Aprendizaje profundo

Descenso por gradiente

Gibran Fuentes-Pineda

Agosto 2023

Estimación de parámetros

- En el entrenamiento de neuronas artificiales y redes neuronales, se desea encontrar los valores de los pesos y sesgos[1] que minimicen una función de pérdida $\mathcal{L} : \boldsymbol{\Theta} \to \mathbb{R}$

$$\bar{\boldsymbol{\theta}} \leftarrow \underset{\boldsymbol{\theta} \in \boldsymbol{\Theta}}{\arg\min}\, \mathcal{L}(\boldsymbol{\theta})$$

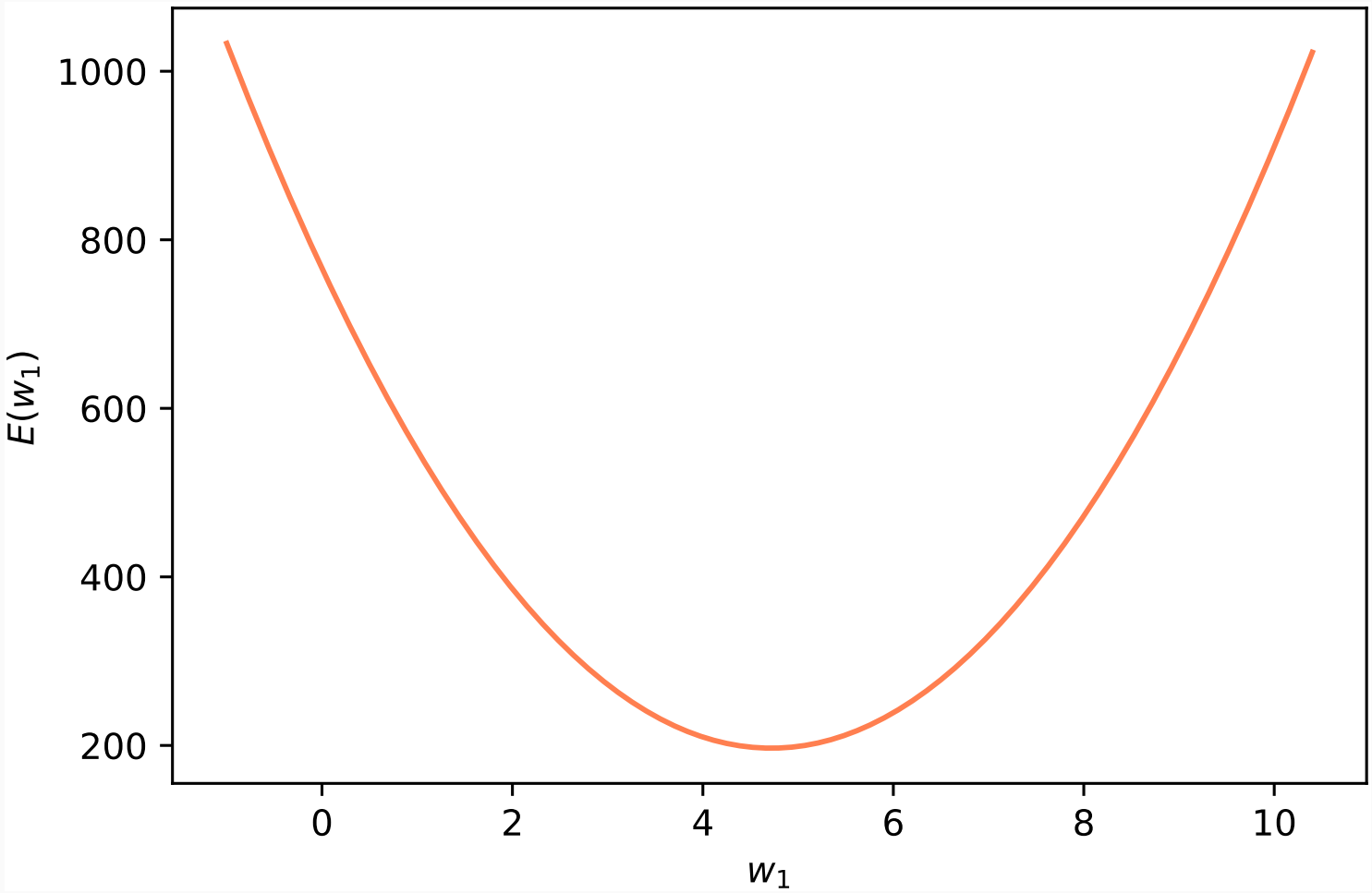
  donde $\boldsymbol{\Theta} \subseteq \mathbb{R}^d$.

- Se dice que $\bar{\boldsymbol{\theta}}$ es el mínimo global.

[1] A esto se le conoce como estimación de parámetros.

Minimización de pérdida

- Para neuronas artificiales individuales la función de pérdida es convexa



Descenso por gradiente (GD)

- Algoritmo iterativo de primer orden que va moviendo los pesos $\mathbf{w}$ y sesgos $\mathbf{b}$ hacia donde la pérdida descienda más rápido en el vecindario, esto es,

$$\boldsymbol{\theta}^{[t+1]} = \boldsymbol{\theta}^{[t]} - \alpha \nabla \mathcal{L}(\boldsymbol{\theta}^{[t]})$$

  donde

$$\boldsymbol{\theta} = \{\mathbf{w}, \mathbf{b}\}$$

$$\nabla \mathcal{L}(\boldsymbol{\theta}^{[t]}) = \left[\frac{\partial \mathcal{L}}{\partial \theta_0^{[t]}}, \cdots, \frac{\partial \mathcal{L}}{\partial \theta_d^{[t]}} \right]$$

- A α se le conoce como tasa de aprendizaje

Descenso por gradiente estocástico (SGD)

