ARTIFICIAL NEURAL NETWORK

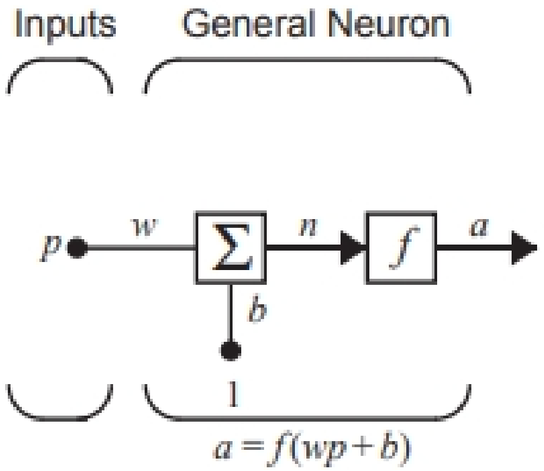
in Python LANGUAGE

Chapter 2.1: Neurons & Layer of neurons

2.1. Neurons and Layer of neurons

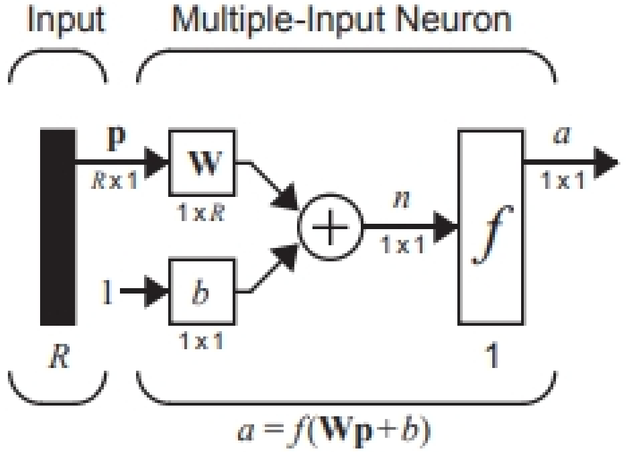
- **Neuron:**

A **neuron** is characterized by its **weight** (represent the strength of a synapse), **the summation and the activation functions** (represent the cell's body) and the **output signal** (represents the **axon signal**).



Single input neuron

Image from "Neural Network Design", M. T. Hagan et al.



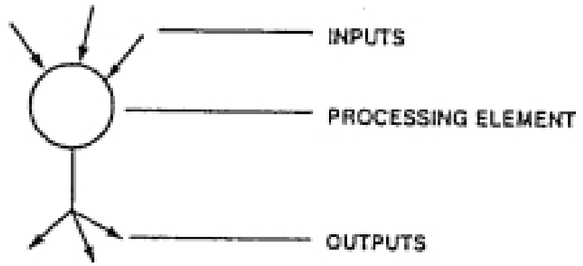
Multi inputs neuron

Image from "Neural Network Design", M. T. Hagan et al.

2.1. Neurons and Layer of neurons

- **Neuron:**

Each neuron has many signal inputs and **one single output**. The output is copied and connect to the inputs of the neural of the next layer. The signal path is **one way. The neurons operates asynchronously.**



