Introduction to Machine Learning

Physics 265 Winter 2025

March 23, 2025

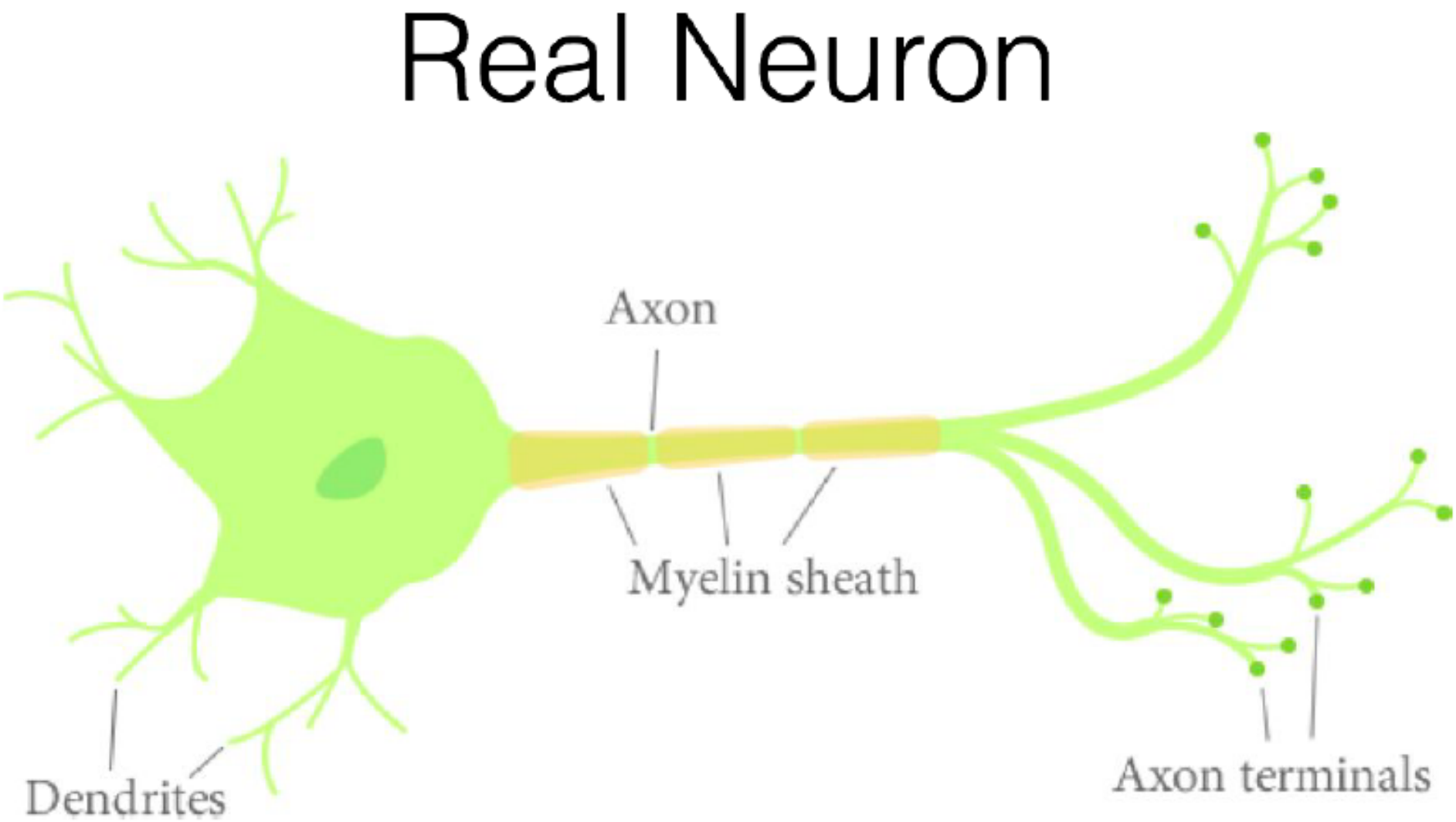
Overview

Introduction

These notes are mostly based on:

- Florian Marquardt's online course "Machine Learning for Physicists": `https://machine-learning-for-physicists.org`
- " Statistical mechanics of neural networks" by Haim Sompolinsky.

