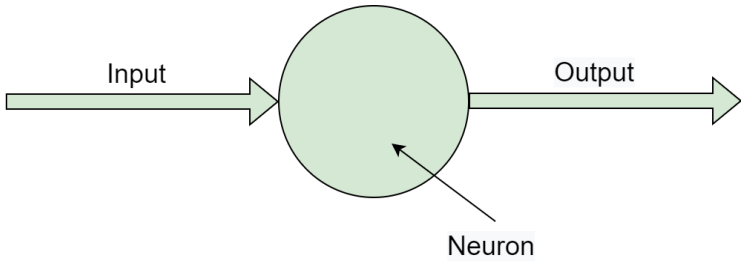
More Neural Networks

So far we've looked at matching X values to Y values when there's a linear relationship between them. So, for example, you matched the X values in this set [-1, 0 , 1, 2, 3, 4] to the Y values in this set [-3, -1, 1, 3, 5, 7] by figuring out the equation Y = 2X - 1.

A very simple neural network with a single neuron within it could be used for this.



This works very well, because, in reality, what is referred to as a 'neuron' here is simply a function that has two learnable parameters, called a 'weight' and a 'bias', where, the output of the neuron will be:

```
Output = (Weight * Input) + Bias
```

So, for learning the linear relationship between our Xs and Ys, this maps perfectly, where we want the weight to be learned as '2', and the bias as '-1'. We saw this in the tutorial.

When multiple neurons work together in layers, the learned weights and biases across these layers can have the effect of letting the neural network learn more complex patterns.

In our first Neural Network we saw neurons that were densely connected to each other. This introduced the **Dense layer** type. There are additional layer types, a few of which are described below:

1. **Convolutional layers** contain filters that can be used to transform data. The values of these filters will be learned in the same way as the parameters in the Dense neuron. Thus, a network containing them can learn how to transform data effectively. This is especially useful in Computer Vision applications. We'll even use these convolutional layers that are typically used for vision models to do speech detection!

2. **Recurrent layers** learn about the relationships between pieces of data in a sequence. There are many types of recurrent layer, with a popular one called LSTM (Long, Short Term Memory), being particularly effective. Recurrent layers are useful for predicting sequence data (like the weather), or understanding text.

We'll also encounter layer types that don't learn parameters themselves, but which can affect other layers. These include layers like **dropouts**, which are used to reduce the density of connection between dense layers to make them more efficient, **pooling** which can be used to reduce the amount of data flowing through the network to remove unnecessary information, and **lambda layers** that allow you to execute arbitrary code.