Introduction to Artificial Intelligence

Philippe Leleux
LAAS-CNRS - Équipe TRUST

Summer school: Cyber in Font-Romeu
July 7th 2025

Who am I ?

Philippe Leleux

- Associate professor at INSA de Toulouse, LAAS-CNRS, Equipe TRUST
- Teaching : machine learning for critical embedded systems
- Research :
  - How to make machine learning techniques more "trustworthy" ?
    => Application to medical diagnostic, pronostic, treatment decision
  - How to use machine learning for safety (including cybersecurity) ?
    => Detection of hardware trojans based on micro-architectural signals

AI
What ? Why ? Where ? When ?

Let's start with some questions

- When did the term artificial intelligence appear ?

Let's start with some questions

- ➢ When did the term artificial intelligence appear ?
  => 1956, Dartmouth College

- ➢ Who among you uses generative AI regularly ?

Let's start with some questions

- ➢ When did the term artificial intelligence appear ?
  => 1956, Dartmouth College

- ➢ Who among you uses generative AI regularly ?

- ➢ Who among you uses AI everyday ?

Let's start with some questions

- When did the term artificial intelligence appear ?
  => 1956, Dartmouth College
- Who among you uses generative AI regularly ?
- Who among you uses AI everyday ?
  => all
- Who has set up machine learning algorithms ?

How many fingers ?

Let's start with some questions

- When did the term artificial intelligence appear ?
  => 1956, Dartmouth College

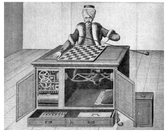
- Who among you uses generative AI regularly ?

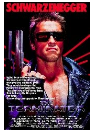
- Who among you uses AI everyday ?
  => all

- Who has set up machine learning algorithms ?
  => scikit-learn, Tensorflow, Pytorch
  => Typically neural networks

How many fingers ?

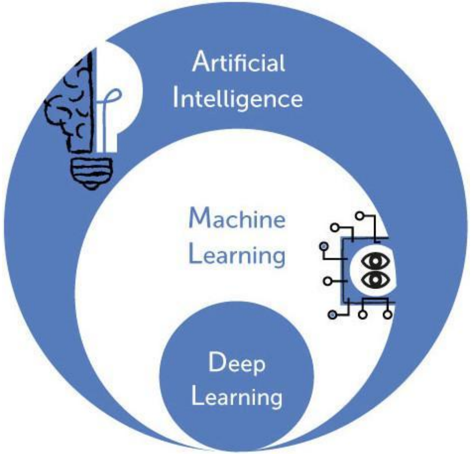
What is AI ?



➢ What AI is:
- IA = program trying to imitate human logic (~50s)
- example : 4 legs + 1 sit + 1 back = chair

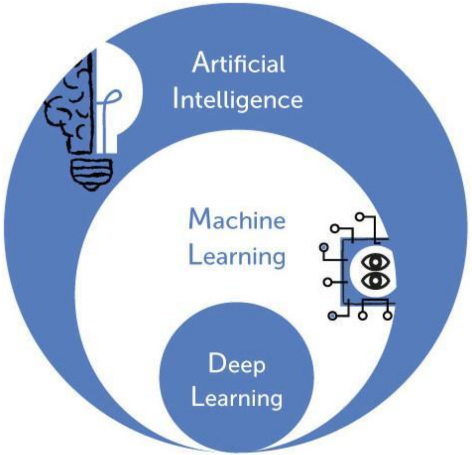




What is AI ?

➢ What AI is:

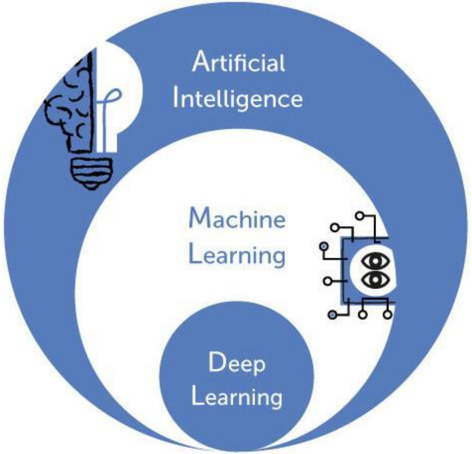
- IA = program trying to imitate human logic (~50s)
- Machine learning
  - data => model => answer
  - example : lots of chairs vs. lots of non-chair

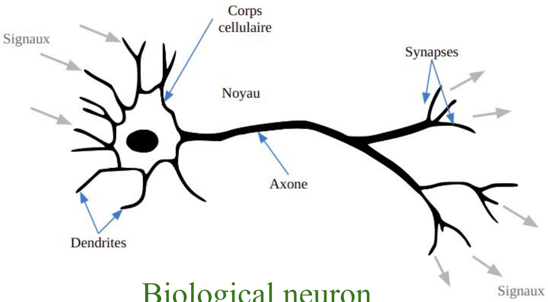


What is AI ?

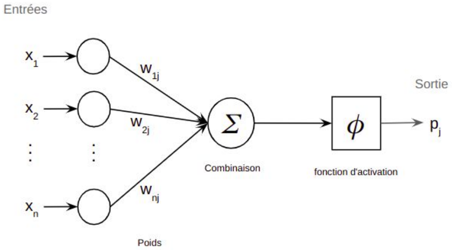
➢ What AI is:

- IA = program trying to imitate human logic (~50s)
- Machine learning
  - data => model => answer
  - workflow + set of algorithms
- Deep learning : neural networks
  - Inspired from the brain
  - example : facial recognition, ChatGPT, …





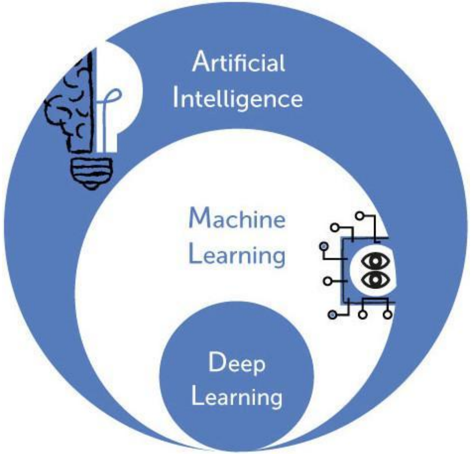
Biological neuron



Artificial neuron

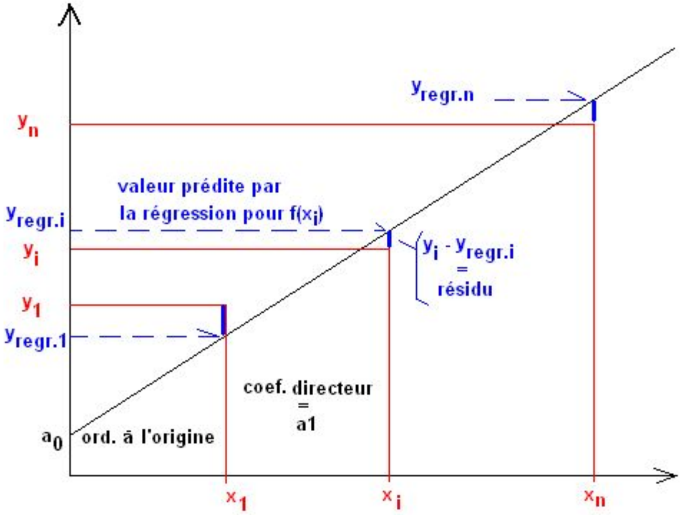
What is AI ?

- **What AI is:**
  - IA = program trying to imitate human logic (~50s)
  - Machine learning
    - data => model => answer
    - workflow + set of algorithms
  - Deep learning : neural networks
    - Inspired from the brain
    - example : facial recognition, ChatGPT, …
- What AI is **not** :
  - **"Intelligent", "sentient", a "mystical entity"**
  - **A miracle solution to all problems**
  - **A danger for humanity**
- Must you be an expert to use machine learning ? Certainly not.



What is AI ?

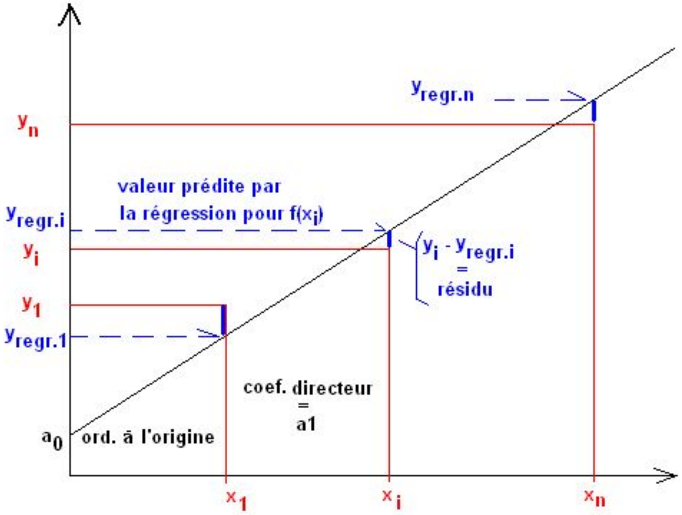
- Must you be an expert to use machine learning ? Certainly not.
- Do you know affine functions ?

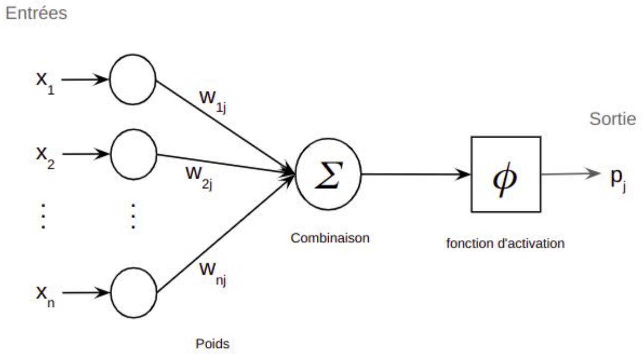


What is AI ?

- Must you be an expert to use machine learning ? Certainly not.
- Do you know affine functions ?

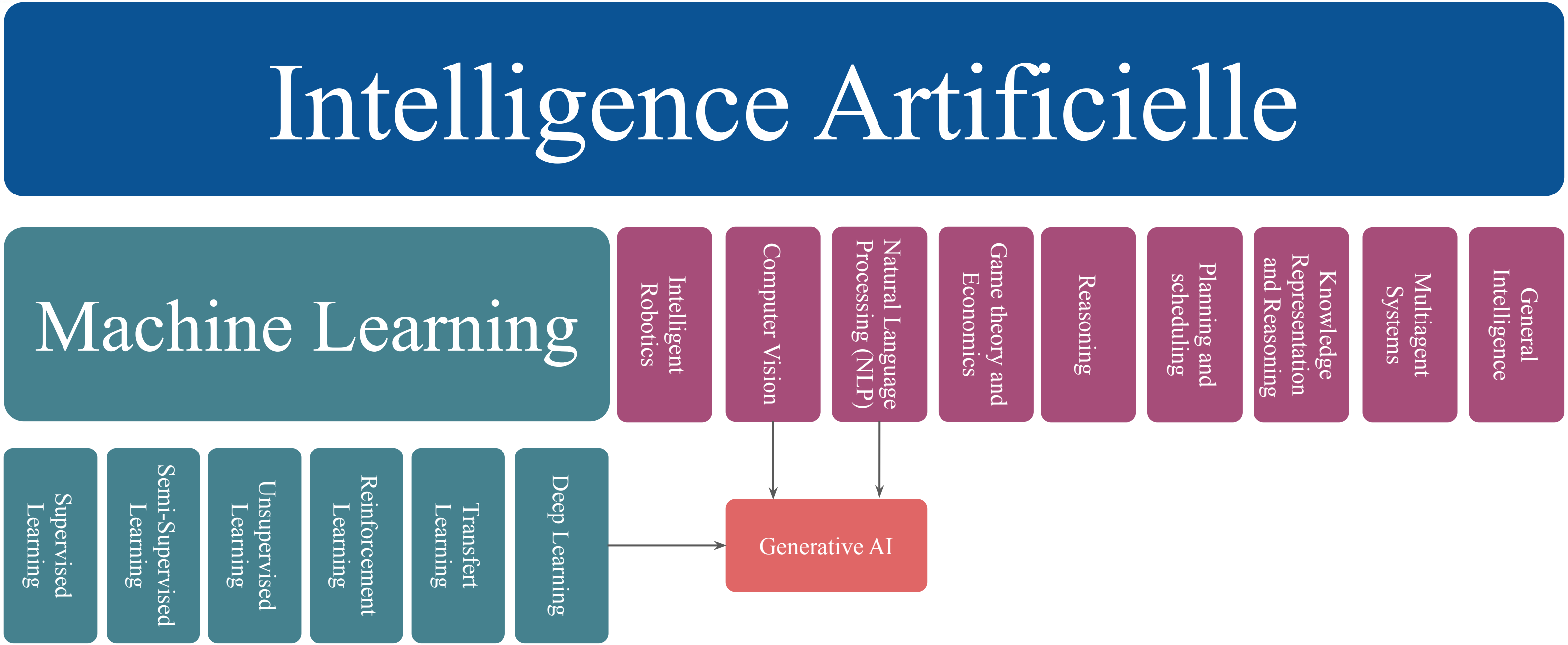
=> Congrats, you now know how an artificial neuron works! (mostly)





Artificial neuron

Types of AI and Tasks



Machine Learning is a subset of Artificial Intelligence. The term Artificial Intelligence is often misused (buzzword in the sense of global intelligence).

