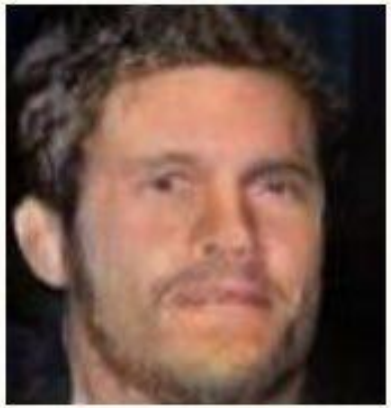
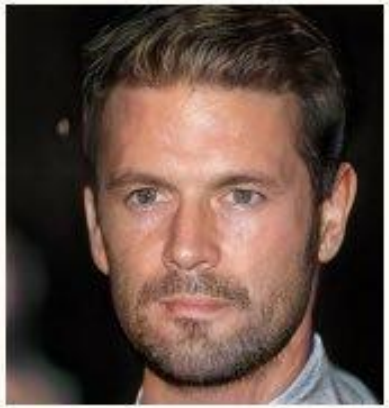

Apprentissage profond
Mila

2014

2015



2016



2017

GAN:

https://www.youtube.com/watch?v=G06dEcZ-QTg&feature=youtu.be

GAN style:

https://www.youtube.com/watch?time_continue=107&v=kSLJriaOumA

Style transfer:

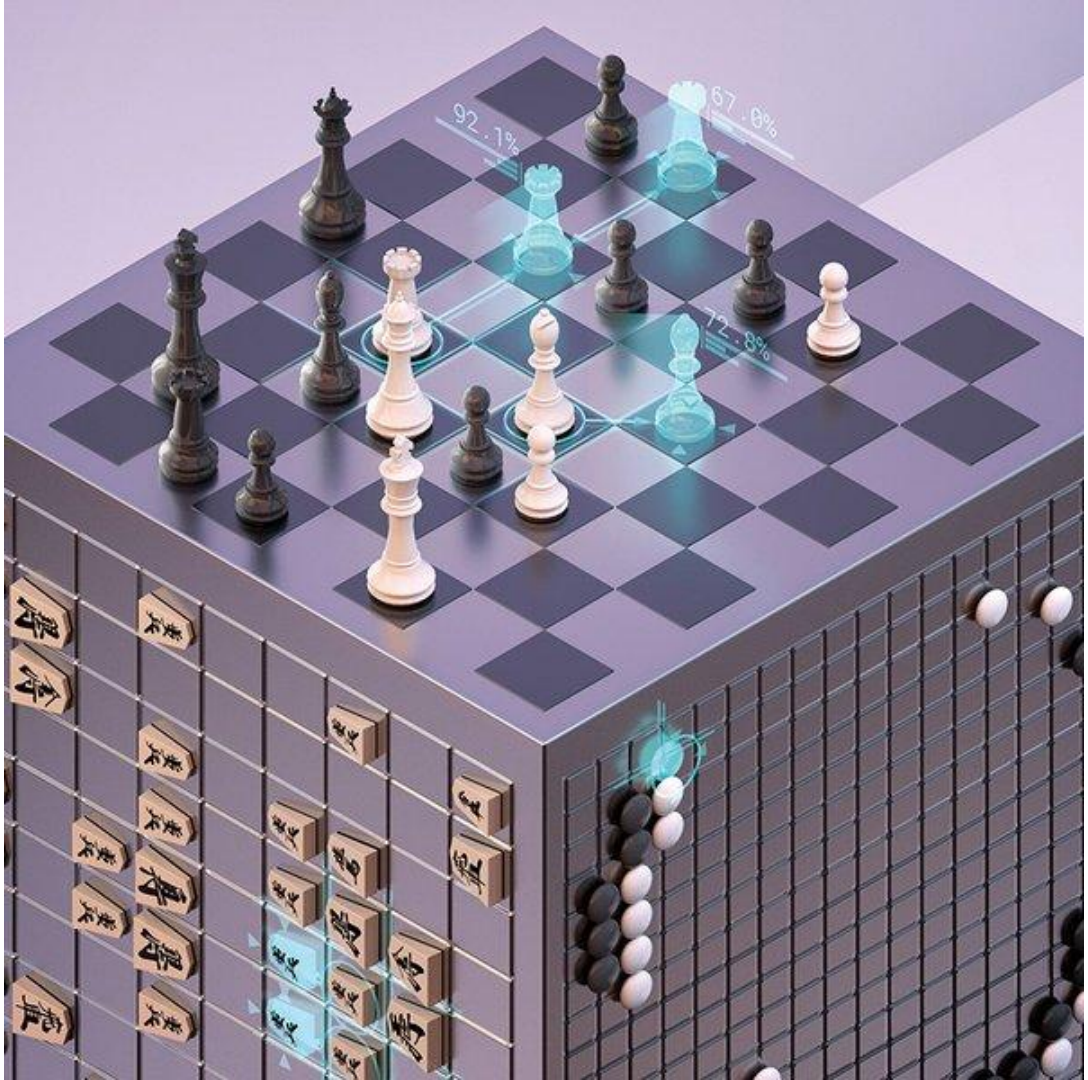
https://www.youtube.com/watch?v=Khuj4ASldmU

Face2Face:

https://www.youtube.com/watch?v=ohmajJTcpNk

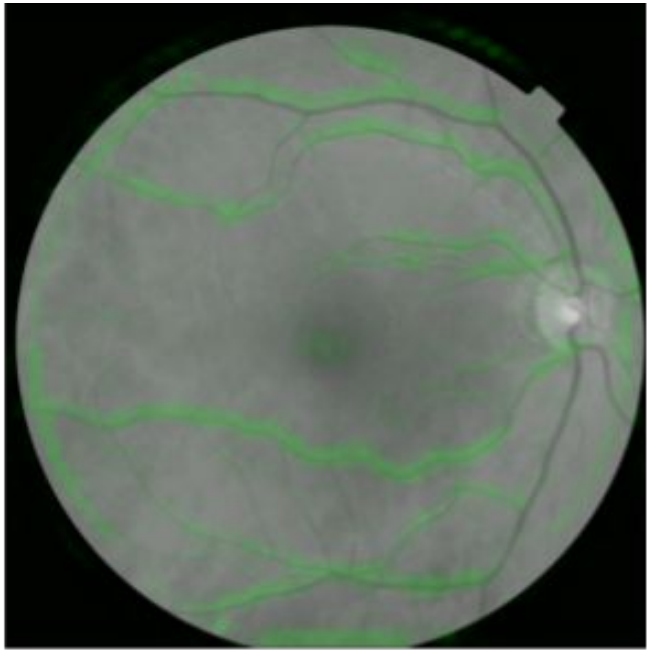
Deepfake (film *The shining*):

https://www.youtube.com/watch?v=Dx59bskG8dc


92.1%
67.8%
72.8%


TYPE
PLAYER
VISION
HEALTH

Image of retina



Blood pressure predictions
focus on blood vessels

https://www.youtube.com/watch?v=uiQ0oYZhGgQ

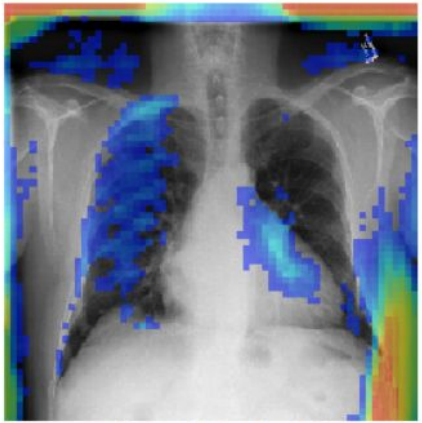
https://lyrebird.ai/

Chester

Input Image

Out Of Distribution reconstruction error

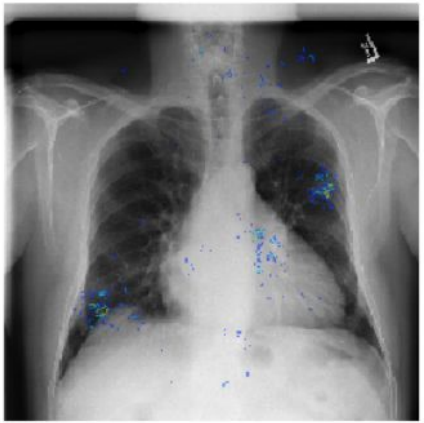
Heatmap where the image varies from the training distribution.



recScore:0.27, ssim:0.39

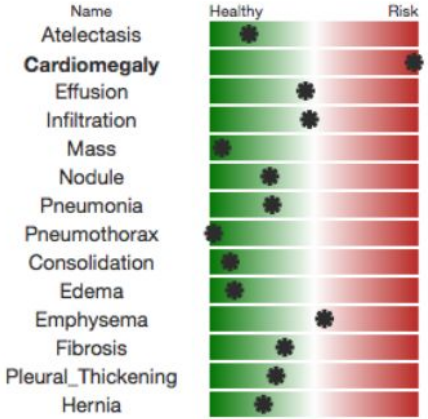
Predictive image regions

Heatmap of image regions which influence the prediction.

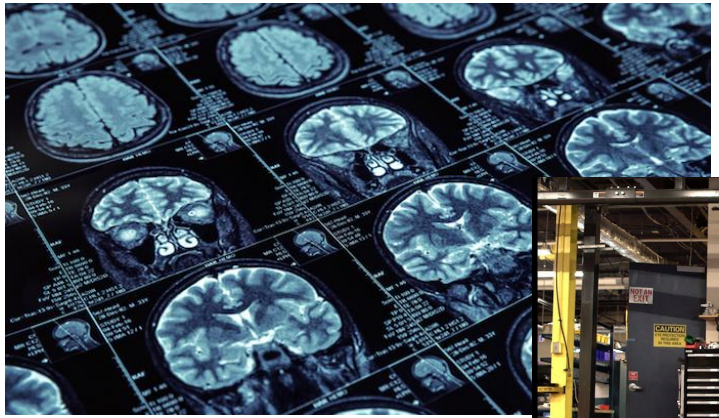


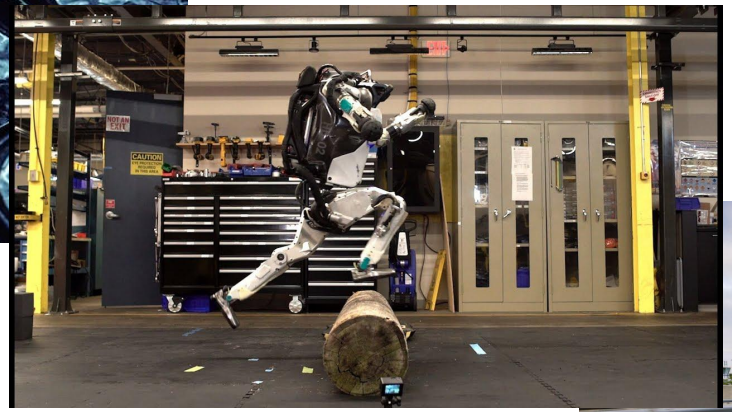
Disease Predictions

Probability of a disease.

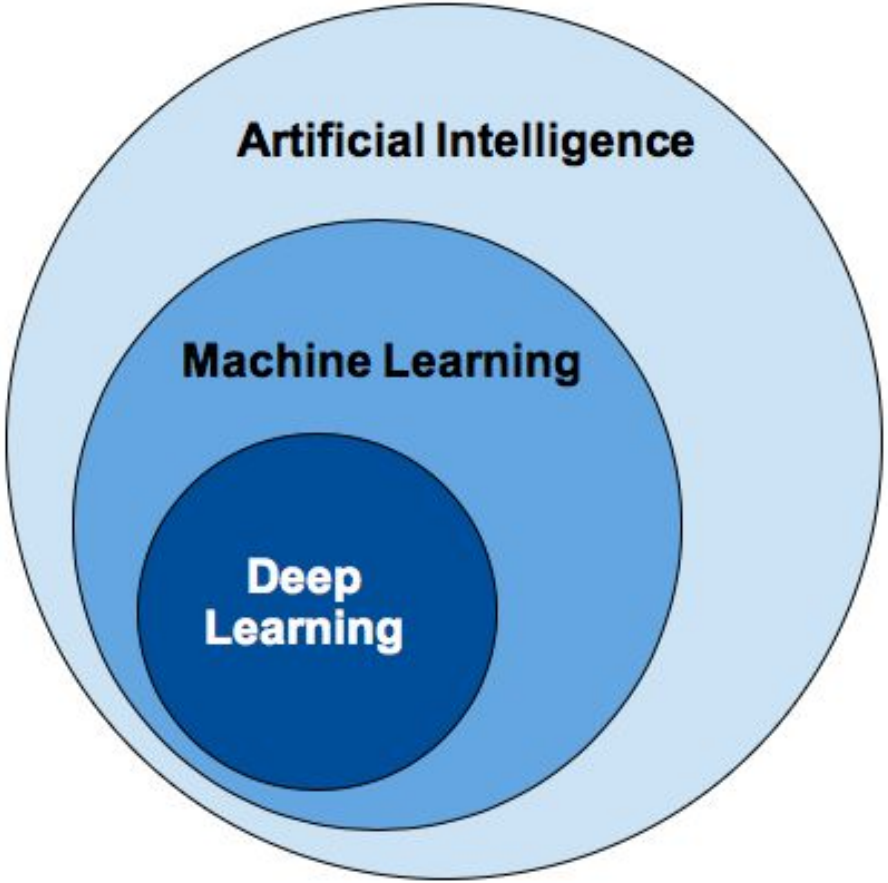


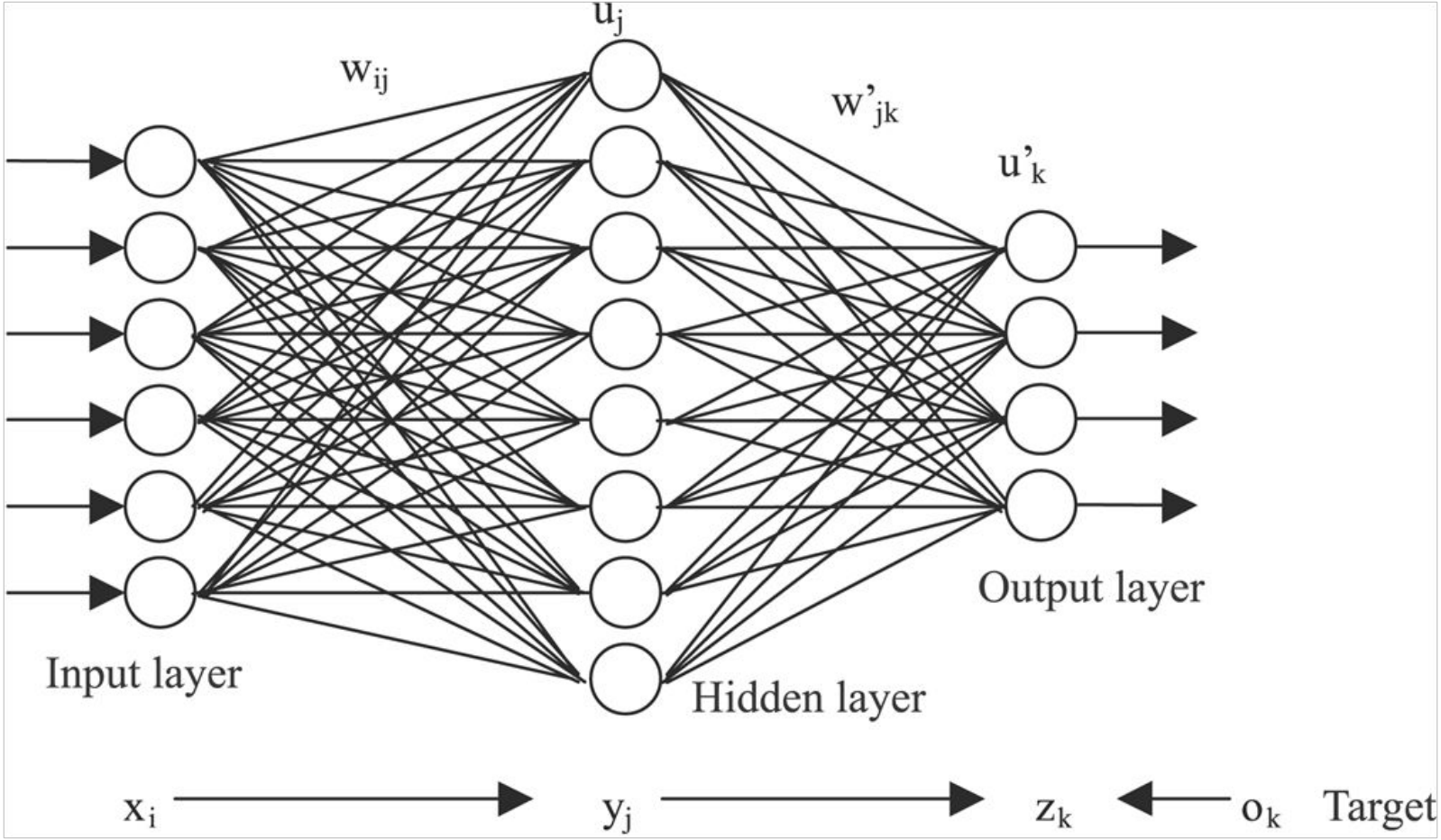
Et beaucoup d’autres domaines...

