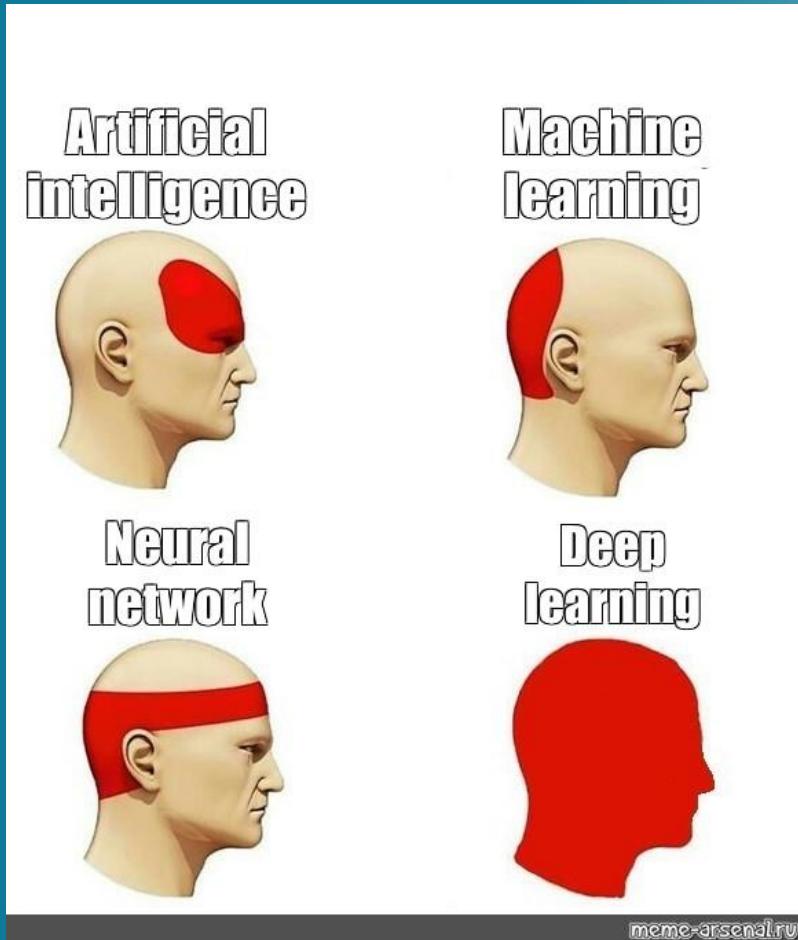


CHPS0704 – Introduction à l'IA

Cours 1 – Machine Learning pour les
Débutants



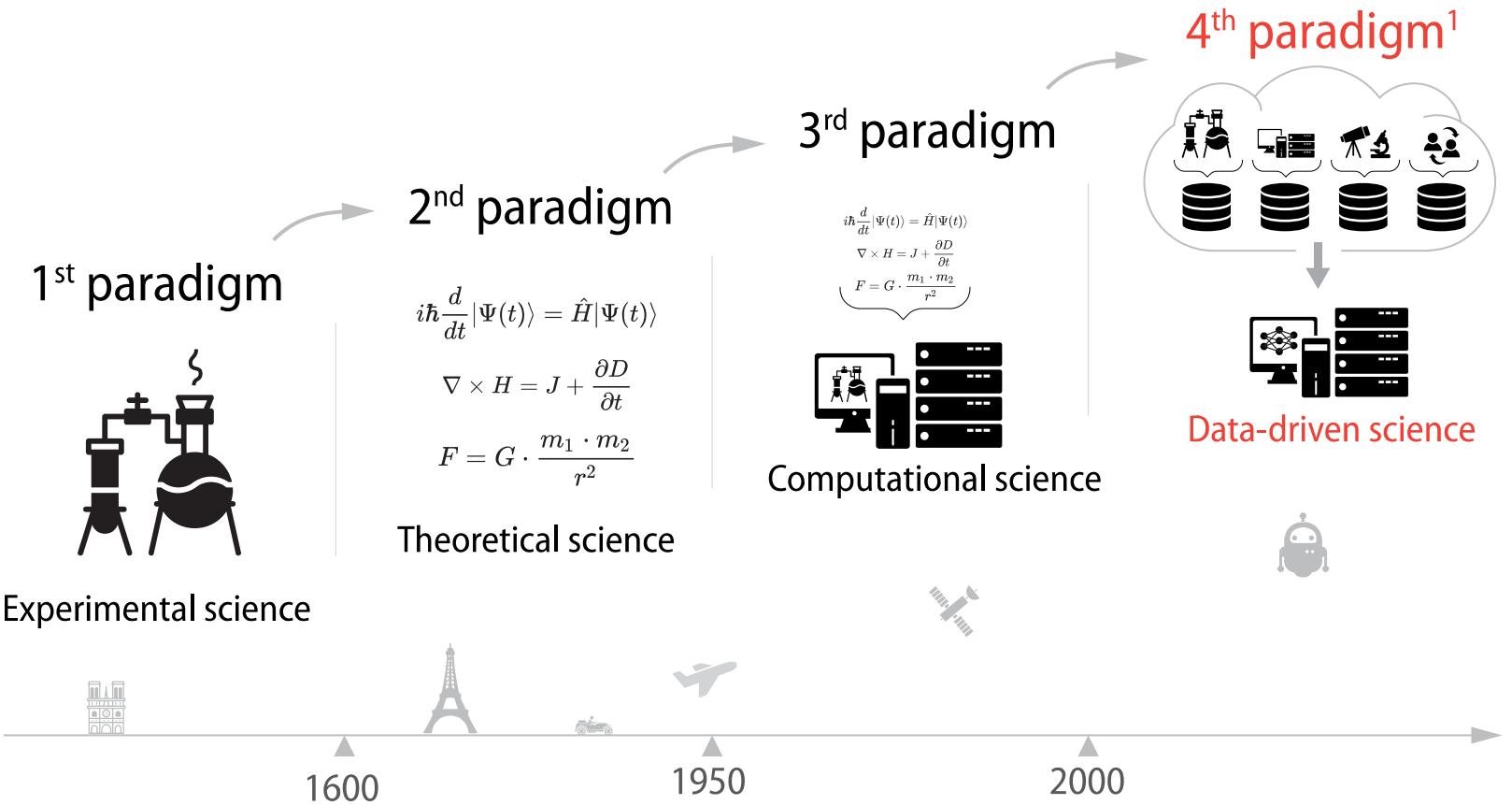
OBJECTIFS DE CE COURS

- Préparation et découpage de données
- Introduction au machine learning avec Scikit-Learn
 - Régression Linéaire
 - régression linéaire simple et polynomiale
 - KNN
 - Régression Logistique (classification)
 - Gaussian
 - arbres de décision / forêts aléatoires
 - boosting
 - SVM
 - réseaux de neurones
 - Clustering
 - KMeans
 - DBSCAN
 - classification hiérarchique
 - Réduction de la dimensionnalité
- Les principales métriques de performance, leurs atouts et inconvénients
- Introduction aux réseaux de neurones profonds
 - frameworks Pytorch et Tensorflow
 - wrappers (Fast.ai, Keras, Pytorch-lightning)

ORGANISATION

- 14h CM, 16h TP
 - On fera surtout des séances mixtes
- Ressources
 - Utilisation prioritaire de Google Colab ou autre environnement en ligne
 - Vous pouvez utiliser votre propre ordinateur
 - ROMEO reste réservé pour des cas exceptionnels (projets ?)
- Evaluation
 - 3 notes de contrôle continu

INTRODUCTION – LA RÉVOLUTION IA

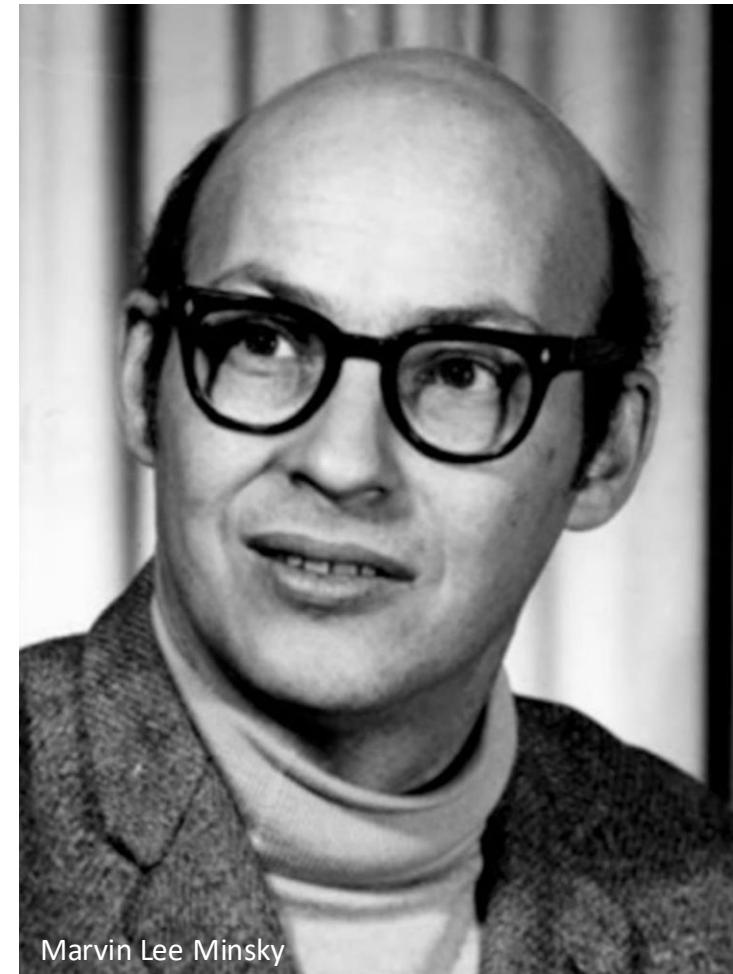


¹ Jim Gray, 2007

L'INTELLIGENCE ARTIFICIELLE

- construction de programmes informatiques capables de prendre en charge des tâches habituellement effectuées par des humains
- L'objectif est de parvenir à transmettre à une machine des fonctions propres au vivant : rationalité, raisonnement, mémoire et perception.

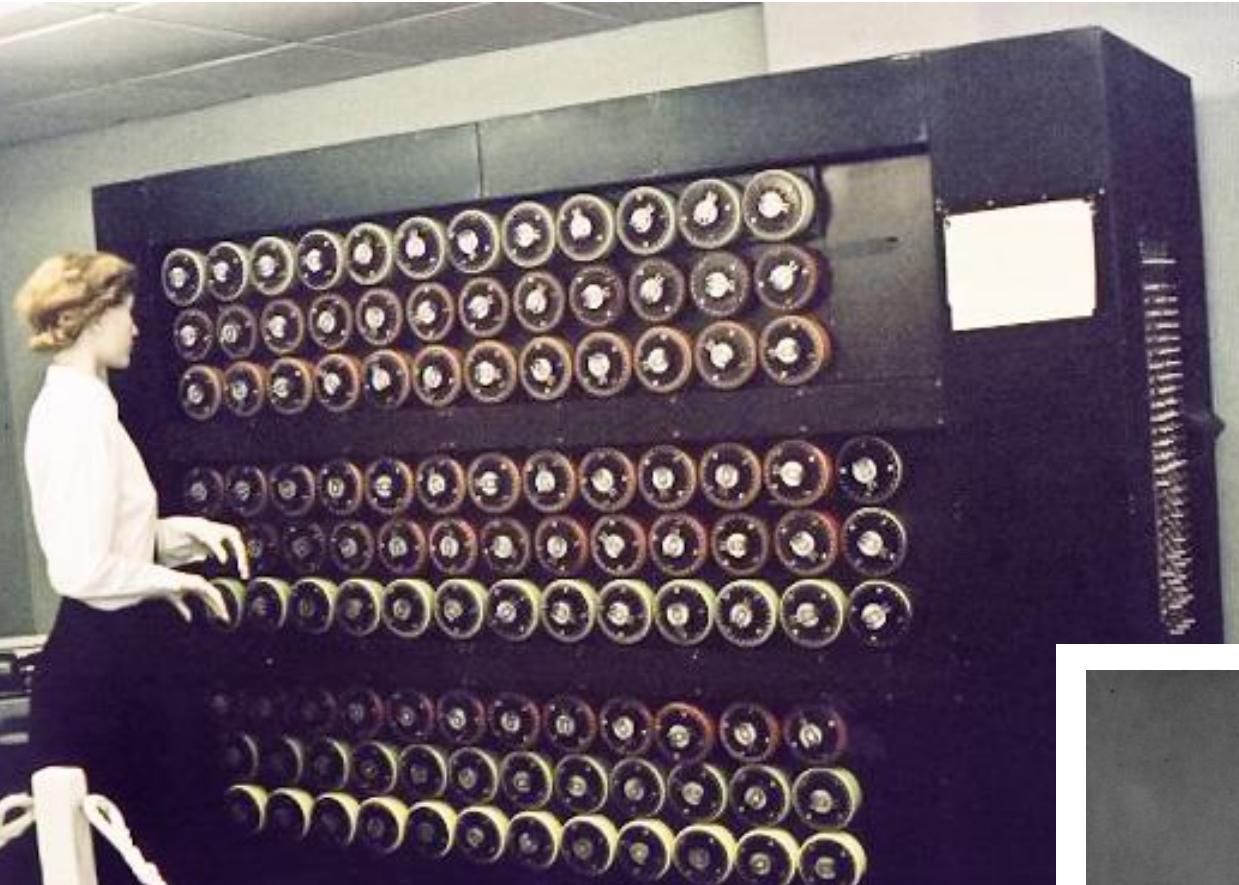
Raisonner / Comprendre / Interagir



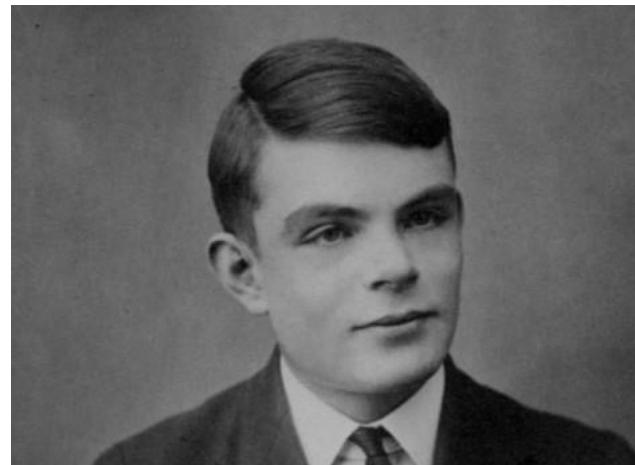
Marvin Lee Minsky

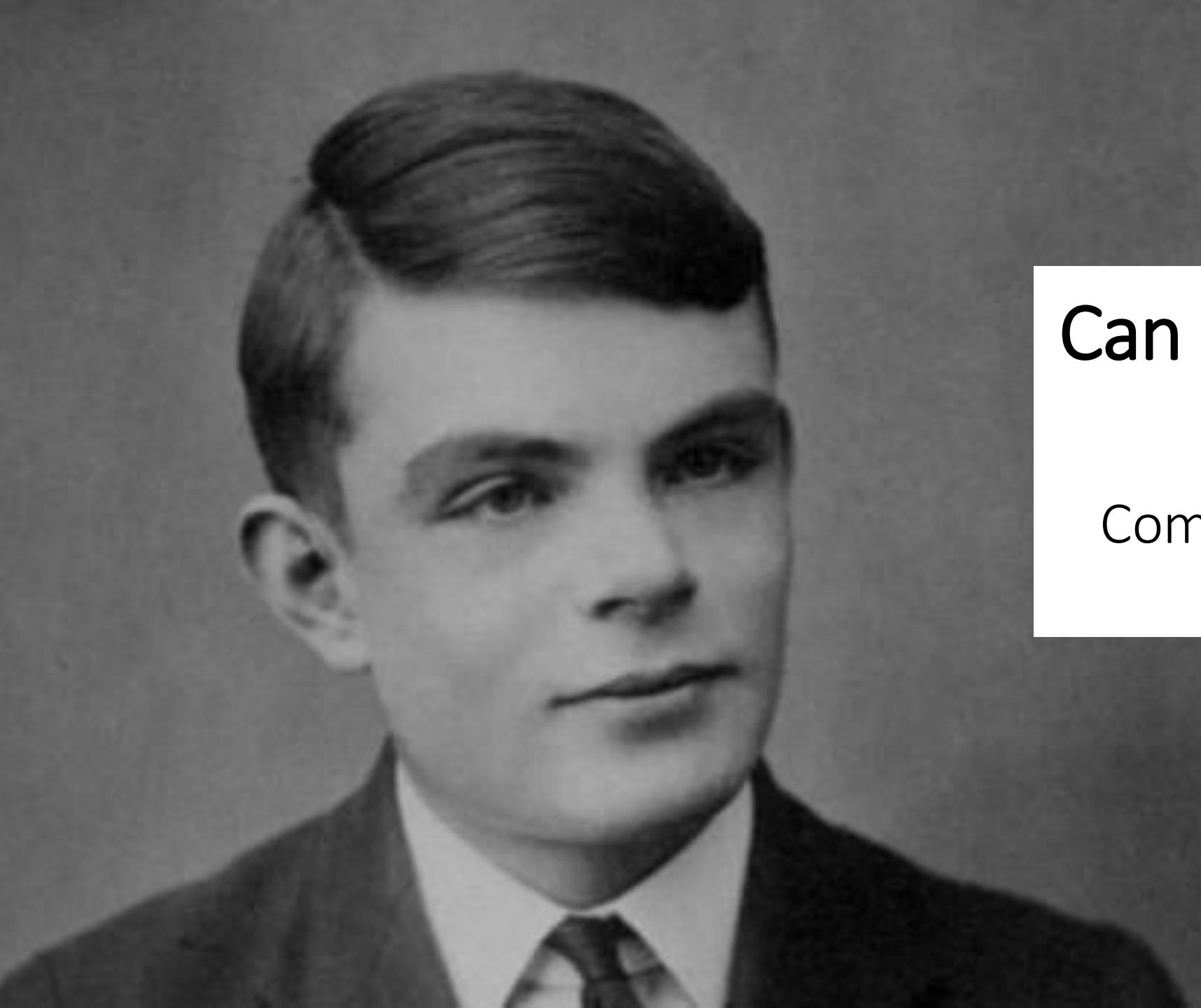
Un ordinateur peut-il
devenir intelligent ?

THE BOMB



- Alan Turing
- Enigma
- The bomb





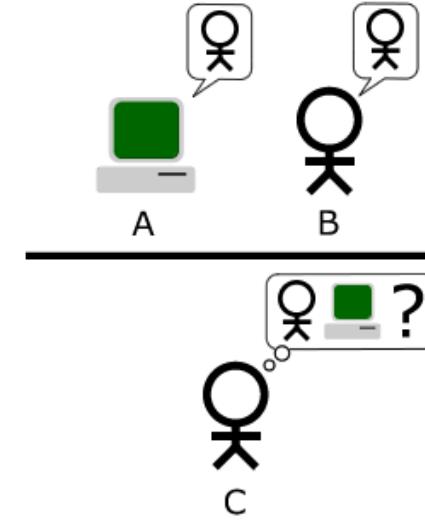
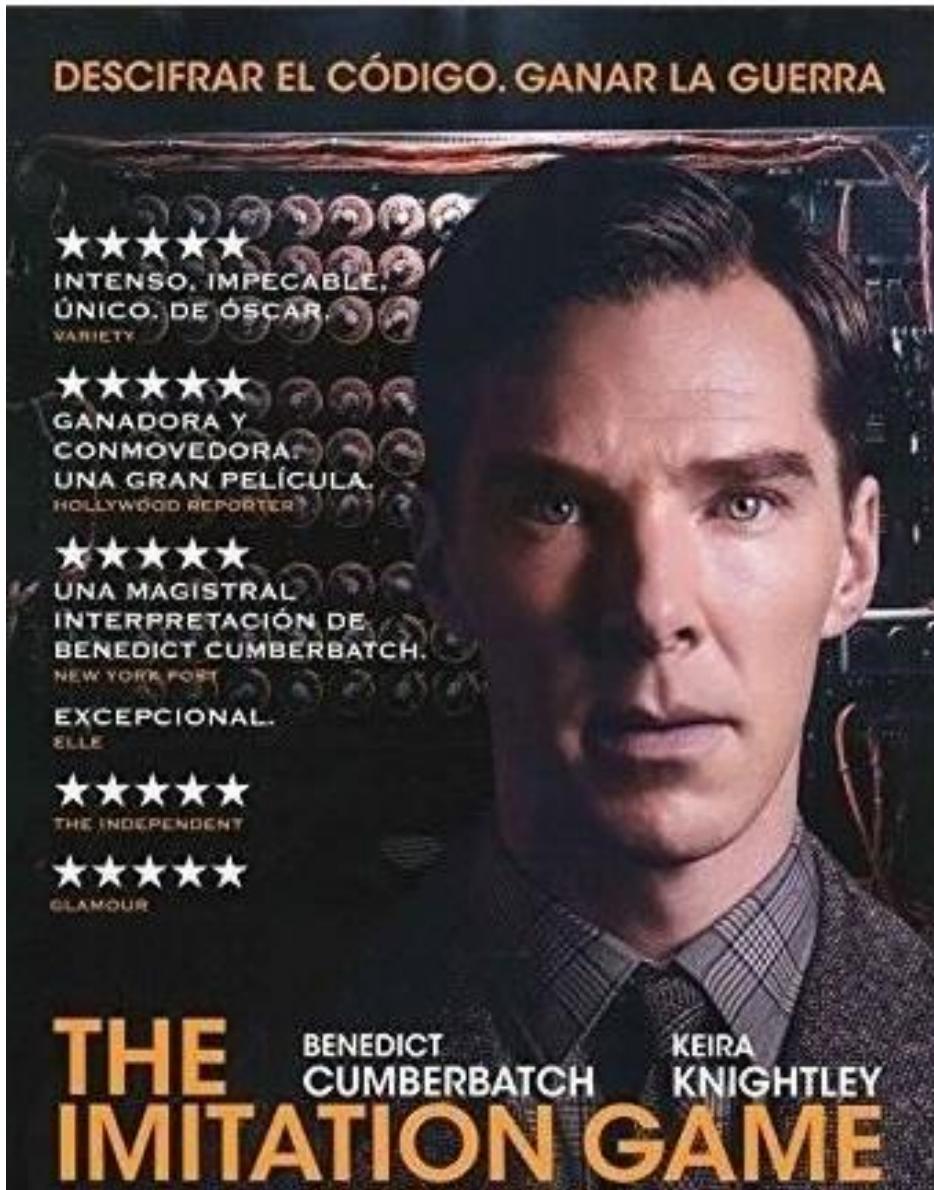
Can machines think ?