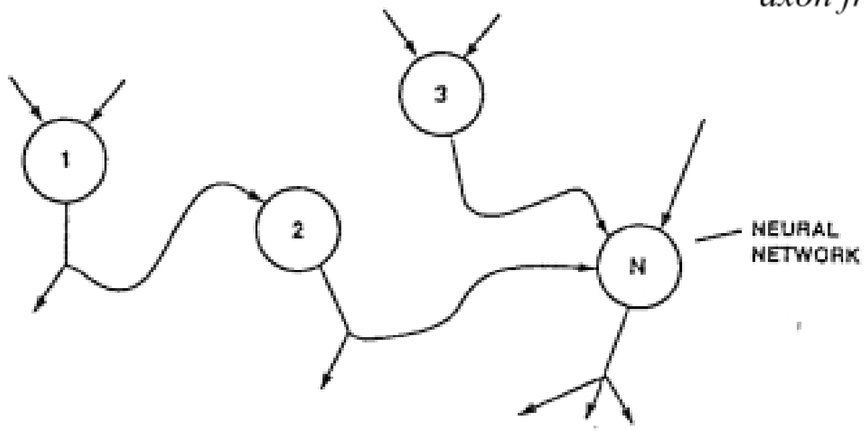
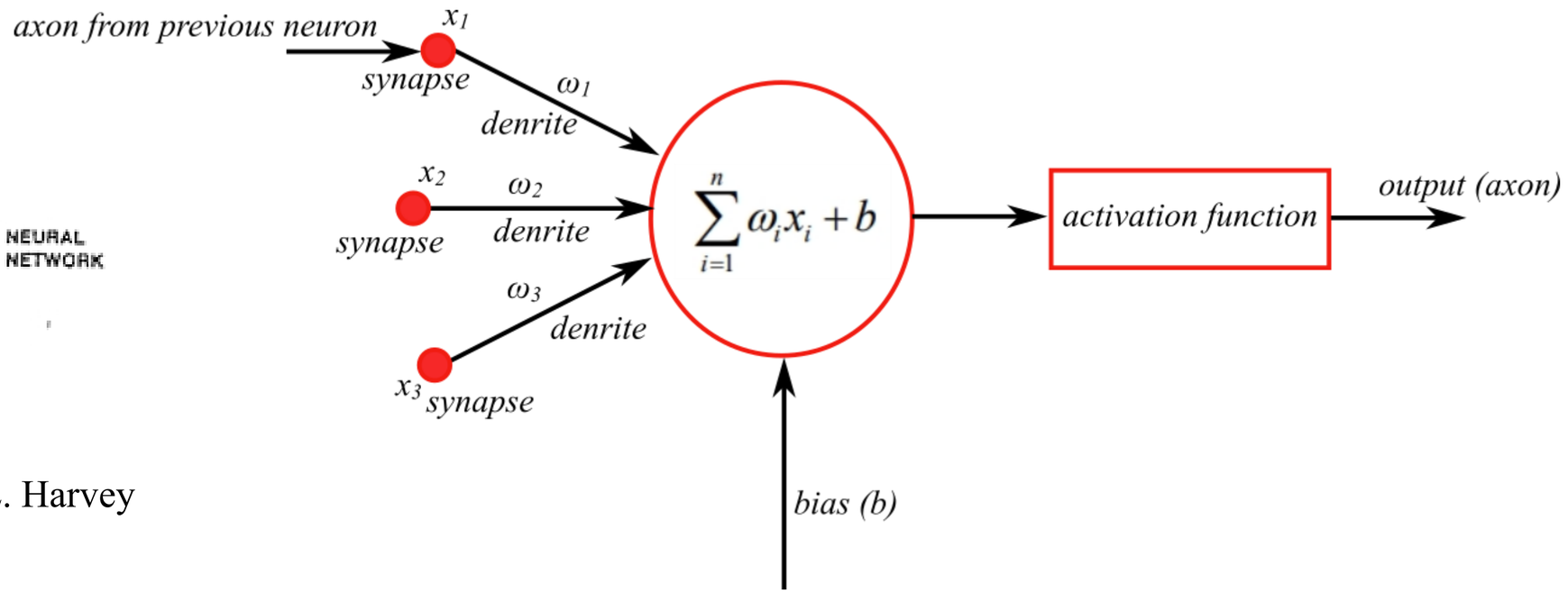


Image from "Neural Network Principal", R. L. Harvey



2.1. Neurons and Layer of neurons

- **Neuron:**

How to code a neuron in Python:

To define a neuron, one just need to define the vectors of inputs & weights as well as the bias value.

```
input = [1, 2, 3]
weight = [0.2, 0.8, -0.5]
bias = 2
```

The output value of the neuron can then be computed by the dot product of inputs and weights, plus the bias value:

```
output = (input[0]*weight[0]+input[1]*weight[1]+input[2]*weight[2]+bias)
```

