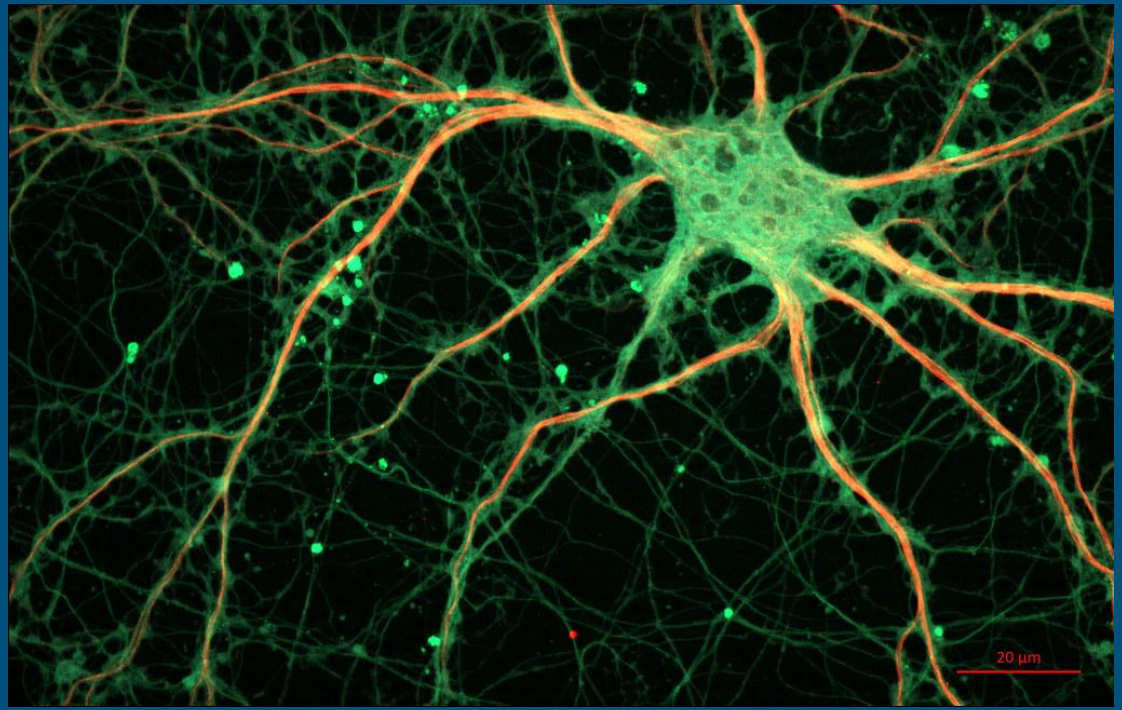
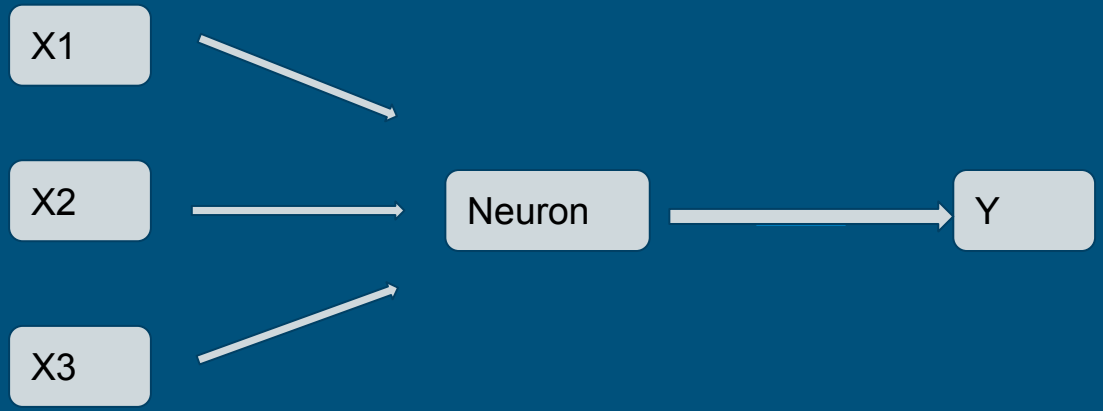
Deep Learning

