

ET-287 – Signal Processing using Neural Networks

2. Introduction to Machine Learning and Artificial Neural Networks

Professor Sarah Negreiros de Carvalho Leite

Aeronautics Institute of Technology

Electronic Engineering Division

Department of Telecommunications

sarahnc@ita.br, sarah.leite@gp.ita.br

room: 221



Course Syllabus

Unit 1 – Introduction and brief review of linear algebra using Python.

Unit 2 - Introduction to Machine Learning and Neural Networks.

- a) Learning processes: supervised, semisupervised and unsupervised.
- b) The human brain and models of a neuron.
- c) What is an artificial neural network?
- d) Artificial intelligence and neural networks.
- e) Neural network architecture.
- f) Activation functions
- g) The perceptron.

Unit 3 - Feedforward Neural Networks.

Unit 4 - Recurrent Neural Networks.



Machine Learning

Machines learn through algorithms and data processing techniques.

- Supervised Learning
- Unsupervised Learning
- Semi-supervised Learning
- Reinforcement Learning



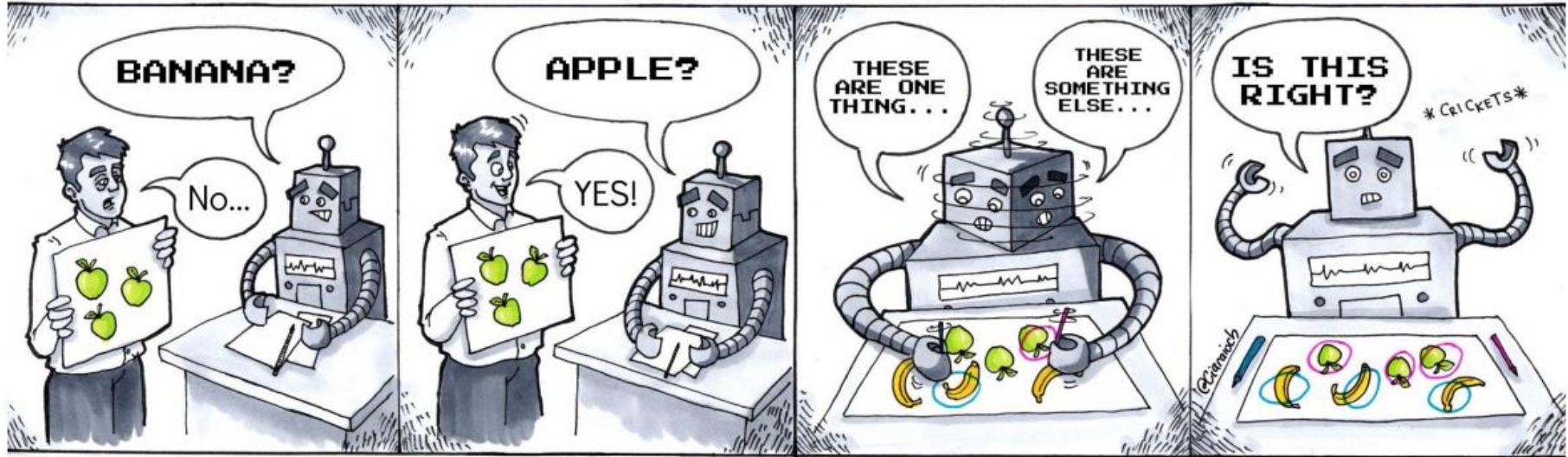
Machine Learning

Supervised Learning: algorithms are trained on labeled data, where each input is paired with the correct output. The algorithm learns to map inputs to outputs, making it suitable for tasks like classification and regression.

Unsupervised Learning: machines are fed with unlabeled data and seek to identify patterns and structures in the data on their own. This can include data clustering, density estimation, and dimensionality reduction.



Machine Learning



Supervised Learning

Unsupervised Learning

Supervised Learning

The dataset can be partitioned disjointly into training, validation, and testing sets. The supervised learning process can be divided into three stages:

- 1. Training:** The algorithm uses the labeled data to learn the mapping between inputs and outputs. It adjusts its internal parameters to minimize the difference between the predicted outputs and the actual labels. During the training stage, a **mathematical model** is generated to relate the input/output behavior, based on a **criterion to measure the model's performance**, such as error in regression problems or accuracy in classification problems.
- 2. Prediction:** After training, the model can make predictions on new, unseen data.
- 3. Evaluation:** The model's predictions on new, unseen data are compared to the actual labels to assess its performance. If the model's performance is not satisfactory, **adjustments** may be made, such as refining the model architecture, collecting more data, or fine-tuning parameters.



Supervised Learning

The database has labeled entries that determine the expected output. The learning stage involves creating a model that maps input data to their respective outputs.

Some examples:

Application	Input	Output
Translator	Text in Portuguese	Text in English
Speech Recognition	Audio	Transcription
Image Recognition	Images	List of Found Objects
Diagnosis of Cardiac Problems	Cardiac Signals	Possible Diseases
Purchase Recommendations	Purchase History	List of Suggested Products

The algorithm must 'learn' from examples of input/output pairs to be able to determine the output for unknown inputs.



Exercise 1

Consider the input values $x = [1, 4, 5, 8, 10]$ and the respective outputs $y = [3, 9, 11, 17, 21]$.

1. What is the relationship between the input and output?
2. How can the machine 'learn' from this data and determine this relationship? Instead of programming a response for all possible inputs, the model seeks to learn the relationship between the provided input/output data and uses it to determine the outputs corresponding to new input data.
3. What happens if the inputs are kept, but the outputs are: $y = [3, 7, 11, 15, 22]$?
4. What will be the system's output for the input $x = 3$ for each case?



Exercise 2

Retake the code from the previous exercise on iris plants.

1. Based on the statistical analyses performed, choose (and justify) just one feature from the data matrix. In this way, we also have an input vector x associated with an output label y .
2. Propose a way for the machine to learn to distinguish between the three flower species.
3. In terms of complexity and accuracy, comment on what happens if we use all four features from the data matrix.



Unsupervised Learning

Unsupervised learning algorithms seek similarities, patterns, and redundancies in the dataset to try to 'learn' about the data behavior. Labels are not used to guide the learning process.

Examples of problems that can be addressed by this approach:

Application	Input
Logistics center	Store locations
Customer profile	Customer usage/purchase history
Classification of series/movies	Series/movies parameters
Diagnosis	Symptoms



Exercise 3

Retake the code from the previous exercise on iris plants and consider only the measurements taken of the length and width of the petals and sepals of the flowers, without taking into account the labels that indicate the species to which each measurement corresponds.

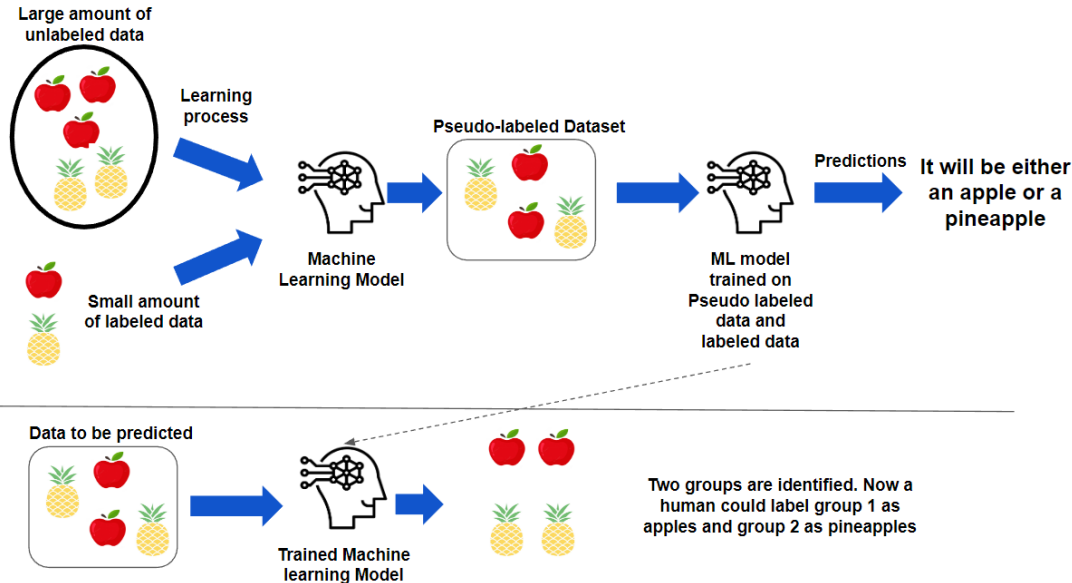
These labels act as a guide, and in unsupervised learning, we don't have this guide.