- Aproximación estocástica de GD: estima $\nabla \mathcal{L}(\boldsymbol{\theta}^{[t]})$ y actualiza pesos y sesgos con un minilote $\mathcal{B}$ de ejemplos de entrenamiento
  - $|\mathcal{B}|$ es un hiperparámetro
  - Es común dividir y ordenar aleatoriamente el conjunto de n ejemplos de entrenamiento en k minilotes ($|\mathcal{B}| \times k \approx n$); una época ocurre cada vez que se han considerado los k minilotes

Gradiente para regresión lineal

- Para una neurona lineal, el gradiente de la función de pérdida de suma de errores cuadráticos respecto a los pesos y sesgos está dado por

$$\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial w_j} = \frac{\partial ECM(\mathbf{y}, \hat{\mathbf{y}})}{\partial w_j} = \sum_{i=1}^{n} \left(\hat{y}^{(i)} - y^{(i)} \right) \cdot x_j^{(i)}$$

$$\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial b} = \frac{\partial ECM(\mathbf{y}, \hat{\mathbf{y}})}{\partial b} = \sum_{i=1}^{n} \left(\hat{y}^{(i)} - y^{(i)} \right)$$

Algoritmo del descenso por gradiente para regresión lineal

1. Asignar valores aleatorios a los parámetros $\boldsymbol{\theta}$

2. Repetir hasta que converja

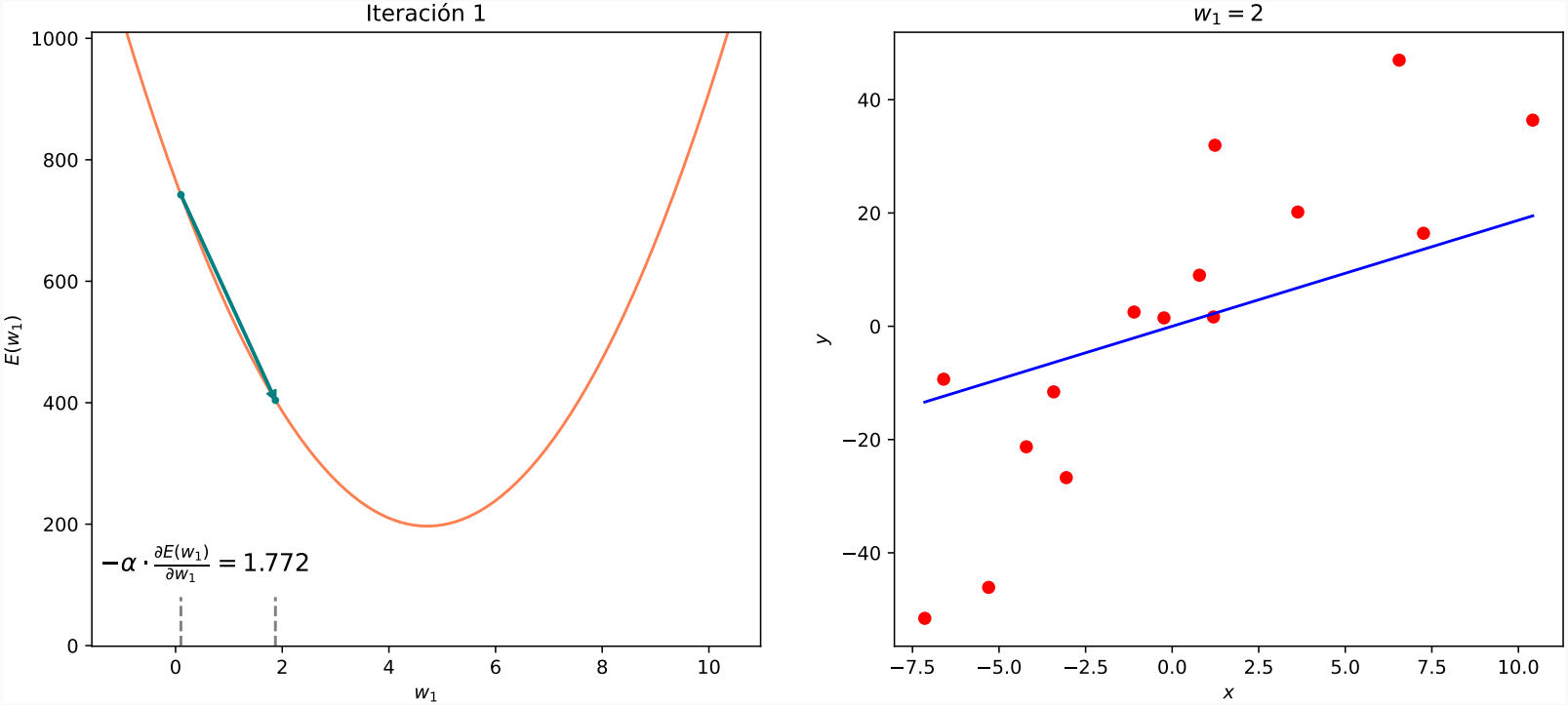
$$b^{\{t+1\}} \leftarrow b^{\{t\}} - \alpha \overbrace{\sum_{i=1}^{n} \left(\hat{y}^{(i)} - y^{(i)} \right)}^{\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial b^{\{t\}}}}$$

$$w_j^{\{t+1\}} \leftarrow w_j^{\{t\}} - \alpha \underbrace{\sum_{i=1}^{n} \left(\hat{y}^{(i)} - y^{(i)} \right) \cdot x_j^{(i)}}_{\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial w_j^{\{t\}}}}$$

(Actualización simultánea de b y todos los w_j)