Types of AI and Tasks

Intelligence Artificielle

Machine Learning

- Supervised Learning
- Semi-Supervised Learning
- Unsupervised Learning
- Reinforcement Learning
- Transfert Learning
- Deep Learning

Intelligent Robotics

Computer Vision

Natural Language Processing (NLP)

Game theory and Economics

Reasoning

Planning and scheduling

Knowledge Representation and Reasoning

Multiagent Systems

General Intelligence

Generative AI

Par exemple, deep-blue exécute un algorithme de recherche alpha-beta test, ce n'est pas du ML.

Machine Learning is a subset of Artificial Intelligence. The term Artificial Intelligence is often misused (buzzword in the sense of global intelligence).

Types of AI and Tasks

Intelligence Artificielle

Machine Learning

- Supervised Learning
- Semi-Supervised Learning
- Unsupervised Learning
- Reinforcement Learning
- Transfert Learning
- Deep Learning

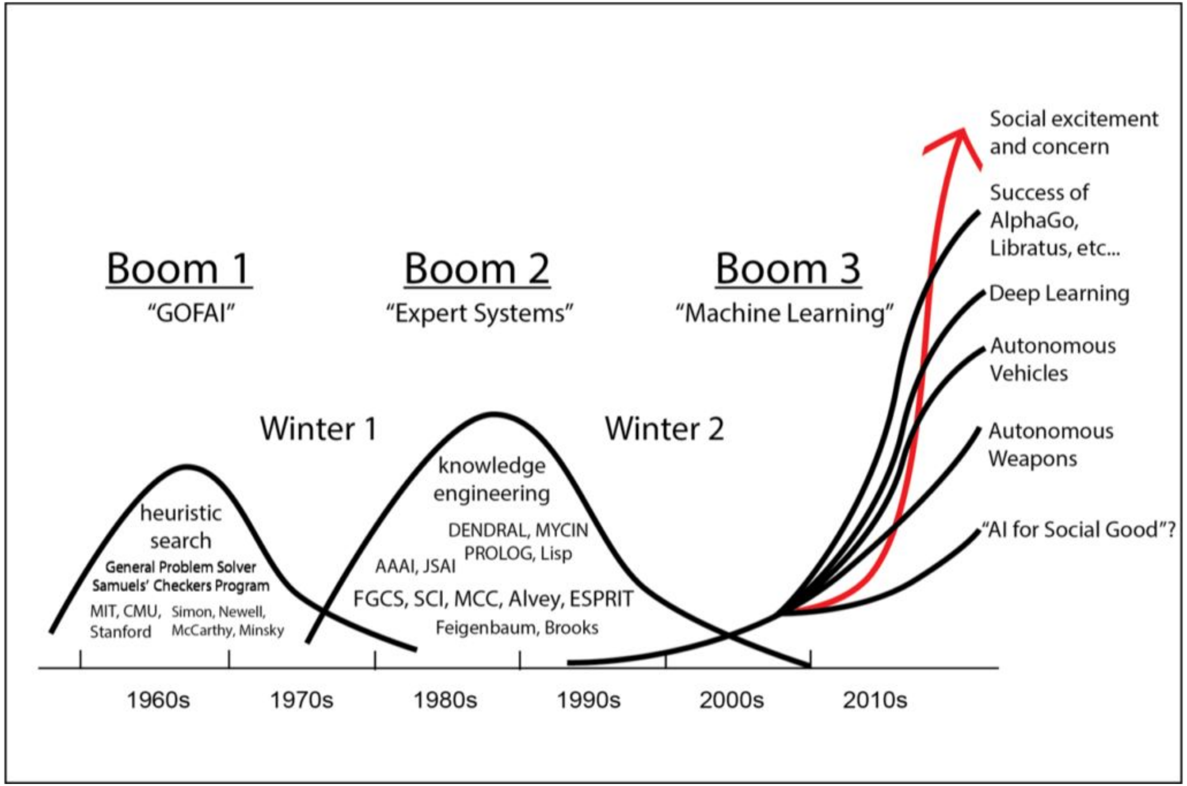
- Intelligent Robotics
- Computer Vision
- Natural Language Processing (NLP)
- Game theory and Economics
- Reasoning
- Planning and scheduling
- Knowledge Representation and Reasoning
- Multiagent Systems
- General Intelligence

Deep Learning → Generative AI
Computer Vision → Generative AI
Natural Language Processing (NLP) → Generative AI

Par exemple, deep-blue exécute un algorithme de recherche alpha-beta test, ce n'est pas du ML.

Machine Learning is a subset of Artificial Intelligence. The term Artificial Intelligence is often misused (buzzword in the sense of global intelligence).

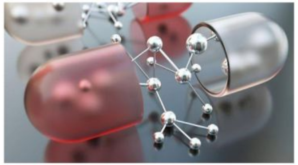
Types of AI and Tasks

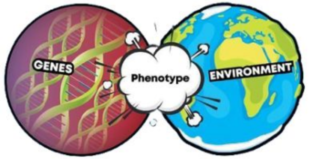


Real-life examples

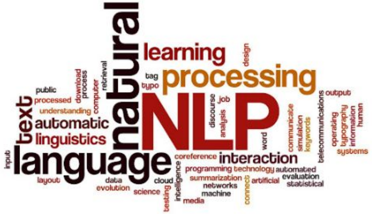
Welcome to the AI era

Biology



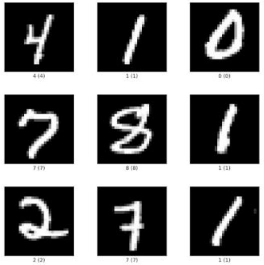


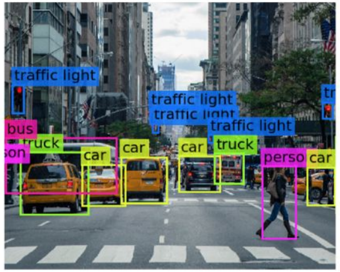
Natural Language Processing

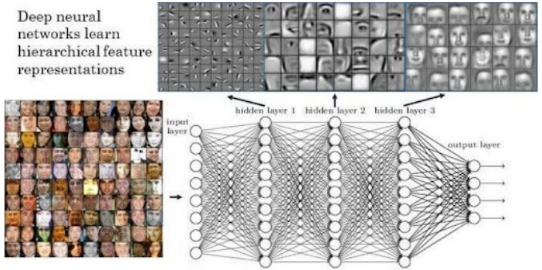


ChatGPT

Computer Vision



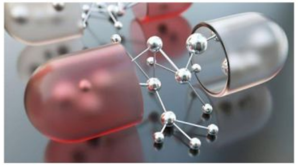


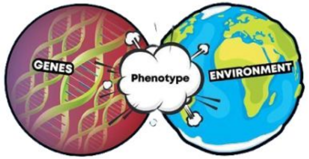


Real-life examples

Welcome to the AI era

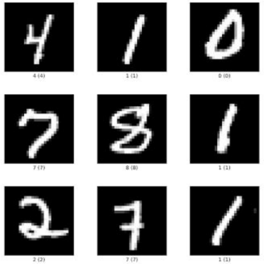
Biology

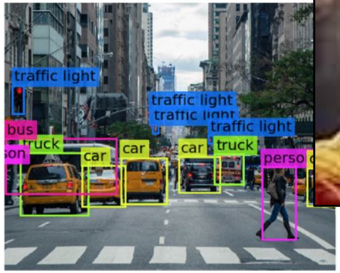


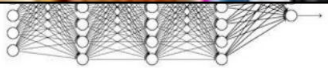


Natural Language Processing

Compute







Real-life examples

Welcome to the AI era

Finance worker pays out $25 million after video call with deepfake 'chief financial officer'

By Heather Chen and Kathleen Magramo, CNN

2 minute read · Published 2:31 AM EST, Sun February 4, 2024

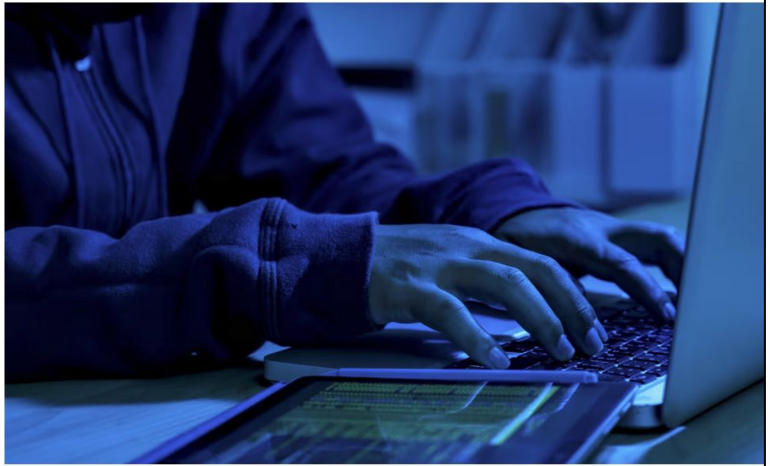
Natural Language Processing

Real-life examples

Welcome to the AI era

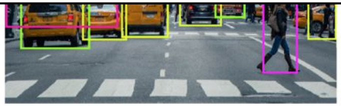
Finance worker pays out $25 million after video call with deepfake 'chief financial officer'

By Heather Chen and Kathleen Magramo, CNN
2 minute read · Published 2:31 AM EST, Sun February 4, 2024



Natural Language Processing

Viral scam: French woman duped by AI Brad Pitt love scheme faces cyberbullying



AI
How ?

~~AI~~

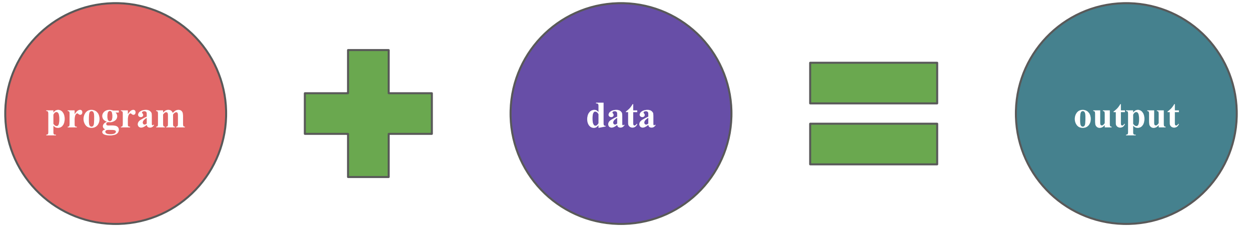
Machine Learning

How ?

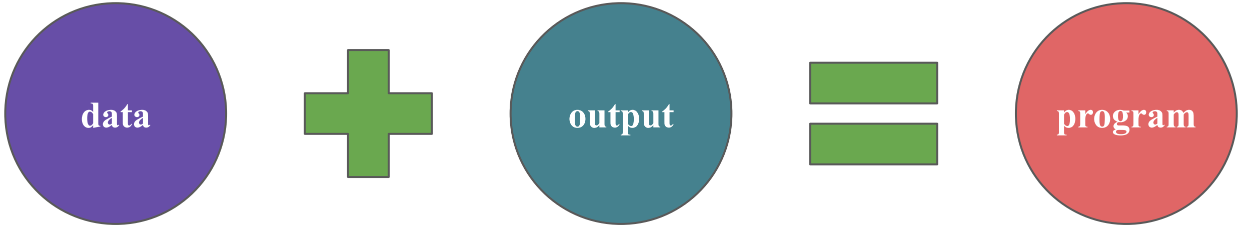
Machine learning

Paradigm

- Traditional approach



- Machine Learning



“Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed”. Arthur Samuel (1959)

Machine learning Steps

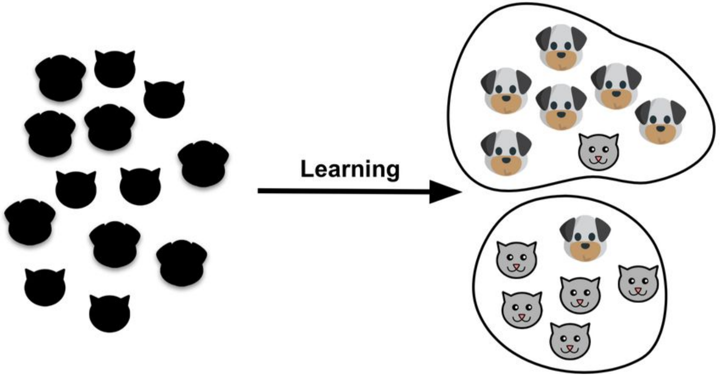
Phase 1

Data

(Creation/collection/storage/ grooming)

Phase 2

Training and testing

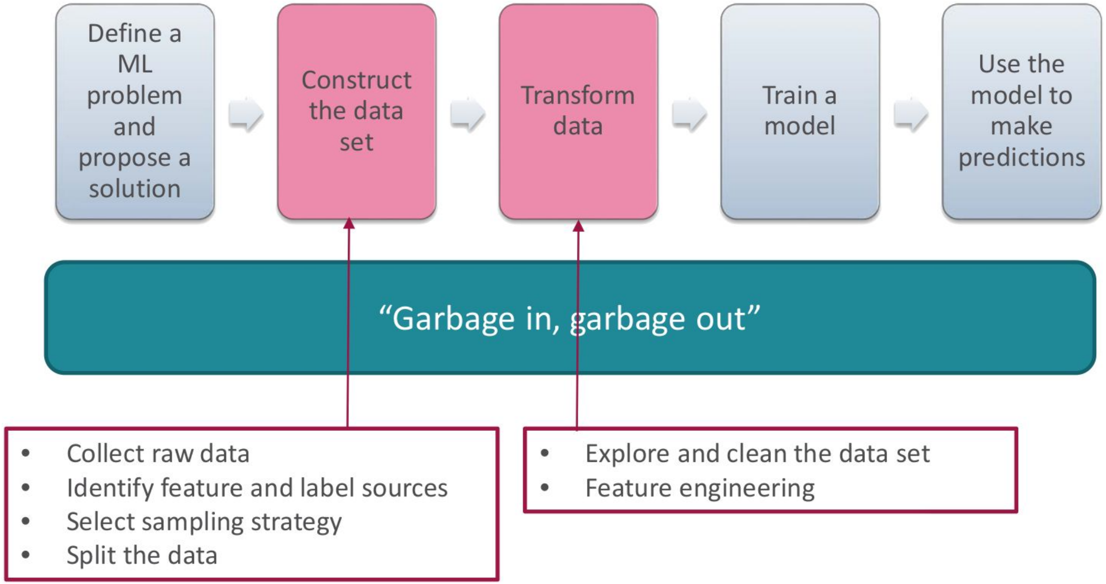


Phase 3

Use (inference)

Machine learning

Data preparation

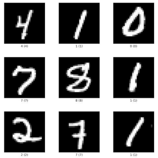


What types of data ?

