Deep Learning

Seq1- History, Fundamental Concepts

Objectives

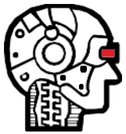
Understand what Deep Learning is, its concepts and basics,

Develop a first experience on simple and ...understandable cases

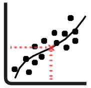
Learn how to use tools and mutualized resources (Jupyter)

Roadmap

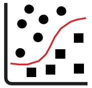
1



History, Fundamental Concepts



Basic Regression DNN



Basic Classification DNN

2

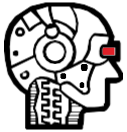
Hight Dimensionnal Data

CNN

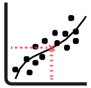
Roadmap

Episode :S01E01

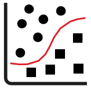
1



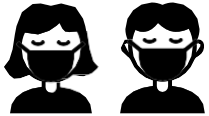
History,
Fundamental
Concepts



Basic
Regression
DNN



Basic
Classification
DNN



1.1 Introduction
Context, tools and ressources

1.2 From the linear regression
to the first neuron

1.3 Neurons incontroversy

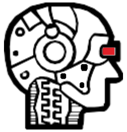
1.4 Data and neurons

- Basic Regression
- Basic Classification

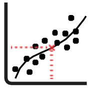
Roadmap

Episode :S01E01

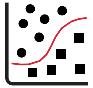
1



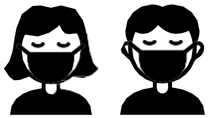
History,
Fundamental
Concepts



Basic
Regression
DNN



Basic
Classification
DNN



1.1 Introduction
Context, tools and ressources

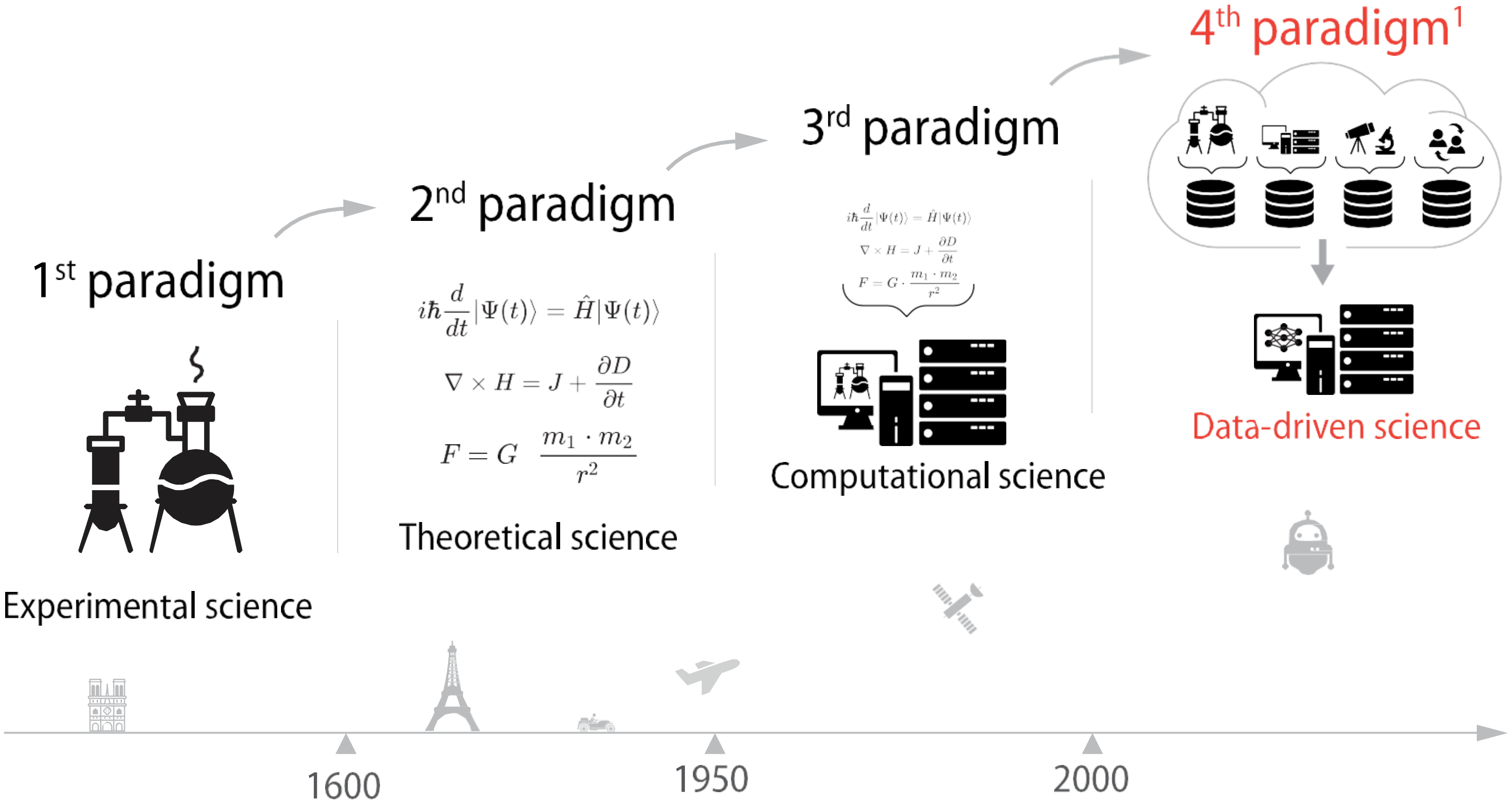
1.2 From the linear regression
to the first neuron

1.3 Neurons incontroversy

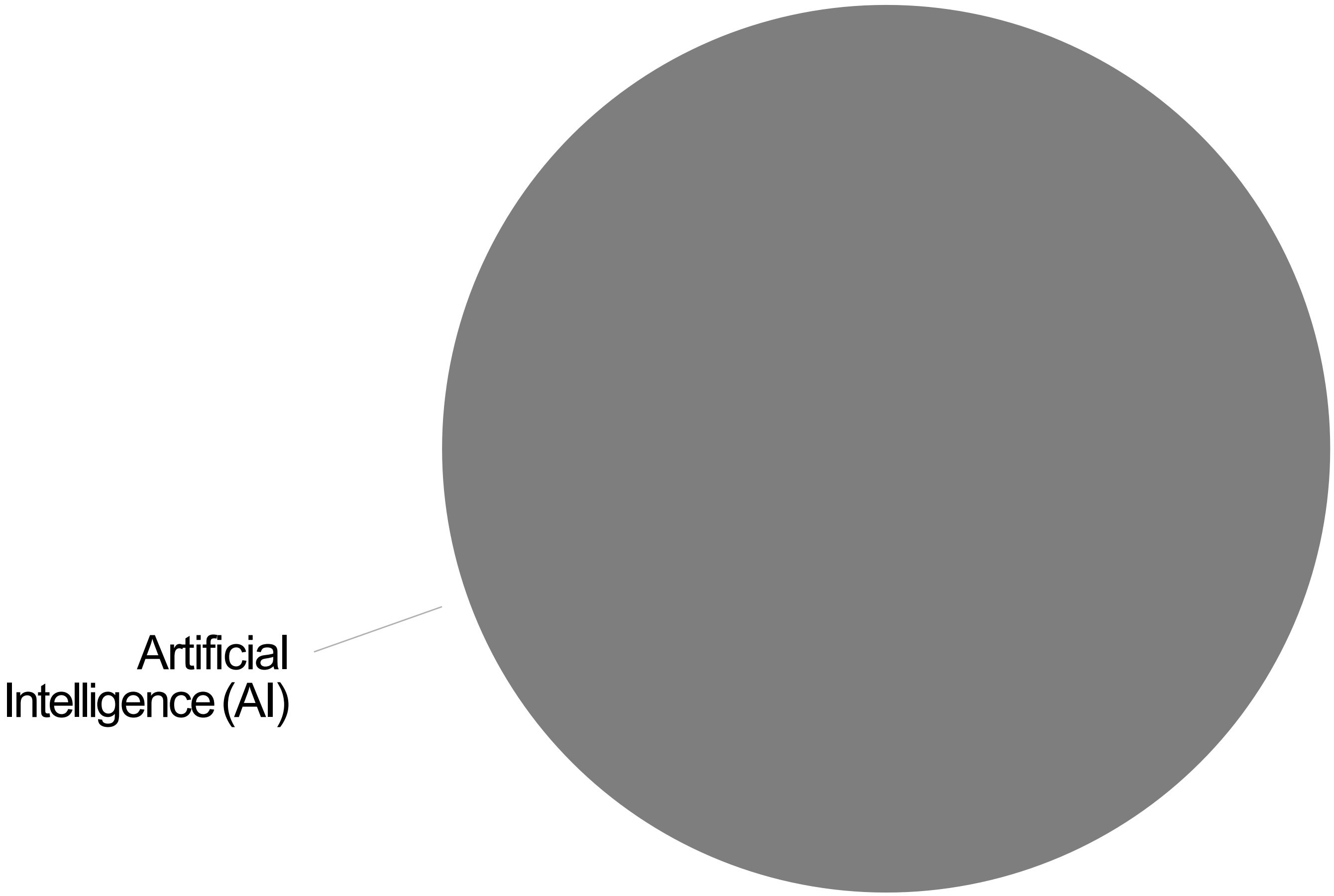
1.4 Data and neurons

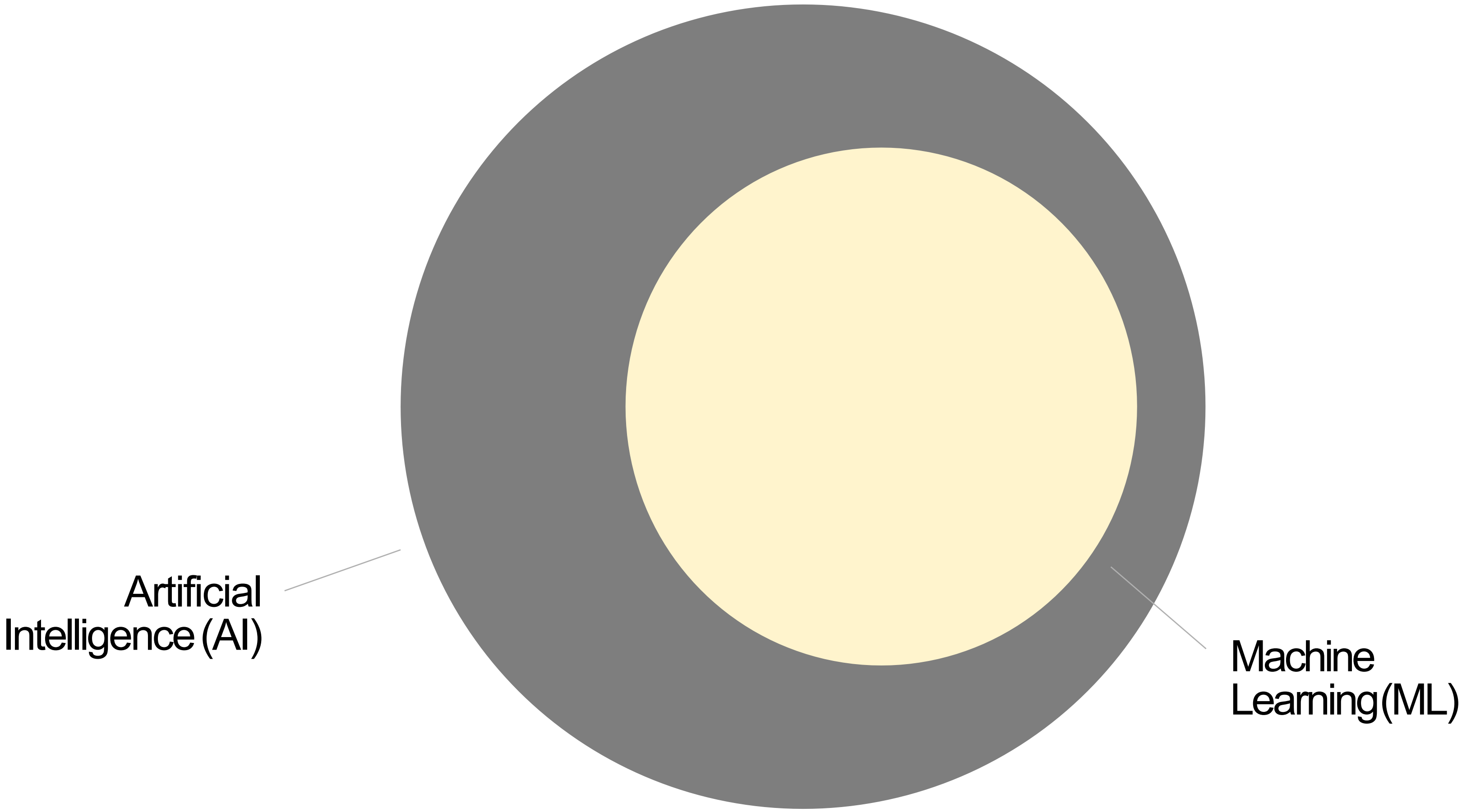
- Basic Regression
- Basic Classification

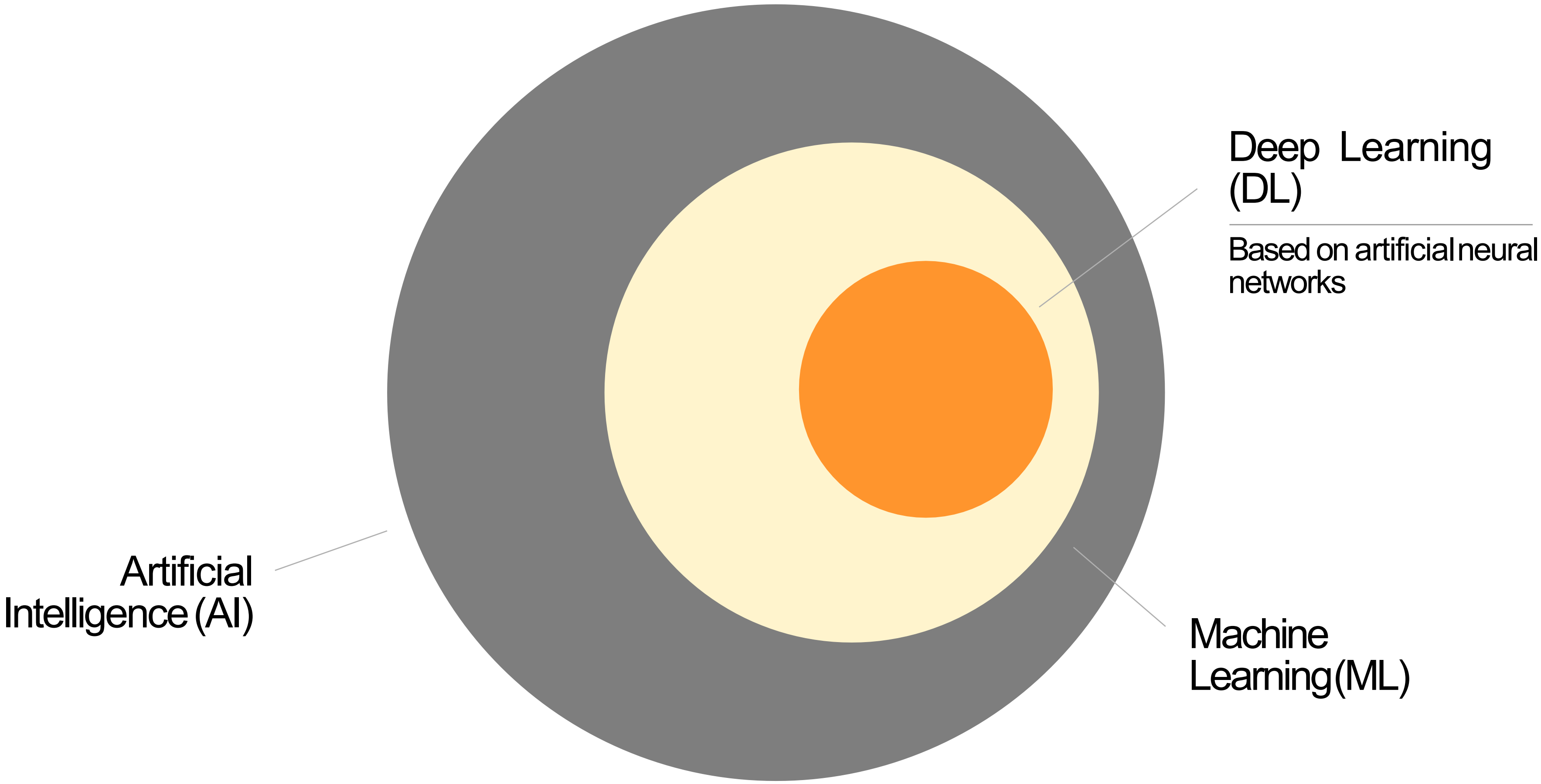
[Méthode scientifique]



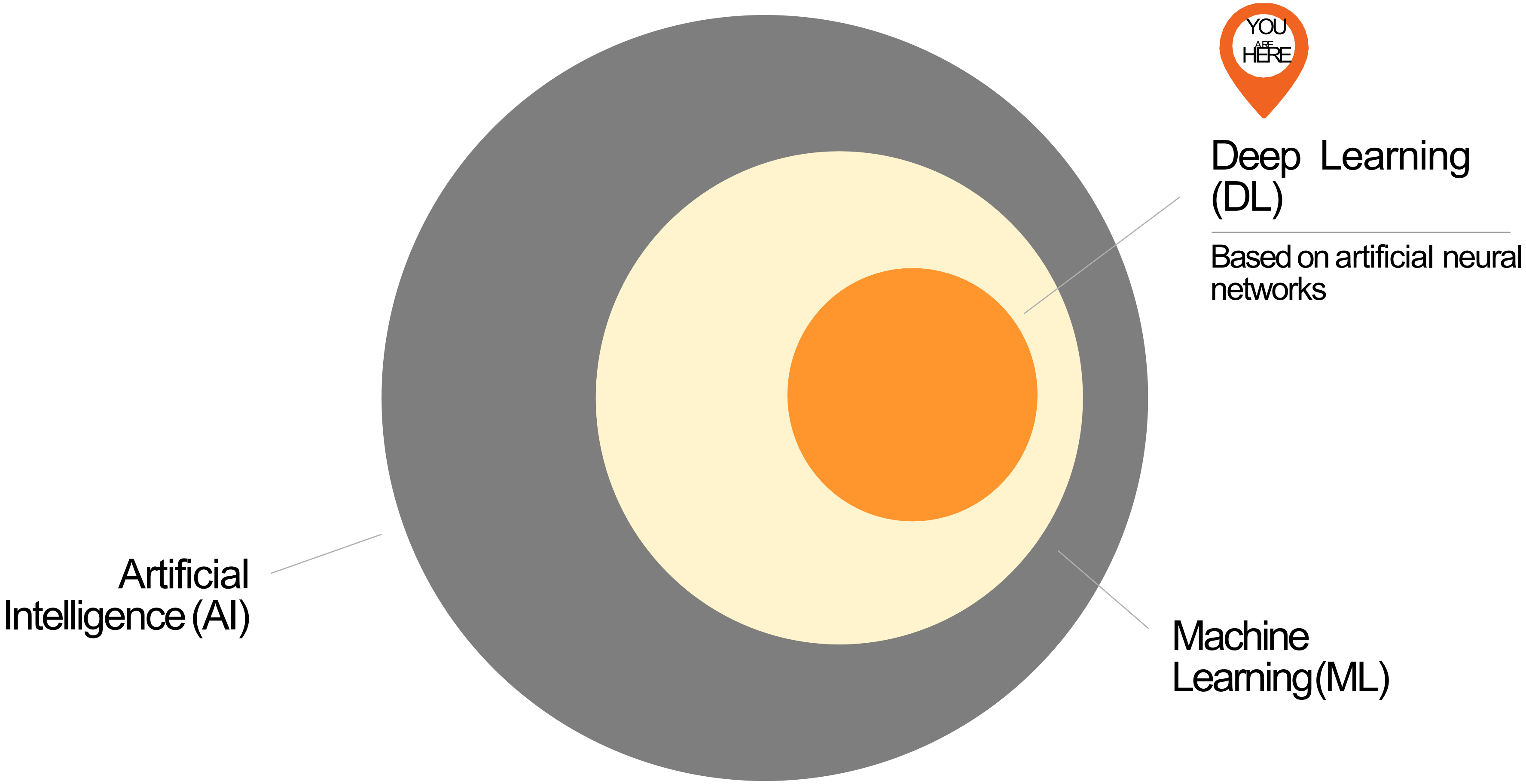
[1] Jim Gray, 2007 [GRAY]


Artificial
Intelligence (AI)

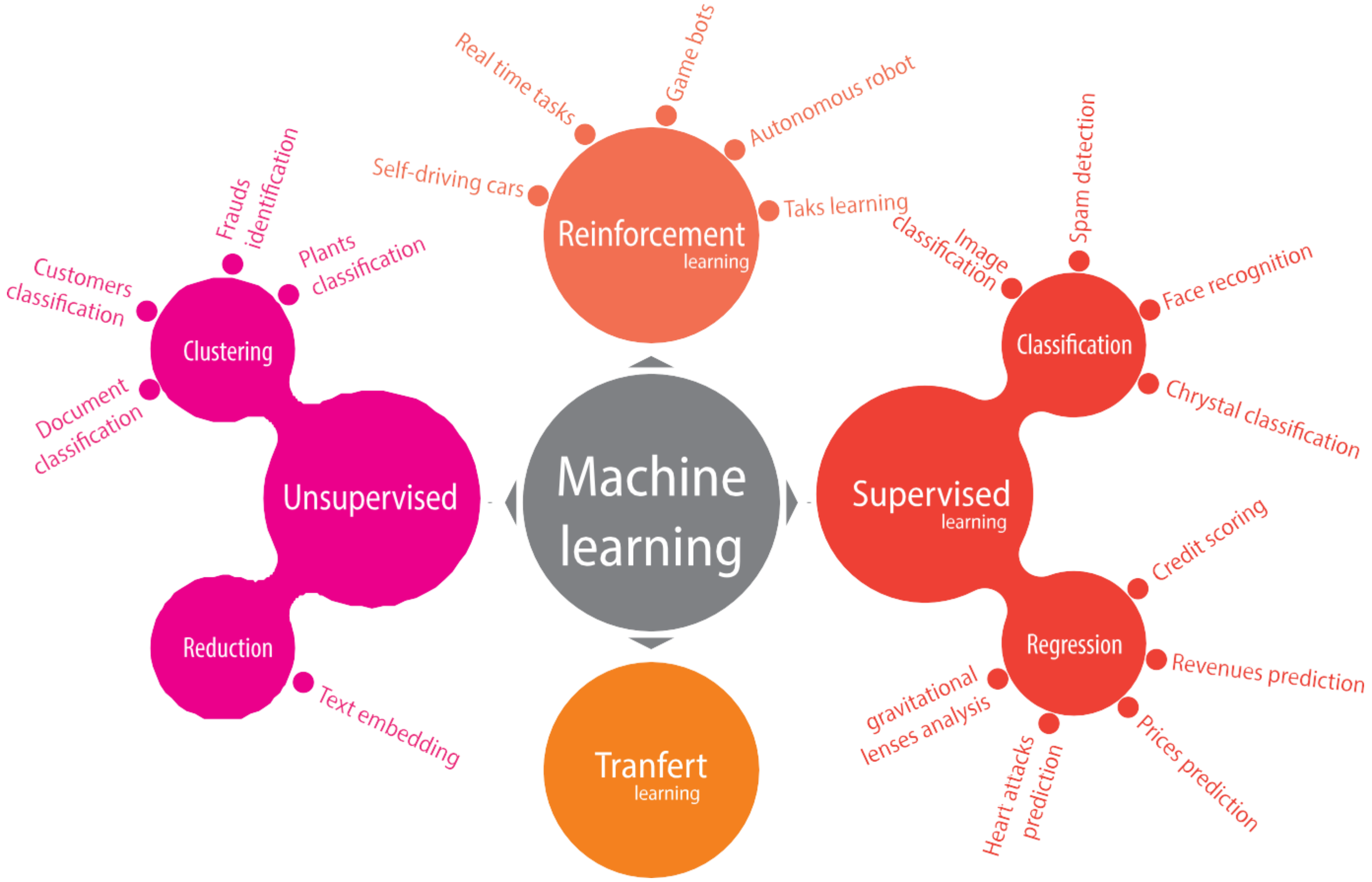

Artificial
Intelligence (AI)
Machine
Learning (ML)


