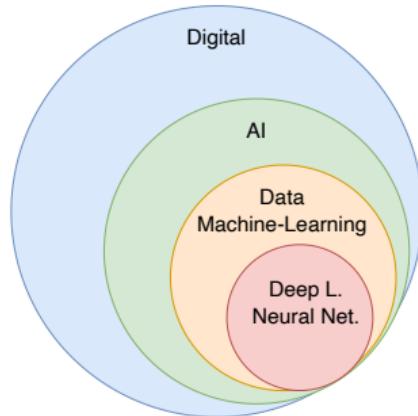


INTRODUCTION À NUMPY (ET MATPLOTLIB+PANDAS)

Vincent Guigue
vincent.guigue@agroparistech.fr



Artificial Intelligence & Machine Learning



| Input (\mathbf{X}) | Output (\mathbf{Y}) | Application |
|------------------------|-------------------------|---------------------|
| email | spam? (0/1) | spam filtering |
| audio | text transcript | speech recognition |
| English | Chinese | machine translation |
| ad, user info | click? (0/1) | online advertising |
| image, radar info | position of other cars | self-driving car |
| image of phone | defect? (0/1) | visual inspection |

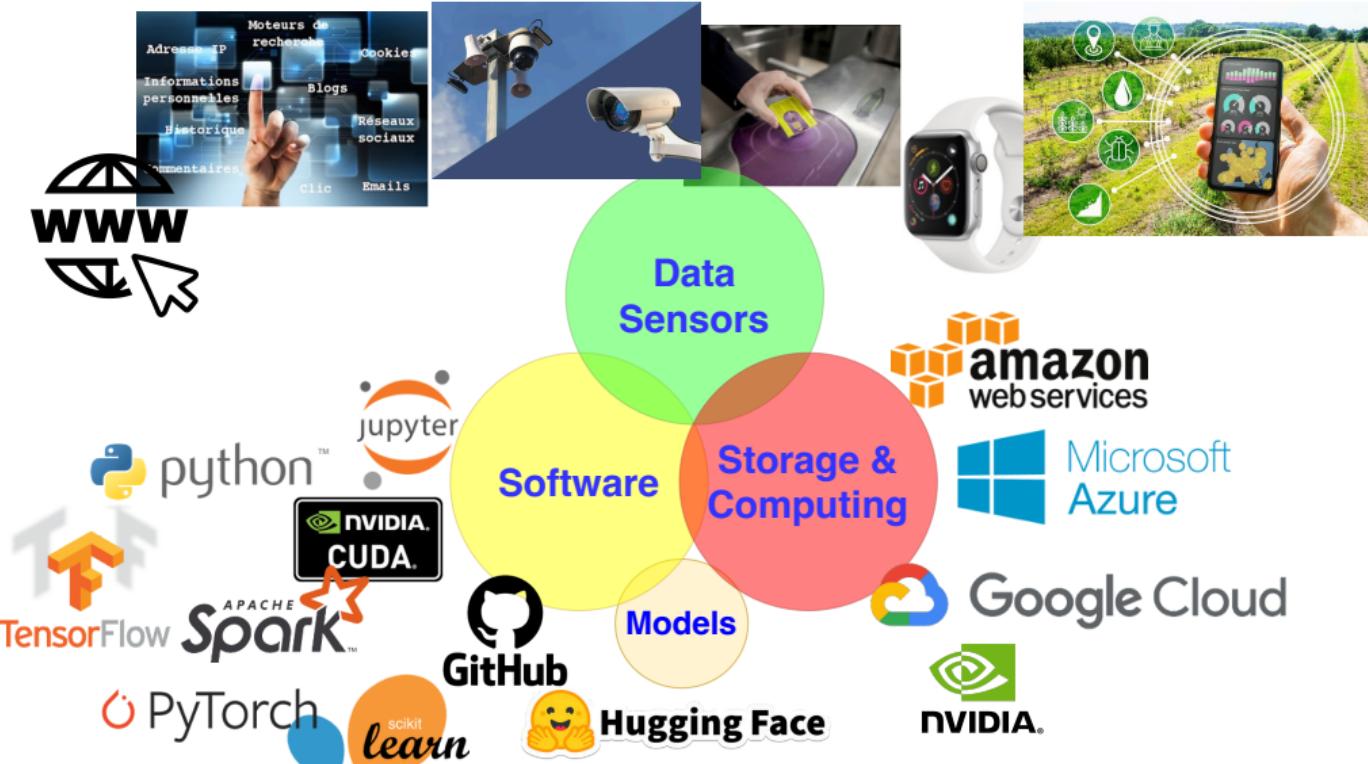
AI: computer programs that engage in tasks which are, for now, performed more satisfactorily by human beings because they require high-level mental processes.