A. Turing

Computing machinery and
intelligence, 1950

Mind (en), Oxford University Press,
vol. 59, no 236, octobre 1950 (DOI
[10.1093/mind/LIX.236.433](https://doi.org/10.1093/mind/LIX.236.433))

The imitation game

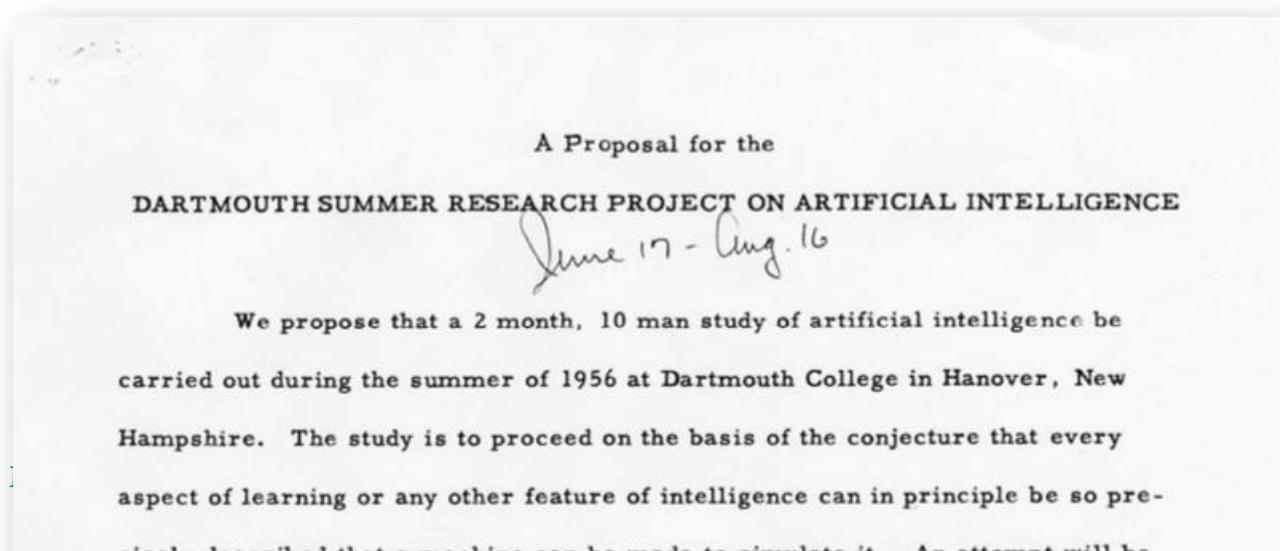


- Test de Turing
- « Un ordinateur peut-il penser »
- Intelligence multiple
- Faire semblant de faire des erreurs
- Machine expérimentale (pas de preuve ou d'algo)
- Ordinateur digital : disponibilité, **puissance**
- Ordinateur à variable discrète : mémoire ?

Historique de l'IA

- 1956 : L'intelligence artificielle devient un véritable domaine scientifique
 - Dartmouth Summer Research Project on Artificial Intelligence
 - Organisée par John McCarty

“ find how to make machines [...] solve kinds of problems now reserved for humans ”



Nathaniel Rochester

Marvin L. Minsky

John McCarthy

Oliver G. Selfridge Ray Solomonoff

Trenchard More

Claude E. Shannon

Historique de l'IA

- 1956 : L'intelligence artificielle devient un véritable domaine scientifique

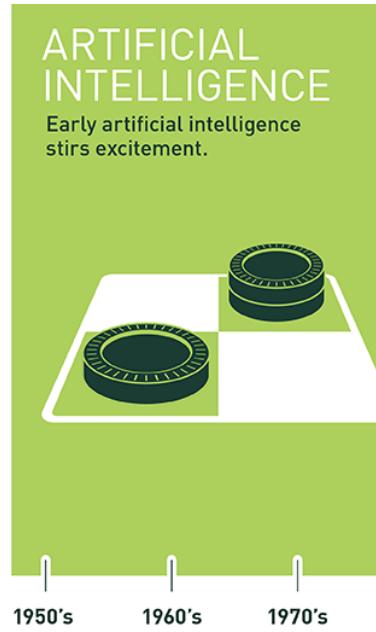
- 60' : financements, laboratoires USA & UK

« des machines seront capables, d'ici 20 ans, de faire le travail que toute personne peut faire »



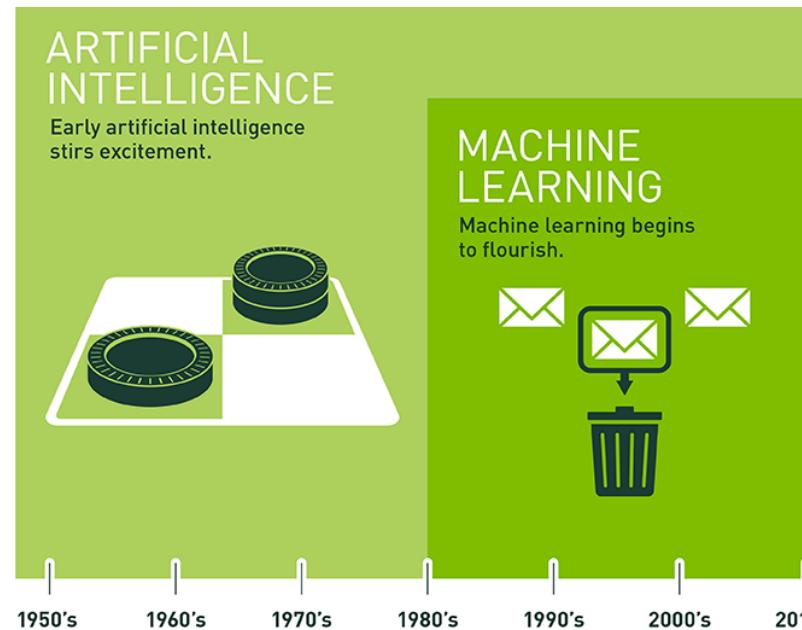
Historique de l'IA

- 1956 : L'intelligence artificielle devient un véritable domaine scientifique
- 60' : financements, laboratoires USA & UK
 - « des machines seront capables, d'ici 20 ans, de faire le travail que toute personne peut faire »
- 1974 : AI Winter : les projets n'aboutissent pas, les ordinateurs ne sont pas assez puissants, les financements sont réduits



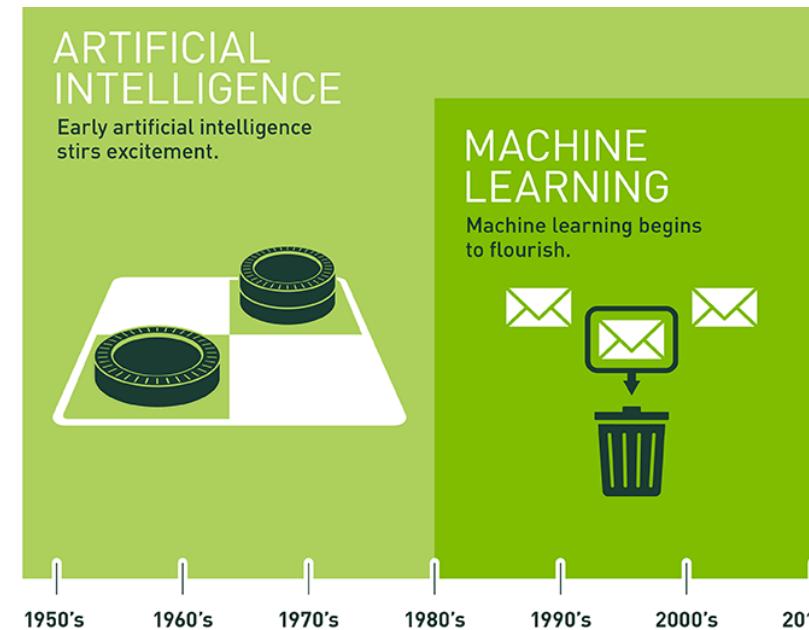
Historique de l'IA

- 1956 : L'intelligence artificielle devient un véritable domaine scientifique
- 60' : financements, laboratoires USA & UK
 - « des machines seront capables, d'ici 20 ans, de faire le travail que toute personne peut faire »
- 1974 : AI Winter : les projets n'aboutissent pas, les ordinateurs ne sont pas assez puissants, les financements sont réduits
- 80' : Systèmes experts (une seule tâche)



Historique de l'IA

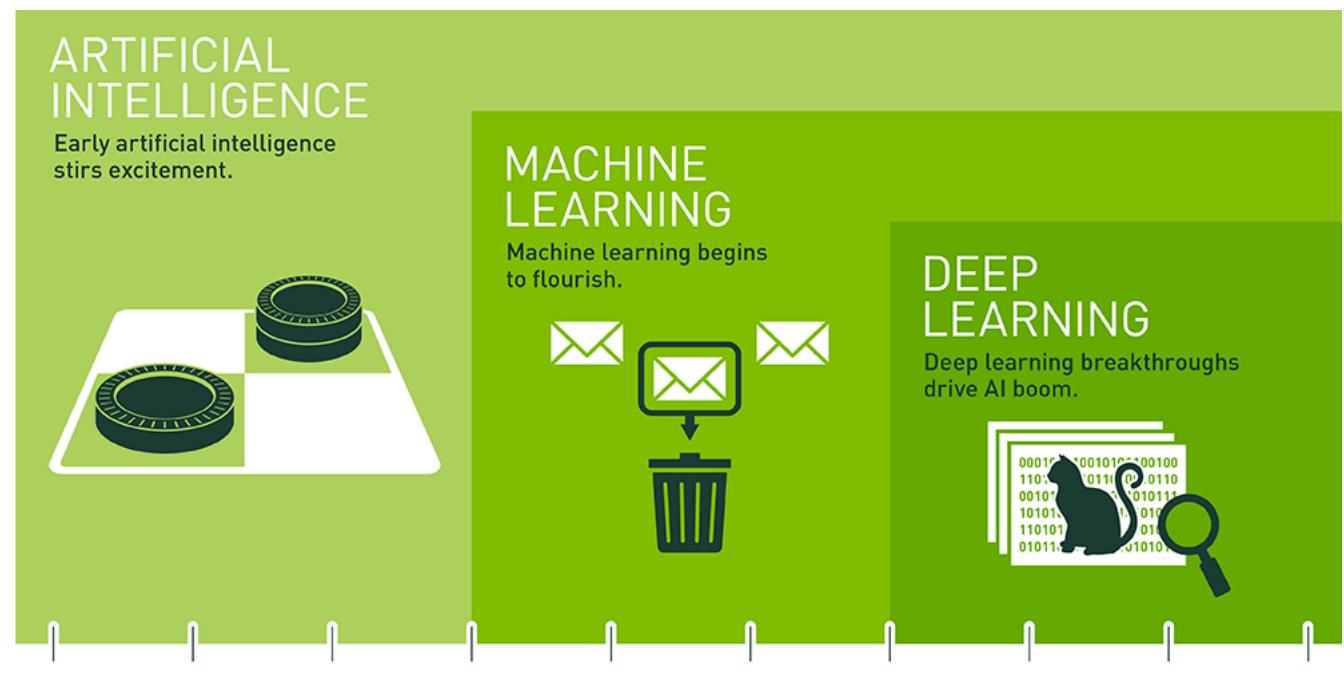
- 1956 : L'intelligence artificielle devient un véritable domaine scientifique
- 60' : financements, laboratoires USA & UK
 - « des machines seront capables, d'ici 20 ans, de faire le travail que toute personne peut faire »
- 1974 : AI Winter : les projets n'aboutissent pas, les ordinateurs ne sont pas assez puissants, les financements sont réduits
- 80' : Systèmes experts (une seule tâche)
- 90' : Ordinateurs puissants, nouveaux domaines



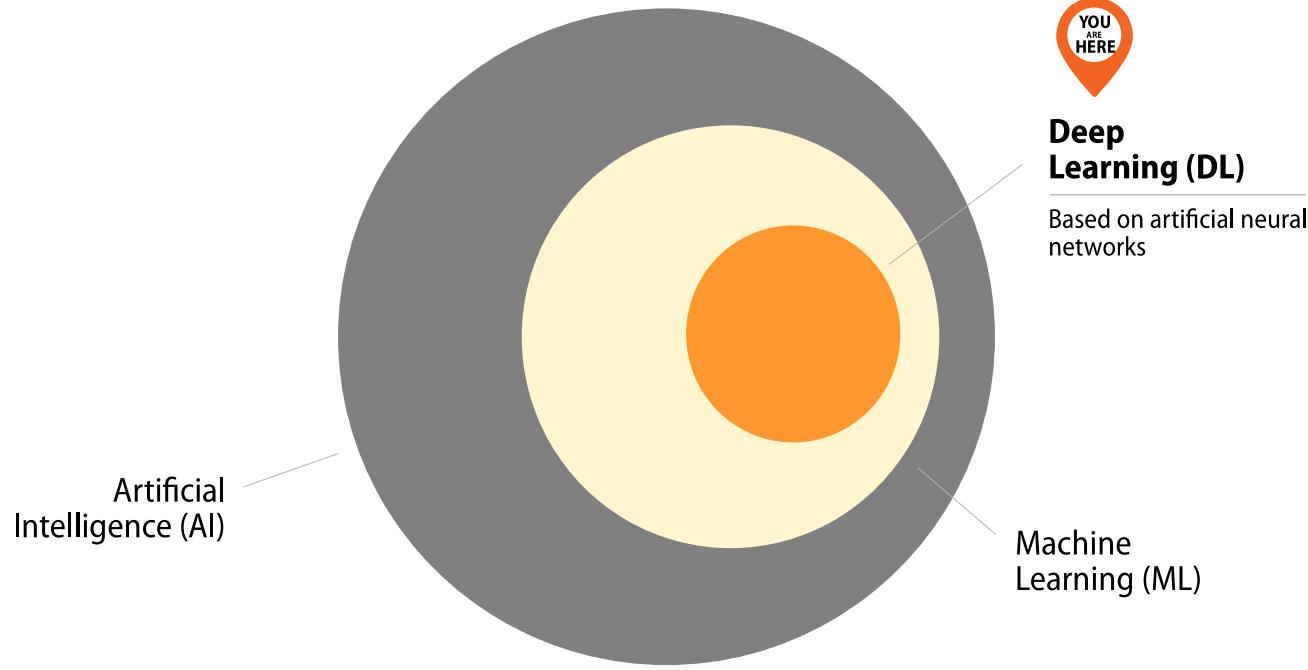
Historique de l'IA

■ 2000' : l'avènement des données

- Les nouvelles algorithmes
- Les puissances de calcul phénoménales
- Apprentissage profond : Deep Learning



TOUTE IA EST PAREILLE ALORS ?

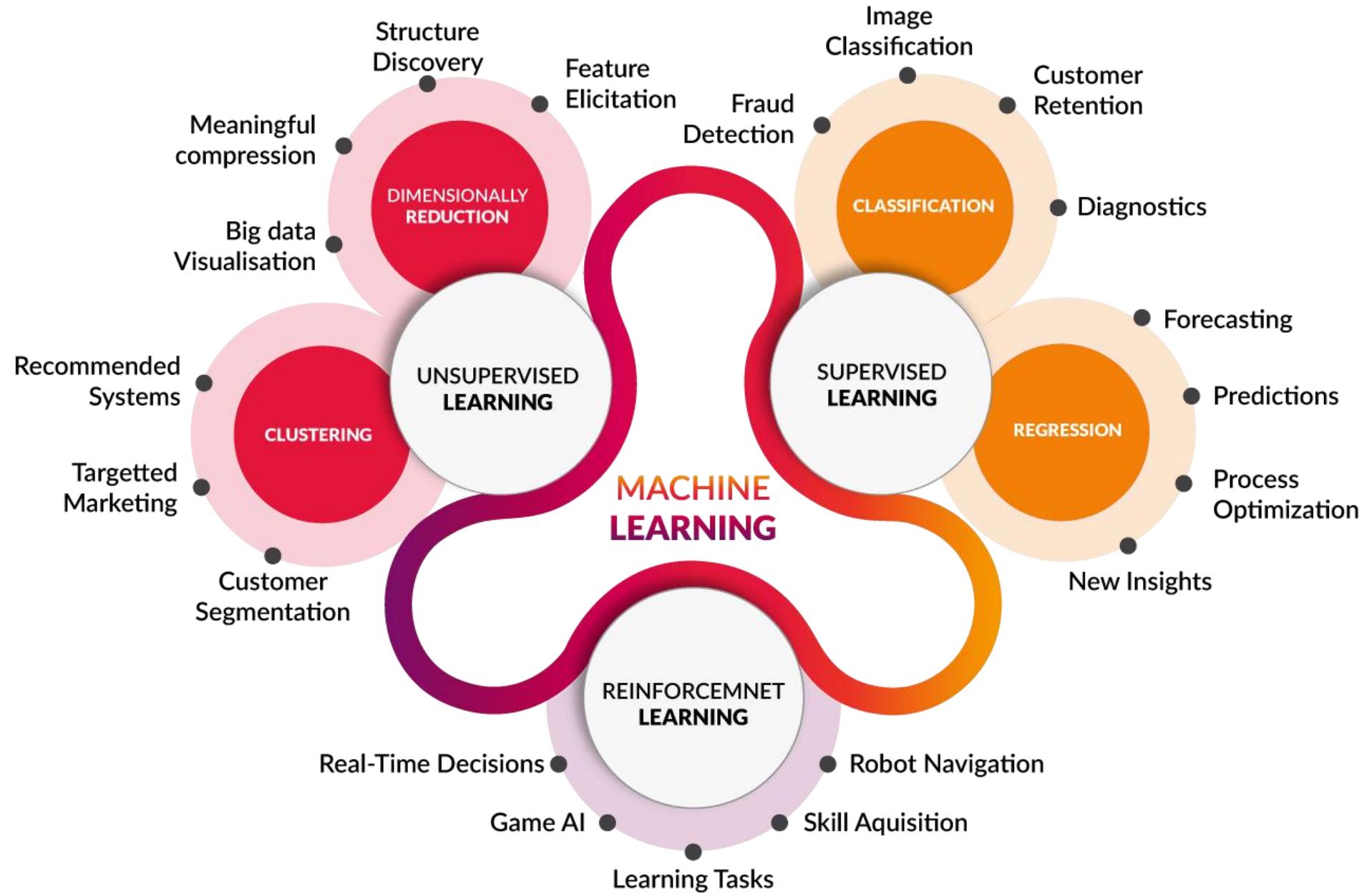


Intelligence Artificielle = ensemble des techniques permettant à des machines d'accomplir des tâches et de résoudre des problèmes normalement réservés aux humains

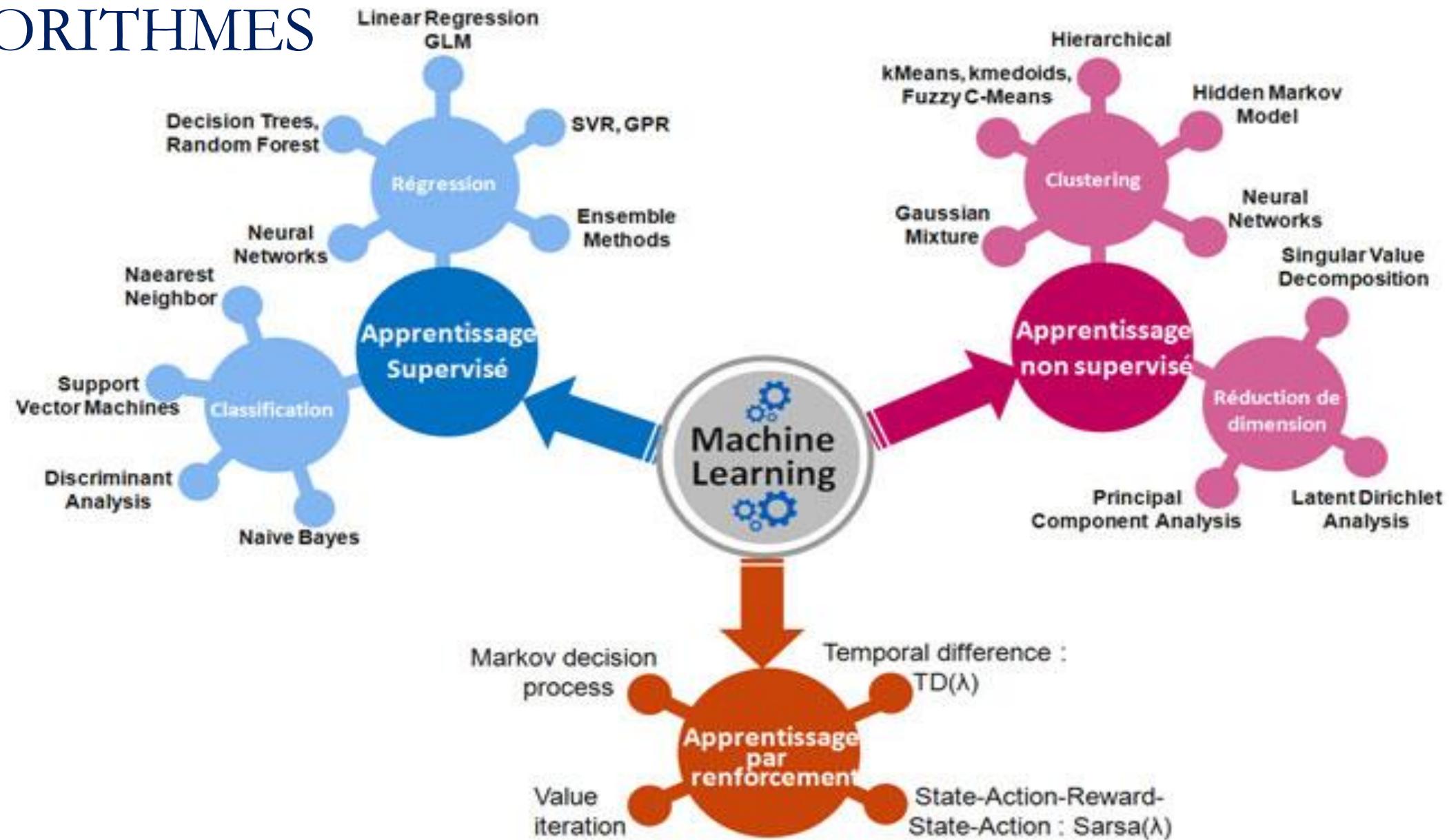
Machine Learning = apprentissage automatique « champ d'étude de l'intelligence artificielle qui se base sur des approches pour donner aux ordinateurs la capacité d'apprendre à partir de données, c à d d'améliorer leurs performances à résoudre des tâches sans être explicitement programmés pour chacune »