1. Suggest a way to discover how many plant species there are using only the features matrix x .



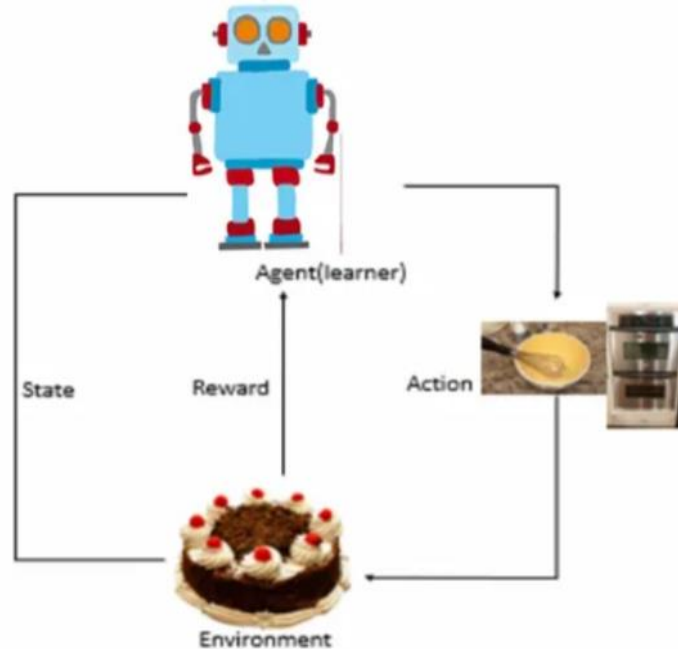
Machine Learning

Semi-Supervised Learning: this technique combines elements of supervised and unsupervised learning, where algorithms are trained on a combination of labeled and unlabeled data. It's useful when obtaining labeled data is costly or time-consuming.

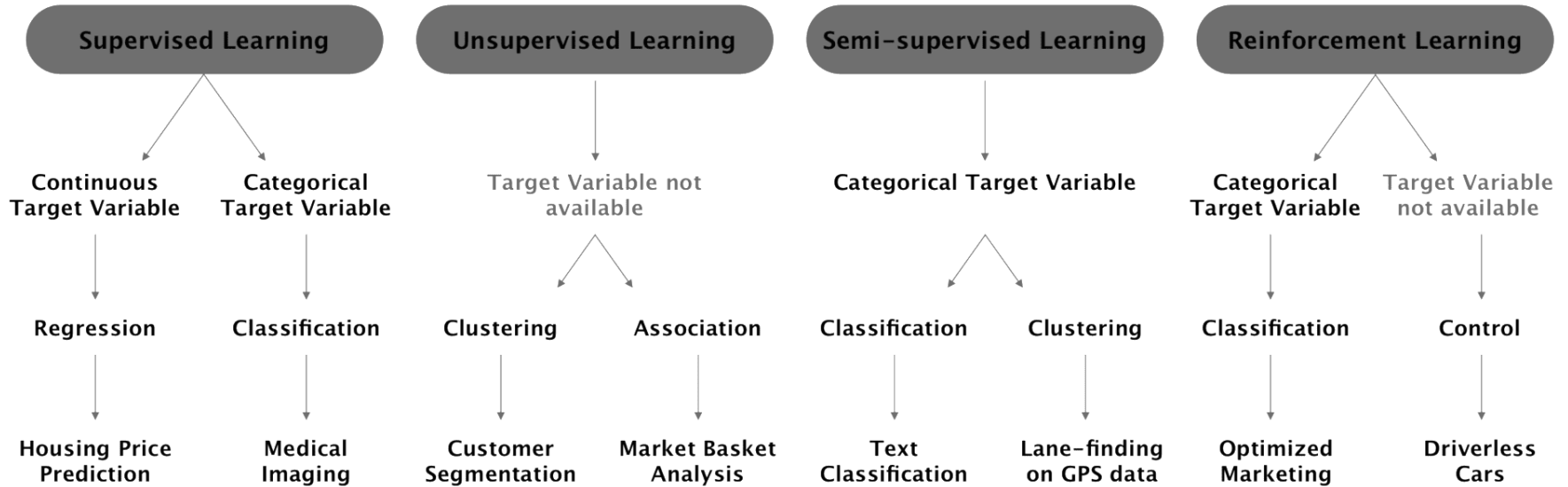


Machine Learning

Reinforcement Learning: agents learn to make a sequence of decisions in an environment to maximize a cumulative reward.



Machine Learning



Neural Network

Each of us has a highly interconnected set of around 10^{11} neurons. The neuron is the fundamental cell that makes up the brain, and each of them is capable of processing received stimuli and connecting with other neurons continuously and in parallel.



A neuron or nerve cell is the fundamental structural and functional unit of the nervous system, including the brain and spinal cord.

They are specialized cells that play a crucial role in transmitting information within the nervous system.

Neurons form the basis for all cognitive processes, such as perception, reasoning, memory and learning.

Source: "<http://www.shutterstock.com/gallery-61166p1.html>"



Neuron

Cell Body (Soma): central part of the neuron and contains the nucleus, which houses the genetic information and controls the cell's activities.

Dendrites: receive electrical signals and chemical messages from other neurons or sensory receptors.

Axon: carries electrical signals (action potentials) away from the cell body and transmits them to other neurons or target cells. They are often covered by a myelin sheath for faster signal conduction.

Myelin Sheath: fatty substance that wraps around some axons forming a protective sheath. It acts as an insulator, speeding up the transmission of electrical impulses along the axon.

Axon Terminals: release neurotransmitters into synapses to transmit signals to the next neuron in the chain.

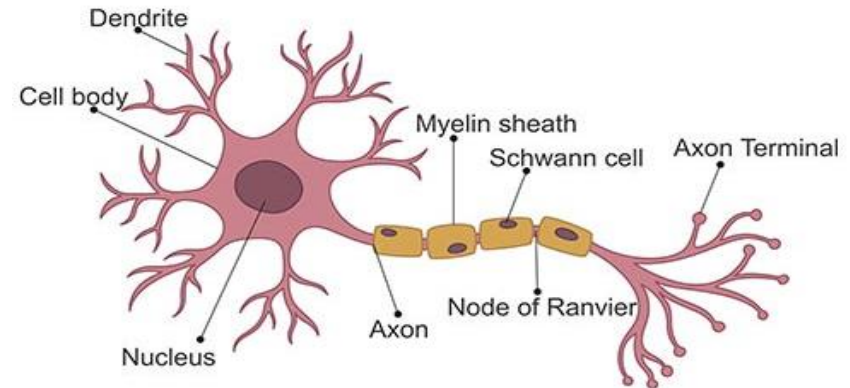


Figure extracted from: <https://www.cusabio.com/Cell-Marker/Neuron-Cell.html>

Synapse

Neurons communicate with each other through a process known as synaptic transmission.

When an electrical signal reaches the axon terminals, it triggers the release of neurotransmitters into the synapse. These neurotransmitters bind to receptors on the dendrites or cell body of the receiving neuron, which can either excite or inhibit the receiving neuron's activity. This way, information is transmitted from one neuron to another throughout the nervous system.

