Lecture 3: Neural Networks

PUSL3123 AI and Machine Learning

Neamah Al-Naffakh

School of Engineering, Computing and Mathematics

Neamah.al-naffakh@plymouth.ac.uk

Today’s Topics

Neural Networks

Lesson learning outcomes: By the end of today’s lesson, you would be able to:

- Understand the Concepts of Artificial Neural Networks (ANN)
- Implement ANN using MATLAB

UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

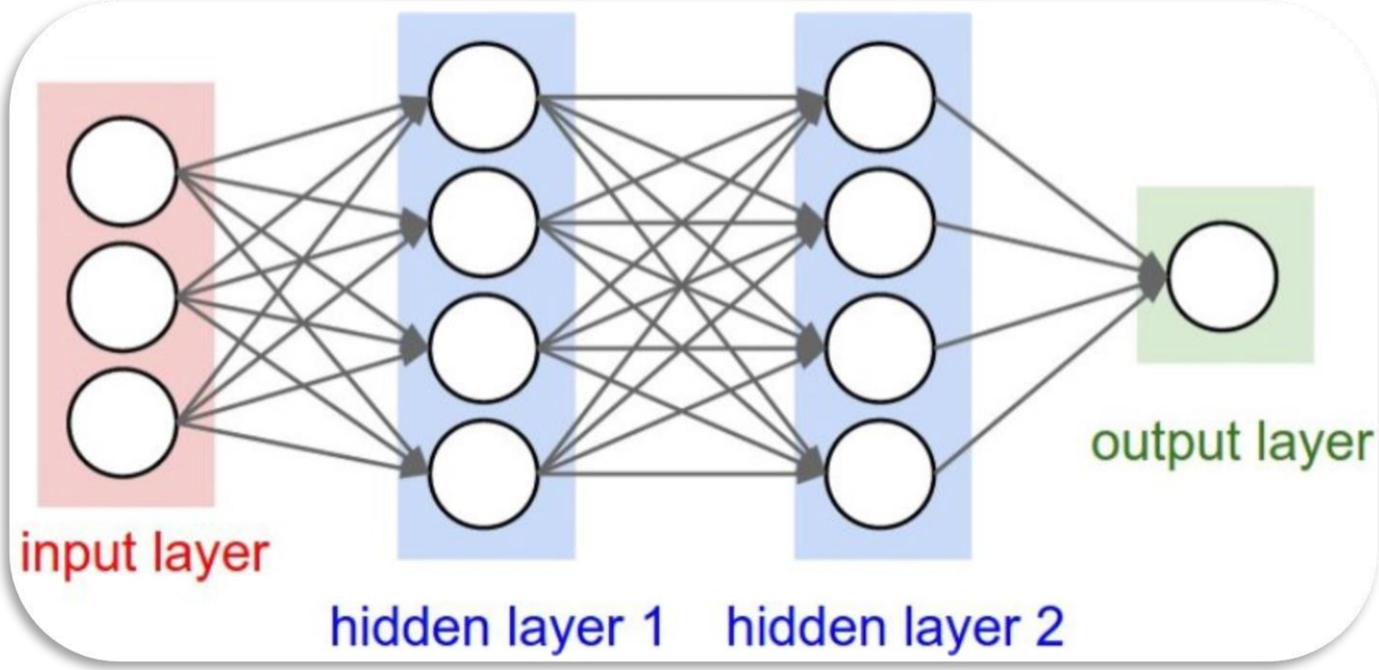
What is Neural networks

- Neural networks reflect the behavior of the human brain, allowing computer programs to recognize patterns and solve common problems in the fields of AI, machine learning, and deep learning.

- Neural networks can be expressed by many layers, i.e. input, output and hidden layers.

- NNs can be used for supervised and unsupervised learning.