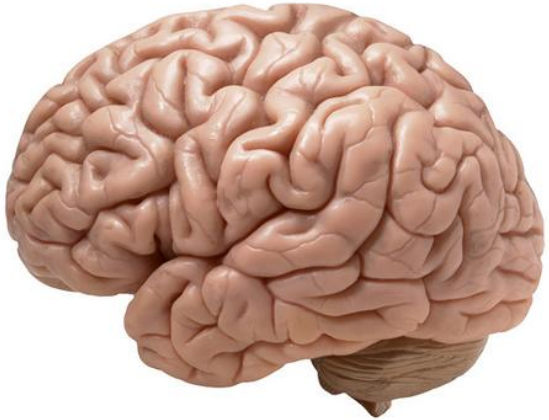





IA, machine learning,
deep learning

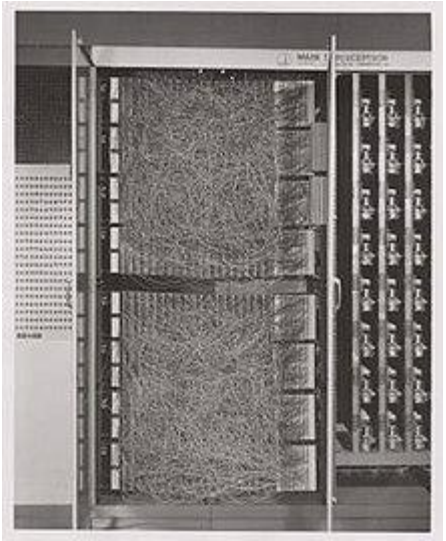

Artificial Intelligence
Machine Learning
Deep
Learning


u_j
w_{ij}
w'_{jk}
u'_k
Input layer
Hidden layer
Output layer
x_i
y_j
z_k
o_k
Target



Inspiration du cerveau:

- Massivement parallèle
- Plastique



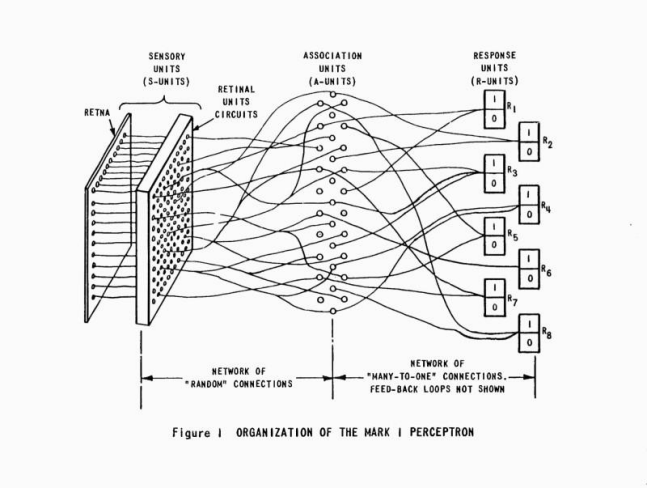
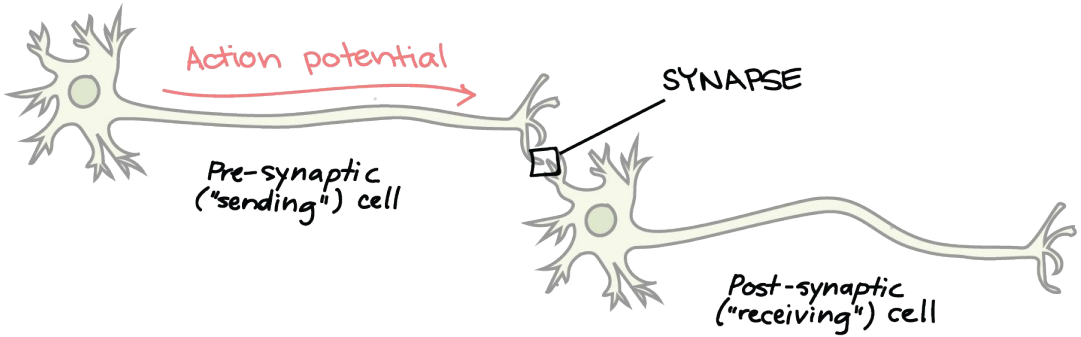
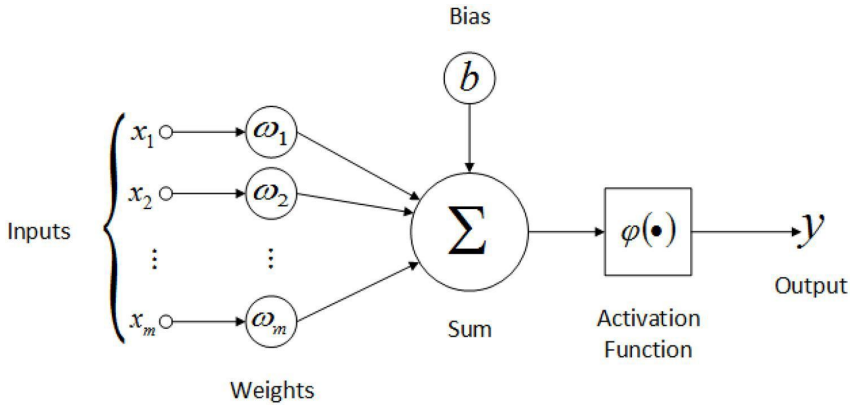
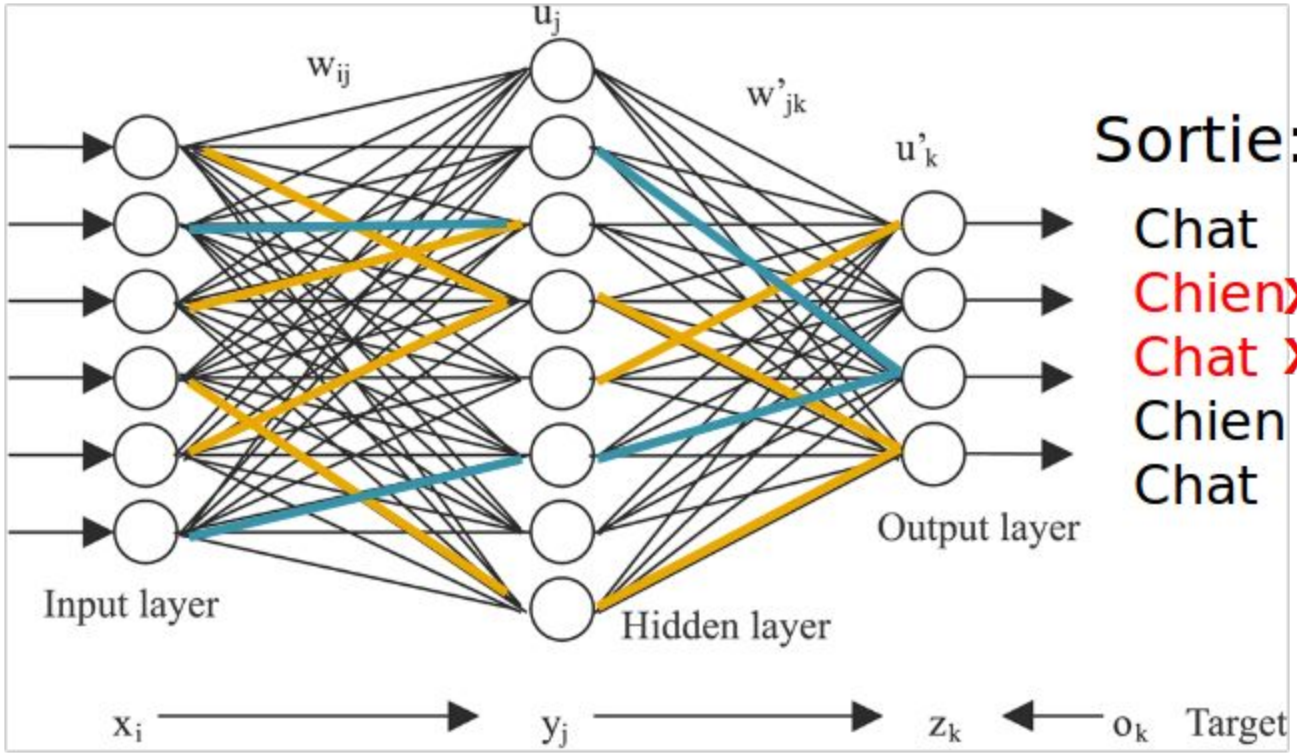


Figure 1 ORGANIZATION OF THE MARK 1 PERCEPTRON


Action potential
SYNAPSE
Pre-synaptic
("sending") cell
Post-synaptic
("receiving") cell


Bias
b
Inputs
x_1
x_2
x_m
ω_1
ω_2
ω_m
Σ
$\varphi(\bullet)$
y
Output
Sum
Activation
Function
Weights

Entrée:



Sortie:

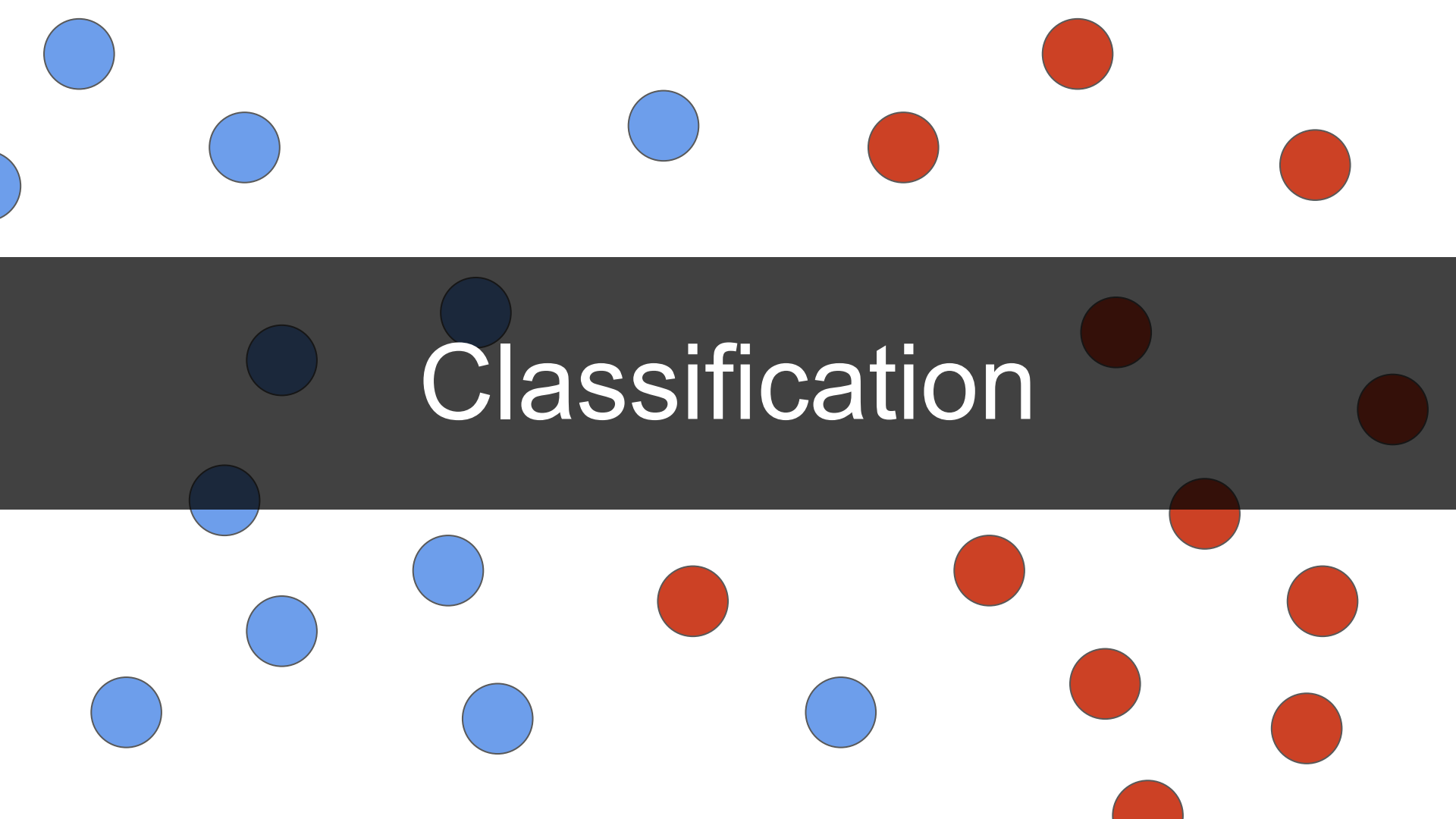
Chat

Chien X

Chat X

Chien

Chat


Classification

Étiquette (y)
(chien = 0, chat = 1)

Input (x)

1

1

0

0

1

Classifieur

Prédiction $(\hat{y})$
(chien = 0, chat = 1)

1

Input (x)