Deep Learning (DL)
Based on artificial neural networks
Machine Learning (ML)
Artificial Intelligence (AI)

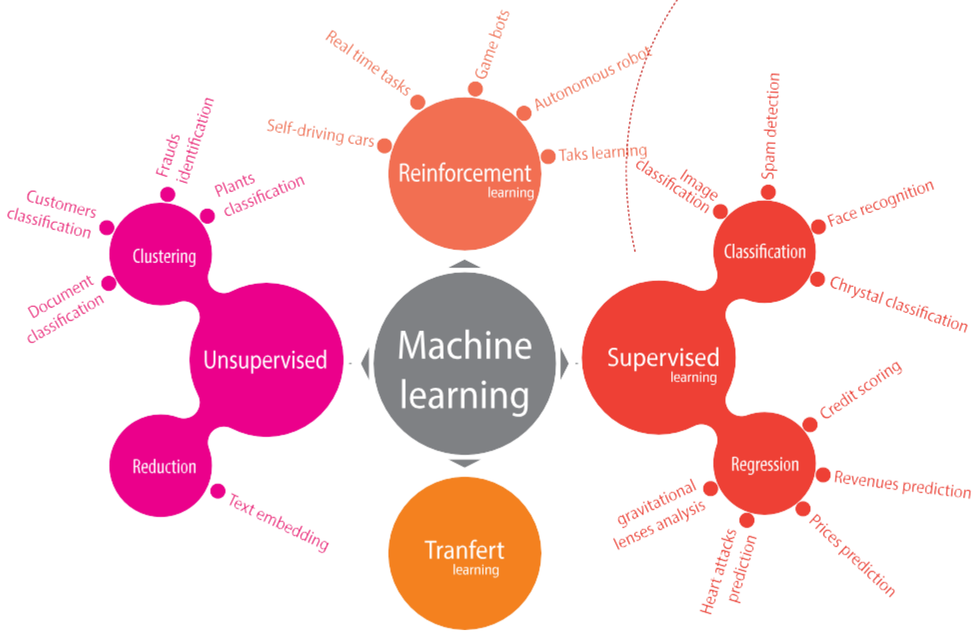
[*-learning]


YOU ARE HERE
Deep Learning (DL)
Based on artificial neural networks
Machine Learning (ML)
Artificial Intelligence (AI)

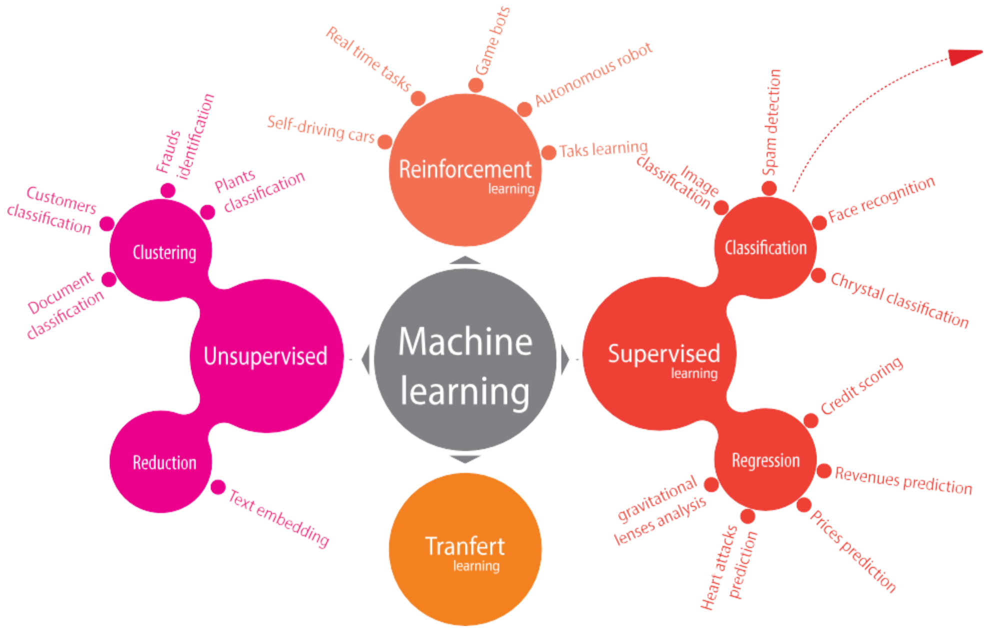
[*-learning]


Machine learning
Reinforcement learning
Real time tasks
Game bots
Autonomous robot
Self-driving cars
Taks learning
Unsupervised
Clustering
Frauds identification
Plants classification
Customers classification
Document classification
Reduction
Text embedding
Supervised learning
Classification
Image classification
Spam detection
Face recognition
Chrystal classification
Regression
Credit scoring
Revenues prediction
gravitational lenses analysis
Heart attacks prediction
Prices prediction
Tranfert learning

Supervised learning

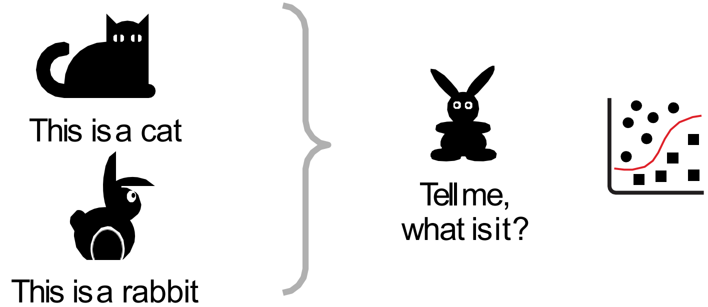


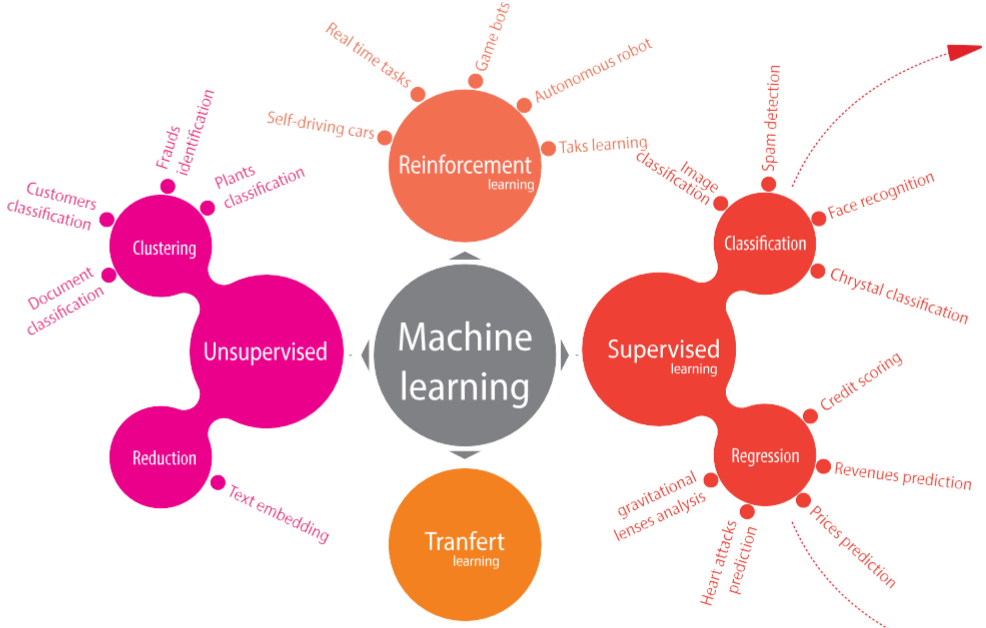
Learning from examples



Classification :

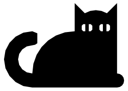
Predict qualitative informations



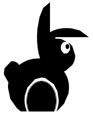


Classification :

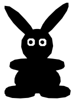
Predict qualitative informations



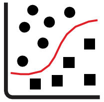
This is a cat



This is a rabbit

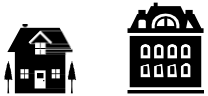


Tell me, what is it?



Régression :

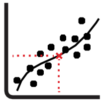
Predict quantitative informations



150K€ 400K€

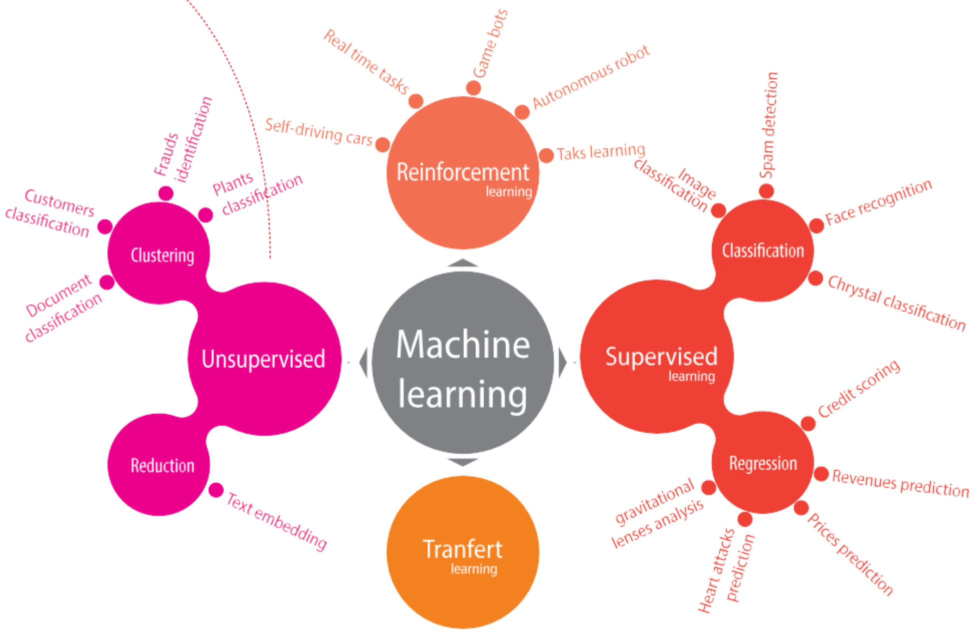
120K€ 100K€

Tell me, what's the price?



Unsupervised learning

Learning
from data
alone



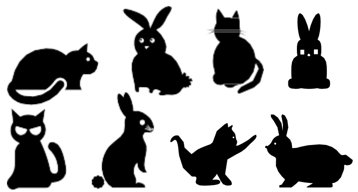
Unsupervised learning

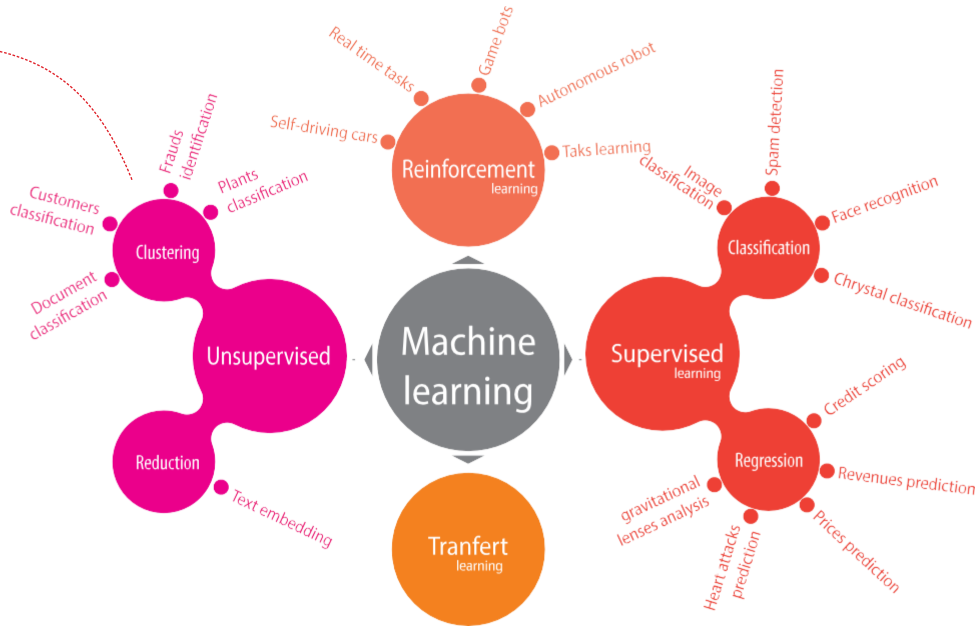
Learning from data alone

Clustering :

Finding Common Relationships

What is the relationship between these data?





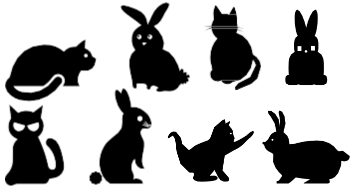
Unsupervised learning

Learning from data alone

Clustering :

Finding Common Relationships

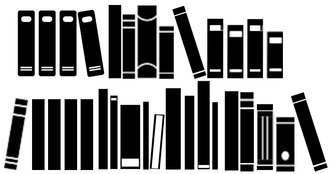
What is the relationship between these data?

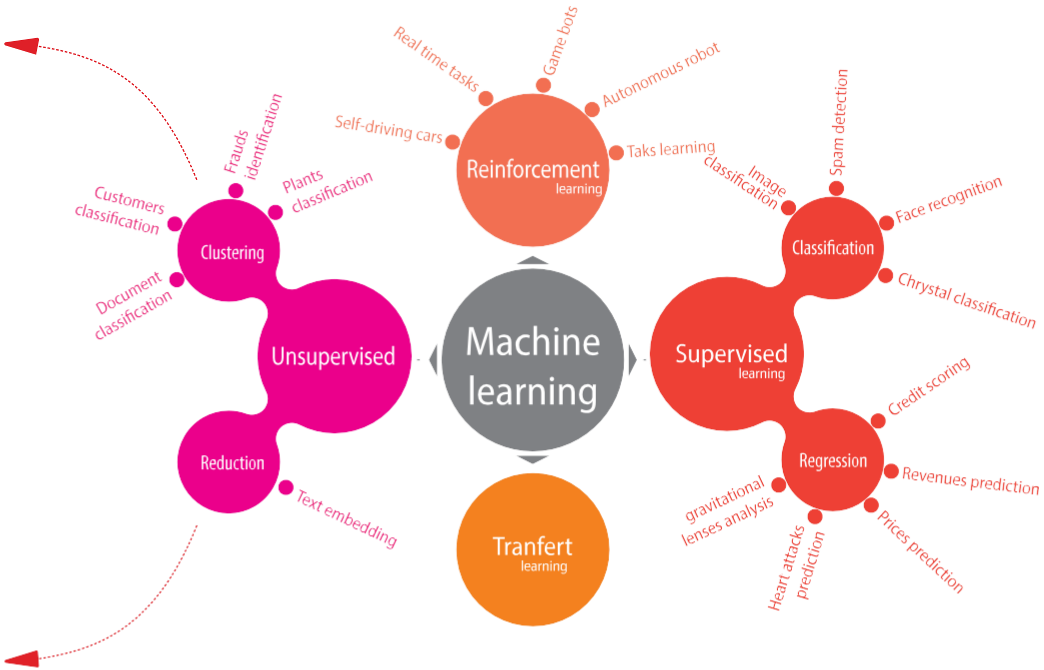


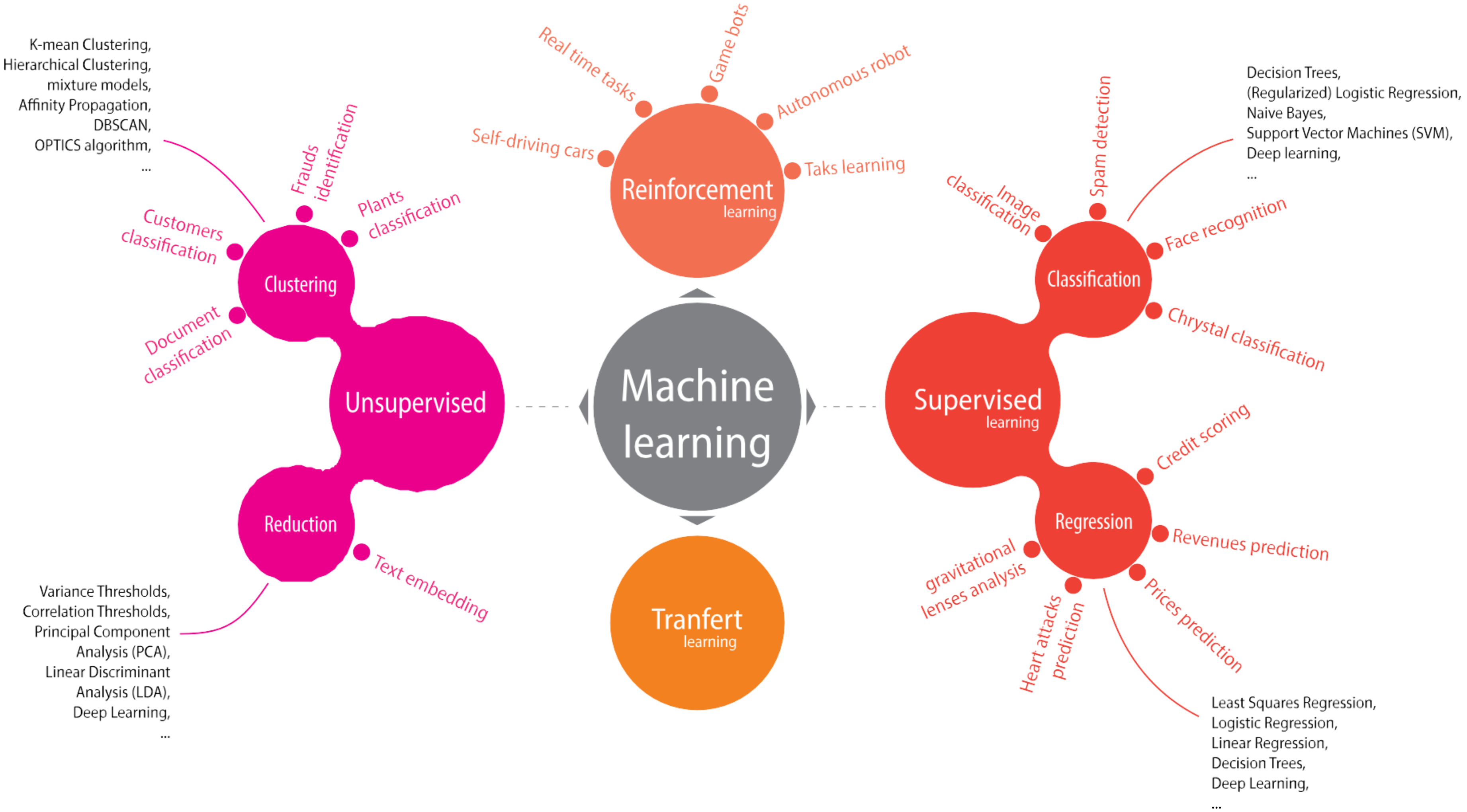
Reduction :

Reduce the number of dimensions

Simplify while keeping meaning






Machine learning
Reinforcement learning
Real time tasks
Game bots
Autonomous robot
Self-driving cars
Taks learning
Unsupervised
Clustering
Reduction
K-mean Clustering,
Hierarchical Clustering,
mixture models,
Affinity Propagation,
DBSCAN,
OPTICS algorithm,
...
Frauds identification
Plants classification
Customers classification
Document classification
Text embedding
Variance Thresholds,
Correlation Thresholds,
Principal Component Analysis (PCA),
Linear Discriminant Analysis (LDA),
Deep Learning,
...
Supervised learning
Classification
Regression
Image classification
Spam detection
Decision Trees,
(Regularized) Logistic Regression,
Naive Bayes,
Support Vector Machines (SVM),
Deep learning,
...
Face recognition
Chrystal classification
Credit scoring
Revenues prediction
gravitational lenses analysis
Heart attacks prediction
Prices prediction
Least Squares Regression,
Logistic Regression,
Linear Regression,
Decision Trees,
Deep Learning,
...
Tranfert learning

ML vs DL

Machine learning in Practice

Input → Feature extraction → Classification →

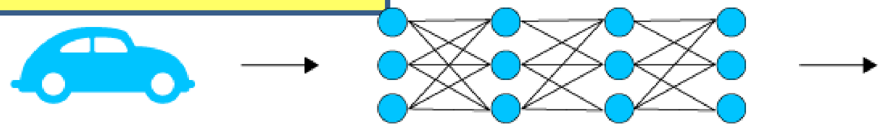
Car
Not Car

Output

Input → Describe your data with features a computer can understand → Learning algorithm → **Output**

Optimizing the weights of features

Deep learning in Practice



Input → Feature extraction + Classification →

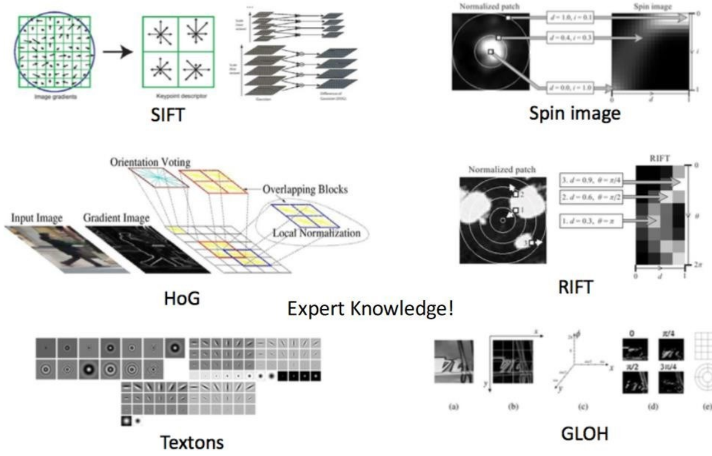
Car
Not Car

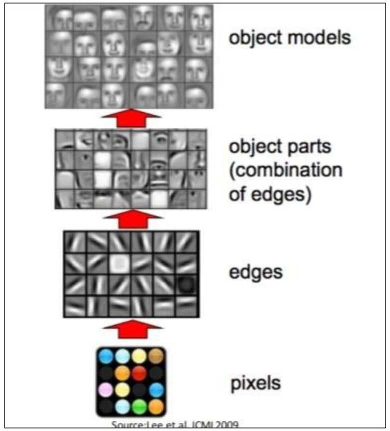
Output

Input → Deep Learning algorithm → **Output**

Optimizing the weights of features

Expert knowledge!

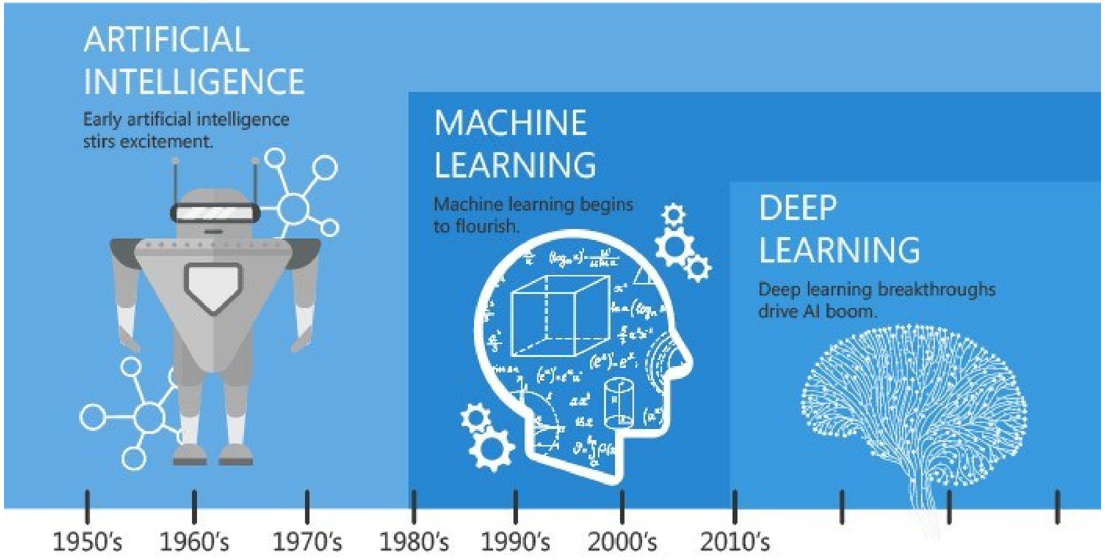




learn representations!