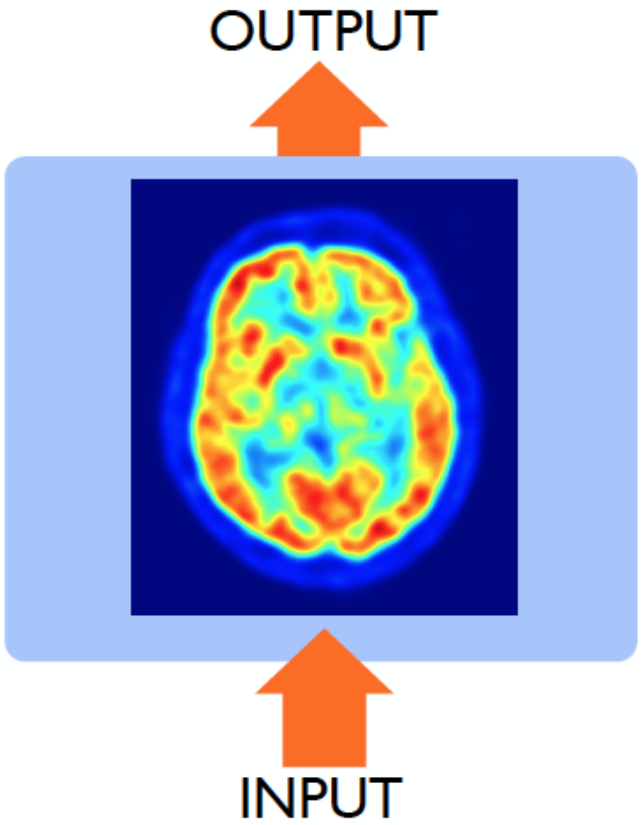
Neurons!



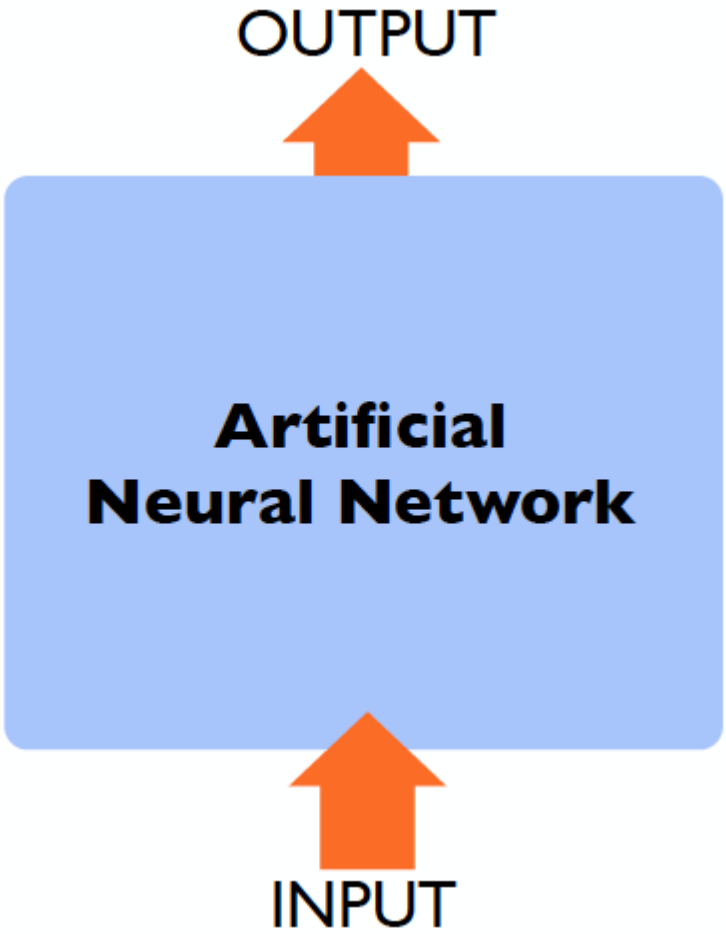
What is a neural network?

Based on nature, neural networks are the usual representation we make of the brain : neurons interconnected to other neurons which forms a network. A simple information transits in a lot of them before becoming an actual thing, like "move the hand to pick up this pencil". The operation of a complete neural network is straightforward : one enters variables as inputs (for example an image if the neural network is supposed to tell what is on an image), and after some calculations, an output is returned (following the first example, giving an image of a cat should return the word "cat").