Classifieur

Prédiction

$$\hat{y} = f(x)$$

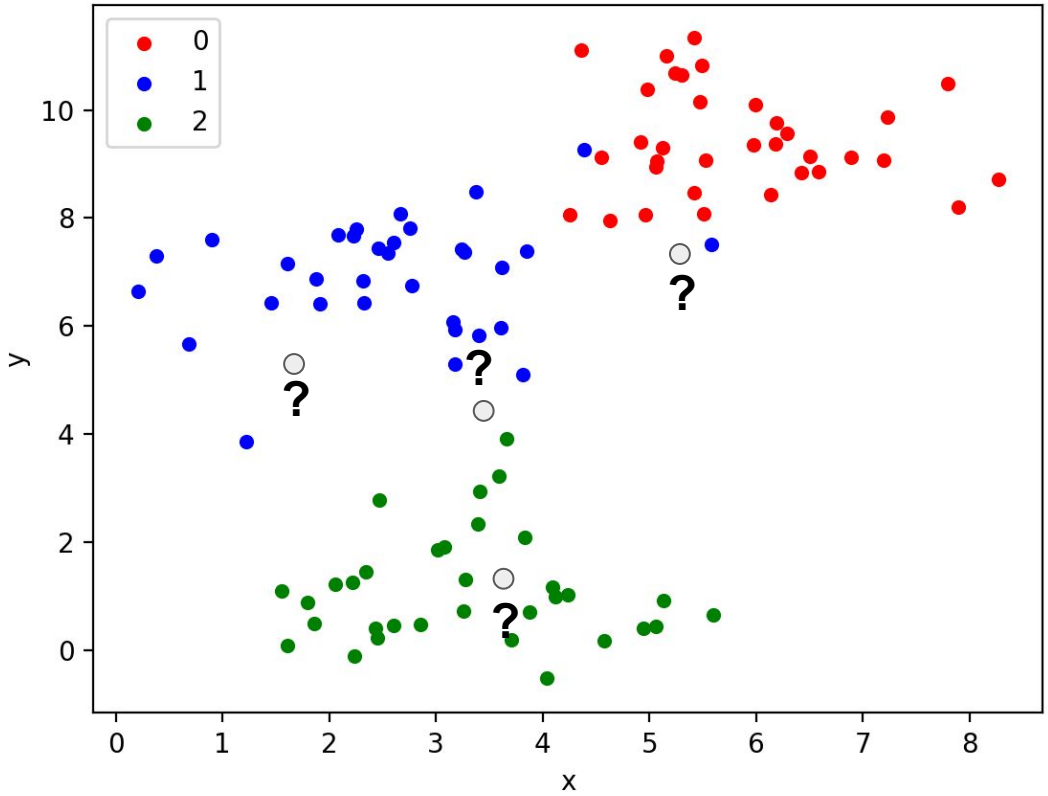
0

Étiquette (y)

1

Erreur

$$\mathcal{L}(\hat{y}, y) = (\hat{y} - y)^2$$


0
1
2
?
?
?
?
y
x
0
1
2
3
4
5
6
7
8
10

Plus proche moyenne

Chaque exemple est un vecteur (disons de 10 éléments):

$$x^{(i)} = \{x_1, x_2, x_3, \ldots, x_{10}\}$$

Pour chaque classe, on va calculer la moyenne, soit:

$$\frac{\sum_{i=1}^{n} x^{(i)}}{n}$$

Pour classer un nouvel exemple, on va calculer sa distance à la moyenne de chaque classe.

On calcule la moyenne par élément

$$
\begin{aligned}
x^{(1)} &= \{x_1, x_2, x_3, \ldots, x_{10}\} \\
x^{(2)} &= \{x_1, x_2, x_3, \ldots, x_{10}\} \\
x^{(3)} &= \{x_1, x_2, x_3, \ldots, x_{10}\} \\
&\vdots \qquad\qquad\quad \vdots \\
x^{(n)} &= \{x_1, x_2, x_3, \ldots, x_{10}\}
\end{aligned}
$$

Exemple de calcul

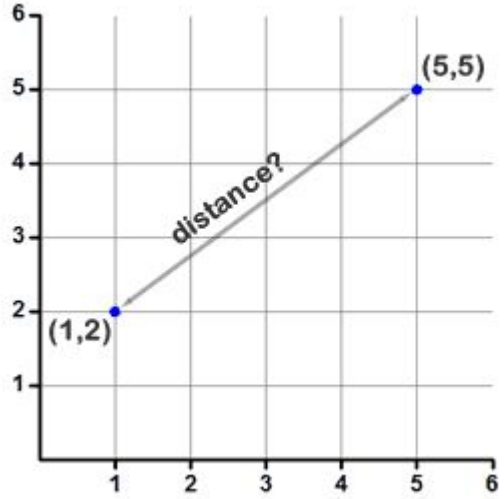
$$x^{(1)} = \{5, 8\}$$

$$x^{(2)} = \{7, 9\}$$

$$x^{(3)} = \{6, 10\}$$

$$MOY_{rouge} = \{6, 9\}$$

Distance



THE DISTANCE FORMULA

$$d = \sqrt{\left(x_2 - x_1\right)^2 + \left(y_2 - y_1\right)^2}$$

Exemple de calcul

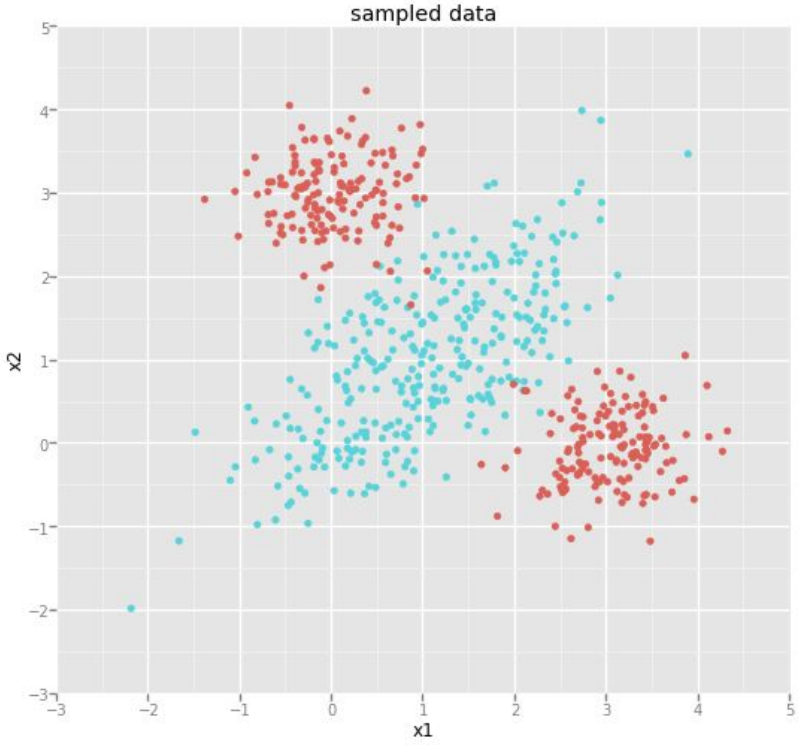
$$MOY_{rouge} = \{6, 9\}$$

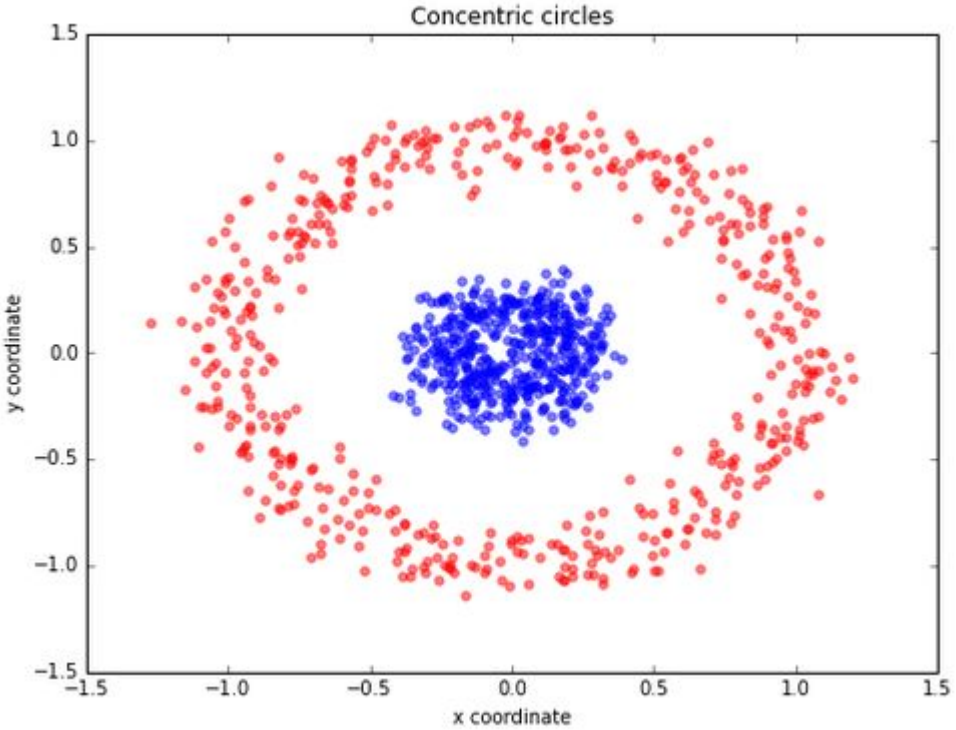
$$MOY_{bleu} = \{3, 7\}$$

$$MOY_{vert} = \{3, 1\}$$

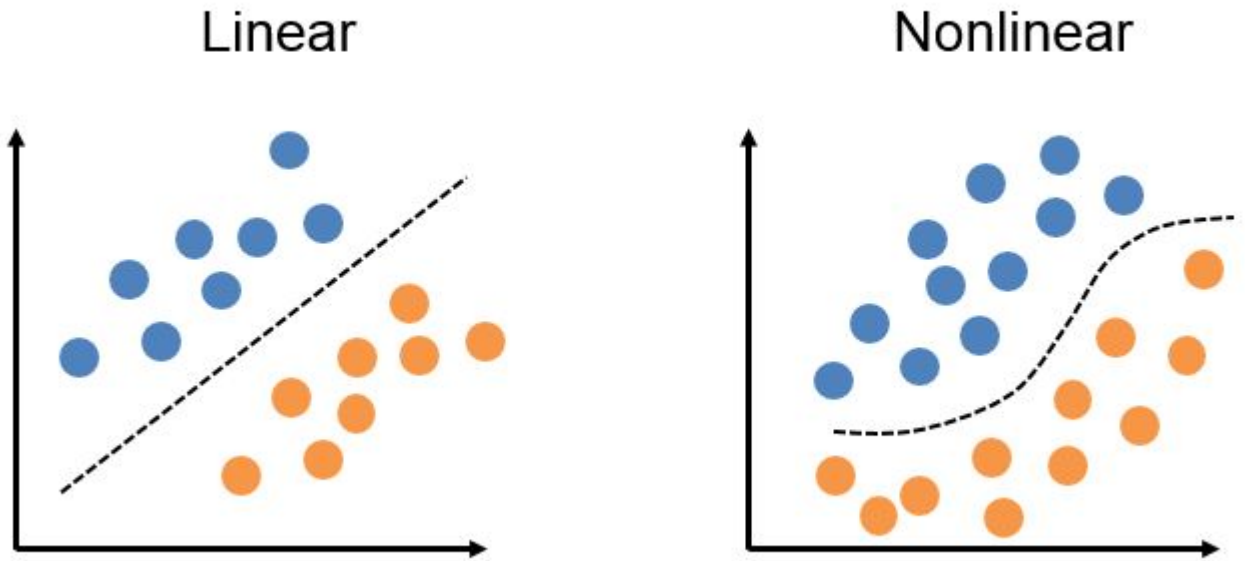
$$x = \{5, 2\}$$

Seulement linéaire...

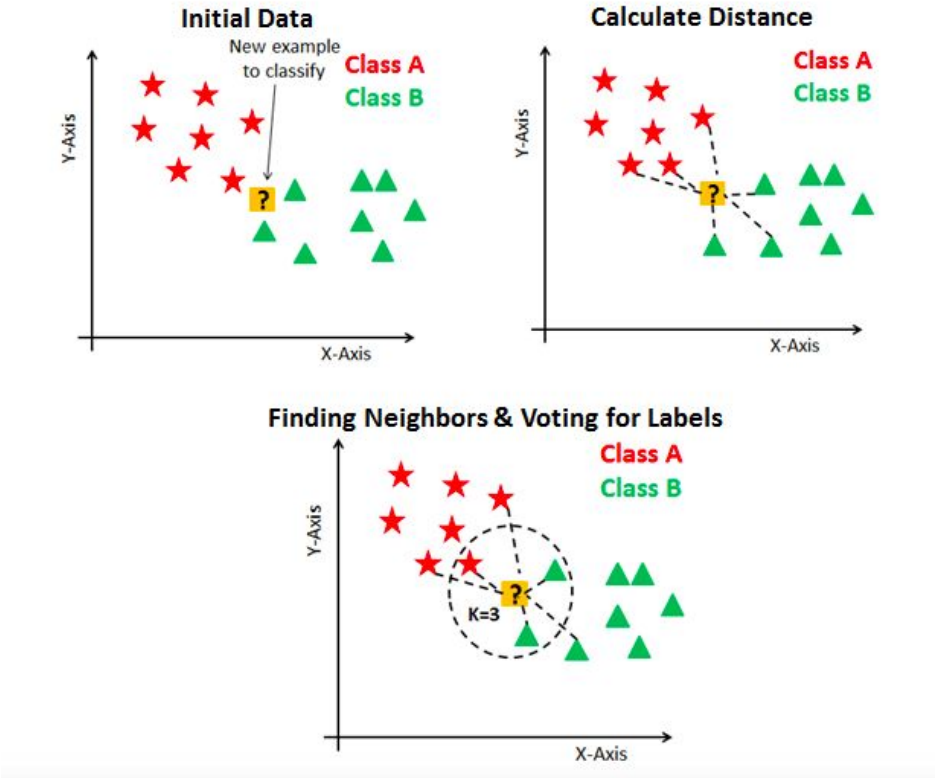


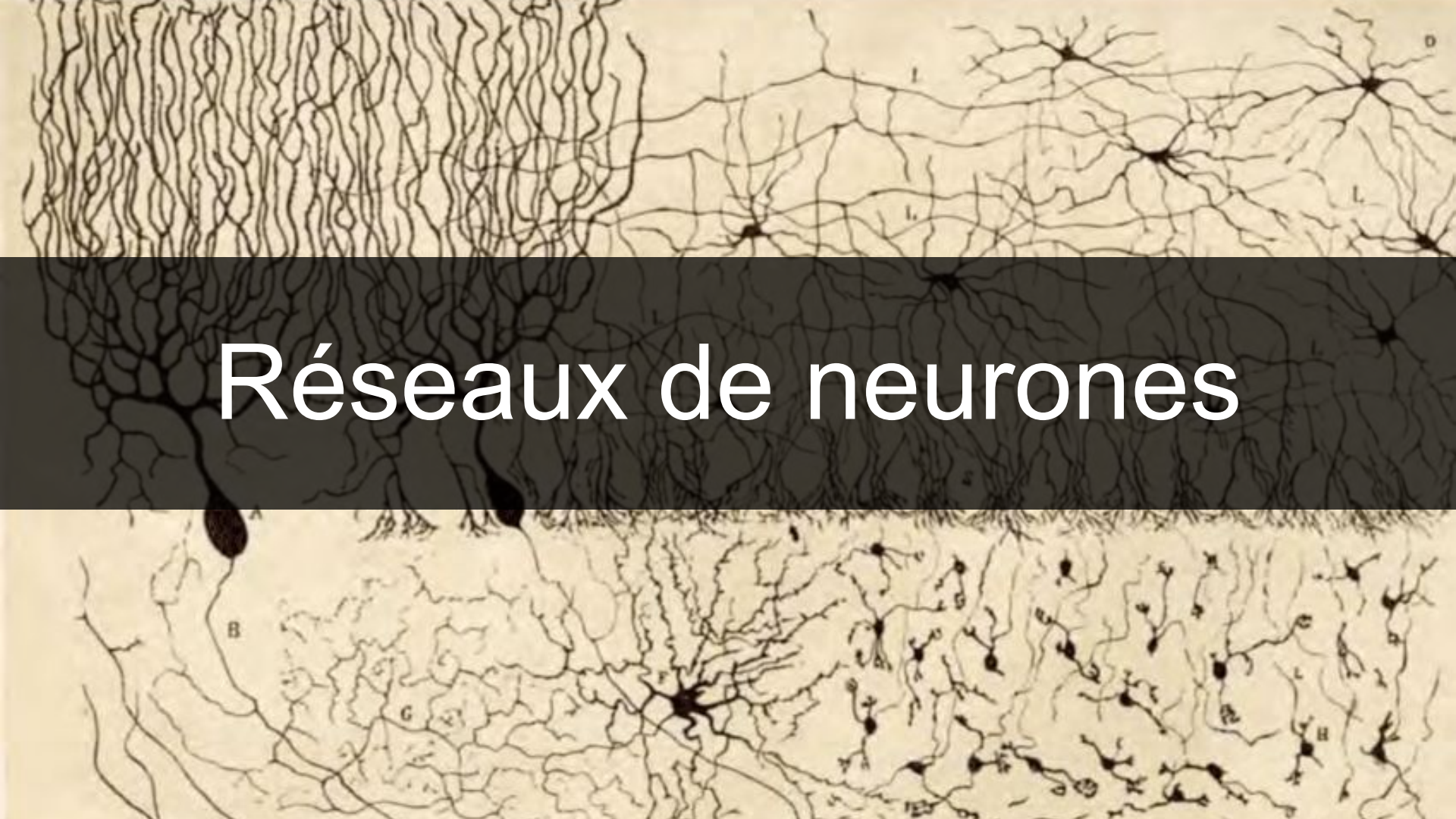


Seulement linéaire...



