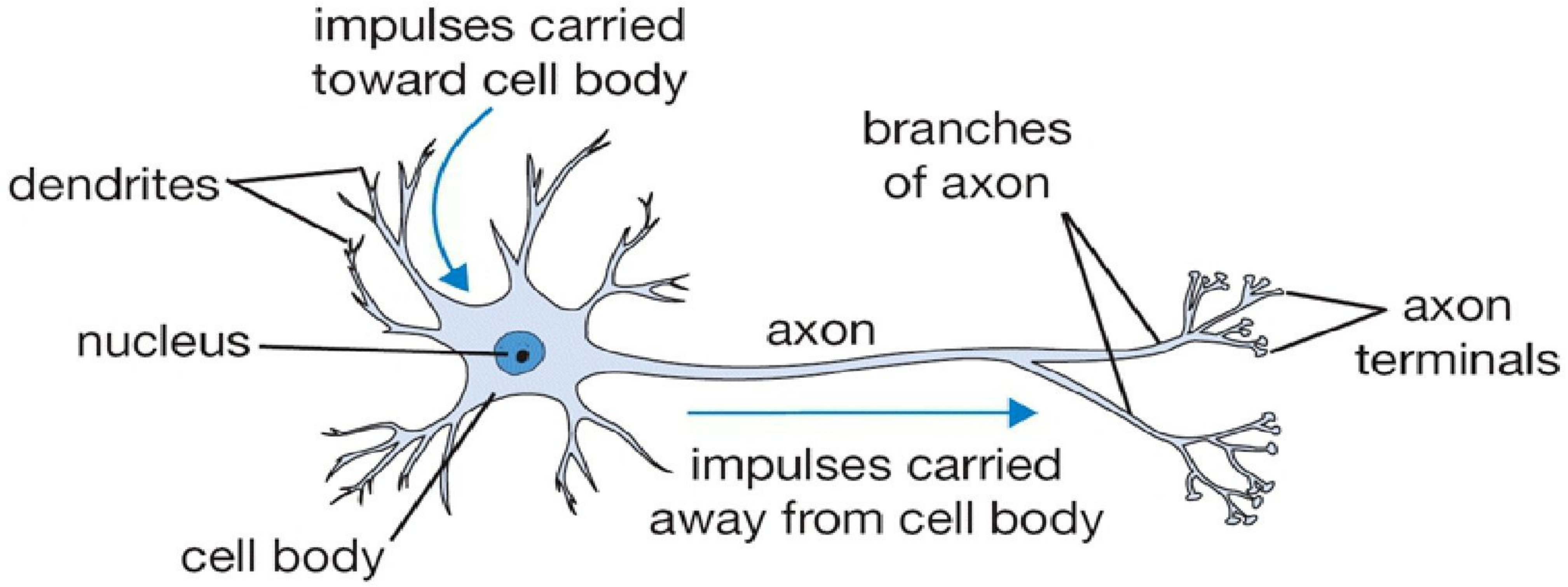
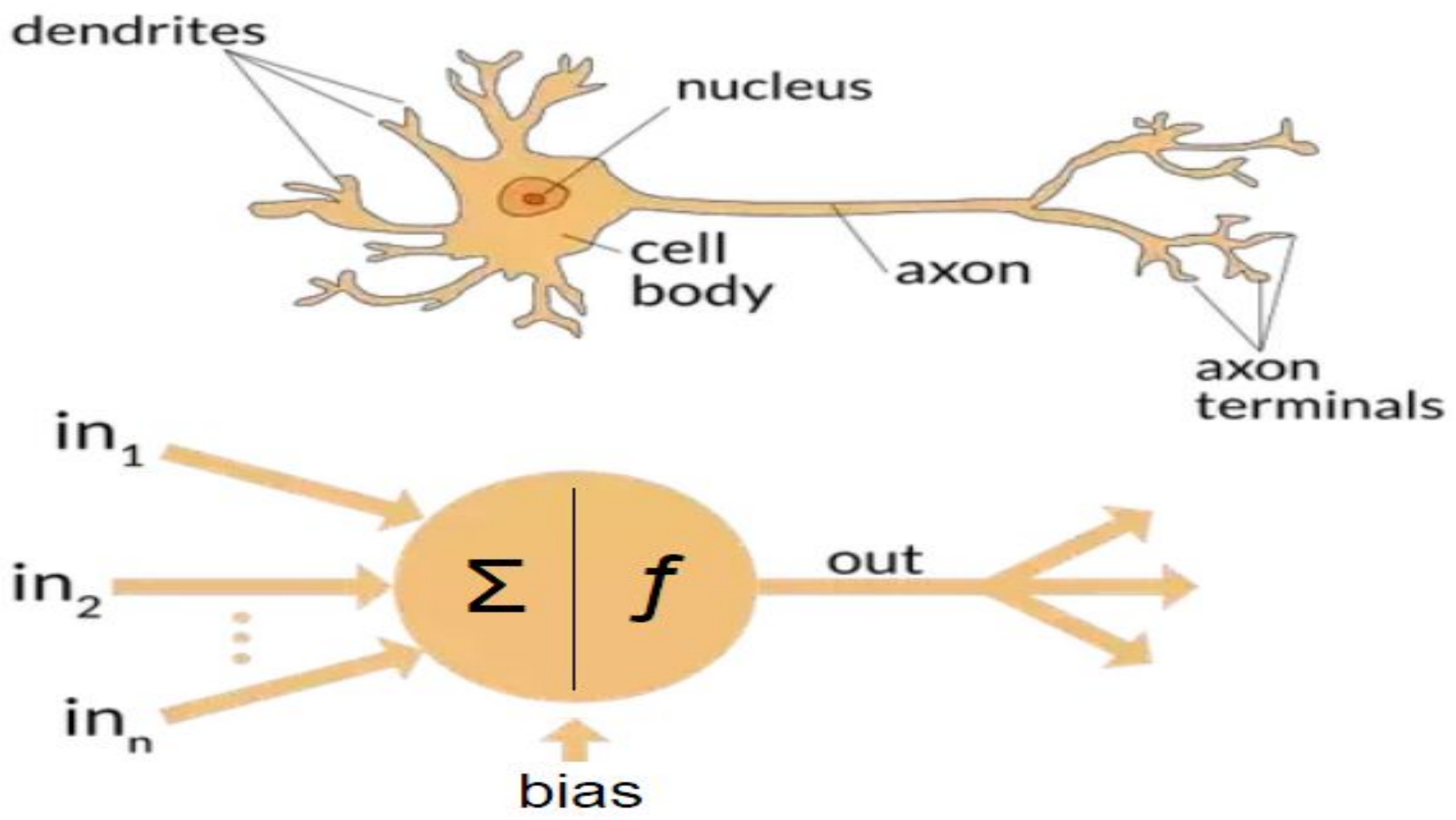
Artificial Neural Network

