Redes neuronales — Clase # 5 — 14/08/2025

```
self.biases = [np.random.randn(y, 1) for y in sizes[1:]]
self.weights = [np.random.randn(y, x)
                for x, y in zip(sizes[:-1], sizes[1:])]
```

Sirve para unir los pares

size = [764, 15, 10]
0 1 2

$[w]_{10,15}$

self.weight = (w_{11} ...) } 15 , ( ) } 10

764

```
Tabnine | Edit | Test | Fix | Explain | Document
def feedforward(self, a):
    """Return the output of the network if ''a'' is input."""
    for b, w in zip(self.biases, self.weights):
        a = sigmoid(np.dot(w, a) + b)
    return a
```

Función sigmoide — Producto matricial

Iteraciones del ciclo

stochastic gradient descent

```
def SGD(self, training_data, epochs, mini_batch_size, eta,
        test_data=None):
    """Train the neural network using mini-batch stochastic gradient descent."""
```

$$training\_data = \left[(x_1, y_1), (x_2, y_2), (x_3, y_3), \dots, \right]$$

One hot - encoding:

```
mini_batches = [
    training_data[k:k+mini_batch_size]
    for k in range(0, n, mini_batch_size)]
for mini_batch in mini_batches:
```

Índices de los datos en cada mini_batch

$$training\_data = \left[\underbrace{(x_1, y_1), (x_2, y_2), (x_3, y_3), \dots,}_{1 \text{ mini\_batch}} \right]$$

↳ 2° mini-batch

```
mini_batches = [ → lista de mini_batches
    training_data[k:k+mini_batch_size]
    for k in range(0, n, mini_batch_size)]
for mini_batch in mini_batches:
    self.update_mini_batch(mini_batch, eta)
if test_data:
```

Nos pasa el código y nos recibimos

Separamos los datos en mini batches

Estocástico →