Artificial Intelligence II

Lesson 7 - Neural Networks

First Code Academy

Today's Plan

Teach Back	00 - 5 min
Neuron	10 - 15 min
Neural Network	15 - 20 min
Quiz	20-25 min
Break	25 - 28 min
Project - Learning XOR	28 - 55 min
Lesson Recap	55 - 60 min

Teach Back

_____ learning uses labeled data so we can see how our model is doing during the training phase.

The _ learning algorithm measures the quality of choosing a certain action in a given state.

What did we learn last time?

Supervised Learning

Supervised Learning means that we know the intended answer during training.

We have a **reference** answer during training.

Labeling Data

However, labeling data can be a time-consuming and expensive process

A lot of it is done by hand!

Exploration vs. Exploitation

Our model wants to **explore** new strategies but also **exploit** the reward

An optimal strategy might involve short-term losses

Q-Learning

In Q-Learning, we have a function that gives us the "quality" of taking an action at a state

$$Q(s, a)$$

If at state *s* we take action *a*, what is the value of **all the following actions**?

Key Terms

Neuron

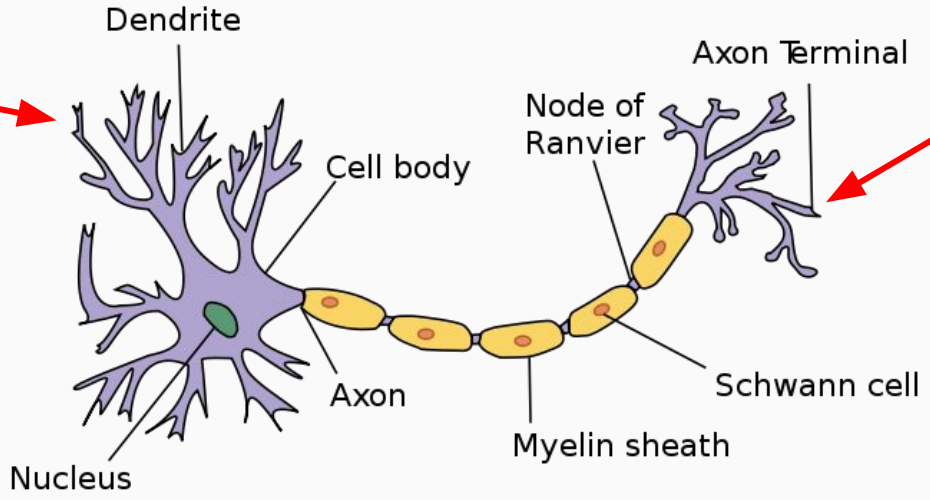
Neural Network

Neuron

Neuron

A **neuron** is the type of cell found in your brain!

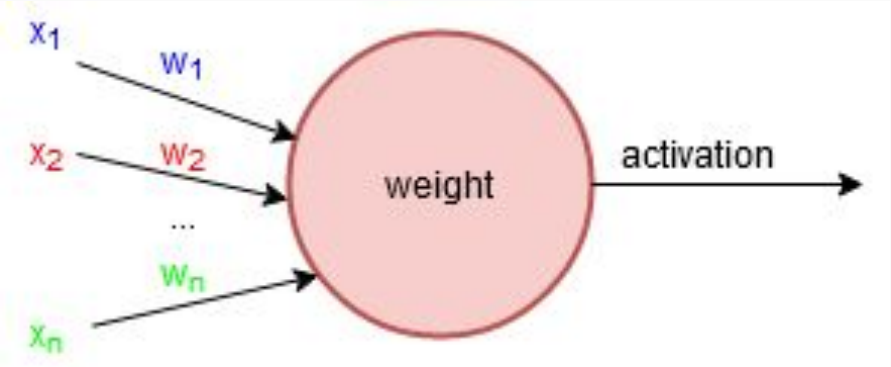
A neuron receives a signal from other neurons...



...and sends it to other neurons after moving through the cell body

Neuron

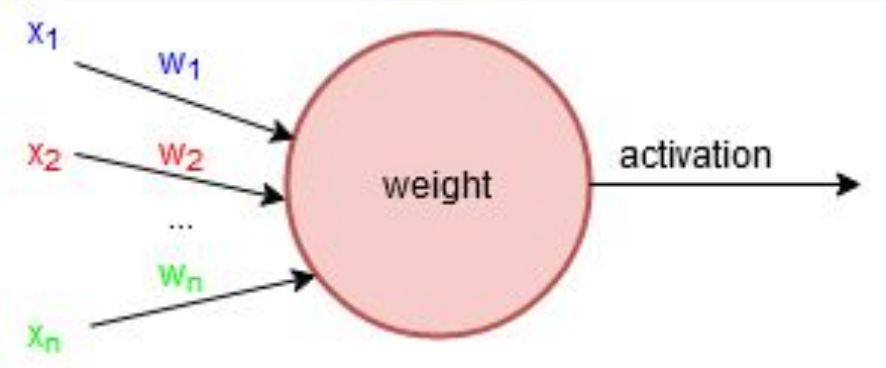
In AI, a “neuron” takes a sum of incoming values and multiplies each with a unique **weight**