Deep learning = Branche du Machine Learning spécialisée dans l'utilisation de réseaux de neurones profonds

USAGES

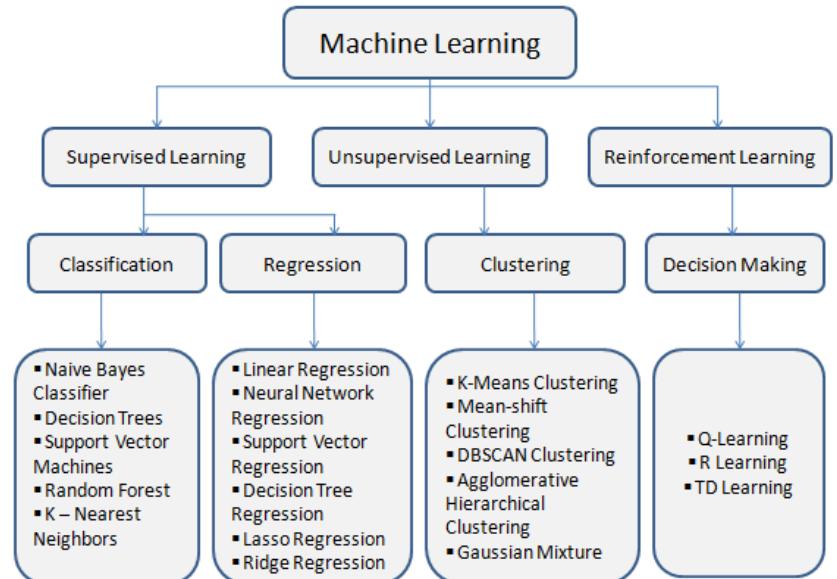


ALGORITHMES

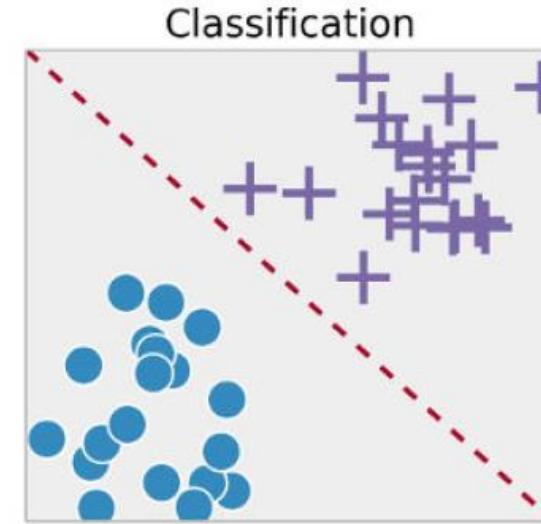
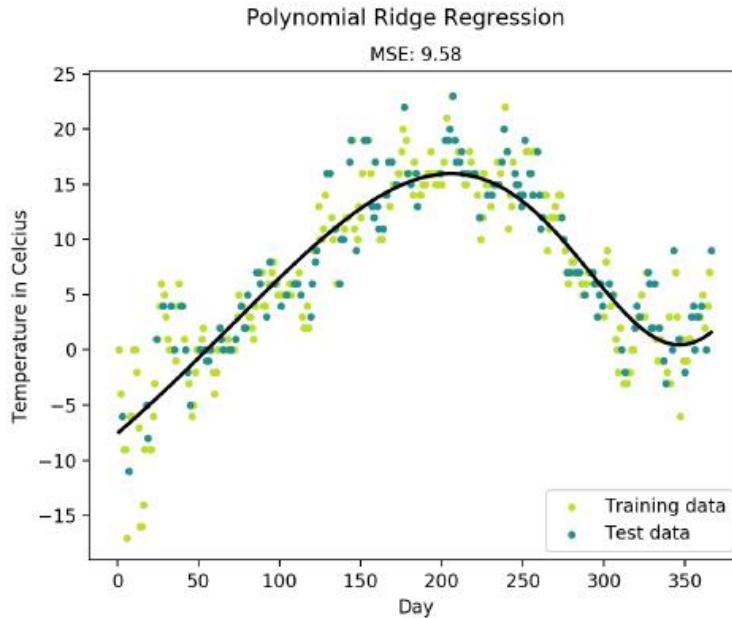


LES BASES : LE MACHINE LEARNING

- Un modèle nécessite quelques paramètres pour fonctionner
 - Ex : le périmètre P d'un cercle de rayon r s'écrit : $P = 2 \times \pi \times r$
 - Que se passerait-il si on n'avait pas π mais juste **plusieurs P et r ?** Peut-on déduire la valeur de π ?
- Le ML couvre plusieurs techniques mathématiques/statistiques permettant d'apprendre des paramètres à partir des données
- Le ML "classique" comprend souvent des techniques qui s'appliquent facilement à des données numériques
- On regarde rapidement deux de ces familles

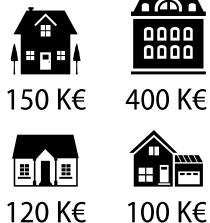


CLASSIFICATION VS REGRESSION

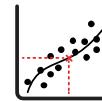


Régression

Prédire une variable quantitative



Tell me,
what's the
price ?



Classification

Prédire une classe (qualitative, discrète)



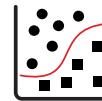
This is a cat



This is a rabbit



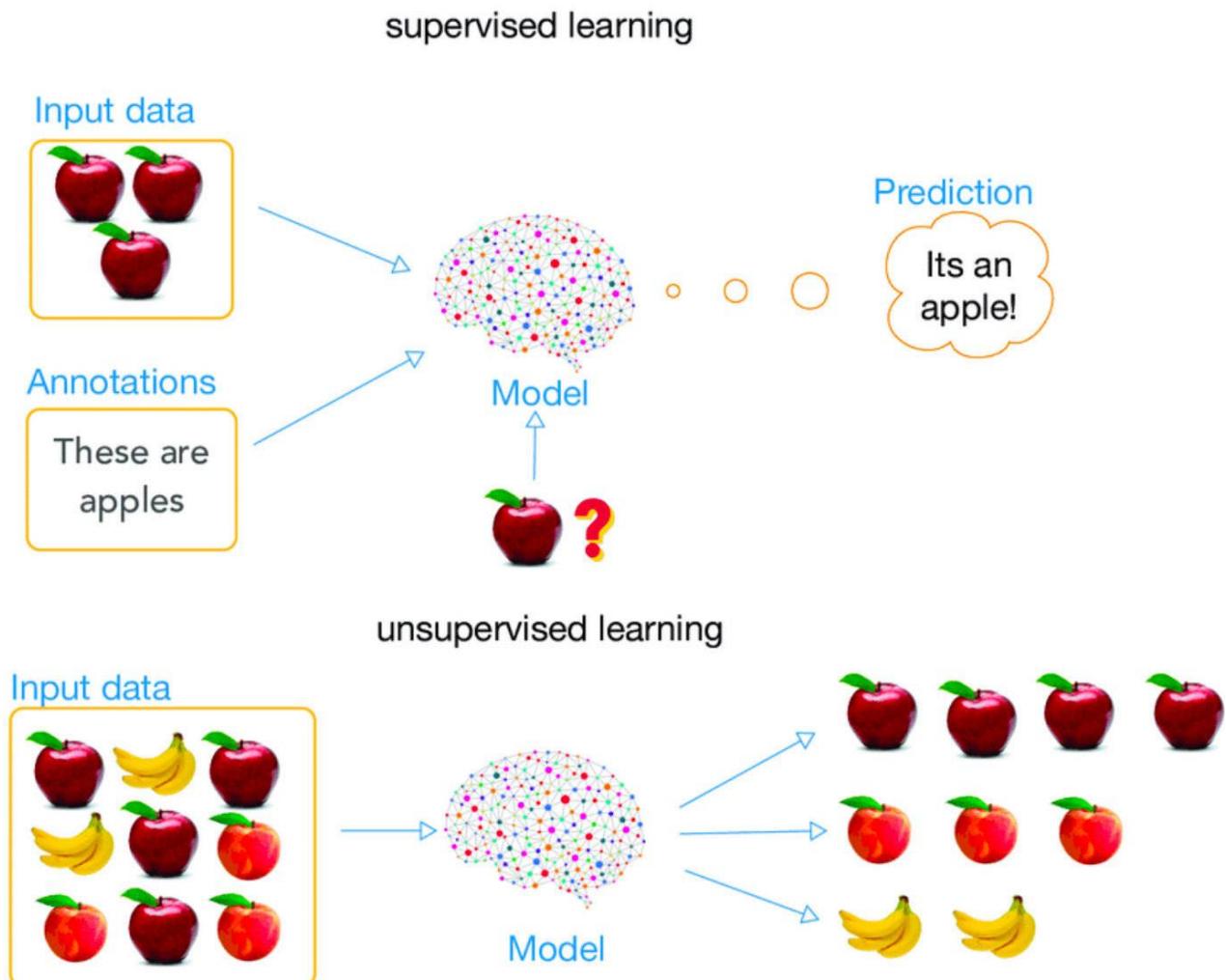
Tell me,
what is it ?



APPRENTISSAGE SUPERVISÉ VS NON-SUPERVISÉ

■ Apprentissage supervisé :

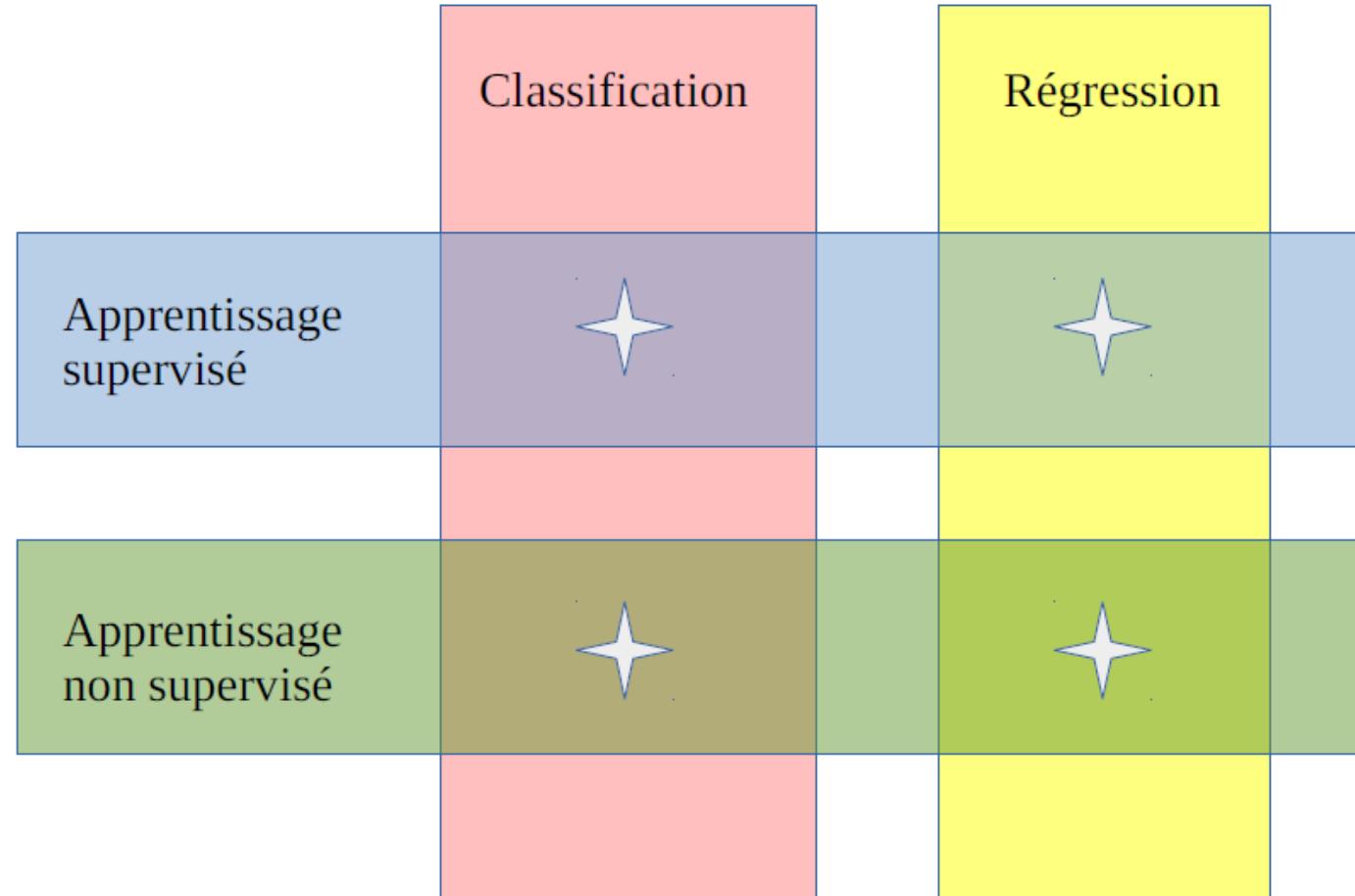
- Nécessite un jeu d'entraînement X, y
 - X : prédicteurs
 - y : variable à prédire



■ Apprentissage non supervisé :

- Nécessite un jeu d'entraînement X
- L'algorithme "décide" quelles sont les caractéristiques importantes

LES GRANDES FAMILLES

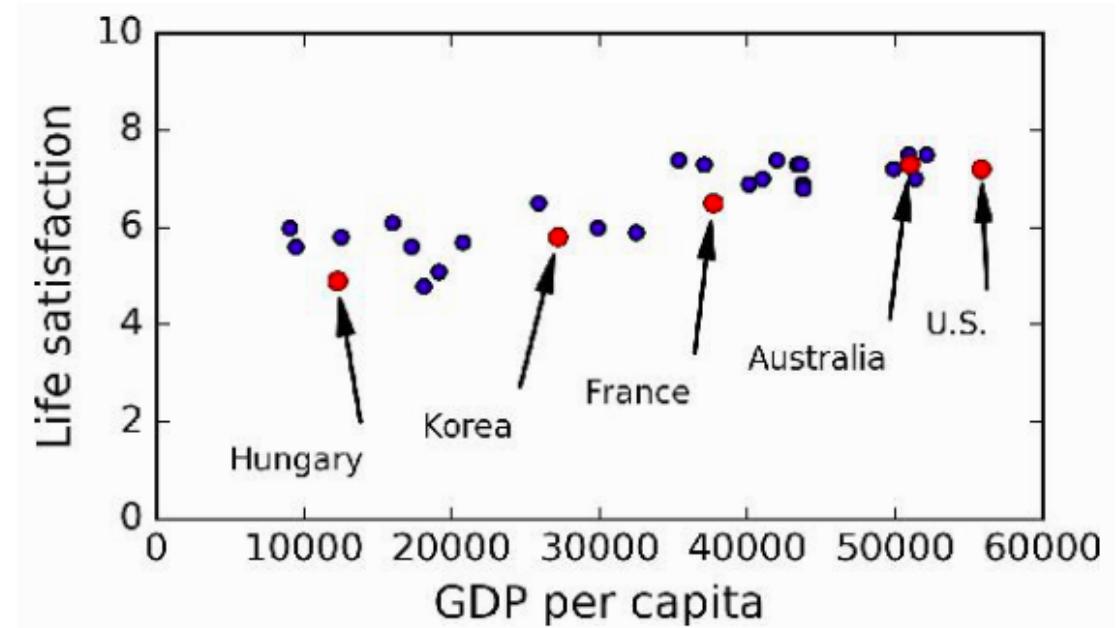


Les principaux algorithmes

EXEMPLE : EST-CE QUE L'ARGENT REND HEUREUX ?

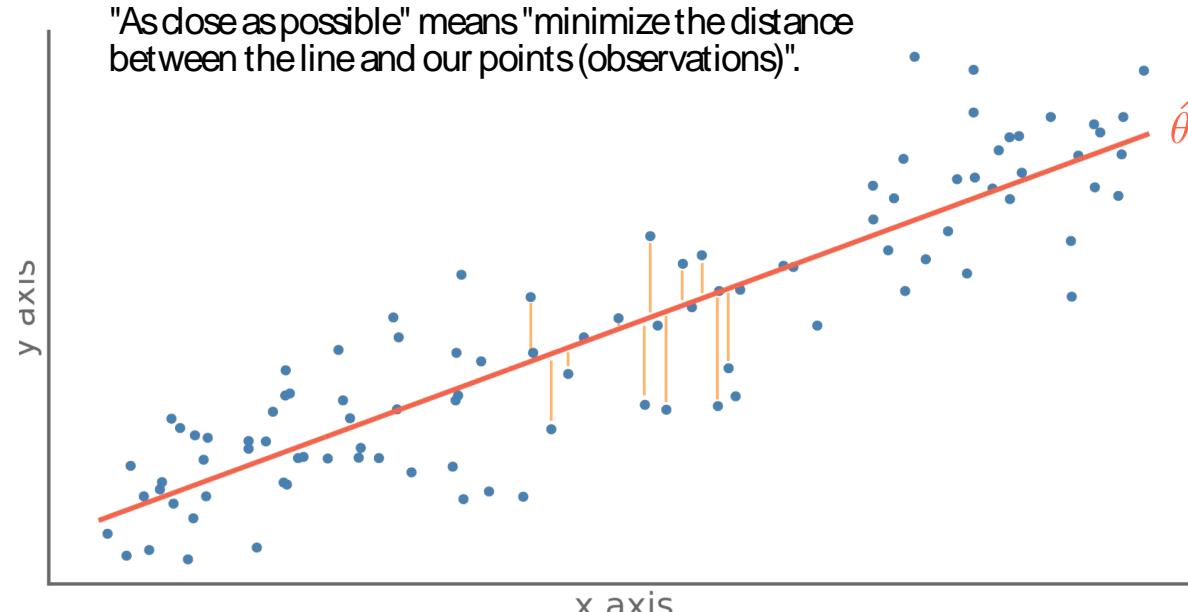
- Idée : croiser la rente per capita des pays et leur index de satisfaction
- Est-ce qu'on observe une tendance ?

Country	GDP per capita (USD)	Life satisfaction
Hungary	12,240	4.9
Korea	27,195	5.8
France	37,675	6.5
Australia	50,962	7.3
United States	55,805	7.2



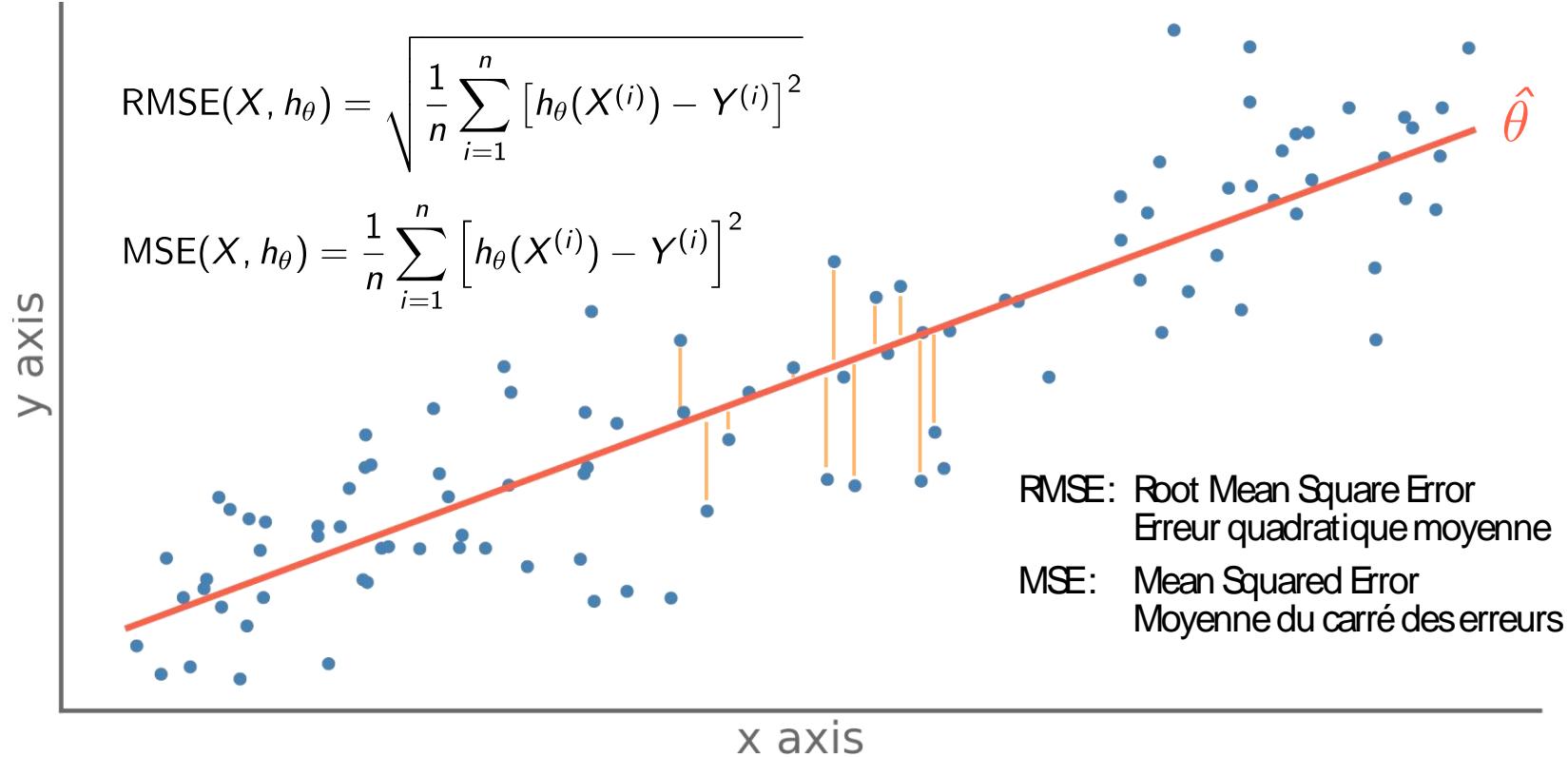
LA RÉGRESSION LINÉAIRE

- Le modèle suppose une fonction de prédiction de forme
 - $f(x) = a_1x_1 + \dots + a_px_p + b = a \cdot x + b$
- L'apprentissage consiste à calculer les coefficients a et b qui minimisent l'erreur de prédiction (coût)
- Mais comment définir le coût ?