Artificial Neural Networks — Mapping the Human Brain

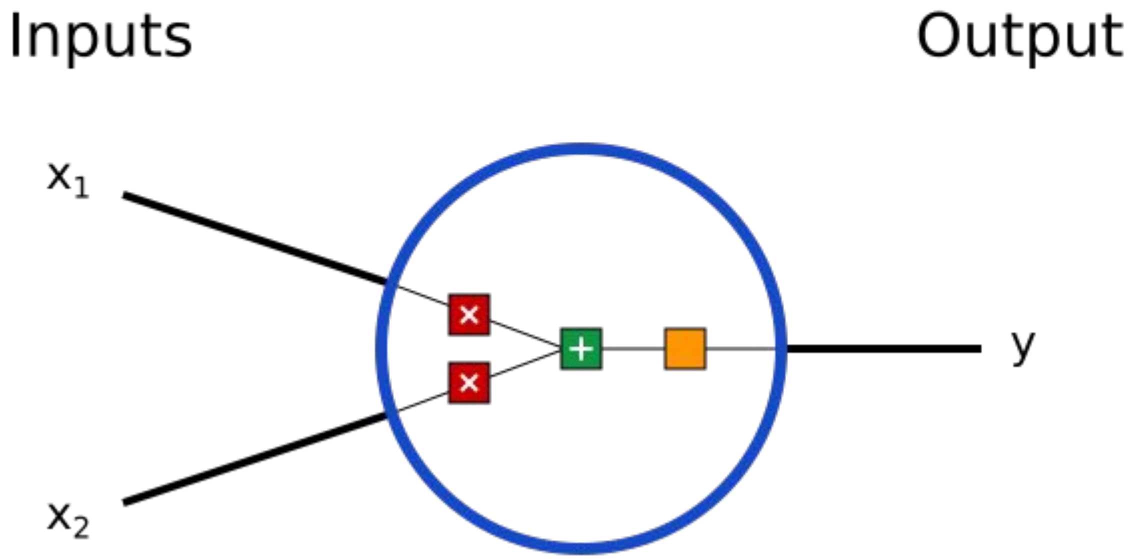

impulses carried toward cell body
dendrites
branches of axon
nucleus
axon
axon terminals
impulses carried away from cell body
cell body