Marvin Lee Minsky, 1956

N-AI (Narrow Artificial Intelligence), dedicated to a single task
≠ **G-AI (General AI)**, which replaces humans in complex systems.

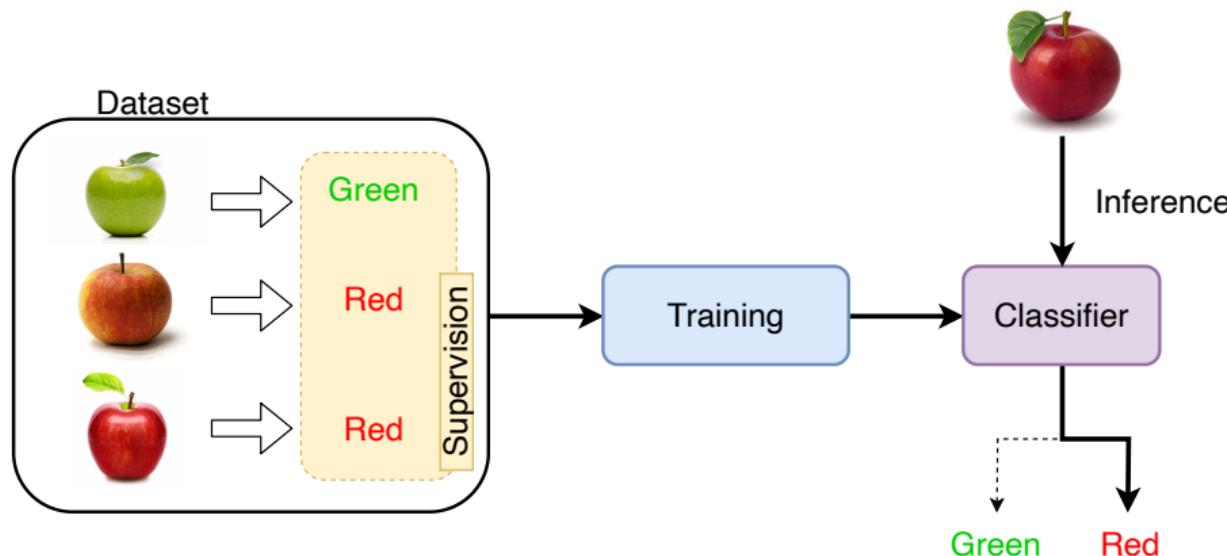
Andrew Ng, 2015

The Ingredients of Artificial Intelligence



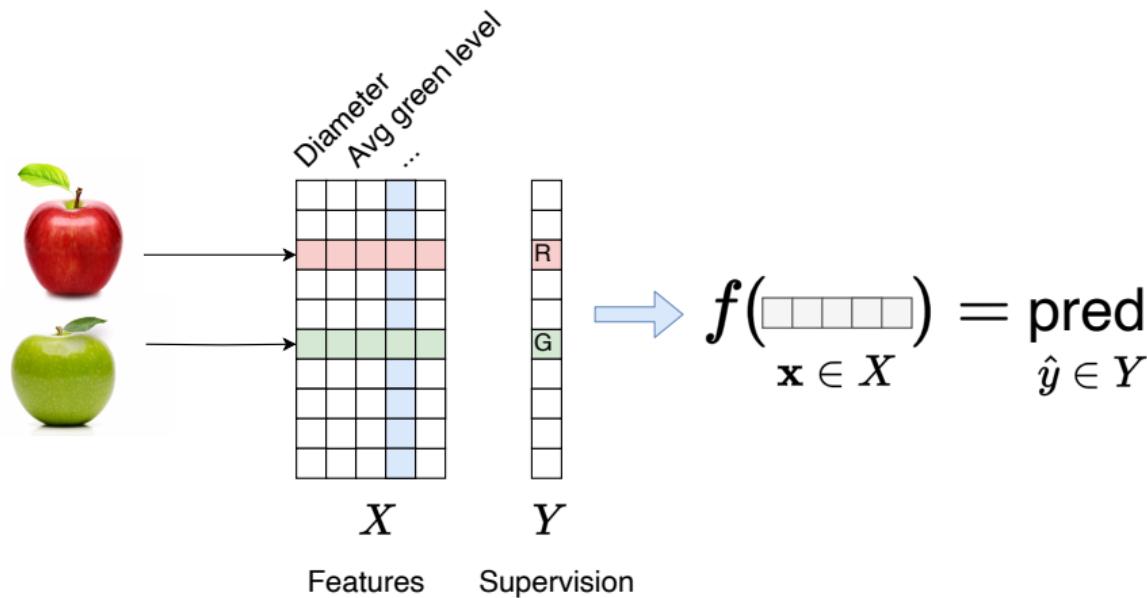
Machine Learning Definition

- 1 Collecting labeled **dataset**
- 2 Training **classifier**
- 3 Exploiting the model