Neuronal Networks

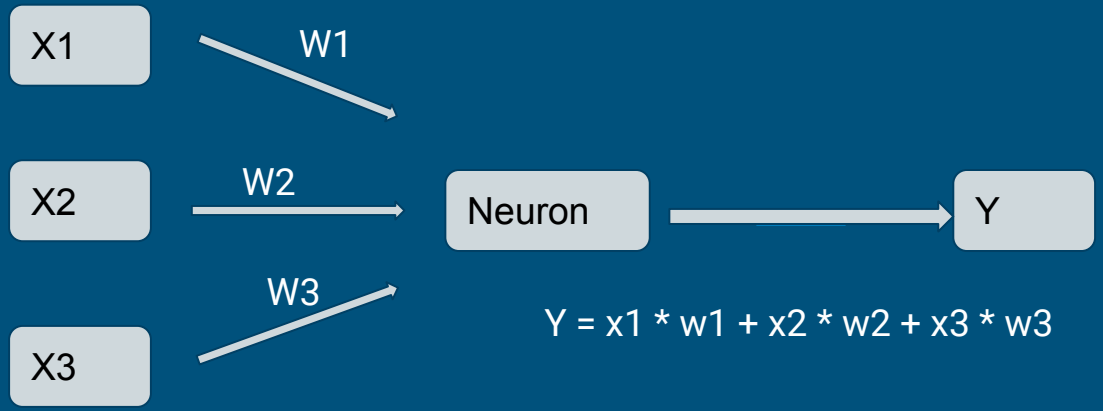
Human Neuron



Artificial Neuron



Artificial Neuron

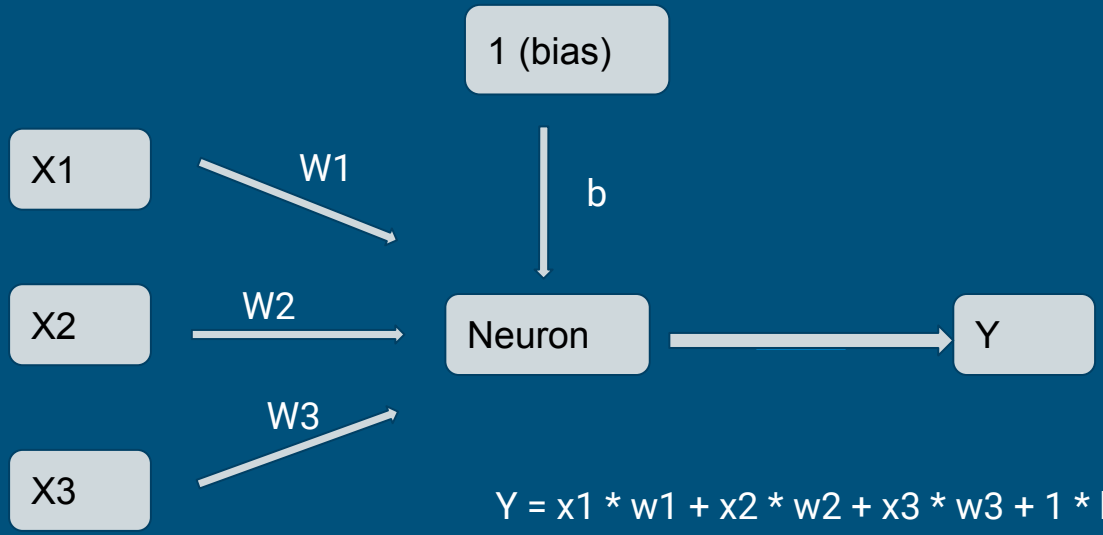


$$Y = x1 * w1 + x2 * w2 + x3 * w3$$

Time for a first example

Let’s train a neuron to predict the conversion from kilometers to miles

Artificial Neuron (BIAS)