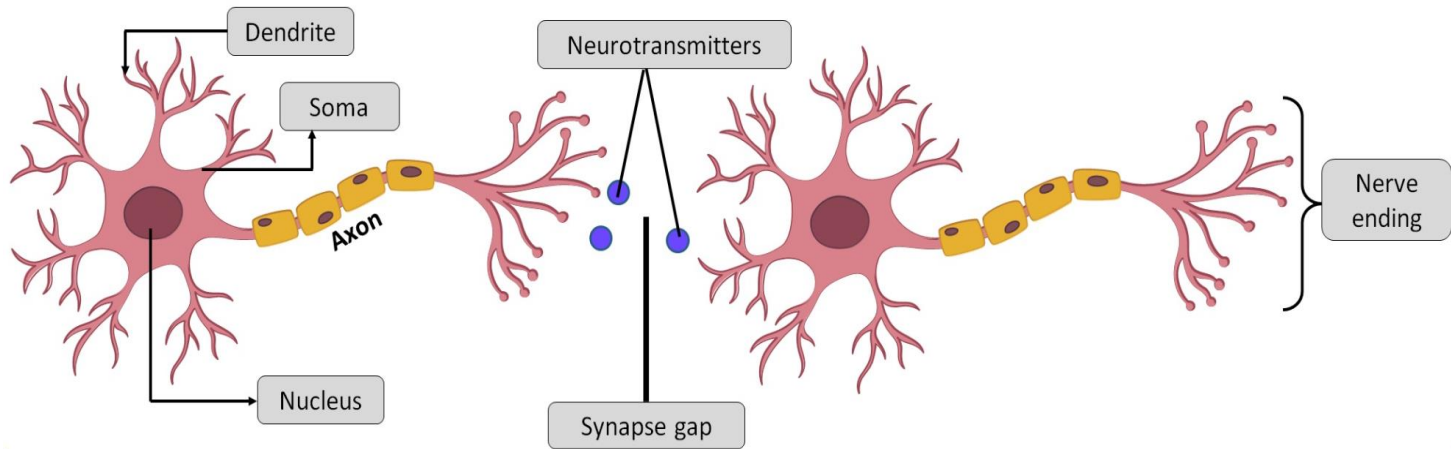


Figure extracted from: <https://www.teachoo.com/17797/3519/Neurons/category/Concepts/>

Neural Network

However, understanding the functioning and behavior of the human brain still challenges science, with several questions remaining unanswered:

- How is information processed in the brain?
- Which mechanisms are involved in brain function?
- How is the brain organized?

Despite not having complete knowledge of cerebral mechanisms, attempting to model biological neurons and some brain structures has been the path sought to instill 'intelligence' in machines.



Neural Network

Our brain is specialized in performing functions such as:

- ✓ Motor control,
- ✓ Perception of the environment,
- ✓ Alertness state,
- ✓ Inference,
- ✓ Homeostasis,
- ✓ Survival motivation,
- ✓ Learning and memory,
- ✓ Pattern recognition,
- ✓ Abstraction.



Illustration of René Descartes depicting the brain's reflex response to a stimulus.

Source: <https://commons.wikimedia.org/wiki/File:Descartes-reflex.JPG#/media/File:Descartes-reflex.JPG>

Biological Neuron

learning ⇔ establishment of new connections between neurons or modification of existing connections

Synapses are the connections that allow the transmission of signals between biological neurons and control the flow of information within the neural network.

Neuron

From a functional perspective, the biological neuron receives excitatory or inhibitory stimuli through its multiple inputs. If the sum of these stimuli surpasses a certain threshold, the neuron emits a nerve impulse.

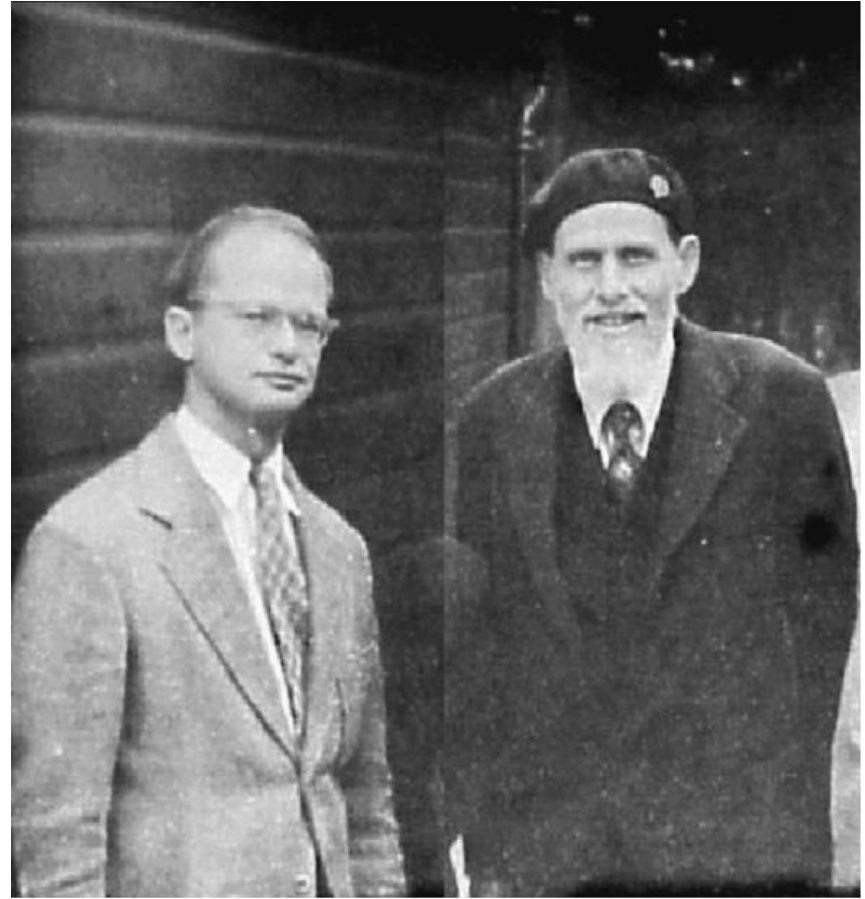
Artificial neurons are simplified models of biological neurons and constitute the basic processing units of artificial neural networks.



Artificial Neuron

The first model of an artificial neural network capable of computing arithmetic and logical functions was proposed in 1943 by McCulloch and Pitts.

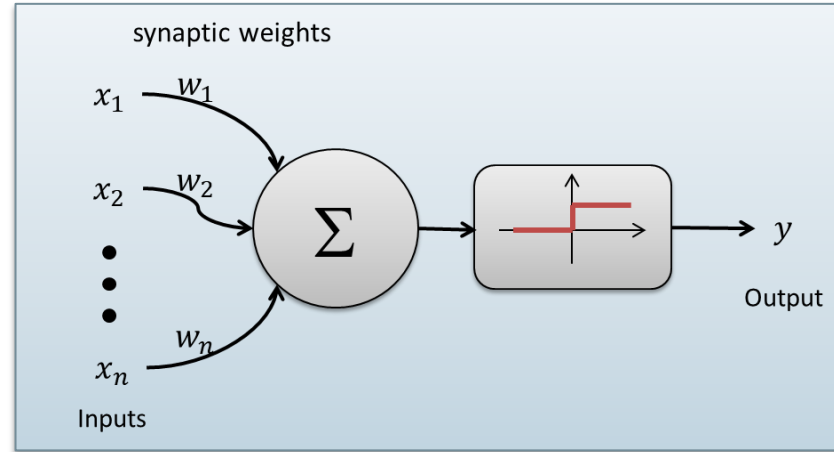
Their work is recognized as the origin of neural networks.



Source: MORENO-DÍAZ, Roberto; MORENO-DÍAZ, Arminda. On the legacy of WS McCulloch. Biosystems, v. 88, n. 3, p. 185-190, 2007.