Autres algorithmes...



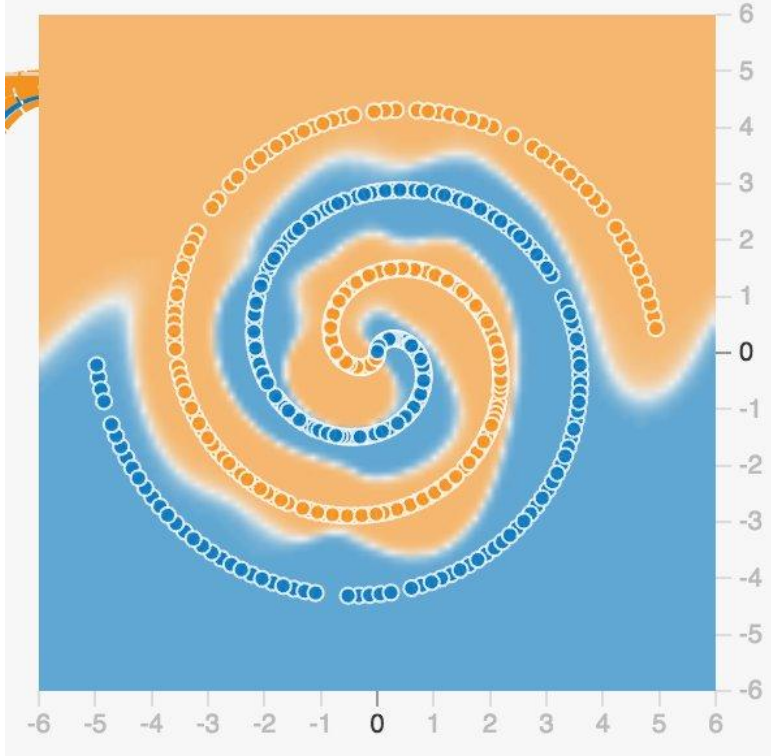


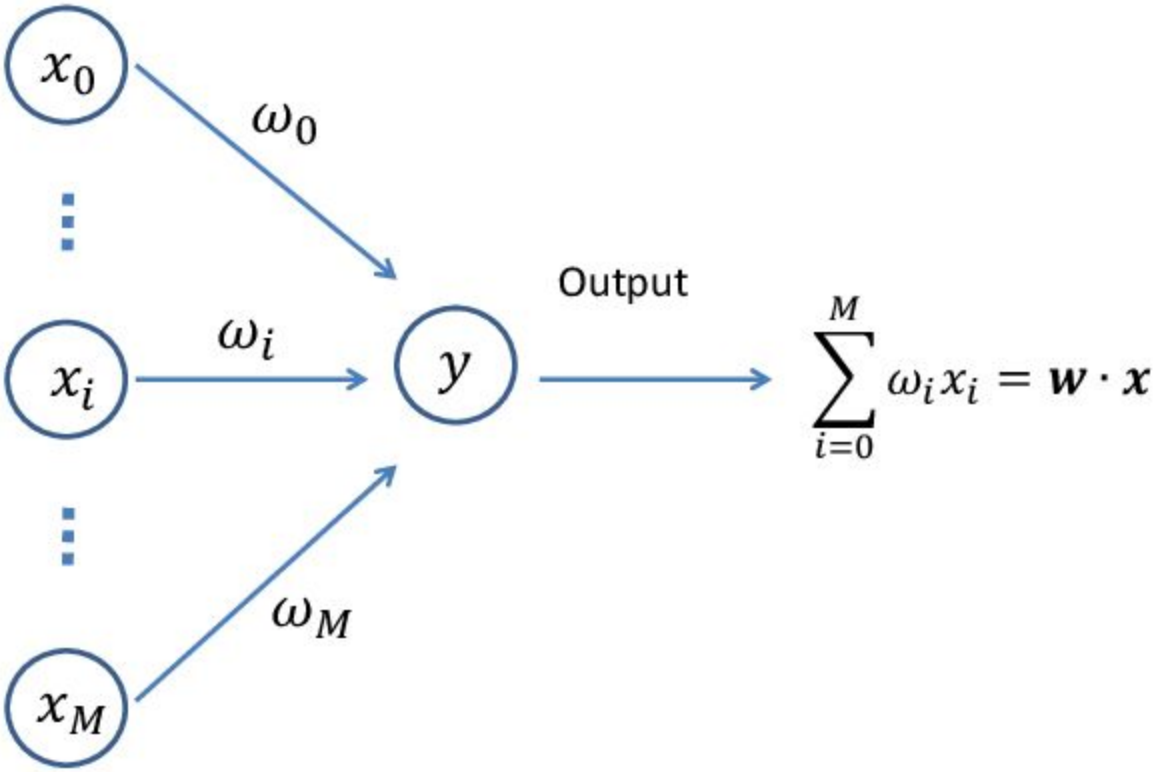
Réseaux de neurones

OUTPUT

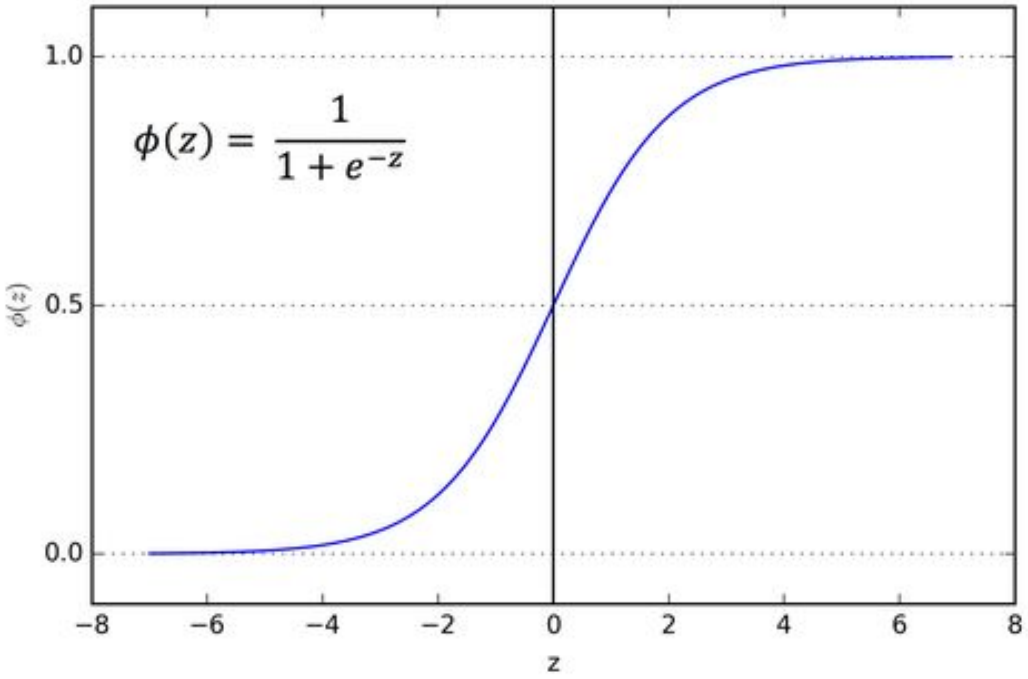
Test loss 0.003

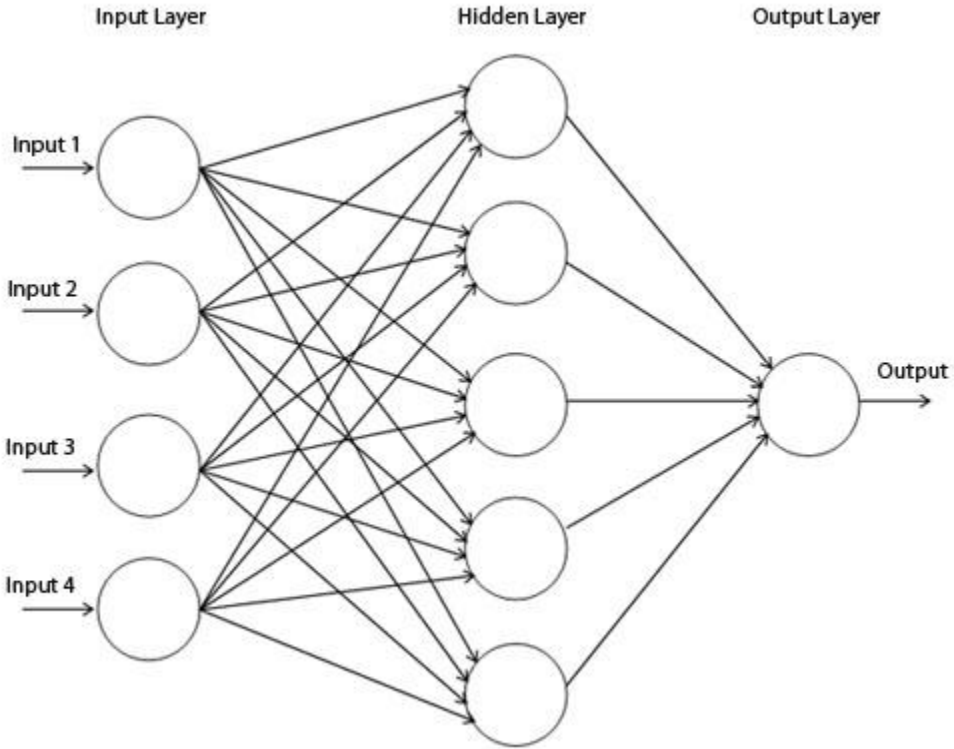
Training loss 0.004



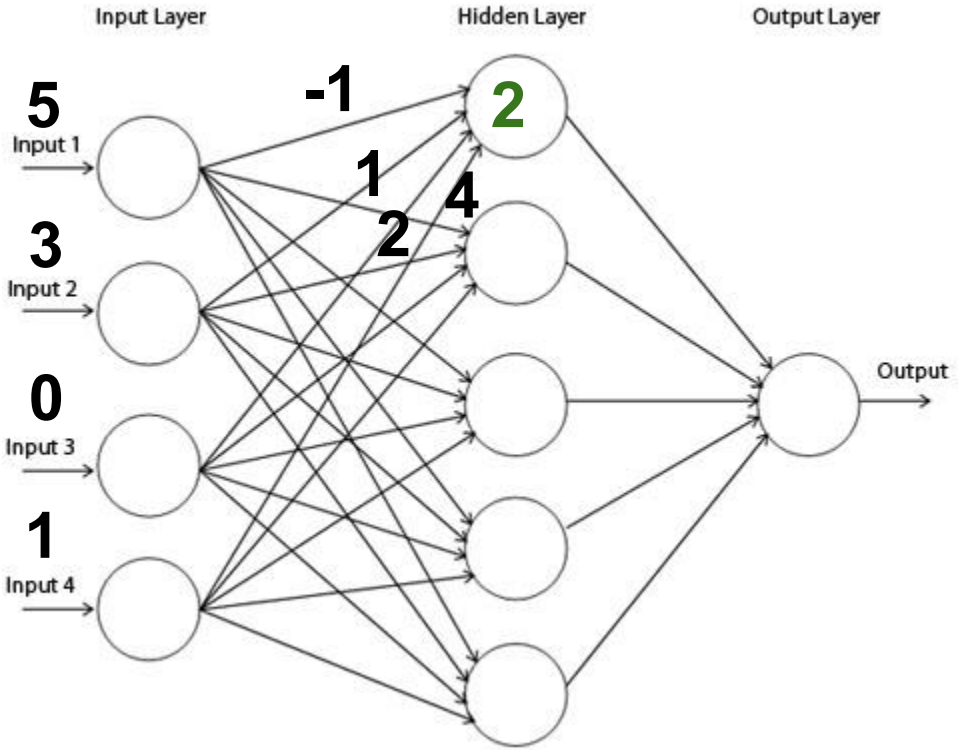

x_0
ω_0
x_i
ω_i
y
Output
$\sum_{i=0}^{M} \omega_i x_i = \boldsymbol{w} \cdot \boldsymbol{x}$
x_M
ω_M

Non-linéarité

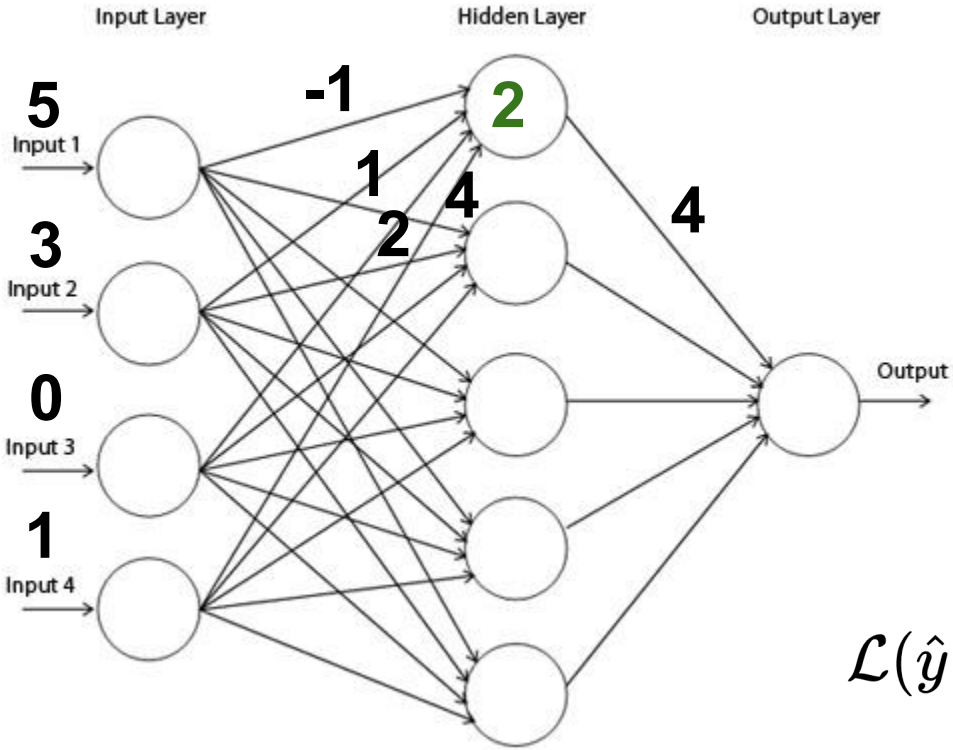



Input Layer
Hidden Layer
Output Layer
Input 1
Input 2
Input 3
Input 4
Output

Calcul


Input Layer
Hidden Layer
Output Layer
5
Input 1
3
Input 2
0
Input 3
1
Input 4
-1
1
2
4
2
Output

Entrainement: comment modifier les poids?