Input and Output



Input and Output



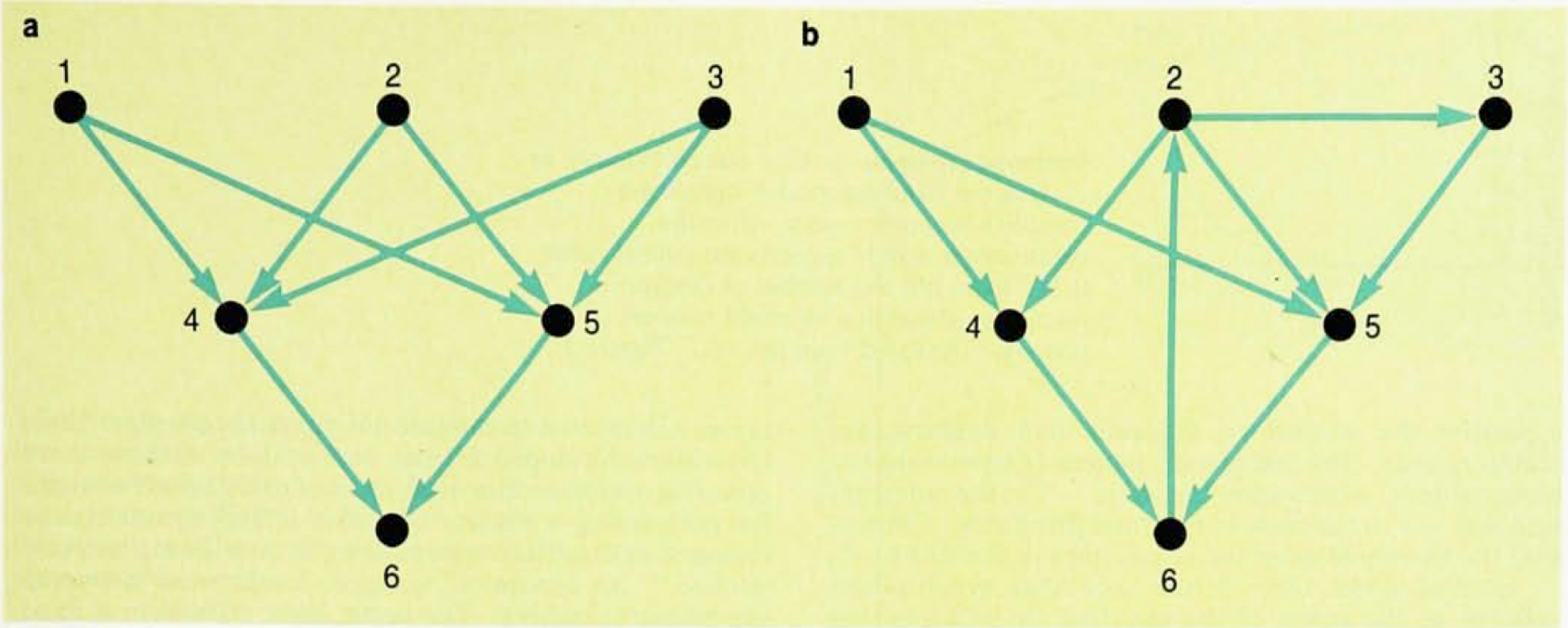
Neural Networks and Statistical Mechanics

We continue with a summary of part of the paper" Statistical mechanics of neural networks" by Haim Sompolinsky.

What is an artificial neural network?

A neural network is a large, highly interconnected assembly of simple elements. The elements, called **neurons** , are usually two-state devices that switch from one state to the other when their input exceeds a specific threshold value. In this respect the elements resemble biological neurons, which fire—that is, send a voltage pulse down their axons—when the sum of the inputs from their synapses exceeds a "firing" threshold. Neural networks therefore serve as models for studies of cooperative behavior and computational properties of the sort exhibited by the nervous system.

Neural Network architecture



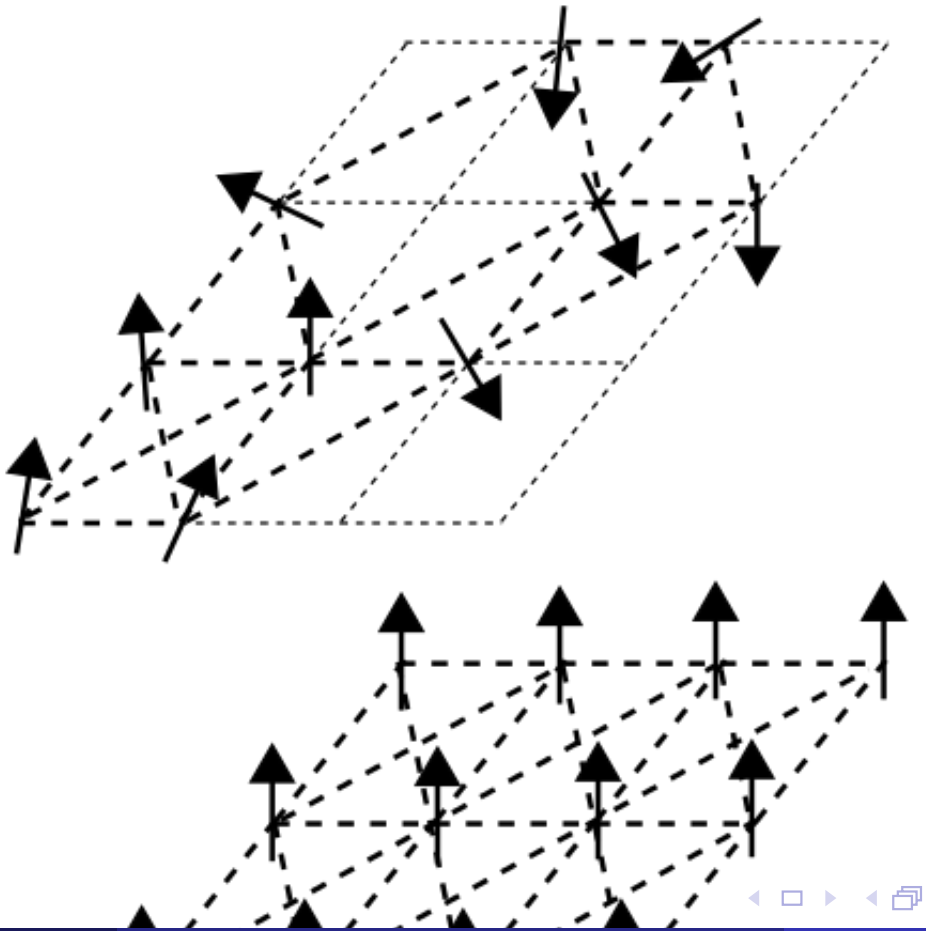
Network architectures. **a:** A feedforward system with three layers. **b:** A neural circuit. A circuit contains feedback loops, such as the directed graph 2→3→5→6→2, that close on themselves. This closure gives rise to recurrent activity in the network. **Figure 1**

Relationship between Neural Networks and spin glass systems

John Hopfield pointed out the equivalence between the long-time behavior of networks with symmetric connections and equilibrium properties of magnetic systems such as spin glasses. In particular, Hopfield showed how one could exploit this equivalence to "design" neural circuits for associative memory and other computational tasks.