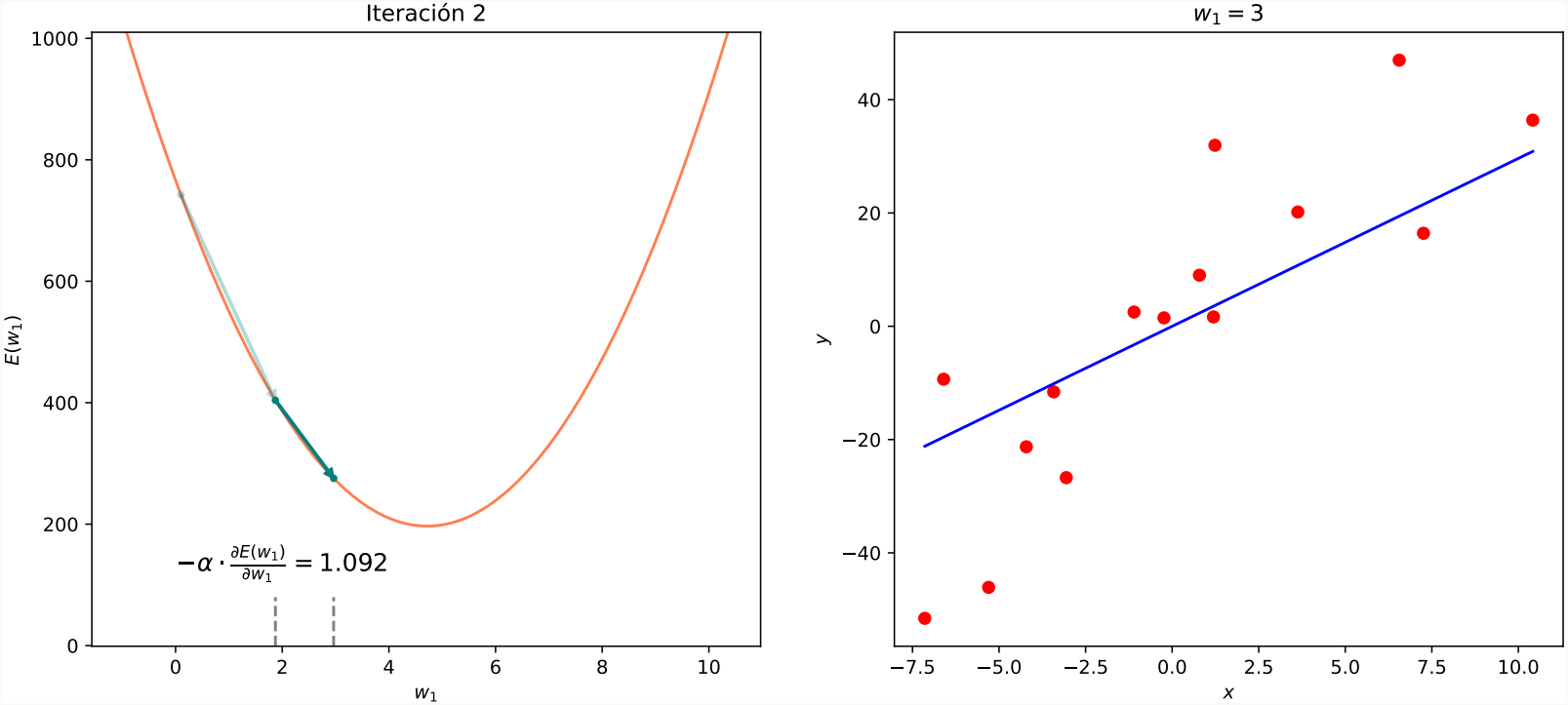
Ejemplo del algoritmo de descenso por gradiente (GD)

- Inicializando w_1 con un valor menor al que minimiza la función de pérdida



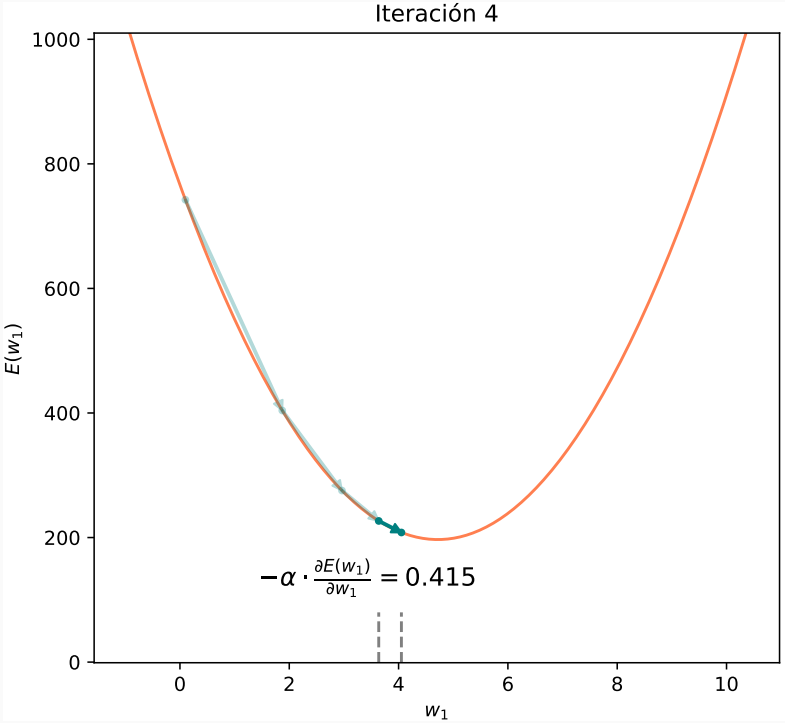
Ejemplo del algoritmo de descenso por gradiente (GD)