Why DL?

- "Deep" Learning has attracted much attention during these past years.
- DL has stood out impressively in several research fields:
  - facial recognition, speech synthesis, machine translation, and many others.
- These research fields have in common to be perceptual problems related to our senses and our expression.



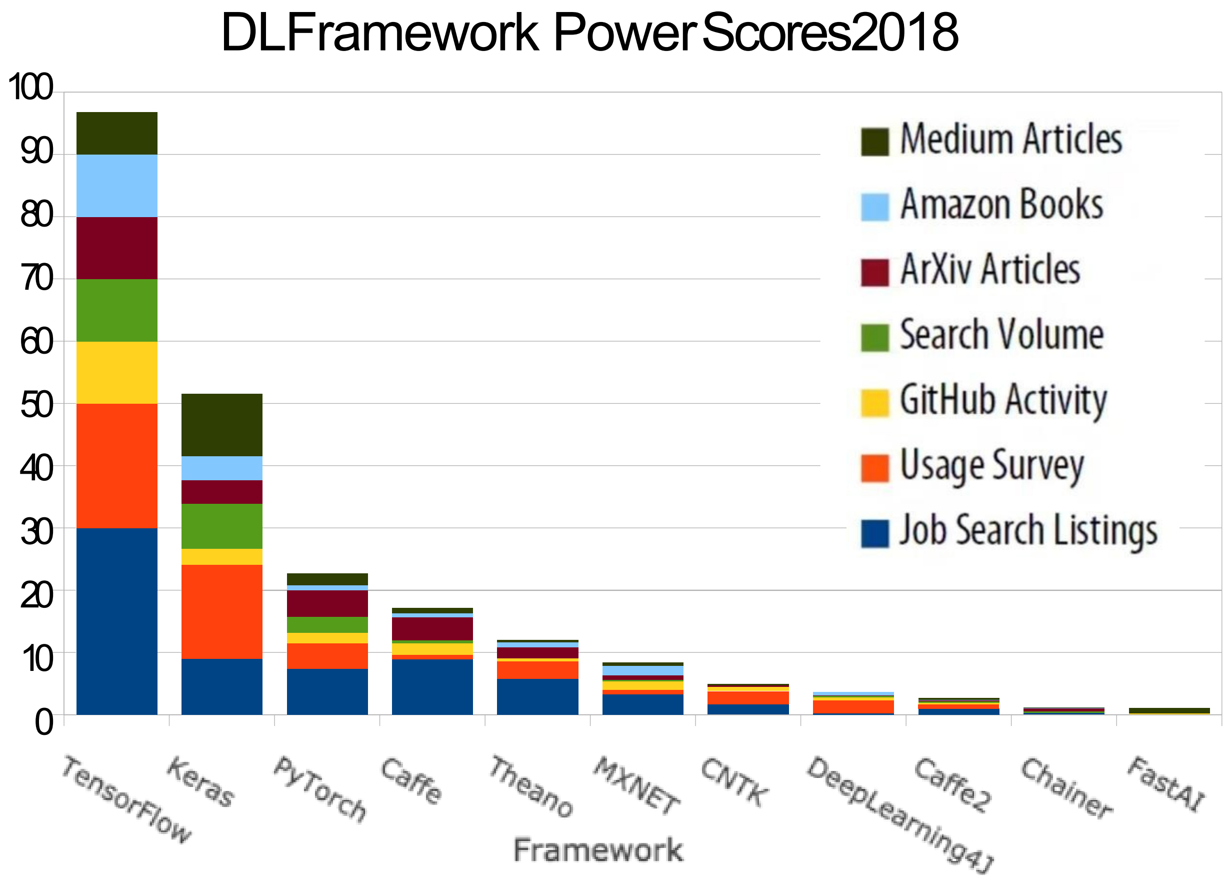
Since an early flush of optimism in the 1950's, smaller subsets of artificial intelligence - first machine learning, then deep learning, a subset of machine learning - have created ever larger disruptions.

Why DL?

- It is extremely difficult to model vision or voice by means of algorithms and mathematical formulas.
- NN attempt to solve problems that would normally be easy for humans but hard for computers!
- Thanks to a very large number of parameters that self-adjust over learning, will learn from implicit links existing in the data.

- Manually designed features are often over-specified, incomplete and take a long time to design and validate !!
- Learned Features are easy to adapt, fast to learn
- Deep learning provides a very flexible, (almost?) universal, learnable framework for representing world, visual and linguistic information.
- Can learn both unsupervised and supervised
- Effective end-to-end joint system learning
- Useful when we have large amounts of training data

A Python centered world



TensorFlow

Most used DL framework
Supported by Google
Low level API – an hard way
Apache licence

Keras

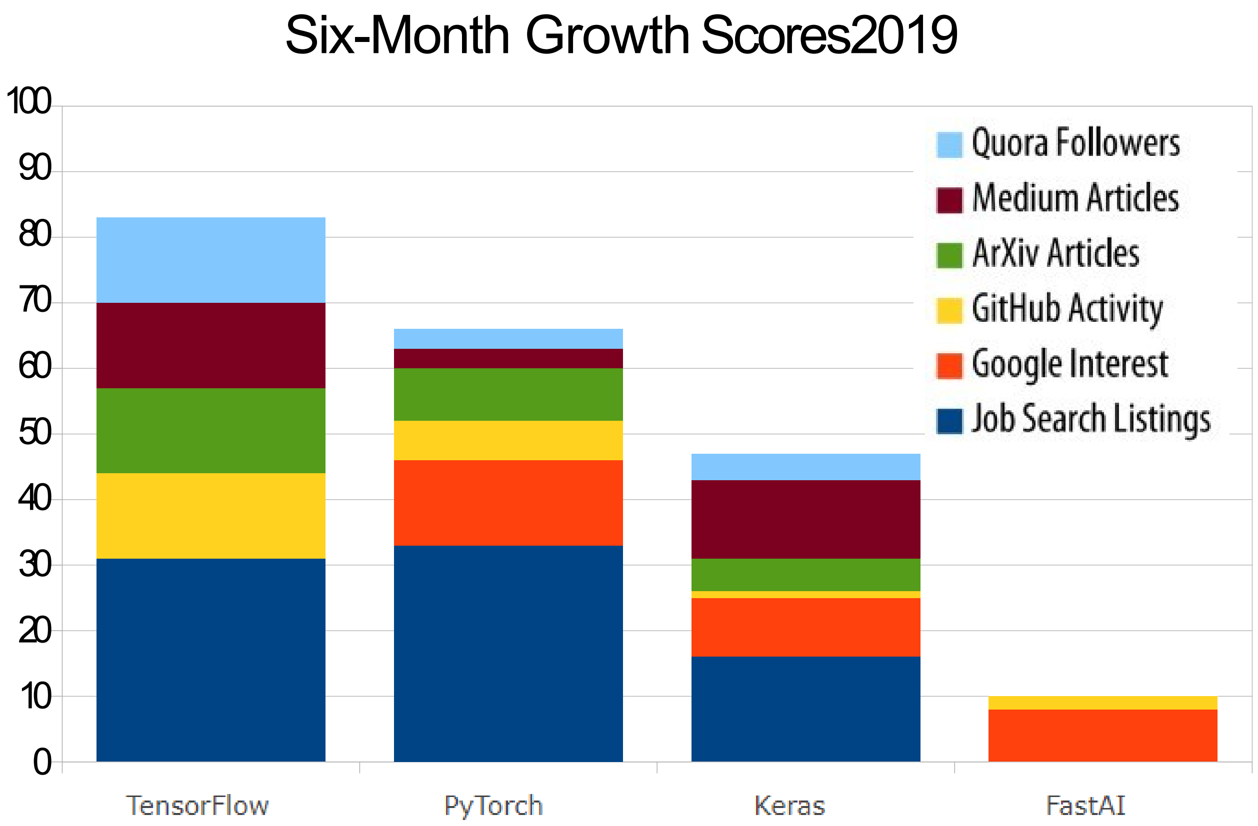
By François Cholet (Google)
Hight level API
Part on TensorFlow since 2017
MIT licence

PYTORCH

From Torch library
Supported by Facebook
BSD licence

Source: Jeff Hale, « Deep Learning Framework Power Scores 2018 » [DLPW]

APython centered world



TensorFlow

PYTORCH

Keras

fast.ai

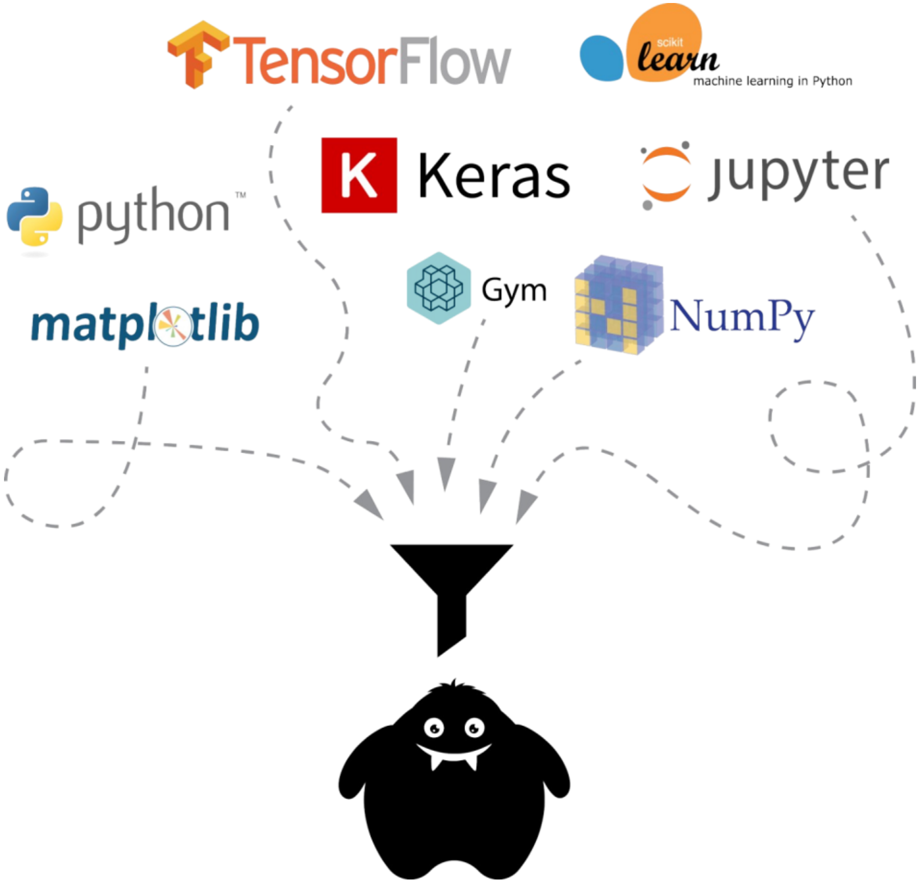
Source: Jeff Hale, «Deep Learning Framework Growth Scores2019 »[DLPW]

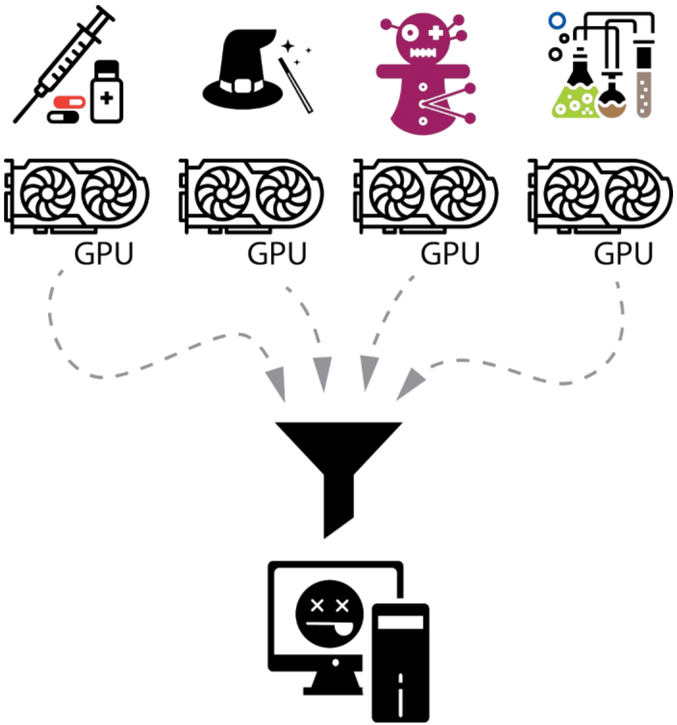
WhyDLisuseful?

- Manually designed features are often over-specified, incomplete and take a long time to design and validate !!
- Learned Features are easy to adapt, fast to learn
- Deep learning provides a very flexible, (almost?) universal, learnable framework for representing world, visual and linguistic information.
- Can learn both unsupervised and supervised
- Effective end-to-end joint system learning
- Useful when we have large amounts of training data

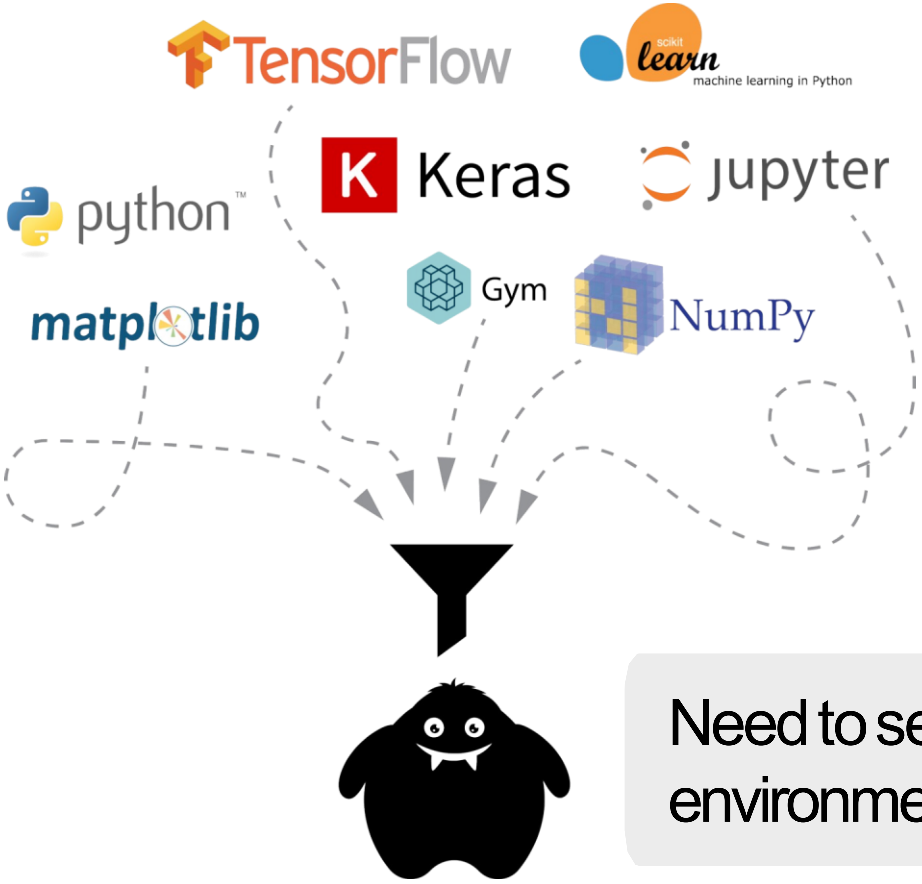
Source: Jeff Hale, «Deep Learning Framework Growth Scores 2019 » [DLPW]

Acertain complexity...

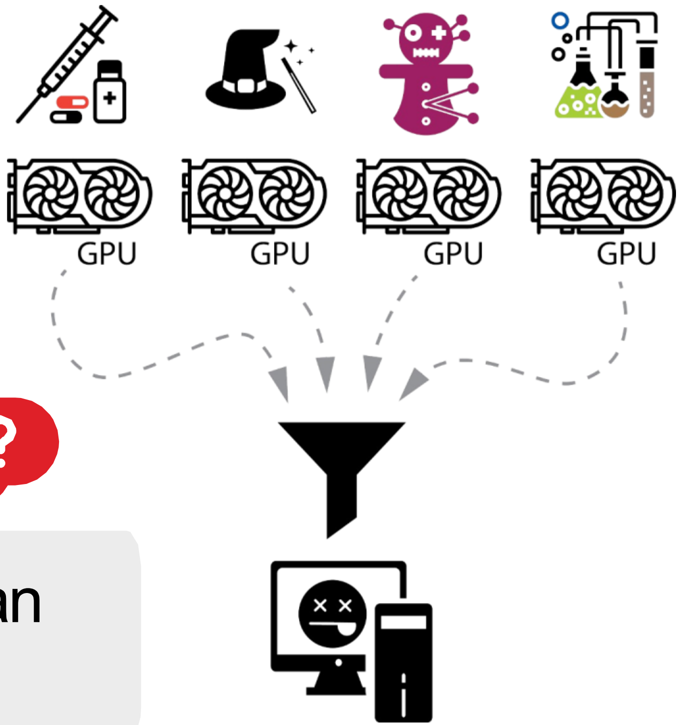


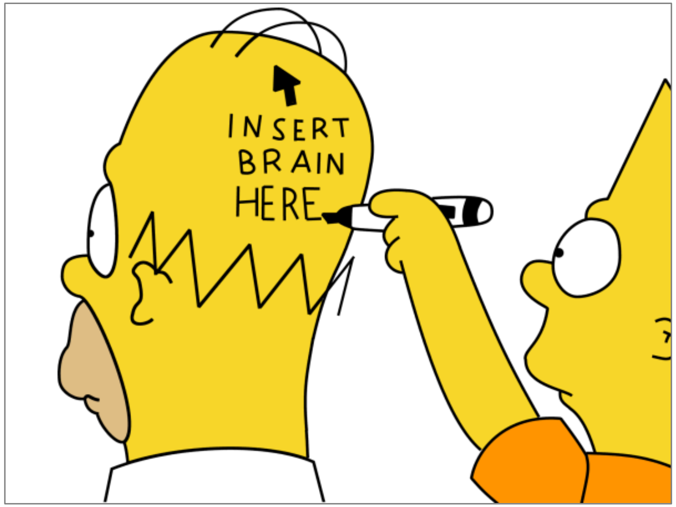


A certain complexity...



Need to set up an environment



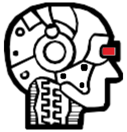


Fine, but Deep Learning What's that?

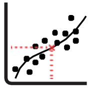
Roadmap

Episode :S01E01

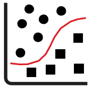
1



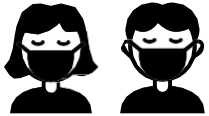
History, Fundamental Concepts



Basic Regression DNN



Basic Classification DNN



1.1 Introduction
Context, tools and ressources

1.2 **From the linear regression to the first neuron**

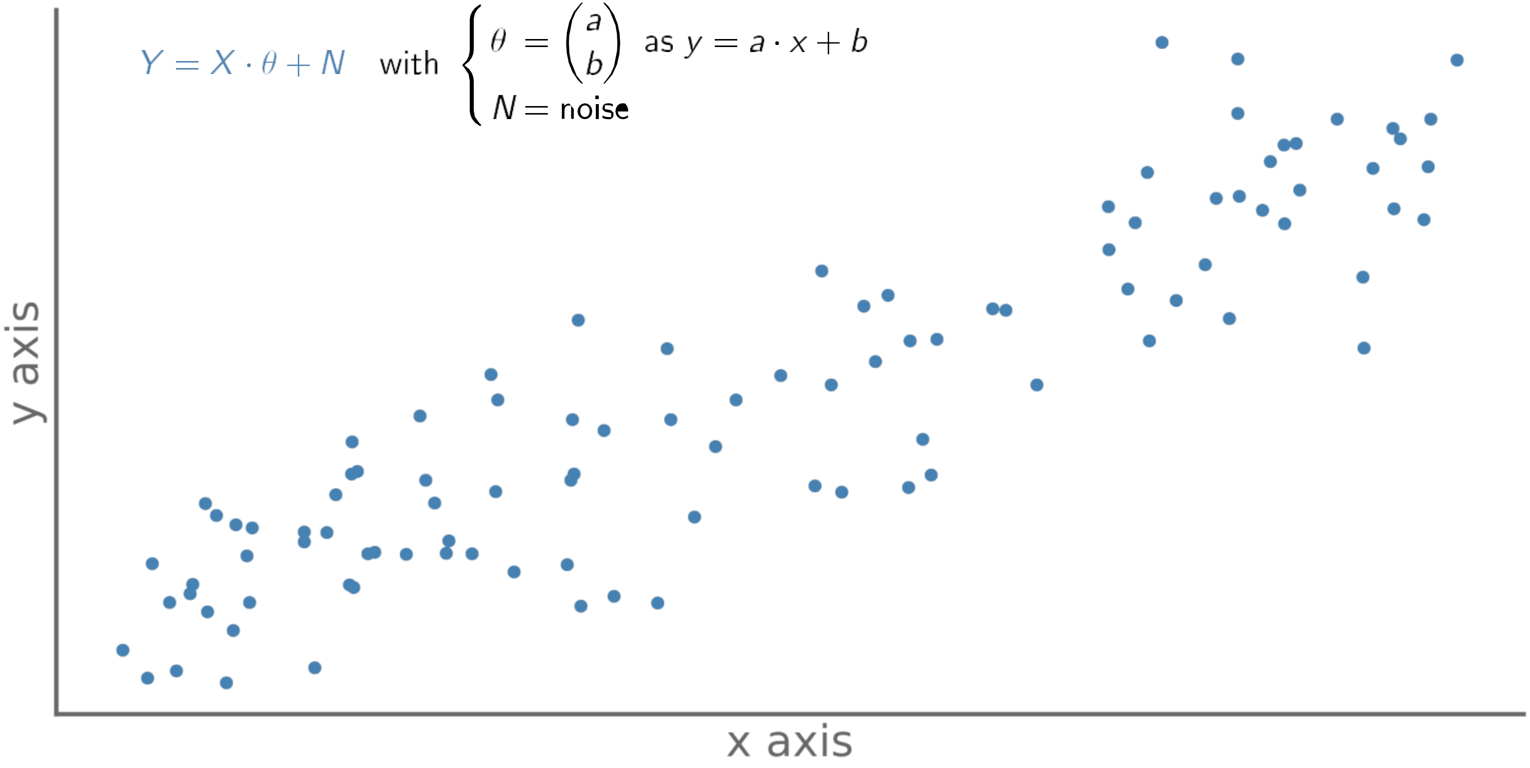
1.3 Neurons incontroversy

1.4 Data andneurons
- Basic Regression
- BasicClassification

Linear regression

We have a phenomenon, for which we have observations

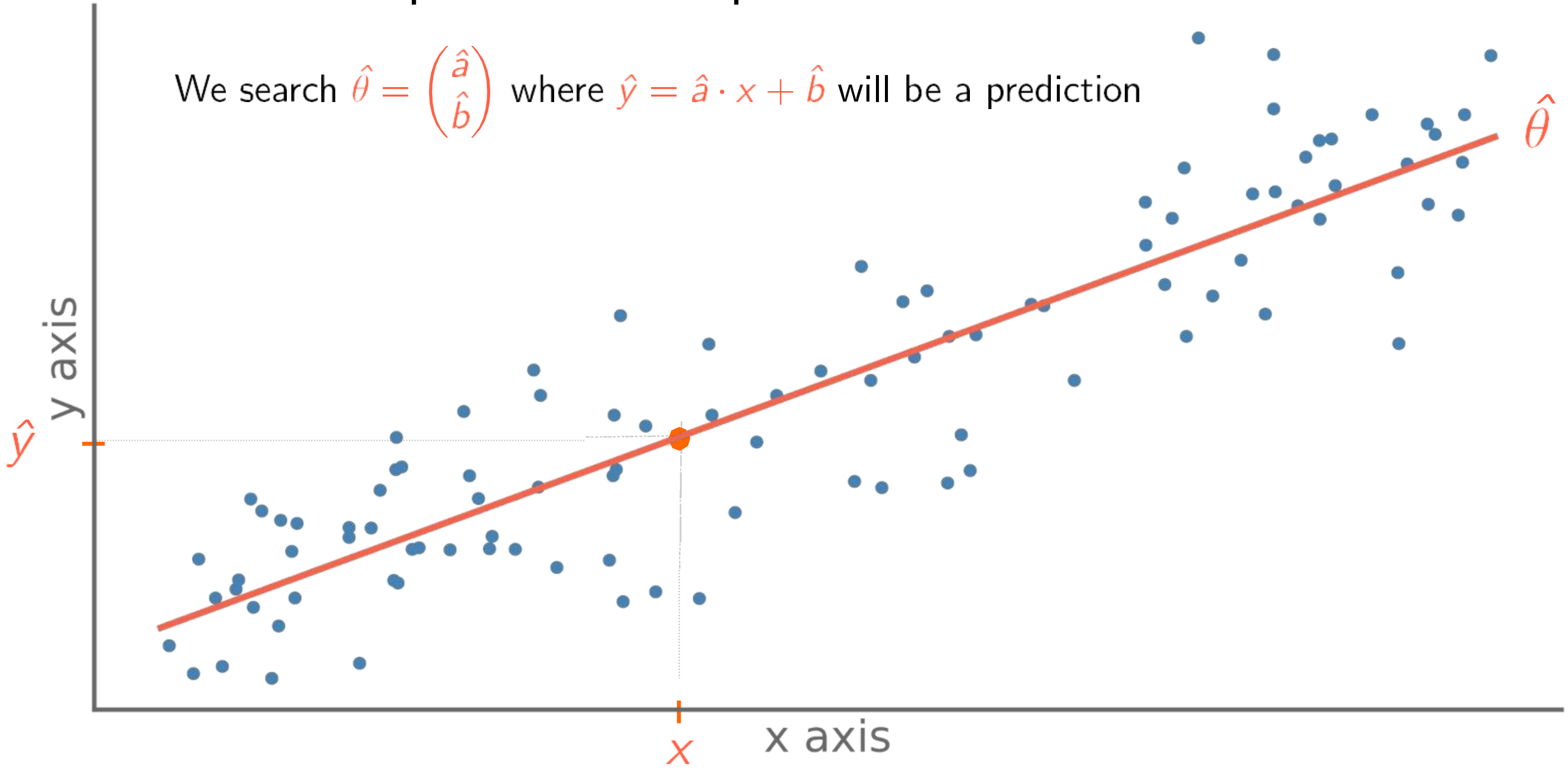
$$Y = X \cdot \theta + N \quad \text{with} \quad \begin{cases} \theta = \begin{pmatrix} a \\ b \end{pmatrix} \text{ as } y = a \cdot x + b \\ N = \text{noise} \end{cases}$$