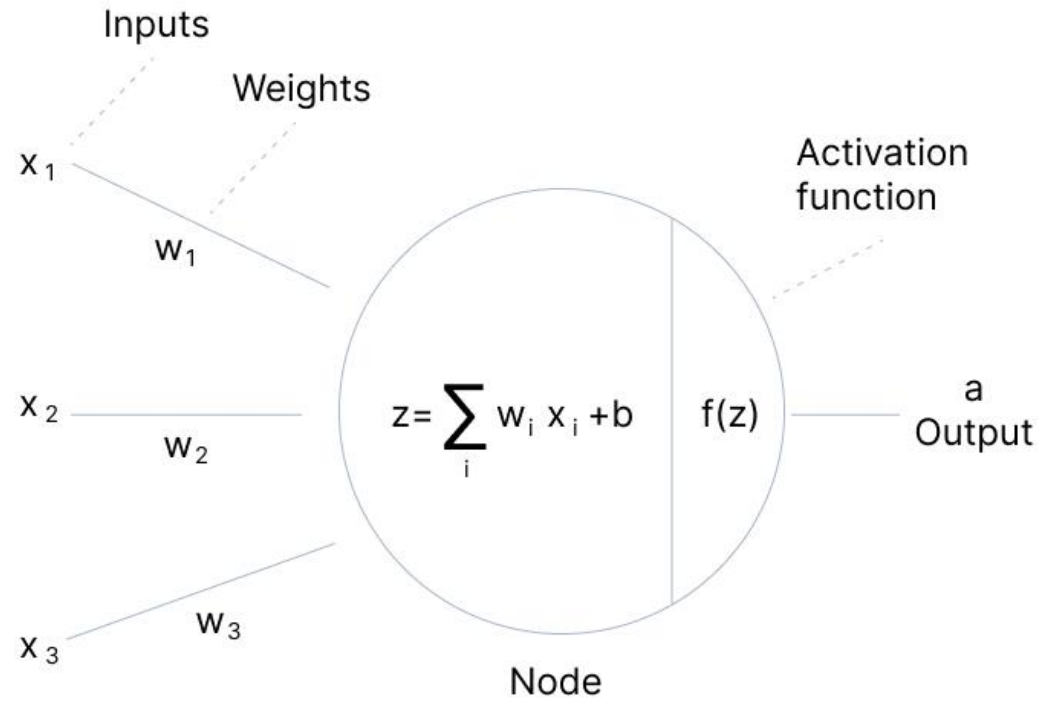
Key Components of the Neural Network Architecture



- **Input layer** - It is the set of features that are fed into the model for the learning process
- **Hidden Layers -** they are intermediate layers that do all the computations and extract the features from the data.
- **Output Layer :** the output layer takes input from preceding hidden layers and comes to a final prediction based on the model's learnings.

Key Components of the Neural Network Architecture

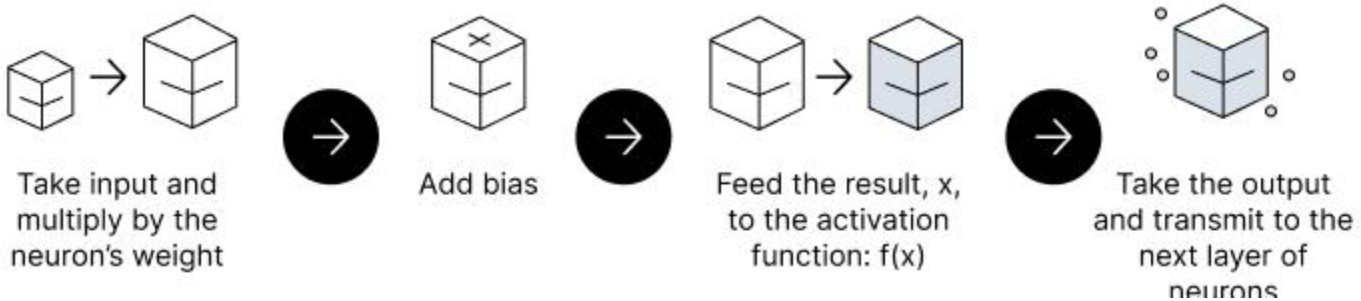
- **Connections**—It connects one neuron in one layer to another neuron in other layer or the same layer
- **Weight** - Its main function is to give importance to those features that contribute more towards the learning. A weight represent the strength of the connection between units
- **Bias** - The role of bias is to shift the value produced by the activation function
- **Activation Function** decides whether a neuron should be activated or not- such as Sigmoid, Hyperbolic Tangent (Tanh), Rectified Linear Unit (ReLU),



Feedforward vs. Backpropagation

Feedforward Propagation - the flow of information occurs in the forward direction.