$y = 3$

$\hat{y} = 9$

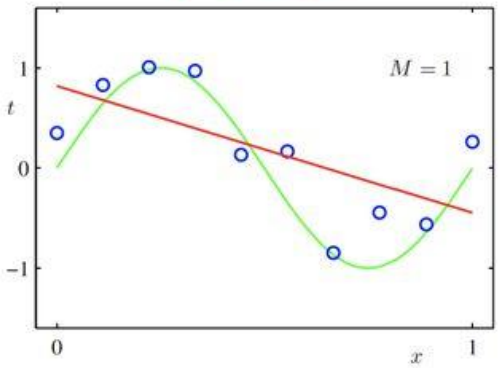
$$\mathcal{L}(\hat{y}, y) = (\hat{y} - y)^2$$

Tensorflow playground

https://playground.tensorflow.org

Overfitting

f is linear

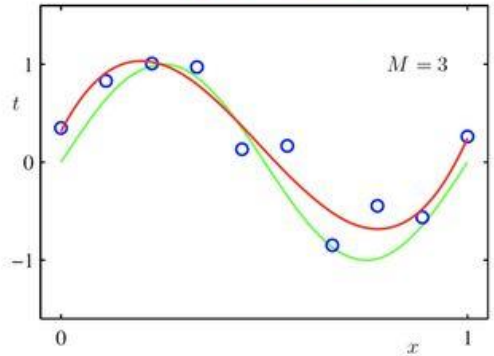


$Loss(w)$ is high

Underfitting

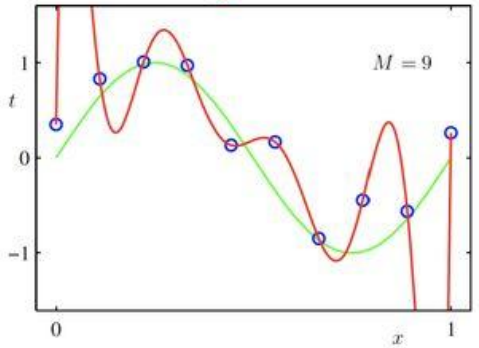
High Bias

f is cubic



$Loss(w)$ is low

f is a polynomial of degree 9

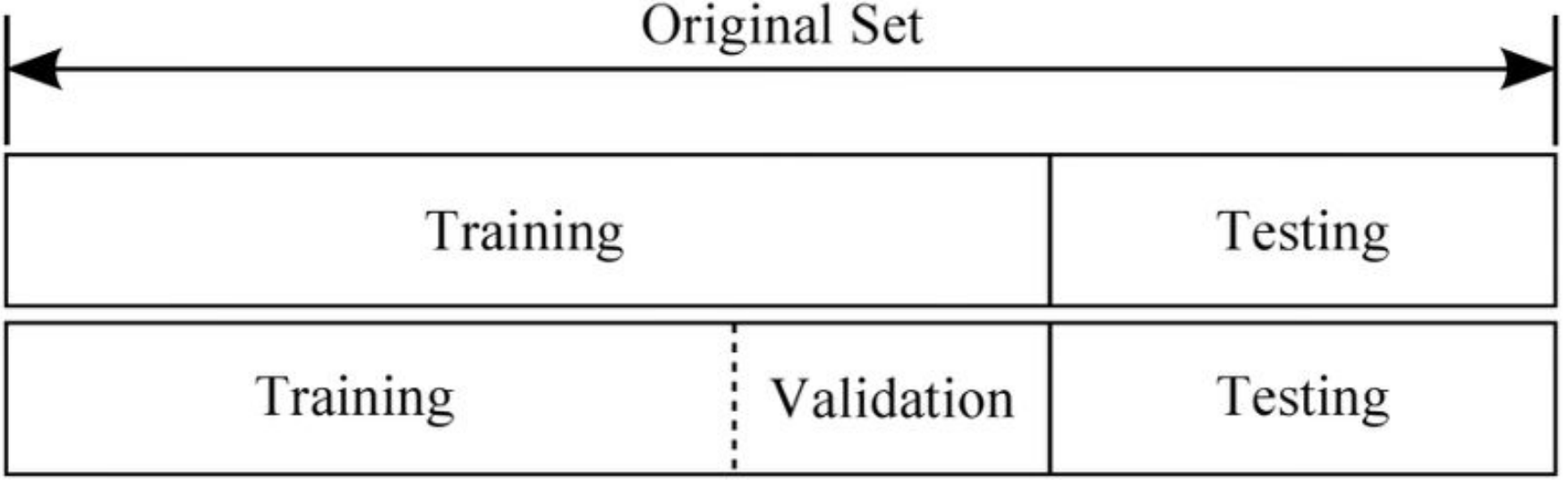


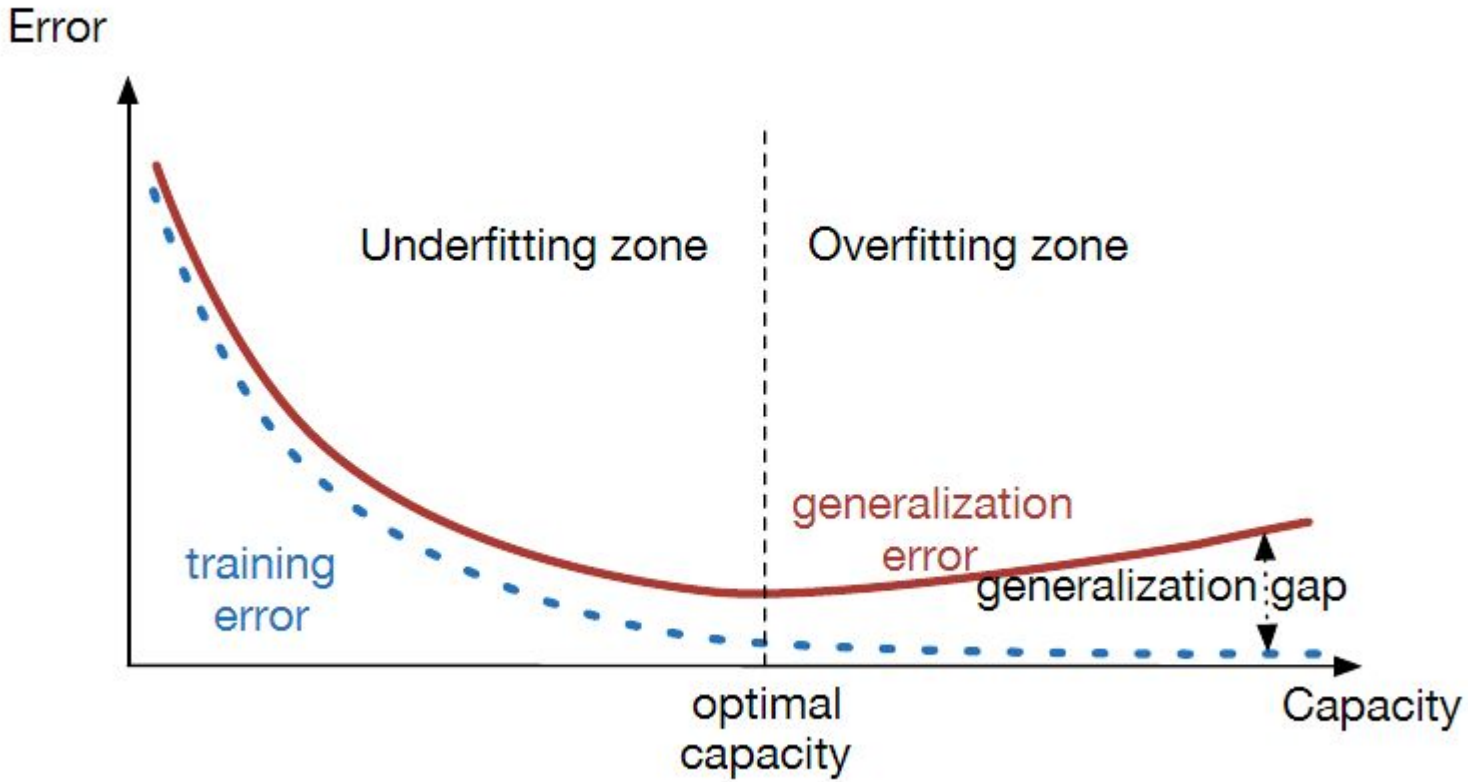
$Loss(w)$ is zero!

Overfitting

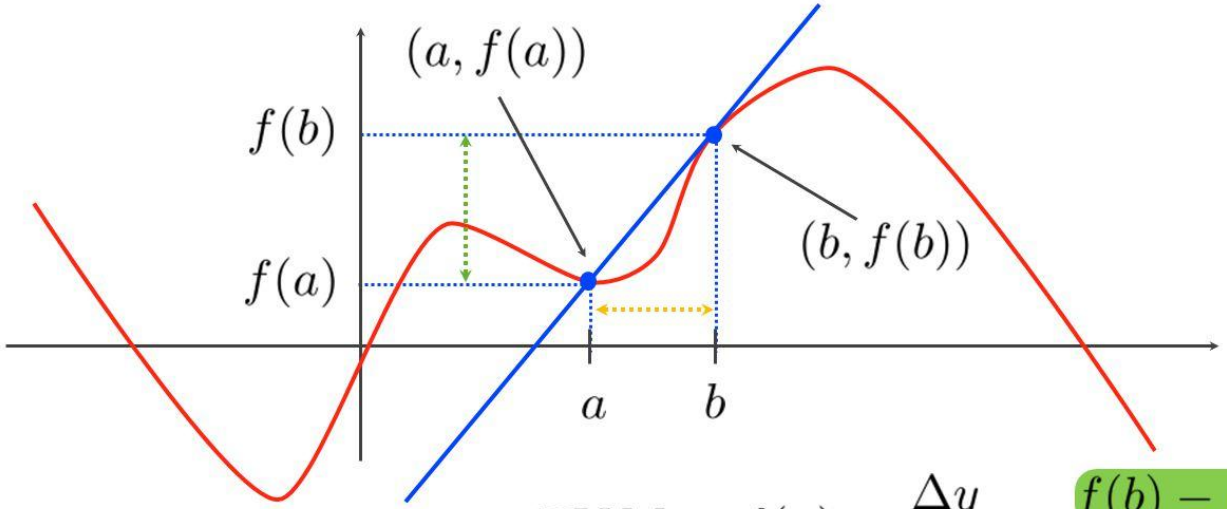
High Variance

Christopher M. Bishop – Pattern Recognition and Machine Learning


Original Set
Training
Testing
Training
Validation
Testing


Error
Underfitting zone
Overfitting zone
generalization
error
training
error
generalization gap
optimal
capacity
Capacity

Descente de gradient

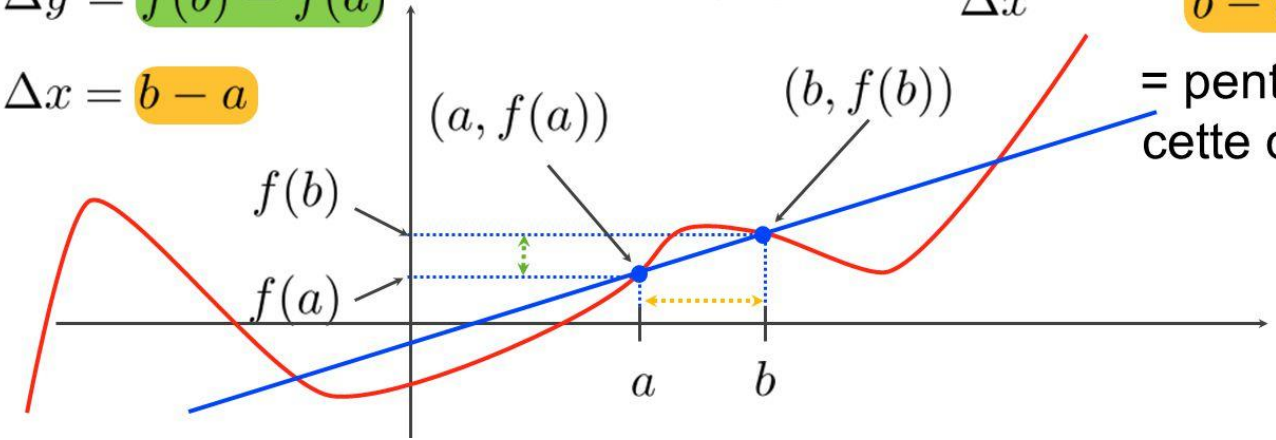


$$TVM_{[a,b]} f(x) = \frac{\Delta y}{\Delta x} = \frac{f(b) - f(a)}{b - a}$$

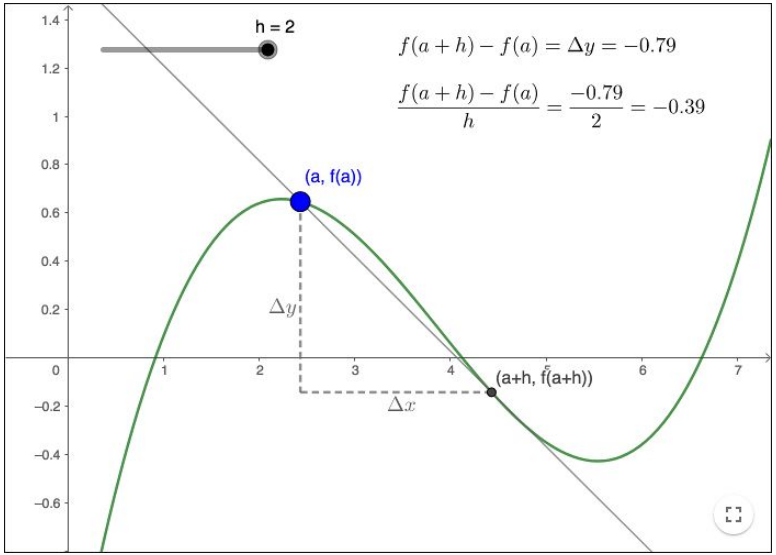
$\Delta y = f(b) - f(a)$

$\Delta x = b - a$

= pente de cette droite



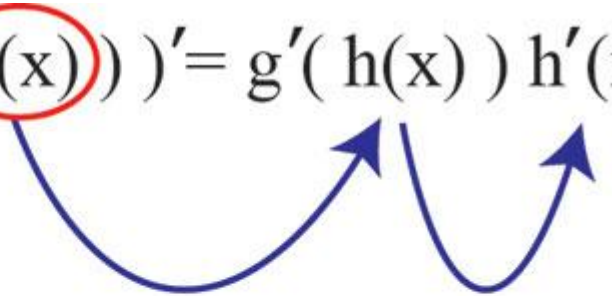
Formule dérivée



$$\frac{dy}{dx} = \lim_{h \to 0} \frac{f(a+h) - f(a)}{h}$$

Dérivée en chaîne

$$f'(x) = (g(h(x)))' = g'(h(x)) h'(x)$$



- keep the inside
- take derivative of outside

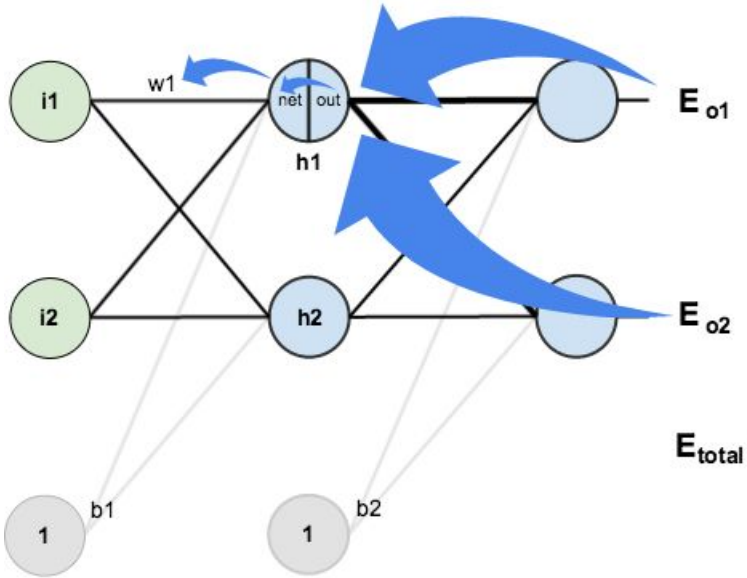
multiply by derivative of the inside

Backpropagation

$$\frac{\partial E_{total}}{\partial w_1} = \frac{\partial E_{total}}{\partial out_{h1}} * \frac{\partial out_{h1}}{\partial net_{h1}} * \frac{\partial net_{h1}}{\partial w_1}$$

$$\downarrow$$

$$\frac{\partial E_{total}}{\partial out_{h1}} = \frac{\partial E_{o1}}{\partial out_{h1}} + \frac{\partial E_{o2}}{\partial out_{h1}}$$

