Introduction to Machine Learning

Ens'IA

Ensimag 2022-2023

24 octobre 2022

Outline

1. Presentation

2. Introduction

3. First Neuron

Outline

1 Presentation

2 Introduction

3 First Neuron

Ens'IA

Who are we ?

- Association founded in may 2019
- Promote artificial intelligence and its learning
- Share knowledge between students

Ens'IA

Why join us ?

- Showing off at the coffee machine
- Impress your grandparents
- Add a line to your resume
- Eventually learn to do AI

No need to be an expert to help us !

Outline

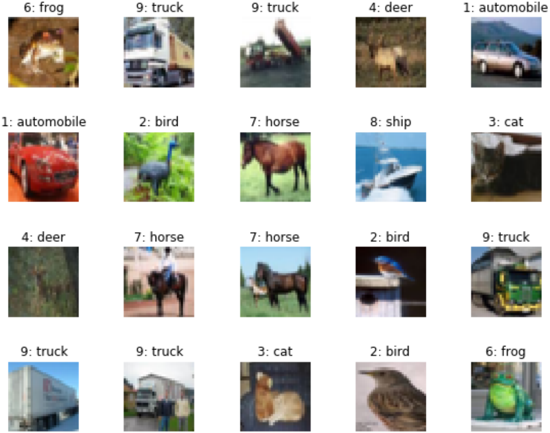
1. Presentation

2. Introduction

3. First Neuron

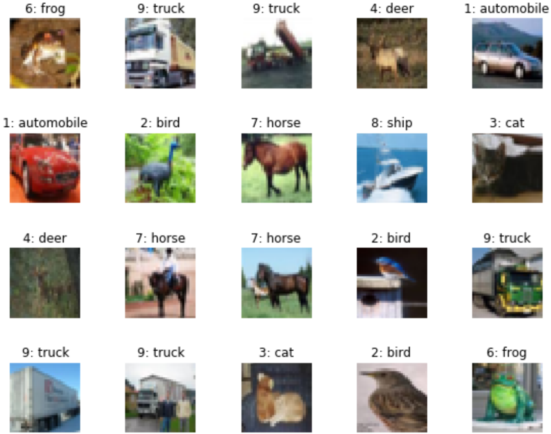
Motivation - example

Supposes we want to create a program capable of classifying images...



Motivation - example

Supposes we want to create a program capable of classifying images...



How so we do that ?

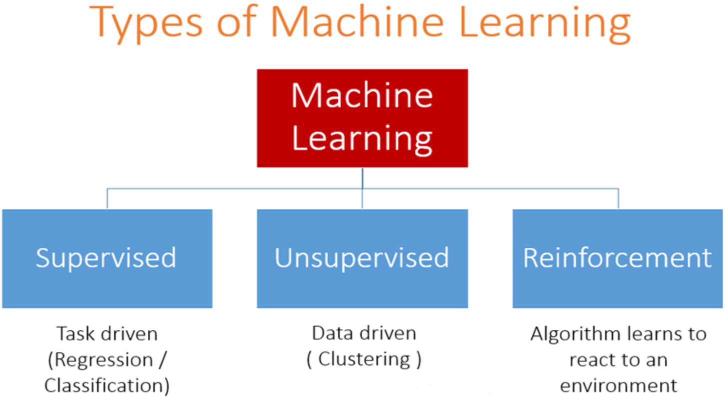
Motivation - example

Solution : Allow the computer to learn from data without having to code it explicitly.

Motivation - example

Solution : Allow the computer to learn from data without having to code it explicitly.
In other words : **Machine Learning !**

Types of Machine Learning



Types of Machine Learning

Machine Learning

- **Supervised**: Task driven (Regression / Classification)
- **Unsupervised**: Data driven (Clustering)
- **Reinforcement**: Algorithm learns to react to an environment

Outline

1. Presentation

2. Introduction

3. First Neuron

Goal

We'll be doing Supervised Learning :

$$f(x) = y$$

Goal

We'll be doing Supervised Learning :

$$f(x) = y$$

How to find f ?

Goal

We'll be doing Supervised Learning :

$$f(x) = y$$

How to find f ?