How much of the whole development process is spent on data ?

Machine learning

3 types of learning

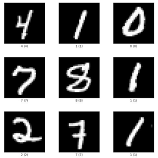


Dataset MNIST :

https://www.kaggle.com/datasets/hojjatk/mnist-dataset

Machine learning

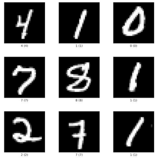
3 types of learning



Dataset MNIST :
https://www.kaggle.com/datasets/hojjatk/mnist-dataset

1. **Supervised learning :** from labelled inputs, train a model
   => e.g. classification : what number is  ?   The patient has cancer ?

Machine learning
3 types of learning

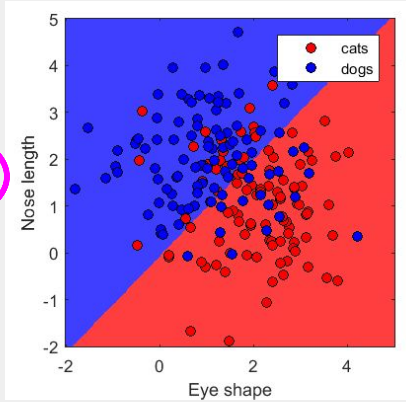


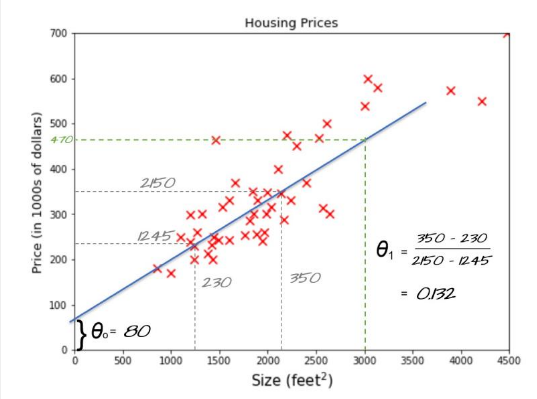
Dataset MNIST :
https://www.kaggle.com/datasets/hojjatk/mnist-dataset

1. **Supervised learning :** from labelled inputs, train a model
   => e.g. classification : what number is  ?

The patient has cancer ?

classification
(qualitative)

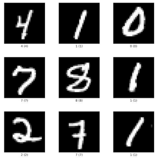




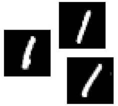
regression
(quantitative)

Machine learning

3 types of learning

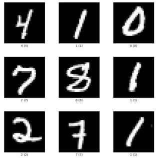


Dataset MNIST :
https://www.kaggle.com/datasets/hojjatk/mnist-dataset

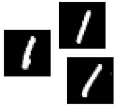
1. **Supervised learning :** from labelled inputs, train a model
   => e.g. classification : what number is  ?   The patient has cancer ?
2. **Unsupervised learning :** from unlabelled inputs, find a structure
   => e.g. clustering : group together  ;   Group of patients => specific drug

Machine learning

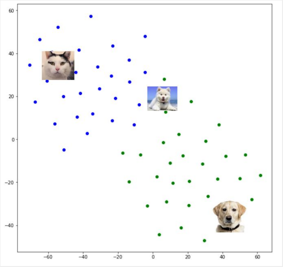
3 types of learning



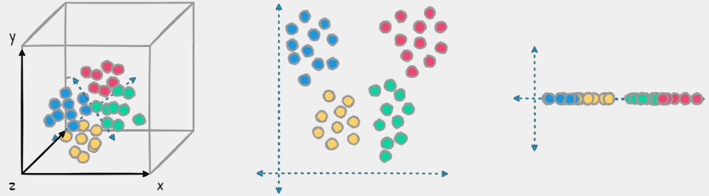
Dataset MNIST :
https://www.kaggle.com/datasets/hojjatk/mnist-dataset

1. **Supervised learning :** from labelled inputs, train a model
   => e.g. classification : what number is  ? The patient has cancer ?
2. **Unsupervised learning :** from unlabelled inputs, find a structure
   => e.g. clustering : group together ; Group of patients => specific drug

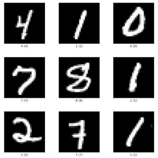
clustering



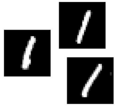
dimension reduction

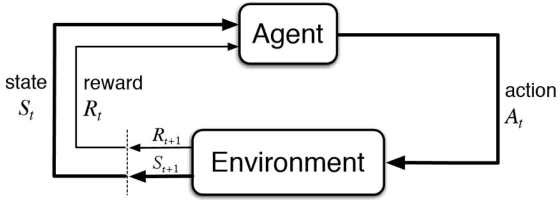


Machine learning
3 types of learning



Dataset MNIST :
https://www.kaggle.com/datasets/hojjatk/mnist-dataset

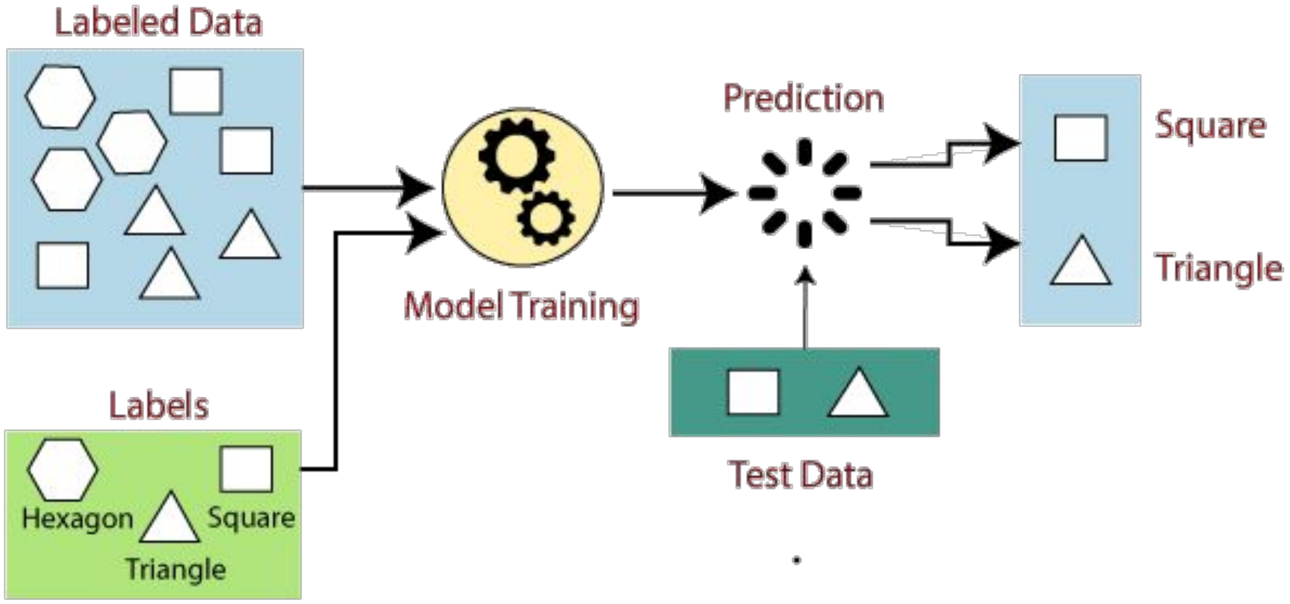
1. **<u>Supervised learning :</u>** from labelled inputs, train a model
   => e.g. classification : what number is  ?   The patient has cancer ?
2. **<u>Unsupervised learning :</u>** from unlabelled inputs, find a structure
   => e.g. clustering : group together  ;   Group of patients => specific drug
3. **<u>Reinforcement learning :</u>** from environment and reward, train an agent



Supervised learning

Supervised learning
Task

➢ Input : dataset with labels (given by experts)

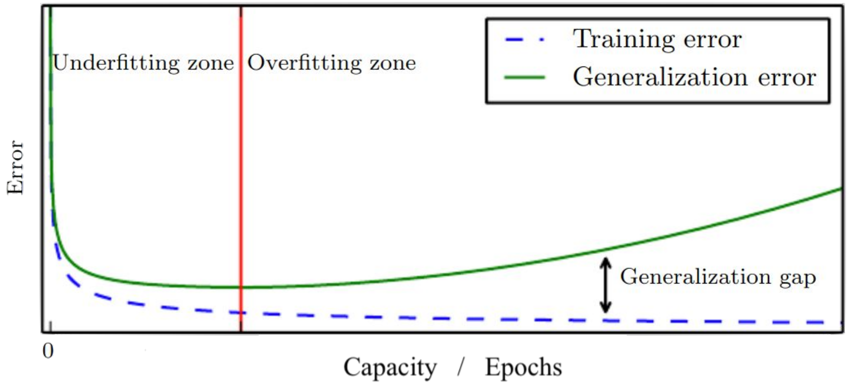


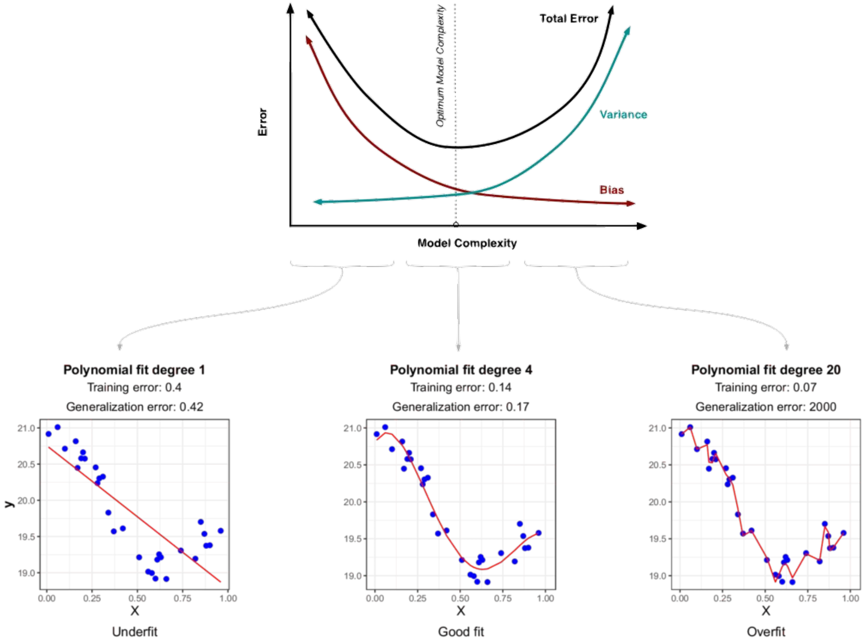
Supervised learning

Task

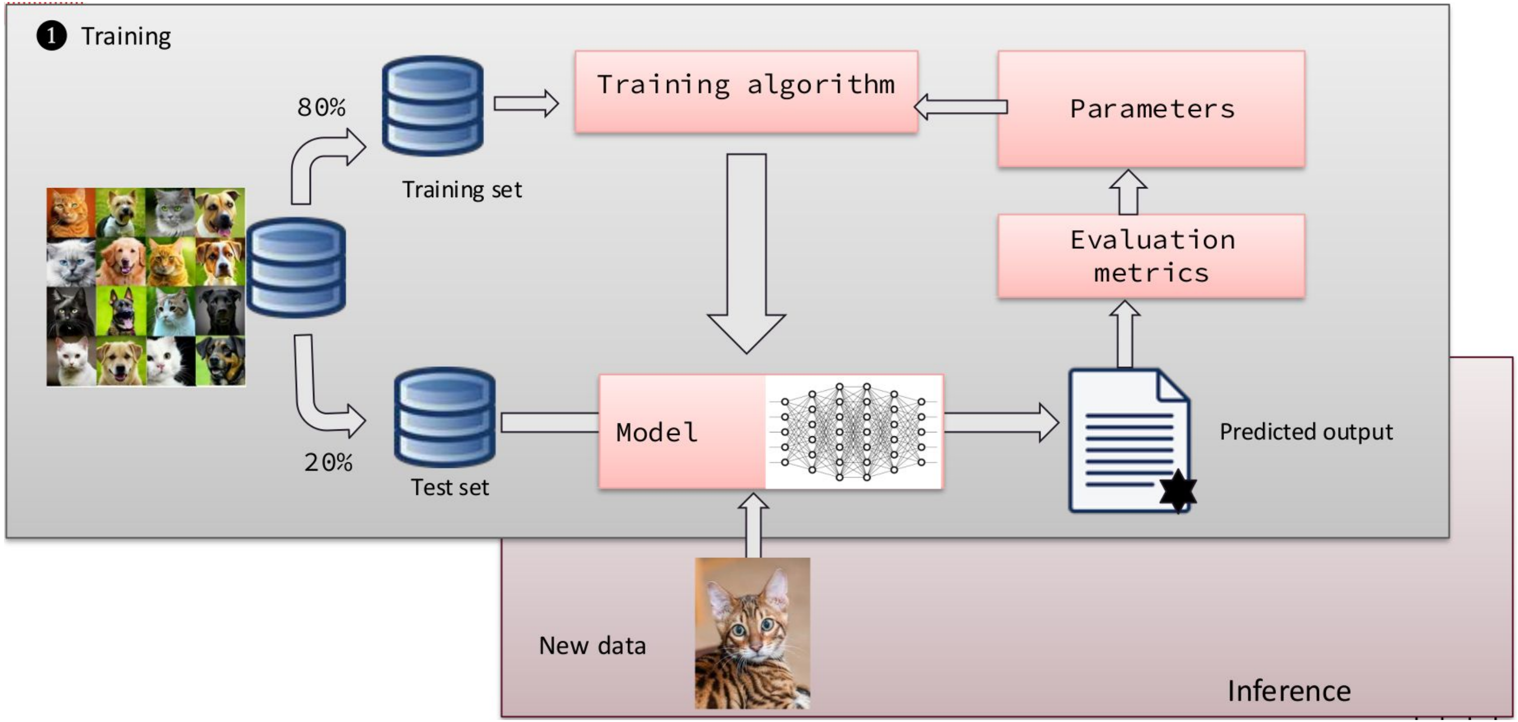
- **Goal :** find f such that $\hat{y}_i = f(x) \approx y_i$ by minimizing an error/loss function
- For example: Mean Squared Error (MSE) :