What is a spin glass system?

Spin glasses are magnetic systems with randomly distributed ferromagnetic and antiferromagnetic interactions. The low-temperature phase of these systems—the spin glass phase—is in many ways a prototype for condensation in disordered systems with conflicting constraints.



What is the connection between spin glass systems and neural networks?

Neural systems share several features with long range spin glasses:

- The spatial configuration of the two systems bears no resemblance to the crystalline order of pure magnets or solids.
- The couplings between spins in spin glasses can be both positive and negative, which is also true of the couplings between neurons.
- And just as each spin in a long-range spin glass is connected to many others, so is each neuron in most neural systems.

Limitations of the analogy

Of course, the analogy between long-range spin glasses and neural systems is far from perfect.

- First, the connections in neural systems, unlike those in spin glasses, are not distributed at random, but possess correlations that are formed both genetically and in the course of learning and adaptation. These correlations alter the dynamical behavior of the system and endow it with useful computational properties.
- Another major difference is the asymmetry of the connections: The pairwise interactions between neurons are, in general, not reciprocally symmetric; hence their dynamic properties may be very different from those of equilibrium magnetic systems, in which the pairwise interactions are symmetric.

The models

- The neurons are represented by simple, point-like elements that interact via pairwise couplings called **synapses** .
- The state of a neuron represents its level of activity. Neurons fire an "action potential" when their "membrane potential" exceeds a threshold, with a firing rate that depends on the magnitude of the membrane potential.
- If the membrane potential is below threshold, the neurons are not active.
- The membrane potential of a neuron is assumed to be a linear sum of potentials induced by the activity of its neighbors. Thus the potential in excess of the threshold, which determines the activity, can be denoted by a local field:

$$h_i(t) = \sum_{j=1}^{N} J_{ij} \frac{S_j(t) + 1}{2} - \theta_i \tag{1}$$

The synaptic efficacy J_{ij} measures the contribution of the activity of the jth, presynaptic neuron to the potential acting on the ith, postsynaptic neuron. The contribution is positive for excitatory synapses and negative for inhibitory ones. The activity of a neuron is represented by the variable $S, = 1$ for firing state and $S, = -1$, for inactive state. The value of the threshold potential of neuron " i" is denoted by θ_i .
In this model there is no clock that synchronizes updating of the states of the neurons. In addition, the dynamics should not be deterministic, because neural networks are expected to function also in the presence of stochastic noise. These features are incorporated by defining the network dynamics in analogy with the single spin- flip Monte Carlo dynamics of Ising systems at a finite temperature.

Probability of a neuron update

$$P = \frac{1}{1 + e^{-4\beta h_i}} \tag{2}$$

where h_i is the local field at time t, and β is the inverse temperature of the network. In the Monte Carlo process, the probability of a neuron's being in, say, state $+1$ at time $t + dt$ is compared with a random number and the neuron's state is switched to $+1$ if the probability is greater than that number. The temperature is a measure of the level of stochastic noise in the dynamics. In the limit of zero temperature, the dynamics consists of single spin flips that align the neurons with their local fields. The behavior of the network depends on the form of the connectivity matrix J_{ij}.

The behavior of the network depends on the form of the connectivity matrix J_{ij}. Studies of neural networks have focused mainly on two architectures.

- One is **the layered network** , in which the information flows forward, so that the computation is a mapping from the state of the input layer onto that of the output layer. The perceptron, consisting of only an input and an output layer, is a simple example of such a **feedforward network** . Although interest in the perceptron declined in the 1960s, interest in feedforward networks that contain hidden layers has revived in the last few years as new algorithms have been developed for the exploitation of these more powerful systems. The usefulness of multilayer networks for performing a variety of tasks, including associative memory and pattern recognition, is now being actively studied. As dynamical systems, feedforward networks are rather primitive: The input layer is held in a fixed configuration and all the neurons in each subsequent layer compute their states in parallel according to the states of the preceding layer at the previous time step.

- Second class of network models contain feedback loops. These networks are known as term **neural circuits** . Many structures in the cortex show extensive feedback pathways, suggesting that feedback plays an important role in the dynamics as well as in the computational performance. Feedback loops are essential also for the function of nervous systems that control stereotypical behavioral patterns in animals. Besides their biological relevance, however, neural circuits are interesting because the long iterative dynamical processes generated via the feedback loops endow them with properties not obtained in layered networks of comparable sizes.