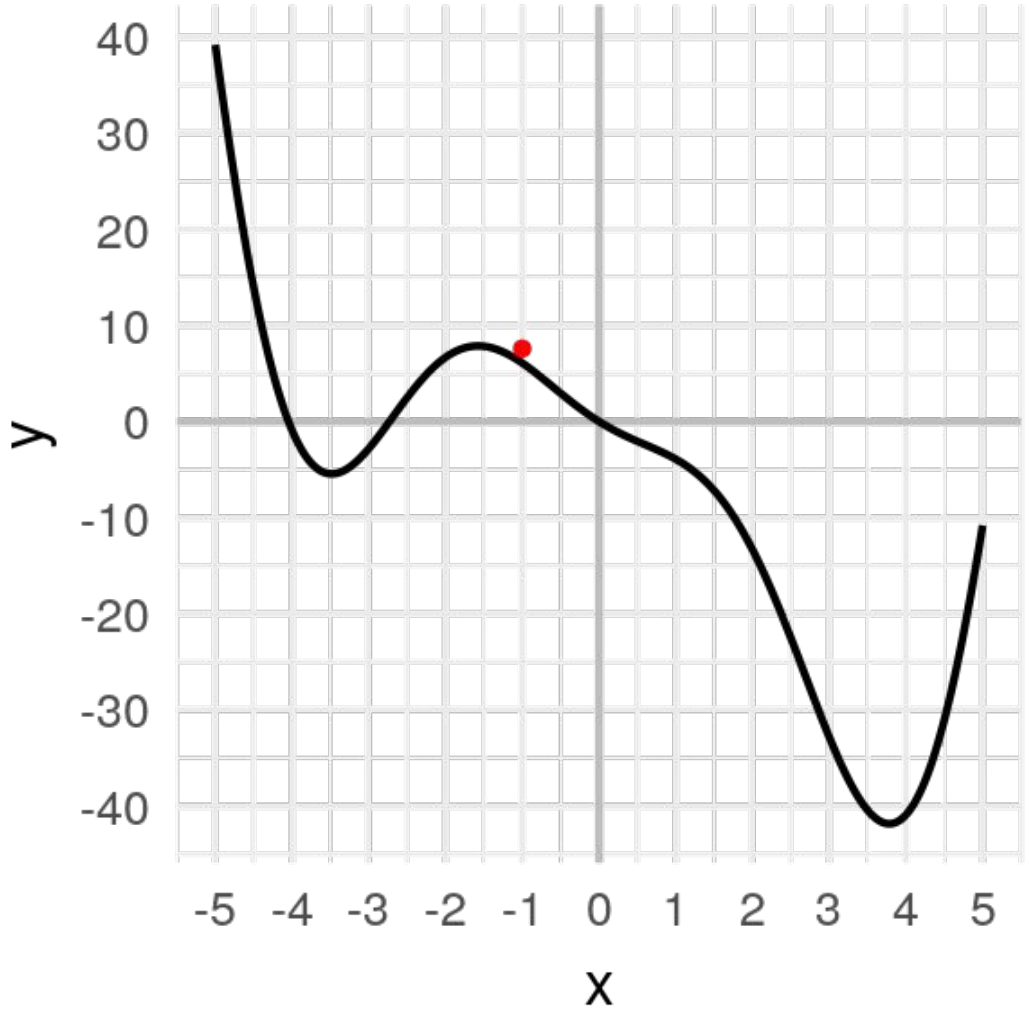


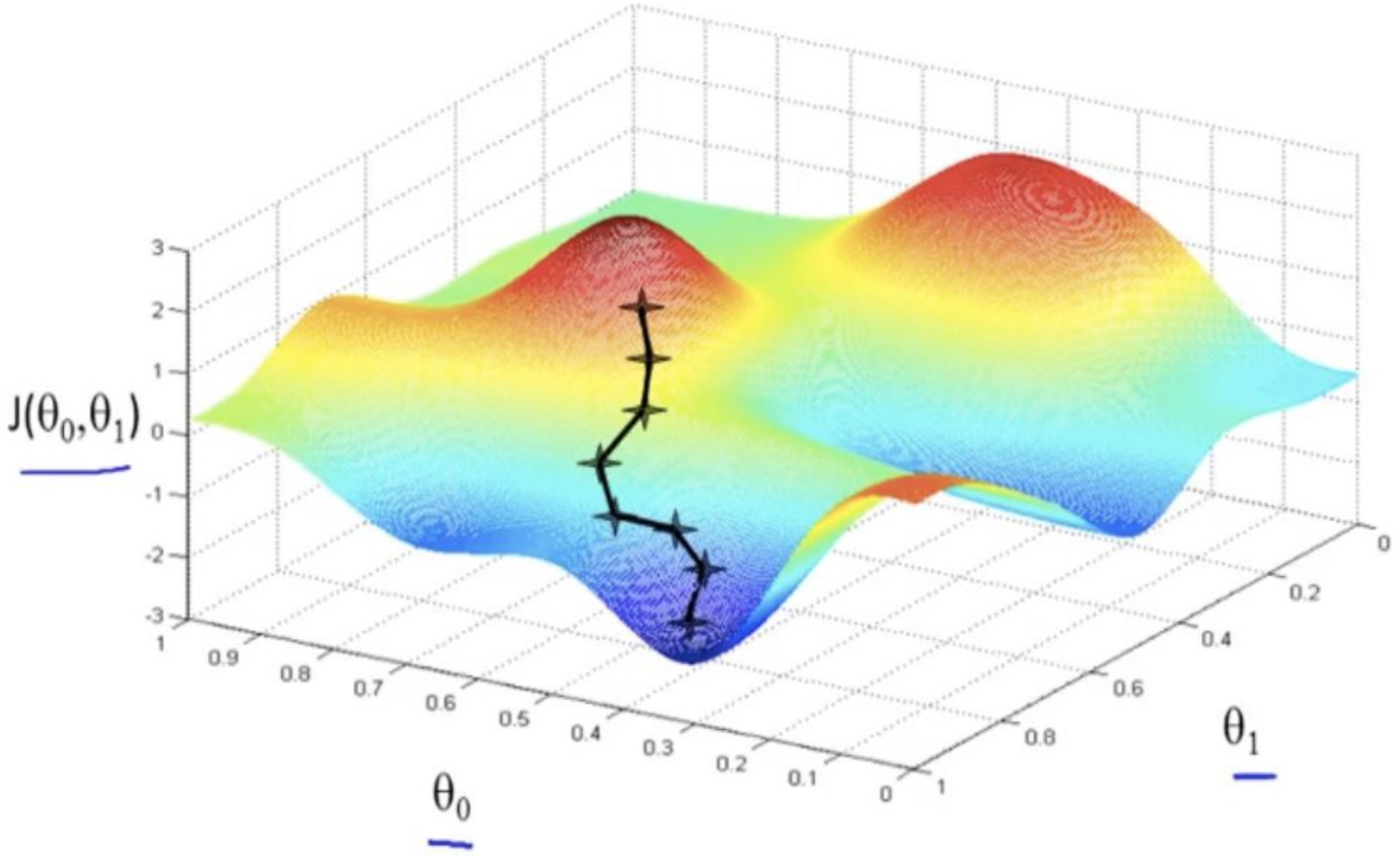
$$E_{total} = E_{o1} + E_{o2}$$

Descente de gradient

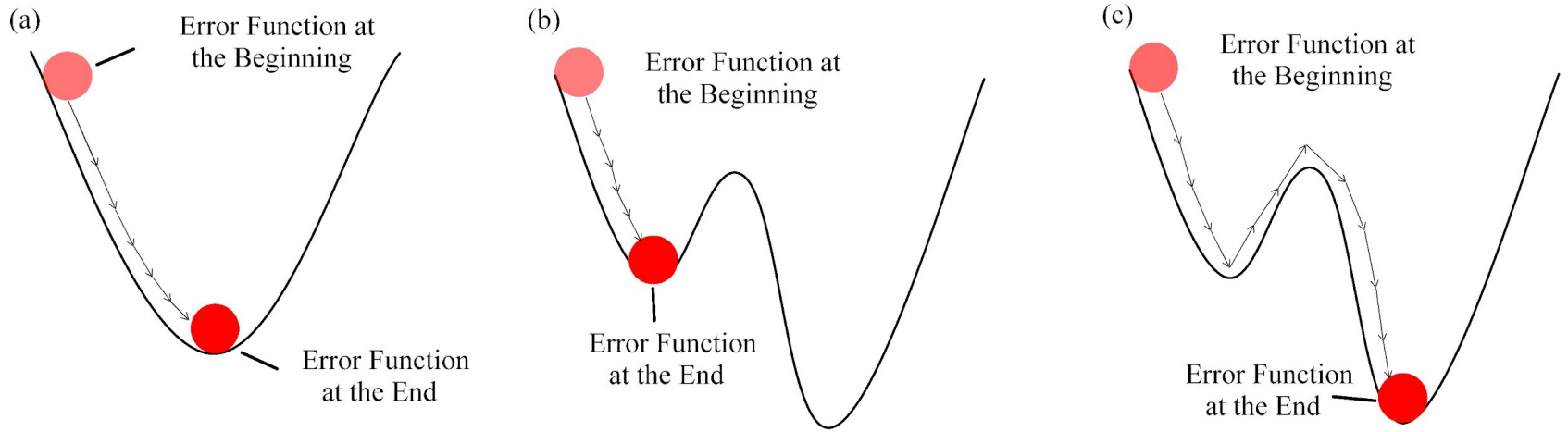
But: trouver minimum d’une fonction

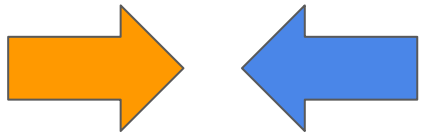
$$w \leftarrow w - \alpha \frac{\partial \mathcal{L}(y,\hat{y})}{\partial w}$$

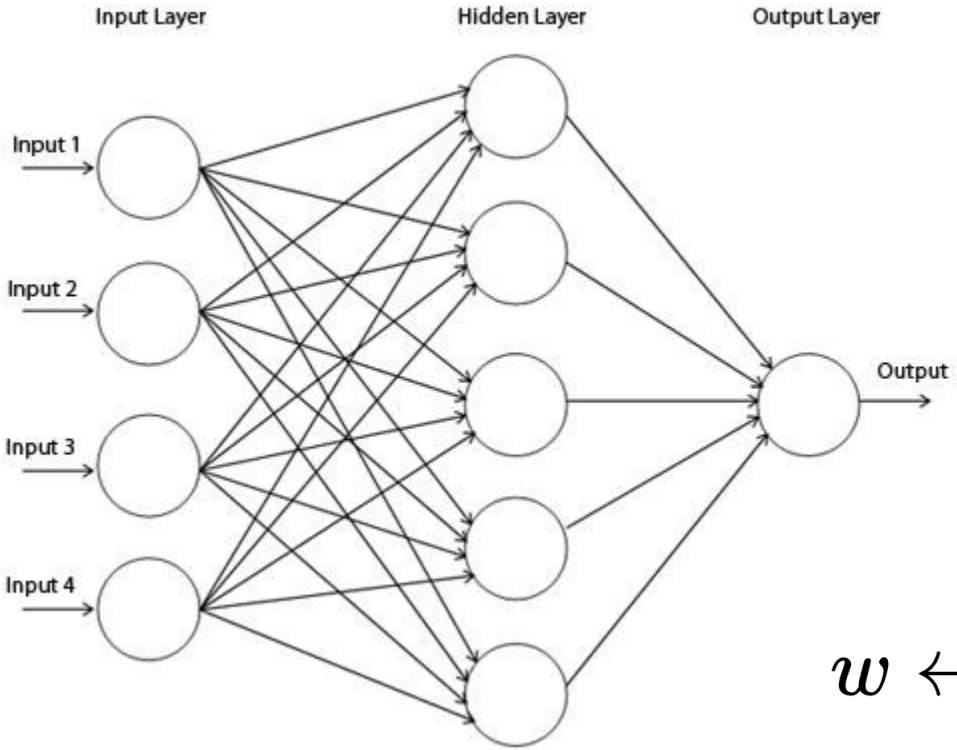

40
30
20
10
0
-10
-20
-30
-40
y
-5 -4 -3 -2 -1 0 1 2 3 4 5
x


J(θ0,θ1)
3
2
1
0
-1
-2
-3
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
θ0
0
0.2
0.4
0.6
0.8
1
θ1