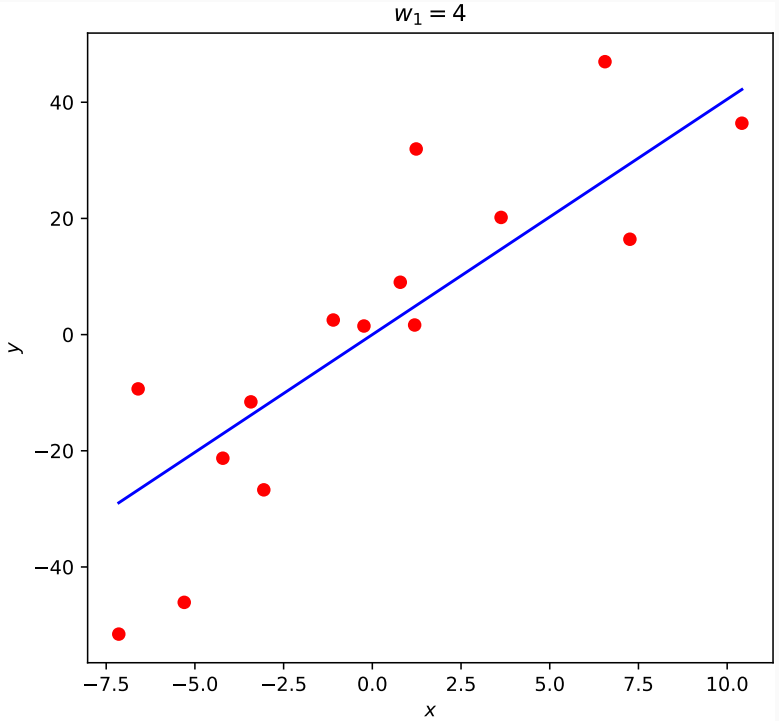
- Inicializando w_1 con un valor menor al que minimiza la función de pérdida



Ejemplo del algoritmo de descenso por gradiente (GD)

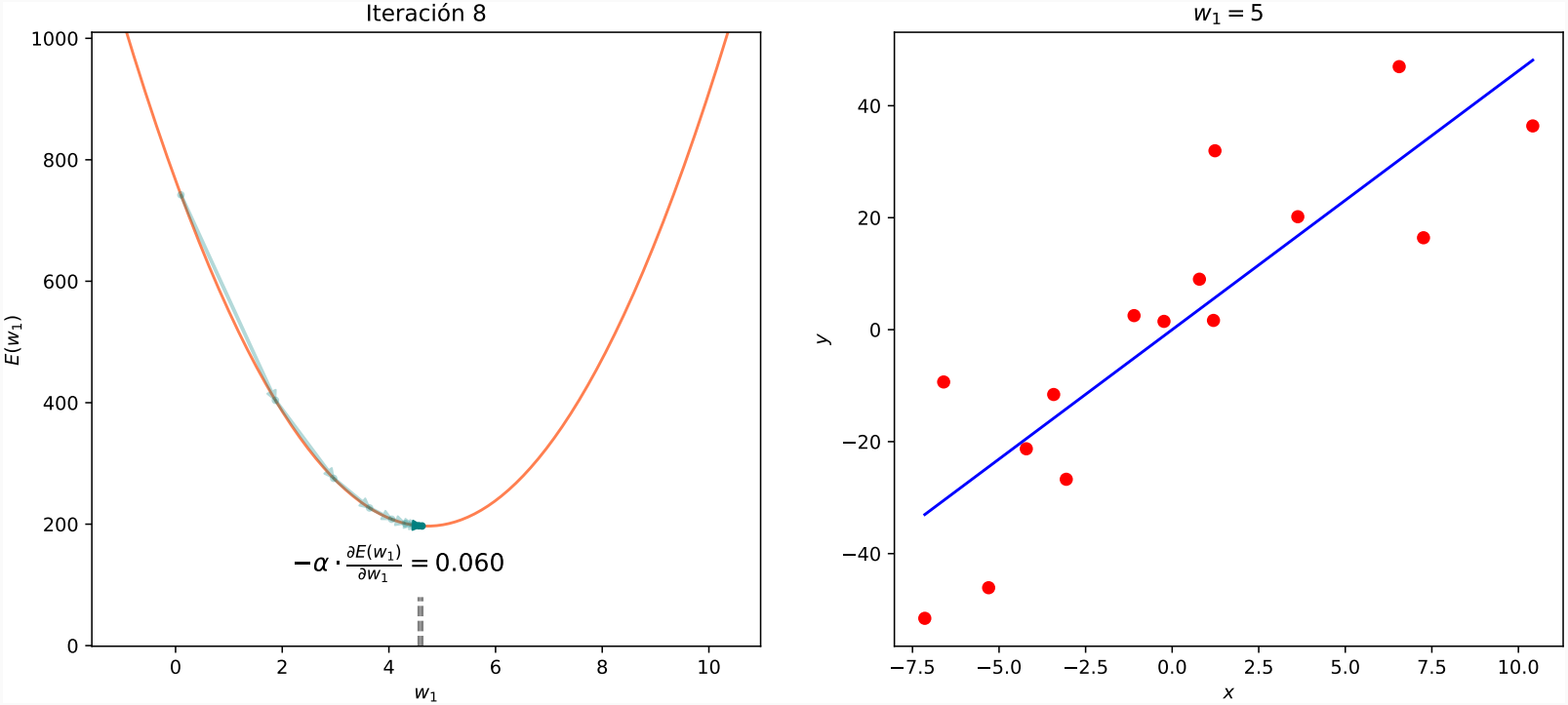
- Inicializando w_1 con un valor menor al que minimiza la función de pérdida