Symmetric circuits and Ising magnets

In symmetric circuits the the synaptic coefficients J_{ij} and J_{ji} are equal for each pair of neurons. In that case the dynamics is purely relaxational: There exists a function of the state of the system, the energy function, such that at $T = 0$ the value of this function always decreases as the system evolves in time. For a circuit of two-state neurons the energy function has the same form as the Hamiltonian of an Ising spin system:

$$E = \frac{1}{2} \sum_{i,j=1}^{N} J_{ij} S_i S_j - \sum_{i=1}^{N} h_i S_i \tag{3}$$

The first term represents the exchange energy mediated by pairwise interactions, which are equal in strength to the respective synaptic coefficients. The last term is the energy due to the interaction with external magnetic fields, which, in our case, are given by: $h_i = \sum_j J_{ij} - 2\theta_i$

Multi-layered neural network

Neural networks can usually be read from left to right. Here, the first layer is the layer in which inputs are entered. There are 2 internals layers (called hidden layers) that do some math, and one last layer that contains all the possible outputs.

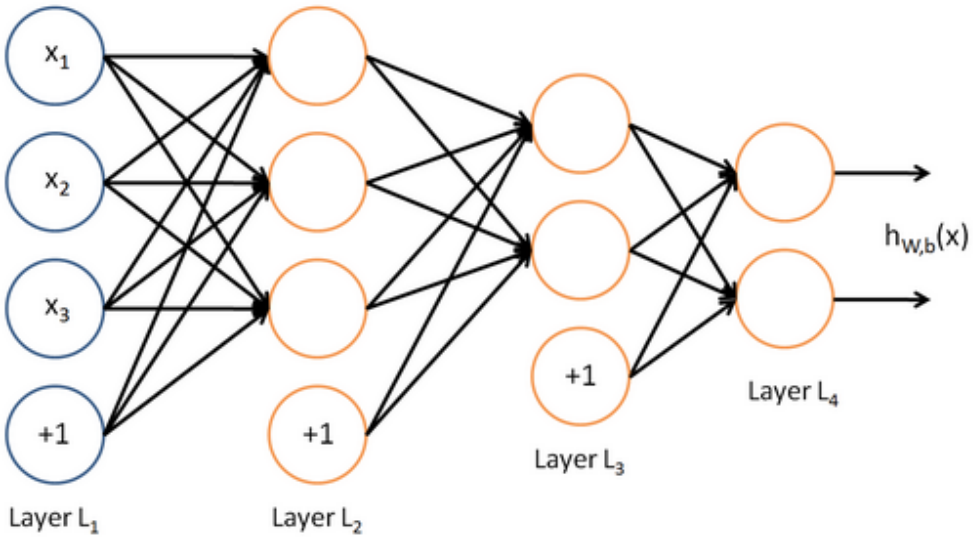


Figure 1 — Representation of a neural network

Representing a neuron operation

What does a neuron do ?

The operations done by each neurons are pretty simple :