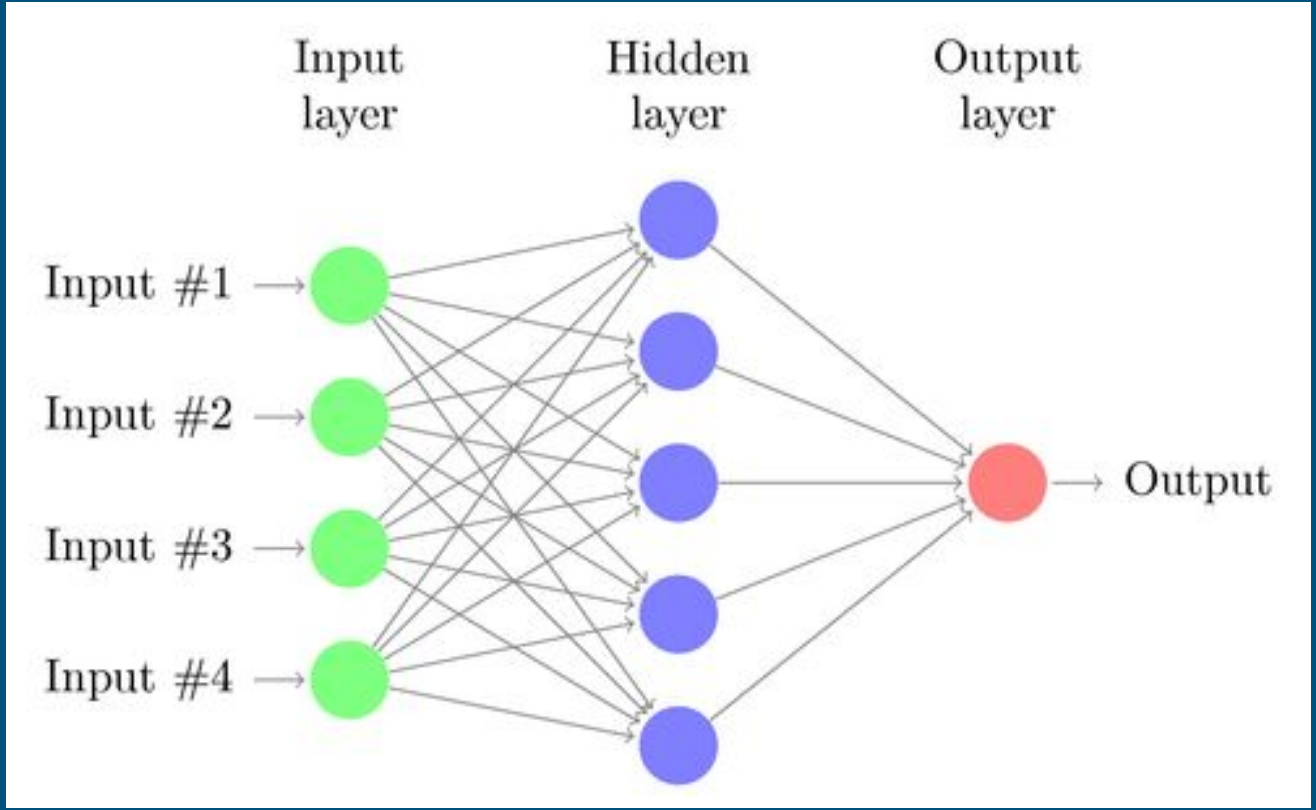
$$Y = x1 * w1 + x2 * w2 + x3 * w3 + 1 * b$$

Example: Celsius to Fahrenheit (different point of zero) [celsius * 1,8 + 32]