The input is used to calculate some intermediate function in the hidden layer, which is then used to calculate an output.


Take input and multiply by the neuron's weight
Add bias
Feed the result, x, to the activation function: f(x)
Take the output and transmit to the next layer of neurons

Backpropagation - the weights of the network connections are repeatedly adjusted to minimize the difference between the actual output vector of the net and the desired output vector.

Error functions

To train a neural network you must identify a set of weights (and NN structure?) that minimizes an error function

Given a set of **targets** $\{t_n\}_{n=1}^{N}$ and **model outputs** $\{y_n\}_{n=1}^{N}$ compute

Input		Target	Actual	△
0	0	0	0.2	0.2
0	1	1	0.3	0.7
1	0	1	0.4	0.6
1	1	0	0.5	0.5
			MAE=	0.5

Mean absolute error

$$MAE = \frac{\sum_{n=1}^{N} |t_n - y_n|}{N}$$

Root Mean Square Error

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (Predicted_i - Actual_i)^2}{N}}$$

***Predicted*$_i$** *= The predicted value for the ith observation.*

***Actual*$_i$** *= The observed(actual) value for the ith observation*

N *= Total number of observations*

UNIVERSITY OF PLYMOUTH
School of Engineering, Computing and Mathematics

QUIZ!

Which of the following statements does not hold true?

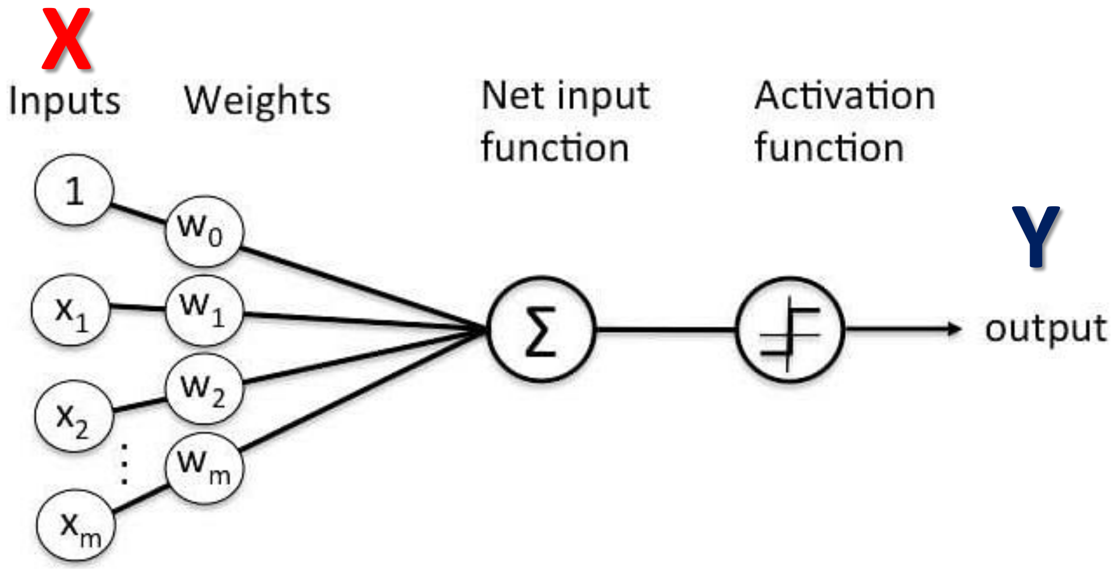
A. Activation functions are threshold functions