Artificial Neural Networks — Mapping the Human Brain

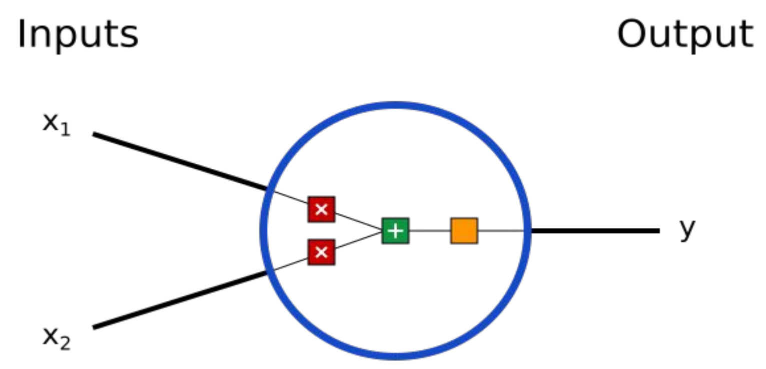


Building Blocks: Neurons

- **Neuron is a basic unit of neural network. A neuron takes inputs, does some math with them, and produces one output**. Here’s what a 2-input neuron looks like:



What is happening here?



- 3 steps are performing here
- 1st step: each input is multiplied by a weight

  x1 -> x1*w1

  x2 -> x2*w2

- 2nd step: weighted inputs are added together with a bias b

  (x1*w1)+(x2*w2)+b

- 3rd step: Finally, the sum is passed through an activation function

  y=f((x1*w1)+(x2*w2)+b)

What are weights(parameters)?

- A weight brings down the importance of the input value.
- Weights near zero means changing this input will not change the output.
- Negative weights mean increasing this input will decrease the output.
- A weight decides how much influence the input will have on the output.

Understanding weights(parameters) with a simple example:

Say you want to decide whether you are going to attend a festival this upcoming weekend. There are three variables that go into your decision:

1. Is the weather good?
2. Does your friend want to go with you?
3. Is it near public transportation?

Write the answers to these question as binary variables xi , with zero being the answer ‘no’ and one being the answer ‘yes’:

1. Is the weather good? x1
2. Does your friend want to go with you? x2
3. Is it near public transportation? x3

We could determine weights w_i indicating how important each feature is to whether you would like to attend. We can then see if:
$x1 \cdot w1 + x2 \cdot w2 + x3 \cdot w3 \geq \text{threshold}$
For some pre-determined threshold. If this statement is true, we would attend the festival, and otherwise we would not.

For example, if we really hated bad weather but care less about going with our friend and public transit, we could pick the weights 6, 2 and 2.

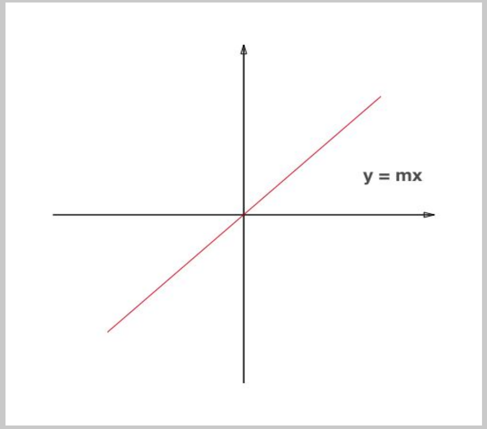
With a threshold of 5, this causes us to go if and only if the weather is good.

What is bias?

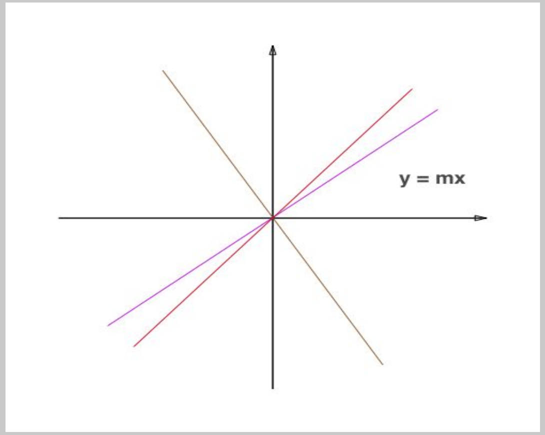
- **Bias** is one of the important terminologies in machine learning. We add bias while creating any model in the artificial neural network.
- It is an extra input to neurons, and has it’s own connection weight.
- **Bias** is a constant which helps the model in a way that it can fit best for the given data.

Understanding bias in simple words

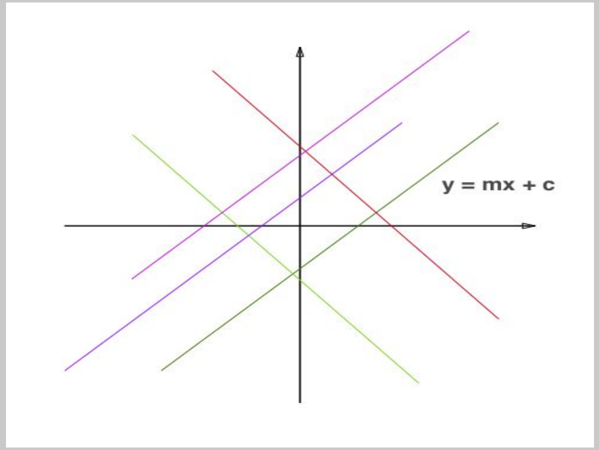
- In the neural network, we are given the input (x) and for that input, we need to predict the output (y). Here, we create a model (mx+c), which predicts the output.
- While training, the model itself finds the appropriate value of the constants m and c.
- Let's say we have the models as y=mx and y=mx+c, so what is the difference between them?