Time for an example

Let’s train a neuron to predict the conversion from celsius into fahrenheit

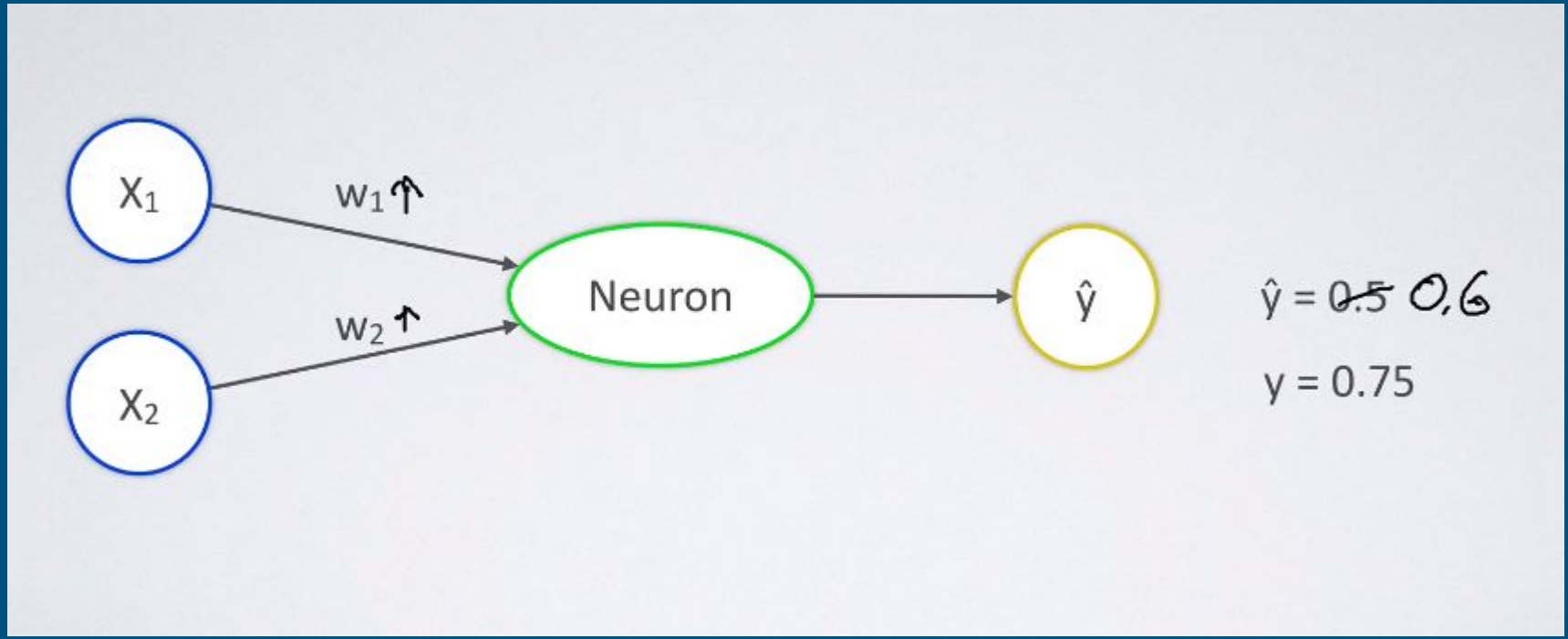
Neuronal Networks



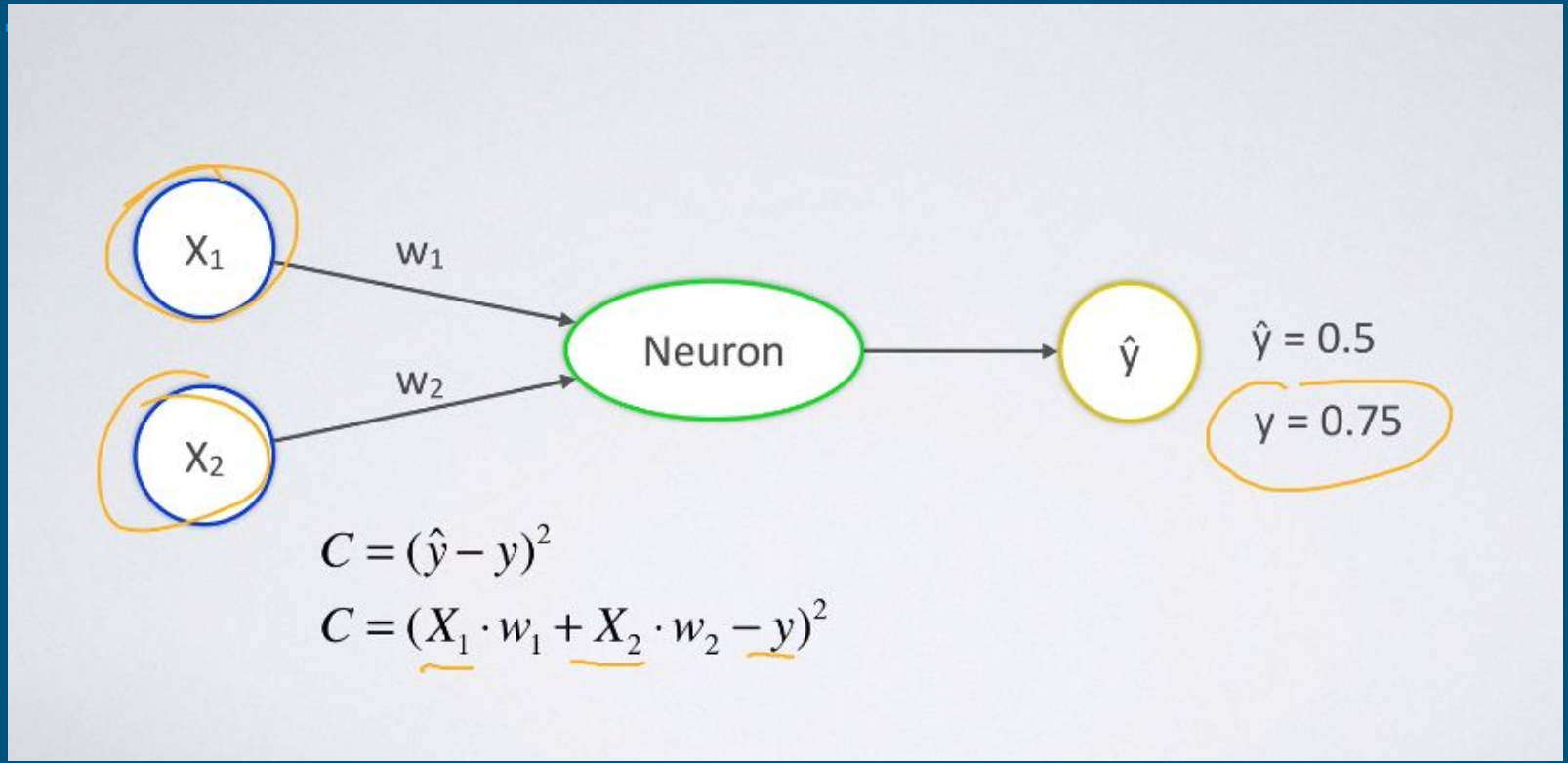
Why we need a hidden layer?