Artificial Neuron

Artificial neurons have multiple inputs that receive a value and simulate dendrites.



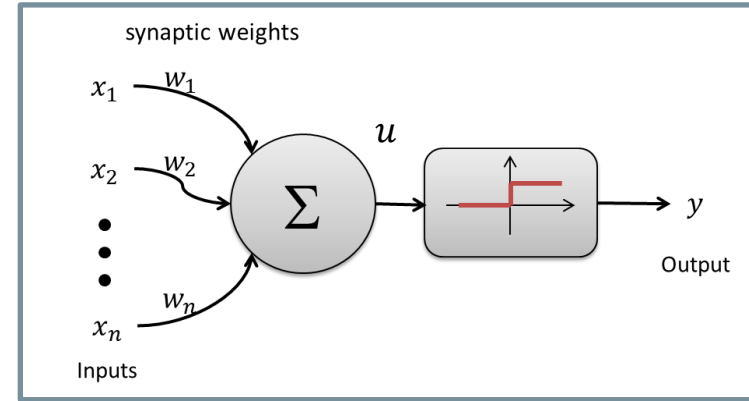
The inputs are weighted by synaptic weights and summed; after this, they pass through the **activation function**.

The activation function attempts to model the biological synaptic process and is responsible for determining whether the artificial neuron fires or not.

Artificial Neuron

The sum of weighted inputs can be expressed as:

$$u = \sum_{i=1}^n x_i \cdot w_i$$



Here, n is the number of inputs received by the neuron, x_i is the value of the i -th input and w_i is the synaptic weight of the i -th connection.

The output of the neuron is given by:





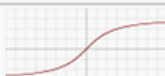

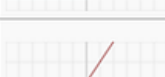


$$y = f(u)$$

Activation function

Inputs don't always trigger a neuron to fire; there must be a sufficient level for the neuron to activate. The activation function plays the role of determining when and how the artificial neuron fires.

There is no general rule for selecting the best activation function. Each type of function is more suitable for addressing a specific type of problem and dataset. Most of the time, determining the best activation function is empirical.



Name	Plot	Equation	Derivative
Identity		$f(x) = x$	$f'(x) = 1$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$
TanH		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2 + 1}$
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Parameteric Rectified Linear Unit (PReLU) [2]		$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Exponential Linear Unit (ELU) [3]		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
SoftPlus		$f(x) = \log_e(1 + e^x)$	$f'(x) = \frac{1}{1 + e^{-x}}$

Source:
<https://patrickhoo.wixsite.com/di-veindatasience/single-post/2019/06/13/activation-functions-and-when-to-use-them>



Architecture of the Neural Network

The configuration of artificial neurons and their interconnections defines the architecture of a neural network.

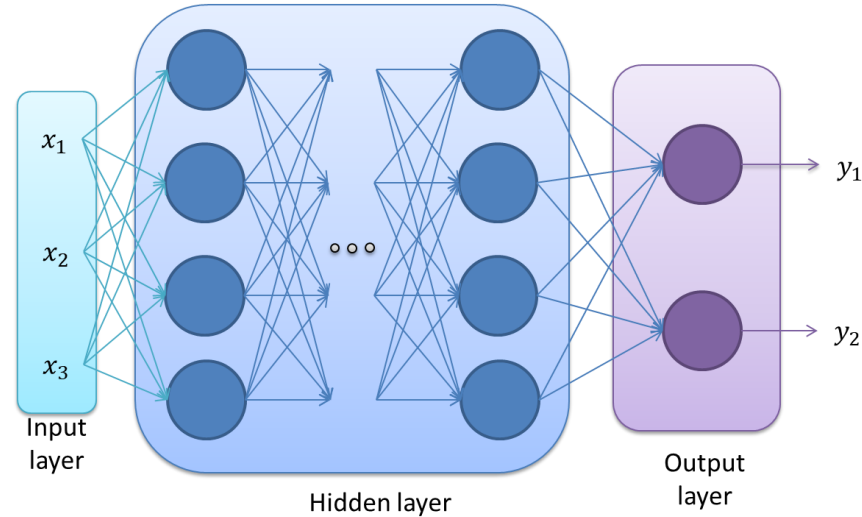
Selecting the appropriate architecture depends on various factors related to the problem at hand, including:

- Complexity of the problem
- Size of the dataset
- Dynamic or static characteristics



Architecture of a Neural Network

Neural networks can be divided into three parts:



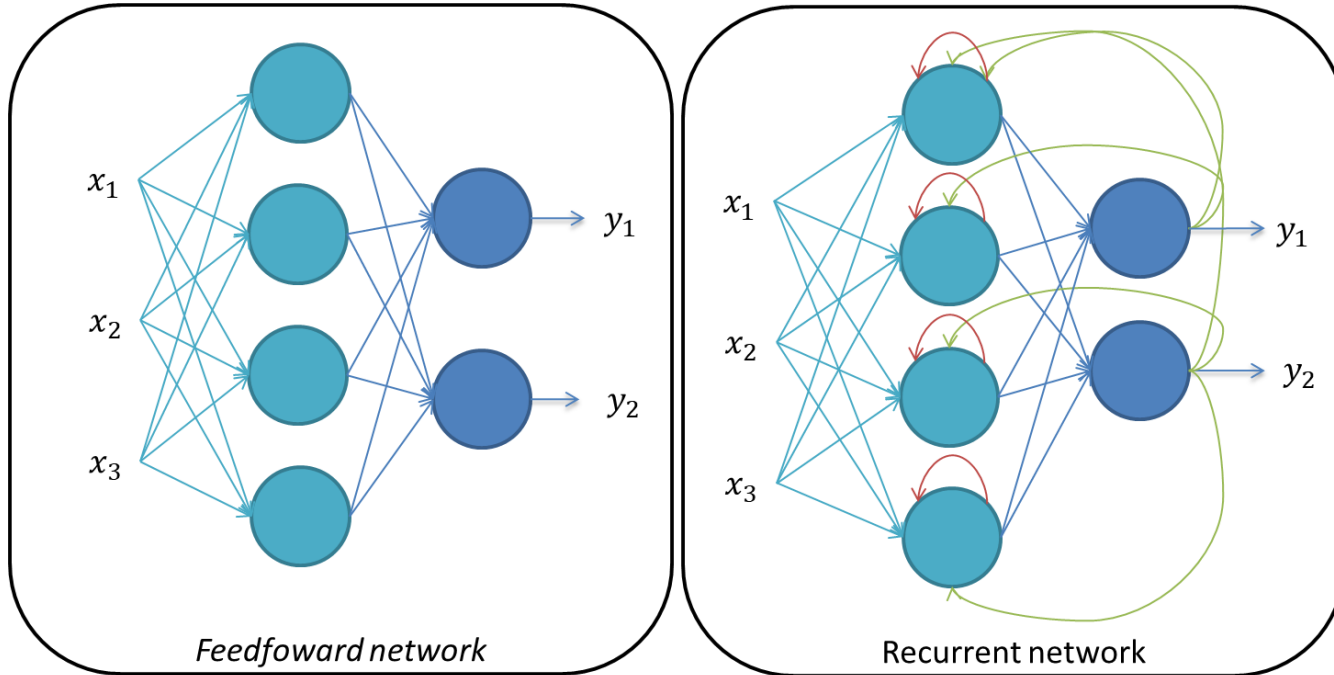
Input layer: receives information from the database.

Hidden or intermediate layer: responsible for extracting the necessary information from the data.

Output layer: generates the final result of the network.