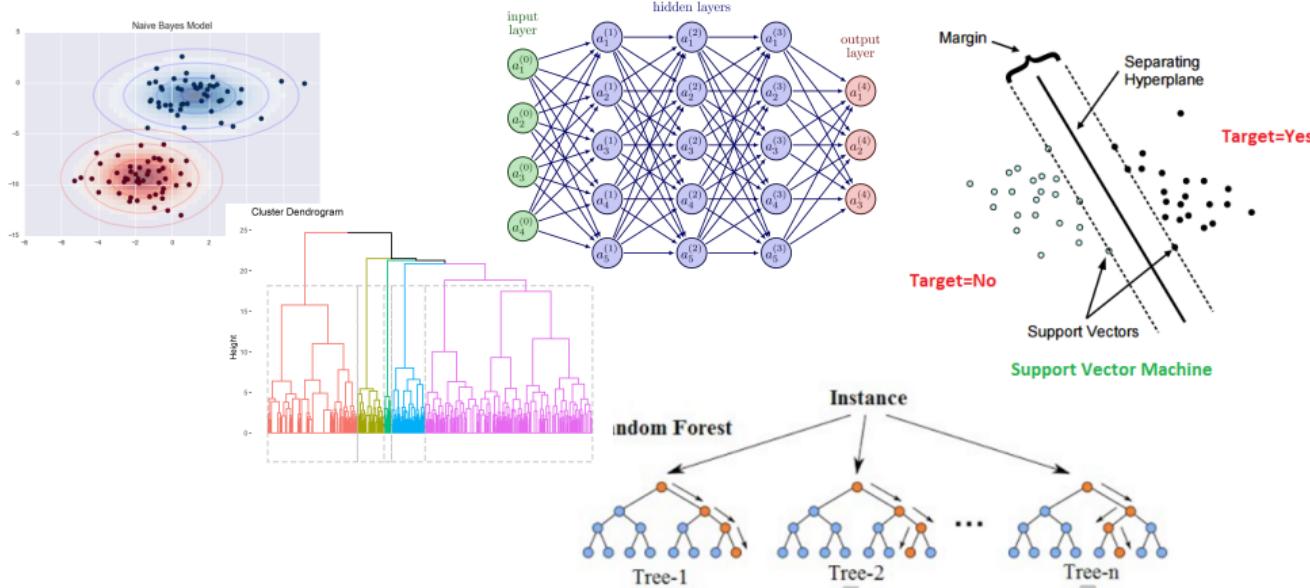
Machine Learning Definition

- 1 Collecting labeled **dataset**
- 2 Training **classifier**
- 3 Exploiting the model



Machine Learning Definition

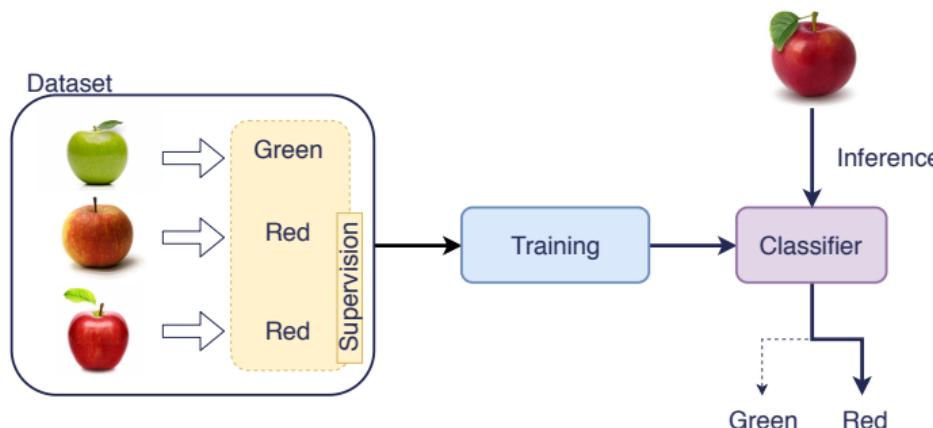
- 1 Collecting labeled **dataset**
- 2 Training **classifier**
- 3 Exploiting the model