$$\frac{1}{m}\sum_{i=1}^{m}(\hat{y}_i - y_i)^2$$

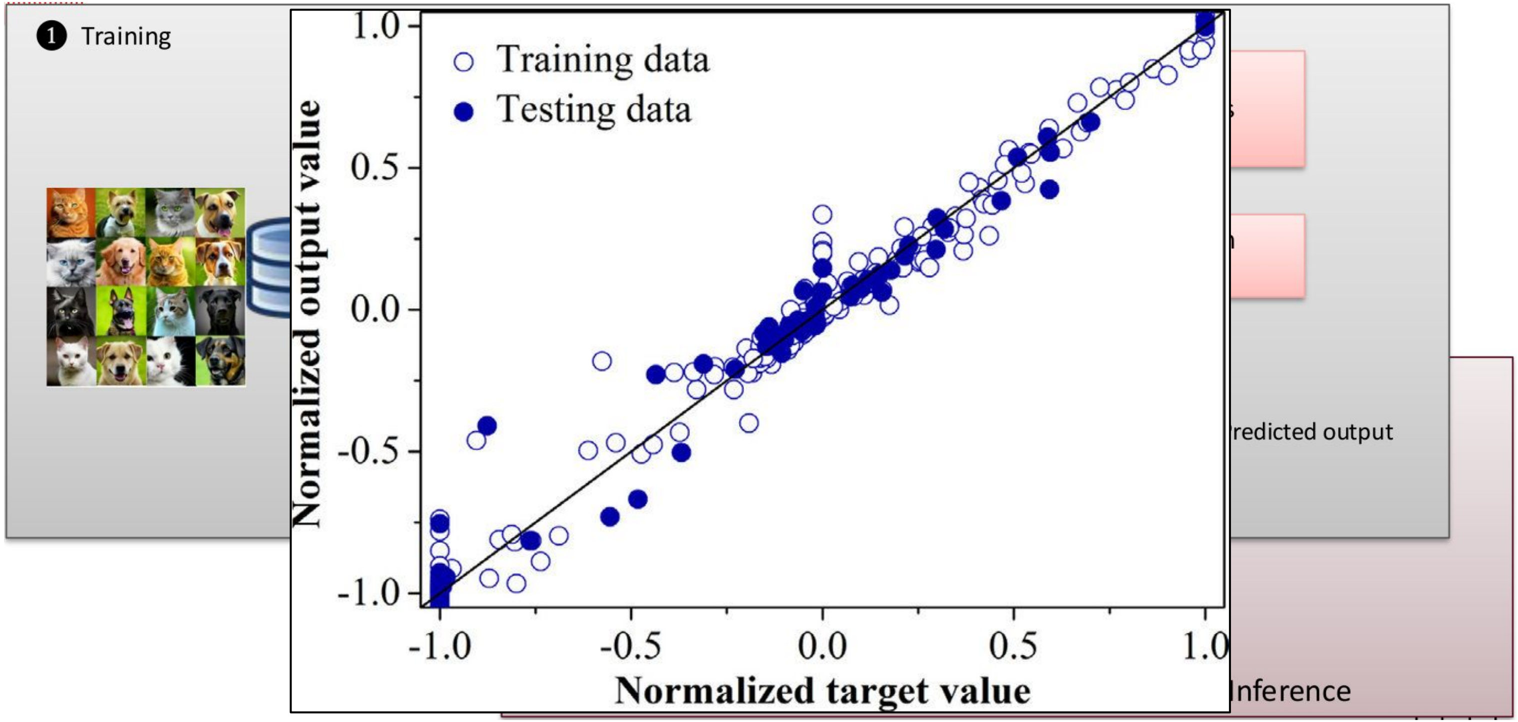




Supervised learning
Classical workflow

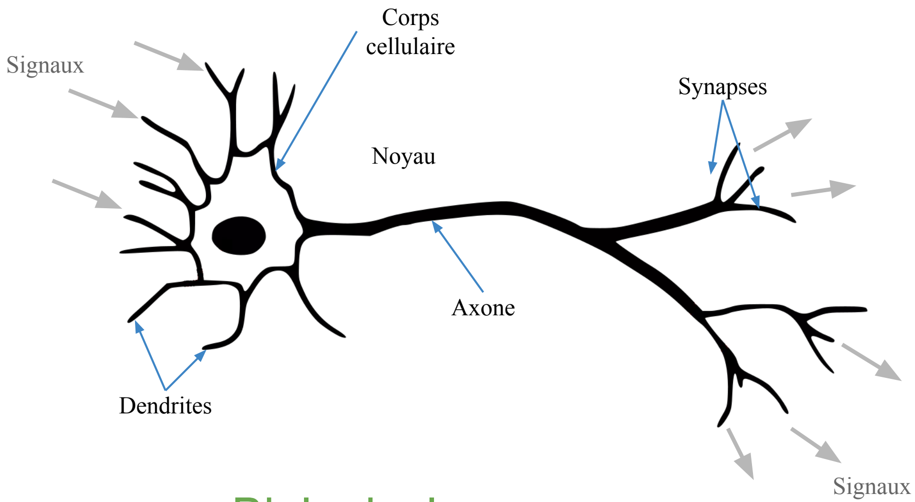


Supervised learning
Classical workflow

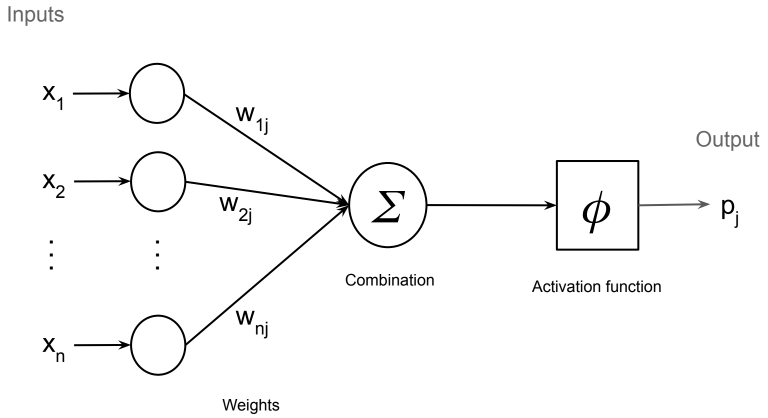


Deep learning

Neural networks
biomimicry



Biological neuron



Artificial neuron

First neural network
Fruits classification

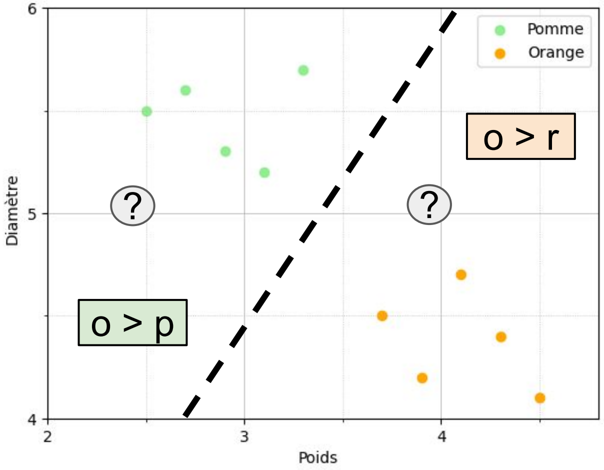
Labels

Data

x_1	x_2
2.5	5.5
2.7	5.6
2.9	5.3
3.1	5.2
3.3	5.7

x_1	x_2
3.7	4.5
3.9	4.2
4.1	4.7
4.3	4.4
4.5	4.1

$\mathbf{x_1}$ = weight
$\mathbf{x_2}$ = diameter



layers: input output

p: probability apple

o: probability orange

First neural network

Combining inputs

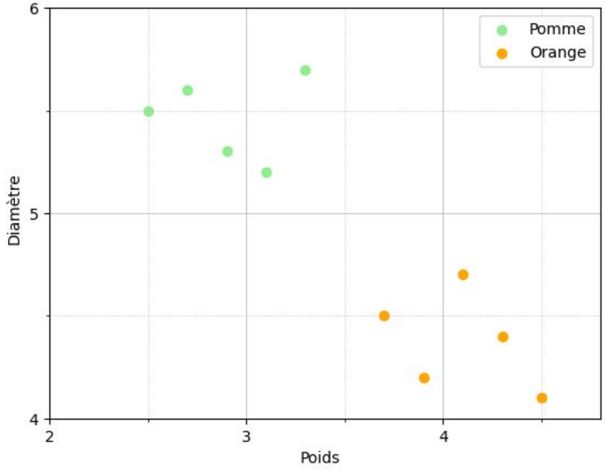
Labels

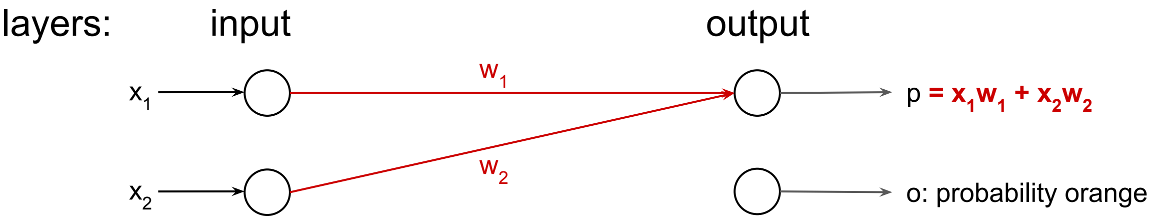
Data

x_1	x_2
2.5	5.5
2.7	5.6
2.9	5.3
3.1	5.2
3.3	5.7

x_1	x_2
3.7	4.5
3.9	4.2
4.1	4.7
4.3	4.4
4.5	4.1

$\mathbf{x_1}$ = weight
$\mathbf{x_2}$ = diameter

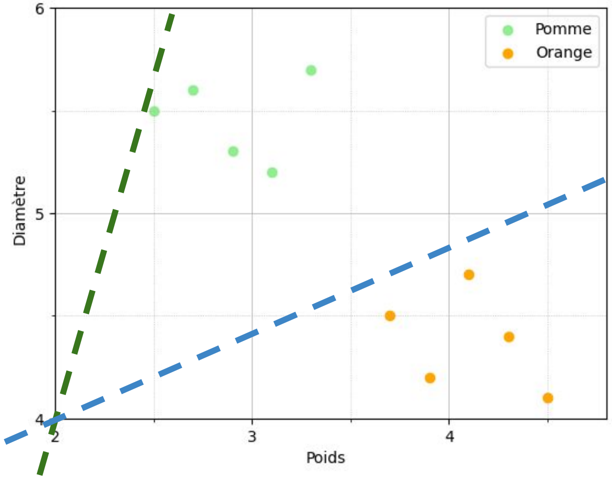




First neural network

Linear separation

Labels



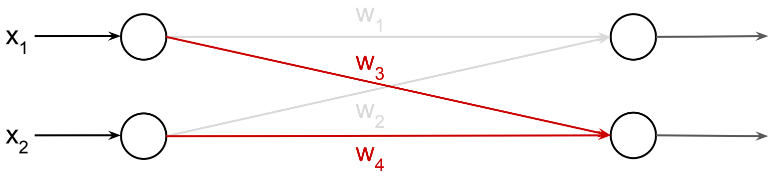
Data

x_1	x_2
2.5	5.5
2.7	5.6
2.9	5.3
3.1	5.2
3.3	5.7

x_1	x_2
3.7	4.5
3.9	4.2
4.1	4.7
4.3	4.4
4.5	4.1

$\mathbf{x_1}$ = weight
$\mathbf{x_2}$ = diameter

layers: input output



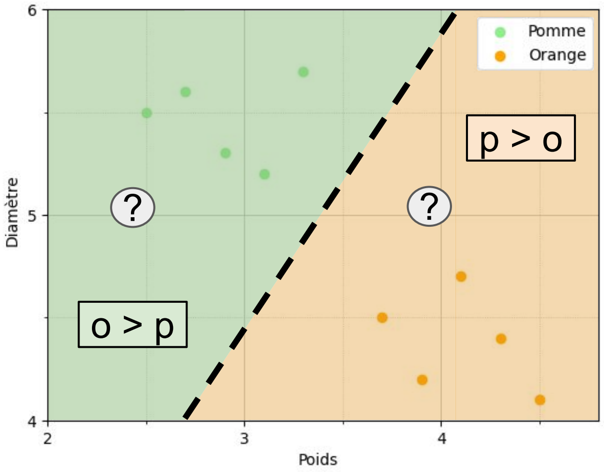
$p = x_1w_1 + x_2w_2$

$o = x_3w_3 + x_4w_4$

First neural network

Affine separation: bias

Labels

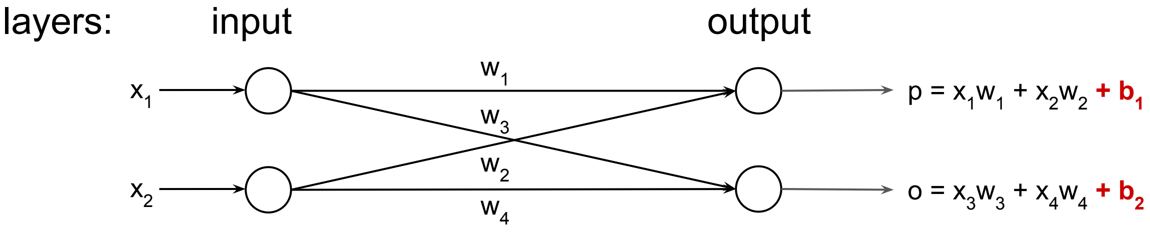


Data

x_1	x_2
2.5	5.5
2.7	5.6
2.9	5.3
3.1	5.2
3.3	5.7

x_1	x_2
3.7	4.5
3.9	4.2
4.1	4.7
4.3	4.4
4.5	4.1

$\mathbf{x_1}$ = weight
$\mathbf{x_2}$ = diameter