Linear regression

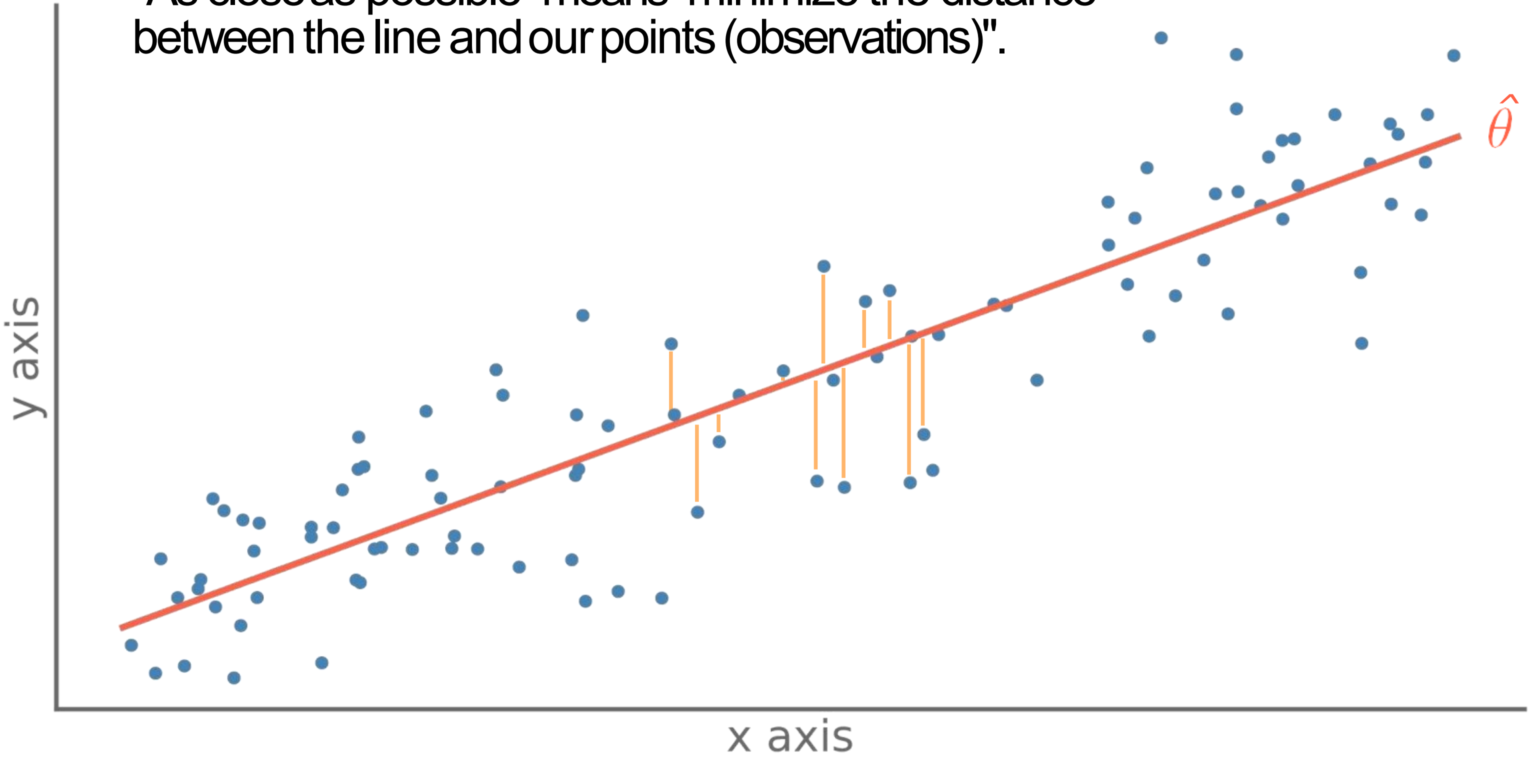
We are looking for a straight line that passes «as close as possible » to our points.

We search $\hat{\theta} = \begin{pmatrix} \hat{a} \\ \hat{b} \end{pmatrix}$ where $\hat{y} = \hat{a} \cdot x + \hat{b}$ will be a prediction



Linear regression

"As close as possible" means "minimize the distance between the line and our points (observations)".

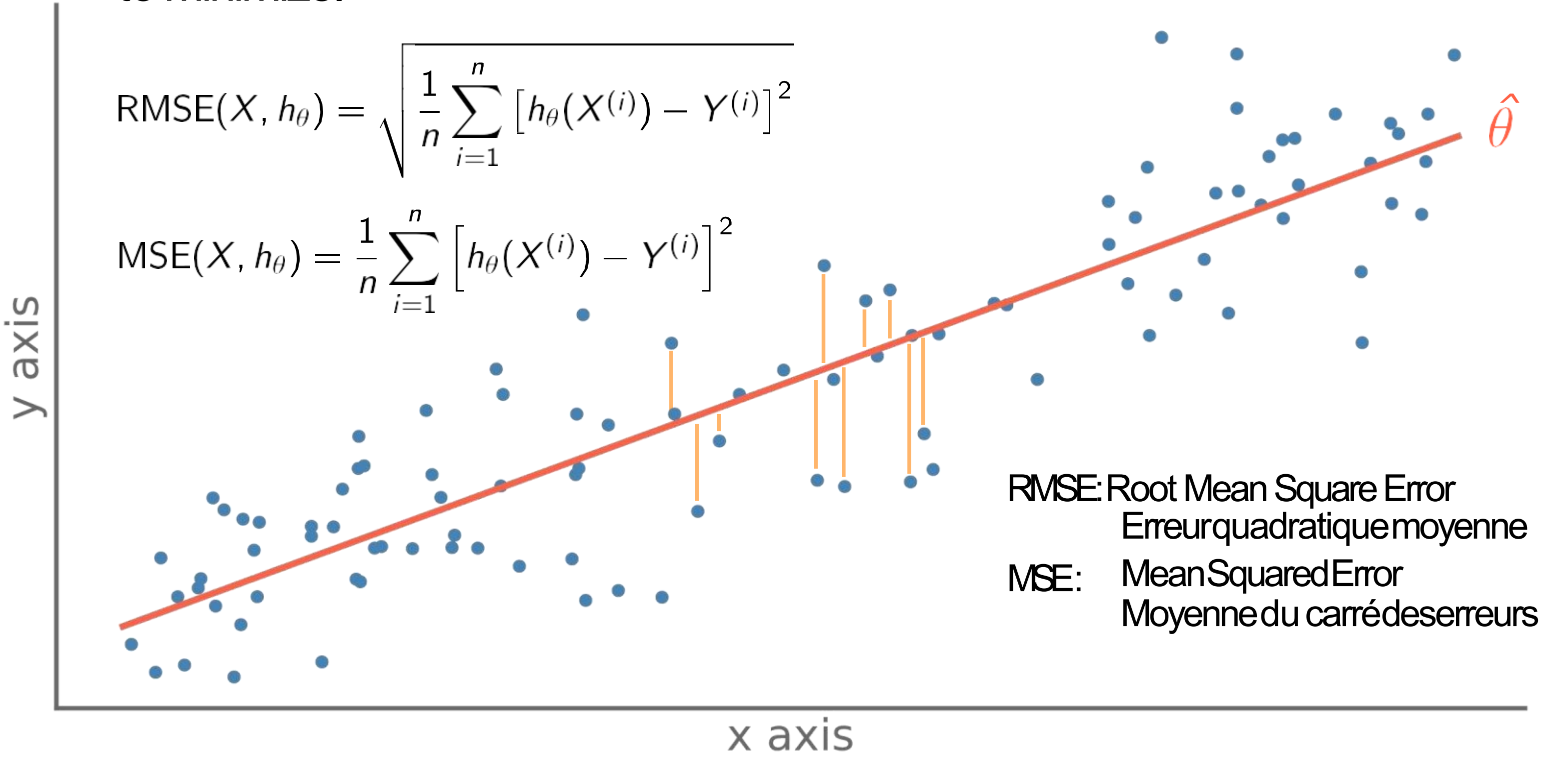


Linear regression

For this, we will use a «loss function », which we will try to minimize.

$$\mathrm{RMSE}(X, h_\theta) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left[h_\theta(X^{(i)}) - Y^{(i)}\right]^2}$$

$$\mathrm{MSE}(X, h_\theta) = \frac{1}{n}\sum_{i=1}^{n}\left[h_\theta(X^{(i)}) - Y^{(i)}\right]^2$$



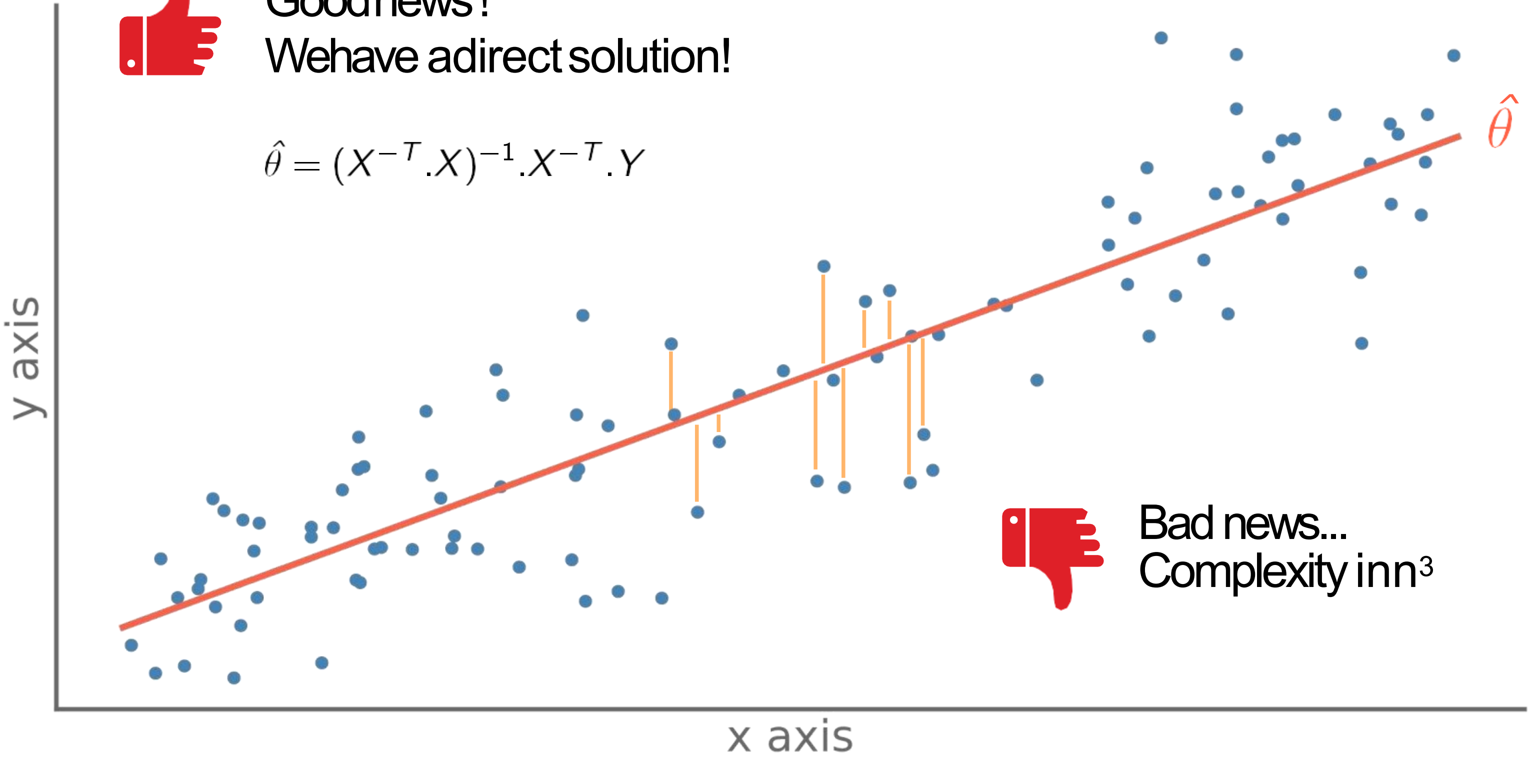
RMSE: Root Mean Square Error
Erreur quadratique moyenne

MSE: Mean Squared Error
Moyenne du carré des erreurs

Linear regression

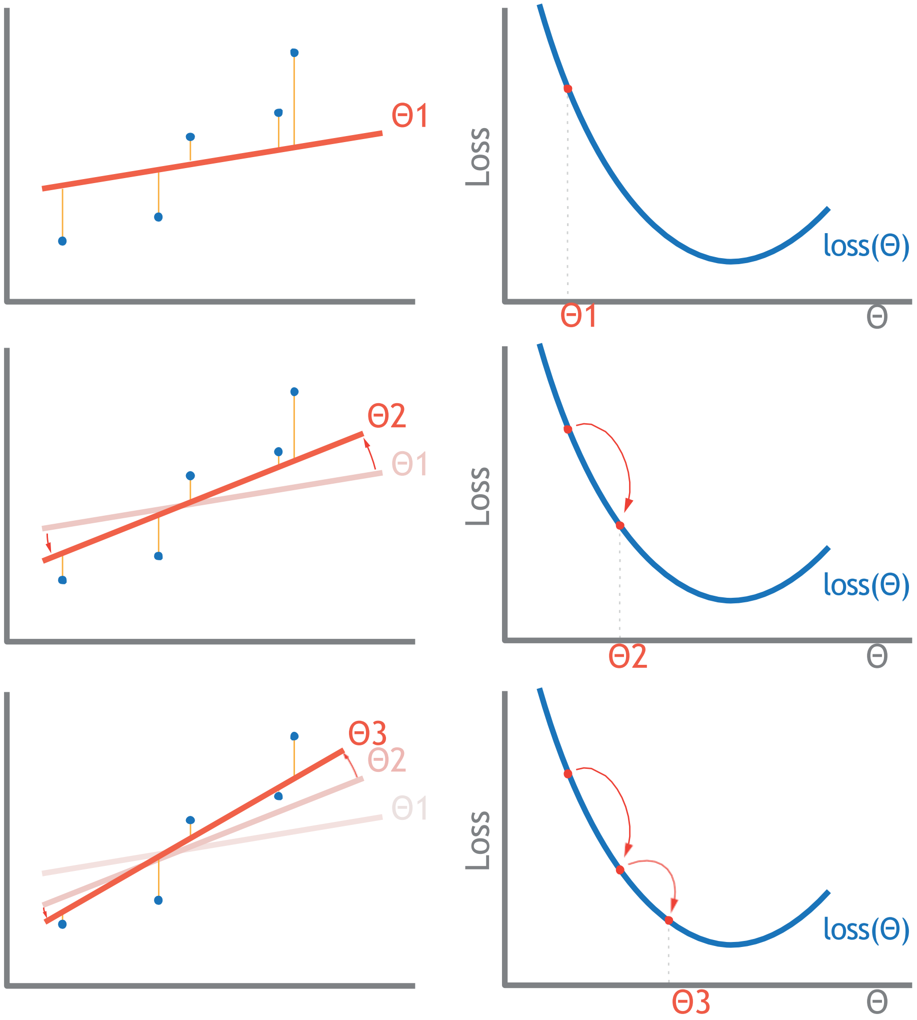
Good news !
We have a direct solution!

$$\hat{\theta} = (X^{-T}.X)^{-1}.X^{-T}.Y$$



Bad news...
Complexity in n^3

Gradient descent



We will iteratively look for the best position of our line, by varying its parameters(Θ).

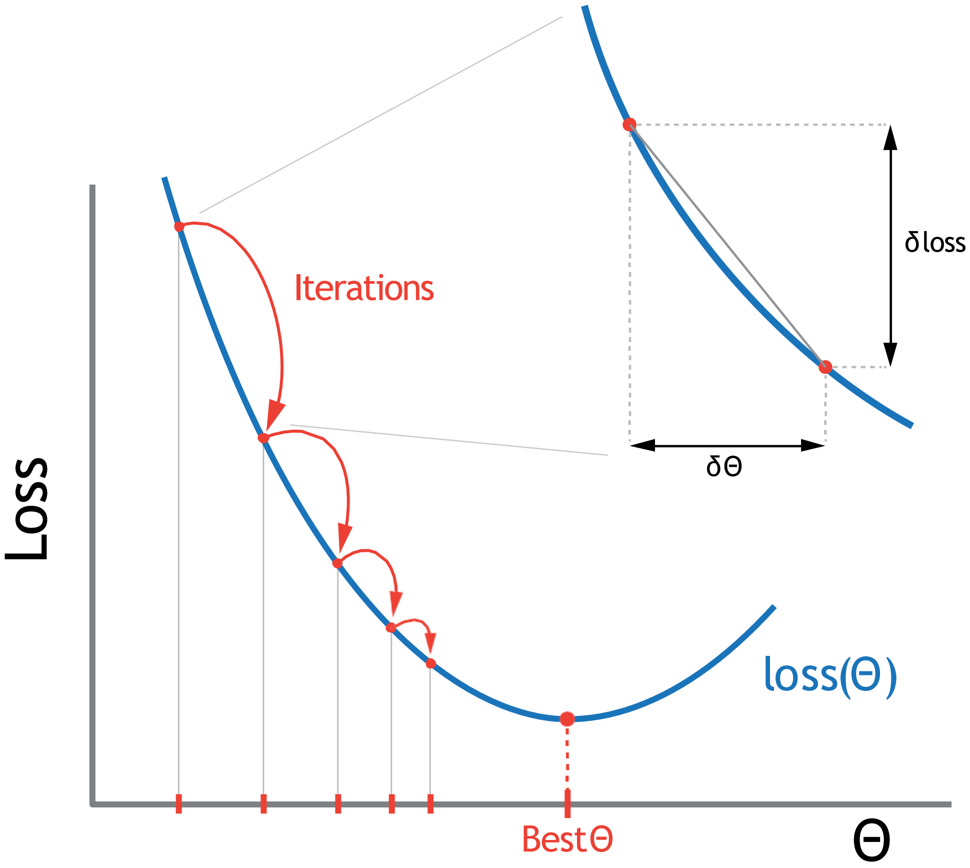
But how can we efficiently vary our parameters(Θ)?