Programmation orientée données

■ Python : langage unificateur (codage vs wrapper)

- *Calcul scientifique* : numpy
- *Machine-learning*: scikit-learn, pandas, matplotlib
- *Deep-learning*: pytorch
- Environnement de développement: Visual Studio Code / jupyter-notebook



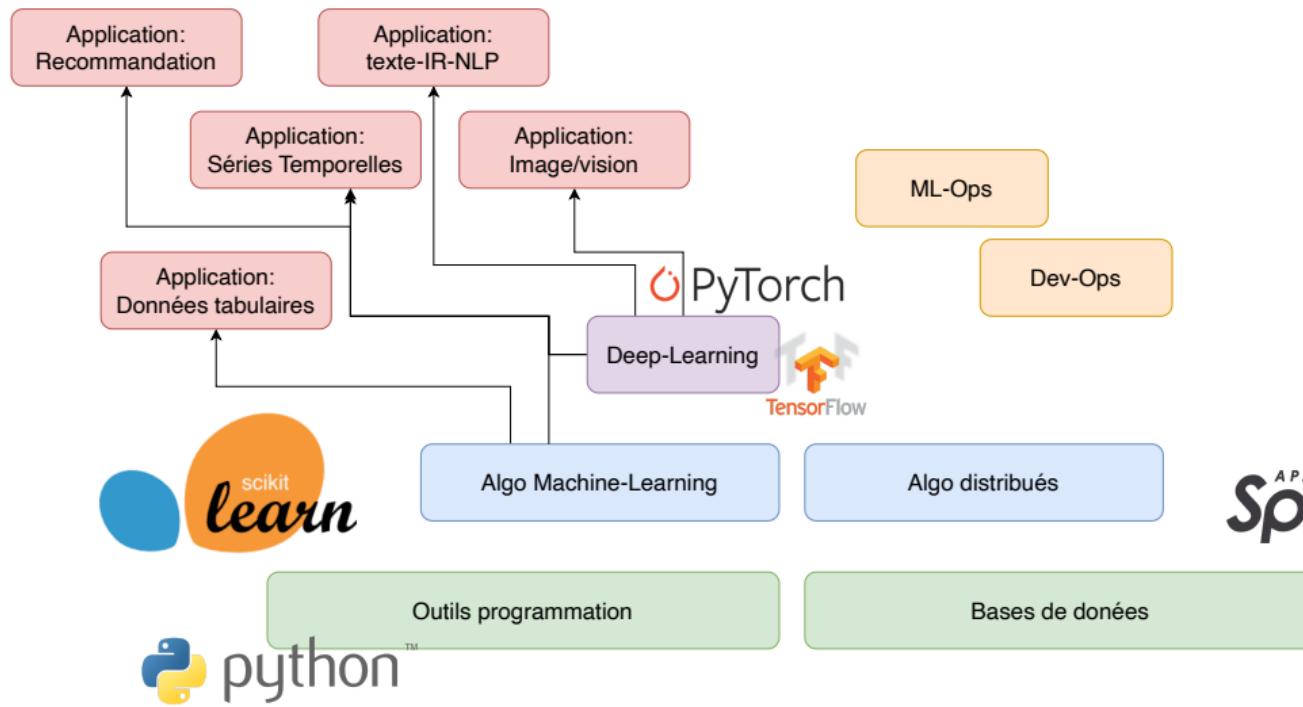
Où se trouve les leviers de performance?
Dans les modèles...
Mais surtout dans les chaînes de traitements !

ORGANISATION



Enseignement de l'IA

- Différents niveaux d'accès
- Différentes branches: types d'outils, application thématiques, ...





Organisation (optimiste/ale) du semestre

■ 5 séances - Machine Learning

- 1 séance - numpy = Mise à niveau en python, numpy, matplotlib
- 1 séance - scikit-learn = outils de base de pour la régression et la classification + évaluation robuste
- 1 séance - chaîne de traitement, sélection de variables et pre-processing
- 1 séance - visualisation des données et optimisation des hyperparamètres
- 1 séance - Ajustement / Support projet en machine learning

■ 8 séances - Deep Learning en pytorch

- 1 séance - Introduction à pytorch, structure de données et gradient
- 1 séance - Perceptron & réseau de neurones
- 1 séance - Convolutional Neural Network & application en image
- 1 séance - Apprentissage de représentation (Embedding) & systèmes de recommandation
- 1 séance - Réseaux de neurones récurrents (RNN)
- 1 séance - Calcul d'attention (pour les RNN)
- 1 séance - Architecture Transformer
- 1 séance - Projet