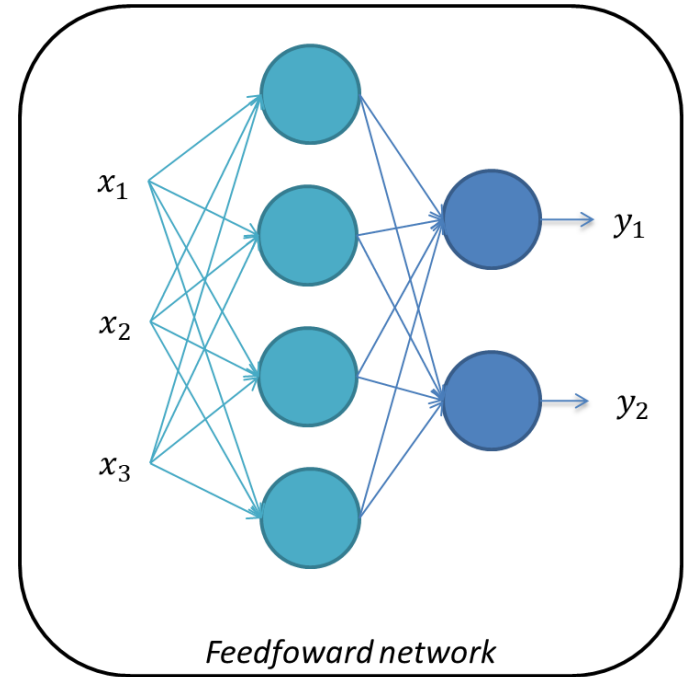
Architecture of the Neural Network

Considering the arrangement of artificial neurons and how they connect, we can define the following types of neural networks.



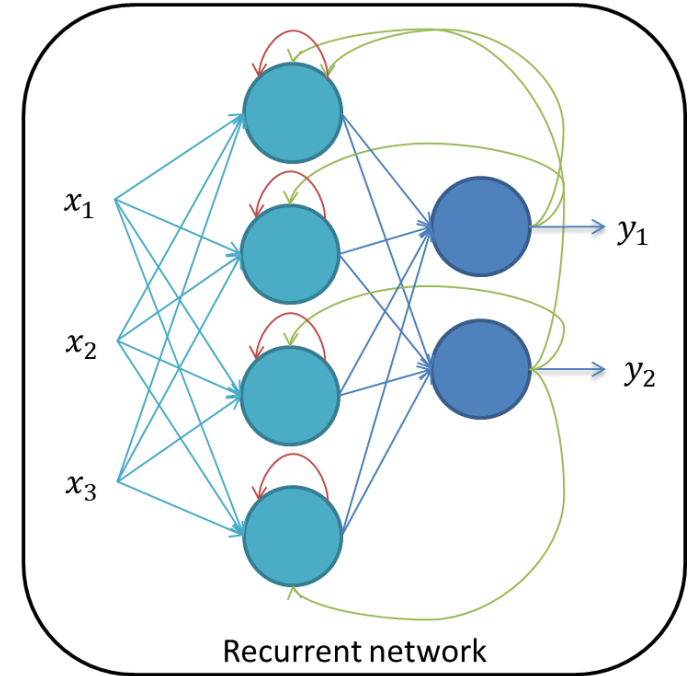
Feedforward Network

Feedforward neural networks were the first to be proposed. In them, information moves from input nodes towards output nodes, passing through hidden layers (if present). They do not have connections between neurons that form cycles or loops. The flow of information is unidirectional, starting from neurons in the input layer to neurons in the output layer.



Recurrent Network

Neurons can receive information through feedback, meaning neurons in the input layer can receive information from neurons in the output layer, from a neuron in the same layer, from the next layer, or even from itself.



Multilayers Network

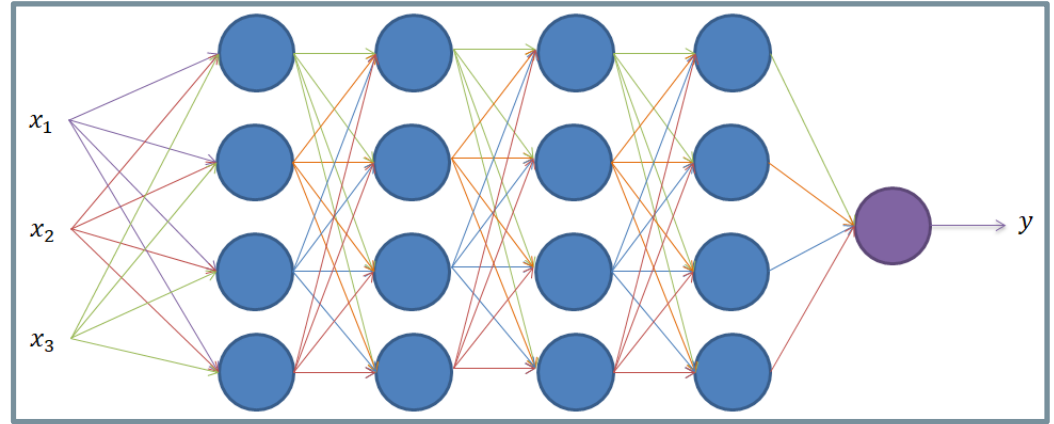
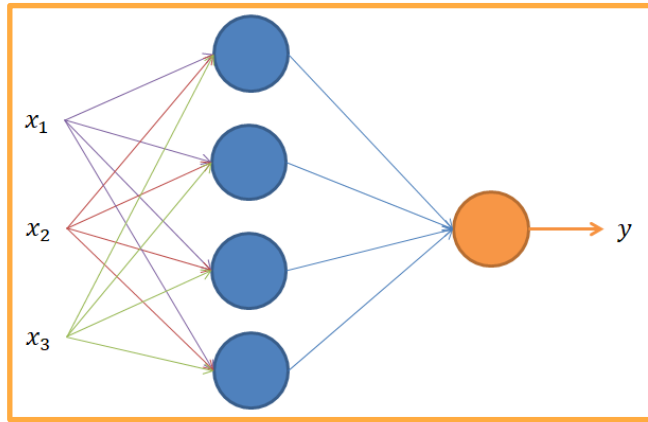
According to the pattern of neuron connections, a multilayer network can be:

- **Fully connected:** neurons are connected to all neurons in the subsequent layers;
- **Partially connected:** neurons connect to only some of the neurons in adjacent layers;
- **Locally connected:** neurons are partially connected in well-defined regions of the adjacent layers.



Shallow vs. Deep Neural Networks

The number of hidden layers defines whether an artificial neural network is shallow or deep. Deep neural networks have three or more hidden layers and usually require a large dataset for effective training.



Shallow vs. Deep Neural Networks

The overall **performance** of the neural network determines whether the system is designed and configured appropriately or if it can be improved with the addition of more hidden layers in the neural network.

It is intuitive to infer that the **computational cost** required to design and use deep neural networks is much higher than that required for shallow neural networks. This provides greater **flexibility** in adjusting weights, increasing the likelihood of finding a better combination for the neural network to solve the proposed problem.

However, we need to be mindful that this extensive tuning capability does not lead to **overfitting**. Deep neural networks are the most frequently used algorithms to facilitate deep learning in machines.