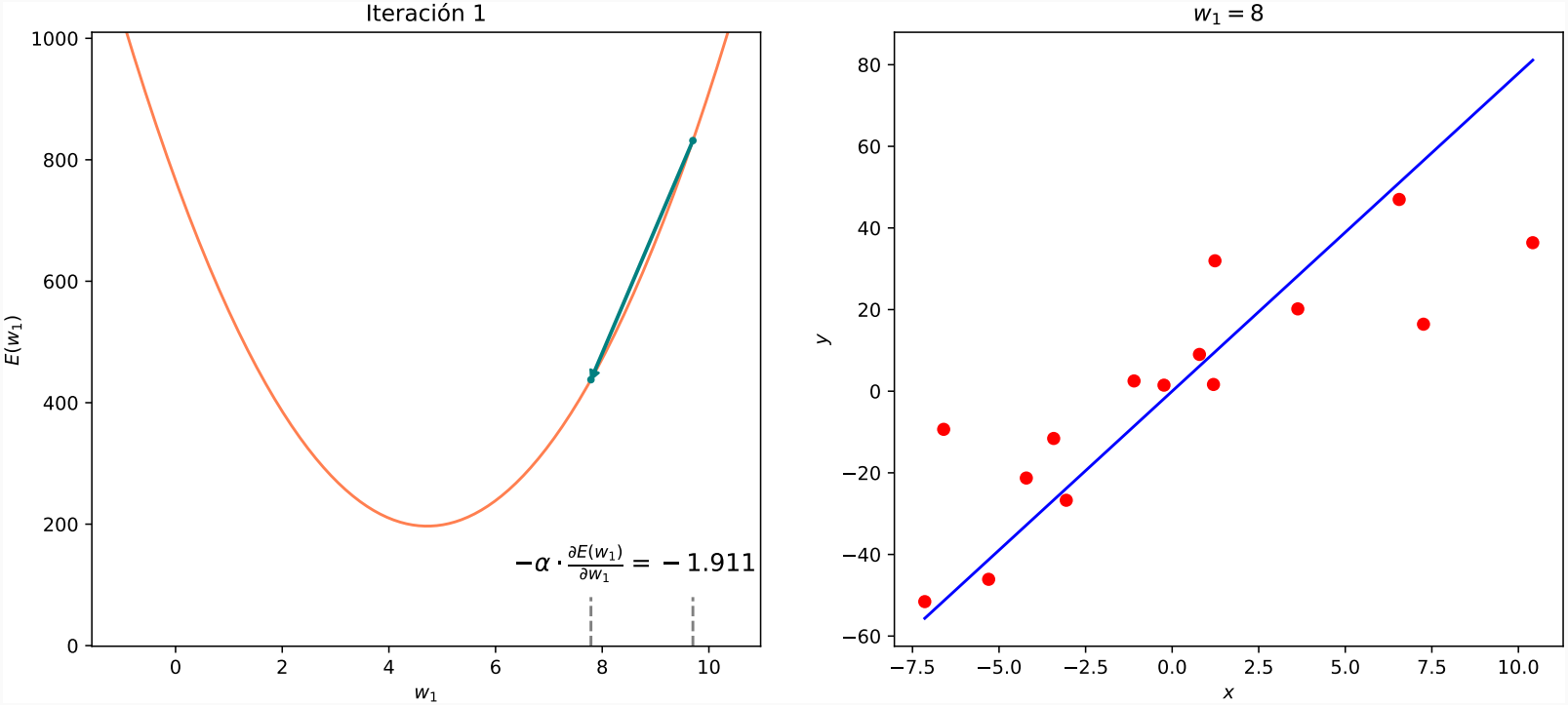
Ejemplo del algoritmo de descenso por gradiente (GD)

- Inicializando w_1 con un valor menor al que minimiza la función de pérdida



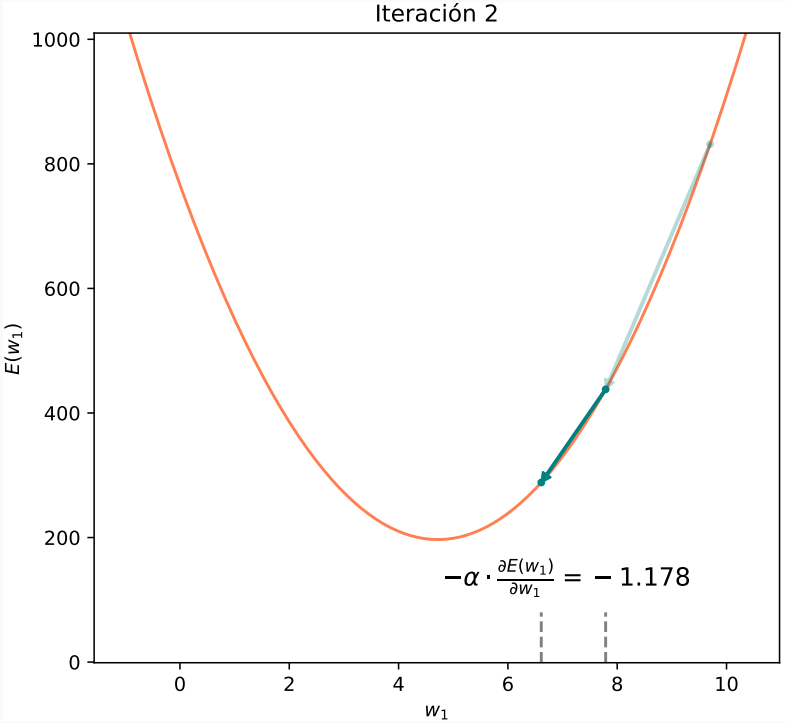
Ejemplo del algoritmo de descenso por gradiente (GD)

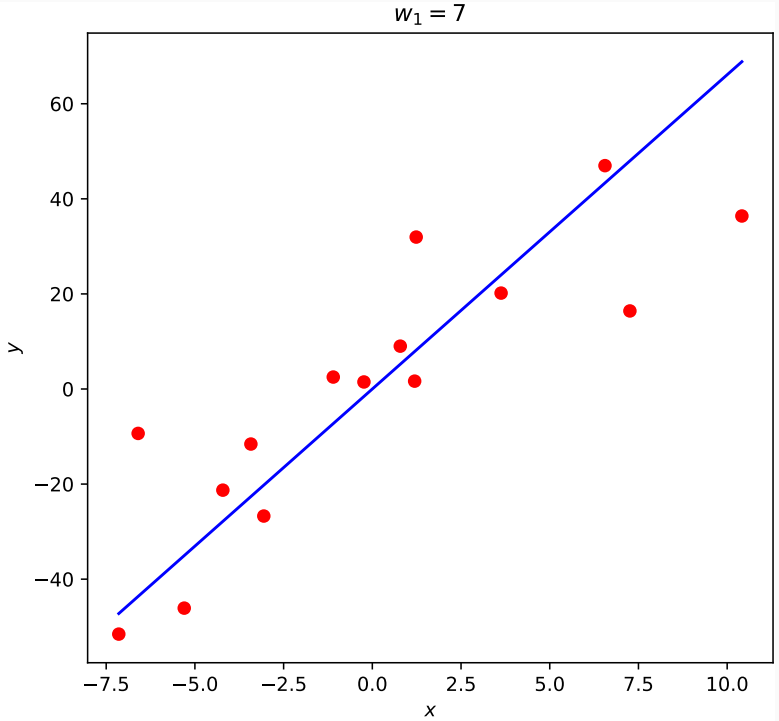
- Inicializando w_1 con un valor mayor al que minimiza la función de pérdida