First neural network

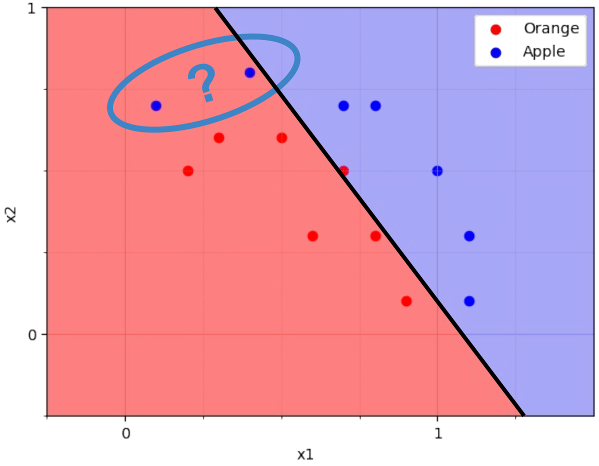
Non-linear separations?

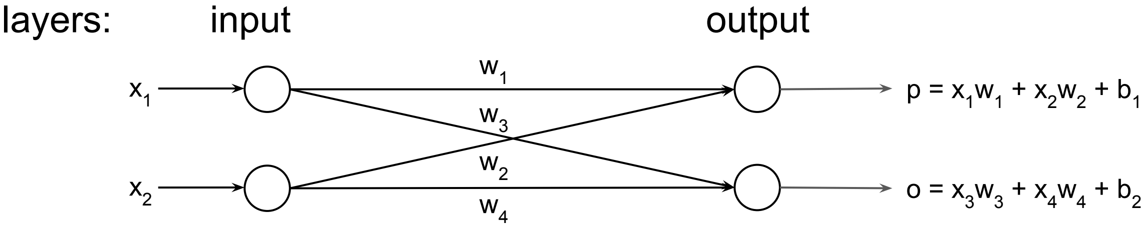
Labels

Data

x_1	x_2
0.2	0.5
0.3	0.6
0.5	0.6
0.6	0.3
0.7	0.5
0.8	0.3
0.9	0.1

x_1	x_2
0.1	0.7
0.4	0.8
0.7	0.7
0.8	0.7
1.0	0.5
1.1	0.1
1.1	0.3

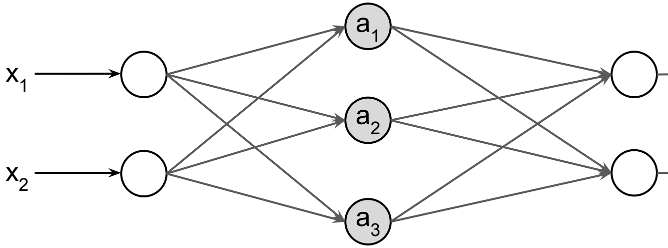




Multi-layer neural network

More complex but still linear

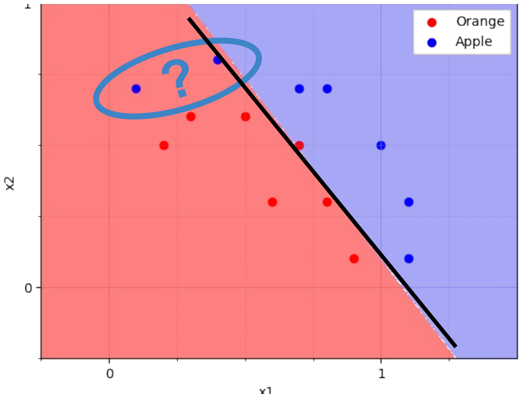
layers: input output



$p = a_1w_7 + a_2w_8 + a_3w_9 + b_4$

$o = a_1w_{10} + a_2w_{11} + a_3w_{12} + b_5$

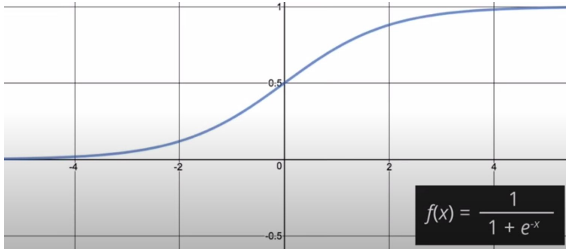
$$\begin{cases} a_1 = x_1w_1 + x_2w_2 + b_1 \\ a_2 = x_1w_3 + x_2w_4 + b_2 \\ a_3 = x_1w_5 + x_2w_6 + b_3 \end{cases}$$

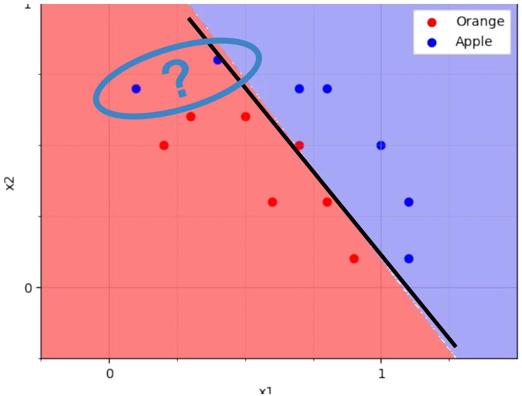


Multi-layer neural network

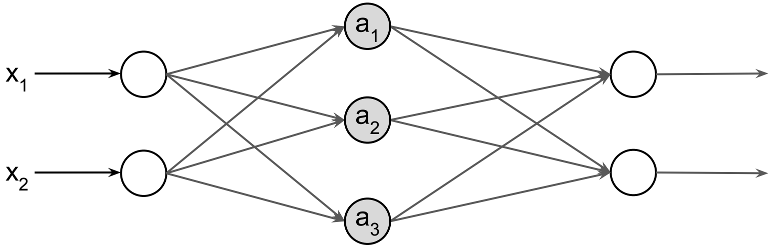
Activation function: one step towards non-linearity

ex: sigmoid function





layers: input output



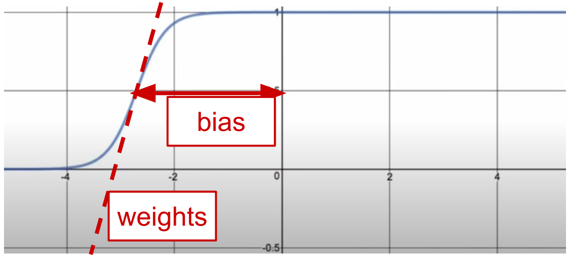
$p = f(a_1w_7 + a_2w_8 + a_3w_9 + b_4)$

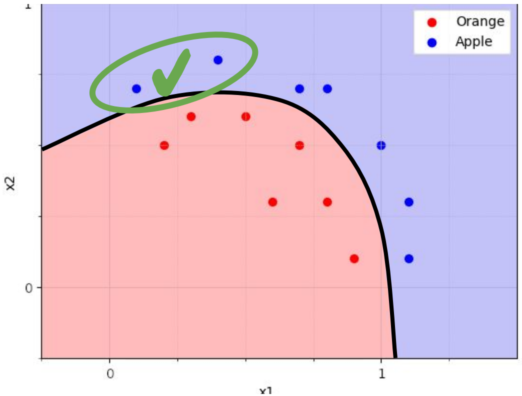
$o = f(a_1w_{10} + a_2w_{11} + a_3w_{12} + b_5)$

$$\begin{cases} a_1 = f(x_1w_1 + x_2w_2 + b_1) \\ a_2 = f(x_1w_3 + x_2w_4 + b_2) \\ a_3 = f(x_1w_5 + x_2w_6 + b_3) \end{cases}$$

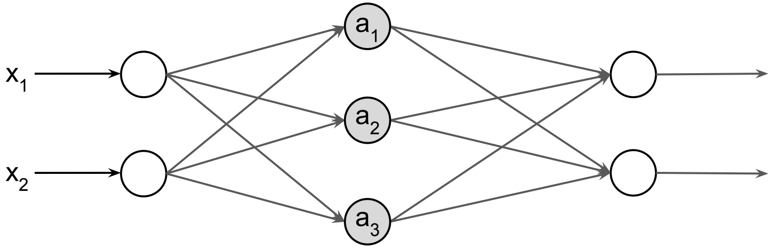
Multi-layer neural network

ex: sigmoid function





layers: input output



$p = f(a_1w_7 + a_2w_8 + a_3w_9 + b_4)$

$o = f(a_1w_{10} + a_2w_{11} + a_3w_{12} + b_5)$

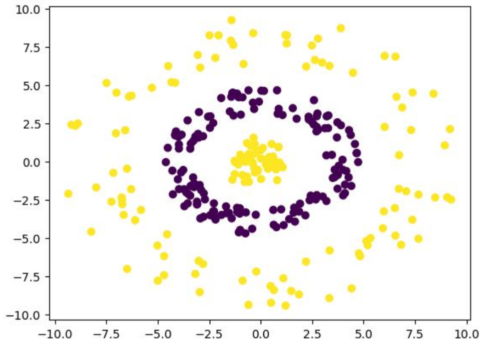
$$\begin{cases} a_1 = f(x_1w_1 + x_2w_2 + b_1) \\ a_2 = f(x_1w_3 + x_2w_4 + b_2) \\ a_3 = f(x_1w_5 + x_2w_6 + b_3) \end{cases}$$

How to set the parameters ?