B. Error is calculated at each layer of the neural network

C. Both forward and back propagation take place during the training process of a neural network

D. Most of the data processing is carried out in the hidden layers

The perceptron



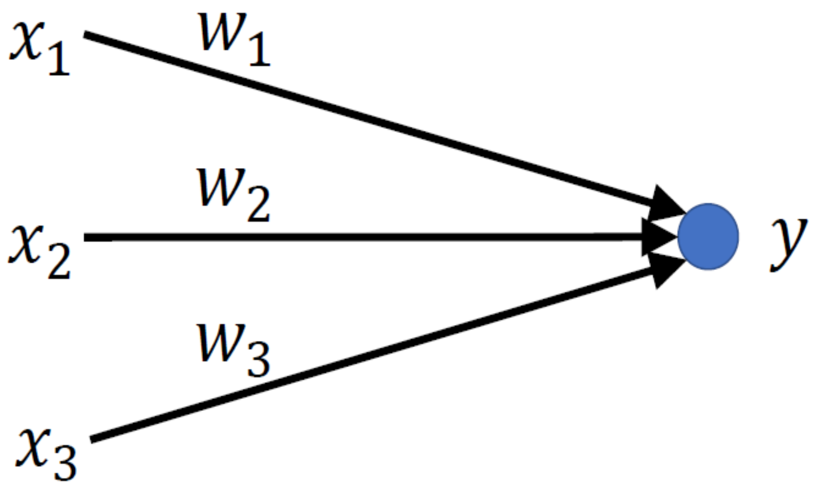
$$y = f(xw)$$

- The **model input** is the *x* variable
- The model has a single **unit** (shown in blue)
- Input is connected to the unit by a **weight**
- Output is a function of the **weighted sum of the input**
- A Perceptron is supervised learning of binary classifiers
- It enables neurons to learn and processes elements in the training set one at a time.
- Single layer and Multilayer are types of Perceptron

How Does Perceptron Work?

Single-layer networks

Step 1: Multiply all input values with corresponding weight values and then add to calculate the weighted sum.



$$y = f\left(\sum_{k=1}^{K} x_k w_k\right)$$

K stands for the number of inputs

y= X_1 W_1 + X_2 W_2 + X_3 W_3 +........ X_k W_k

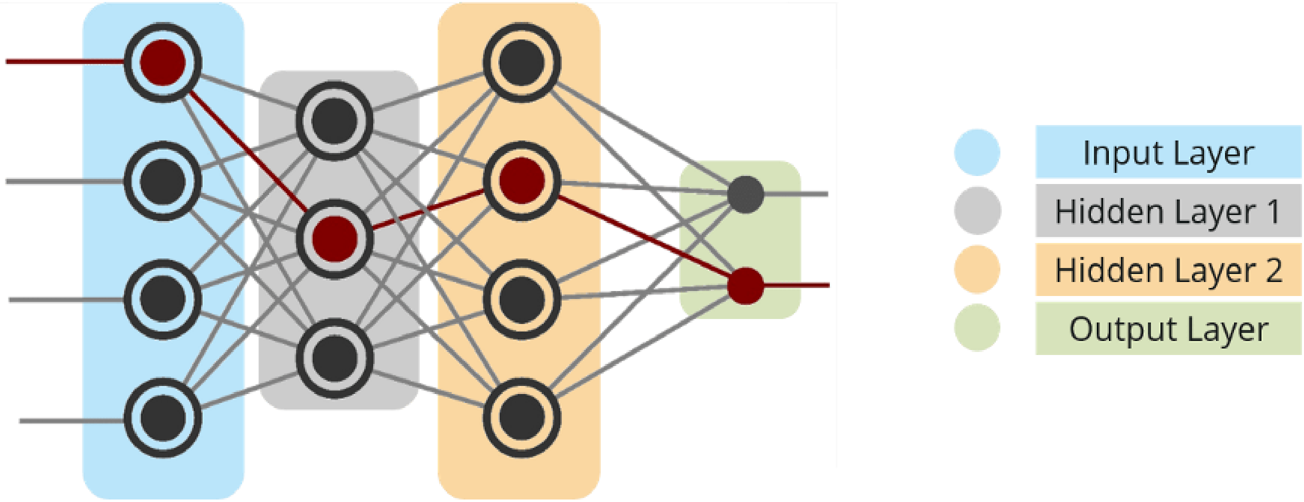
Add bias 'b' to this weighted sum to improve the model's performance