Ejemplo del algoritmo de descenso por gradiente (GD)

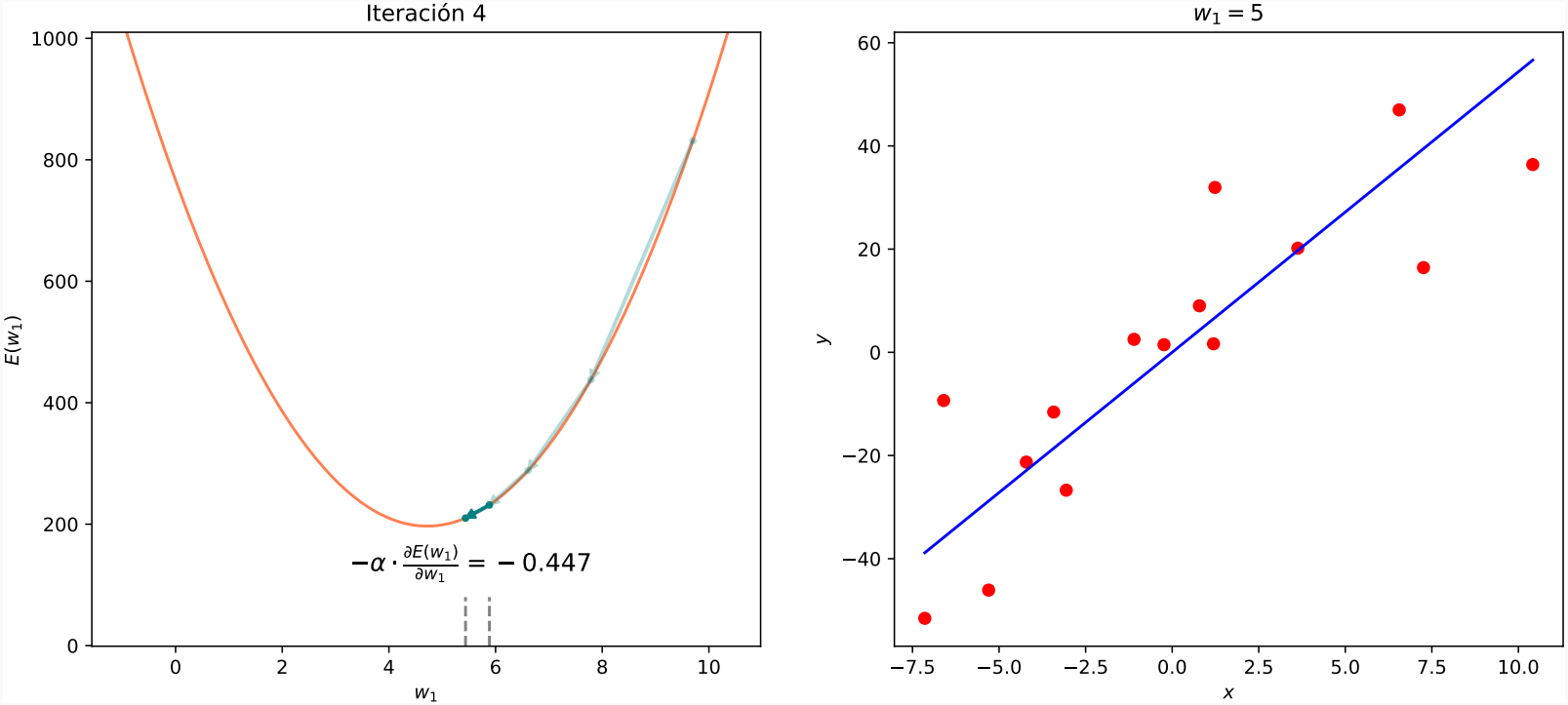
- Inicializando w_1 con un valor mayor al que minimiza la función de pérdida