Minimum local et inertie







$$\mathcal{L}(\hat{y}, y) = (\hat{y} - y)^2$$

$$w \leftarrow w - \alpha \frac{\partial \mathcal{L}(y, \hat{y})}{\partial w}$$


TensorFlow


PyTorch

Tensorflow playground

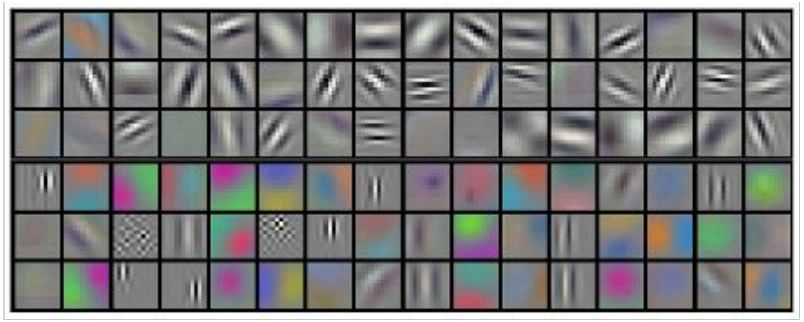
https://playground.tensorflow.org

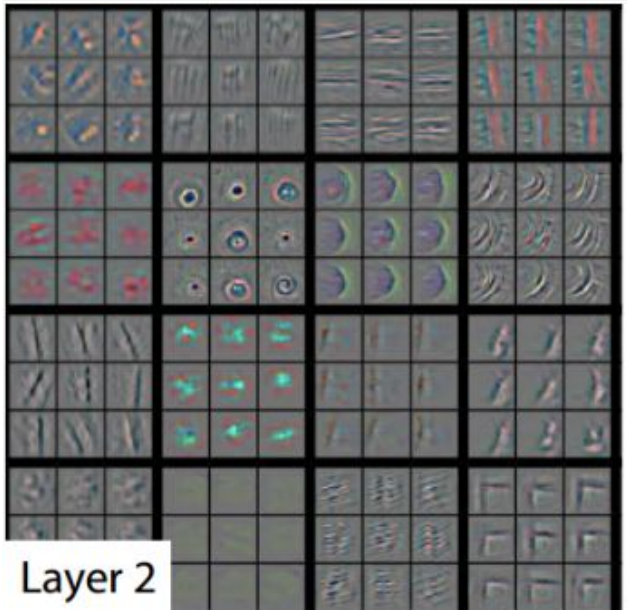
Colab

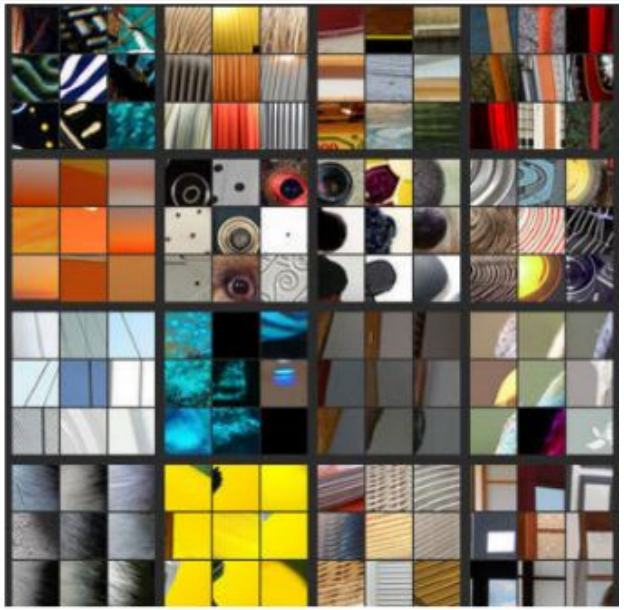
https://colab.research.google.com/github/rpi-techfundamentals/fall2018-materials/blob/master/10-deep-learning/04-pytorch-mnist.ipynb

Qu’apprend le réseau de neurones ?

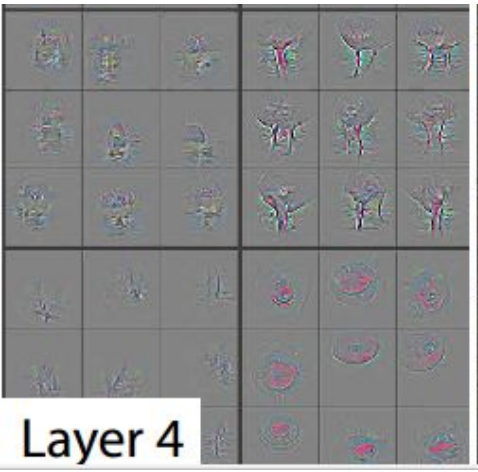
Layer 1

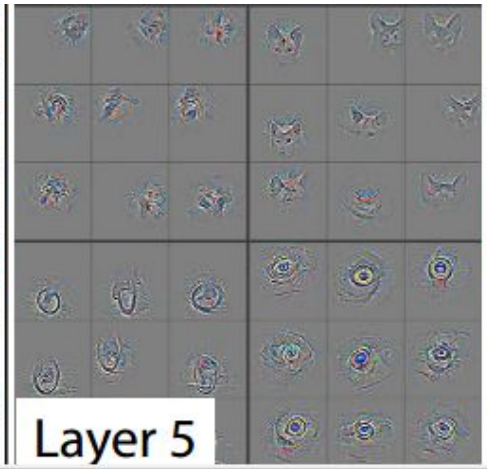


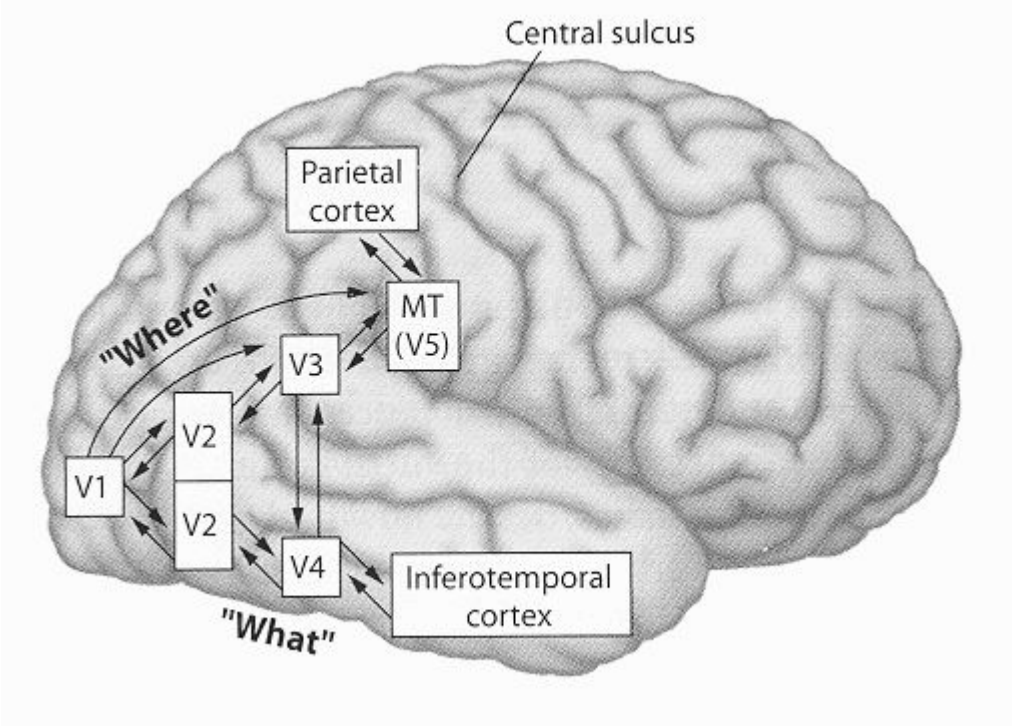




Layer 2

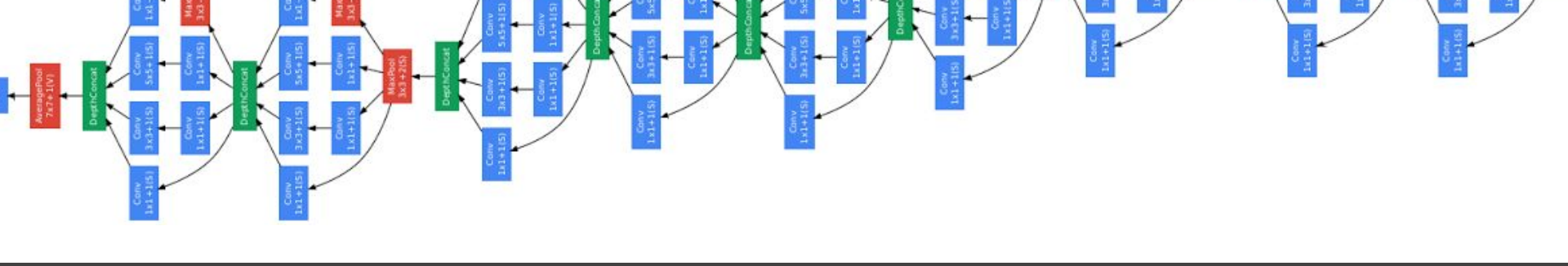

Layer 4

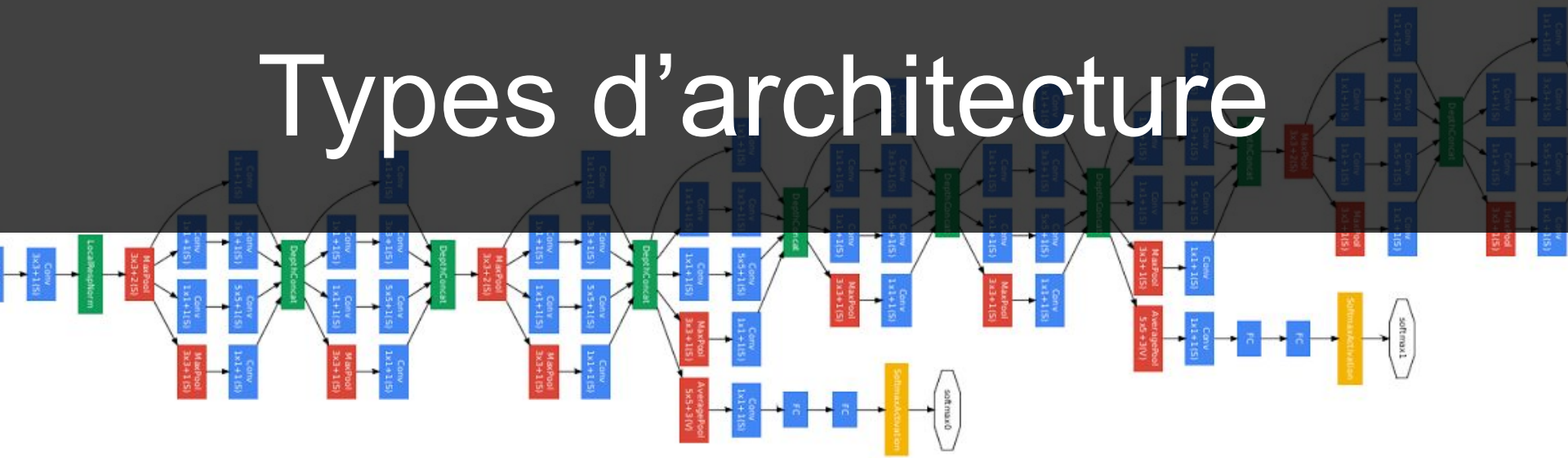

Layer 5



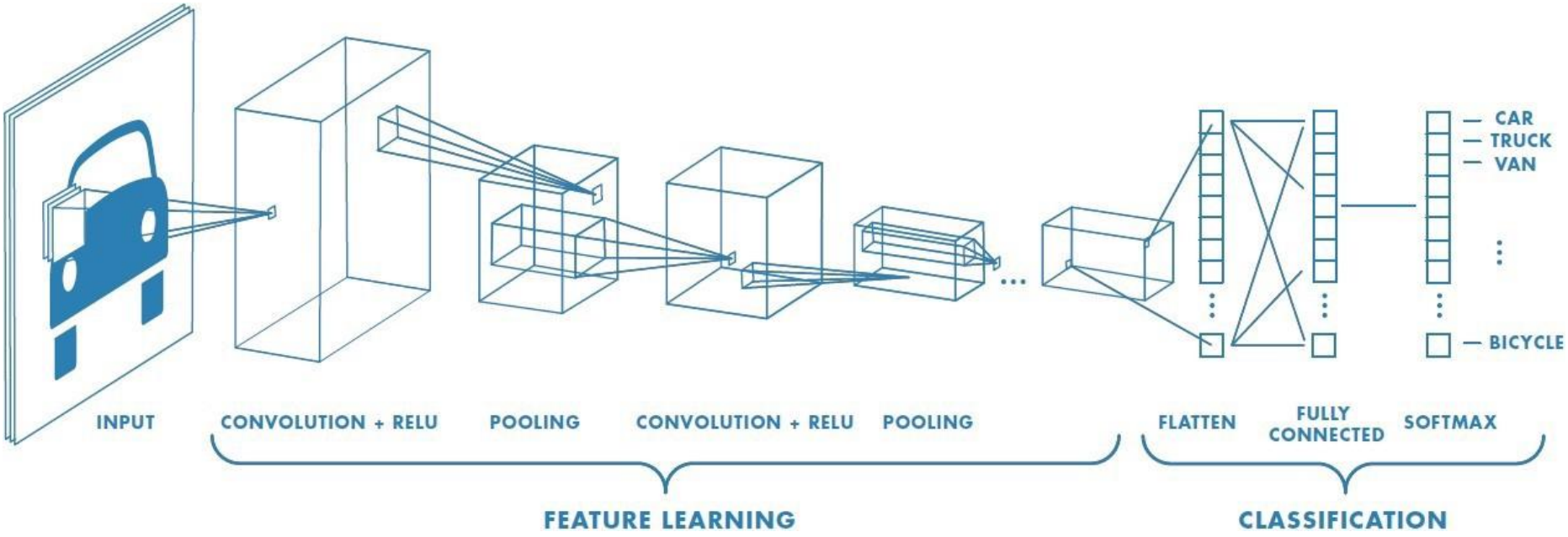
https://distill.pub/2019/activation-atlas/