Neural Networks

Throughout the course, we will cover:

Feedforward Networks:

- Perceptron
- MultiLayer Perceptron (MLP)
- Radial Basis Function Network (RBF)
- Extreme Learning Machine (ELM)
- Convolutional Neural Network (CNN)

Recurrent Networks:

- Hopfield Network
- Long Short-Term Memory (LSTM)
- Gated Recurrent Unit (GRU)

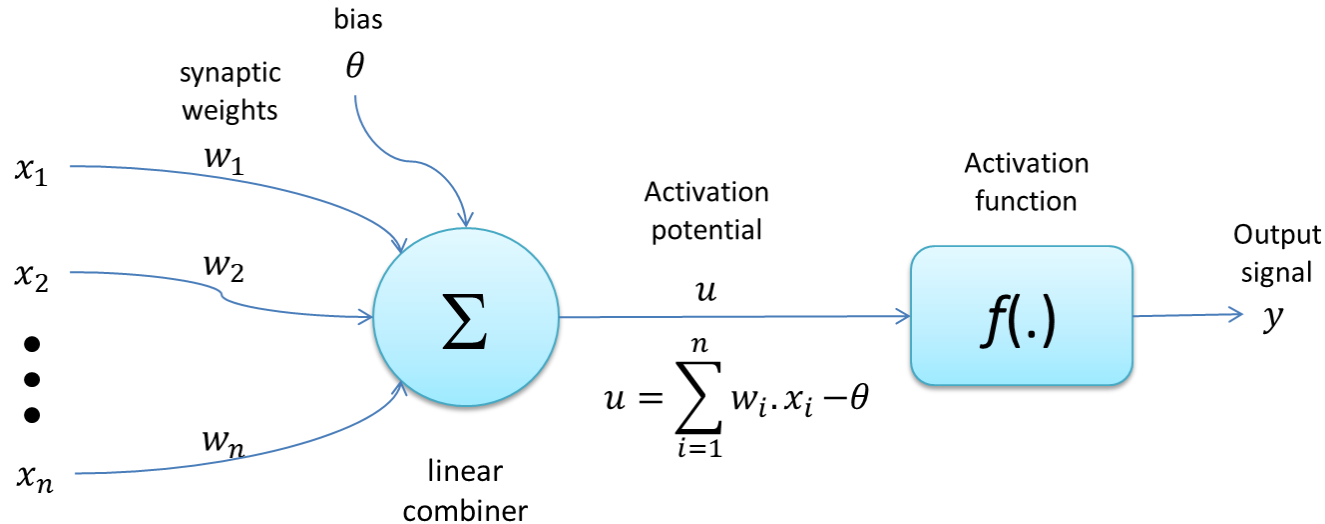


Perceptron

Perceptron

The perceptron is the most rudimentary form of a feedforward neural network used to classify two linearly separable classes.

The perceptron consists of a single artificial neuron with an input layer and connections leading to the output layer. The synaptic weights and bias are adjustable.

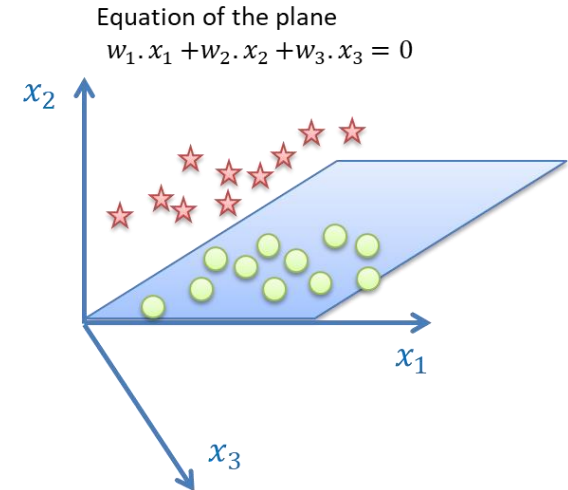
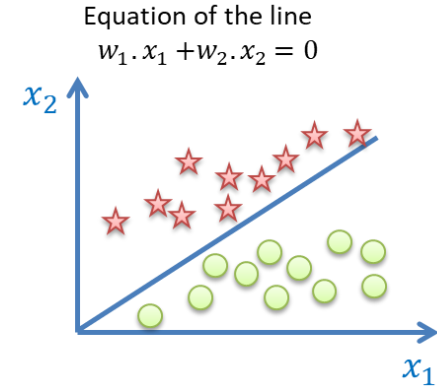


Parameters

Figure presents a two-dimensional graph, where each point corresponds to an input data and the two synaptic weights w_1 and w_2 determine a line.

If there are three inputs (x_1 , x_2 , and x_3), the space changes from two-dimensional to three-dimensional, and the perceptron needs to define a plane to separate the classes.

And what if we have more than three inputs?



Parameters

If there are more inputs, the perceptron needs to determine a hyperplane to separate the n classes.

The expression that generalizes the solution that the perceptron must determine is given by the equation of the hyperplane:

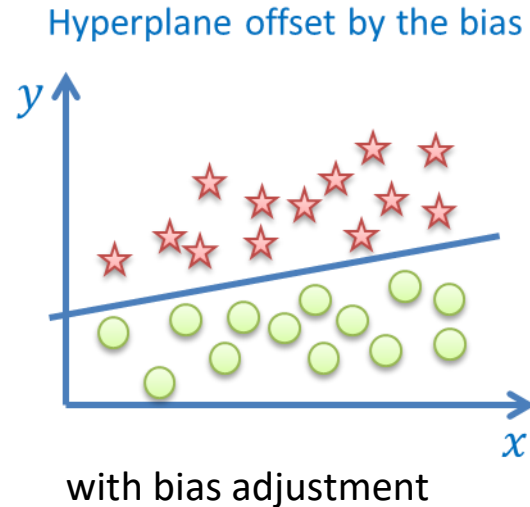
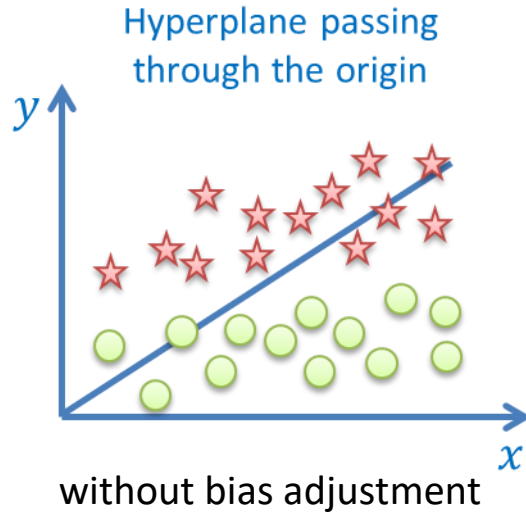
$$\sum_{i=1}^n w_i \cdot x_i = 0$$



Parameters

The **synaptic weights** define the separating hyperplane.

The **bias** determines the slope of the hyperplane that separates the classes.



$$u = \sum_{i=1}^n w_i \cdot x_i = 0$$

$$u = \sum_{i=1}^n w_i \cdot x_i - \theta = 0$$

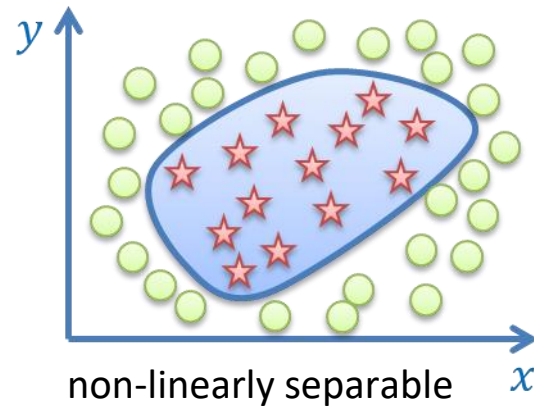
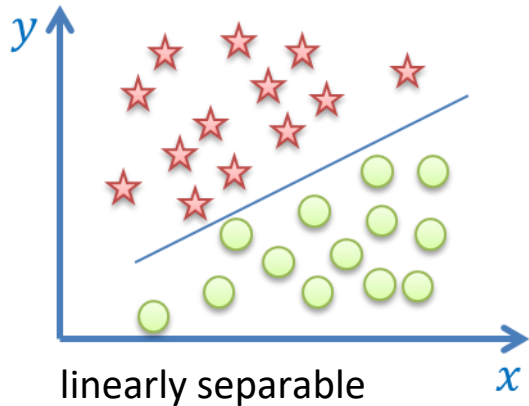
$$u = \sum_{i=0}^n w_i \cdot x_i = 0$$

Perceptron Limitations

The perceptron is an important structure that serves as the basis for understanding more complex neural networks.

It is a fast and reliable network for the class of problems it can solve.

Limitations: Discrimination of two linearly separable classes.



Perceptron training algorithm

1. Initialize:

- a. Set the synaptic weights (w_1, w_2, \dots, w_n) and bias b to small random values.
- b. Set the learning rate η , a small positive constant.