Step 2: An activation function *(f)* is applied with the above-mentioned weighted sum giving us an output either in binary form or a continuous value as follows:

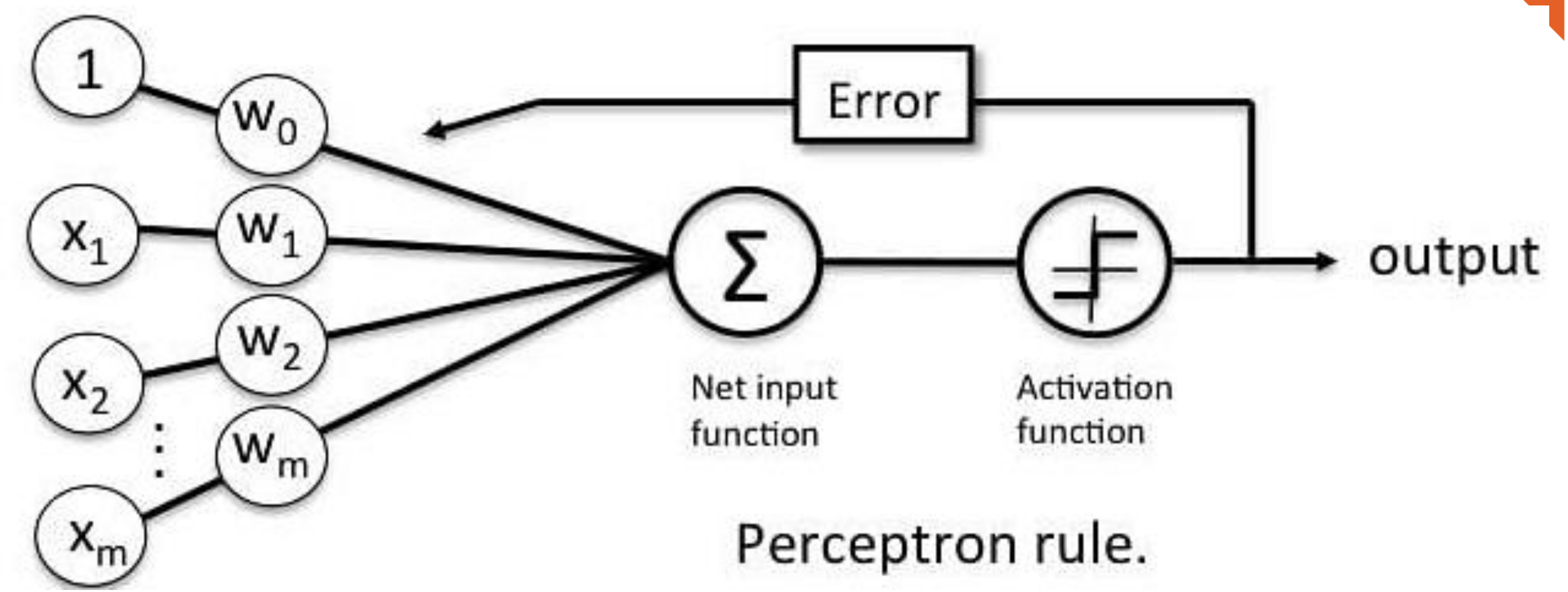
$$Y= f\ (\sum w_i * x_i + b)$$

Multi-Layer Perceptron

- Multi-Layered Perceptron has more hidden layers.
- It can solve complex non-linear problems
- It works well with both small and large input data
- Time-consuming
- The model functioning depends on the quality of training.



Multi-Layer Perceptron



Perceptron rule.

Training: back propagation

Use model error to update the network weights

Back propagation algorithm

```
Initialise all weights to random values
repeat
    for all training examples do
        Forward propagation: pass the training examples through the network to the output
    end for
    Calculate the network error
    Back propagation: update each weight based on a learning term derived from the
    network error
until weights converge (or runtime elapses)
```

Conclusion

- What is Neural Network?
- How Does Perceptron Work?
- Single layer linear network
- Feedforward vs. Backpropagation

Lab 2 Explanation