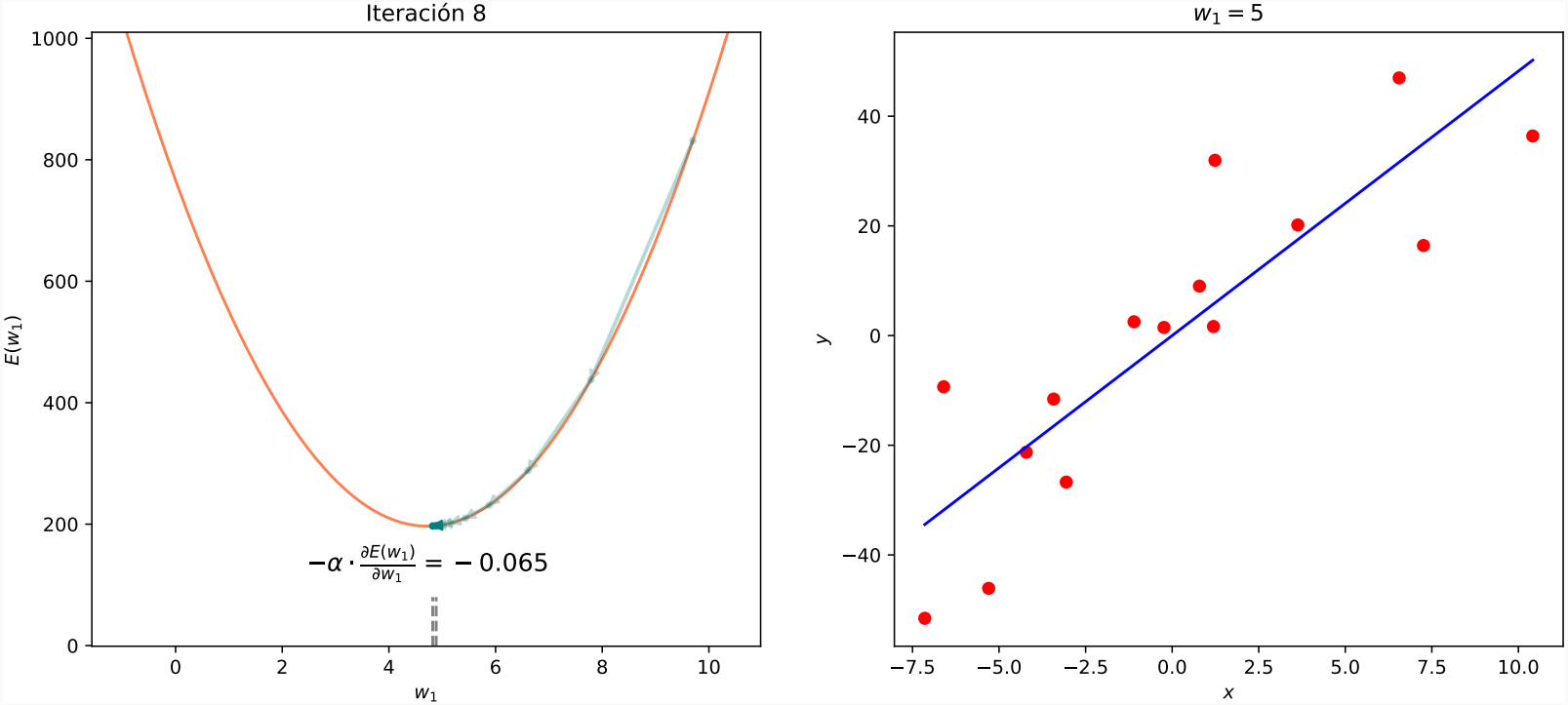
Ejemplo del algoritmo de descenso por gradiente (GD)

- Inicializando w_1 con un valor mayor al que minimiza la función de pérdida



Ejemplo del algoritmo de descenso por gradiente (GD)

- Inicializando w_1 con un valor mayor al que minimiza la función de pérdida



Gradiente para regresión logística

- Para una neurona logística, el gradiente de la función de pérdida de entropía cruzada binaria respecto a los pesos y sesgos está dado por

$$\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial w_j} = \frac{\partial ECB(\mathbf{y}, \hat{\mathbf{y}})}{\partial w_j} = \sum_{i=1}^{n} \left(\hat{y}^{(i)} - y^{(i)} \right) \cdot x_j^{(i)}$$

$$\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial b} = \frac{\partial ECB(\mathbf{y}, \hat{\mathbf{y}})}{\partial b} = \sum_{i=1}^{n} \left(\hat{y}^{(i)} - y^{(i)} \right)$$

donde $\hat{y}^{(i)} = \sigma(\mathbf{w}^\top \mathbf{x}^{(i)} + b)$

Gradiente para regresión softmax

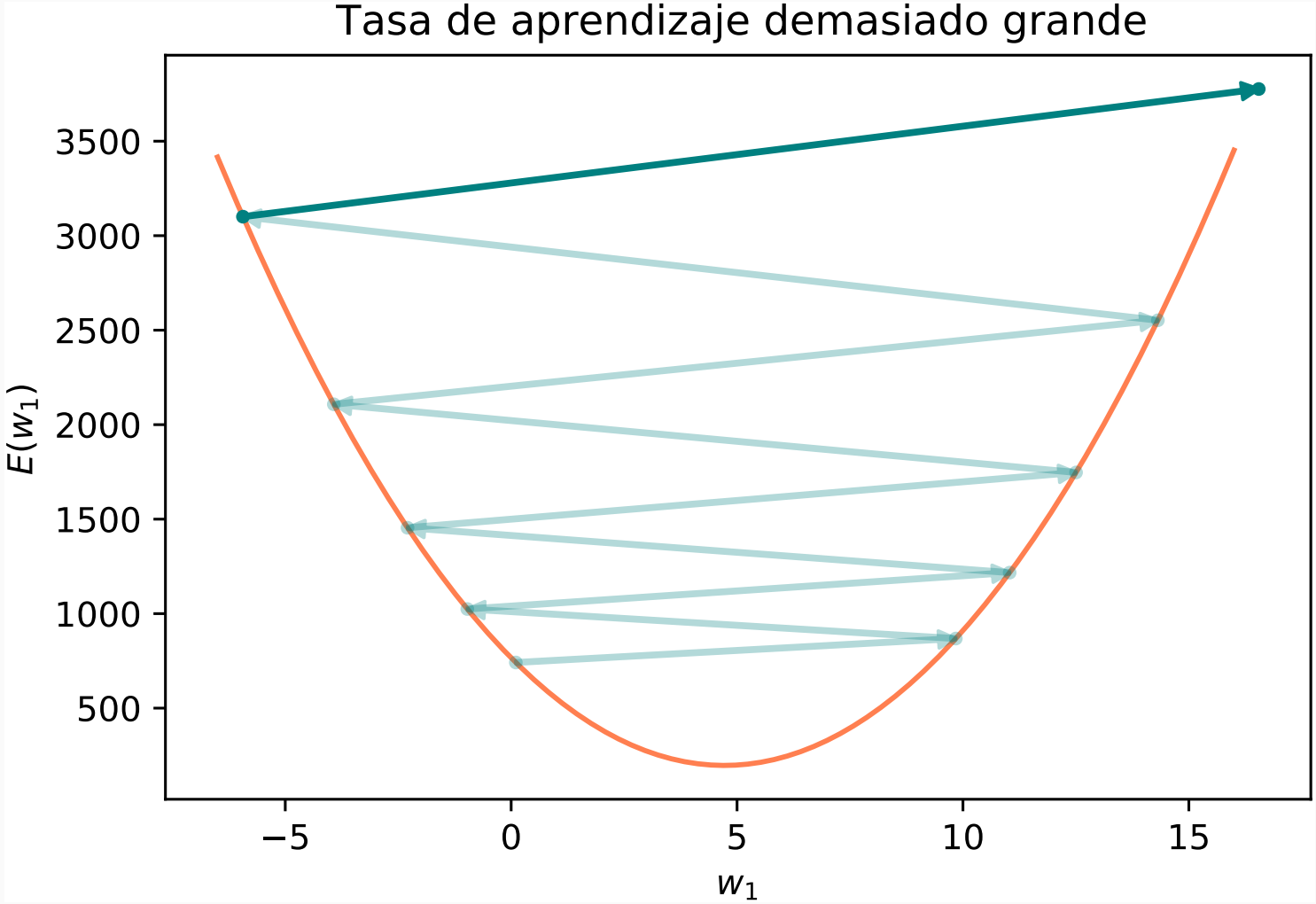
- Para una neurona logística, el gradiente de la función de pérdida de entropía cruzada binaria respecto a los pesos y sesgos está dado por