Note: Loss functions could be:

$$\mathrm{RMSE}(X, h_\theta) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left[h_\theta(X^{(i)}) - Y^{(i)}\right]^2}$$

$$\mathrm{MSE}(X, h_\theta) = \frac{1}{n}\sum_{i=1}^{n}\left[h_\theta(X^{(i)}) - Y^{(i)}\right]^2$$

Gradient descent



By changing Θ from δΘ We improve loss(Θ) of δloss

The gradient is the slope we will follow to minimize our loss function.

$$\text{gradient} = \frac{\delta\text{loss}}{\delta\theta}$$

One iterative solution is : $\theta \leftarrow \theta - \eta \cdot \dfrac{\delta\text{loss}}{\delta\theta}$

where η is the learning rate

This process is called gradient descent and the function used to optimize the descent, optimization function

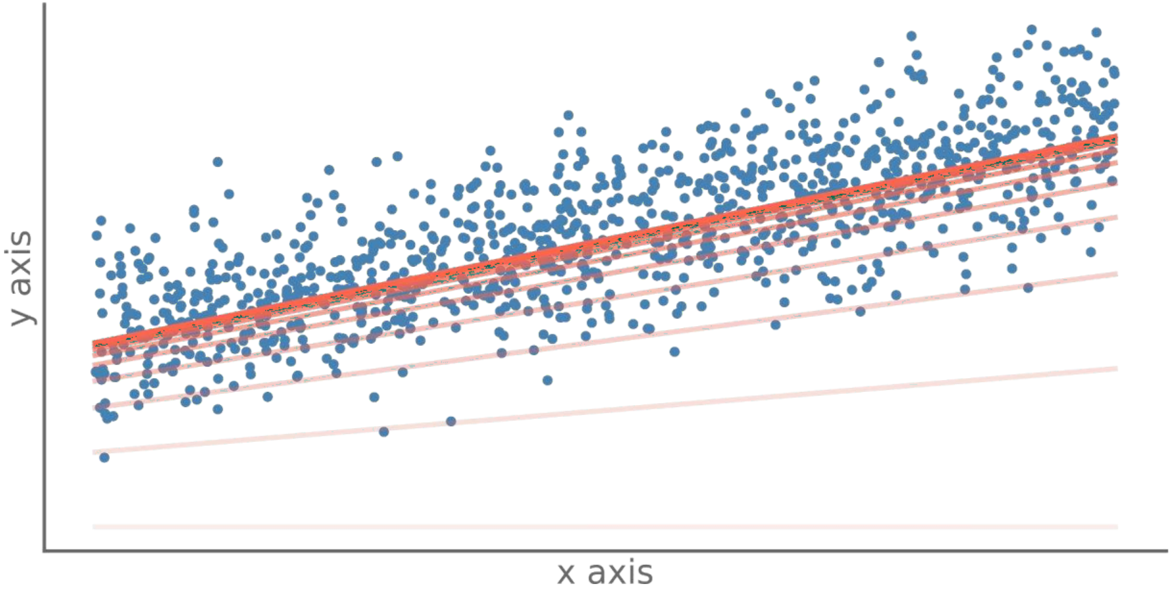
Gradient descent

$$MSE(X, h_\theta) = \frac{1}{n}\sum_{i=1}^{n}\left[h_\theta(X^{(i)}) - Y^{(i)}\right]^2$$

$$\nabla_\theta MSE(\Theta) = \begin{bmatrix} \frac{\partial}{\partial \theta_0} MSE(\Theta) \\ \frac{\partial}{\partial \theta_1} MSE(\Theta) \\ \vdots \\ \frac{\partial}{\partial \theta_n} MSE(\Theta) \end{bmatrix} = \frac{2}{m} X^T \cdot (X \cdot \Theta - Y)$$

Iterative solution is : $\Theta \leftarrow \Theta - \eta \cdot \nabla_\theta MSE(\Theta)$

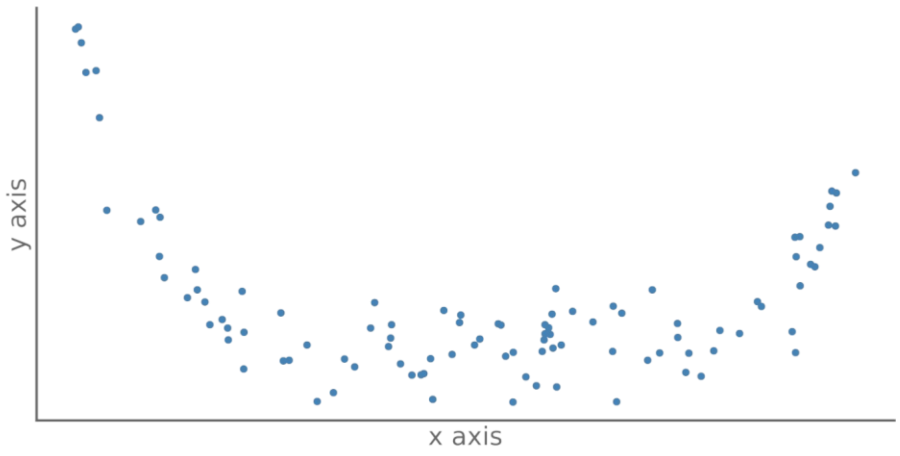
where η is the learning rate

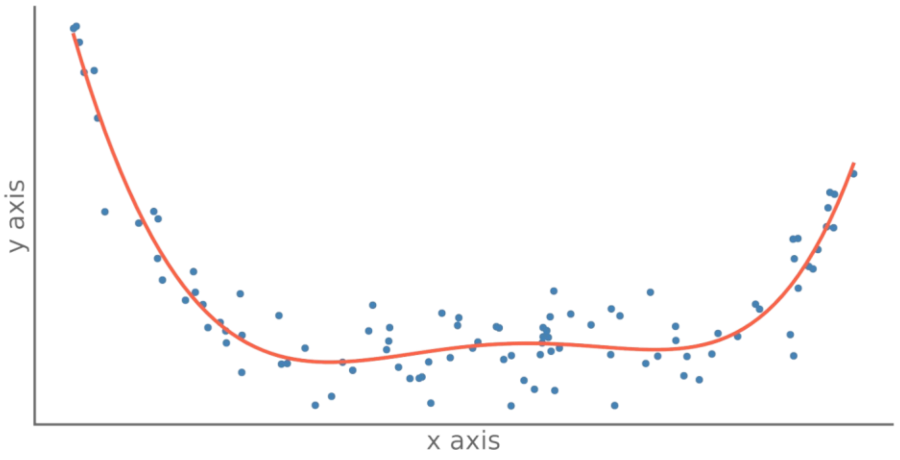


#i	Loss	Gradient		Theta	
0	+12.481	-6.777	-1.732	-3.388	+0.000
20	+4.653	-4.066	-1.039	-2.033	+0.346
40	+1.835	-2.440	-0.624	-1.220	+0.554
60	+0.821	-1.464	-0.374	-0.732	+0.679
80	+0.455	-0.878	-0.224	-0.439	+0.754
100	+0.324	-0.527	-0.135	-0.263	+0.799
120	+0.277	-0.316	-0.081	-0.158	+0.826
140	+0.260	-0.190	-0.048	-0.095	+0.842
160	+0.253	-0.114	-0.029	-0.057	+0.851
180	+0.251	-0.068	-0.017	-0.034	+0.857
200	+0.250	-0.041	-0.010	-0.020	+0.861

Notebook[GRAD1]

Polynomial regression

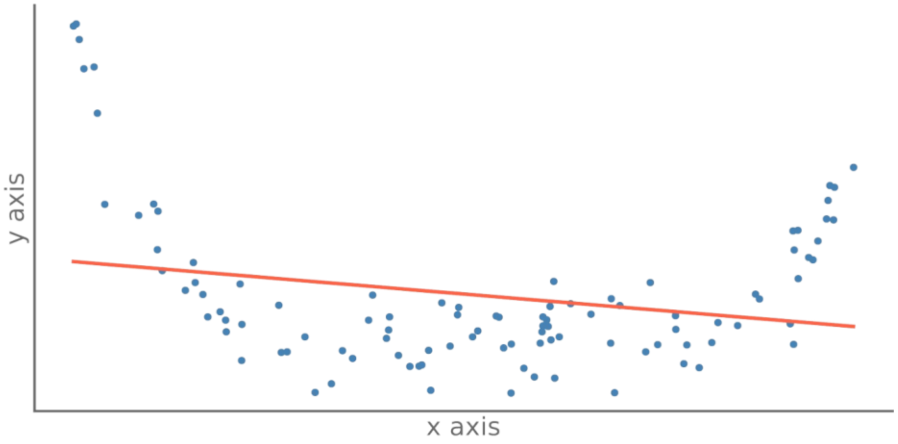




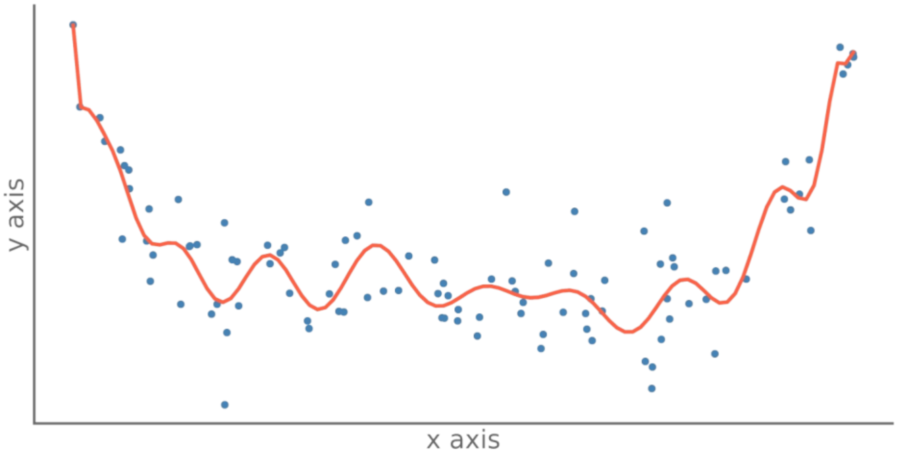
$$P_n(x) = a_0 + a_1 \cdot x + a_2 \cdot x^2 + \cdots + a_n \cdot x^n = \sum_{i=0}^{n} a_i \cdot x^n$$

Notebook[POLR1]

Polynomial regression



Underfiting



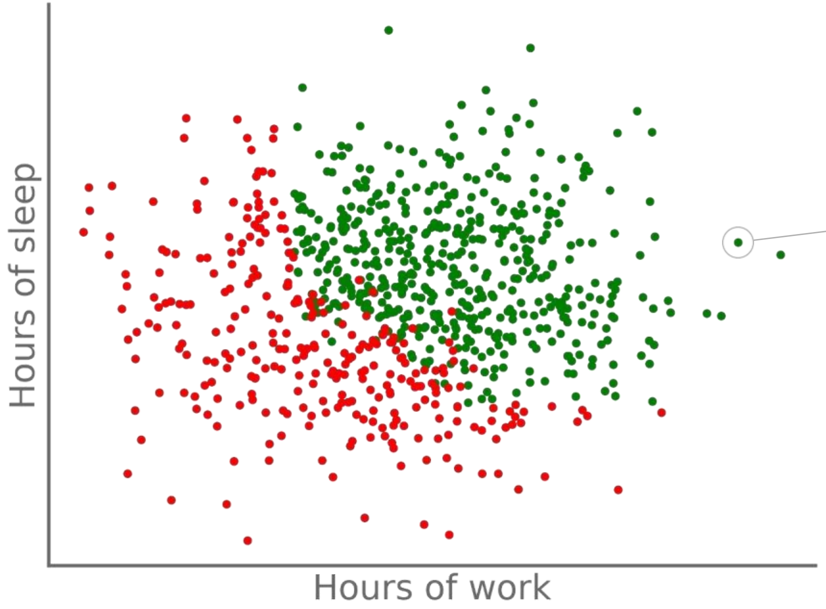
Overfiting

Notebook[POLR1]

Logistic regression

A logistic regression is intended to provide a probability of belonging to a class.

Dataset : X Observations
y Classe



$$(X_i, y_i) \begin{cases} X_i = \begin{pmatrix} x_{i1} = \text{Hours of work} \\ x_{i2} = \text{Hours of sleep} \end{pmatrix} \\ y_i = \begin{cases} 1 & \text{belong to the class} \\ 0 & \text{don't belong} \end{cases} \end{cases}$$

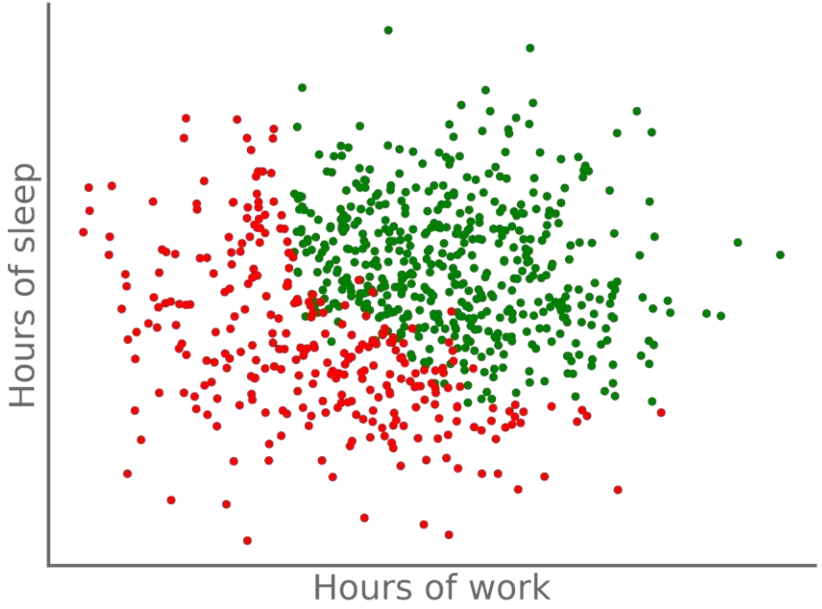
Notebook[LOGR1]

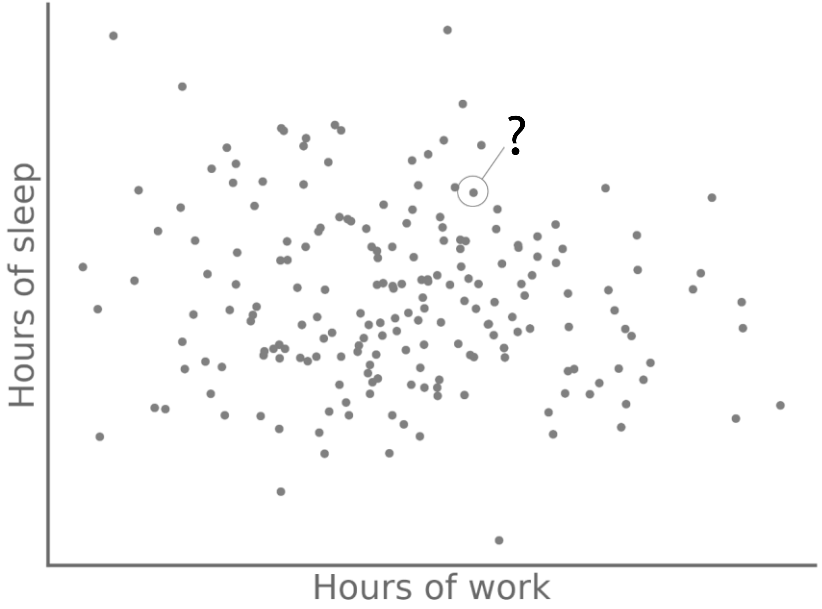
Logistic regression

A logistic regression is intended to provide a probability of belonging to a class.

Dataset : X Observations
y Classe

Objective : Predict the class
x given, we want to predict y
$y_{pred} = f(x)$
where f is a linear function

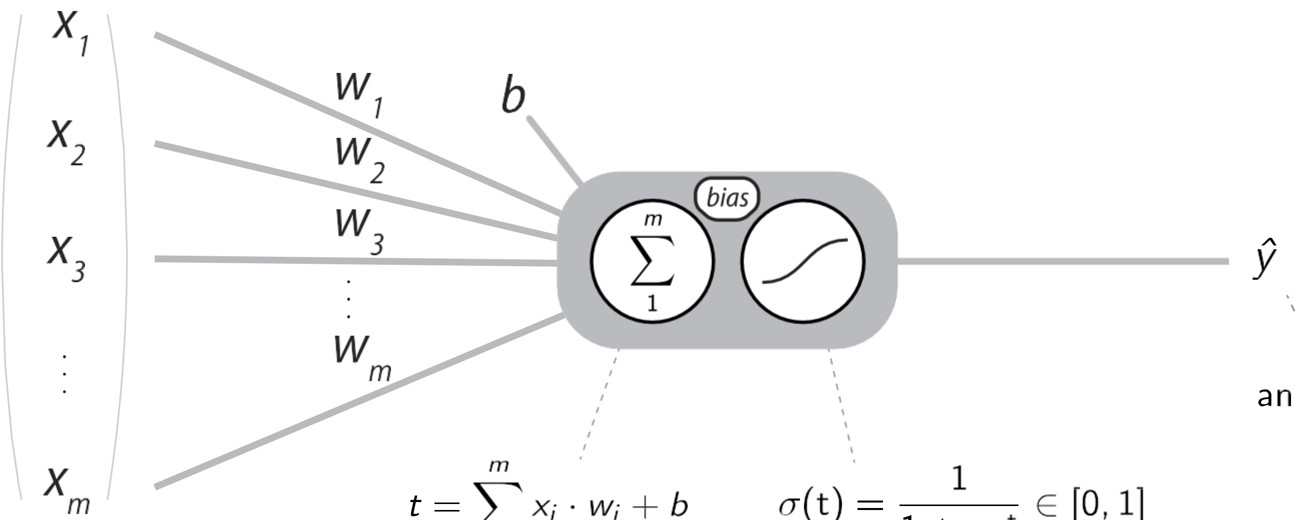




Notebook[LOGR1]

Logistic regression

$$\hat{y} = \sigma(\Theta^T \cdot X + b)$$



$$\text{and } \bar{y} = \begin{cases} 0 & \text{if } \hat{y} < 0.5 \\ 1 & \text{if } \hat{y} \geq 0.5 \end{cases}$$

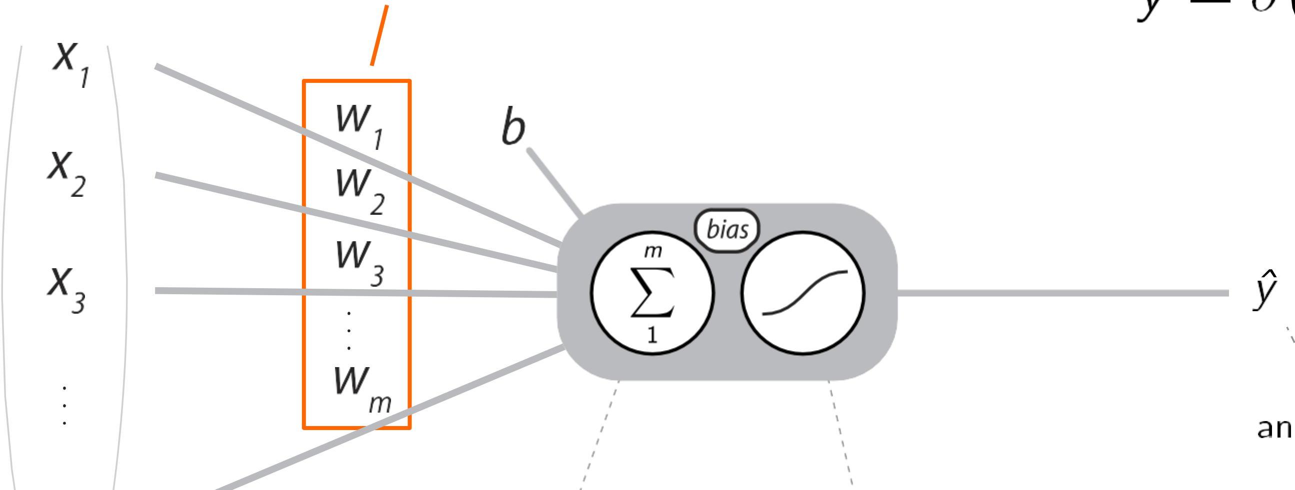
$$t = \sum_1^m x_i \cdot w_i + b \qquad \sigma(t) = \frac{1}{1 + e^{-t}} \in [0, 1]$$

Input	Bias / Weight	Activation function	Output
X	Θ	$\sigma(t)$	$\hat{y}$

Logistic regression

Determined by the minimisation of a cost function J(Θ)

$$\hat{y} = \sigma(\Theta^T \cdot X + b)$$



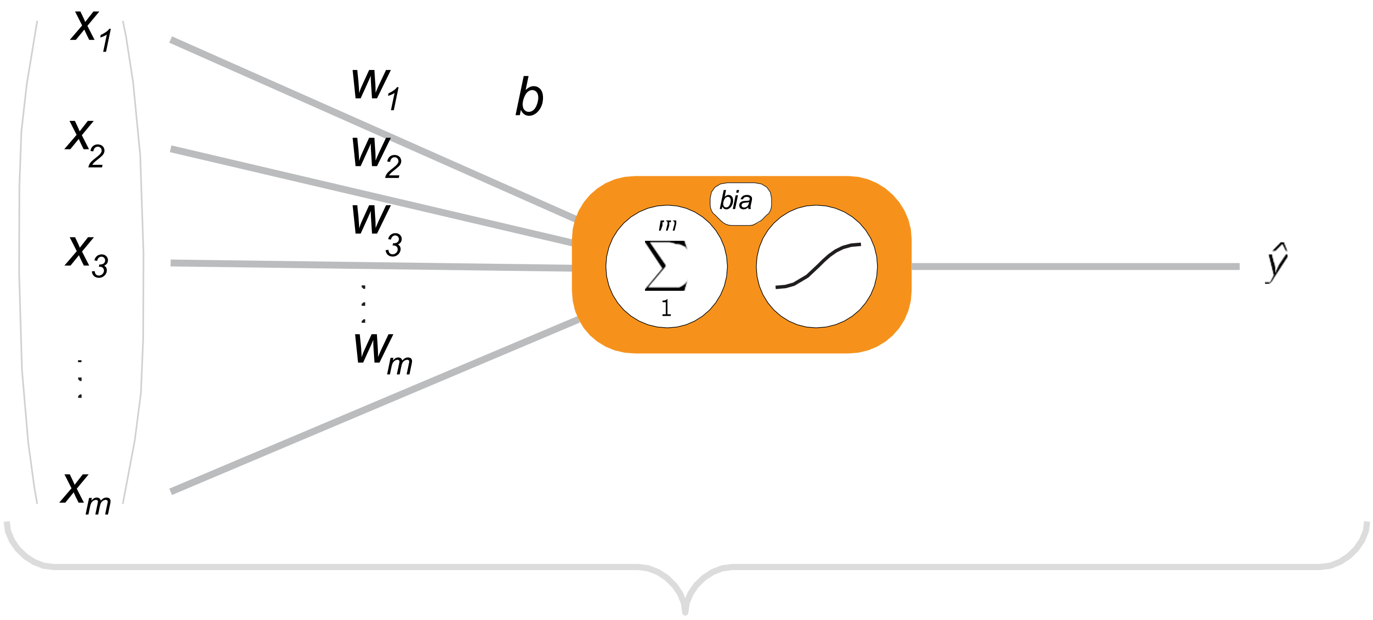
$$t = \sum_1^m x_i \cdot w_i + b \qquad \sigma(t) = \frac{1}{1 + e^{-t}} \in [0, 1]$$

$$\text{and } \bar{y} = \begin{cases} 0 & \text{if } \hat{y} < 0.5 \\ 1 & \text{if } \hat{y} \geq 0.5 \end{cases}$$

Input	**Bias / Weight**	**Activation function**	**Output**
X	Θ	$\sigma(t)$	$\hat{y}$

Logistic regression

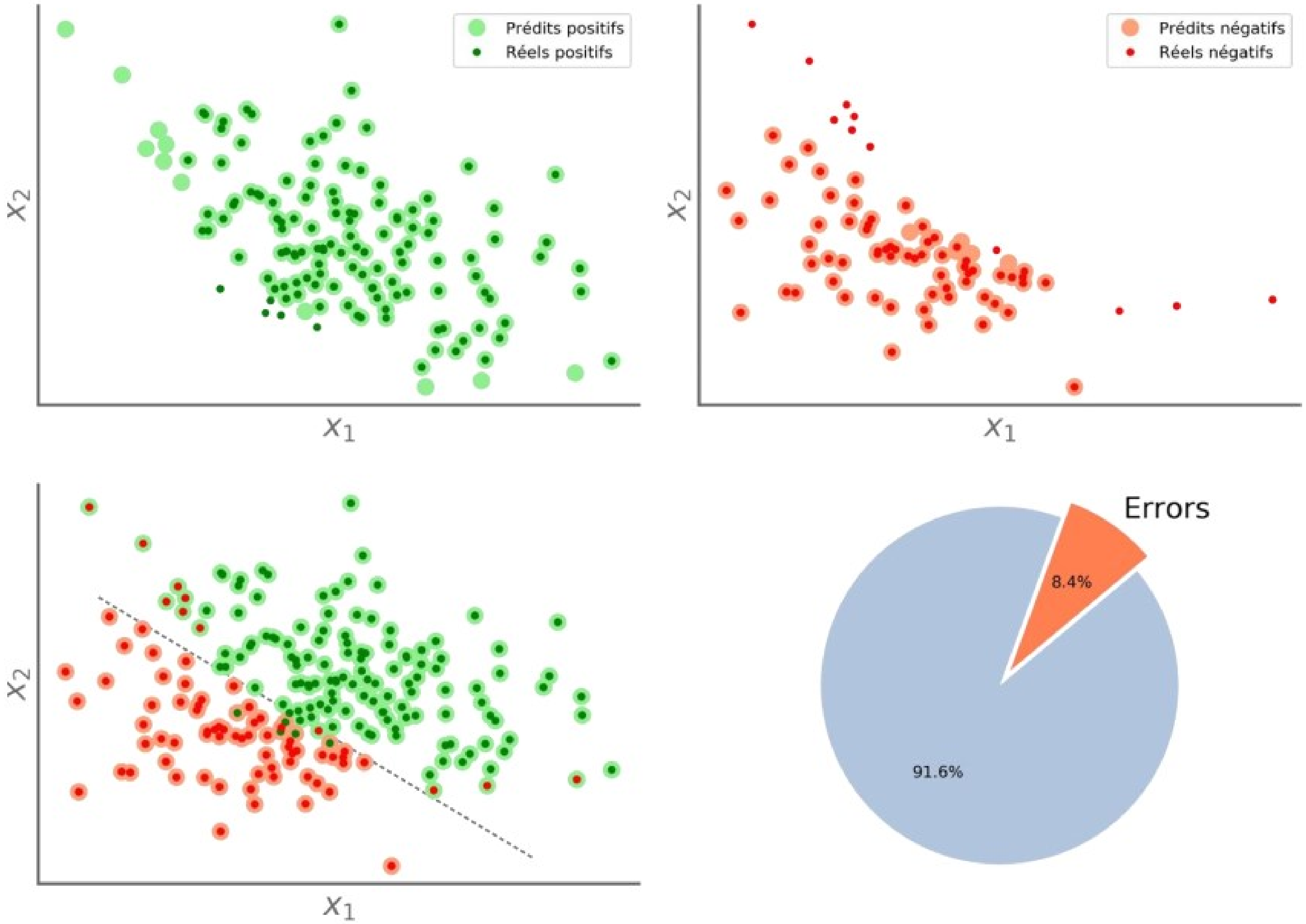
$$\hat{y} = \sigma(\Theta^T \cdot X + b)$$



That's an « artificial neuron »!