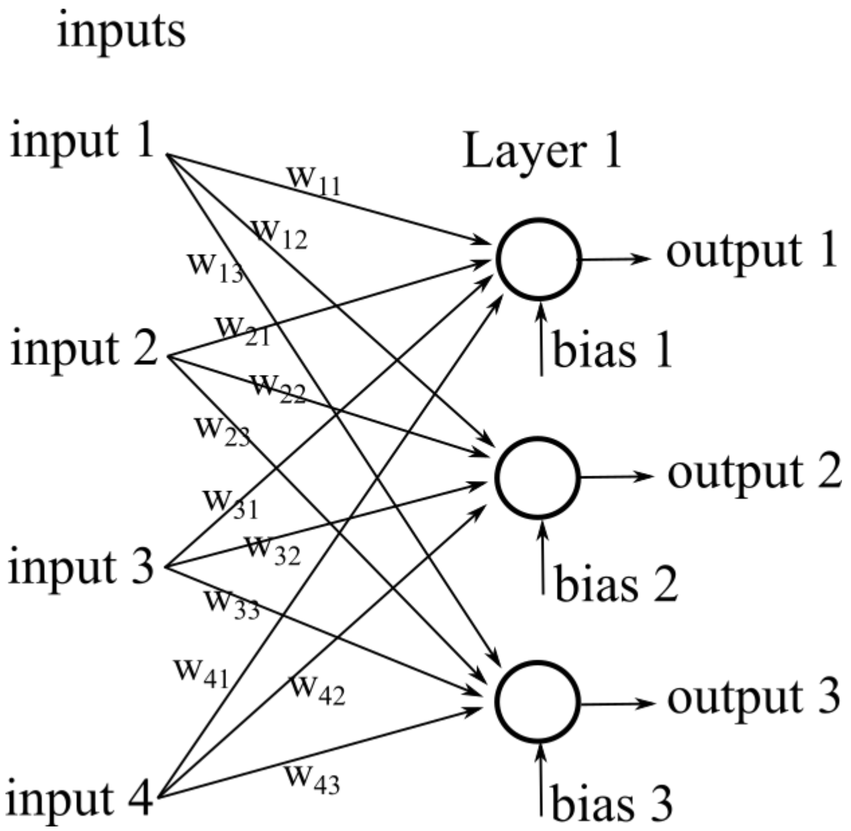
→ `outputs = np.dot(weights,input) + bias;`

Sử dụng hàm dot product của numpy

2.1. Neurons and Layer of neurons

- **Layer of neurons:**

A **layer of neuron** is a group a neurons. In a layer, each neuron has **exactly the same inputs**. Meanwhile, the **weights** of the connection between an input and the neurons are **different** for each neuron.



2.1. Neurons and Layer of neurons

- **Layer of neurons:**

Example of code in Python: A layer of 4 inputs & 3 neurons

```
input = [1.0, 2.0, 3.0, 2.5]
weights = [[0.2, 0.8, -0.5, 1.0],[0.5, -0.91, 0.26, -0.5],[-
0.26, -0.27, 0.17, 0.87]]
biases = [2,3,0.5]

outputs = np.dot(weights,input) + biases;
```

2.1. Neurons and Layer of neurons

- **Layer of neurons:**

Example of code in Python: A layer of 4 inputs & 3 neurons. Inputs are in the form of batch of data