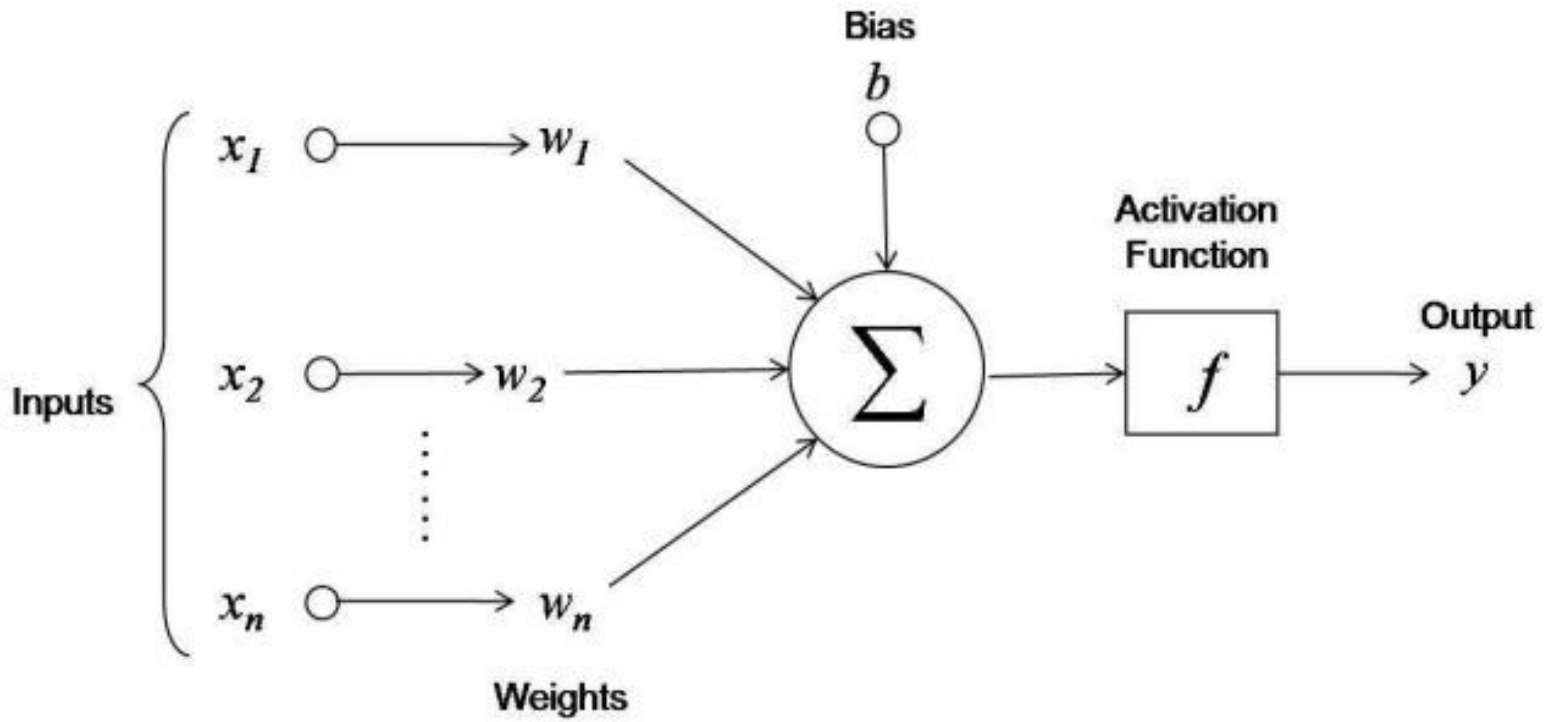


Figure 2 — Operations done by a neuron

Representing a neuron operation

- The neuron adds up the value of every neurons from the previous column it is connected to. On the Figure 2, there are 3 inputs (x1, x2, x3) coming to the neuron, so 3 neurons of the previous column are connected to our neuron.
- This value is multiplied, before being added, by another variable called "weight" (w1, w2, w3) which determines the connection between the two neurons. Each connection of neurons has its own weight, and those are the only values that will be modified during the learning process.
- Moreover, a bias value may be added to the total value calculated. It is not a value coming from a specific neuron and is chosen before the learning phase, but can be useful for the network.
- After all those summations, the neuron finally applies a function called "activation function" to the obtained value.

The activation function

The so-called activation function usually serves to turn the total value calculated before to a number between 0 and 1 (done for example by a sigmoid function shown by Figure 3). Other function exist and may change the limits of our function, but keeps the same aim of limiting the value.