Here, the model is having constraint to train itself and find a line which passes only through the origin.



Many times for the given data, it is impossible for the algorithm to fit the model so that it passes through the origin.



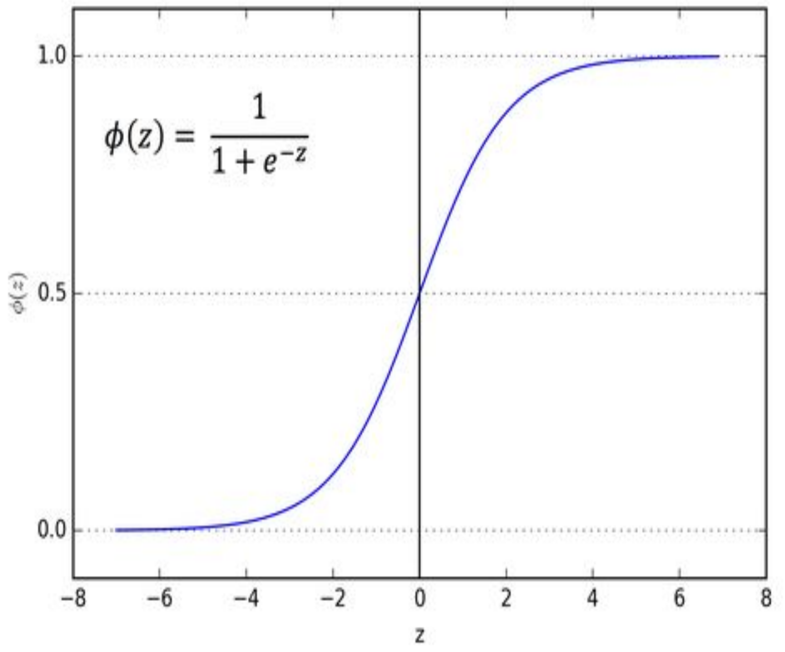
- Now, it is having the full freedom to train itself and find a model which fits the best for the given data.
- Here, the constant c, is the **bias**.
- **Bias** is a constant which helps the model in a way that it can fit best for the given data.

What is activation function?

- What function to use to go from x · w + b to an output is called the activation function.
- It is used to get the output of node. It is also known as **Transfer Function**.
- It is used to determine the output of neural network like yes or no. It maps the resulting values in between 0 to 1 or -1 to 1 etc. (depending upon the function).

What is sigmoid activation function?

- It is also called logistic activation function.
- The sigmoid function curve looks like a S-shape.
- It exists between **(0 to 1).**
- It is especially used for models where we have to **predict the probability** as an output. Since probability of anything exists only between the range of **0 and 1,** sigmoid is the right choice.



A Simple Example

Assume we have a 2-input neuron that uses the sigmoid activation function and has the following parameters:

w=[0, 1] and b=4

w=[0, 1] is just a way of writing *w*1=0, *w*2=1 in vector form.

Now, let's give the neuron an input of *x*=[2, 3]. We'll use the dot product :

$$\begin{aligned}(w \cdot x) + b &= ((w_1 * x_1) + (w_2 * x_2)) + b \\ &= 0 * 2 + 1 * 3 + 4 \\ &= 7\end{aligned}$$

$$y = f(w \cdot x + b) = f(7) = \boxed{0.999}$$

The neuron outputs 0.999 given the inputs *x*=[2,3]. That's it! This process of passing inputs forward to get an output is known as **feedforward**.

Coding a Neuron

Time to implement a neuron! We’ll use NumPy, a popular and powerful computing library for Python, to help us do math:

```
import numpy as np
def sigmoid(x):
# Our activation function: f(x) = 1 / (1 + e^(-x))
return 1 / (1 + np.exp(-x))

class Neuron:
def __init__(self, weights, bias):
self.weights = weights
self.bias = bias

def feedforward(self, inputs):
# Weight inputs, add bias, then use the activation function

total = np.dot(self.weights, inputs) + self.bias
return sigmoid(total)

weights = np.array([0, 1]) # w1 = 0, w2 = 1
bias = 4 # b = 0
n = Neuron(weights, bias)

x = np.array([2, 3]) # x1 = 2, x2 = 3
print(n.feedforward(x)) # 0.9990889488055994
```