LA FONCTION DE COÛT (LOSS)

- C'est l'écart moyen entre les prédictions et la vérité terrain

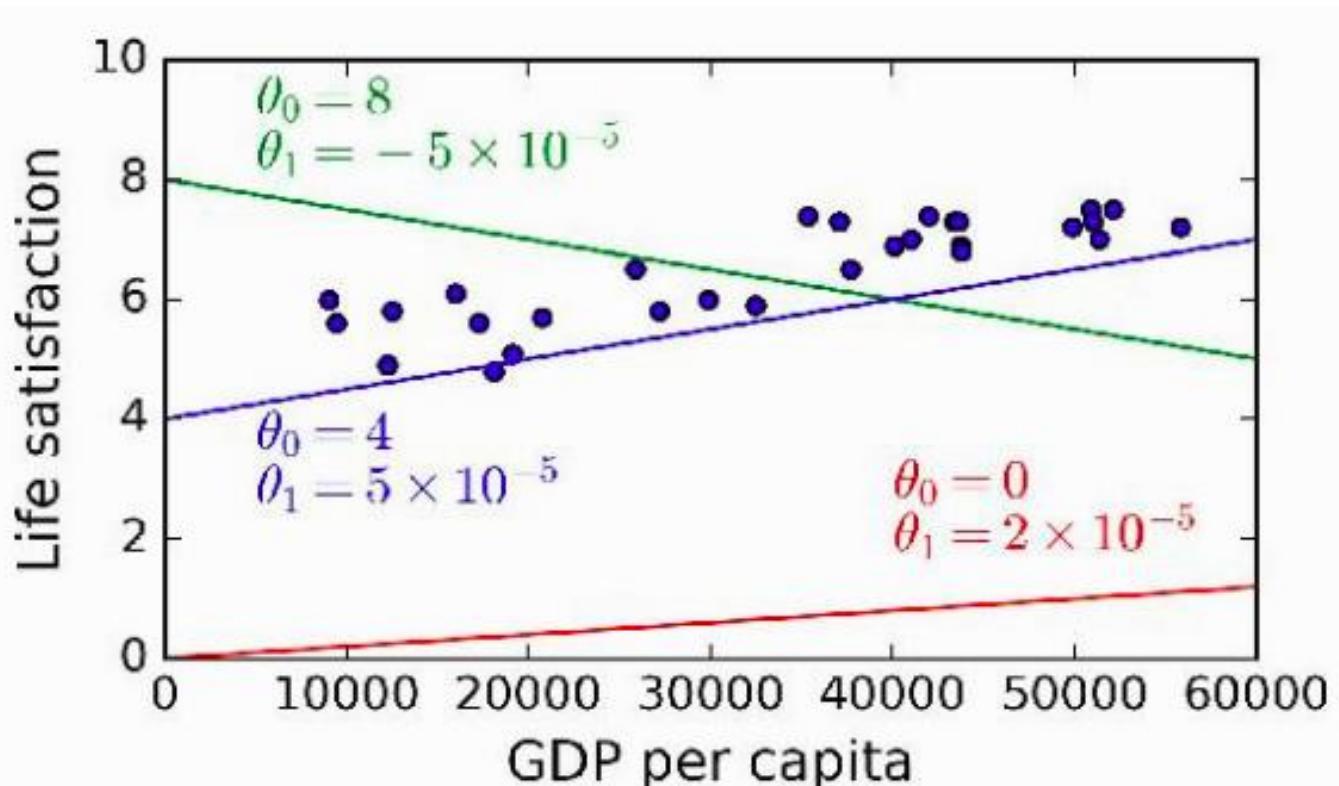


MAE : Mean Absolute Error
Erreur absolue moyenne

$$MAE(X, h_{\theta}) = \frac{1}{n} \sum_{i=1}^n |h_{\theta}(X^{(i)}) - Y^{(i)}|$$

EXEMPLE : EST-CE QUE L'ARGENT REND HEUREUX ?

- Ce modèle a deux paramètres, θ_0 et θ_1 .
- On peut créer une formule simple et essayer plusieurs combinaisons
 - $satisfaction = \theta_0 + \theta_1 \times GDP_per_capita$



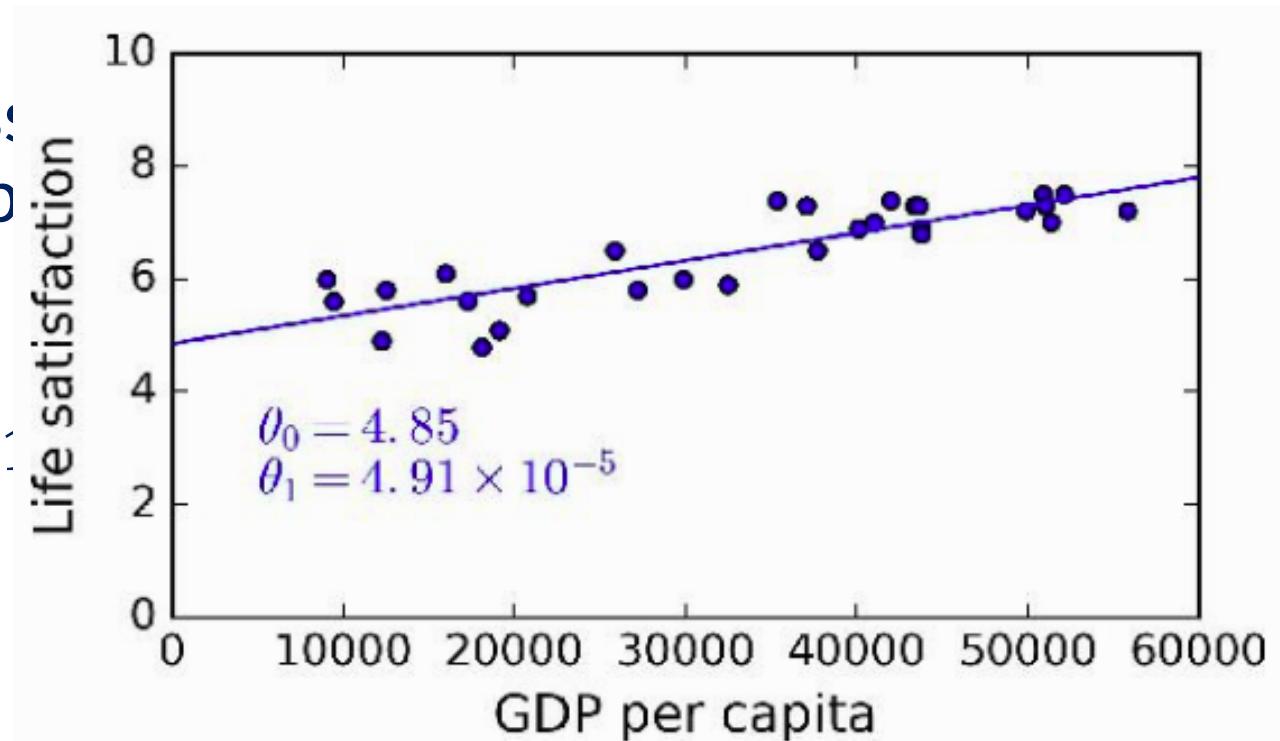
EXEMPLE : EST-CE QUE L'ARGENT REND HEUREUX ?

- Le modèle qui mieux correspond aux données est

- $\theta_0 = 4.85$
- $\theta_1 = 4.91 \times 10^{-5}$

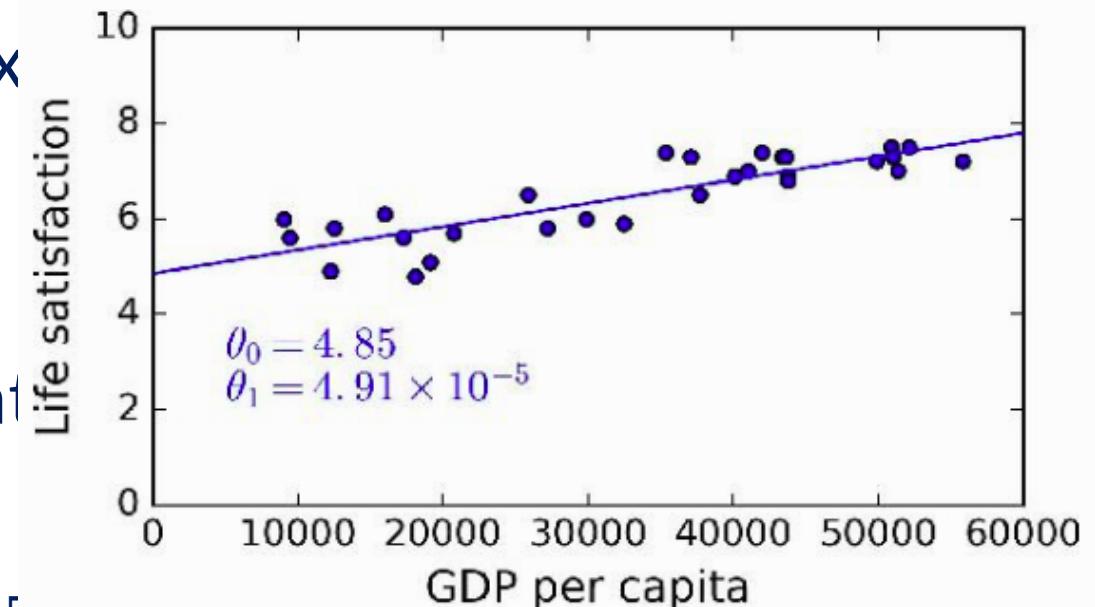
- Grâce à ce modèle, on peut essayer d'estimer la satisfaction de la population de Chypre

- GDP_per_capita = 22875 USD
- Estimation = $4.85 + 22875 * 4.91 \times 10^{-5}$
- Estimation = 5.96



EXEMPLE : EST-CE QUE L'ARGENT REND HEUREUX ?

- Le modèle qui mieux correspond aux
 - $\theta_0 = 4.85$
 - $\theta_1 = 4.91 \times 10^{-5}$
- Grâce à ce modèle, on peut essayer d'estimer la satisfaction de la population de Chypre
 - GDP_per_capita = 22875 USD
 - Estimation = $4.85 + 22875 * 4.91 * 10^{-5}$
 - Estimation = 5.96
- Problème : complexité $O(n^3)$
 - Tester plein de combinaisons de θ_0 et θ_1

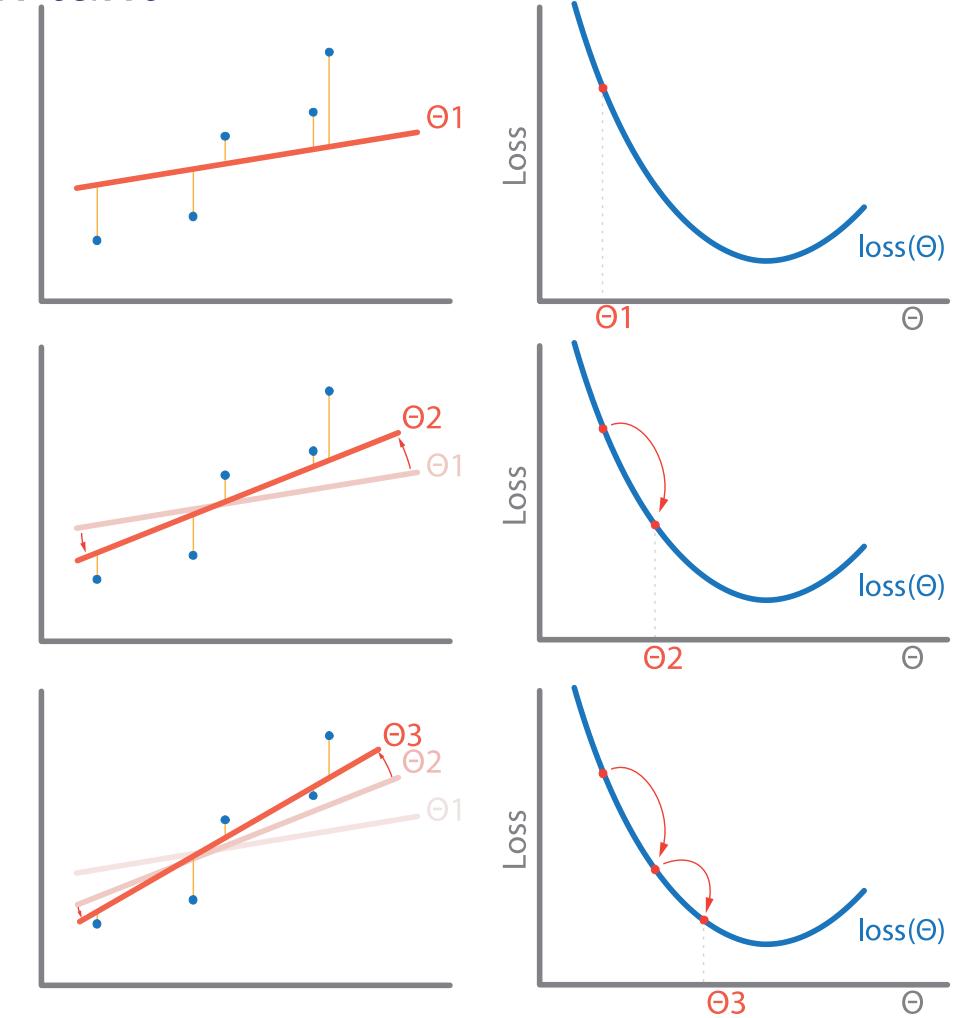


Comment accélérer la recherche de la solution ?

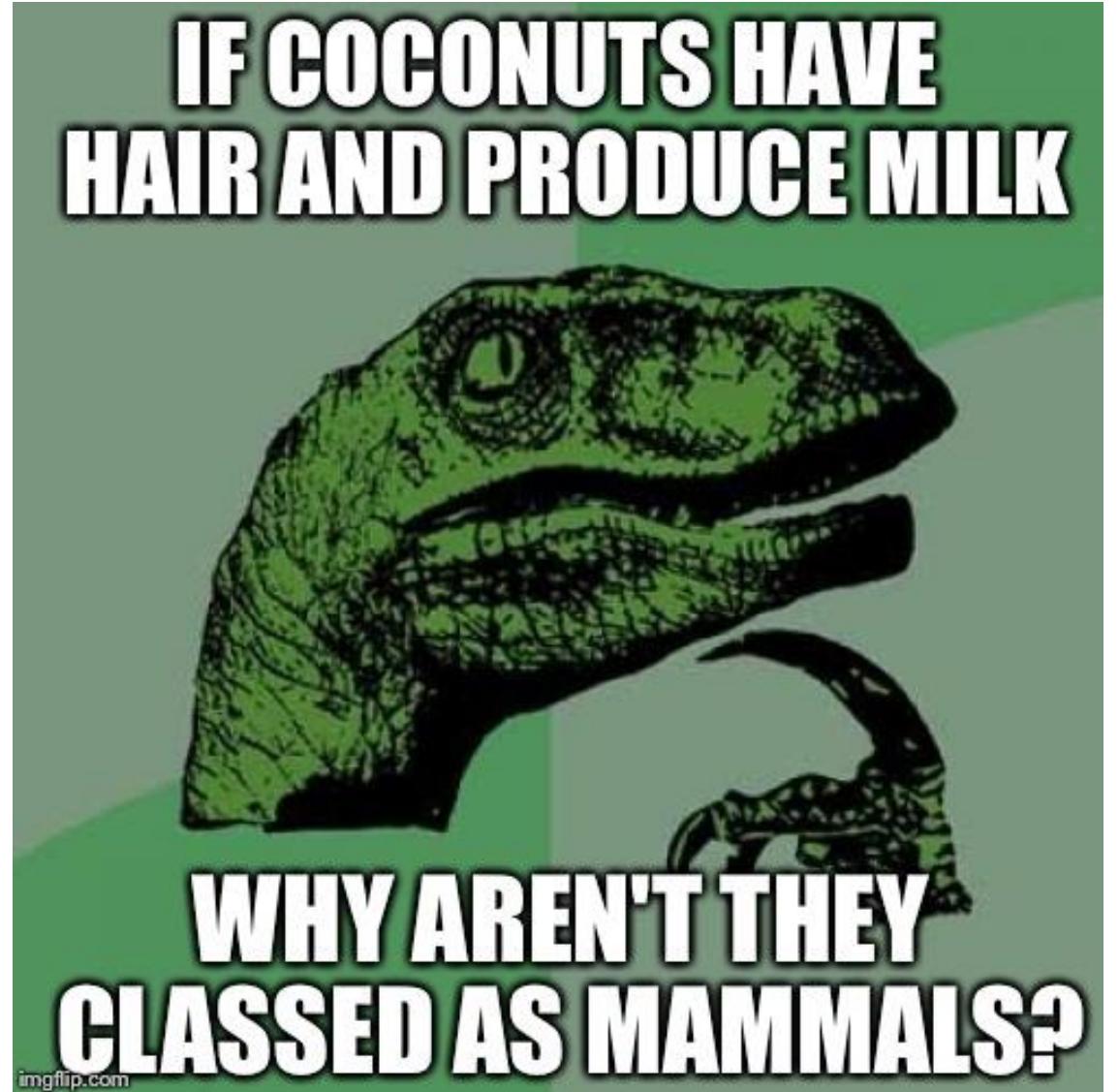


FAISANT FACE À LA DESCENTE

- La descente de gradient permet de prendre l'élan tant que la pente nous est favorable
 - Avancer par des petits sauts, répéter plusieurs fois

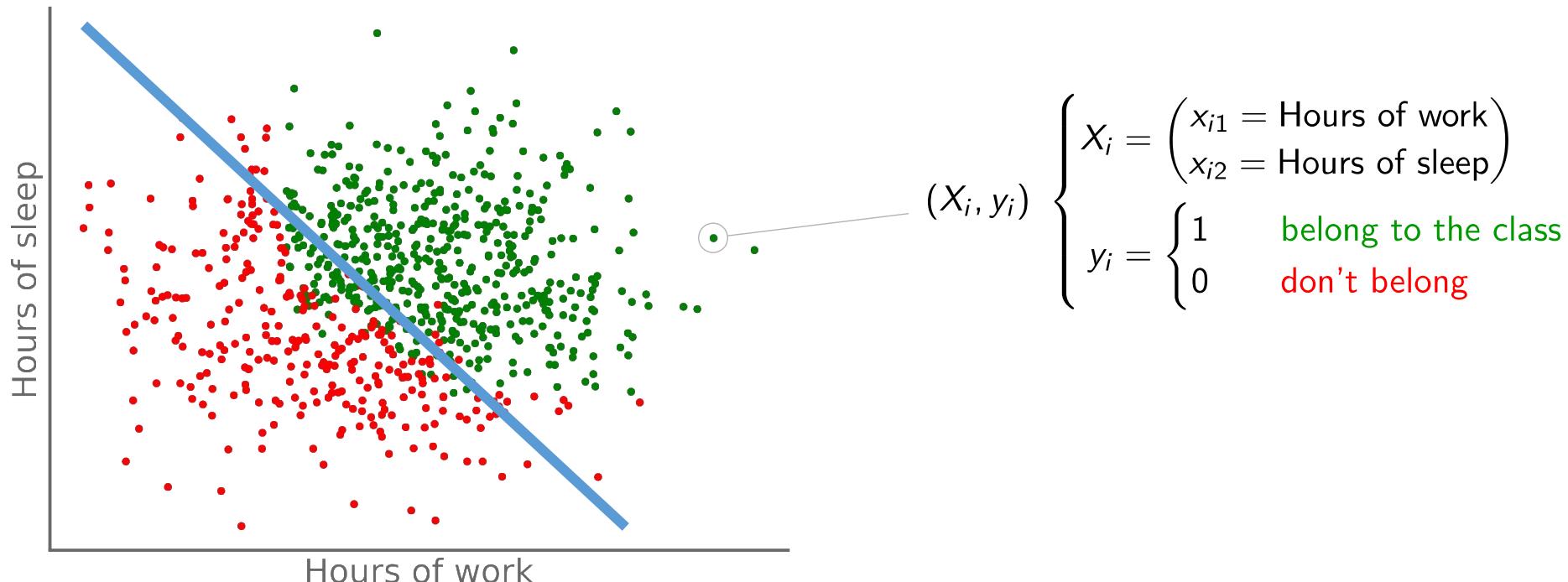


De la Régression à la Classification



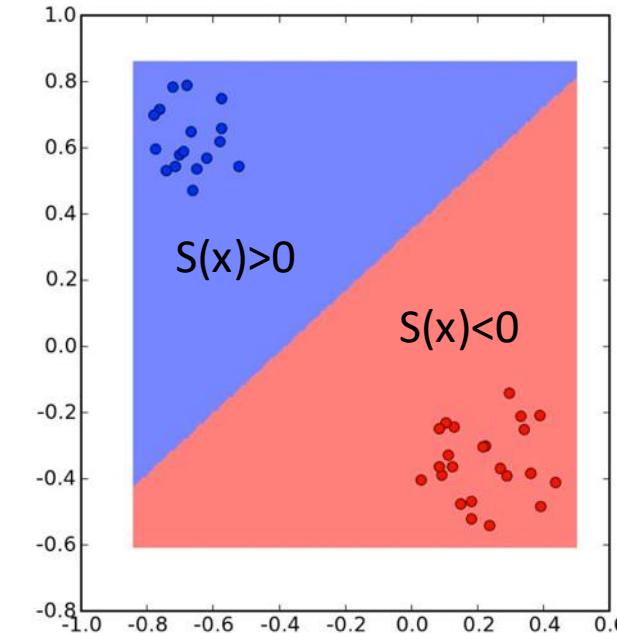
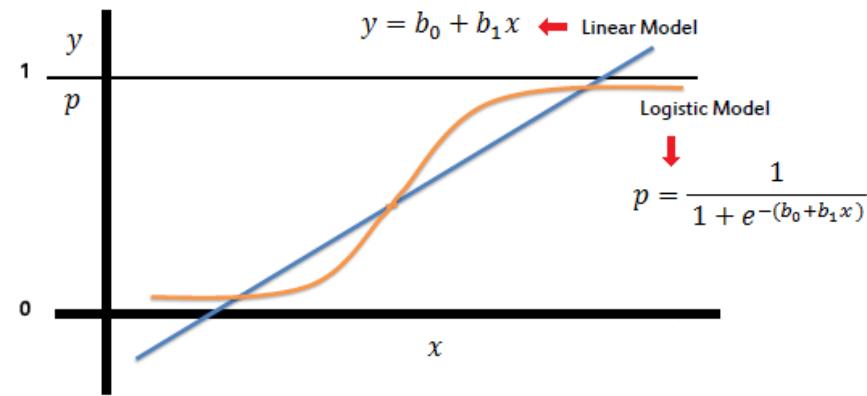
LE CAS DE LA RÉGRESSION LOGISTIQUE