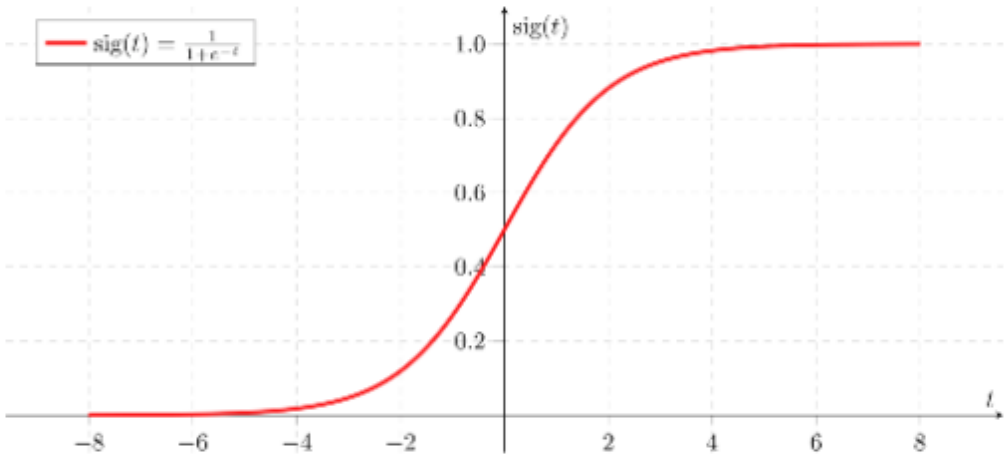


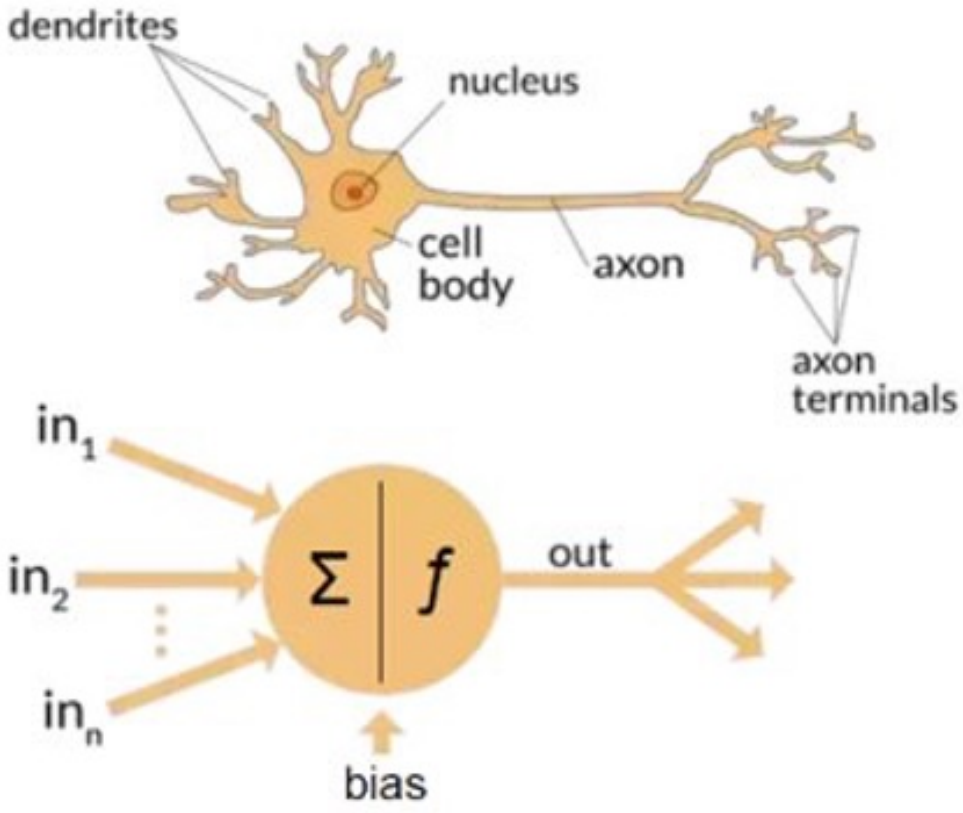
Figure 3 — Sigmoid function

How do networks learn?

When an input is given to the neural network, it returns an output. On the first try, it can't get the right output by its own (except with luck) and that is why, during the learning phase, every input comes with its label, explaining what output the neural network should have guessed. If the choice is the good one, actual parameters are kept and the next input is given. However, if the obtained output doesn't match the label, weights are changed. Those are the only variables that can be changed during the learning phase. This process may be imagined as multiple buttons, that are turned into different possibilities every times an input isn't guessed correctly.

The perceptron

The Neuron fires an action signal when the cell meets a particular threshold. This action either happen or they don't. Therefore, perceptrons can be applied in solving Binary Classification problems where the sample is to be identified as belonging to one of the predefined two classes.



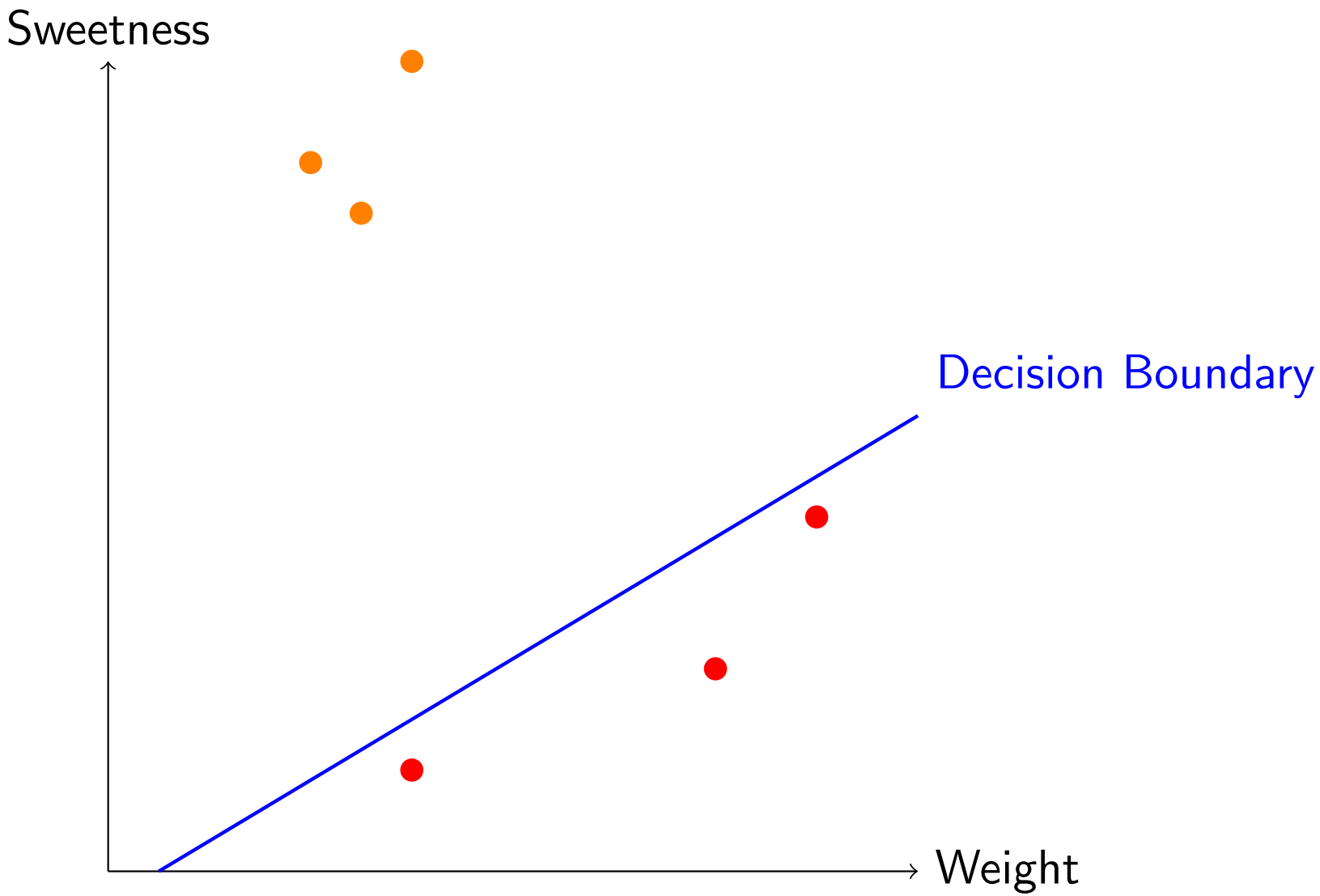
Sorting Fruits Using Linear Separability