The **activation** is how “sure” our neuron is

Neuron

We first add all the inputs and their **weights**



$$sum = w_1x_1 + w_2x_2 + w_3x_3$$

Neuron

Then, we pass the result through an **activation function**

This tells us the final estimate from our model

```
output = activation(sum)
```

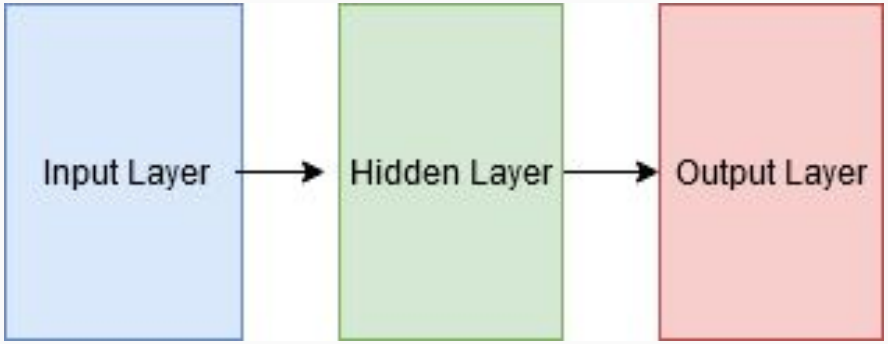
The **output** could be how sure we are that a pattern is there

Neural Network

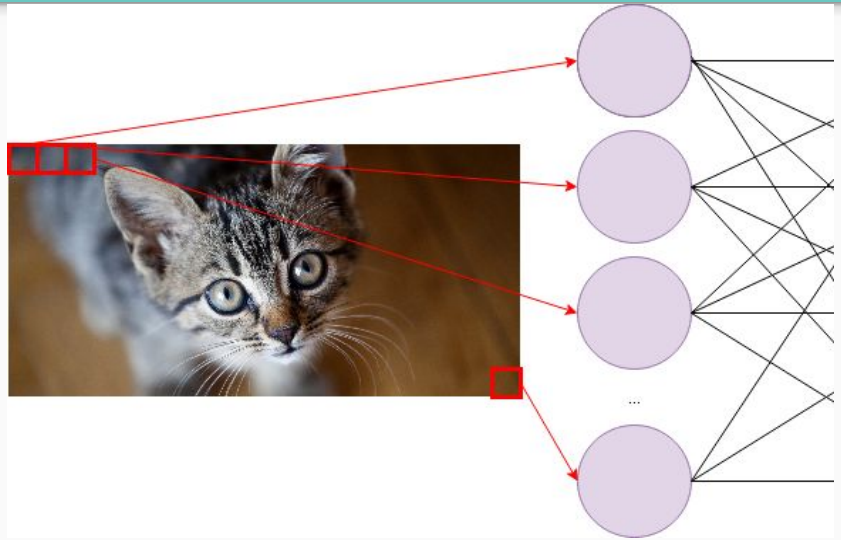
Neural Networks

How could we connect multiple neurons to each other?

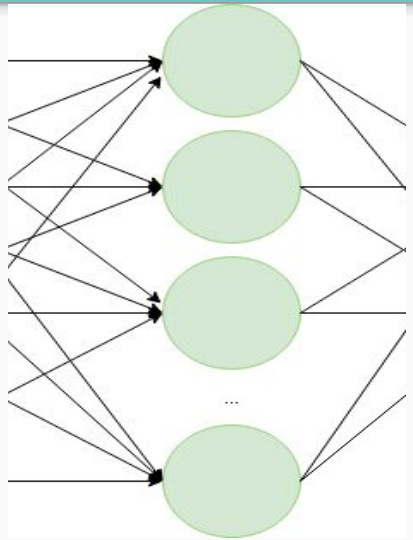
We do so in **layers**.



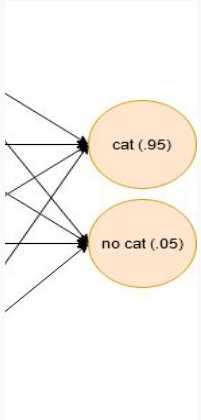
Neural Networks



Input Layer takes in information from outside world



Hidden Layer(s) do extra processing if we need it.



Output Layer classifies the input and tells us how confident it is in the result.

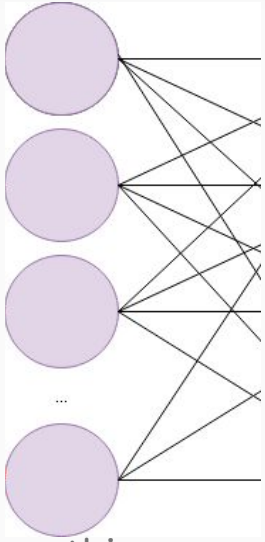
Backpropagation

How do we know how accurate our model is? How do we improve it.