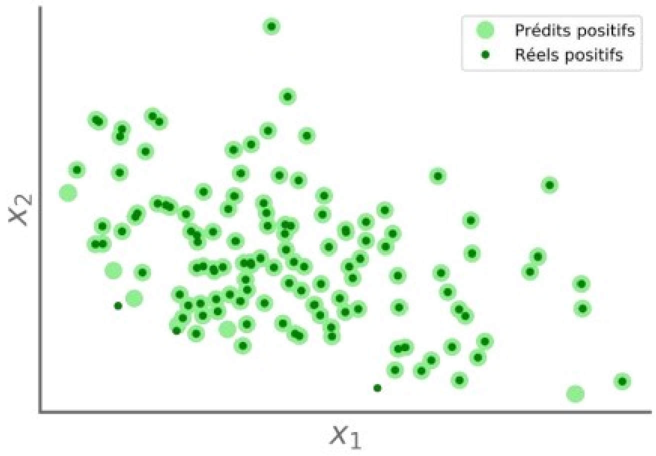
So, we have a neural network of… 1 neuron!

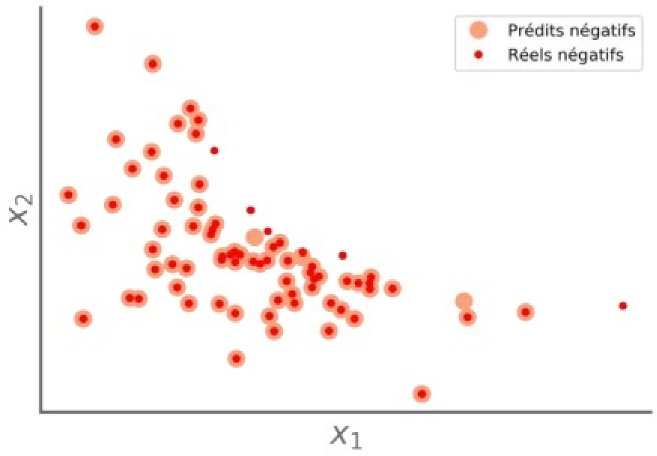
Logistic regression

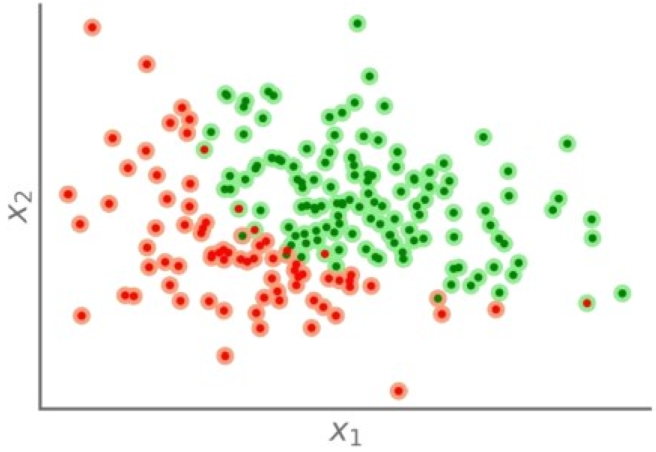


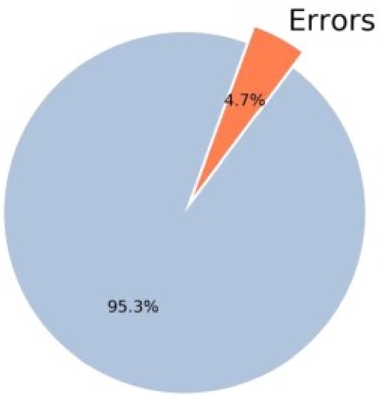
Notebook[LOGR1]

Logistic regression









Linear => Nonlinear

$\forall i \in [0, m]$, we add : $x_{i1}^2, x_{i2}^2, x_{i1}^3, x_{i2}^3$ to X_i

so, for :

$$X = \begin{bmatrix} 1 & x_{11} & x_{12} \\ \vdots & \dots & \\ 1 & x_{m1} & x_{m2} \end{bmatrix}$$

we have :

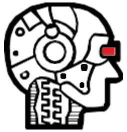
$$\mathring{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & x_{11}^2 & x_{12}^2 & x_{11}^3 & x_{12}^3 \\ \vdots & & & \dots & & & \\ 1 & x_{m1} & x_{m2} & x_{m1}^2 & x_{m2}^2 & x_{m1}^3 & x_{m2}^3 \end{bmatrix}$$

Notebook[LOGR1]

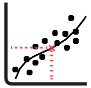
Roadmap

Episode :S01E01

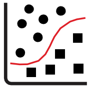
1



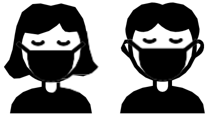
History, Fundamental Concepts



Basic Regression DNN



Basic Classification DNN



1.1 Introduction
Context, tools and ressources

1.2 From the linear regression to the first neuron

1.3 Neurons in controversy

1.4 Data and neurons
- Basic Regression
- Basic Classification

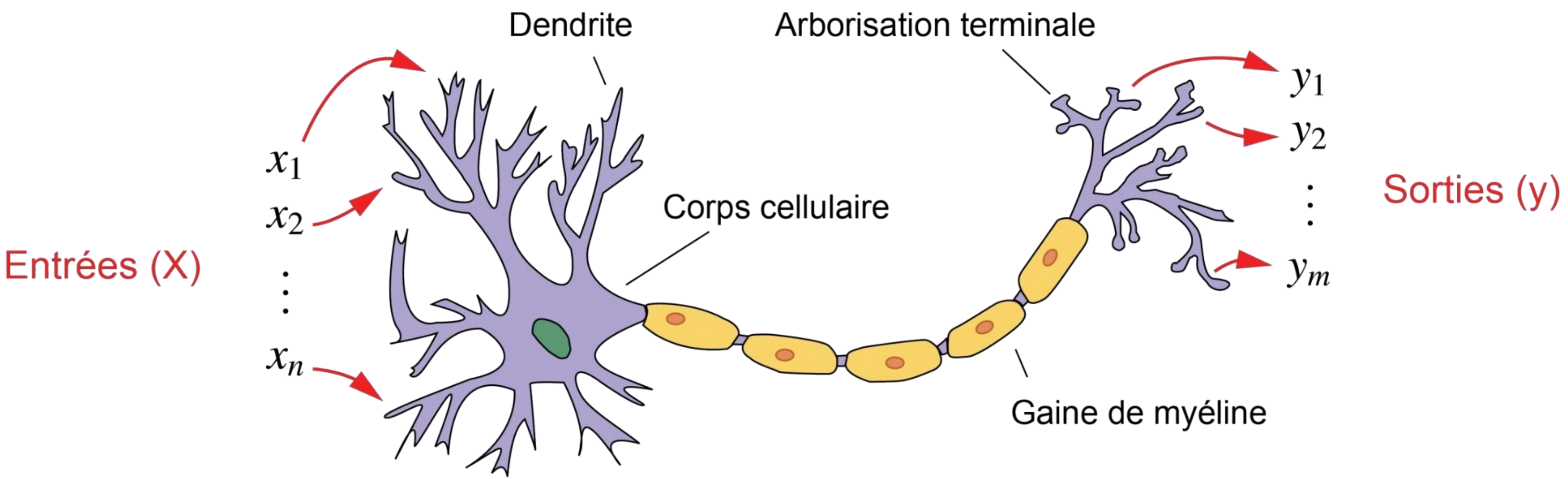
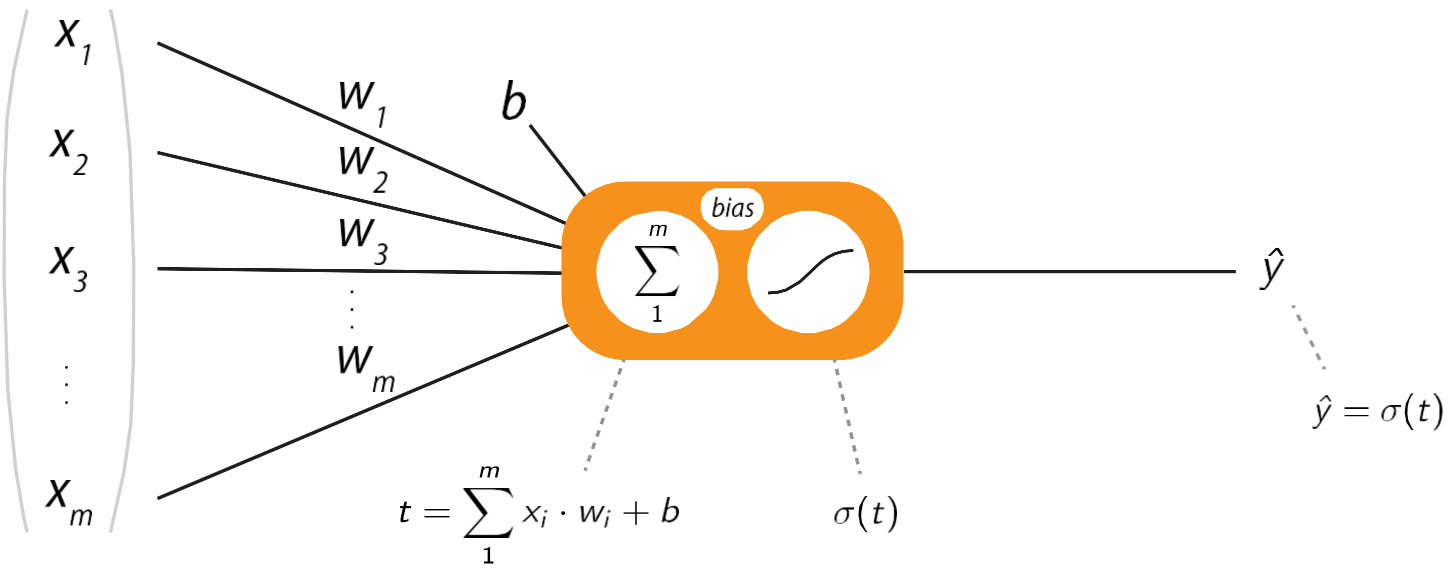


Illustration :[NEURON]

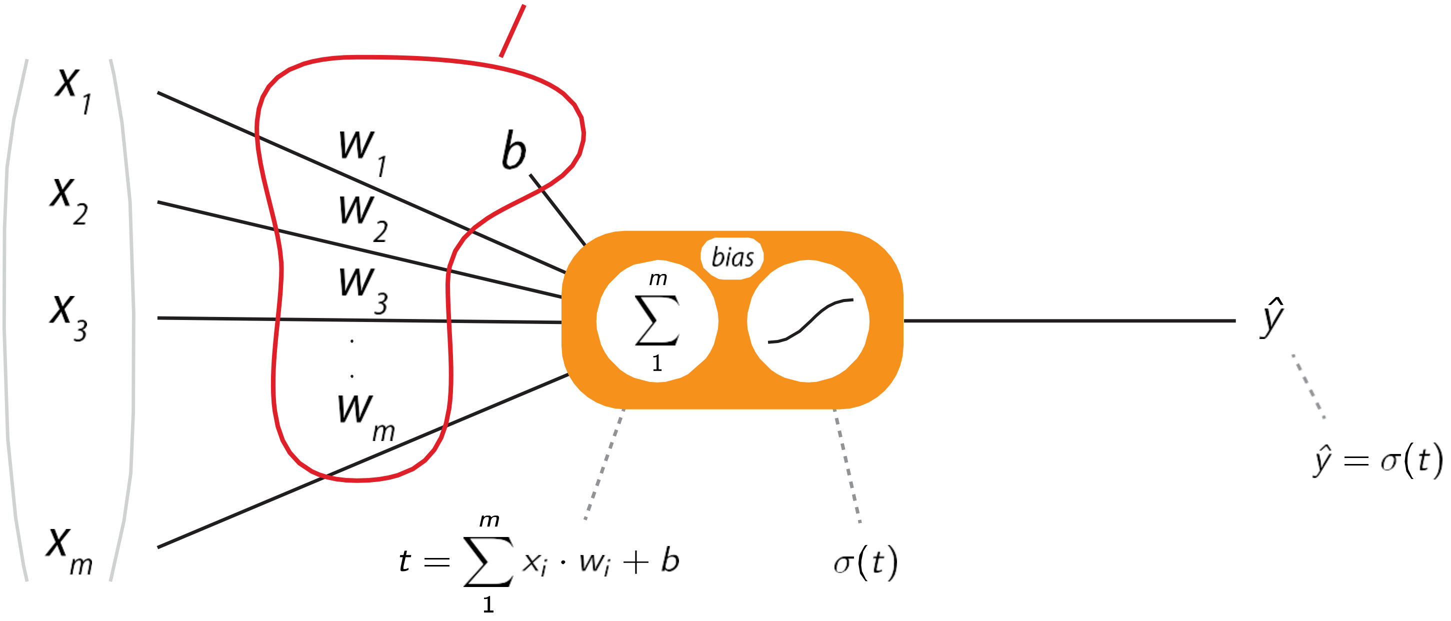
$$\hat{y} = \sigma(\Theta^T \cdot X)$$



Input X | **Bias / Weight** Θ, b | **Activation function** $\sigma(t)$ | **Output** $\hat{y}$

Determined by the minimisation of a cost function

$$\hat{y} = \sigma(\Theta^T \cdot X + b)$$



Input X

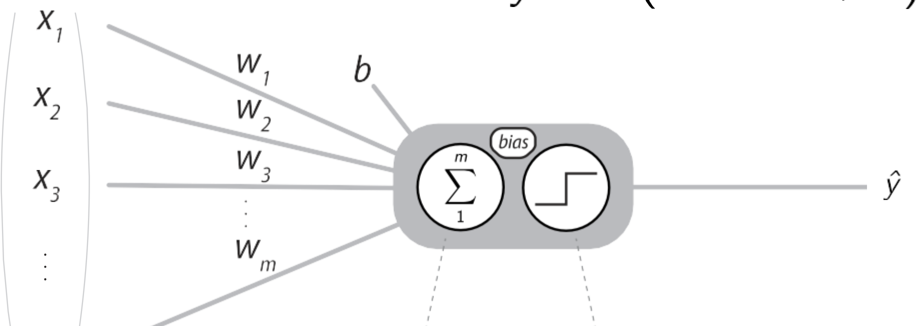
Bias / Weight Θ, b

Activation function $\sigma(t)$

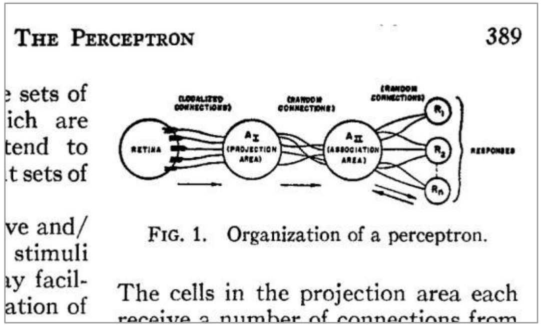
Output $\hat{y}$

Perceptron

$$\hat{y} = \sigma(\Theta^T \cdot X + b)$$



$$t = \sum_{1}^{m} x_i \cdot w_i + b \qquad \sigma(t) = \begin{cases} 1 & \text{if } t > 0 \\ 0 & \text{if } t \leq 0 \end{cases}$$



THE PERCEPTRON 389

e sets of
ich are
tend to
t sets of

ve and/
stimuli
y facil-
ation of

FIG. 1. Organization of a perceptron.

The cells in the projection area each
receive a number of connections from

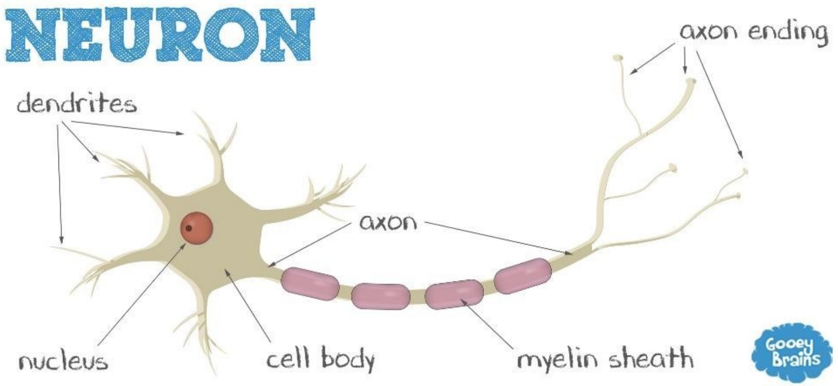
Perceptron
Frank Rosenblatt
1958

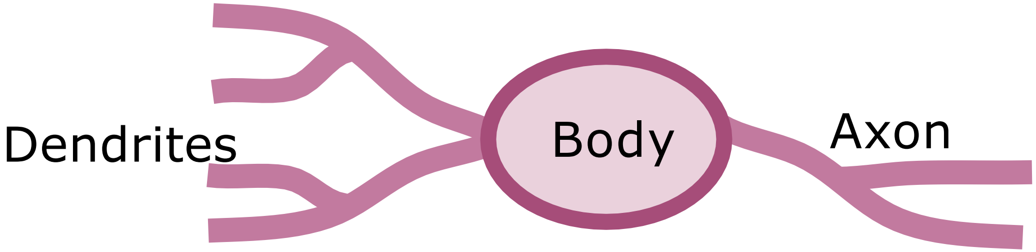
Linear and binary classifier

F.Rosenblatt, 1958[FROS]

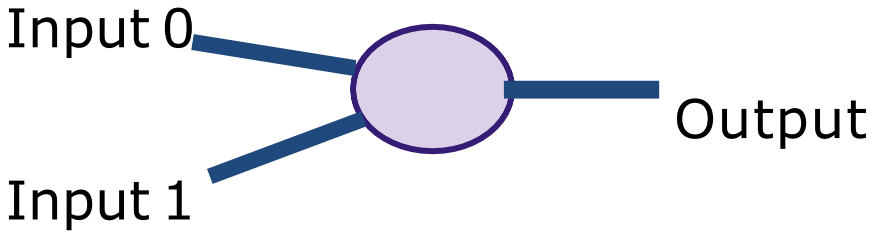
Neuron vs. perceptron

- The biological neuron:



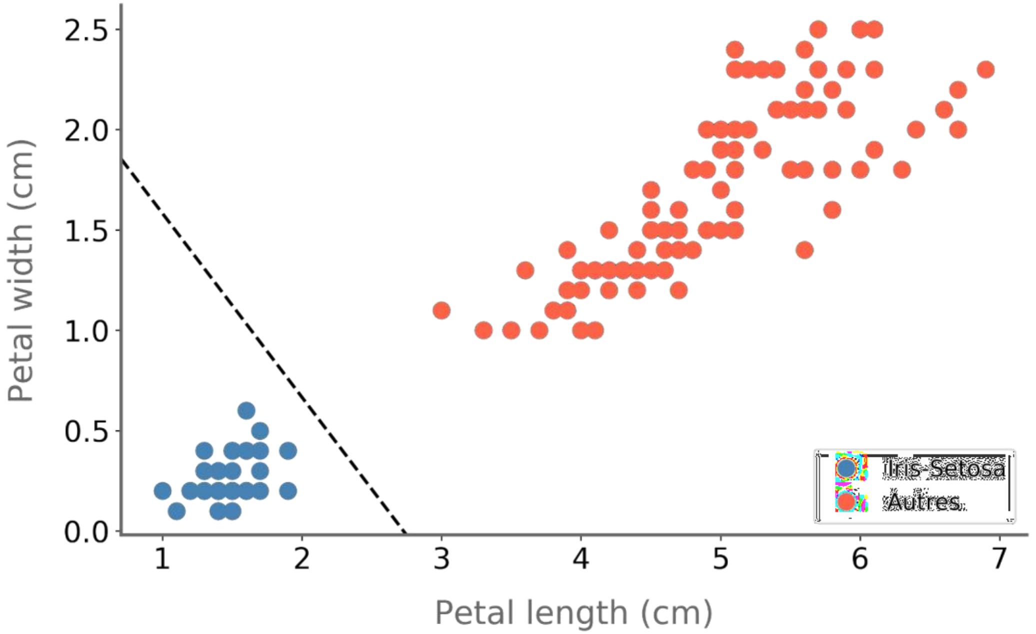


- The artificial neuron also has input and output!



