université PARIS-SACLAY

INTRODUCTION À NUMPY (ET MATPLOTLIB)

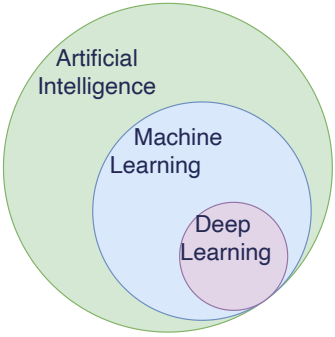
Vincent Guigue
vincent.guigue@agroparistech.fr

MIA PARIS-SACLAY
EKINOCS

AGROPARISTECH
Institut des Sciences et Industries du Vivant et de l'Environnement

Intelligence Artificielle, Machine Learning et Programmation

Input (**X**)	Output (**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



IA : programmes informatiques qui s'adonnent à des tâches qui sont, pour l'instant, accomplies de façon plus satisfaisante par des êtres humains car elles demandent des processus mentaux de haut niveau.

Marvin Lee Minsky, 1956

N-AI (Narrow Artificial Intelligence), dédiée à une tâche

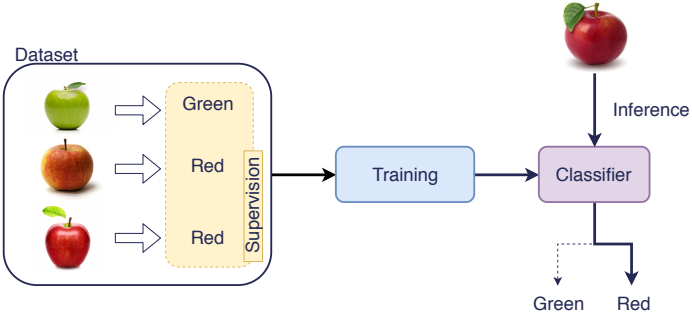
$\neq$ **G-AI (General AI)** qui remplace l'humain dans des systèmes complexes.

Andrew Ng, 2015

Ingrédients de l'Intelligence Artificielle

Programmation *orienté data*

- **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

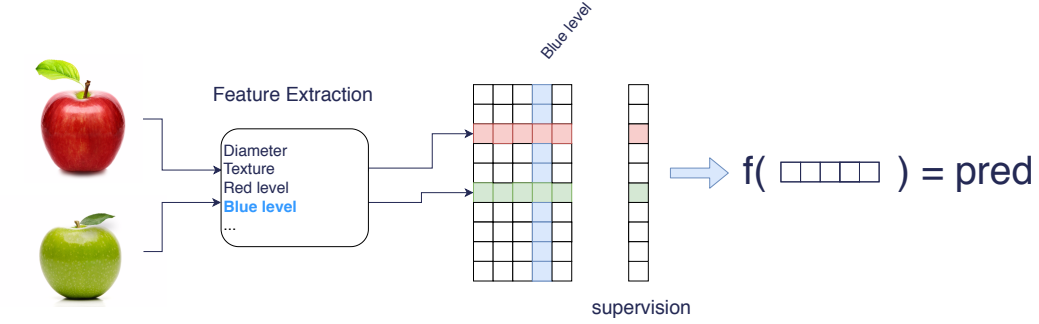


Où se trouve les leviers de performance ?

Dans les modèles...
Mais surtout dans les chaînes de traitements !

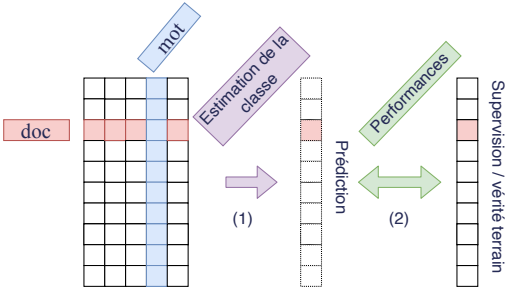
Programmation *orienté data*

- **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



Programmation *orienté data*

- **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



Organisation

Organisation (optim-iste/ale) du semestre

- **5 séances - Machine Learning**
  - 1.5 séances - 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 - chaine de traitement, sélection de variables et pre-processing
  - 1 séance - visualisation des données et optimisation des hyperparamètres
  - 0.5 séance - 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 $\Rightarrow$ 1 séance et demi

1. Python & numpy
2. Classification bayesienne
3. 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 :-)

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)