- Imagine you're a fruit vendor who wants to automatically sort fruits.
- You have two types of fruits: **Apples** and **Oranges**.
- Two key features characterize each fruit:
  - **Weight** (in grams)
  - **Sweetness Level** (on a scale from 1 to 10)

The Fruit Characteristics

- **Apples:** Tend to be heavier but not very sweet.
- **Oranges:** Tend to be lighter but sweeter.
- When you plot these fruits on a graph:
  - Horizontal axis: Weight.
  - Vertical axis: Sweetness.

Visualizing the Data



What is Linear Separability?

- A dataset is **linearly separable** if you can draw one straight line (or a flat plane in higher dimensions) that separates the two classes.
- In this fruit example, if you can draw a line that separates all apples from all oranges, the data is linearly separable.
- This line divides the graph into two regions:
  - One region contains only apples.
  - The other region contains only oranges.

Simpler Math Terms

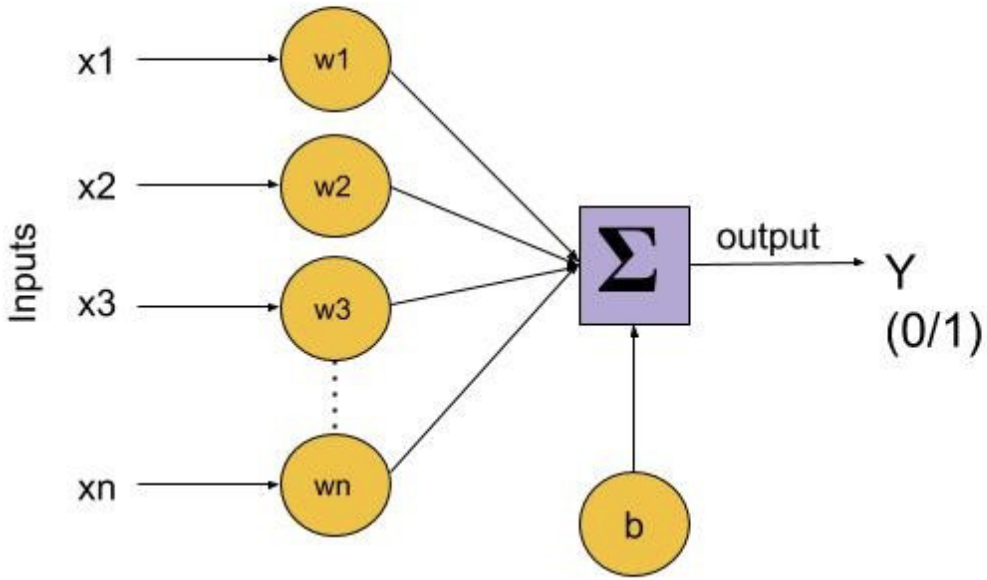
- Each fruit is described by two numbers: its weight (x_1) and its sweetness (x_2).
- We can define a line (or decision boundary) by the equation:

$$w_1 x_1 + w_2 x_2 + b = 0$$

- The classification rule is:
  - If $w_1 x_1 + w_2 x_2 + b > 0$, classify as one fruit (say, an apple).
  - If $w_1 x_1 + w_2 x_2 + b < 0$, classify as the other (an orange).
- If such a line exists, then the fruit data is linearly separable.

The perceptron: algorithm

The Algorithm



Schematic of Perceptron

Since Perceptrons are Binary Classifiers (0/1), we can define their computation as follows:

Limitations

1. A single-layer perceptron works only if the dataset is linearly separable.

2. The algorithm is used only for Binary Classification problems. However, we can extend the algorithm to solve a multiclass classification problem by introducing one perceptron per class. i.e., each perceptron results in a 0 or 1 signifying whether or not the sample belongs to that class.

The perceptron: tutorials

Examples of perceptron scripts:

- Perceptron With Scikit-Learn: `https://machinelearningmastery.com/perceptron-algorithm-for-classification-in-python/`