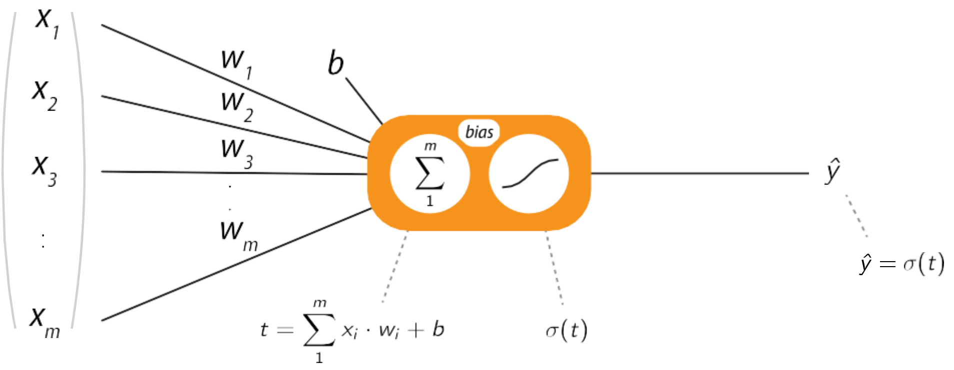
Iris plants dataset

Dataset from : Fisher, R.A. "The use of multiple measurements in taxonomic problems" Annual Eugenics, 7, Part II, 179-188 (1936)

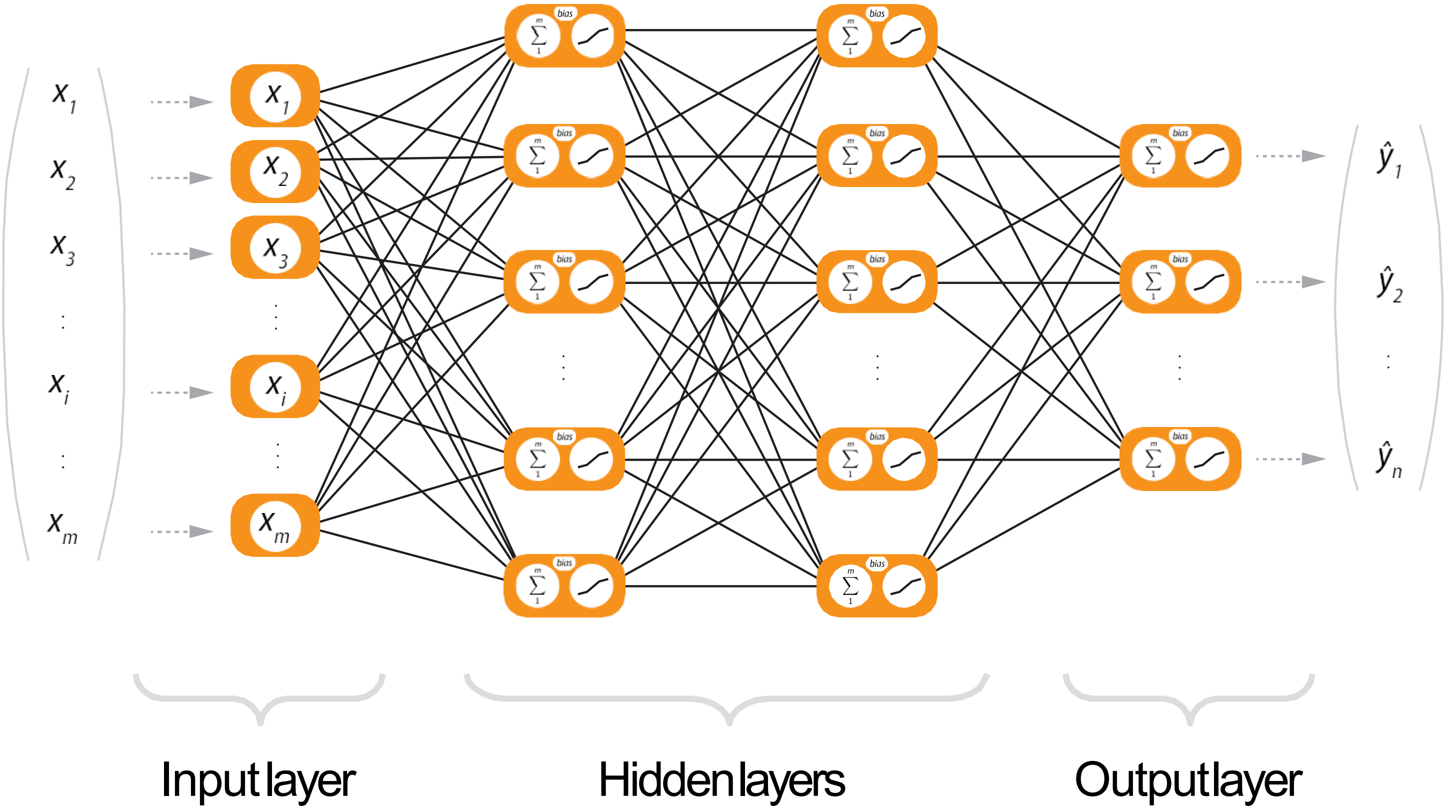


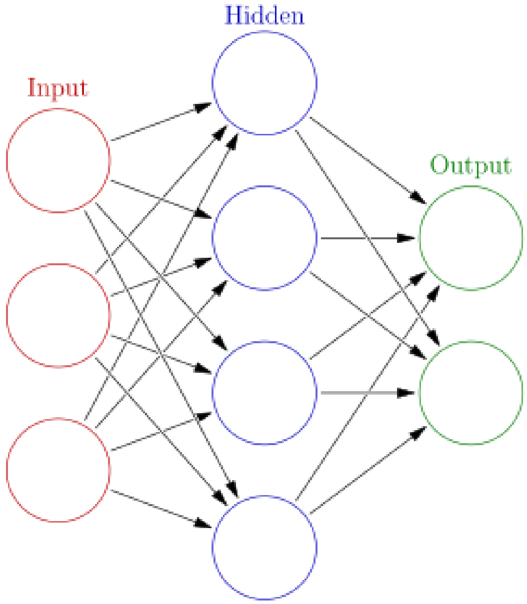
Length	Width	Iris Setosa (0/1)
x_1	x_2	y
1.4	1.4	1
1.6	1.6	1
1.4	1.4	1
1.5	1.5	1
1.4	1.4	1
4.7	4.7	0
4.5	4.5	0
4.9	4.9	0
4.0	4.0	0
4.6	4.6	0
(...)		

Notebook [PER57]


x_1
x_2
x_3
$\vdots$
x_m
w_1
w_2
w_3
w_m
b
bias
$\sum_1^m$
$\hat{y}$
$\hat{y} = \sigma(t)$
$t = \sum_1^m x_i \cdot w_i + b$
$\sigma(t)$

Deep Neural Networks



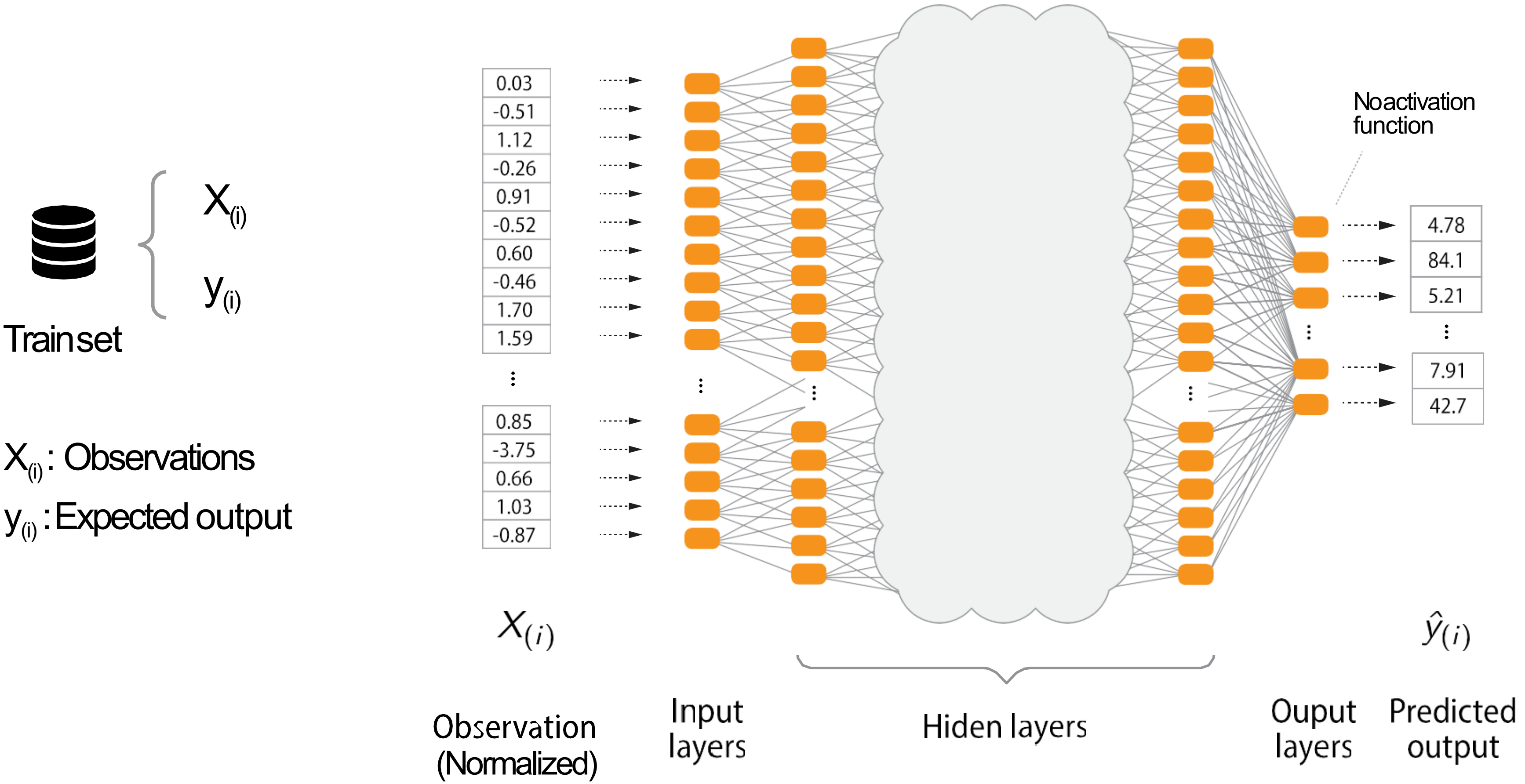


Keras Sequential Model

The Sequential model is a linear stack of layers.

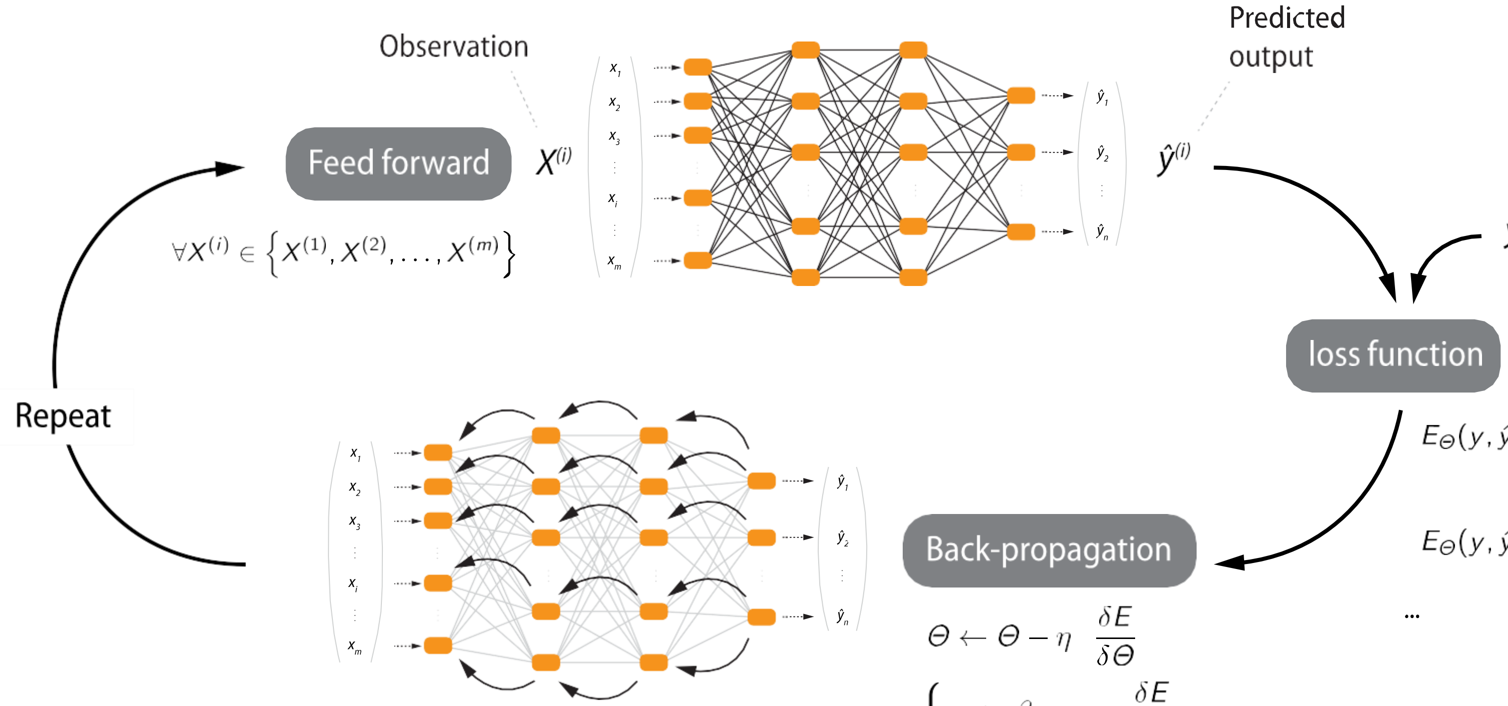
```
from keras.models import Sequential
from keras.layers import Dense, Activation

model = Sequential([
    Dense(4, input_shape=(3,), activation='relu'),
    Dense(2 activation='softmax'),
])
```

Deep Neural Networks



$X_{(i)}$: Observations

$y_{(i)}$: Expected output



$$E_\Theta(y,\hat{y}) = \frac{1}{2m}\sum_{i=1}^{m}\left\|y^{(i)} - \hat{y}^{(i)}\right\|^2$$

$$E_\Theta(y,\hat{y}) = -\sum_{i=1}^{m}\sum_{j=1}^{n} y_j^{(i)} \log(\hat{y}_j^{(i)})$$

...

$$\Theta \leftarrow \Theta - \eta \frac{\delta E}{\delta \Theta}$$

$$\begin{cases} m \leftarrow \beta m - \eta \dfrac{\delta E}{\delta \Theta} \\ \Theta \leftarrow \Theta + m \end{cases}$$

...

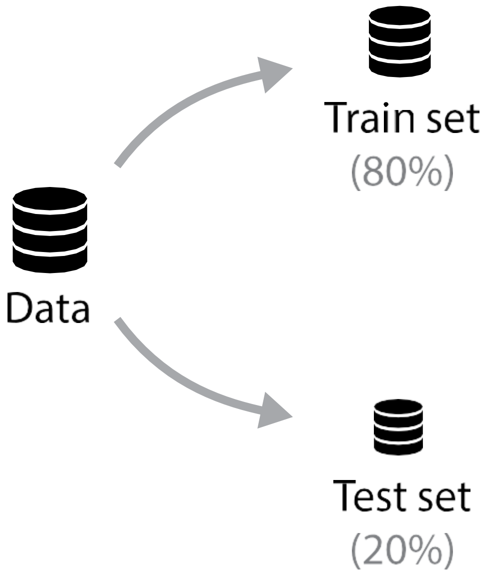
η : Learning rate
β : Momentum

To define :
- Optimization
- Activation
- Loss
- Metrics
- ...

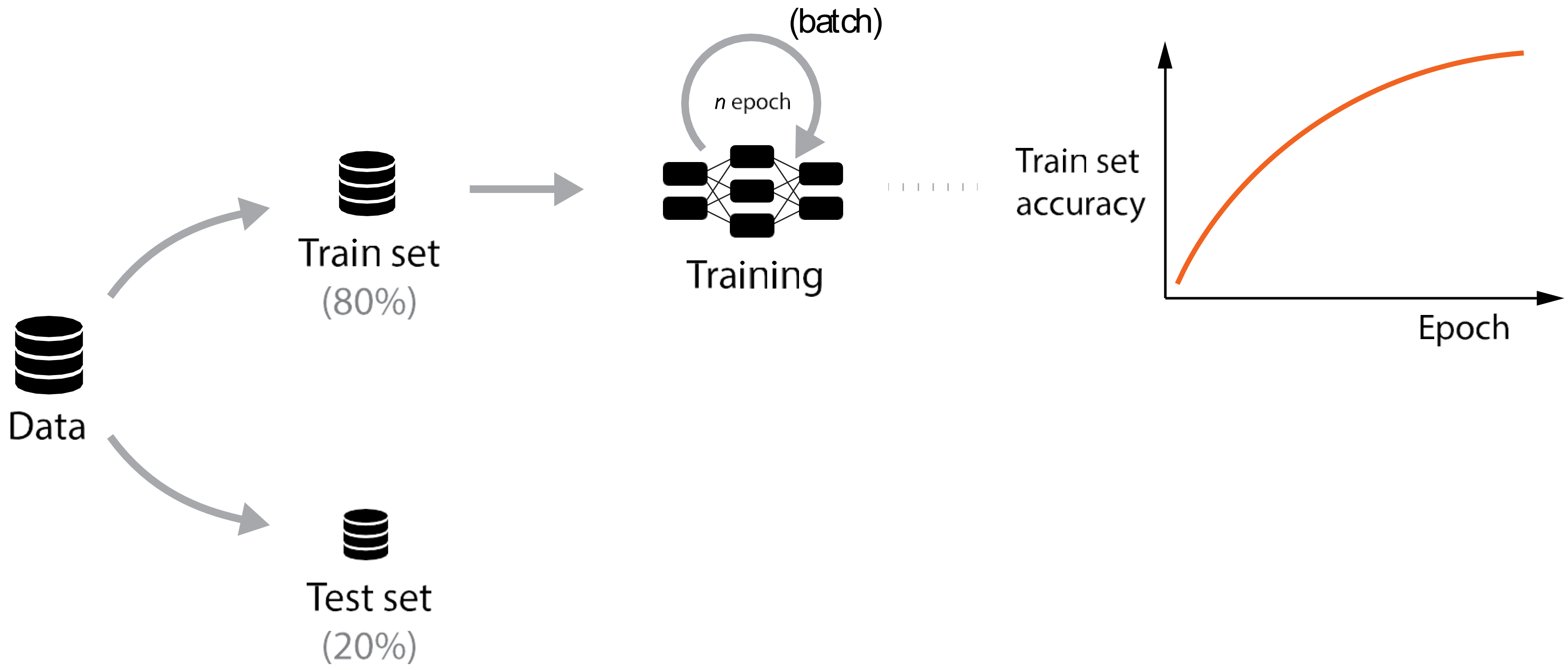
Back-propagation

Learning process

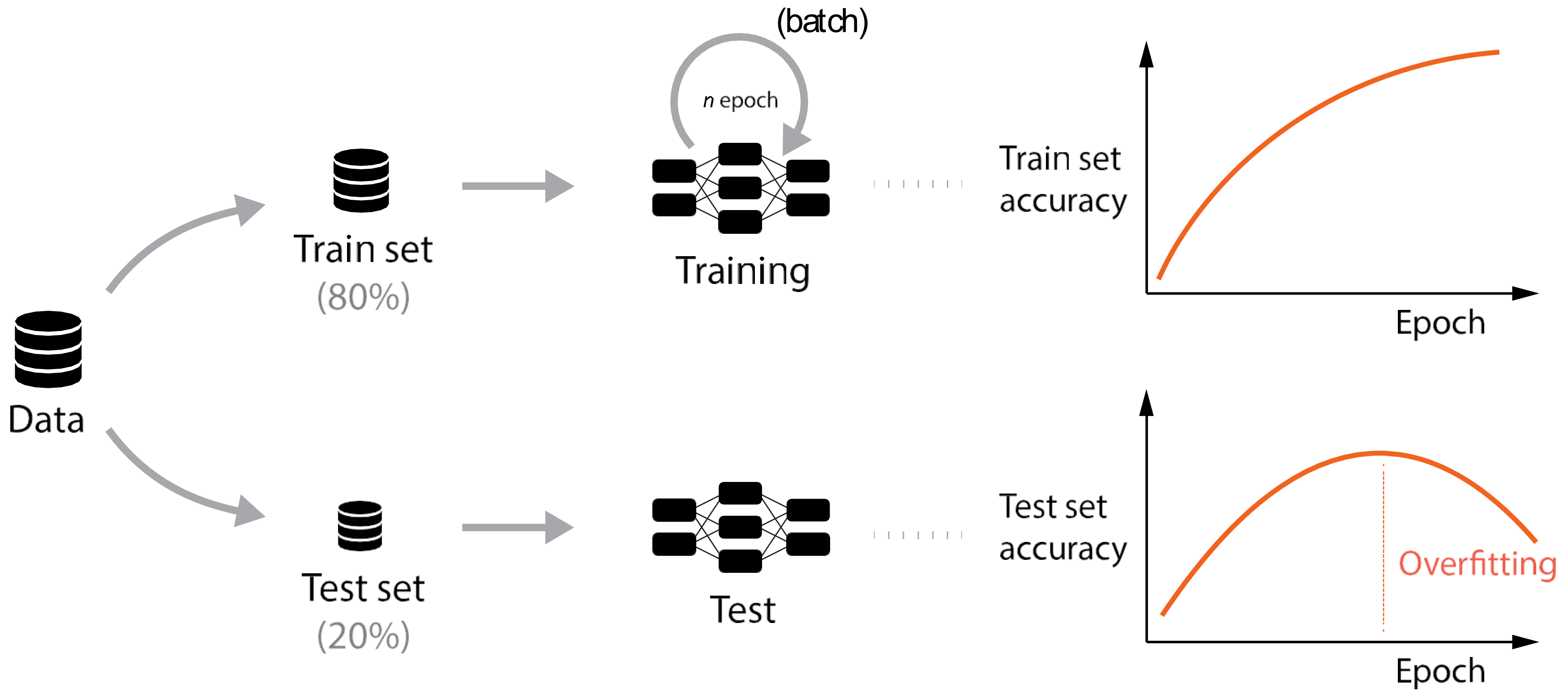
Training process- general



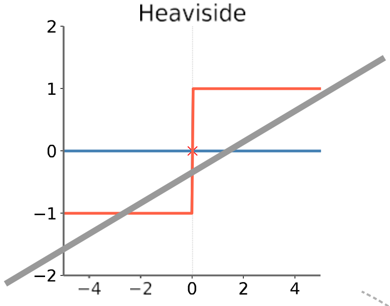
Training process - general



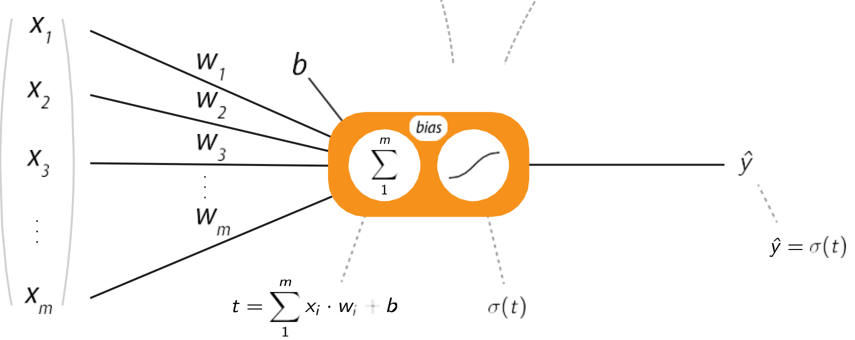
Training process - general

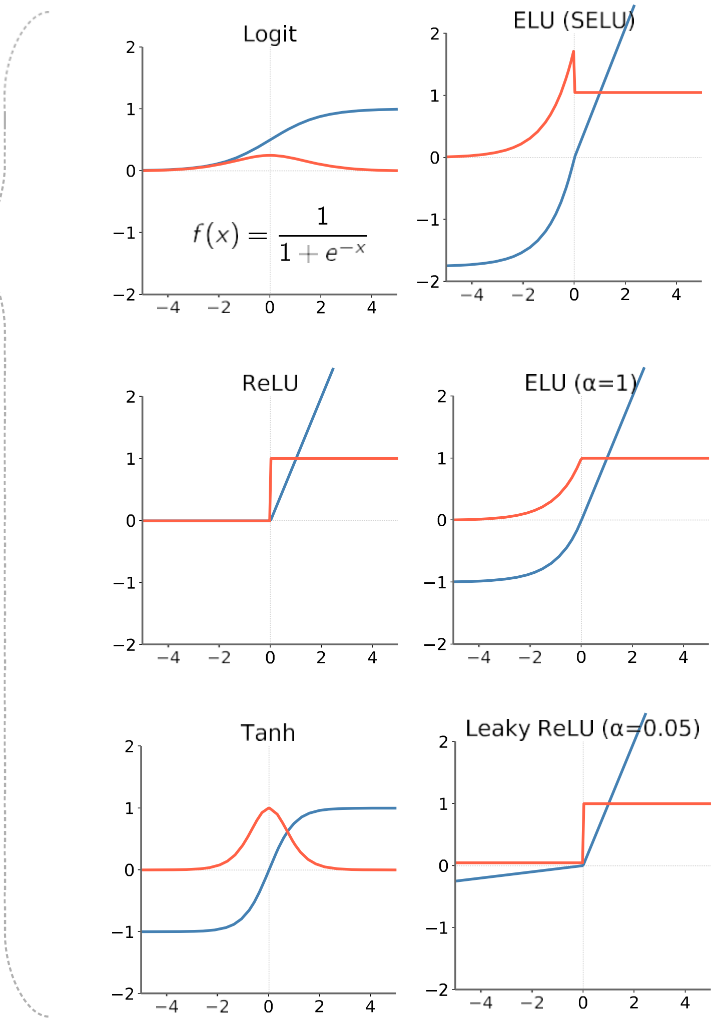


Deep Neural Networks



1958





Input X

Bias / Weight Θ

Activation function $\sigma(t)$

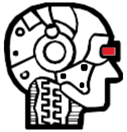
Output $\hat{y}$

Notebook[ACTF1]