- Au lieu de trouver la ligne qui est la moins éloignée des points, **on cherche la ligne qui mieux les sépare**
 - Une Régression Logistique sert à donner une probabilité d'appartenance à une classe



RÉGRESSION LOGISTIQUE

- Comme pour la régression linéaire, on cherche une fonction $S(x) = a_1x_1 + \dots + a_px_p$ appelée **Score** qui doit délimiter deux groupes (classes) de données
- Le principe est de trouver des coefficients a de manière à ce que la valeur de $S(x)$ soit positive lorsque les chances d'appartenir au groupe 1 sont grandes, et $S(x)$ est négative si la probabilité est grande pour le groupe 0
- Une fonction d'interpolation $\text{logit}(S) = 1/[1 + \exp(-S)]$ est souvent utilisée pour exprimer cette probabilité



EXEMPLE : PROBABILITÉ DE PASSER LES EXAMENS

- Dans cet exemple issu de Wikipedia, on a un tableau avec 20 étudiants, indiquant combien d'heures ils ont étudié et s'ils ont passé les examens ou pas

Hours	0.50	0.75	1.00	1.25	1.50	1.75	1.75	2.00	2.25	2.50	2.75	3.00	3.25	3.50	4.00	4.25	4.50	4.75	5.00	5.50
Pass	0	0	0	0	0	0	1	0	1	0	1	0	1	0	1	1	1	1	1	1

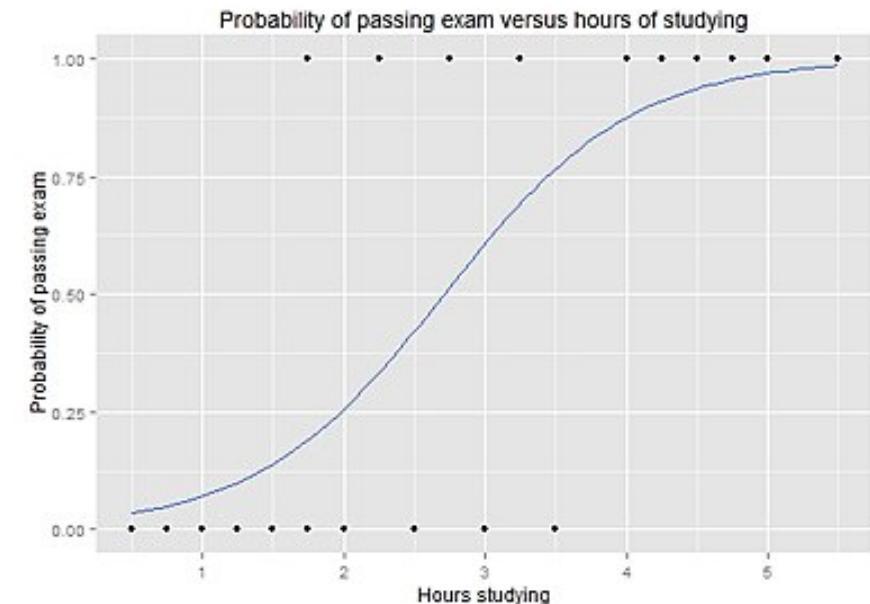
- L'analyse de ce graphique donne une fonction score

$$s(x) = 1.5046 \times \text{Hours} - 4.0777$$

- Ainsi, la probabilité de passer un examen est donné par

- $$\text{prob} = \frac{1}{1+\exp(-(1.5046 \cdot \text{Hours} - 4.0777))}$$

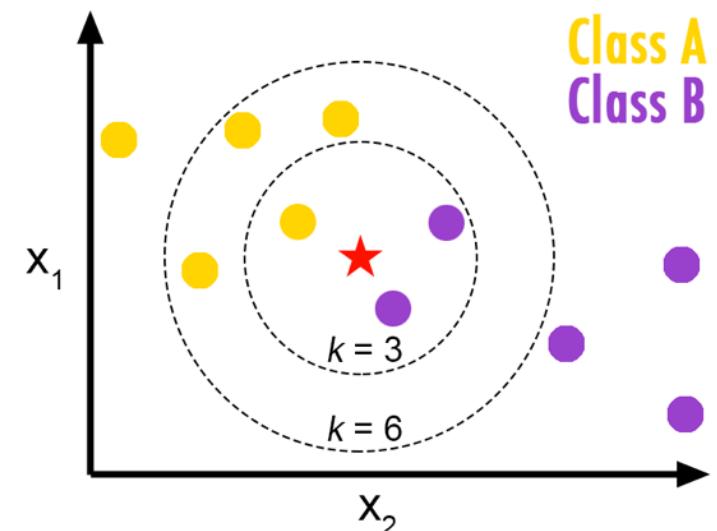
- Pour un étudiant qui s'est dédié uniquement 2h, la probabilité est de 26%
- Un étudiant qui a révisé 4h a 87% de probabilité de réussir



Graph of a logistic regression curve showing probability of passing an exam versus hours studying

FAIRE UNE RÉGRESSION AVEC UNE CLASSIFICATION

- L'algorithme KNN (pour *K Nearest Neighbors*) est à la base un algorithme de classification supervisé
 - On représente les observations (données) dans un espace aux dimensions des variables prédictives
 - Ces données sont étiquetées
 - Pour une nouvelle donnée, on cherche l'étiquette qui est la plus fréquente dans un rayon autour de cette nouvelle entrée (les *K* premiers voisins)
 - Besoin d'un concept de distance
 - Euclidienne ?



EXEMPLE D'USAGE KNN

- On reprend l'exemple sur l'indice de satisfaction des pays
- Au lieu d'essayer d'estimer les paramètres θ_0 et θ_1 , on regarde les pays avec des rentes proches de celle de Chypre (22587 USD)
 - K=1
 - La Slovénie a la rente per capita la plus proche (20732 USD)
 - Son indice de satisfaction est de 5.7
 - K=3
 - En plus de la Slovénie, on trouve aussi le Portugal (19122 USD, sat=5.1) et l'Espagne (25865 USD, sat=6.5). Si on fait une moyenne, on obtient un indice de 5.77
 - Pas si loin de l'indice 5.96 obtenu par la régression linéaire

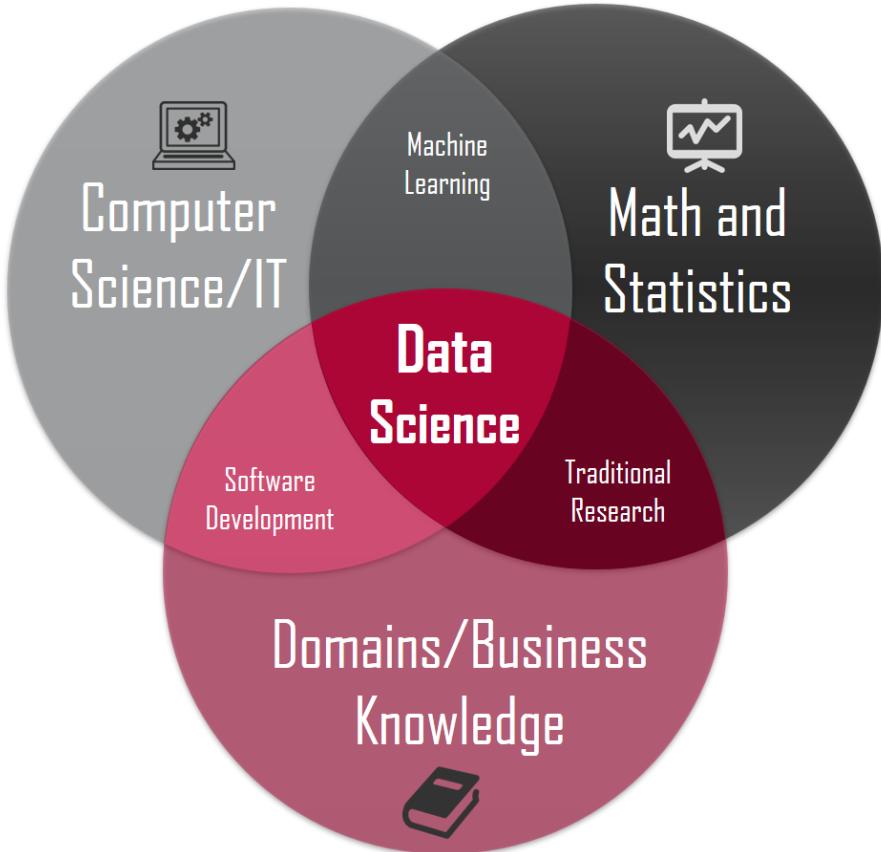
Et les données ?

Intro rapide à des structures de données en Python

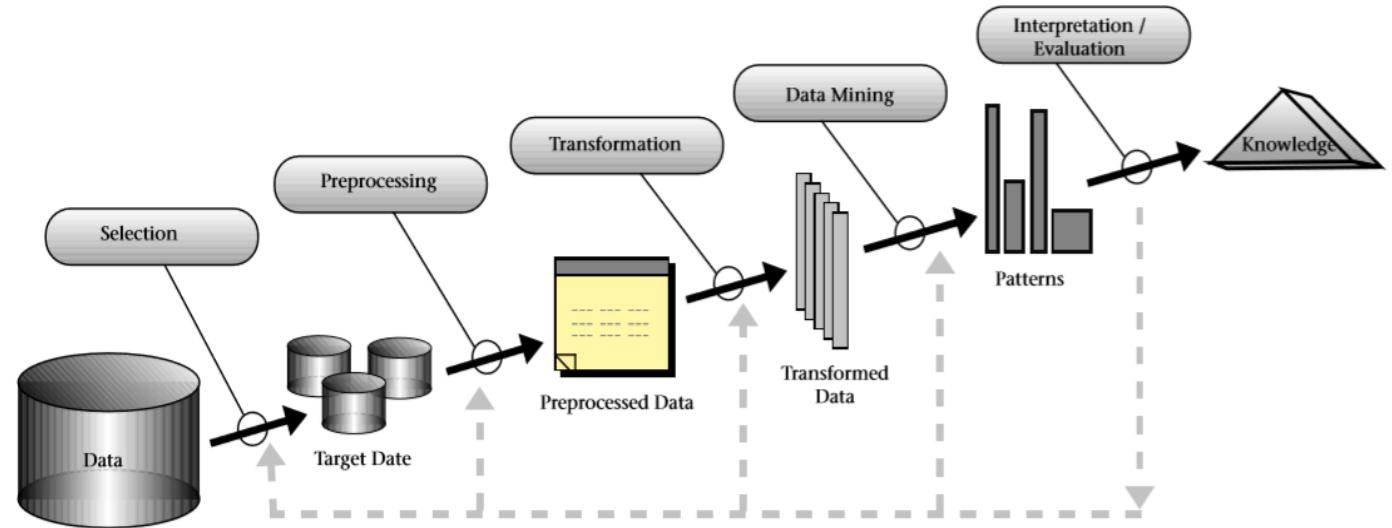
LES PREMIERS 80% DU TRAVAIL

- Souvent il est dit que la préparation des données représente les premiers 80% du travail
- Les outils Big Data ou même les bases de données ne sont que la partie ETL
 - Extraction
 - Transformation
 - Load (chargement)
- Une fois les données prêtes, il faut savoir les utiliser/analyser (les autres 80%)
 - Utilisation directe (bases de données, requêtes SQL)
 - Analyse des données (Machine Learning, Data Mining)
- Il faut aussi les visualiser (les troisièmes 80% ?)
- Dans le cas du Machine Learning, un langage s'est imposé : Python

DATA SCIENCE / DATA MINING



La fouille de données est un ensemble de techniques et méthodes permettant l'analyse et l'exploitation de données afin d'en extraire des connaissances.



Processus de découverte de connaissance KDD

Source : Fayyad et al., 1996

DIFFÉRENTES ÉTAPES AVANT LE MODÈLE IA

- Obtention des données
- Nettoyage et sélection
- Formatage
- Exploratory Data Analysis (EDA)
- Séparation des données
 - Groupe d'entraînement (train)
 - Groupe de validation (val)
 - Groupe de test (test)

TOUR DE TABLE

- Pour démarrer notre exploration du machine learning, nous avons besoin de quelques outils
 - Notebooks Jupyter
 - Les bibliothèques Pandas et Scikit-Learn



NOTEBOOK JUPYTER

■ C'est quoi un Notebook Jupyter ?

- Document « actif » contenant des **blocs de texte** (en format « *markdown* ») et des **blocs de code Python**

The screenshot shows a Jupyter Notebook interface with the following components:

- Bloc de texte Texte libre et formaté**: A green box pointing to the first text cell containing "Series".
- Bloc de code Code Python et résultat d'exécution**: A green box pointing to the code cell "Entrée [4]" and its output.
- Code Python**: A green box pointing to the code cell "Entrée [5]".
- Résultat d'exécution**: A green box pointing to the output cell "Entrée [5]".

Series

La classe Series (sur Pandas) permet la manipulation des séries de valeurs indexées (bref, une séquence de valeurs de type *key, value*), particulièrement appréciées pour la manipulation des séries temporelles.

Entrée [4]:

```
from pandas import Series
maserie = Series ([8, 70, 320, 1200], index=["Suisse", "France", "USA", "Chine"])
print(maserie)
```

Suisse 8
France 70
USA 320
Chine 1200
dtype: int64

On peut créer une série à partir d'une liste, d'un dictionnaire ou même d'un Array NumPy.

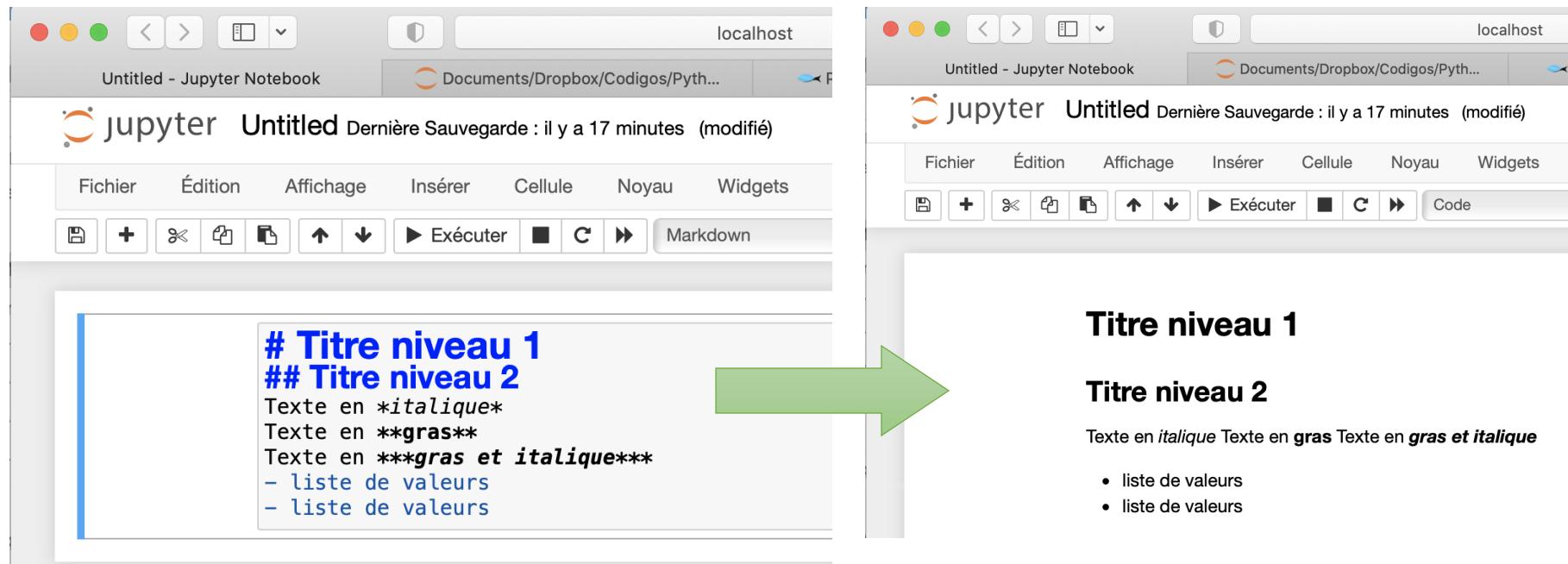
Entrée [5]:

```
import numpy as np
monarray = np.random.randn(5)
s2 = Series (monarray, index=["A","B","C","D","E"])
print (monarray, '\n', s2)
```

[-0.96850466 -1.99478449 -1.6655539 -1.22318014 -0.35159315]
A -0.968505
B -1.994784

NOTEBOOK

- Texte en « markdown »
 - Langage de marquage simple pour formater le texte



NOTEBOOK

■ Bloc de code Python

- Des petits **blocs de code** qu'on peut exécuter
- Equivalent au mode « **itératif** »

Attention : ça dépend de l'ordre dans laquelle **on exécute les blocs, pas nécessairement de l'ordre des blocs**

The screenshot shows a Jupyter Notebook interface with a toolbar at the top and two code cells below.

Toolbar: Fichier, Édition, Affichage, Insérer, Cellule, Noyau, Widgets, A. The "Exécuter" button is highlighted with a green oval.

Cell 1 (Left):

Titre niveau 1

Titre niveau 2

Texte en *italique* Texte en **gras** Texte en **gras et italique**

- liste de valeurs
- liste de valeurs

Entrée [2]:

```
a = 2
b = 'To'
print (a*b)
```

A large green arrow points from the bottom of Cell 1 to the top of Cell 2.