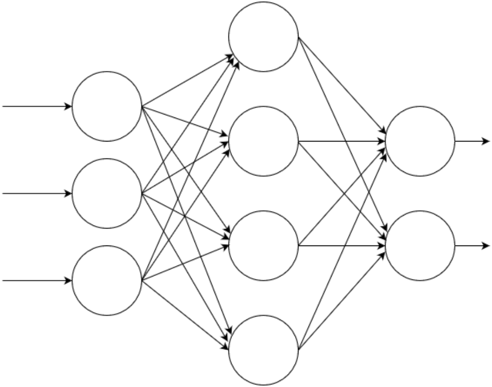
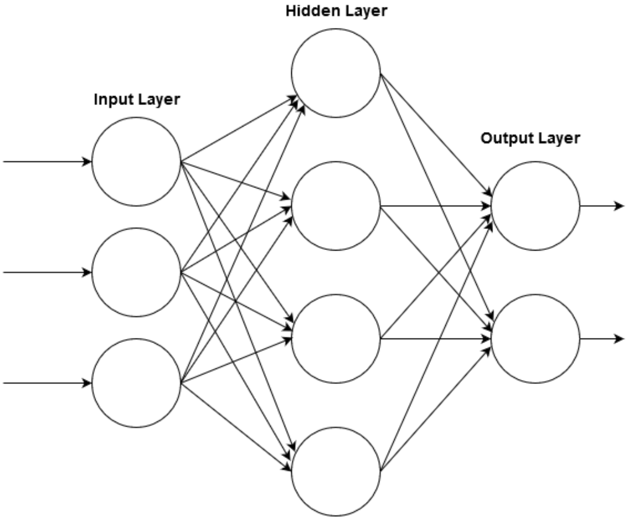


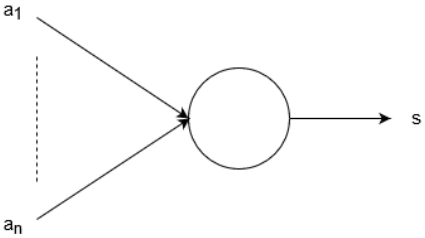
FIGURE 1 – Neural network

Neural network



→ Succession of neuron layers

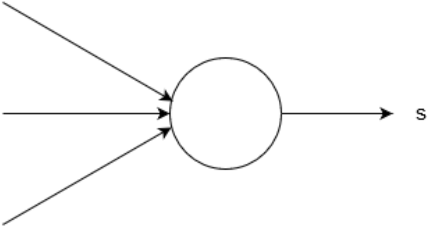
Perceptron



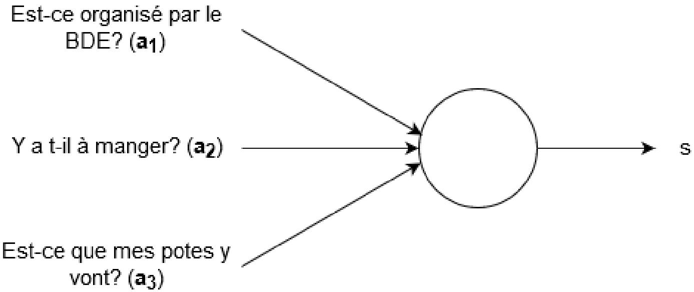
$a_1, ..., a_n, s \in 0, 1$

$$s = \begin{cases} 1 & \text{if } \sum_{i=0}^{n} a_i * w_i + b > 0 \\ 0 & \text{otherwise.} \end{cases}$$

Perceptron - Example

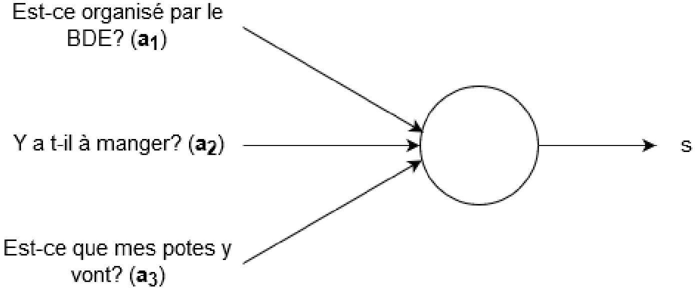


Perceptron - Example


Est-ce organisé par le BDE? (a_1)
Y a t-il à manger? (a_2)
Est-ce que mes potes y vont? (a_3)
s