Numpy

3 séries de slides + notebooks



1 séance

- 1 Python & numpy
- 2 [OPT] Classification bayesienne
- 3 [OPT] Algorithme(s) de gradient

repo : `tuto_numpy`

vguigue 2025

1_python

2_numpy

3_classif_bayes

4_gradient

5_use_case

- ⇒ On ne va pas tout faire! Mais on peut expliquer les idées
- ⇒ Chacun doit en tirer un message personnel optimal :-)

Numpy

3 séries de slides + notebooks



1 séance

1 Python & numpy

- Clé d'accès aux outils en IA \Rightarrow besoin d'aisance
- Possibilité de demander de l'aide aux LLM...
Avec des questions techniques, pas des énoncés!

2 [OPT] Classification bayesienne

3 [OPT] Algorithme(s) de gradient

- \Rightarrow On ne va pas tout faire! Mais on peut expliquer les idées
 \Rightarrow Chacun doit en tirer un message personnel optimal :-)

repo : tuto_numpy

vguigue 2025

1_python

2_numpy

3_classif_bayes

4_gradient

5_use_case



Numpy

3 séries de slides + notebooks



1 séance

1 Python & numpy

2 [OPT] Classification bayesienne

- Construire un classifieur à la main...
 ⇒ comprendre les détails, astuces, divisions par 0...
- Mais dans la suite, on prendra des classifieurs tout fait.

3 [OPT] Algorithme(s) de gradient

- ⇒ On ne va pas tout faire! Mais on peut expliquer les idées
⇒ Chacun doit en tirer un message personnel optimal :-)

repo : tuto_numpy

vguigue 2025

1_python

2_numpy

3_classif_bayes

4_gradient

5_use_case



Numpy

3 séries de slides + notebooks



1 séance

1 Python & numpy

- 2 [OPT] Classification bayesienne
- 3 [OPT] Algorithme(s) de gradient
 - Optimisation = outils omniprésent

$$w^* = \arg \min_w E_{(x,y) \sim \mathcal{D}} [(xw - y)^2]$$

- Soit intégré dans les méthodes, soit délégué au moteur de calcul de pytorch.
- Intrégrer a minima les notions d'optimisation convexe ou pas

repo : tuto_numpy

| | | |
|--|-----------------|------|
| | vguigue | 2025 |
| | 1_python | |
| | 2_numpy | |
| | 3_classif_bayes | |
| | 4_gradient | |
| | 5_use_case | |

⇒ On ne va pas tout faire! Mais on peut expliquer les idées

Ordonnancement des savoirs

1 Python : la clé d'entrée de base

- Syntaxe de base: boucles, conditions, listes, ...
- Moins critique sur les dictionnaires, Counter, ...
⇒ Les outils de haut niveau seront dans numpy

2 Numpy : le point critique pour démarrer

- Un language à l'intérieur de python
- Nombreuses fonctions pour éviter les boucles, des réflexes spécifiques à prendre

3 Scikit-learn :

- Des outils clé en main... Plus de travail sur les protocoles de ML que sur la syntaxe

4 Pytorch :

- Des outils à construire: plus de code, de programmation objet...

repo : tuto_numpy

| |
|--|
|  vguigue 2025 |
| └── 1_python |
| └── 2_numpy |
| └── 3_classif_bayes |
| └── 4_gradient |
| └── 5_use_case |



A l'intérieur de numpy

■ Crédit (1) / manipulation (2) des matrices

- Zéros, uns, identité, aléatoire, ...
- Opérateurs : +-* , extraction de sous-matrices

■ Opérateurs de haut niveau (3)

- Nouveaux réflexes: where, ...

■ Matplotlib (4+5)

- Courbes, histogrammes, ...

■ Pandas (6)

tuto_numpy / 2_numpy /

vguigue maj

Name

..

ressources

1_Tutoriel_numpy_creation_matrice.ipynb

2_Numpy_Matrices.ipynb

3_Numpy_advanced.ipynb

4_Affichage_matplotlib.ipynb

5_Affichage_avance.ipynb

6_Tuto_pandas.ipynb

readme.md

Conclusion : passer à un nouveau langage...

■ Cout faible

- une fois que vous avez compris la logique générale

■ Cout non négligeable:

- Comprendre les forces et les faiblesses du langage
 - ... Et des environnements de développement
- Adapter sa manière de programmer (e.g. calculer un décile)
- Reprendre les bons reflexes (=aller vite)

⇒ https://github.com/vguigue/tuto_numpy