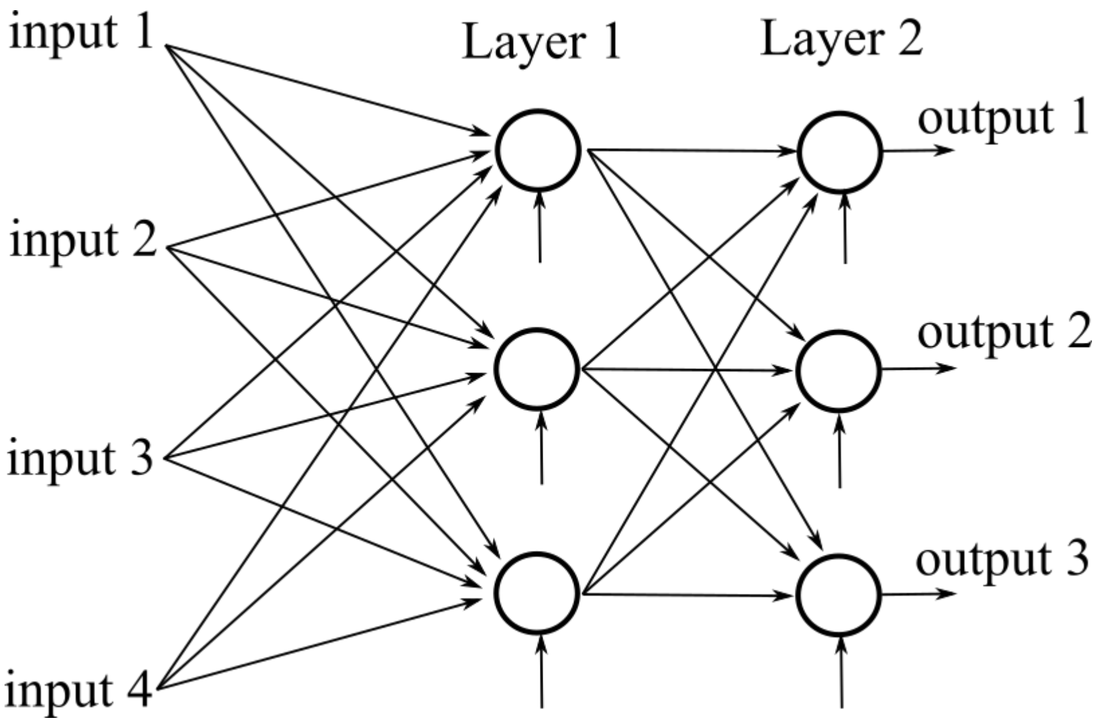
```
# input batch of data - 3 neurons - 4 inputs
import numpy as np
inputs = [[1,2,3,2.5],[2,5,-1,2],[-1.5,2.7,3.3,-0.8]]
weights = [[0.2,0.8,-0.5,1],[0.5,-0.91,0.26,-0.5],[-0.26,-0.27,0.17,0.87]]
biases = [2,3,0.5]
outputs = np.dot(inputs,np.array(weights).T) + biases
```

2.1. Neurons and Layer of neurons

- **Layer of neurons:**

Example of code in Python:

4 inputs & 2 layers of neurons




```
# 2 layers 3 neurons each - 4 inputs (batch)
import numpy as np

inputs = [[1,2,3,2.5],
          [2,5,-1,2],
          [-1.5,2.7,3.3,-0.8]]

#layer 1 -- 3 neurons
weights1 = [[0.2,0.8,-0.5,1],
            [0.5,-0.91,0.26,-0.5],
            [-0.26,-0.27,0.17,0.87]]
biases1 = [2,3,0.5]

#layer 2 -- 3 neurons
weights2 = [[0.1,-0.14,0.5],
            [-0.5,0.12,-0.33],
            [-0.44,0.73,-0.13]]
biaises2 = [-1,2,-0.5]

#Feedforward calculation
layer1_outputs = np.dot(inputs,np.array(weights1).T) + biases1
layer2_outputs = np.dot(layer1_outputs,np.array(weights2).T) + biaises2
```

2.1. Neurons and Layer of neurons

- **Layer of neurons:**

Example of code in Python: Write a class Dense

```
class Dense:
    def __init__(self, n_inputs, n_neurons):
        #init eights and biases
        self.weights = 0.01*np.random.randn(n_inputs,n_neurons)
        self.biaises = np.zeros((1,n_neurons))

    def forward(self,inputs):
        #calculate outputs
        self.output = np.dot(inputs,self.weights) +self.biaises
```

Define a layer of neuron using the predefined class Dense:

```
denses1 = Layer_Dense(2,3)
denses1.forward(X)
```

Artificial Neural Network

END OF CHAPTER 2 – Part 1