If $a_1 + a_2 + a_3 = 3, s = 1$

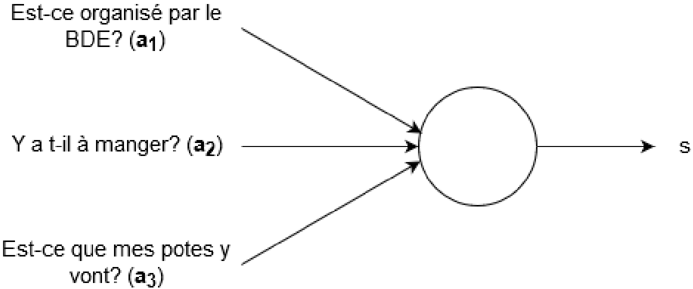
Perceptron - Example


Est-ce organisé par le BDE? (a_1)
Y a t-il à manger? (a_2)
Est-ce que mes potes y vont? (a_3)
s

If $a_1 + a_2 + a_3 = 3, s = 1$

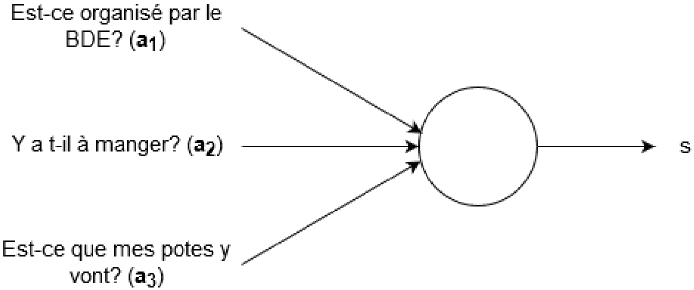
We go to the party !

Perceptron - Example


Est-ce organisé par le BDE? (a1)
Y a t-il à manger? (a2)
Est-ce que mes potes y vont? (a3)
s

If $a_1 + a_2 + a_3 > threshold, s = 1$

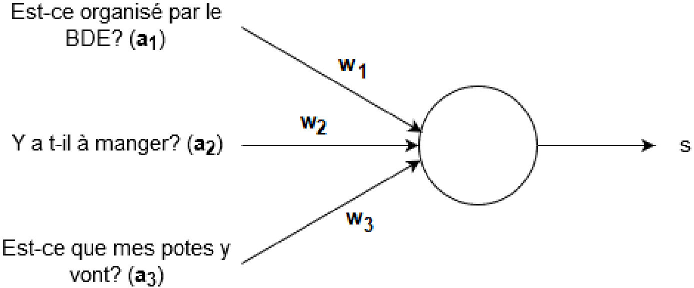
Perceptron - Example


Est-ce organisé par le BDE? ($\mathbf{a_1}$)
Y a t-il à manger? ($\mathbf{a_2}$)
Est-ce que mes potes y vont? ($\mathbf{a_3}$)
s

If $a_1 + a_2 + a_3 > threshold, s = 1$

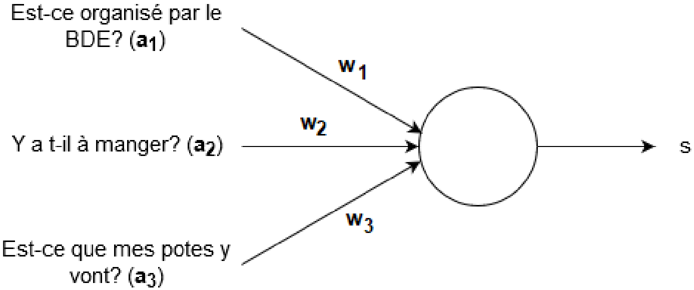
We go to the party !

Perceptron - Example


Est-ce organisé par le BDE? (a1)
Y a t-il à manger? (a2)
Est-ce que mes potes y vont? (a3)
w1
w2
w3
s

If $a_1 * w_1 + a_2 * w_2 + a_3 * w_3 + b > 0, s = 1$

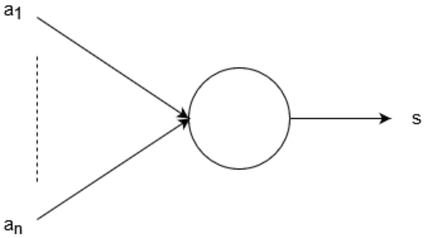
Perceptron - Example


Est-ce organisé par le BDE? (a_1)
Y a t-il à manger? (a_2)
Est-ce que mes potes y vont? (a_3)
w_1
w_2
w_3
s

$$\text{If } a_1 * w_1 + a_2 * w_2 + a_3 * w_3 + b > 0, s = 1$$

We go to the party !