Types d’architecture

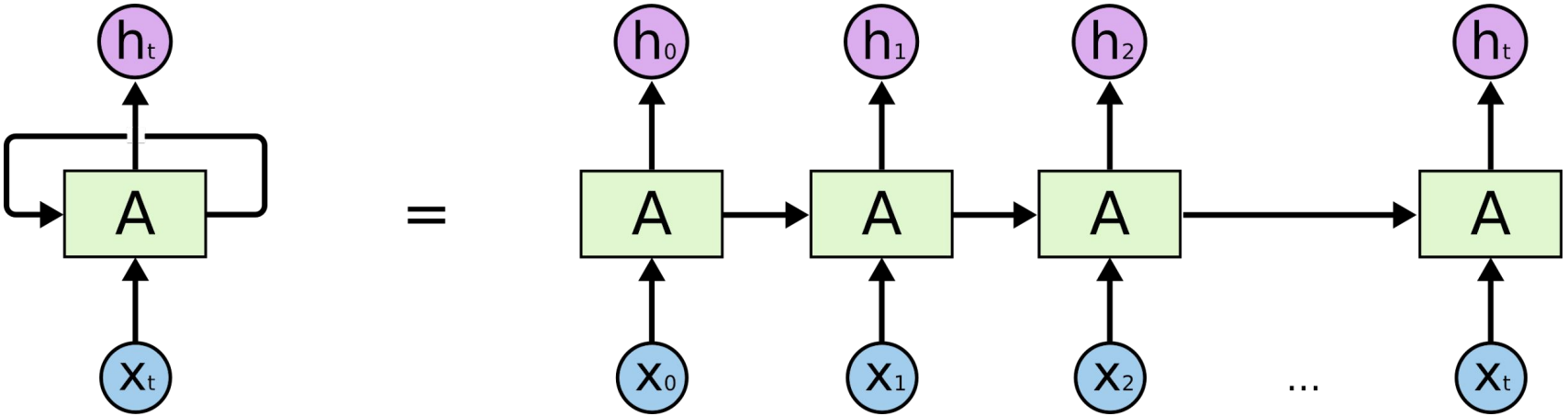




CNN: pour les images

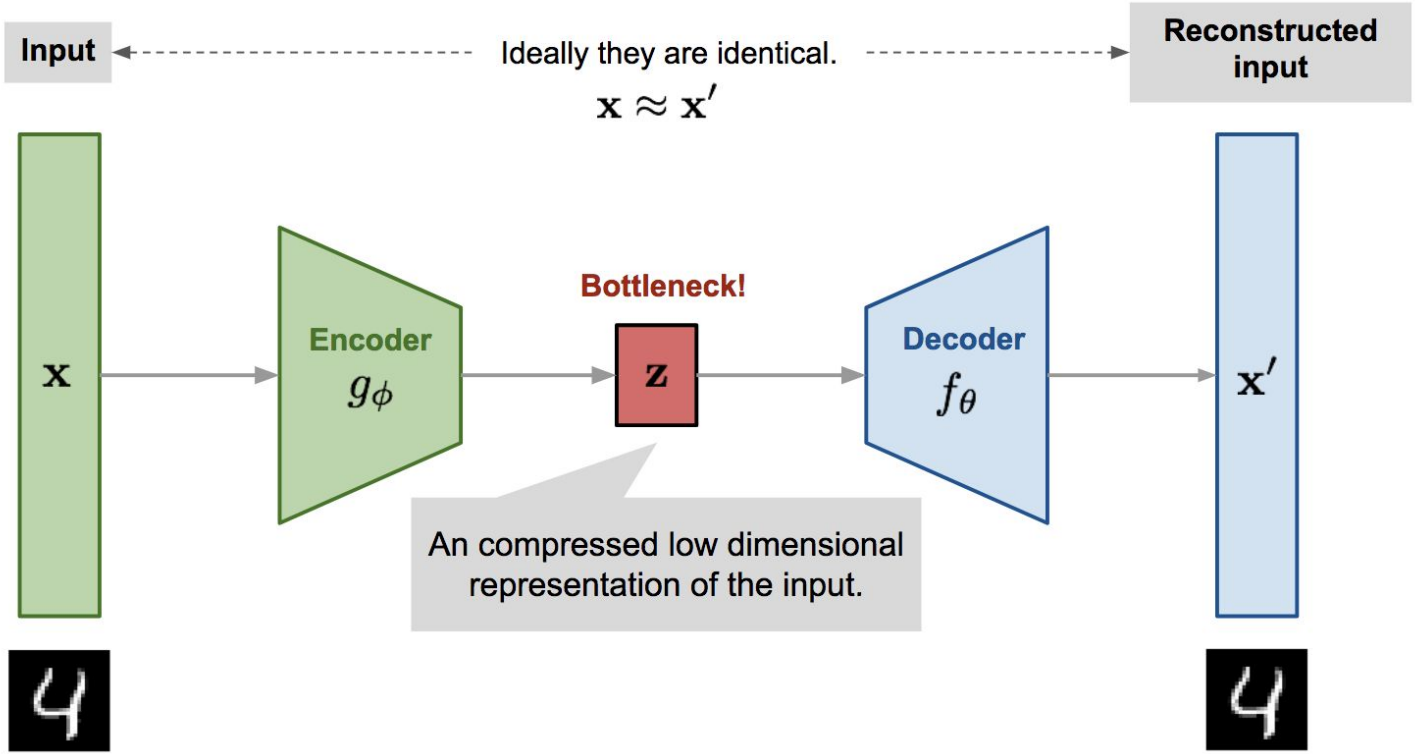


RNN: pour les données séquentielles

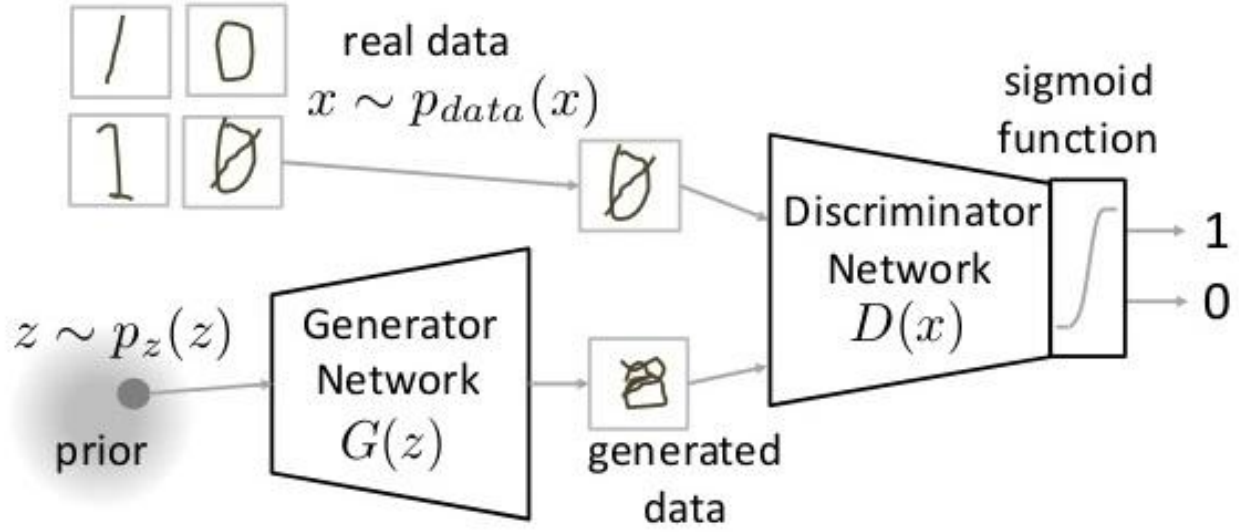


Wavenet:
https://deepmind.com/blog/wavenet-generative-model-raw-audio/

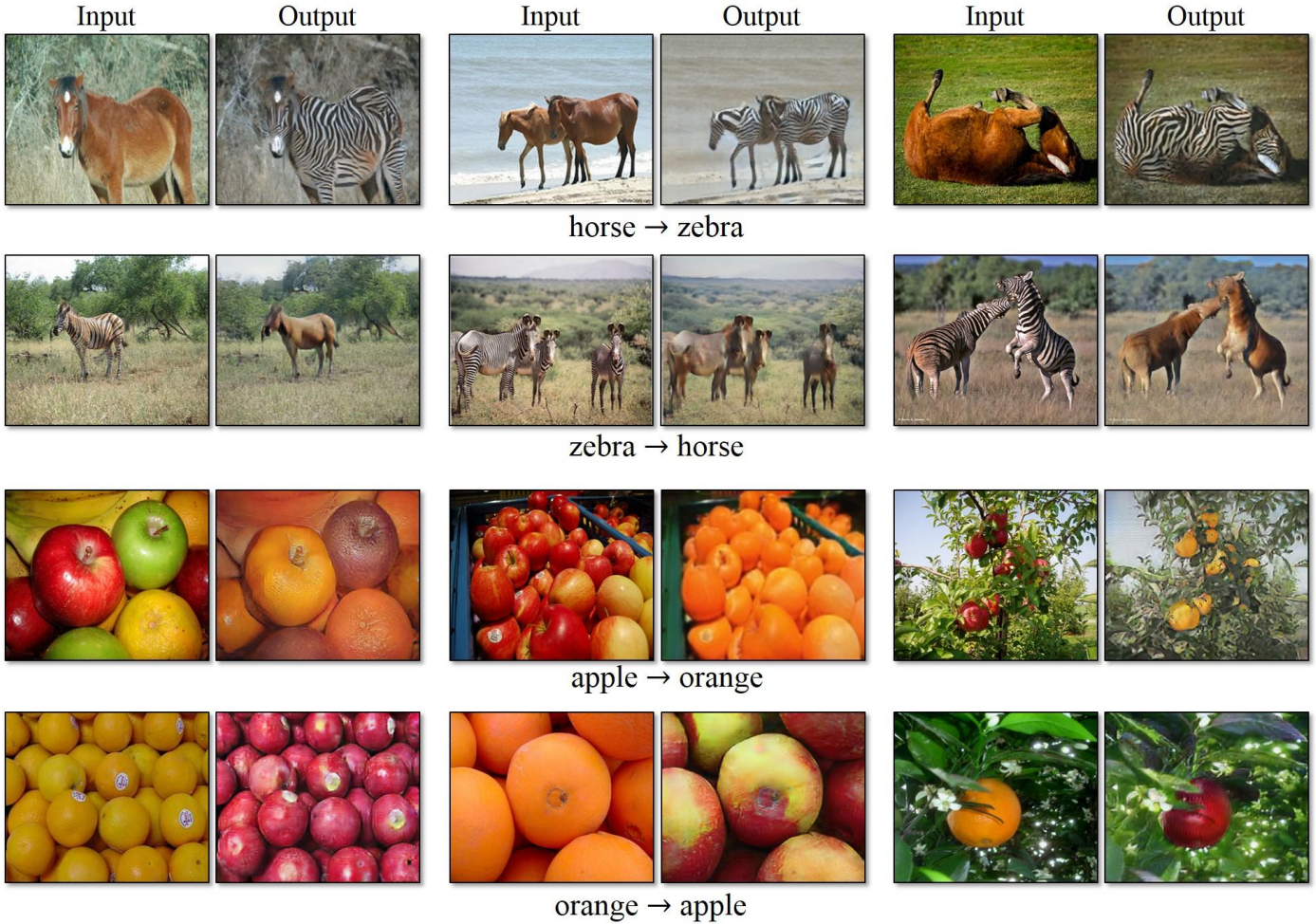
Auto-encodeur



GAN: pour générer des images



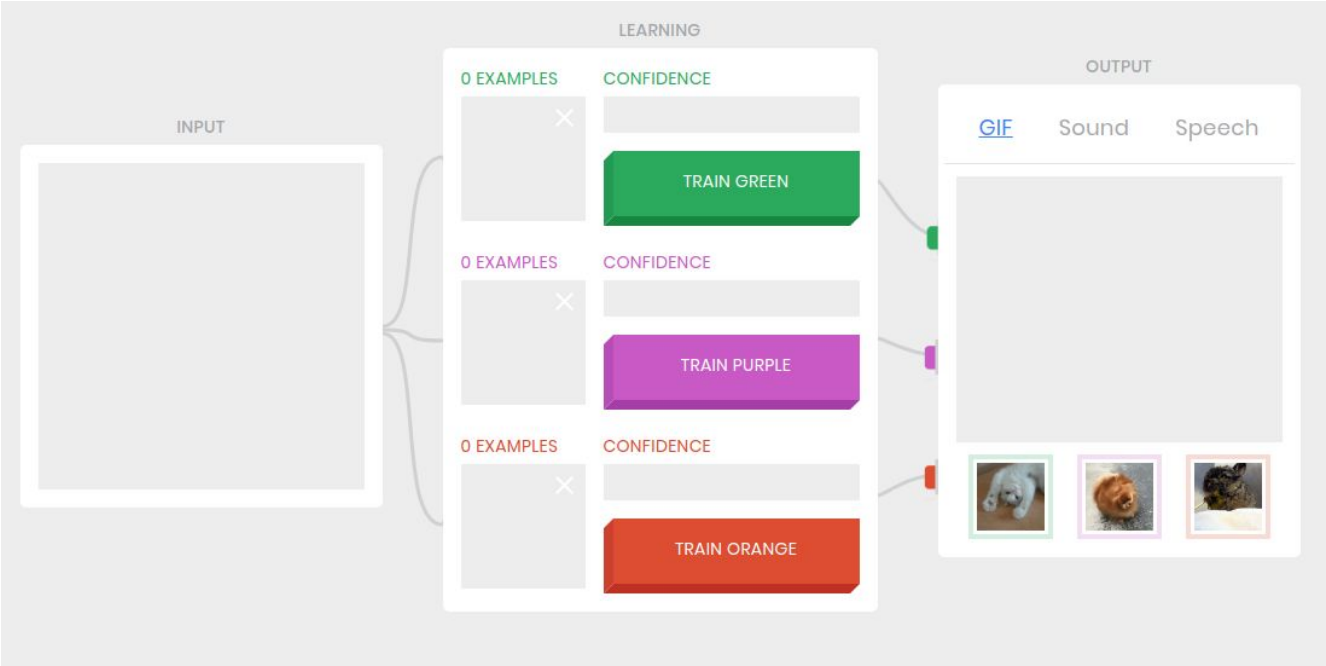
https://zaidalyafeai.github.io/pix2pix/cats.html


Input
Output
Input
Output
Input
Output
horse → zebra
zebra → horse
apple → orange
orange → apple

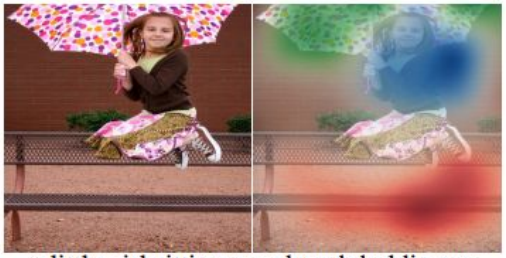
Défis futurs

Apprendre avec peu de données

Démo



https://teachablemachine.withgoogle.com/



a little girl sitting on a bench holding an umbrella.

a herd of sheep grazing on a lush green hillside.

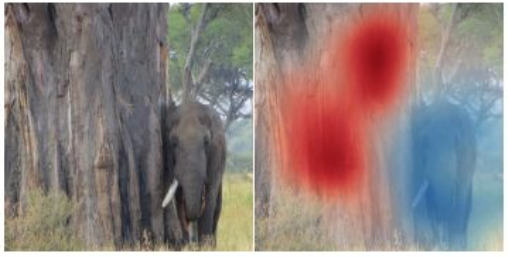
a close up of a fire hydrant on a sidewalk.

a yellow plate topped with meat and broccoli.

a zebra standing next to a zebra in a dirt field.

a stainless steel oven in a kitchen with wood cabinets.

two birds sitting on top of a tree branch.



an elephant standing next to rock wall.

a man riding a bike down a road next to a body of water.

Autres expériences intéressantes:

https://experiments.withgoogle.com/collection/ai

https://magenta.tensorflow.org/demos

http://ml4a.github.io/guides/

Deep Neural Network (Pretraining)

SVM

Multi-layered Perceptron (Backpropagation)

ADALINE

XOR Problem

Perceptron

Golden Age

Dark Age ("AI Winter")

Electronic Brain

1943 1957 1960 1969 1986 1995 2006

1940 1950 1960 1970 1980 1990 2000 2010

S. McCulloch - W. Pitts

F. Rosenblatt

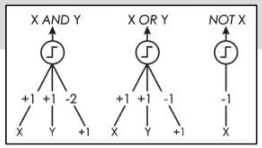
B. Widrow - M. Hoff

M. Minsky - S. Papert

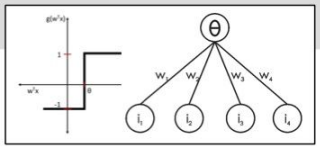
D. Rumelhart - G. Hinton - R. Wiliams

V. Vapnik - C. Cortes

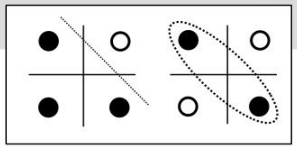
G. Hinton - S. Ruslan



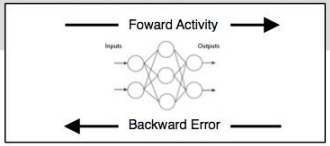
- Adjustable Weights
- Weights are not Learned



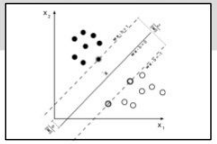
- Learnable Weights and Threshold



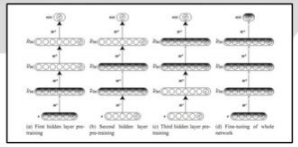
- XOR Problem



- Solution to nonlinearly separable problems
- Big computation, local optima and overfitting



- Limitations of learning prior knowledge
- Kernel function: Human Intervention



- Hierarchical feature Learning