It’s proven that we can approximate each functional problem with a huge hidden layer

Update weights

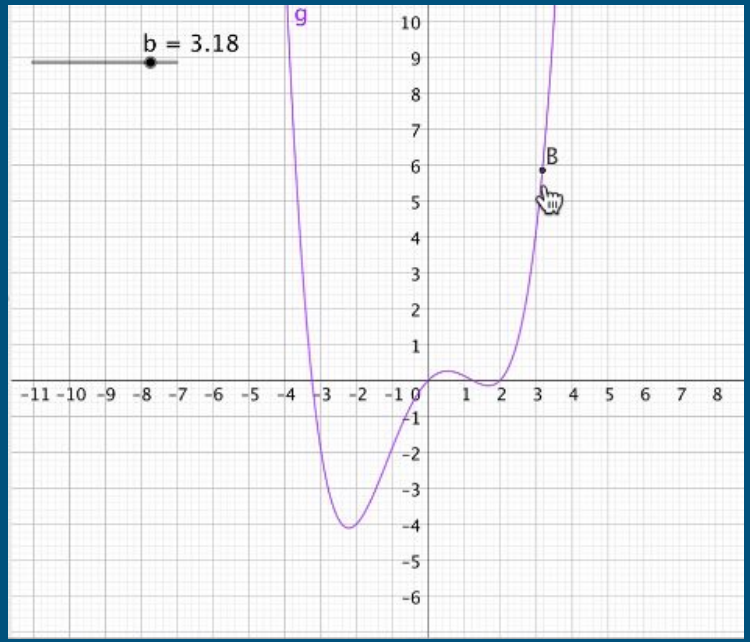


Update weights



The linear gradient thing

- Update weights with linear gradient descent
- Find minimum (cost function)
- “Local minimal” problem (reduce by Many weights)



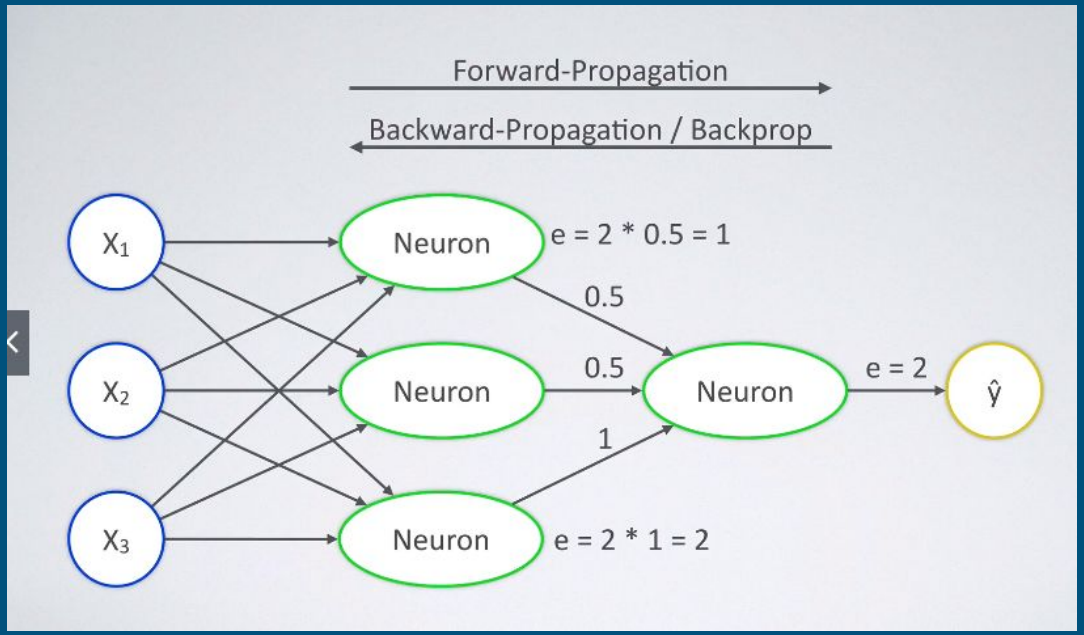
The stochastic linear gradient thing

- Update weights with linear gradient descent after calc each dataset
- To much time!
- Statistical approach

$$C^{approx} = \sum (\hat{y} - y)^2$$

Backpropagation

- How to update weights from the previous layer?



Real data example

Rain prediction in australia