Cell 2 (Right):

Titre niveau 1

Titre niveau 2

Texte en *italique* Texte en **gras** Texte

- liste de valeurs
- liste de valeurs

Entrée [2]:

```
a = 2
b = 'To'
print (a*b)
```

ToTo

Google Collab



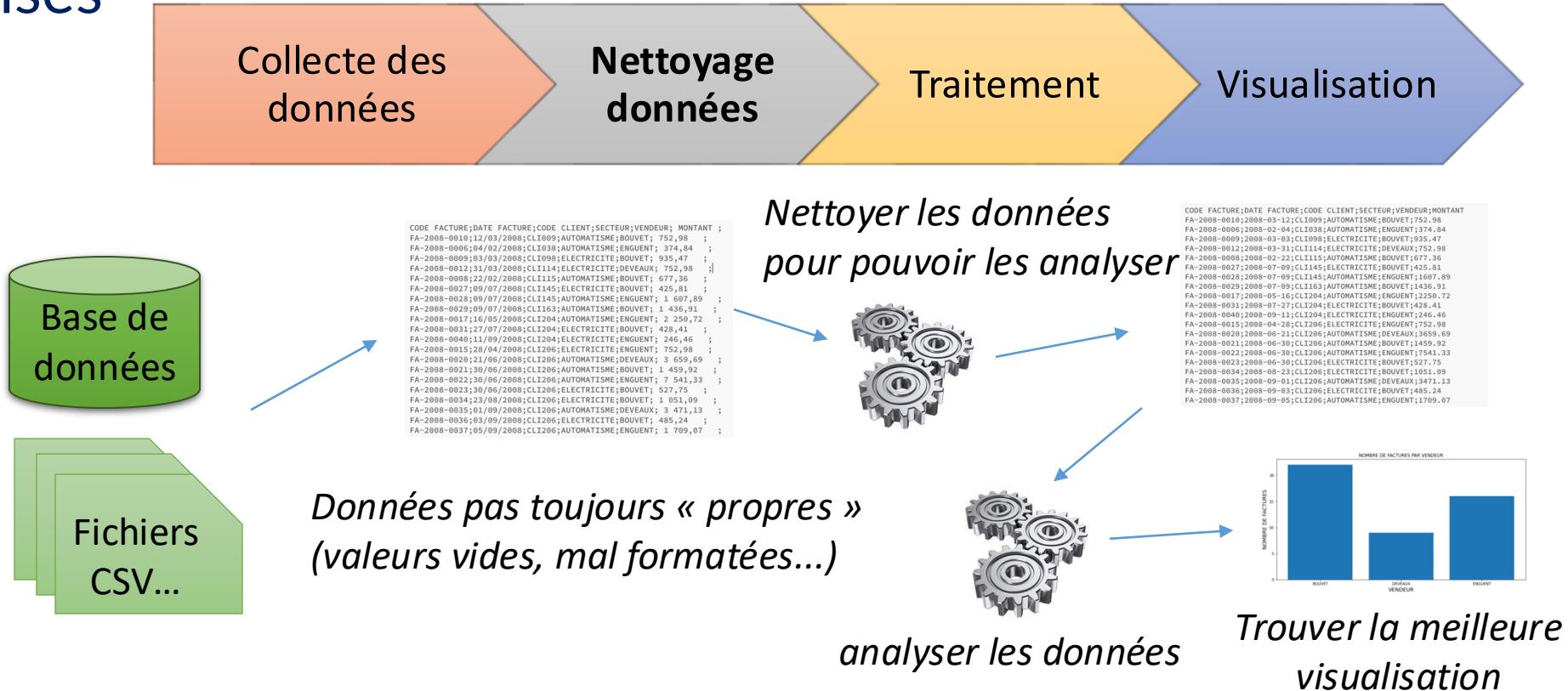
Accessible avec un simple compte Google
<https://colab.research.google.com>

- On peut créer ses Notebooks et les enregistrer sur son Google Drive.
- Partage d'un Notebook avec d'autres personnes possible

Manipuler des données avec Pandas

ÉTAPES DE TRAITEMENT DES DONNÉES

- Les projets d'analyse de données suivent un ensemble d'étapes bien précises



MANIPULATION DE DONNÉES TABULAIRES

- Pandas permet de stocker et traiter des données en format tableau
 - Ex : lire un fichier excel

Excel = Pandas !

The diagram illustrates the mapping of an Excel spreadsheet to Pandas data structures. It features three boxes: 'variable à prédire' (top left), 'Colonnes' (top right), and 'observations (X,Y)' (bottom right). Arrows point from these boxes to specific parts of the Excel table. The 'variable à prédire' arrow points to the 'Gross Profit' column header. The 'Colonnes' arrow points to the first column of headers ('Estimated Revenue', 'Estimated Costs', etc.). The 'observations (X,Y)' arrow points to the bottom row of data.

	E	F	G	H	I	J	K	L	M	N	O	P	Q	
1														
2			Total Contract			From Inception to June 13, 2016			At June 13, 2016			For the Period Ended June 13, 2016		
3	Estimated Revenue	Estimated Costs	Estimated Gross Profit	Earned Contract Revenue	Contract Costs	Gross Profit	Contract Billings	Estimated Costs to Complete	Percent Complete	Under (Over) Billings	Earned Contract Revenue	Contract Costs	Gross Profit (Loss)	
4	29,831,262	22,771,956	7,059,306	12,113,470	9,246,924	2,866,546	11,987,630	15,525,052	41%	125,840	3,740,588	2,855,269	885,319	
5	4,765,875	3,915,859	850,016	4,761,592	3,912,340	849,252	4,748,777	3,519	100%	12,815	319,663	185,925	133,738	
6	3,165,949	2,635,676	530,273	3,073,180	2,558,445	514,735	3,092,332	77,231	97%	(19,152)	1,212,380	1,019,868	192,512	
7	6,845,696	5,348,200	1,497,496	5,935,890	4,637,414	1,298,476	5,727,306	710,786	87%	208,584	2,985,189	2,344,782	640,407	
8	3,202,917	2,139,767	1,063,150	3,197,769	2,136,328	1,061,441	3,199,414	3,439	100%	(1,645)	386,839	241,974	144,865	
9	3,267,627	2,402,206	865,421	3,122,086	2,295,211	826,875	3,143,402	106,995	96%	(21,316)	254,751	101,060	153,691	
10	3,513,815	2,260,925	1,252,890	2,839,759	1,827,211	1,012,548	2,573,819	433,714	81%	265,940	1,823,265	1,173,159	650,106	
11	3,913,079	3,104,573	808,506	3,591,755	2,849,640	742,115	3,503,374	254,933	92%	88,381	2,651,445	2,039,028	612,417	
12	12,187,491	13,500,000	(1,312,509)	2,193,165	3,505,674	(1,312,509)	2,476,537	9,994,326	26%	(283,372)	2,193,165	3,505,674	(1,312,509)	
13	3,274,077	2,798,357	475,720	35,779	30,580	5,199	0	2,767,777	1%	35,779	35,779	30,580	5,199	
14	3,835,139	4,296,527	(461,388)	2,578,713	3,040,101	(461,388)	2,386,461	1,256,426	71%	192,252	2,578,713	3,040,101	(461,388)	
15	13,500,000	10,227,273	3,272,727	8,553,041	6,479,577	2,073,464	8,321,142	3,747,696	63%	231,899	8,553,041	6,479,577	2,073,464	
16	3,849,262	3,137,190	712,072	274,615	223,814	50,801	1,741,936	2,913,376	7%	(1,467,321)	274,615	223,814	50,801	
17	74,614,943	64,402,779	10,212,164	46,921,464	41,803,708	5,117,756	43,715,328	22,599,071			29,854,173	27,271,295	2,582,878	
18	169,767,132	142,941,288	26,825,844	99,192,278	84,546,961	14,645,311	96,614,458	58,394,321			(651,316)	36,863,606	50,512,106	6,351,500
19														

LECTURE D'UN FICHIER

- On peut lire le **contenu d'un fichier** vers un **DataFrame**
- Différents **formats** : CSV, Excel, JSON, SQL...
- Opérations **read_xxx** : **read_csv**, **read_excel**, **read_sql**...

Fichier CSV

CODE FACTURE;DATE FACTURE;CODE CLIENT; MONTANT
FA-2008-0010;2008-03-12;CLI009; 752.98
FA-2008-0006;2008-02-04;CLI038; 374.84

Données séparées par ;
(séparateur)

Autres options possibles

Par ex. si index est une date :

parse_dates=True

dayfirst=True

ventes = pnd.read_csv('files/VentesAgenceU.csv',

Séparateur delimiter=';',

header=[0],

Ligne(s) d'en-tête index_col=[0])

Index

EXEMPLE MANIPULATION DE FICHIER CSV

- Pour créer un DataFrame à partir d'un **fichier csv**, il suffit de faire
 - `df = pd.read_csv('nomfichier')`
- Des options pour gérer les entêtes, la transformation des données, etc.
 - `pd.read_csv('tmp.csv', index_col=[0], parse_dates=[0], header=None)`
 - `pd.read_csv('tmp.csv', index_col='Date', parse_dates=True, delimiter=';')`
- On peut aussi **exporter sur un fichier** avec `to_csv('nomfichier')`
 - On peut aussi lire/exporter Excel, JSON, HTML, HDF5, ...

IMPORTATION À PARTIR D'AUTRES SOURCES

- Excel :
 - Selon le système il faut installer bibliothèques xlrd/xlwt et openpyxl à part
- Deux possibilités
 - `pd.read_excel()`
- `credit = pd.read_excel("fichier.xls", sheetname="donnees", usecols="A:E")`
- `pd.ExcelFile()`
- JSON : déjà inclus dans Pandas
 - `jdata = pd.read_json("fichier.json")`

PRINCIPAUX INDICATEURS D'UN FICHIER

```
ventes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 50 entries, FA-2008-0010 to nan
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   DATE FACTURE    47 non-null    object  
 1   CODE CLIENT     47 non-null    object  
 2   SECTEUR         47 non-null    object  
 3   VENDEUR         47 non-null    object  
 4   MONTANT         47 non-null    object  
 5   Unnamed: 6       0 non-null    float64
dtypes: float64(1), object(5)
memory usage: 2.7+ KB
```

Une fois le fichier lu, il faut vérifier les informations :

- monDF.info ()
- monDF.describe ()
- monDF.head (n)
- monDF.tail (n)

info : informations sur le DF

```
[1]:
```

head : premières lignes du DF

```
[2]: ventes.head()
```

```
[2]:
```

CODE FACTURE	DATE FACTURE	CODE CLIENT	SECTEUR	VENDEUR	MONTANT	Unnamed: 6
FA-2008-0010	12/03/2008	CLI009	AUTOMATISME	BOUDET	752,98	NaN
FA-2008-0006	04/02/2008	CLI038	AUTOMATISME	ENGUENT	374,84	NaN
FA-2008-0009	03/03/2008	CLI098	ELECTRICITE	BOUDET	935,47	NaN
FA-2008-0012	31/03/2008	CLI114	ELECTRICITE	DEVEAUX	752,98	NaN
Master CHPS	FA-2008-0008	22/02/2008	CLI115	AUTOMATISME	BOUDET	677,36

Signes de problèmes :

- Mauvais types de données
- NaN / NaT

PROBLÈMES CONSTATÉS

```
[4]: ventes.head(10)
```

```
[4]:
```

	DATE FACTURE	CODE CLIENT	SECTEUR	VENDEUR	MONTANT	Unnamed: 6
CODE FACTURE						
FA-2008-0010	12/03/2008	CLI009	AUTOMATISME	BOUVET	752,98	Nan
FA-2008-0006	04/02/2008	CLI038	AUTOMATISME	ENGUENT	374,84	Nan
...			
FA-2008-0029	09/07/2008	CLI163	AUTOMATISME	BOUVET	1_436,91	Nan
FA-2008-0017	16/05/2008	CLI204	AUTOMATISME	ENGUENT	2_250,72	Nan
FA-2008-0031	27/07/2008	CLI204	ELECTRICITE	BOUVET	428,41	Nan

Colonne vide

```
[5]: ventes.tail(10)
```

```
[5]:
```

	DATE FACTURE	CODE CLIENT	SECTEUR	VENDEUR	MONTANT	Unnamed: 6
CODE FACTURE						
FA-2008-0014	18/04/2008	CLI312	AUTOMATISME	ENGUENT	917,38	Nan
...			
FA-2008-0005	26/01/2008	CLI520	ELECTRICITE	BOUVET	739,83	Nan
Nan	Nan	Nan	Nan	Nan	Nan	Nan
Nan	Nan	Nan	Nan	Nan	Nan	Nan
Nan	Nan	Nan	Nan	Nan	Nan	Nan

Chiffres mal formatés

```
[3]: ventes.describe()
```

```
[3]:
```

Unnamed: 6

count	0.0
mean	Nan
std	Nan
min	Nan
25%	Nan
50%	Nan
75%	Nan
max	Nan

NaN NaN

Lignes vides

COMMENT NETTOYER LES DONNÉES ?

- Différentes opérations peuvent être nécessaires pour nettoyer les données
 - *Supprimer les lignes/colonnes contenant des cellules vides*
 - Remplacer les cellules vides
 - *Corriger les types des données*
 - Remplacer les « , » par des « . » dans les nombres
 - Supprimer les espaces vides des chiffres et des noms des colonnes
 - ...

Les opérations nécessaires varient en fonction de la qualité du Dataset et de l'analyse

TRAITER LES CASES VIDES

- Les cases vides sont indiquées par des valeurs spéciales :
 : **NaN et NaT**

- On peut soit les remplacer par une autre valeur : **fillna**

- `monDF.fillna(value = { 'A': 0 } , inplace = True)`

valeurs à remplacer

Dictionnaire { colonne : valeur }

modifie le DataFrame (True)

ou non (False)

- Soit les supprimer avec **dropna**

- Différentes options : `inplace=True, axis='index' / 'columns'`,
`how = all`

Supprime si toutes
les valeurs sont vides

- `monDF.dropna(axis='index', inplace = True)`

- `monDF.dropna(axis='columns' , how ='all', inplace = True)`

Supprime les lignes ou les colonnes
contenant les valeurs vides

TRAITER LES TYPES DE DONNÉES

- Les types de données reconnus à la lecture du fichier ne correspondent pas toujours à la réalité
- Il faut parfois convertir les données vers les **bons formats**
- Plusieurs opérations Panda peuvent le faire
 - `pnd.to_datetime`
 - `pnd.to_numeric`
 - `pnd.to_timedelta`



CONVERTIR DU TEXTE VERS DES DATES

- Convertir un texte (type « object ») en date (« daytime »)

```
dfDates['Debut'] = pd.to_datetime(dfDates['Debut'], yearfirst=True)
```

*to_datetime ne modifie pas les données.
Il faut donc les réaffecter au DataFrame.*

Colonne contenant les valeurs à convertir

*Indication format
yearfirst : yyyy-mm-dd
dayfirst : dd-mm-yyyy*

	pnd.to_datetime (dfDates['Fin'], dayfirst=True)			
	pnd.to_datetime (dfDates['Debut'], yearfirst=True)		pnd.to_datetime (dfDates['Examens'], format='%m%d%Y')	
	Debut	Fin	Examens	Semestre
0	2021-09-13	17/12/2021	01042022	S1
1	2022-01-24	23/04/2022	05042022	S2

TRANSFORMATION DES DONNÉES TEXTE EN NUMÉRIQUE

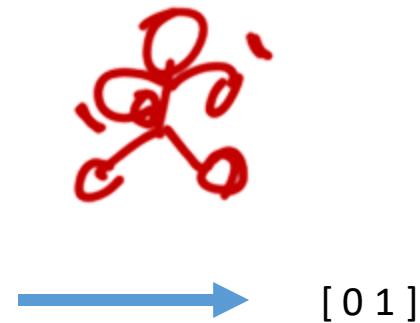
- Difficulté à résoudre :
 - Lorsque certaines **colonnes** contiennent des **textes** (des « **catégories** »), il faut les « **traduire** » en **valeurs numériques**
 - Exemples :
 - Genre : « **male** » / « **female** »,
 - Navigateurs : « **Safari** » / « **Chrome** » / « **Firefox** »
 - Les différents modèles de ML proposés sur **SKLearn** ne savent pas travailler avec de texte
- Solution possible :
 - Utiliser des « **encodeurs** »
 - Les encodeurs **traduisent des catégories exprimées en texte en format numérique**
 - Différents types de ces « **encodeurs** » existent
 - Proposés par la bibliothèque **Sklearn**, mais aussi par la bibliothèque **Pandas**

TRANSFORMATION DES DONNÉES TEXTE EN NUMÉRIQUE

- Transformation des données en Sklearn

- SKLearn ne manipule que des données numériques
- On considère que le texte indique des catégories
 - Pas un texte « libre »

[*'approves'* , *'disapproves'*]



- Encoders sur Sklearn

- Différents *encoders* disponibles sur `sklearn.preprocessing`
- **LabelEncoder** : Transformation des **labels** (target) en valeurs **entières** (0 à n-1)
- **OrdinalEncoder** : Transformation des **données** (features) en valeurs **entières** (de 0 à n-1)
- **OneHotEncoder** : Transformation des **données** (features) en valeurs **binaires**
- Quel que soit l'encoder, on suit les mêmes étapes :

Création encodeur → entraînement (« fit ») → application (« transform »)

TRANSFORMATION DES DONNÉES TEXTE EN NUMÉRIQUE

Données « symboliques »				
	Catégories			
	sex	region	browser	vote
0	male	from US	uses Safari	approves
1	female	from Europe	uses Firefox	disapproves
2	female	from US	uses Safari	approves
3	male	from Europe	uses Safari	approves
4	female	from US	uses Firefox	disapproves
5	male	from Europe	uses Chrome	disapproves
6	female	from Asia	uses Chrome	approves
7	male	from Asia	uses Chrome	approves

Target (classes)
« Y_set »

`Y = labEnc.inverse_transform(Yenc)`

Transformation inverse
(des valeurs aux labels)

`['approves' 'disapproves' 'approves' 'approves' 'disapproves' 'disapproves'
'approves' 'approves']` ← Y_set : valeurs originales

`[0 1 0 0 1 1 0 0]` ← Y_enc : valeurs encodées

LabelEncoder

Conversion des targets en valeurs numériques

```
from sklearn.preprocessing import LabelEncoder
```

```
labEnc = LabelEncoder()  
labEnc.fit( Y_set )
```

Création et
entraînement de
l'encoder

`Yenc = labEnc.transform(Y_set)`

Transformation
des valeurs

`labEnc.classes_` ← Classes retrouvées
`['approves' 'disapproves']`

valeurs

TRANSFORMATION DES DONNÉES TEXTE EN NUMÉRIQUE

Données « symboliques »				
	Catégories			
	sex	region	browser	vote
0	male	from US	uses Safari	approves
1	female	from Europe	uses Firefox	disapproves
2	female	from US	uses Safari	approves
3	male	from Europe	uses Safari	approves
4	female	from US	uses Firefox	disapproves
5	male	from Europe	uses Chrome	disapproves
6	female	from Asia	uses Chrome	approves
7	male	from Asia	uses Chrome	

Features (variables) « X_set »

L'encoder attend un DataFrame avec un ensemble de features

X_set : valeurs originales

```
['male' 'from Europe' 'uses Safari']  
['female' 'from US' 'uses Firefox']  
['male' 'from Europe' 'uses Chrome']  
['female' 'from Asia' 'uses Chrome']
```

Xord : valeurs encodées

```
[1. 1. 2.]  
[0. 2. 1.]  
[1. 1. 0.]  
[0. 0. 0.]
```

OrdinalEncoder

Conversion des **features** en valeurs **entiers**

```
from sklearn.preprocessing import OrdinalEncoder
```

```
ordEnc = OrdinalEncoder()  
ordEnc.fit( X_set )
```

Création et entraînement de l'encoder

```
Xord = ordEnc.transform( X_set )
```

Transformation des valeurs

```
X = ordEnc.inverse_transform(Xord)
```

Transformation inverse (des valeurs aux données)

```
ordEnc.categories_
```

Liste des catégories

TRANSFORMATION DES DONNÉES TEXTE EN NUMÉRIQUE

Données « symboliques »

	Catégories			
	sex	region	browser	vote
0	male	from US	uses Safari	approves
1	female	from Europe	uses Firefox	disapproves
2	female	from US	uses Safari	approves



Chaque **colonne** sera « éclatée » en plusieurs, en fonction du nombre de **catégories** présentes.

sex	region	browser
[0. 1.]	[0. 0. 1.]	[0. 0. 1.]
[1. 0.]	[0. 1. 0.]	[0. 1. 0.]
[1. 0.]	[0. 0. 1.]	[0. 0. 1.]

OneHotEncoder

Conversion des **features** en valeurs **binaires**

```
from sklearn.preprocessing import OneHotEncoder
```

```
ohEnc = OneHotEncoder()  
ohEnc.fit( X_set )
```

Création et entraînement de l'encoder

```
Xoh = ohEnc.transform( X_set )
```

Transformation des valeurs

DataFrame avec les valeurs

```
X = ohEnc.inverse_transform(Xoh)
```

Transformation inverse (des valeurs aux données)

```
ohEnc.get_feature_names()
```

Liste des catégories

TRANSFORMATION DES DONNÉES TEXTE EN NUMÉRIQUE

■ *Encoding - Bibliothèque Pandas*

- La bibliothèque Pandas propose un *encoding* de type One Hot appelé « *get_dummies* »
- Pas besoin de phase d'entraînement, on l'utilise directement

```
df_weather.value_counts( df_weather['Description'] )
```

On a 3 **catégories** (« Normal », « Warm » et « Cold »),
chacune deviendra une nouvelle **colonne**.