2. For each training sample (x, r)

- a. Compute the weighted sum: $u = w_1 \cdot x_1 + \dots + w_n \cdot x_n + b$
- b. Apply the activation function $f(u)$ (commonly a step function):

$$y = f(u) = \begin{cases} 1, & \text{if } u > 0 \\ 0, & \text{otherwise} \end{cases}$$

3. Update weights and bias:

Calculate the output error: $\varepsilon = r - y$ and update the weights and bias:

$$\mathbf{w} \leftarrow \mathbf{w} + \eta \varepsilon \mathbf{x}$$

$$b \leftarrow b + \eta \varepsilon$$

4. Repeat step 2 and 3 until $\varepsilon=0$ for all samples in the training set or for a specified number of epochs.

Note: The bias can be considered as $w_0 \cdot 1$.



Exercise 1 – AND

Consider the two possible outputs of the AND gate (0 and 1).

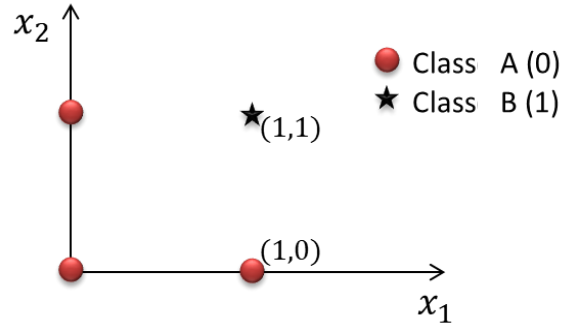


- a) Determine the inputs and outputs sets.
- b) Is it a linear problem? Draw a graph.
- c) Apply the perceptron algorithm to separate them. Determine a set of perceptron parameters that solve this problem.

Example 1 – AND - Solution

- Determine the inputs and outputs sets.
- Is it a linear problem? Draw a graph.

	Input			Output	Class
	x_0	x_1	x_2	r	
1	1	0	0	0	A
2	1	0	1	0	A
3	1	1	0	0	A
4	1	1	1	1	B



Applying the algorithm

1. Initialize synaptic weights: $w_0 = 1, w_1 = 1, w_2 = 1$ and the learning rate $\eta = 0.5$.

2. For each training sample (x, r)

Input 1 - first epoch

a. Compute the weighted sum:

$$u = w_0 \cdot x_0 + \dots + w_n \cdot x_n$$

$$u = 1.1 + 1.0 + 1.0 = 1$$

b. $y = f(1) = 1$

3. Update weights and bias:

$\varepsilon = r - y = 0 - 1 = -1$ and update the weights:

$$\mathbf{w} \leftarrow \mathbf{w} + \eta \varepsilon \mathbf{x}$$

$$w_0 = 1 + 0.5 \cdot (-1) \cdot 1 = 0.5$$

$$w_1 = 1 + 0.5 \cdot (-1) \cdot 0 = 1$$

$$w_2 = 1 + 0.5 \cdot (-1) \cdot 0 = 1$$

4. Repeat step 2 and 3.

Input				Output	Class
	x_0	x_1	x_2	r	
1	1	0	0	0	A
2	1	0	1	0	A
3	1	1	0	0	A
4	1	1	1	1	B



Applying the algorithm

2. For each training sample (x, r)

Input 2 - first epoch

a. Compute the weighted sum:

$$u = w_0 \cdot x_0 + \dots + w_n \cdot x_n$$
$$u = 0.5 \cdot 1 + 1.0 + 1.1 = 1.5$$

a. $y = f(1) = 1$

3. Update weights and bias:

$\varepsilon = r - y = 0 - 1 = -1$ and update the weights:

$$\mathbf{w} \leftarrow \mathbf{w} + \eta \varepsilon \mathbf{x}$$
$$w_0 = 0.5 + 0.5 \cdot (-1) \cdot 1 = 0$$
$$w_1 = 1 + 0.5 \cdot (-1) \cdot 0 = 1$$
$$w_2 = 1 + 0.5 \cdot (-1) \cdot 1 = 0.5$$

4. Repeat step 2 and 3.

	Input			Output	Class
	x_0	x_1	x_2	r	
1	1	0	0	0	A
2	1	0	1	0	A
3	1	1	0	0	A
4	1	1	1	1	B



Applying the algorithm

2. For each training sample (x, r)

Input 3 – first epoch

a. Compute the weighted sum:

$$u = w_0 \cdot x_0 + \dots + w_n \cdot x_n$$
$$u = 0.1 + 1.0 + 0.5 \cdot 0 = 1$$

b. $y = f(1) = 1$

3. Update weights and bias:

$\varepsilon = r - y = 0 - 1 = -1$ and update the weights:

$$\mathbf{w} \leftarrow \mathbf{w} + \eta \varepsilon \mathbf{x}$$
$$w_0 = 0 + 0.5 \cdot (-1) \cdot 1 = -0.5$$
$$w_1 = 1 + 0.5 \cdot (-1) \cdot 1 = 0.5$$
$$w_2 = 0.5 + 0.5 \cdot (-1) \cdot 0 = 0.5$$

4. Repeat step 2 and 3.

	Input			Output	Class
	x_0	x_1	x_2	r	
1	1	0	0	0	A
2	1	0	1	0	A
3	1	1	0	0	A
4	1	1	1	1	B



Applying the algorithm

2. For each training sample (x, r)

Input 4 – first epoch

a. Compute the weighted sum:

$$u = w_0 \cdot x_0 + \dots + w_n \cdot x_n$$
$$u = -0.5 \cdot 1 + 0.5 \cdot 1 + 0.5 \cdot 1 = 0.5$$

b. $y = f(0.5) = 1$

3. Update weights and bias:

$\varepsilon = r - y = 1 - 1 = 0 \rightarrow$ no weights update!

*** All samples were evaluated --> the first epoch has finished.

	Input			Output	Class
	x_0	x_1	x_2	r	
1	1	0	0	0	A
2	1	0	1	0	A
3	1	1	0	0	A
4	1	1	1	1	B



Applying the algorithm

2. For each training sample (x, r)

Input 1 – second epoch

a. Compute the weighted sum:

$$u = w_0 \cdot x_0 + \dots + w_n \cdot x_n$$
$$u = -0.5 \cdot 1 + 0.5 \cdot 0 + 0.5 \cdot 0 = -0.5$$

a. $y = f(-0.5) = 0$

3. Update weights and bias:

$$\varepsilon = r - y = 0 - 0 = 0 \rightarrow \text{no weights update!}$$

4. Repeat step 2 and 3.

	Input			Output	Class
	x_0	x_1	x_2	r	
1	1	0	0	0	A
2	1	0	1	0	A
3	1	1	0	0	A
4	1	1	1	1	B



Applying the algorithm

2. For each training sample (x, r)

Input 2 – second epoch

$$u = -0.5.1 + 0.5.0 + 0.5.1 = 0$$

$$y = f(0) = 0$$

$$\varepsilon = r - y = 0 - 0 = 0 \rightarrow \text{no weights update!}$$

Input 3 – second epoch

$$u = -0.5.1 + 0.5.1 + 0.5.0 = 0$$

$$y = f(0) = 0$$

$$\varepsilon = r - y = 0 - 0 = 0 \rightarrow \text{no weights update!}$$

Input 4 – second epoch

$$u = -0.5.1 + 0.5.1 + 0.5.1 = 0.5$$

$$y = f(0.5) = 1$$

$$\varepsilon = r - y = 1 - 1 = 0 \rightarrow \text{no weights update!}$$

	Input			Output	Class
	x_0	x_1	x_2	r	
1	1	0	0	0	A
2	1	0	1	0	A
3	1	1	0	0	A
4	1	1	1	1	B

