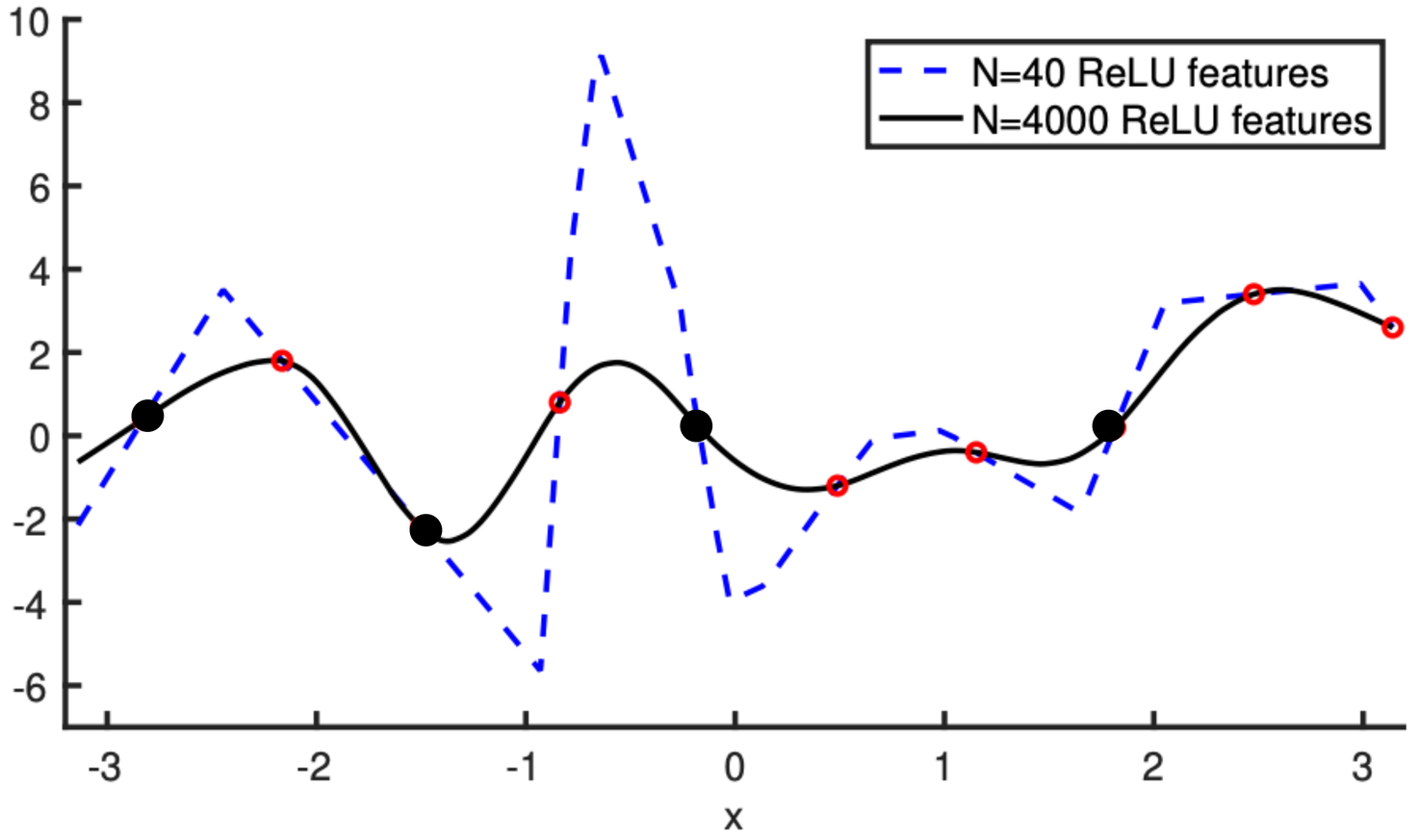
Redes Neuronales: Double Descent



Para mas detalles: Rocks & Mehta 2022

Belkin et al. 2019

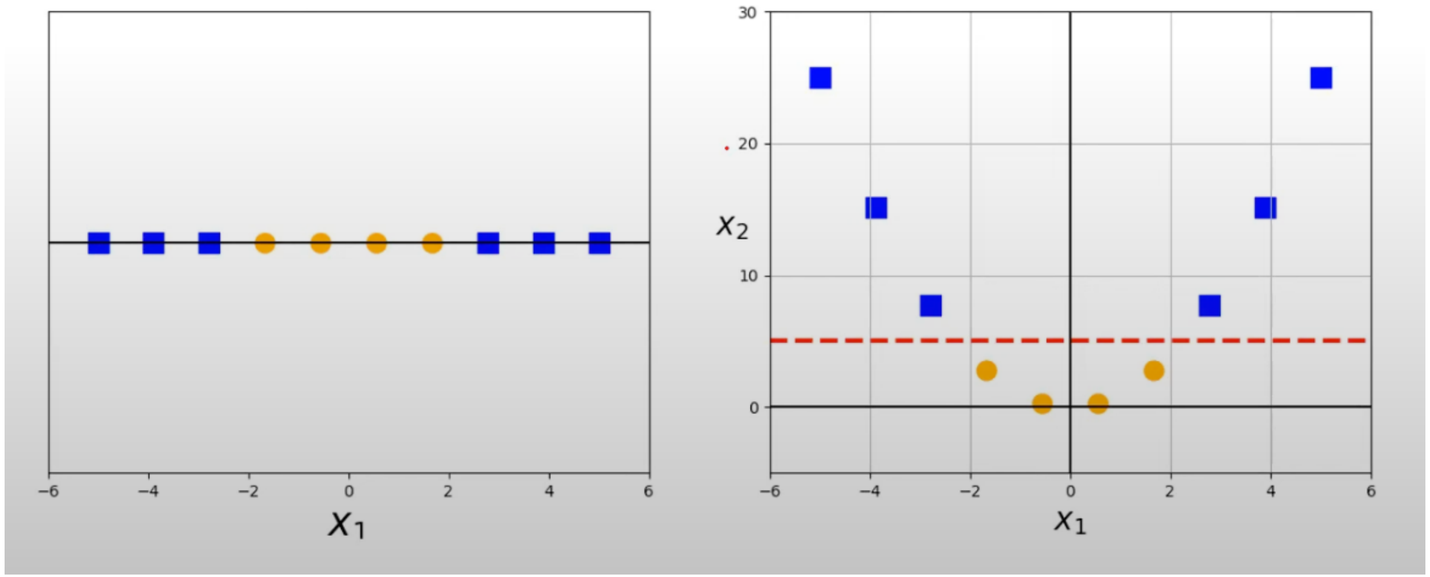
Double Descent: Estocasticidad



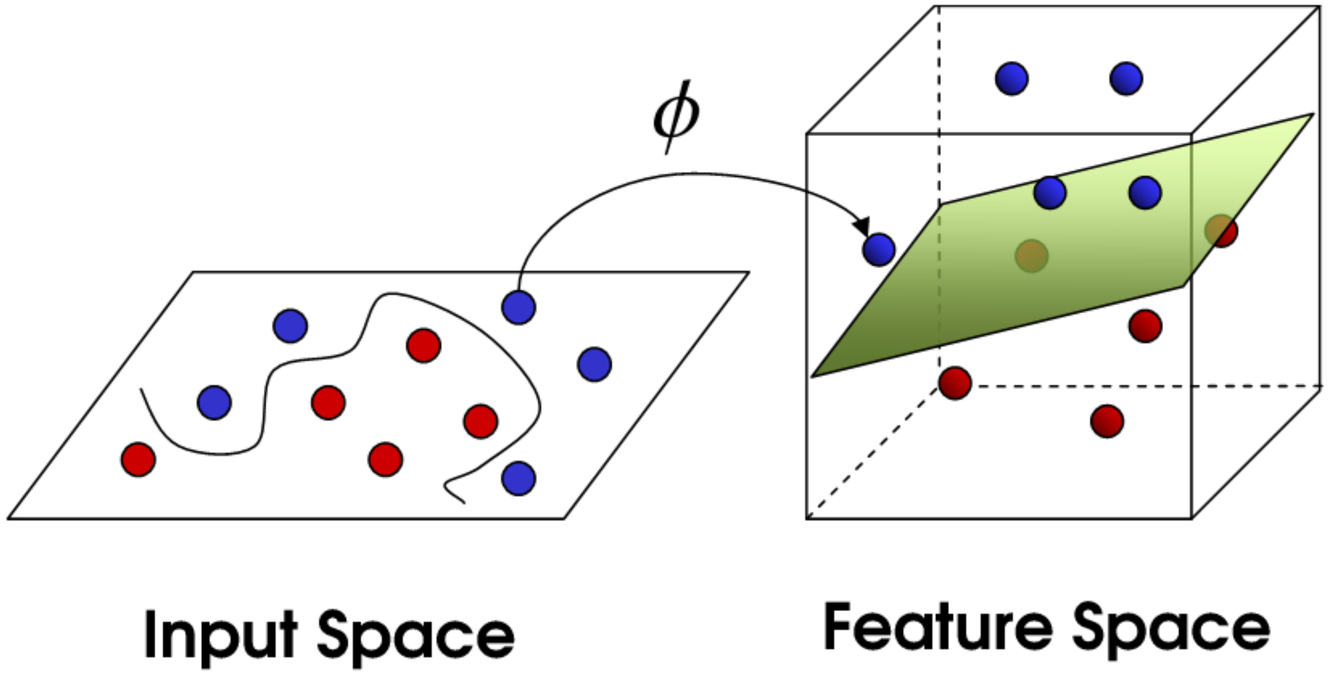
Para mas detalles: Drew Wilimitis 2018, Shiyu Ji

Belkin et al. 2019

Double Descent: Espacio de Atributos



Double Descent: Espacio de Atributos



Mañana

- Desafíos para MLP en Análisis de Imágenes
- De MLP a Redes Neuronales Convolucionales (CNN)
- Ajuste de Curvas con Redes Neuronales Shallow
- Como Diseñar y Entrenar una CNN