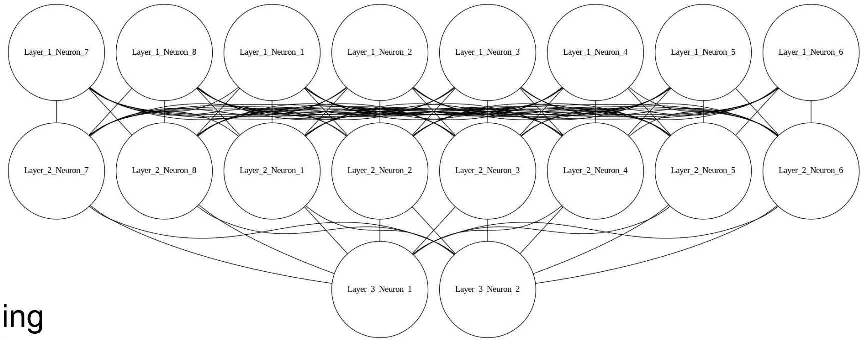
Multi-layer neural network

Automatic training

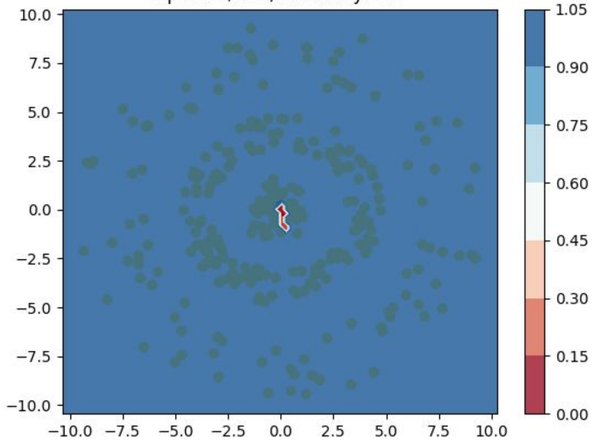
Données - labels



Network selection + activation function



Training



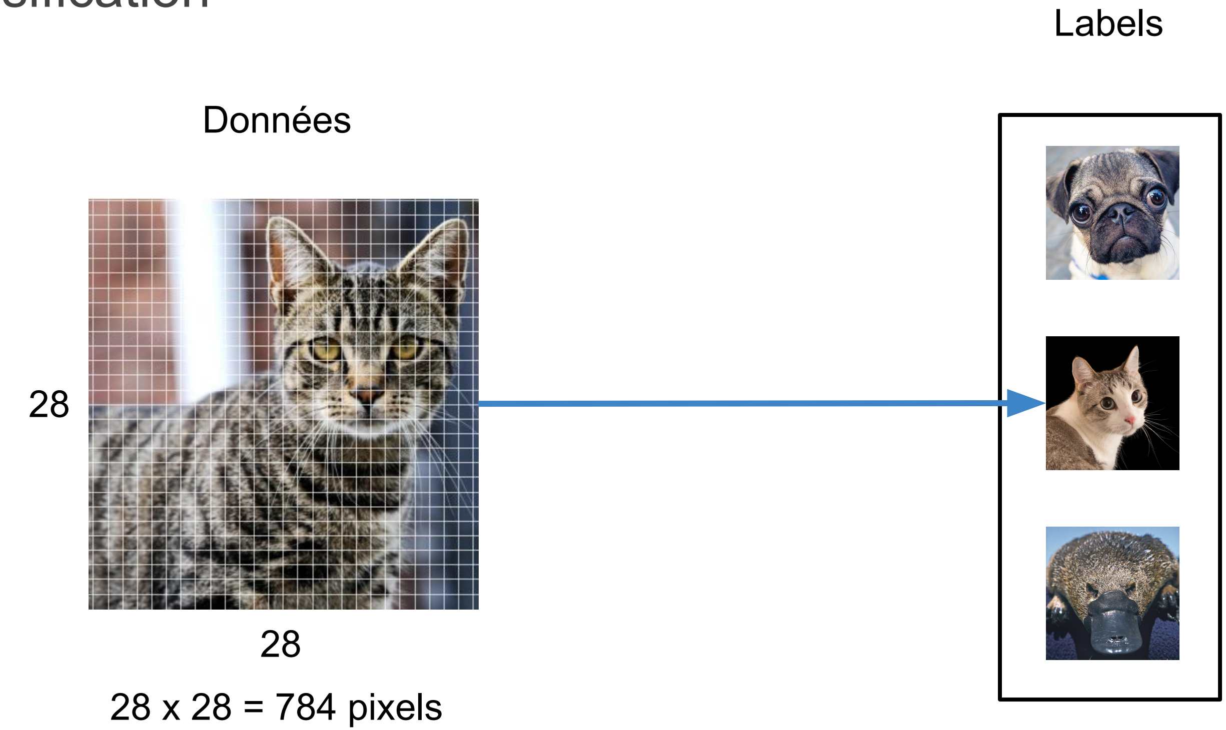
Iterative process:
updating neural network weights

Accuracy = precision:
Ratio of well-ranked points

How does it work?

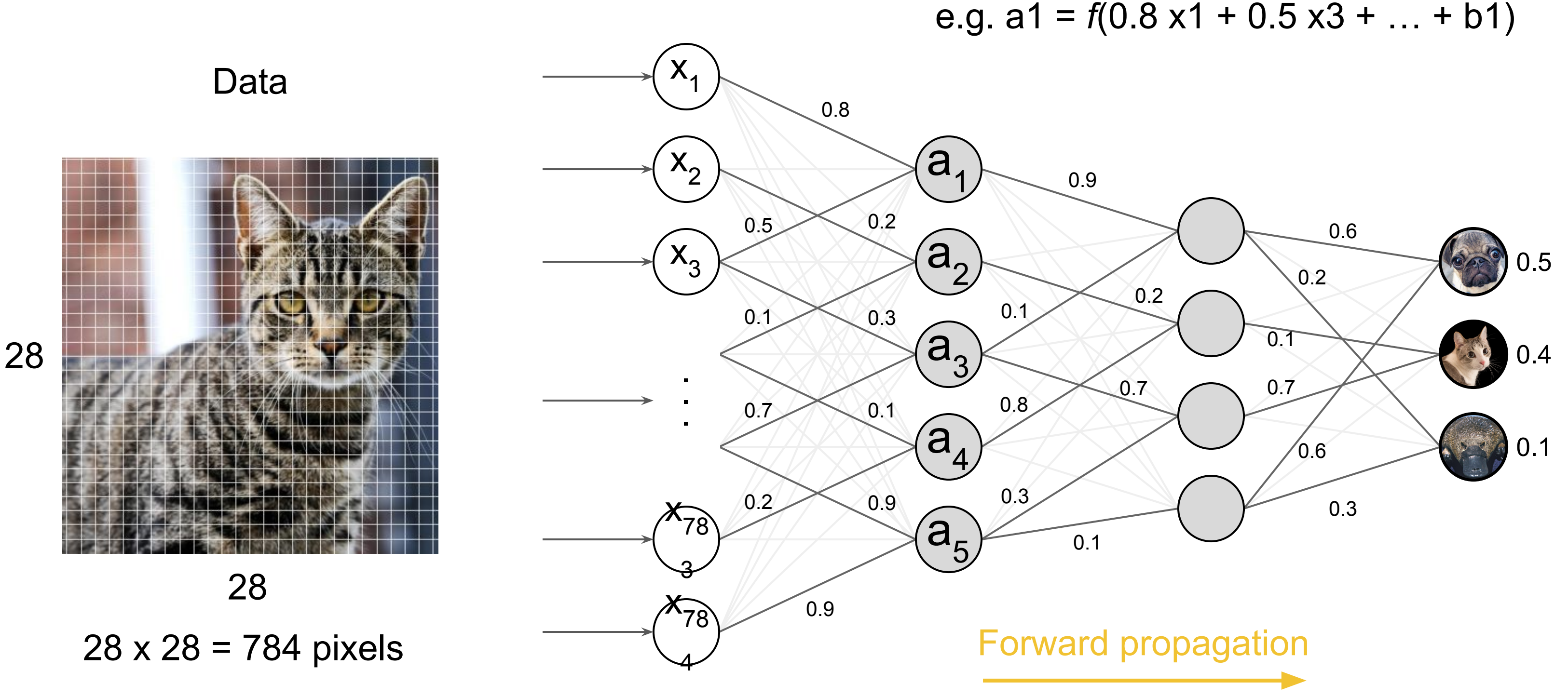
Training the neural network

Image classification



Training the neural network

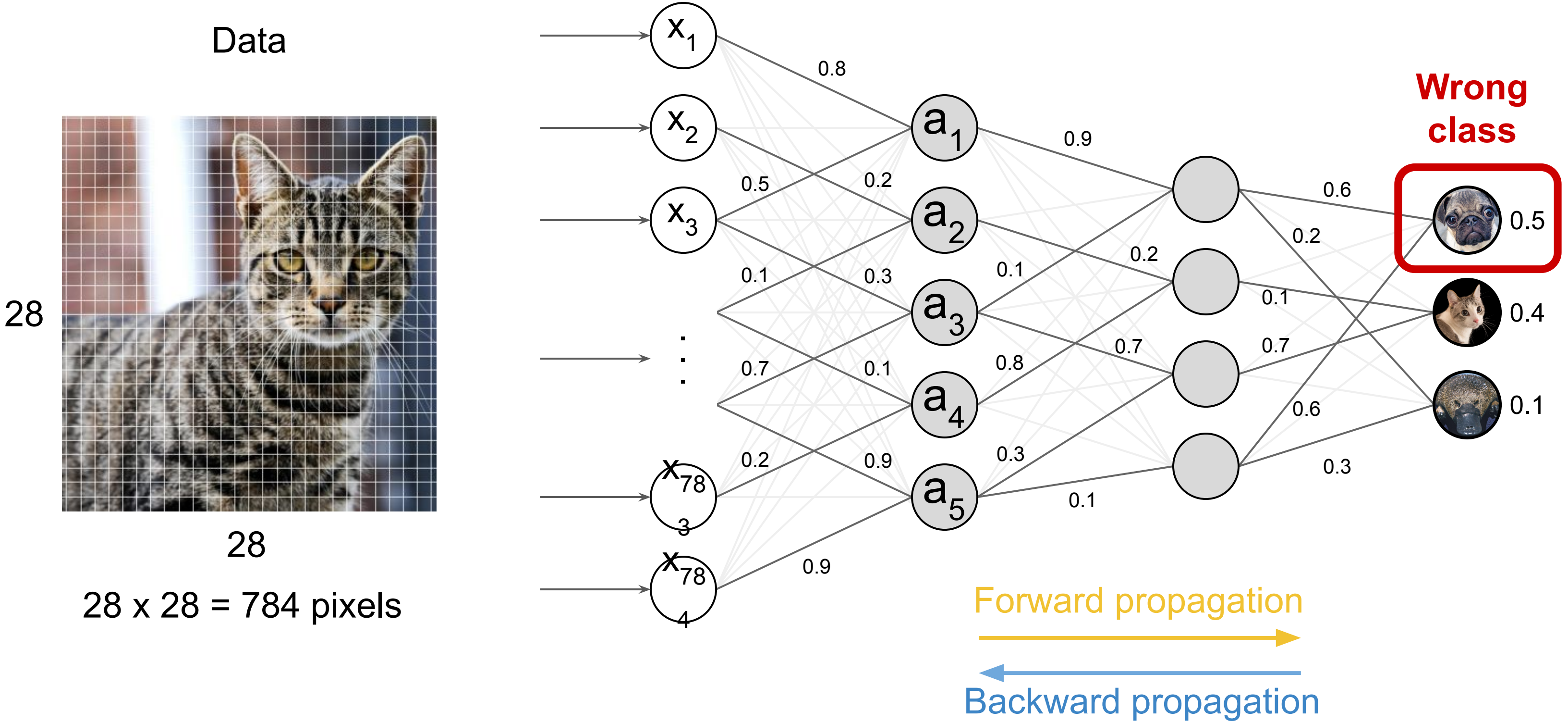
Forward propagation



Training the neural network

Backward propagation

e.g. $a1 = f(0.8\ x1 + 0.5\ x3 + \ldots + b1)$



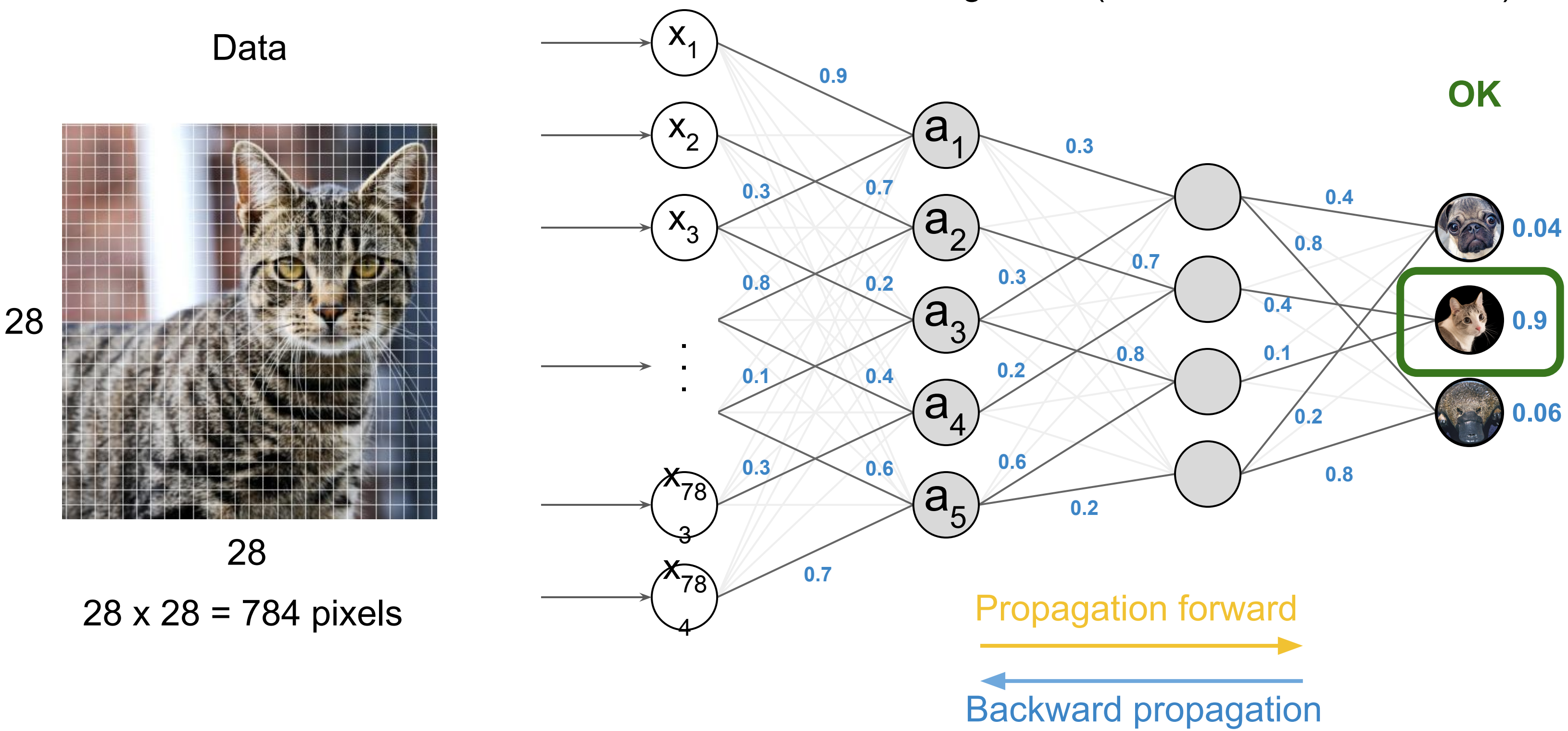
Truth	Error
0	-0.5
1	0.6
0	-0.1

Training the neural network

Convergence ?