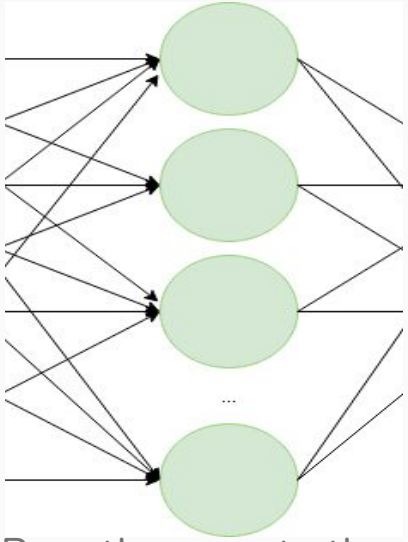
Backward **propagation of errors.**

Similar to **gradient descent**

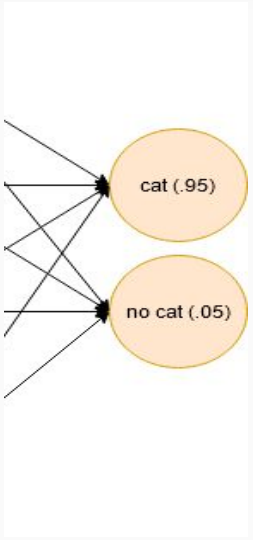
Backpropagation



Continue this way until the input layer.



Pass the error to the previous layer and calculate this layer's



First calculate the **error** in the output layer.

Quiz: bit.ly/FCA_Quiz_AI

Project : Exclusive Or

Exclusive OR

Exclusive or is a type of **logic function**

Similar to AND, OR, etc...

A	B	A XOR B
True	True	False
True	False	True
False	True	True
False	False	False

Exclusive Or

We will write a simple neural network to learn exclusive or!

We will only use an input layer and output layer.

Get the starter file

http://bit.ly/FCA_AI2_Starter

Main Approach

1. Set our weights to random
2. Multiply our training input with our weights, pass them along to the second layer.
3. Adjust our weights, and repeat the process.

Forward Information

First, we pass the information through our network.

```
54  #Start training
55  for i in range(epochs):
56
57      # Forward
58      a1, z1, a2, z2 = forward(X, w1, w2)
59
```

Pass Information Through Network

The input layer will apply the first round of multiplications.

```
11  def forward(x, w1, w2, predict=False):
12      # Multiply our test info with the input layer
13      a1 = np.matmul(x, w1)
14      z1 = sigmoid(a1)
15
```

Pass Information Through Network

Add a bias and pass our testing information to the output layer.

```
16      # Create and add bias
17      bias = np.ones((len(z1), 1))
18      z1 = np.concatenate((bias, z1), axis=1)
19      a2 = np.matmul(z1, w2)
20      z2 = sigmoid(a2)
21
```

Calculate the Backpropagation

Get the amount we need to modify our weights by with **backpropagation**.

```
61      # Calculate the error with backpropagation
62      delta2, Delta1, Delta2 = backprop(a2, X, z1, z2, y)
63
```

Adjusting the Weights

We adjust the weights based on the value given by backpropagation.

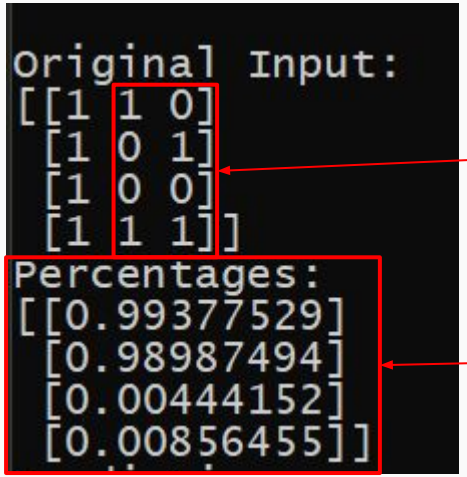
```
64      # Modify our weights
65      w1 -= LEARNING_RATE * (1 / m) * Delta1
66      w2 -= LEARNING_RATE * (1 / m) * Delta2
67
68      # Add the costs
69      c = np.mean(np.abs(delta2))
70      costs.append(c)
71
```

Predict the value

Once we adjusted the weights for a long time, see if we can predict the XOR function

```
76
77    # Make predictions
78    z3 = forward(X, w1, w2, True)
79    print('Percentages: ')
80    print(z3)
81    print('Predictions: ')
82    print(np.round(z3))
```

Try it out!



Values of A and B (0 = False, 1 is True)

How confident we are in our guess.

That’s it for today!

Key Terms

Neuron

Neural Network

Artificial Intelligence II

Lesson 7 - Neural Networks

First Code Academy