Description		
Normal	4992	
Warm	2507	
Cold	2501	
		dtype: int64

Description
Cold
Warm
Normal
Cold
Cold

```
pnd.get_dummies ( df_weather )
```



	Description_Cold	Description_Normal	Description_Warm
	1	0	0
	0	0	1
	0	1	0
	1	0	0
	1	0	0

TRANSFORMATION DES DONNÉES TEXTE EN NUMÉRIQUE

Dans `OneHotEncoder`, on écrit
`OneHotEncoder(drop='first')`

■ Encoding - Bibliothèque Pandas

- On peut réduire le nombre de colonnes créées avec l'option « `drop_first = True` »

```
df_dummies = pd.get_dummies ( df_weather, drop_first=True )
```

On représente la 1^{ère} catégorie par la combinaison de 0 sur les autres

Data columns (total 9 columns):			
#	Column	Non-Null Count	Dtype
0	Temperature_c	10000	non-null float64
1	Humidity	10000	non-null float64
2	Wind_Speed_kmh	10000	non-null float64
3	Wind_Bearing_degrees	10000	non-null int64
4	Visibility_km	10000	non-null float64
5	Pressure_millibars	10000	non-null float64
6	Rain	10000	non-null int64
7	Description_Normal	10000	non-null uint8
8	Description_Warm	10000	non-null uint8

Nouvelles colonnes

On obtient 1 colonne en moins

	Description_Normal	Description_Warm
Là où on avait « 1 » sur la colonne « Description_cold »,	0	cold 0
on se retrouve juste avec « 0 » sur les autres colonnes.	0	1 0
	1	0 0
	0	0 0
	0	0 0

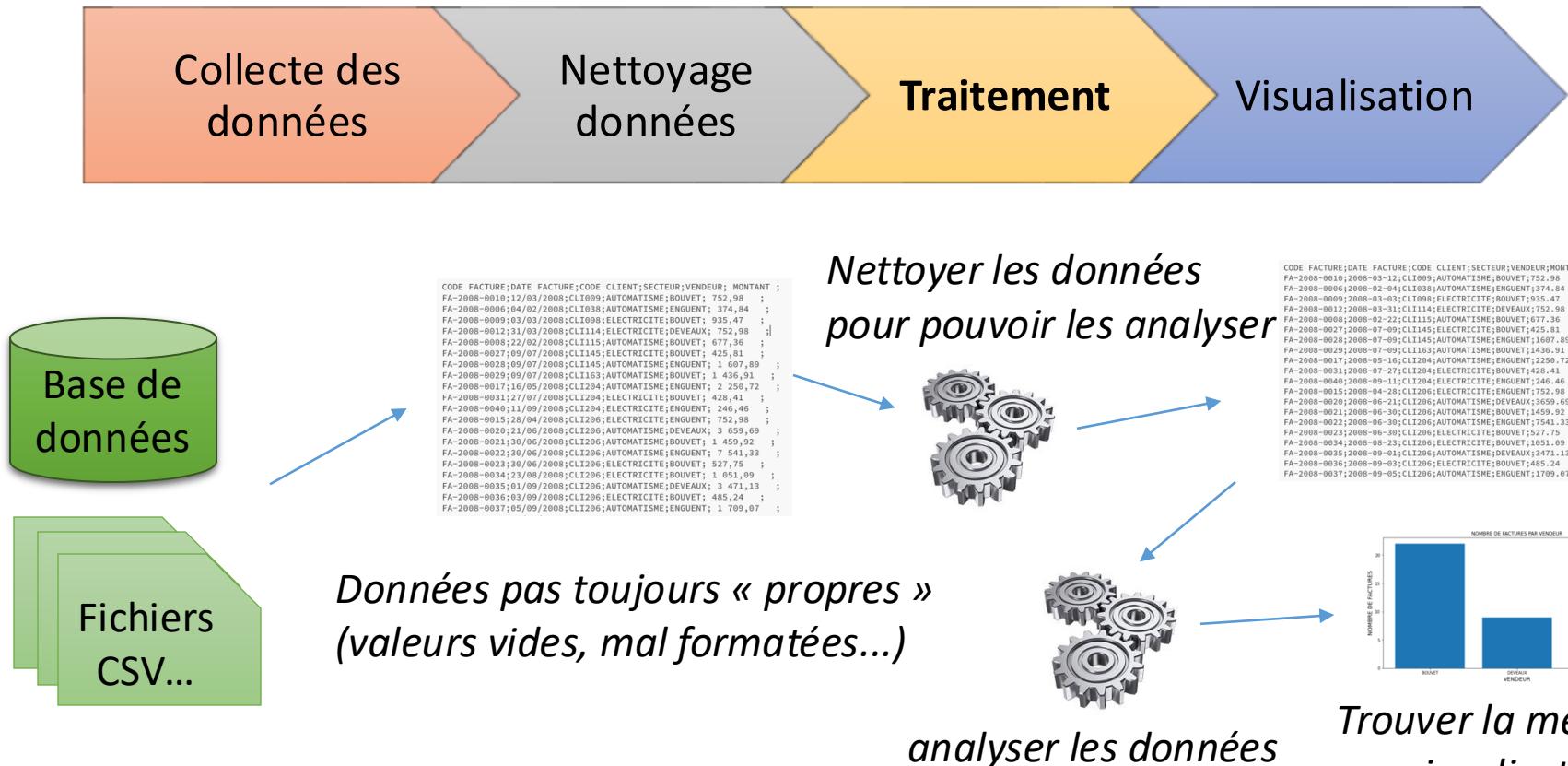
TRANSFORMATION DES DONNÉES TEXTE EN NUMÉRIQUE

- Pourquoi utiliser « **OneHotEncoder** » (ou « **get_dummies** ») plutôt qu'un « **OrdinalEncoder** » ?
 - **OrdinalEncoder** transforme les catégories dans une séquence des valeurs
 - « uses Safari » → 0, « uses Firefox » → 1, « uses Chrome » → 2
 - Cette séquence de valeurs va être interprétée comme telle par les algorithmes de Machine Learning
 - Par exemple : $0 < 1 < 2$
 - Safari < Firefox < Chrome ???
0 plus proche de 1 que de 2 Safari plus proche de Firefox que de Chrome ??
 - Cette interprétation peut induire à des conclusions erronées



ÉTAPES DE TRAITEMENT DES DONNÉES

- Les projets d'analyse de données suivent un ensemble d'étapes bien précises



TRAITEMENTS SIMPLES : MIN, MAX, SUM, MEAN, MEDIAN, COUNT...

■ Pandas offre différentes opérations d'analyse simples

- Somme (sum), moyenne (mean), médiane (median), écart type (std), variance (var), min, max, nombre d'éléments (count)...

ventes.sum(numeric_only=True) → MONTANT 79958.04
dtype: float64

ventes.mean(numeric_only=True) → MONTANT 1701.234894
dtype: float64

On ne prend en compte que les valeurs numériques

ventes.count() → DATE FACTURE 47
CODE CLIENT 47 count :
SECTEUR 47 compte le **nombre de lignes**
VENDEUR 47 ou le **nombre de colonnes**
MONTANT 47 (**axis='columns'**)

dtype: int64

TRAITEMENT : TROUVER LES DONNÉES

QUERY, GROUP BY

- Pour réaliser le traitement que l'on souhaite, il faut trouver les **bonnes données**

groupby

Regrouper des données en fonction de(s) **colonne(s)** choisie(s)

query

Récupérer des données

Alternative à loc

Syntaxe semblable à SQL

Requête semblable à SQL :

Colonne Opération Valeur

Opérateurs logiques

and (&)

or (|)

Attention aux `` et aux ''

'nom colonne avec espace'

'valeur string ou date'

ventes.query("MONTANT > 2000 and `DATE FACTURE` >= '2008-09-01'")

	DATE FACTURE	CODE CLIENT	SECTEUR	VENDEUR	MONTANT
CODE FACTURE					
FA-2008-0035	2008-09-01	CLI206	AUTOMATISME	DEVEAUX	3471.13
FA-2008-0041	2008-09-13	CLI222	AUTOMATISME	DEVEAUX	4374.79
FA-2008-0042	2008-09-15	CLI300	AUTOMATISME	ENGUENT	3667.68
FA-2008-0039	2008-09-09	CLI403	AUTOMATISME	BOUVET	3521.80

TRAITEMENT : TROUVER LES DONNÉES

QUERY, GROUP BY

- Pour réaliser le traitement que l'on souhaite, il faut trouver les **bonnes données**

groupby

Regrouper des données en fonction de(s) **colonne(s)** choisie(s)

query

Récupérer des données
Alternative à loc
Syntaxe semblable à SQL

```
ventes.groupby(by='VENDEUR').count()
```

*On regroupe les données par vendeur
(**by** = 'colonne')*

*Puis on compte le nombre de lignes (**count**)*

	DATE FACTURE	CODE CLIENT	SECTEUR	MONTANT
VENDEUR				
BOUVET	22	22	22	22
DEVEAUX	9	9	9	9
ENGUENT	16	16	16	16

TRAITEMENT : TROUVER LES DONNÉES QUERY, GROUP BY

```
ventes.query("VENDEUR == 'DEVEAUX'")
```

		DATE FACTURE	CODE CLIENT	SECTEUR	VENDEUR	MONTANT
	CODE FACTURE					
1	FA-2008-0012	2008-03-31	CLI114	ELECTRICITE	DEVEAUX	752.98
2	FA-2008-0020	2008-06-21	CLI206	AUTOMATISME	DEVEAUX	3659.69
3	FA-2008-0035	2008-09-01	CLI206	AUTOMATISME	DEVEAUX	3471.13
4	FA-2008-0045	2008-09-23	CLI206	AUTOMATISME	DEVEAUX	750.88
5	FA-2008-0024	2008-06-30	CLI209	AUTOMATISME	DEVEAUX	9367.87
6	FA-2008-0041	2008-09-13	CLI222	AUTOMATISME	DEVEAUX	4374.79
7	FA-2008-0004	2008-01-17	CLI235	AUTOMATISME	DEVEAUX	606.66
8	FA-2008-0033	2008-08-14	CLI300	AUTOMATISME	DEVEAUX	535.90
9	FA-2008-0030	2008-07-18	CLI403	ELECTRICITE	DEVEAUX	436.86

```
ventes.groupby(by='VENDEUR').count()
```

	DATE FACTURE	CODE CLIENT	SECTEUR	MONTANT
VENDEUR				
BOUVET	22	22	22	22
DEVEAUX	9	9	9	9
ENGUENT	16	16	16	16

Nombre de valeurs
retrouvées dans
chaque colonne
(count)
PAR Vendeur
(groupby)

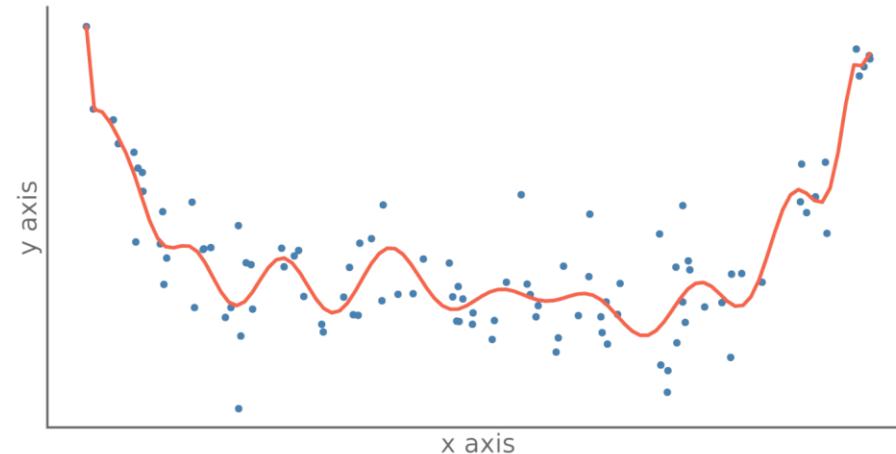
DISCRÉTISATION

- Transformer une variable quantitative (âge) en variable qualitative (classe d'âge)
- Deux fonctions dans Panda : **cut()** et **qcut()**
- Intervalles constants – on indique le nombre de classes, réparties entre min et max
 - `pd.cut(produits["prix"], bins=5)`
- Intervalles définis par l'utilisateur – on donne les bornes des intervalles
 - `pd.cut(produits["prix"], bins=[produits["prix"].min(), 50, 100, 500, produits["prix"].max()])`
- Intervalles de fréquence constante – nombre constant d'individus dans chaque classe
 - `pd.qcut(produits["prix"], q=5)`

DÉCOUPER SES DONNÉES

- Première impulse : utiliser toutes les données !!!
 - Pas bon, on risque de surentraîner (apprendre par cœur) et le modèle ne pourra pas généraliser

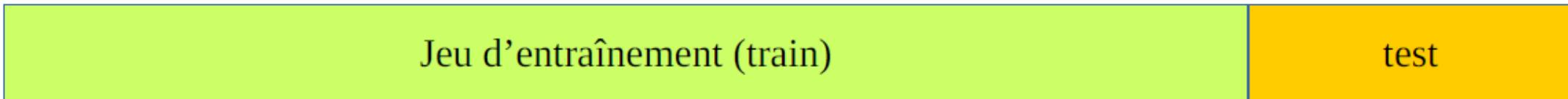
Jeu de données d'entraînement



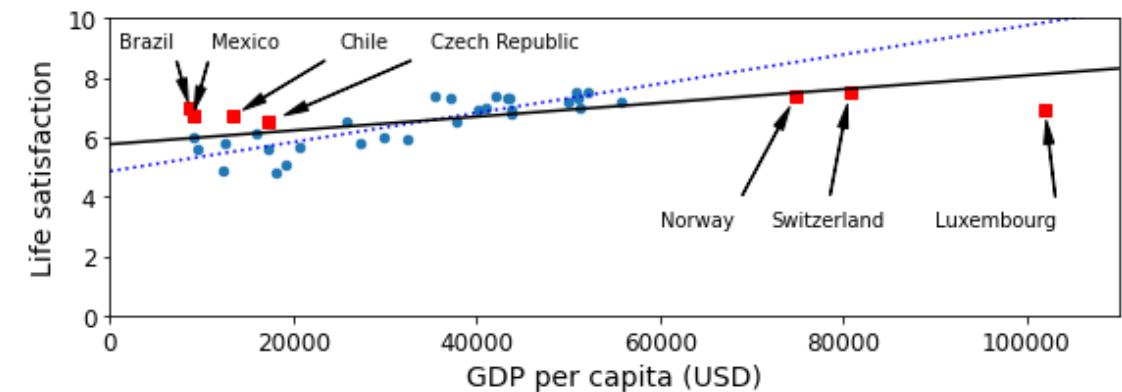
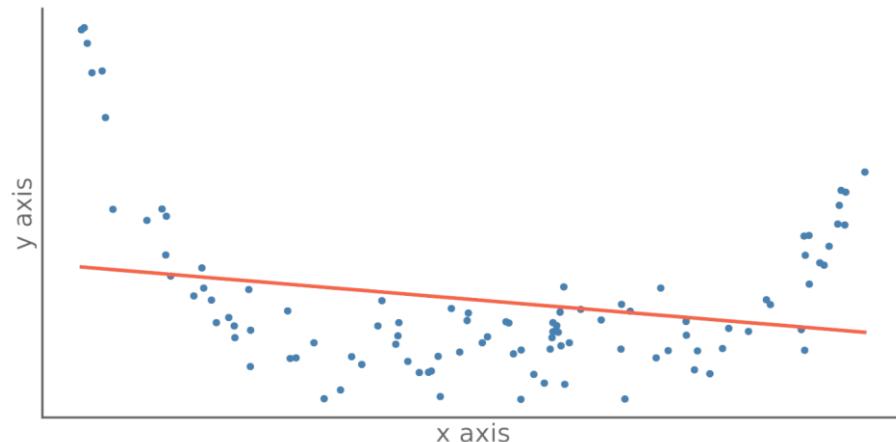
Overfitting

DÉCOUPER SES DONNÉES

- Séparer un groupe pour l'entraînement et autre pour la validation
 - *** CONFUSION : "val" est souvent appelé "test" si on découpe en 2 groupes



- Aucune garantie que l'algorithme fonctionnera bien avec de nouvelles données
 - Underfitting : modèle trop simple pour expliquer la variance



DÉCOUPER SES DONNÉES

- Approche recommandée (pas tjs suivie) :

1. entraîner sur le jeu d'entraînement (train),
2. vérifier si le modèle marche sur un jeu de validation (valid),
3. Répéter 1-2 jusqu'à ce que le modèle soit acceptable
4. Une fois le modèle entraîné, l'évaluer sur un jeu de test (test)

Jeu d'entraînement (train)	validation	test
----------------------------	------------	------

EVALUATION PAR VALIDATION CROISÉE

Consiste à réaliser k fois l'apprentissage du modèle en prenant un ensemble de test différent à chaque itération
→ permet d'éviter un biais lié à une mauvaise distribution des données

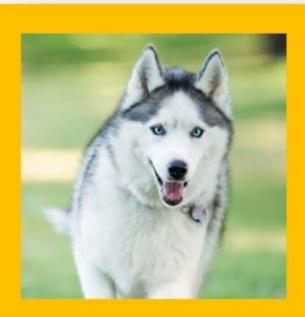


LES POINTS D'ATTENTION : LES BIAIS

l'IA amplifie
les biais

Raciste et détestable, l'intelligence artificielle tente de progresser

Depuis ses premiers pas, l'outil GPT-3 – une IA capable d'écrire des textes originaux – a fait l'admiration de tout un chacun. OpenAI, avec cette troisième évolution du projet parvenait à faire rédiger poésie, articles de presse ou même code de programmation. Cependant, les dérives sont rapidement arrivées, avec une Intelligence qui virait à la grossièreté, voire aux propos toxiques. Que faire ? Une bonne correction, tout simplement.



écolier



écolière



ATTENTION : DONNÉES PERSONNELLES

Contexte : RGPD

- **Transparence** : « Que fait-on de vos données »
- **Évaluation des risques** : évaluer et atténuer les risques de confidentialité à l'avance
- **Audits** : mieux éclairer la position de l'entreprise sur le plan de l'IA et la confidentialité. Difficulté de l'audit des algorithmes et de l'
- **Explicabilité**

ETHIQUE



Nous prenons au quotidien des décisions morales
Une voiture autonome va heurter une grand-mère et un enfant.
En déviant un peu, un des deux peut être sauvé. Qui choisir ?
Ou un fossé : Qui choisir entre les piétons et les passagers

ETHIQUE

2016, « the Next Rembrandt »

- un tableau de Rembrandt
- conçu par un ordinateur
- réalisé par une imprimante 3D,
351 ans après la mort du peintre.

2019, « la Symphonie inachevée ... achevé »

- Les 2 derniers mouvements de la Symphonie n°8
- la Symphonie inachevée que Franz Schubert
- commencée en 1822
quelques 197 années auparavant.



ETHIQUE

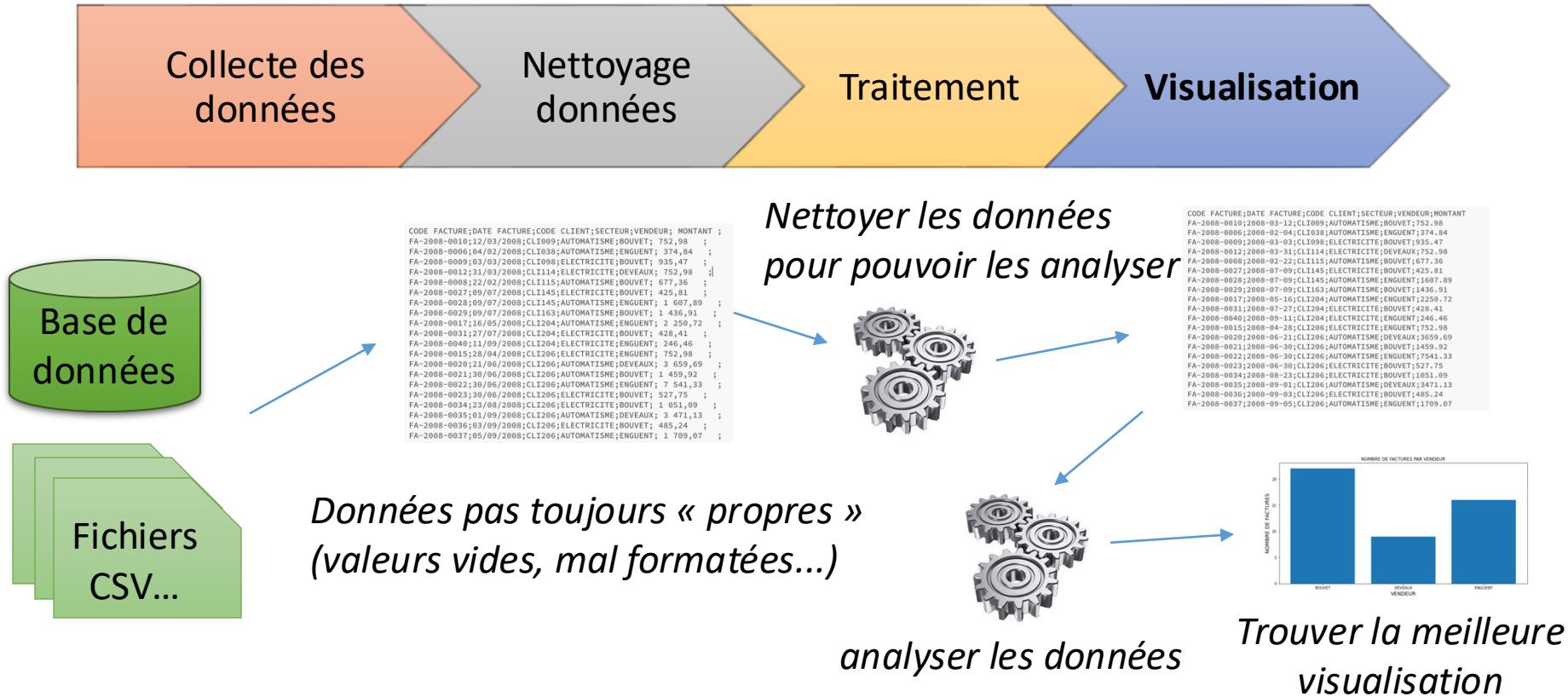


L'IA pourrait-elle évaluer des affaires judiciaires et appliquer la justice mieux qu'un juge ?

Les défis éthiques :

- Le manque de transparence des outils d'IA
- L'IA n'est pas neutre : les décisions inexactes, discriminatoires, ou de contenant des biais.
- Collecte de données & la protection de la vie privée.
- Equité et risques pour les droits humains et d'autres valeurs fondamentales.