$$a1 = f(0.9\ x1 + 0.3\ x3 + \ldots + b1)$$

e.g. a1 = f(0.9 x1 + 0.3 x3 + … + b1)



Truth	Error
0	-0.04
1	0.1
0	-0.06

➢ We introduce the empirical loss over the entire dataset $\mathcal{D}$:

$$EmpLoss_{L,\mathcal{D}}(h_w) = \frac{1}{m}\sum_{(x,y)\in\mathcal{D}} L(y, h_w(x)).$$

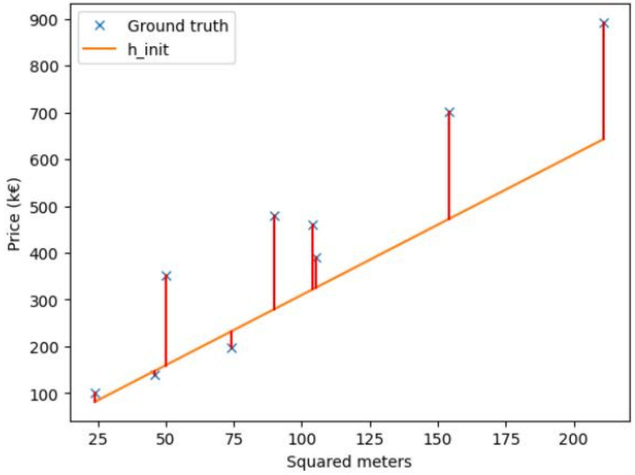
➢ For an example (x, y) and predictor h_w, we can use the loss functions :

- L_1-loss : $L_1(y, \hat{y}) = |y - h_w(x)|$,
- L_2-loss : $L_2(y, \hat{y}) = (y - h_w(x))^2$

To optimize the perceptron, we solve :

$$\hat{w}^* = \arg\min_w Loss(w).$$

$\Longrightarrow$ using L2-loss :
Perceptron is equivalent to linear regression !



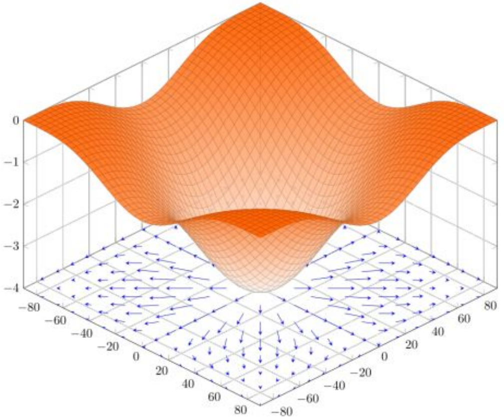
Supervised learning

Task

Algorithm Gradient descent algorithm

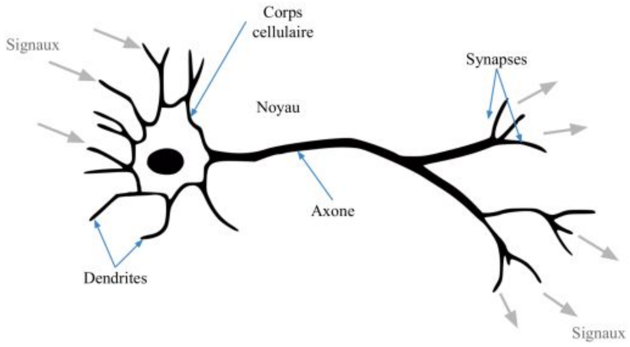
Dataset $\mathcal{D}$: inputs $X \rightarrow$ outputs y
Initialize weights w_i
while not converged **do**
Compute prediction $h_w(x)$ and loss $Loss(w)$
Update weights with step size α :

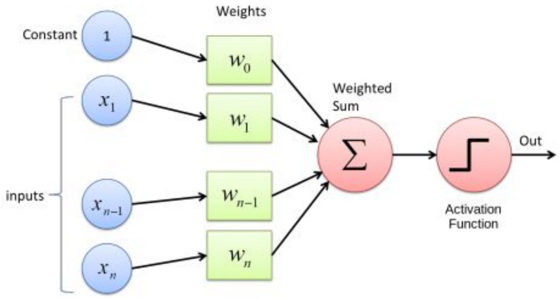
$$w \leftarrow w - \alpha \times \vec{\nabla} Loss(w)$$



$$\vec{\nabla} Loss(w) = \begin{bmatrix} \frac{\partial}{\partial w_0} Loss(w) \\ \frac{\partial}{\partial w_1} Loss(w) \\ \vdots \\ \frac{\partial}{\partial w_m} Loss(w) \end{bmatrix}$$

Neural networks
Perceptron





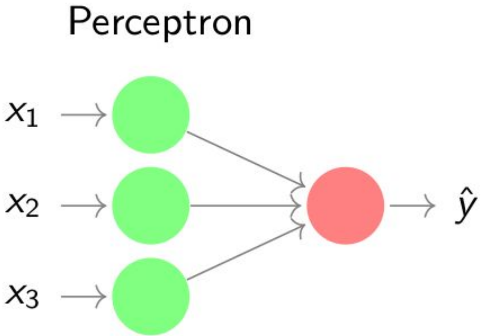
Given an **input** $x^T = \begin{bmatrix} x_1 & \cdots & x_n \end{bmatrix}$, we define a **perceptron** with the (synaptic) **weights** $w^T = \begin{bmatrix} w_1 & \cdots & w_n \end{bmatrix}$ and bias w_0 to compute the **output** $h_w(x)$ as

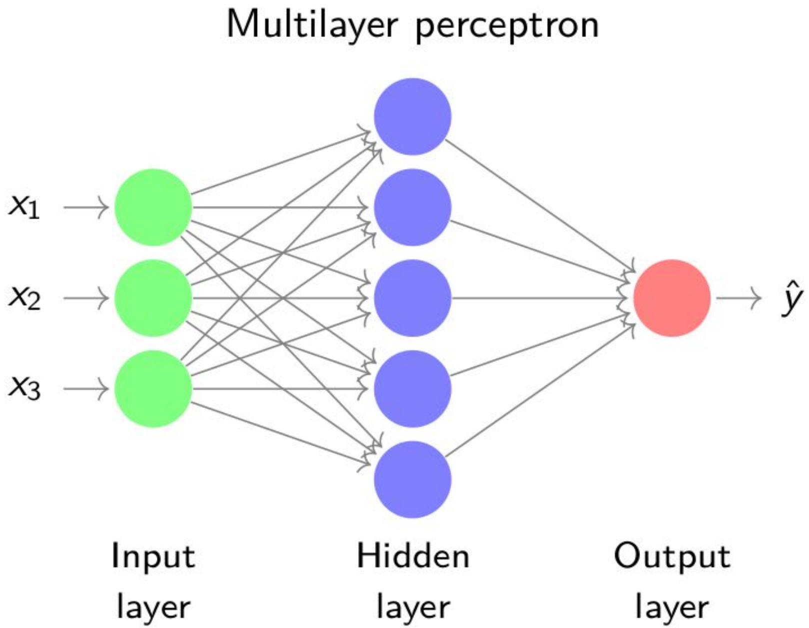
$$h_w(x) = g(w_0 + \sum_{i=1}^{n} w_i x_i) \tag{1}$$

Hypothesis space : linear functions, Loss $L2$-loss (e.g.)
Training : gradient descent updates $w \leftarrow w - \alpha \times \vec{\nabla} Loss(w)$

Neural networks

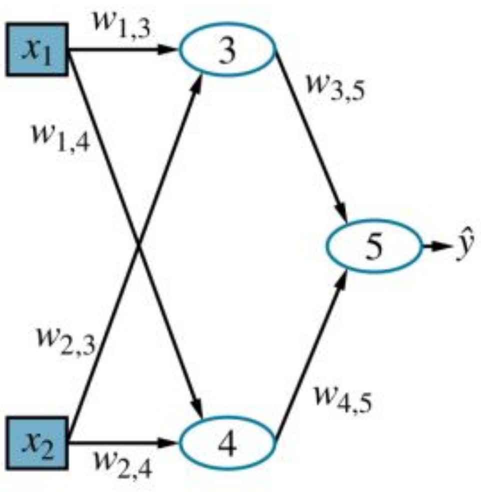
From 1 neuron to a brain

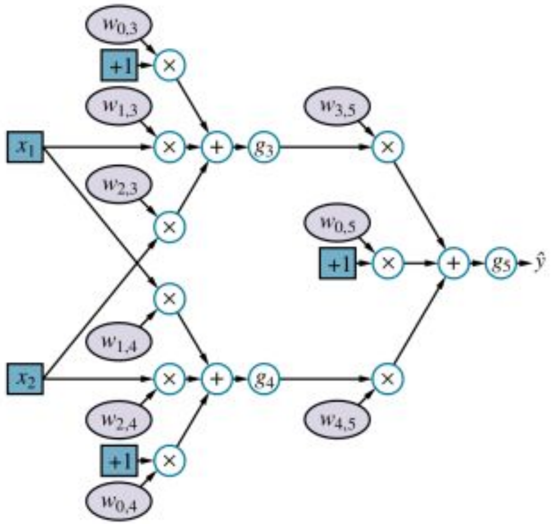




Neural networks

From 1 neuron to a brain: the chain rule





Neural networks
Full training

Network $\leftarrow$ neural network with initial weights
while not converged **do**
  $\textsc{Backprop-Iter}(E, \text{Network})$

Problem :

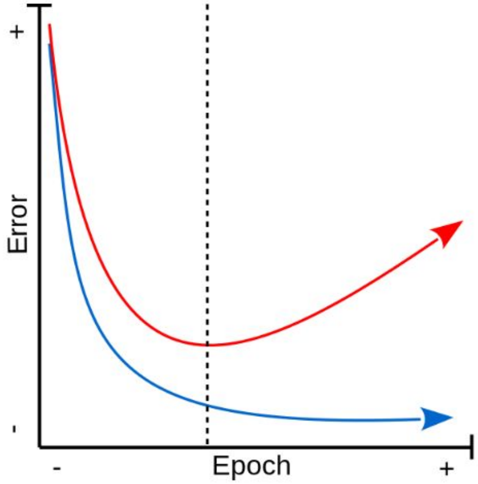
- slow, requires the derivatives
- gradient computation is costly and increases with
  - number of weight
  - number of examples

$\Longrightarrow \quad O(|w| \times |E|)$

Solution : *(Stochastic/mini-batch gradient descent)* :
select a small subset of example on which to propagate the error

Network $\leftarrow$ neural network with initial weights
while not converged **do**
  MiniBatch $\leftarrow$ *sample*(E, k)
  $\textsc{Backprop-Iter}(\textit{MiniBatch}, \text{Network})$

Neural networks
Convergence



Error on training set (blue) and test set (red)

Problem :

- training tend to overfit the data
- we cannot touch the test data

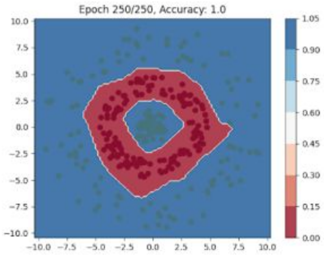
Solution :

- stop when performance decreases on the validation set,
- do not use validation set for training !

Neural networks
In practice

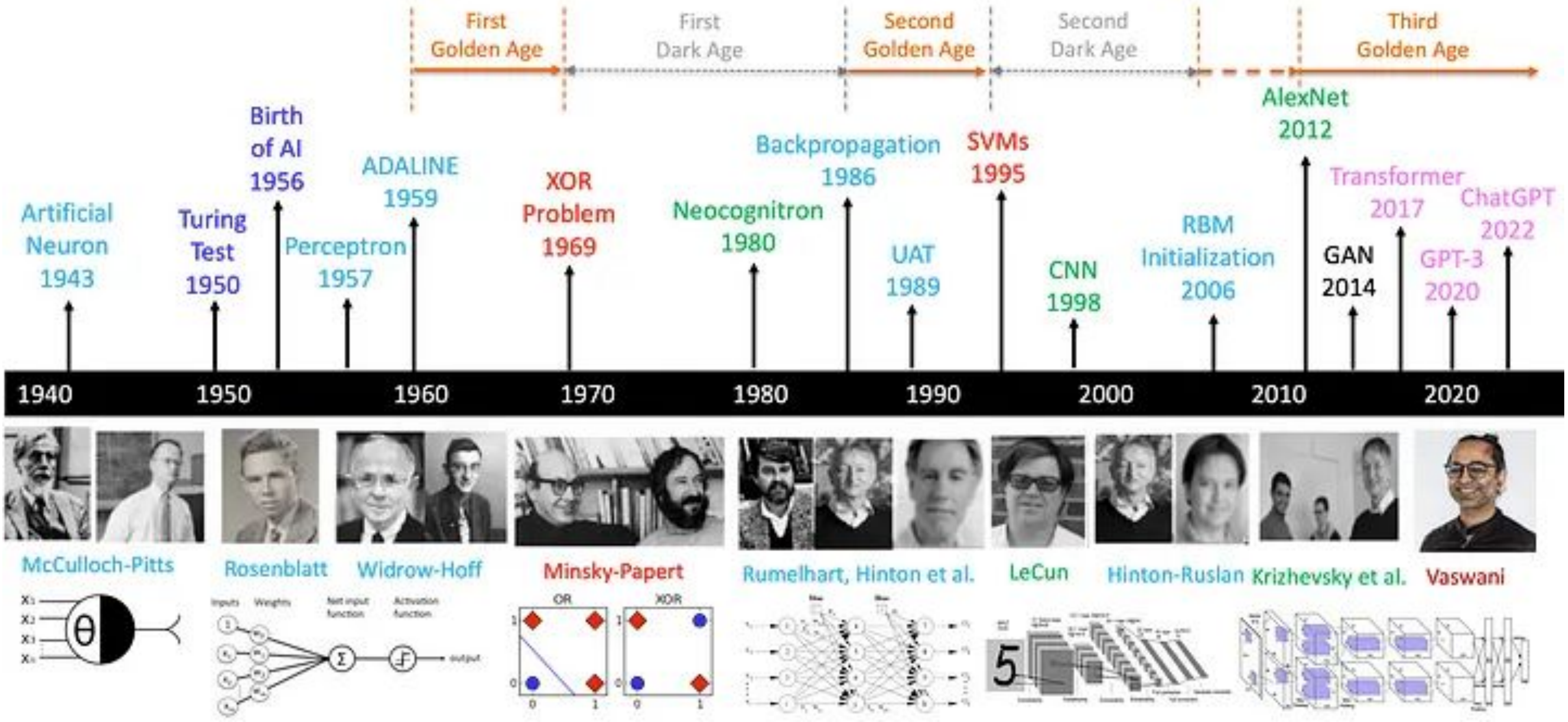
Use existing libraries ! Also contains all elements to develop new machine learning methods (used in research) :