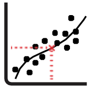
Roadmap

Episode :S01E01

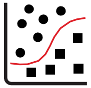
1



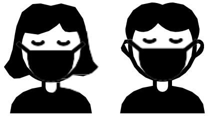
History,
Fundamental
Concepts



Basic
Regression
DNN



Basic
Classification
DNN



1.1 Introduction
Context, tools and ressources

1.2 From the linear regression
to the first neuron

1.3 Neurons incontroversy

1.4 Data and neurons

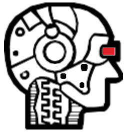
Basic Regression

Basic Classification

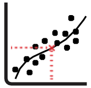
Roadmap

Episode :S01E01

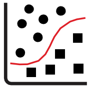
1



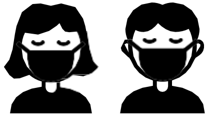
History,
Fundamental
Concepts



Basic
Regression
DNN



Basic
Classification
DNN



1.1 Introduction
Context, tools and ressources

1.2 From the linear regression
to the first neuron

1.3 Neurons incontroversy

1.4 Data and neurons

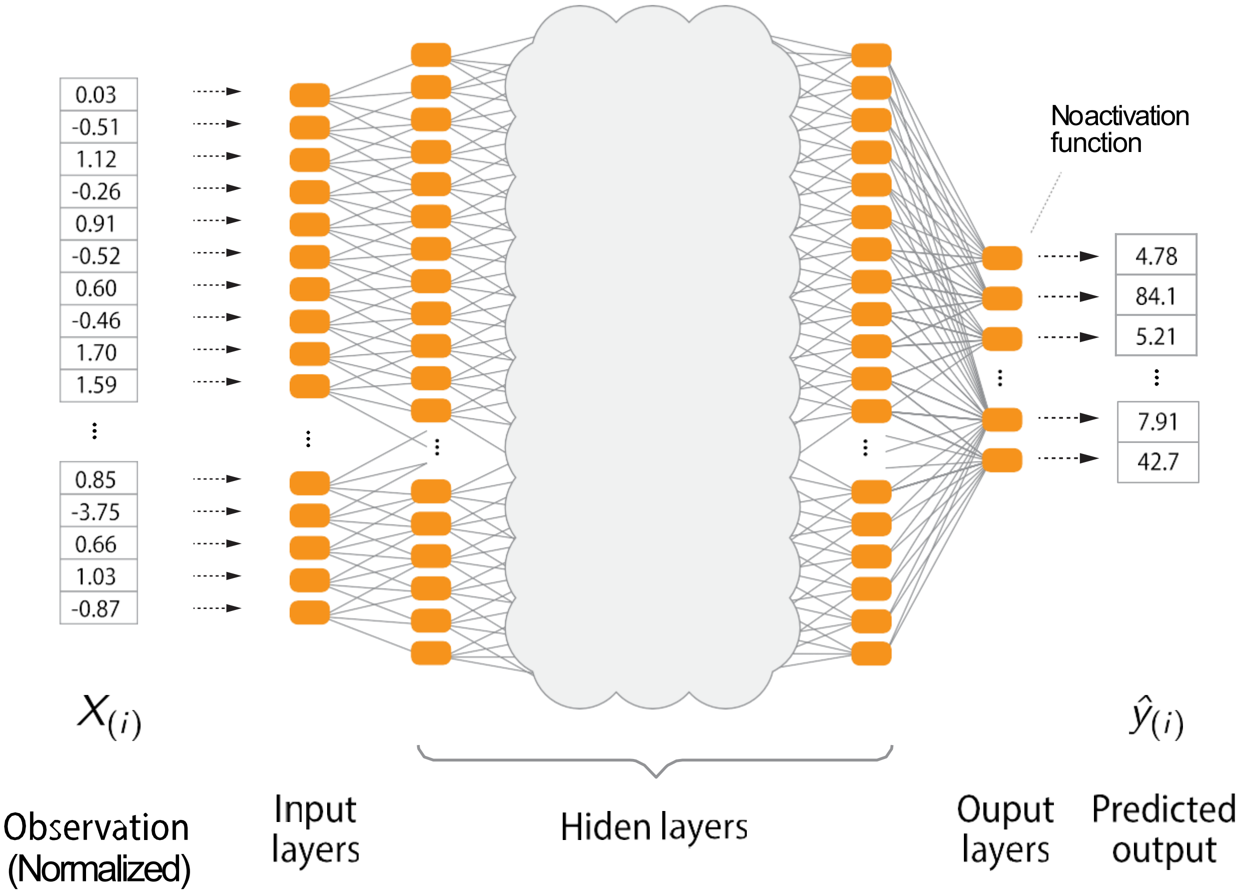
Basic Regression

Basic Classification

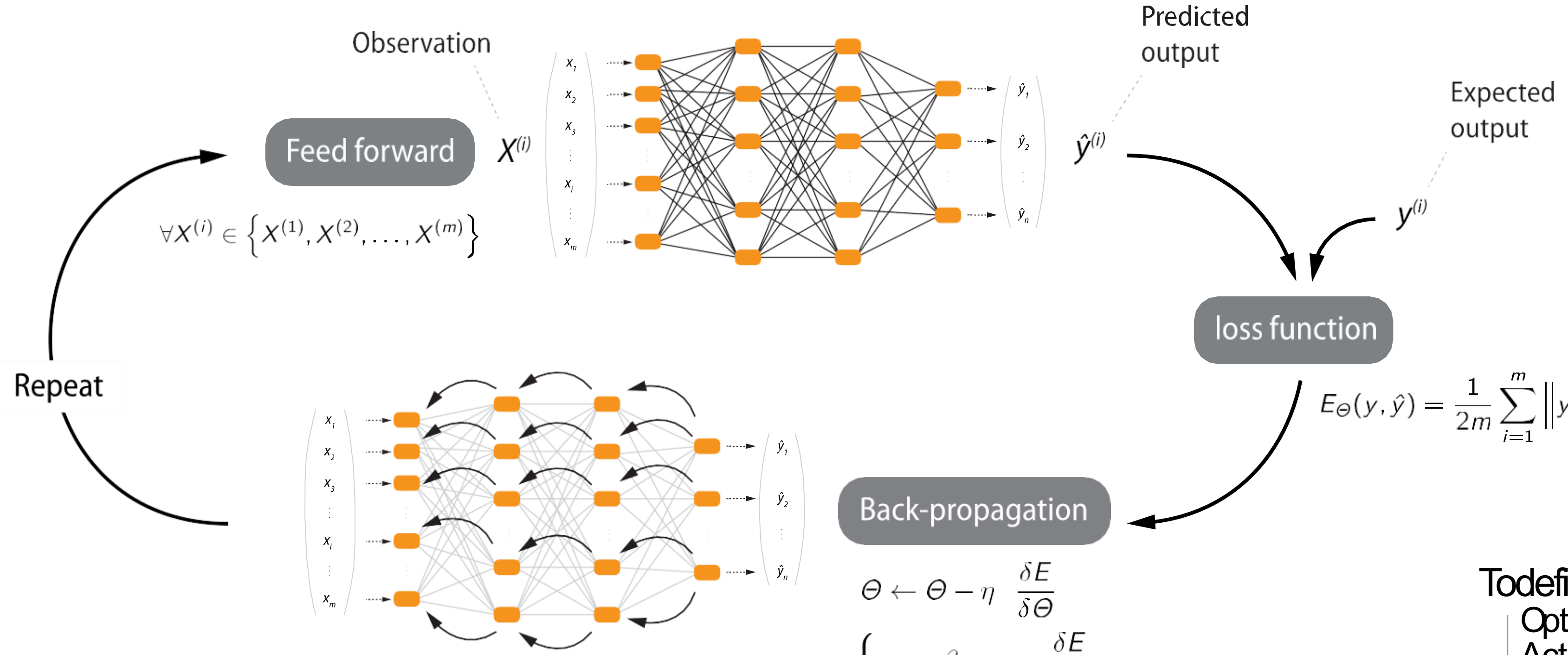
Regression with a DNN

$X_{(i)}$: Observations

$y_{(i)}$: Expected output



Training process - general



$$E_{\Theta}(y, \hat{y}) = \frac{1}{2m} \sum_{i=1}^{m} \left\| y^{(i)} - \hat{y}^{(i)} \right\|^2$$

$$\Theta \leftarrow \Theta - \eta \frac{\delta E}{\delta \Theta}$$

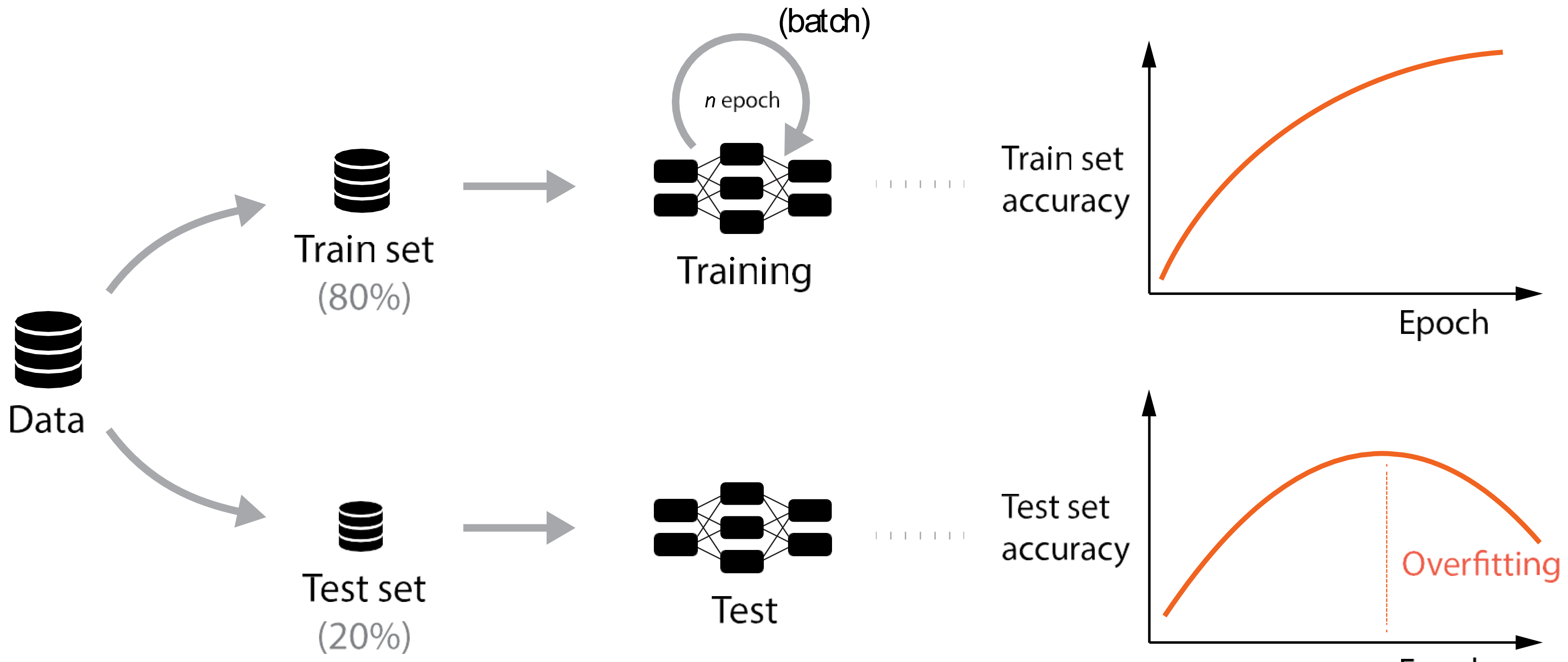
$$\begin{cases} m \leftarrow \beta m - \eta \dfrac{\delta E}{\delta \Theta} \\ \Theta \leftarrow \Theta + m \end{cases}$$

...

η : Learning rate
β : Momentum

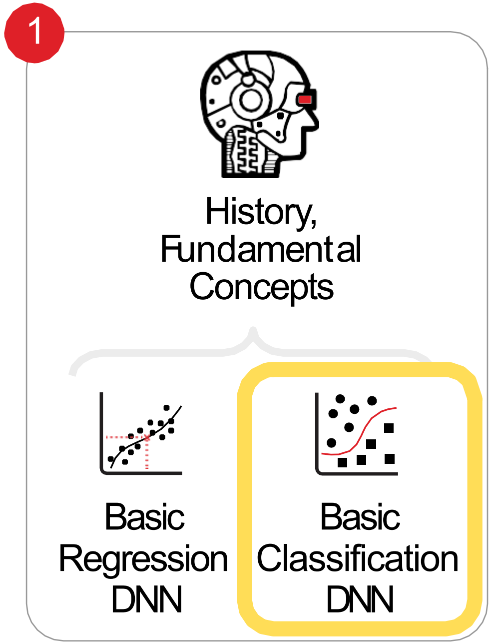
Todefine :
- Optimization
- Activation
- Loss
- Metrics
- ...

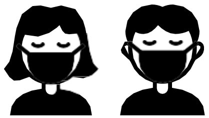
Training process- general



Roadmap

Episode :S01E01





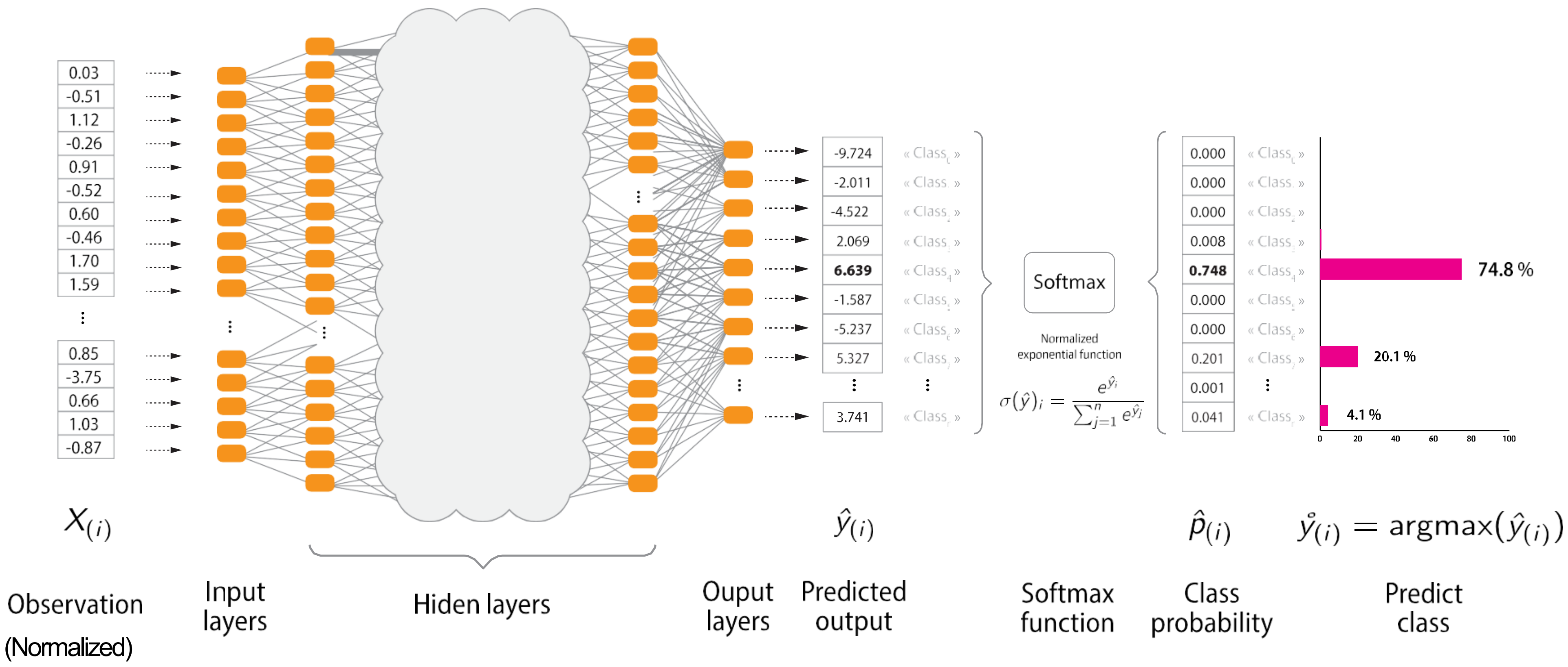
1.1 Introduction
Context, tools and ressources

1.2 From the linear regression
to the first neuron

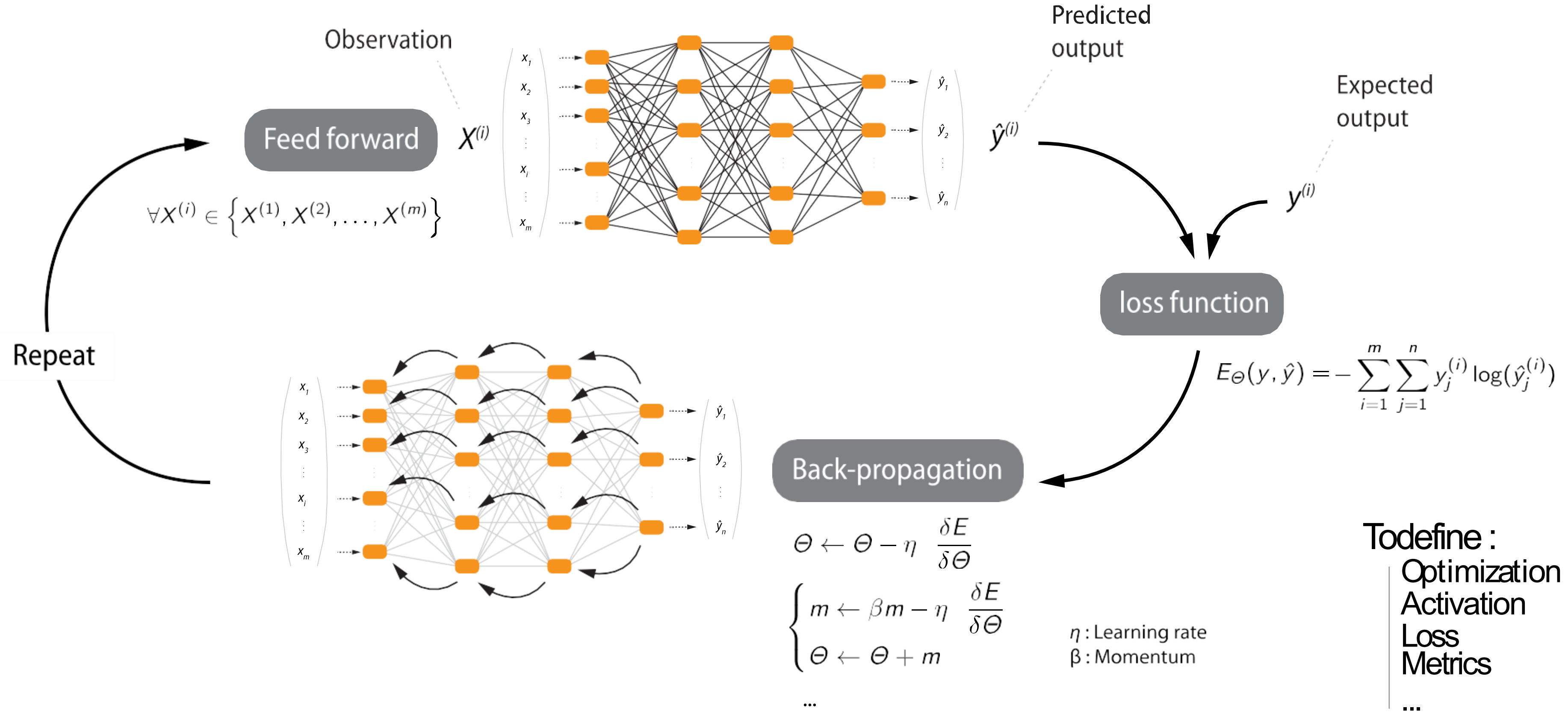
1.3 Neurons incontroversy

1.4 Data and neurons
Basic Regression
BasicClassification

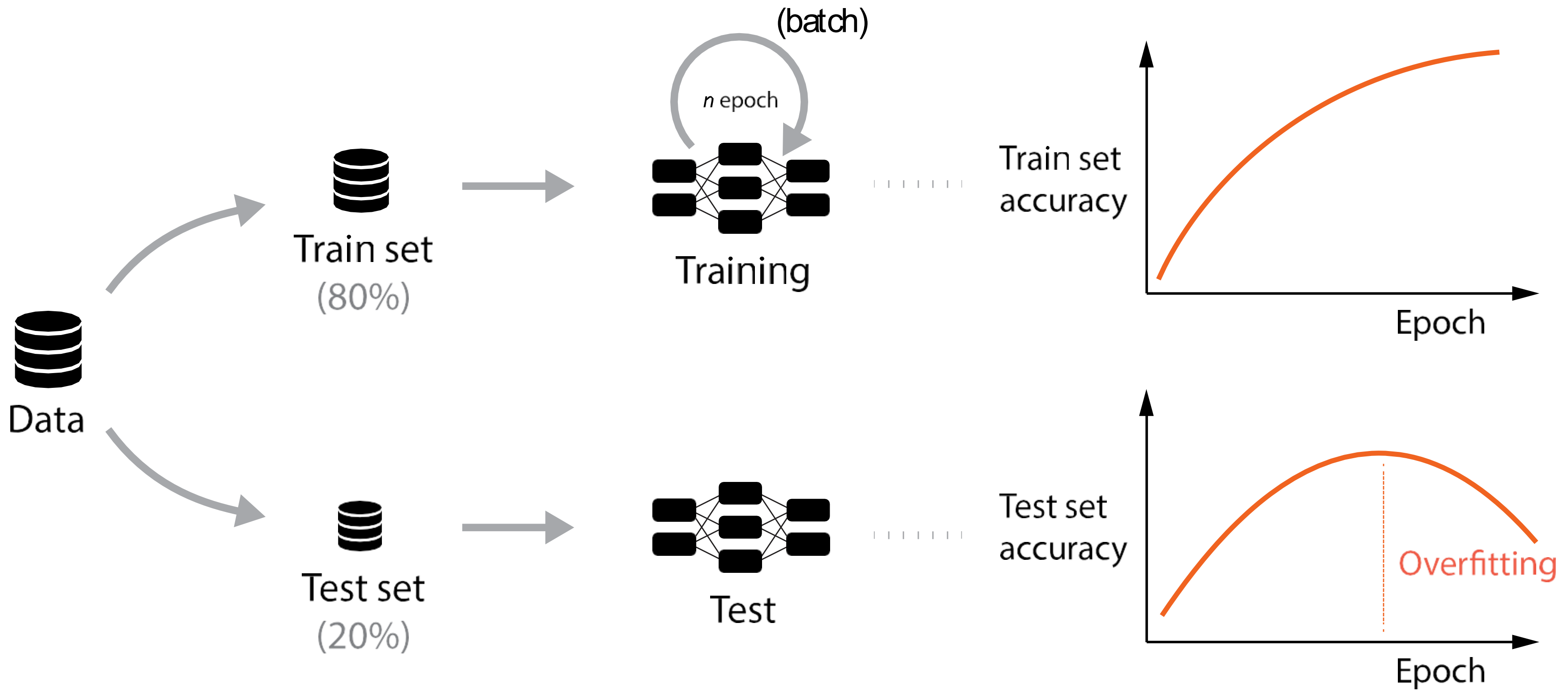
Classification with a DNN



Training process- general



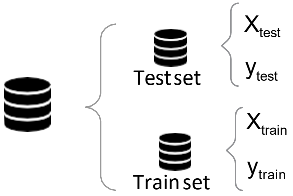
Training process- general



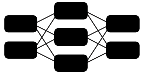
Step 1 - Import and init

Step 2 - Retrieve data

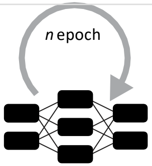
Step 3 - Preparing the data



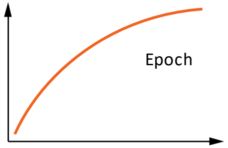
Step 4 - Build a model



Step 5 - Train the model

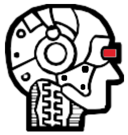


Step 6 - Evaluate

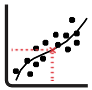


Few little things and concepts to keep in mind

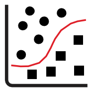
1



History,
Fundamental
Concepts



Basic
Regression
DNN



Basic
Classification
DNN

- Regression vs.Classification
- Data normalization
- Training and validation
- Epochs and Batchs
- Activation functions
- Loss function
- Optimization and gradient descent
- Metrics
- Softmax and Argmax function
- Numpy shape

Simple classification with DNN

Notebook : [MNIST1]

Objective :
Recognizing handwritten numbers

Dataset :
Modified National Institute of Standards and Technology (MNIST)

Next :

2. Principles and concepts of convolutional networks (CNN).