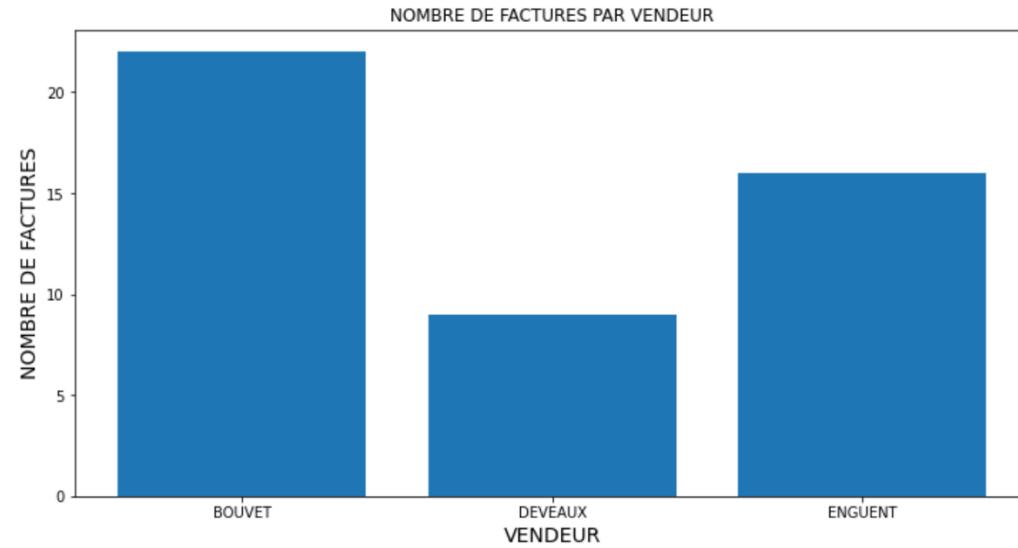
VISUALISATION DES DONNÉES



VISUALISATION

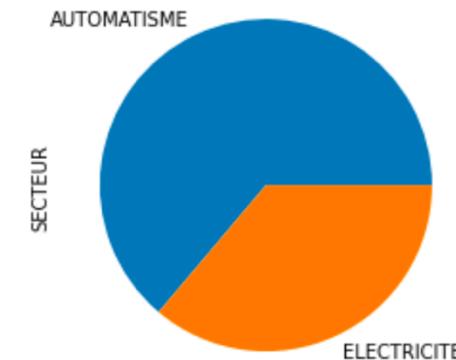
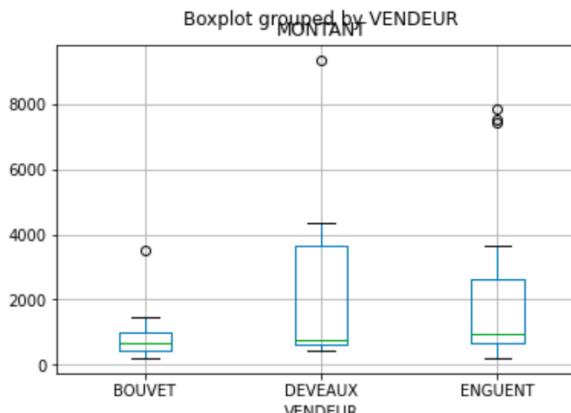
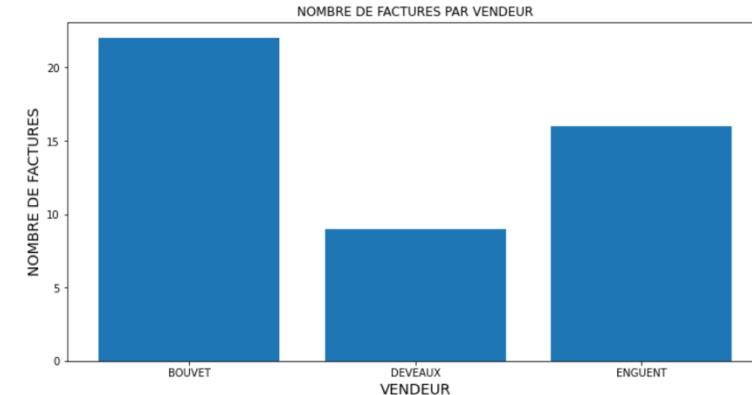
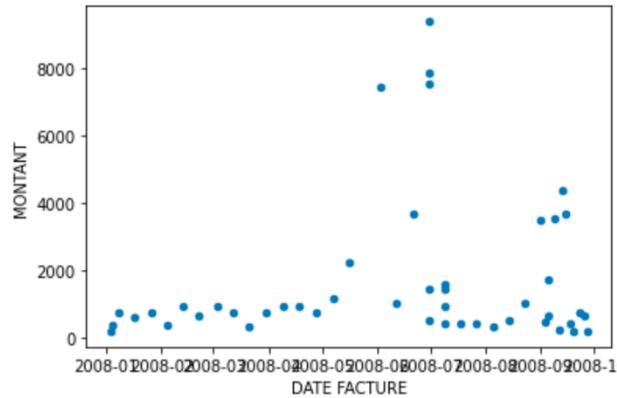
- La visualisation est une partie essentielle de l'analyse de données
- Elle est essentielle pour :
 - Comprendre ce qu'on cherche
 - Mieux comprendre les résultats
 - Mieux communiquer

```
VENDEUR
BOUVET      22
DEVEAUX      9
ENGUENT     16
dtype: int64
```



VISUALISATION AVEC MATPLOTLIB

- Matplotlib est une bibliothèque de **visualisation des données**
- Possibilité de créer différents types de graphiques



VISUALISATION AVEC MATPLOTLIB

```
import matplotlib.pyplot as plt
```

```
%matplotlib inline
```

Important : ne pas oublier le **pyplot**



Nécessaire pour afficher les graphiques sur le notebook
plt.show() en mode « batch »

```
donnees = pd.DataFrame({'note': [2, 5, 12, 18, 10],  
                        'rendus':[1, 3, 5, 7, 9 ],  
                        'nom':['Titi','Toto','Tata','Tarbes','Titus']})
```

On va créer une **figure** qui contiendra le graphique

```
plt.figure(figsize=(10, 4))
```

```
plt.title("Rendus par notes")  
plt.xlabel("Nb de rendus")  
plt.ylabel("Notes")
```

Données : listes, array (Numpy), Series ou **DataFrames**

figsize= (largeur , hauteur)

On définit les **paramètres** titre, labels x et y

```
plt.plot(donnees['rendus'], donnees['note'])
```

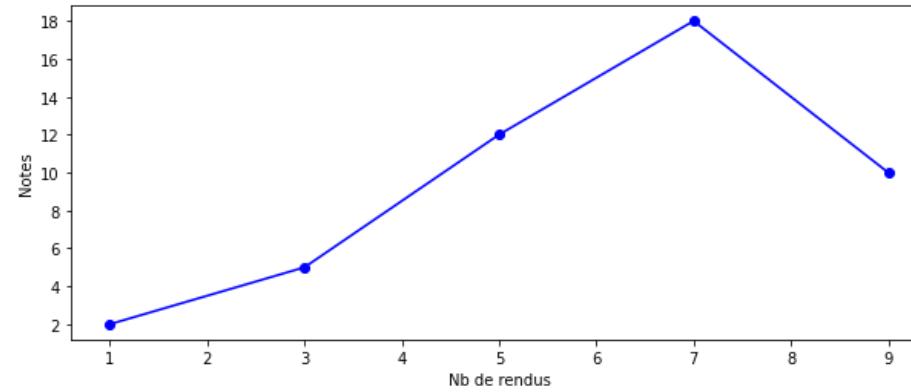
données axes x et y

On définit le **graphique** proprement dit

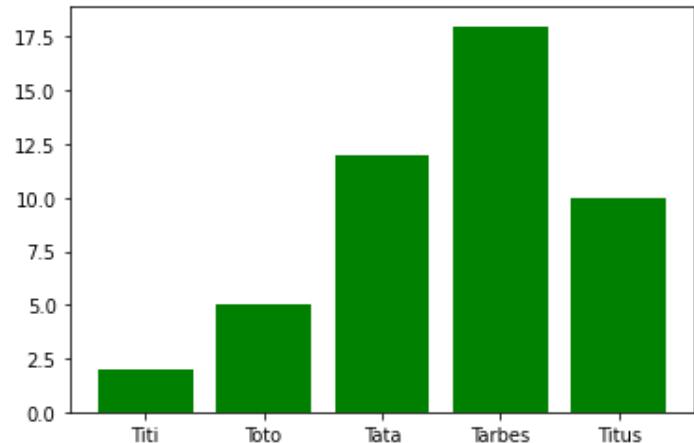
VISUALISATION AVEC MATPLOTLIB

```
plt.plot(donnees['rendus'],  
         donnees['note'],  
         color='blue', marker='o' )
```

Possibilité de choisir le marqueur

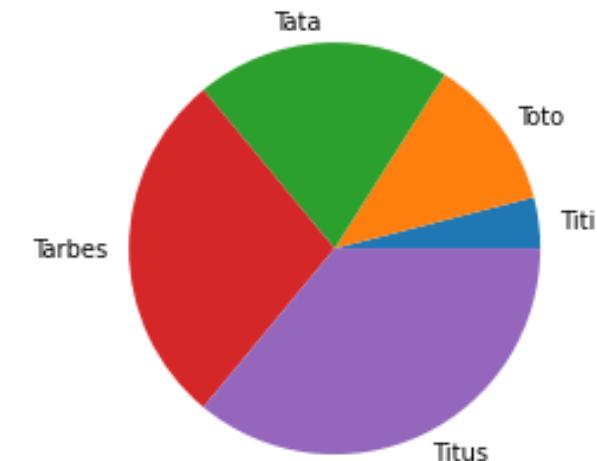


```
plt.bar(donnees['nom'], donnees['note'], color='green')
```



```
plt.pie(donnees['rendus'],labels=donnees['nom'])
```

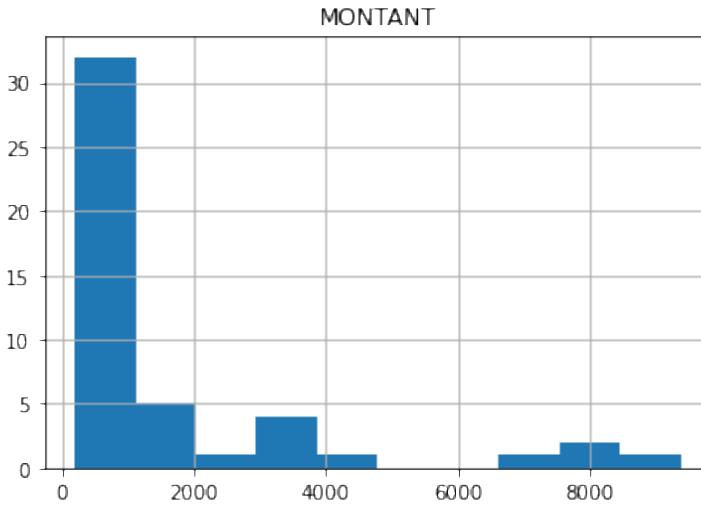
Possibilité de choisir la couleur



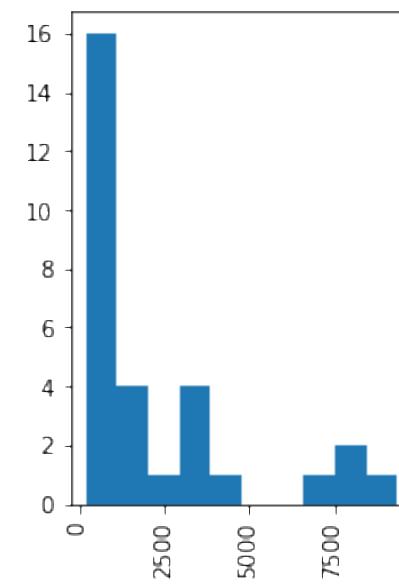
VISUALISATION AVEC MATPLOTLIB

- On peut utiliser Matplotlib directement à partir d'un DataFrame
 - Opérations par type de graphique : hist, boxplot et plot (cas général)

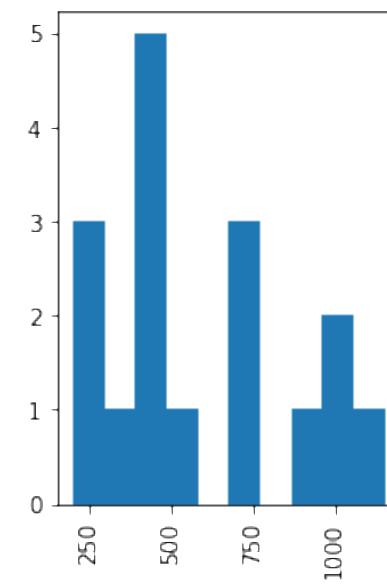
ventes.hist(column='MONTANT')



AUTOMATISME



ELECTRICITE

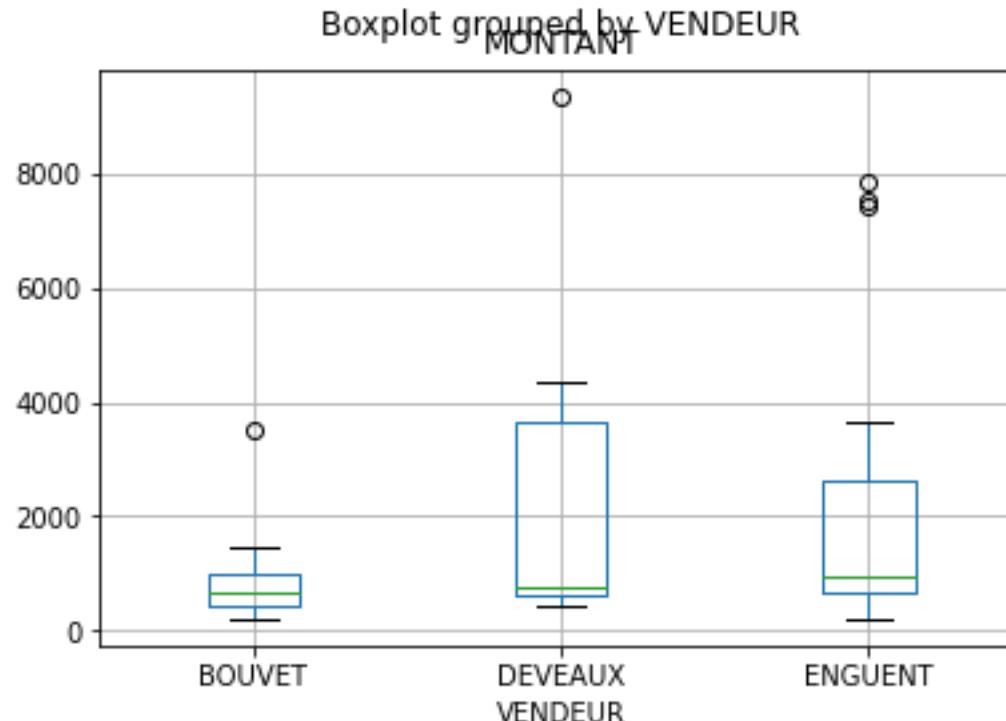


ventes.hist(column='MONTANT', by='SECTEUR')

VISUALISATION AVEC MATPLOTLIB

- On peut utiliser **Matplotlib** directement à partir d'un **DataFrame**
 - Opérations par type de graphique : **hist**, **boxplot** et **plot** (cas générale)

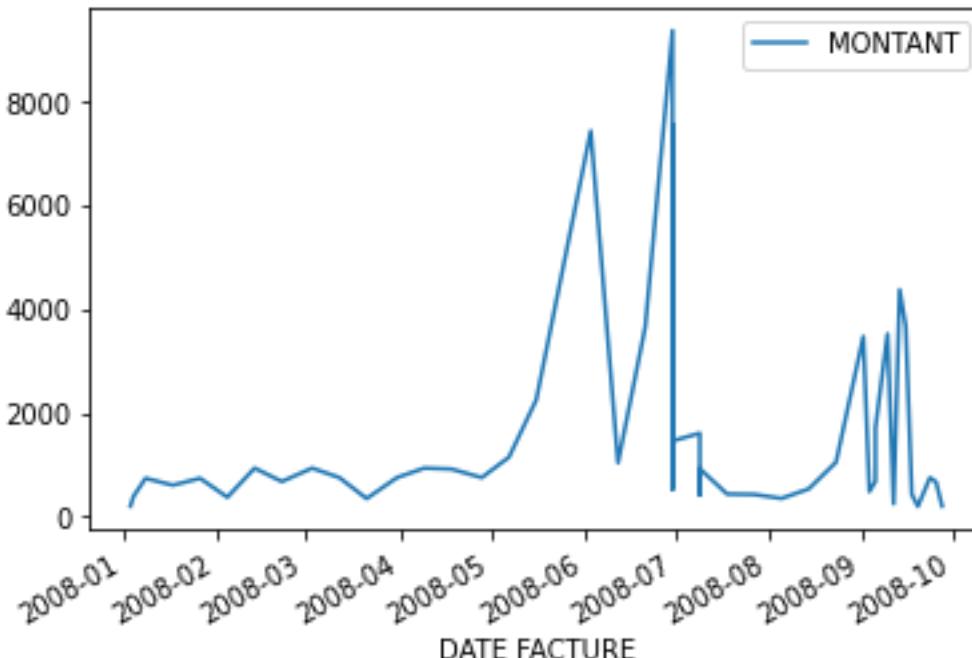
```
ventes.boxplot(column='MONTANT',by='VENDEUR')
```



VISUALISATION AVEC MATPLOTLIB

- On peut utiliser Matplotlib directement à partir d'un DataFrame
 - Opérations par type de graphique : `hist`, `boxplot` et `plot` (cas générale)

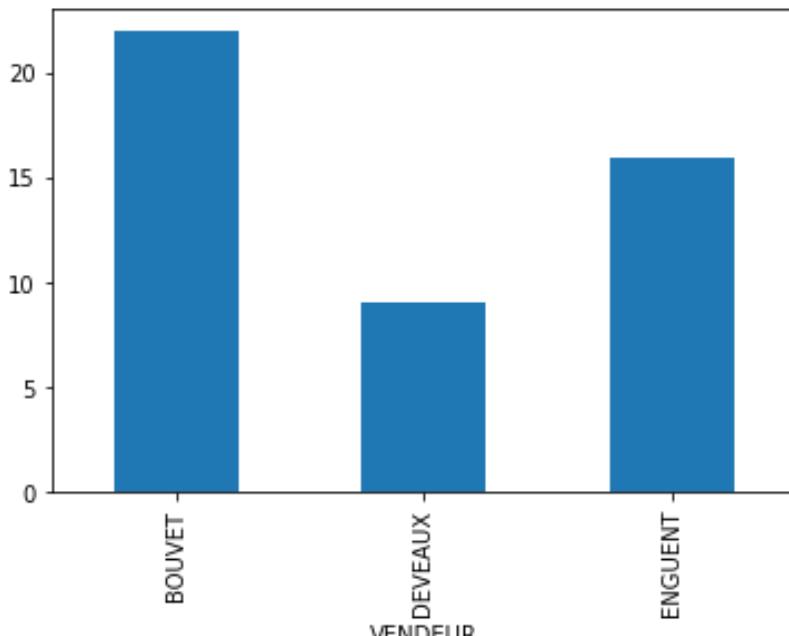
```
ventes.plot(x='DATE FACTURE', y='MONTANT')
```



VISUALISATION AVEC MATPLOTLIB

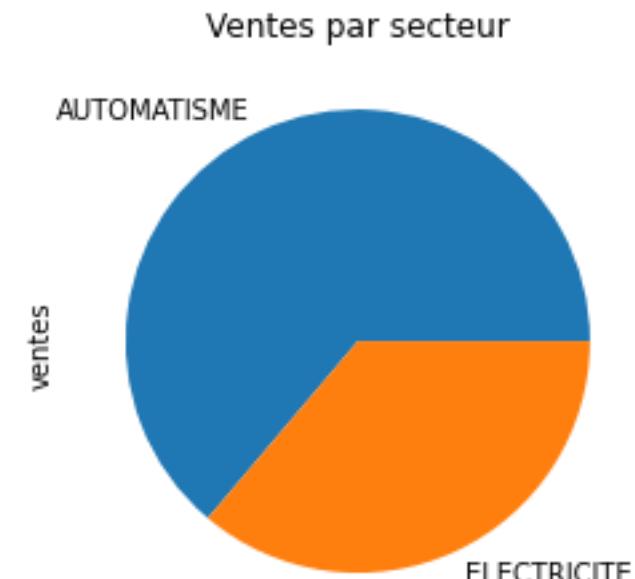
*Choix des données:
ventes par vendeur*

Nombre de ventes



```
ventes.groupby(by='VENDEUR').size().plot(kind='bar')
```

*On plot dans un graphique
type barres*



*Choix des paramètres :
titre, label axes x et y*

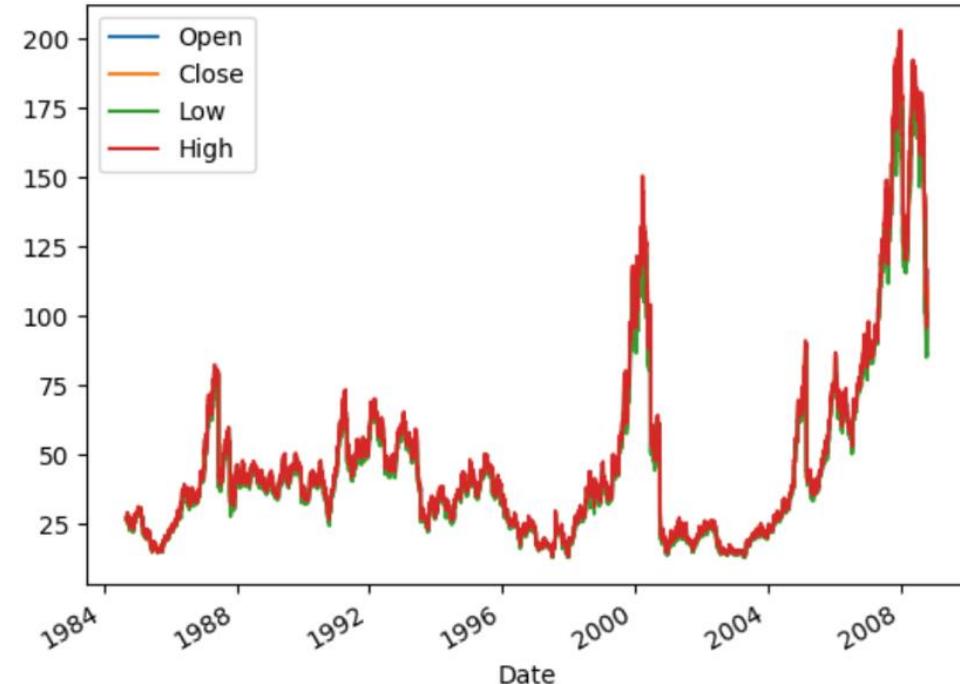
ENREGISTRER LES GRAPHIQUES

- La méthode **show()** fait l'affichage sur l'écran
- On peut enregistrer l'image dans un fichier
 - Plusieurs formats supportés (png, jpg, pdf)

```
detail = aapl.loc[:,['Open','Close','Low','High']]
```

```
detail.plot()
```

```
plt.savefig('plot.png')
```



QUELQUES EXERCICES

- Ouvrir lien suivant et suivre les liens des exercices

- <https://tinyurl.com/mr3ndc2w>

- Ou
 - <https://github.com/lsteffenel/CHPS0704>