- Scikit-learn
- Keras + Tensorflow




```python
# Création du modèle de réseau de neurones
model = tf.keras.Sequential([
    tf.keras.layers.Dense(8, activation='relu', input_shape=(2,)),
    tf.keras.layers.Dense(8, activation='relu'),
    tf.keras.layers.Dense(2, activation='softmax')
])
# Compilation du modèle
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
# Entraînement sur data avec labels
model.fit(data, labels, epochs=250, verbose=0)
# Prédiction sur data_test
predicted_labels = model.predict(data_test)
```

A brief history of AI with deep learning



Convolutional Neural networks

Image analysis

Is there a left turn in the following images?

$$\begin{bmatrix}0 & 0 & 0\\0 & 1 & 1\\0 & 1 & 0\end{bmatrix} \begin{bmatrix}0 & 0 & 0\\0 & 1 & 1\\0 & 1 & 1\end{bmatrix} \begin{bmatrix}0 & 1 & 0\\0 & 1 & 1\\0 & 1 & 0\end{bmatrix} \begin{bmatrix}0 & 0 & 0\\1 & 1 & 0\\0 & 1 & 0\end{bmatrix}$$

Convolutional Neural networks

Convolution Kernel

$$input = \begin{bmatrix} x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 \\ x_7 & x_8 & x_9 \end{bmatrix}$$

$$kernel = \begin{bmatrix} w_1 & w_2 & w_3 \\ w_4 & w_5 & w_6 \\ w_7 & w_8 & w_9 \end{bmatrix}$$

$$f_w(x) = \sum_i w_i x_i$$

$$kernel = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 1 & 1 \\ -1 & 1 & -1 \end{bmatrix}$$

$$f_w(\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}) = 3 \quad f_w(\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}) = 2 \quad f_w(\begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}) = 1 \quad f_w(\begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}) = 2$$

- When $f_w(x) = 3$ our kernel is able to detect a "right turn" in a 3x3 image. [4]
- Our kernel is essentially a neural unit (perceptron).
- The weights could be learned

Convolutional Neural networks

Scaling up to 4

$$\begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix}$$

$$TL = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad TR = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

$$BL = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix} \quad BR = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

Key idea : apply the convolutional unit to each 3x3 sub-images.

$$\begin{bmatrix} f_w(TL) & f_w(TR) \\ f_w(BL) & f_w(BR) \end{bmatrix} = \begin{bmatrix} 3 & -3 \\ -1 & -4 \end{bmatrix} = \begin{bmatrix} a_{17} & a_{18} \\ a_{19} & a_{20} \end{bmatrix}$$

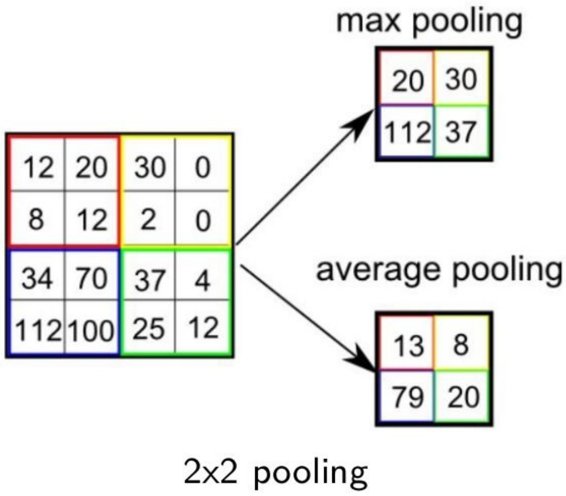
Interpretation : there is a “right turn” in the top left corner, the rest is garbage.

Key insight :

- in this convolutional layer, we have 4 (2x2) output nodes
- each uses the **same** function, with the **same weights**
- the kernel is trained to detect a feature independently of its location in the source image

Convolutional Neural networks

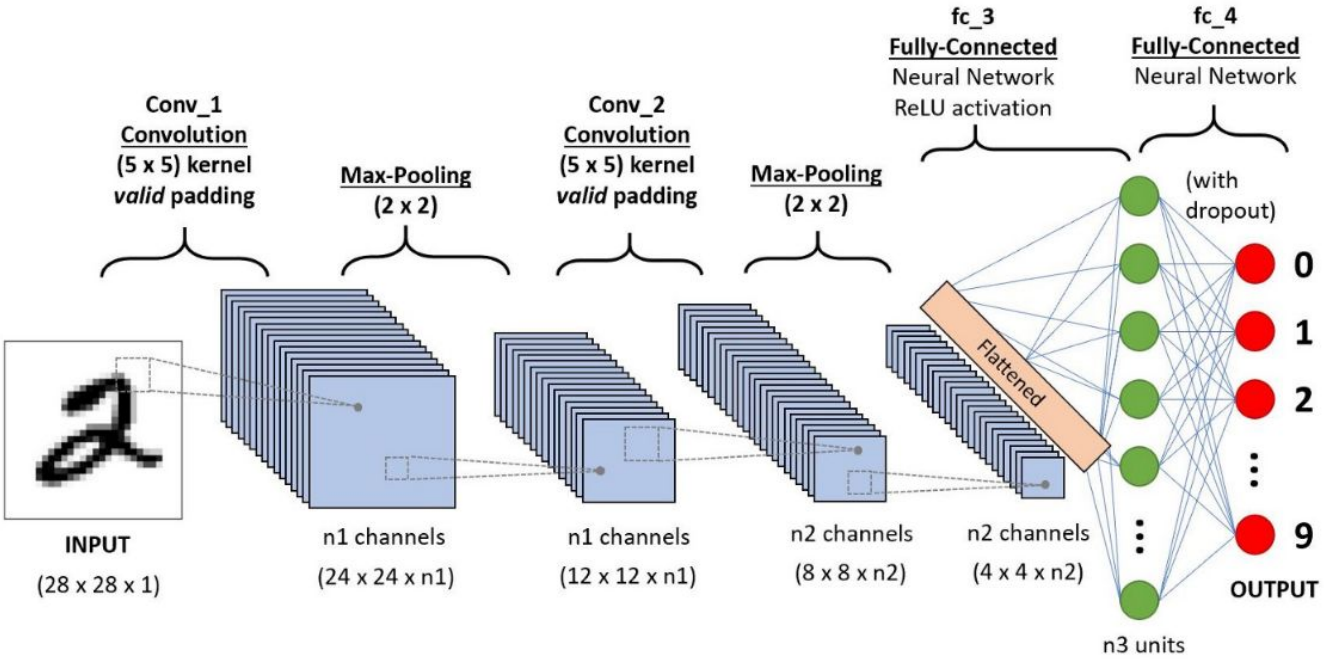
Combine with other types of kernels



- reduces dimensionality and variance
- suppresses the noise

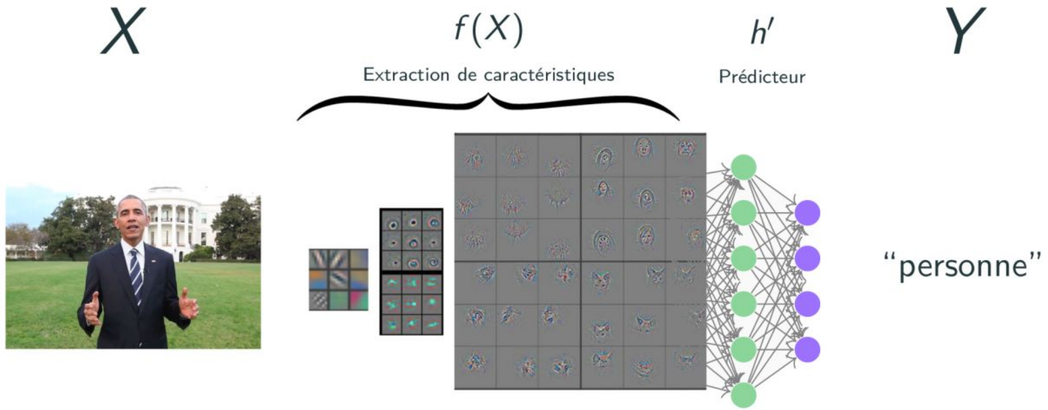
Convolutional Neural networks

Combine with other types of kernels



Convolutional Neural networks

Learns what to look at



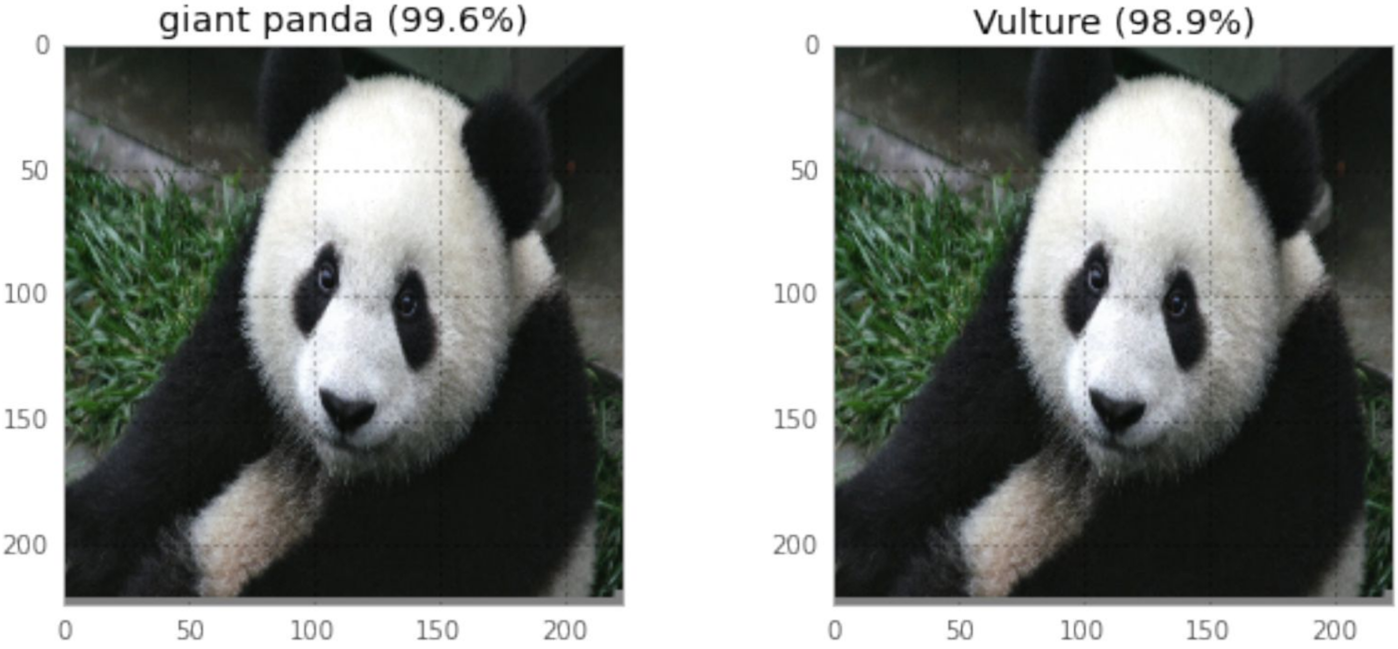
We can interpret CNN w.r.t. representation learning :

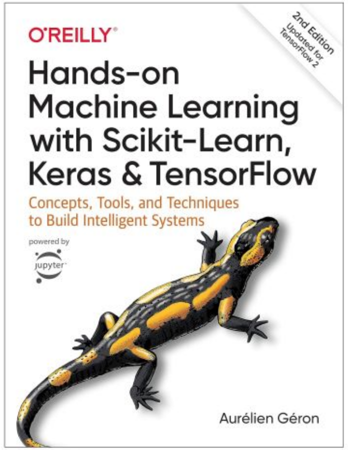
- the convolutional part is extractor of features/characteristics $f(X)$,
- the dense layers at the end play the role of our predictor h'.

Thus, deep learning allows to learn characteristics additionally to the predictor !

Some work left…

See you on wednesday





Got time a demo ?

Thank you for your attention