$$\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial w_{jk}} = \frac{\partial ECC(\mathbf{Y}, \hat{\mathbf{Y}})}{\partial w_{jk}} = \sum_{i=1}^{n} (y_k^{(i)} - [y^{(i)} = k]) \cdot x_j^{(i)}$$

$$\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial b_k} = \frac{\partial ECC(\mathbf{Y}, \hat{\mathbf{Y}})}{\partial b_k} = \sum_{i=1}^{n} (y_k^{(i)} - [y^{(i)} = k])$$

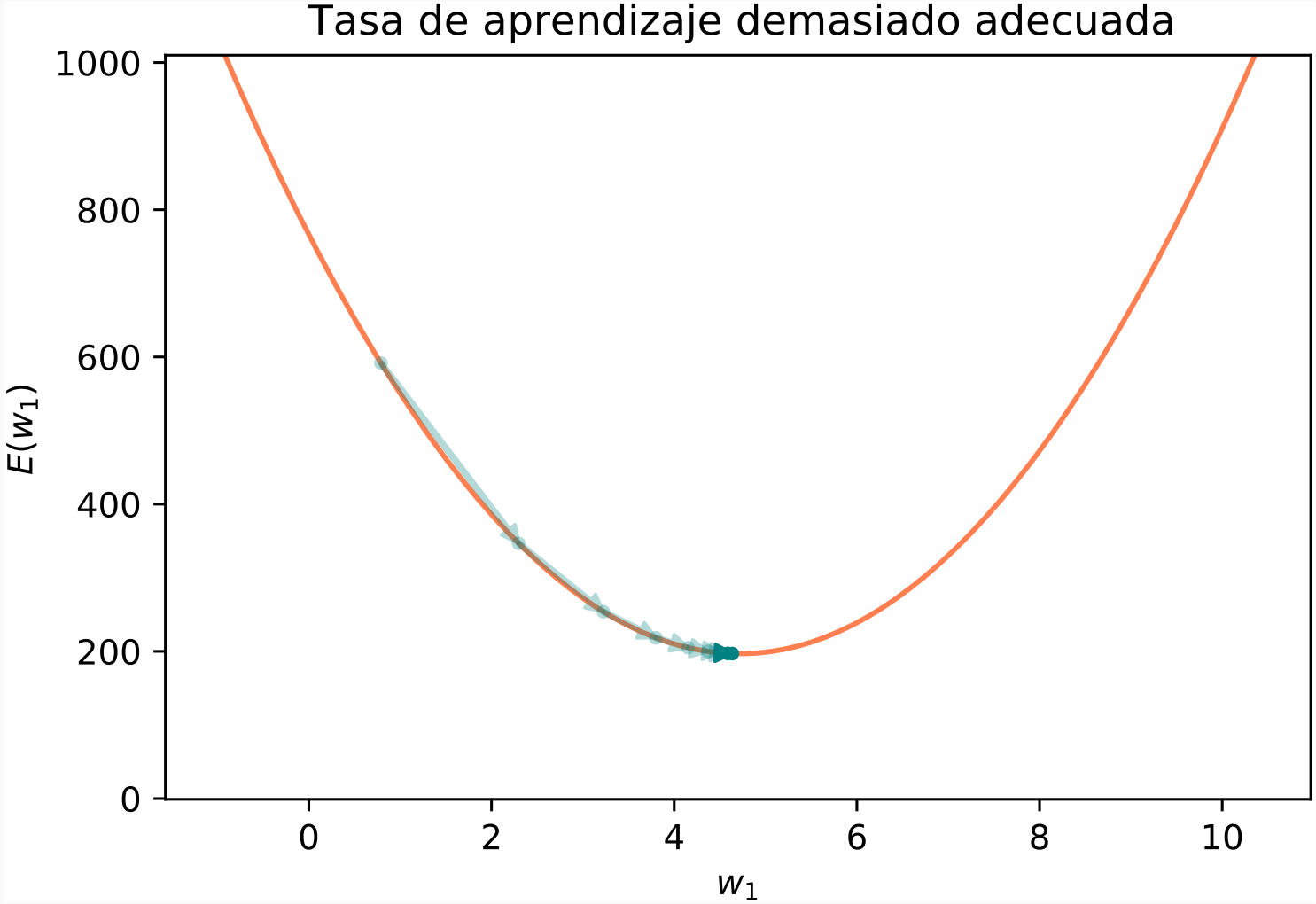
donde $\hat{y}_k^{(i)} = softmax(\mathbf{W}^\top \mathbf{x}^{(i)} + \mathbf{b})_k$

Sensibilidad a tasa de aprendizaje α



Sensibilidad a tasa de aprendizaje α

Sensibilidad a tasa de aprendizaje α



Sensibilidad a tasa de aprendizaje α