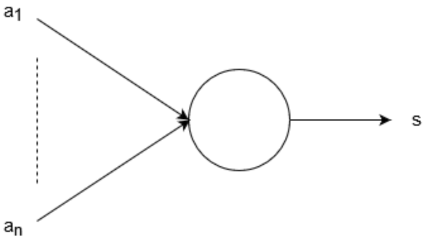
Perceptron



$a_1, ..., a_n, s \in 0, 1$

$$s = \begin{cases} 1 & \text{if } \sum_{i=0}^{n} a_i * w_i + b > 0 \\ 0 & \text{otherwise.} \end{cases}$$

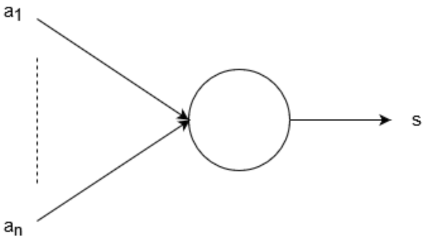
Perceptron



→ Capable of reproducing logical gates !

→ Finding the w and b by hand is a pain

Perceptron



→ Capable of reproducing logical gates !

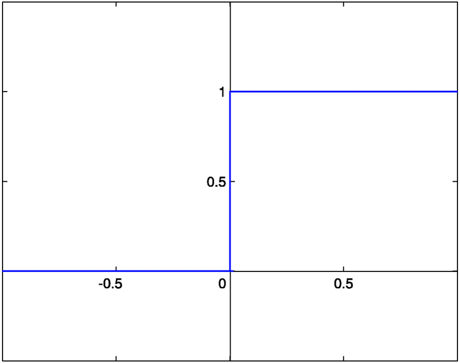
→ Finding the w and b by hand is a pain

You have to "**learn**" the w and b.

Perceptron

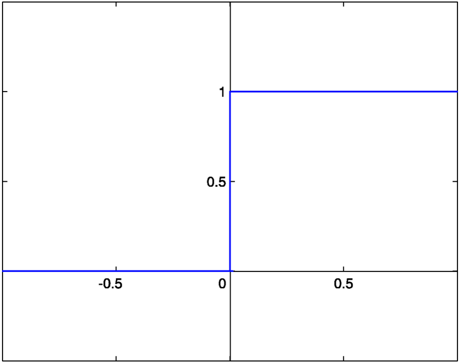
How to learn ?

Small change of w and b → small change of the output ?



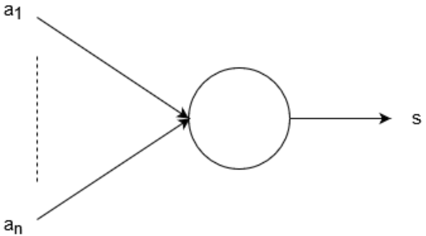
Perceptron

How to learn ?
Small change of w and b → small change of the output ?



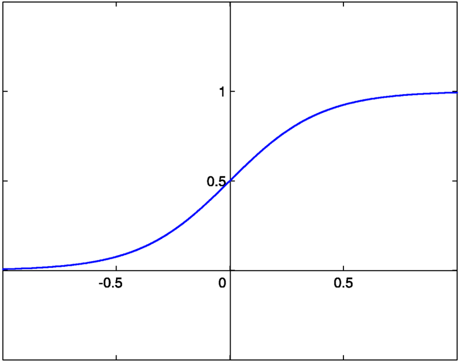
Not possible here

Sigmoid neuron



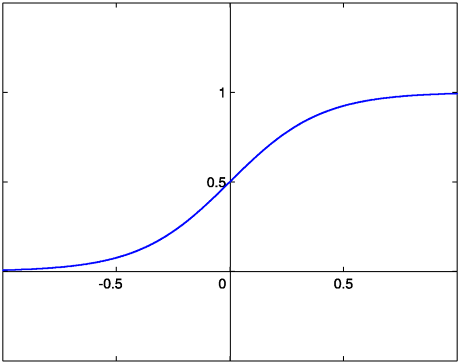
$a_1, ..., a_n \in [0, 1]$
$s = \sigma(\sum_{i=0}^{n} a_i * w_i + b)$ with $\sigma(x) = \frac{1}{1+e^{-x}}$

Sigmoid neuron



Small change of w and b → small change of the output ✓

Sigmoid neuron



Small change of w and b → small change of the output ✓

Goal how do we train ?

Training

Objective : minimize the error on the predictions !