Applying the algorithm

2. For each training sample (x, r)

Input 2 – second epoch

a. Compute the weighted sum:

$$u = w_0 \cdot x_0 + \dots + w_n \cdot x_n$$
$$u = -0.5 \cdot 1 + 0.5 \cdot 0 + 0.5 \cdot 1 = 0$$

a. $y = f(0) = 0$

3. Update weights and bias:

$$\varepsilon = r - y = 0 - 0 = 0 \rightarrow \text{no weights update!}$$

4. Repeat step 2 and 3.

	Input			Output	Class
	x_0	x_1	x_2	r	
1	1	0	0	0	A
2	1	0	1	0	A
3	1	1	0	0	A
4	1	1	1	1	B

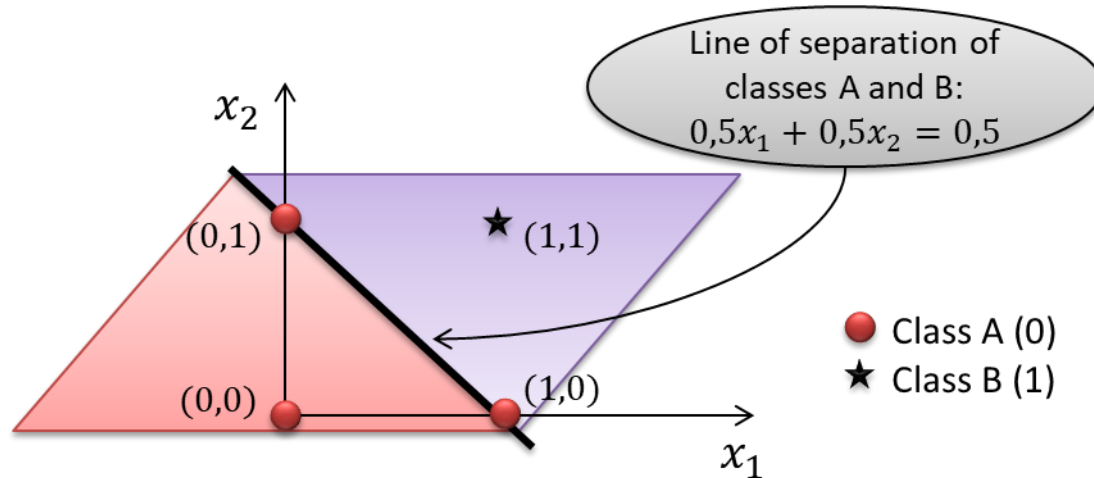


Example 1 – AND - Solution

There were no more errors, so a possible solution is given by:

$$w_0 = -0.5, \quad w_1 = 0.5, \quad w_2 = 0.5$$

Thus, a possible line equation which determines the separation of the classes is give by:
 $0,5x_1 + 0.5x_2 = 0.5$



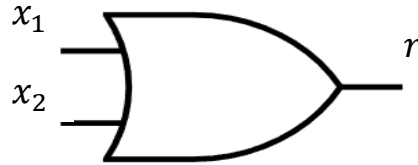
Example 1 – AND - Solution

We see that the perceptron was able to solve the problem, finding a line capable of separating classes A and B of the AND gate. We can observe in the graph that this is not the only line capable of solving the problem. Other solutions can be determined by the perceptron and depend on the initialization values of the synaptic weights and the learning rate configured in the algorithm.



Exercise 2 – Gate OR

Redo the previous exercise considering the logical OR gate now.



Exercise 3 – Gate AND/OR

- a) Code in Python using the NumPy library the perceptron algorithm and solve the AND and OR logic gate problems again.
- b) Analyze how the initial value of synaptic weights and the parameters affect the convergence of the perceptron algorithm. Comment.
- c) Vary the parameters of the perceptron network and verify that the separator line indicated as a solution is not unique. Plot one of the solutions found.



Considerations about perceptron parameters

- The **learning rate** affects both the number of epochs and the final performance of the neural network. A very small value of η can result in a long time for the algorithm to converge. A very high value of η can lead to instability when updating synaptic weights, and the network may never converge to a good solution. The best value of η depends on the problem. In practice, you can assign an arbitrary value to initialize the learning rate η and adjust this value based on the performance of the training algorithm.
- **Synaptic weight** high values can cause saturation in the neuron, impairing the convergence of the algorithm and requiring several epochs for adjustment. Assigning small, symmetric values around 0 to synaptic weights facilitates the task.

Despite all the mathematics surrounding them, the choice of network architecture and parameters configuration is primarily empirical.



Exercise 4 – Iris Flower Species Classification

In this exercise, you should adapt your perceptron to propose hyperplanes that separate the species of iris flowers.

- 1 - Since the perceptron can only linearly separate 2 classes, choose just 2 species of iris flowers and design a perceptron to separate them. Clearly present your choice, considerations and assumptions adopted to propose the solution. If necessary, use the exploratory data analysis performed as an exercise in Unit 1.
- 2 - Select only the relevant features. Justify your choice using graphs (scatter, box plots, or heat map).
- 3 - Split the dataset into training and testing sets. Illustrate what happens to the perceptron convergence if the data are presented sequentially (first all samples from class 1, then all from class 2). Compare with the case where there is sample shuffling.
- 4 - Comment on what happens to the convergence of the perceptron when the number of epochs and the learning rate are changed. Illustrate or present values considering extreme cases for these parameters.
- 5 - Present and compare the confusion matrix considering the perceptron input using 4 features and using just the selected features (from item 2).
- 6 - Is it possible to separate the 3 classes of flowers using the perceptron? Present a solution.



Exercise 5 - Breast cancer diagnosis

Breast cancer is the most common type affecting women in Brazil, according to the Brazilian Society of Mastology.

One of the main characteristics of breast cancer is that the earlier the diagnosis, the higher the chances of successful treatment. It is estimated that early diagnosis allows treatment and cure in 95% of cases.

Routine breast examination allows for the identification and treatment of a potential tumor before it causes noticeable symptoms.

The early detection process involves performing a biopsy to analyze abnormal nodules or masses in breast tissue. The abnormal cells are evaluated by a specialist to determine if the mass is malignant or benign.

Machine learning techniques can work together with specialist doctors to contribute to faster and more accurate diagnoses of breast cancer.



Exercise 5 - Breast cancer diagnosis

The goal of this project is to train a perceptron to assist in diagnosing the type of tumor (benign/malignant).

To do this, we will use a real dataset called Breast Cancer Wisconsin Diagnostic, which contains measurements of cells from biopsies of women with abnormal breast nodules.

This dataset is available in scikit-learn and can be loaded as follows:

```
python

from sklearn import datasets
bc = datasets.load_breast_cancer()
print("Description of the dataset :: ", bc['DESCR'])
```

Exercise 5 - Breast cancer diagnosis

To do:

- Implement the perceptron that takes as input the 30 characteristics representing the measurements made on cells from biopsies and provides the diagnosis as output, indicating whether it is a benign or malignant tumor.
- Randomly separate 75% of the samples to train the perceptron and the remaining samples to validate the system. The code should provide the overall accuracy of the classification system.

Answer the questions:

1. How do the initialization values of the weight vector impact the accuracy of the system? Illustrate the accuracy obtained with some extreme cases.
2. How does the learning rate affect the accuracy of the system? Illustrate the accuracy obtained with some extreme cases.
3. Perform various tests altering the parameters and indicate the configuration that resulted in the best performance.
4. Based on the previous answer and the simulations performed, are the classes of this dataset linearly separable?
5. Challenge: Note that the first 10 measurements in the dataset correspond to the mean values of each of the measured variables, the next 10 to the standard errors, and the last 10 to the worst case observed for each variable. Make the necessary changes to the code so that instead of the 30 characteristics, it receives only 10, considering the three scenarios: mean values, standard error, and worst case.
6. Adjust the perceptron parameters again for each of the 3 scenarios and comment on the performance obtained.



Project 2

Unraveling the Human Brain with Brain-Computer Interfaces.