Training

Objective : minimize the error on the predictions !

$$\left\{\omega, b | E(\omega, b) = \min_{\omega', b'} E(\omega', b')\right\}$$

Training

Problems to solve :

- How to quantify the error ?

Training

Problems to solve :

- How to quantify the error ?
  → *Quadratic loss* : Average over multiple inputs

$$L = \frac{1}{n}\sum(desired - predicted)^2$$

Training

Problems to solve :

- How to quantify the error ?
  → *Quadratic loss* : Average over multiple inputs
  $$L = \frac{1}{n}\sum(desired - predicted)^2$$
- How to minimize ?

Training

Problems to solve :

- How to quantify the error ?
  → *Quadratic loss* : Average over multiple inputs

$$L = \frac{1}{n} \sum (desired - predicted)^2$$

- How to minimize ?
  → *Backpropagation*

Training

Gradient Descent

Training

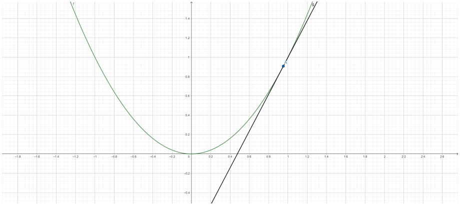
Gradient Descent

Idea : reach the minimum of a function iteratively

Training

Gradient Descent

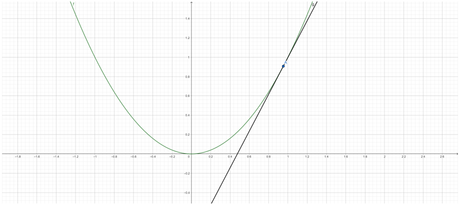
Idea : reach the minimum of a function iteratively

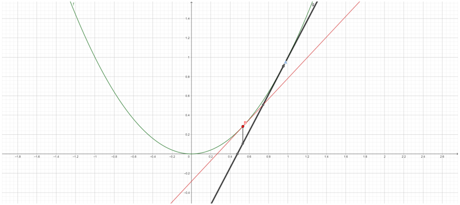


Training

Gradient Descent

Idea : reach the minimum of a function iteratively





Training

For each neuron :

$$\omega' = \omega - \eta \frac{\partial L}{\partial \omega} \tag{1}$$

$$b' = b - \eta \frac{\partial L}{\partial b} \tag{2}$$

Training

For each neuron :

$$\omega' = \omega - \eta \frac{\partial L}{\partial \omega} \tag{1}$$

$$b' = b - \eta \frac{\partial L}{\partial b} \tag{2}$$

Objective : Compute ∇L

Training

For each neuron :

$$\omega' = \omega - \eta \frac{\partial L}{\partial \omega} \tag{1}$$

$$b' = b - \eta \frac{\partial L}{\partial b} \tag{2}$$

Objective : Compute ∇L

→ one layer ✓

Training

For each neuron :

$$\omega' = \omega - \eta \frac{\partial L}{\partial \omega} \tag{1}$$

$$b' = b - \eta \frac{\partial L}{\partial b} \tag{2}$$

Objective : Compute ∇L
→ one layer ✓
→ multiple layers :

Training

For each neuron :

$$\omega' = \omega - \eta \frac{\partial L}{\partial \omega} \tag{1}$$

$$b' = b - \eta \frac{\partial L}{\partial b} \tag{2}$$

Objective : Compute ∇L
→ one layer ✓
→ multiple layers : Propagation of the gradient upstream of the network with the chain rule : *Backpropagation* → **cs231**

Training

- **1st approach**

  For each entry :

  → Compute the error (loss)

  → Compute the gradient

  → Update the parameters

Training

- **1st approach**
  For each entry :
  → Compute the error (loss)
  → Compute the gradient
  → Update the parameters
- **2nd approach**
  For each *batch* :
  → Compute the mean error
  → Compute the gradient
  → Update the parameters

Training

Summary :

→ Select a *batch*

→ For each entry, compute the output : *Forward propagation*

→ Compute the mean error (loss)

→ Compute the gradient and update the parameters : *Backpropagation*

Training

Summary :

→ Select a *batch*

→ For each entry, compute the output : *Forward propagation*

→ Compute the mean error (loss)

→ Compute the gradient and update the parameters : *Backpropagation*

And now, it's your turn !

Discord

Join our Discord server !

Useful to ask questions, contact Ens'IA team, and to share news !

→ https ://discord.gg/UgTRbRFqNv