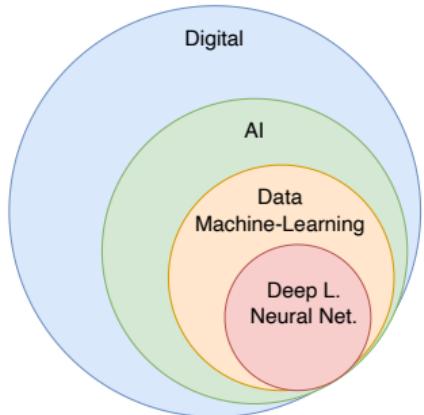


# INTRODUCTION À NUMPY (ET MATPLOTLIB+PANDAS)

Vincent Guigue

[vincent.guigue@agroparistech.fr](mailto:vincent.guigue@agroparistech.fr)



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

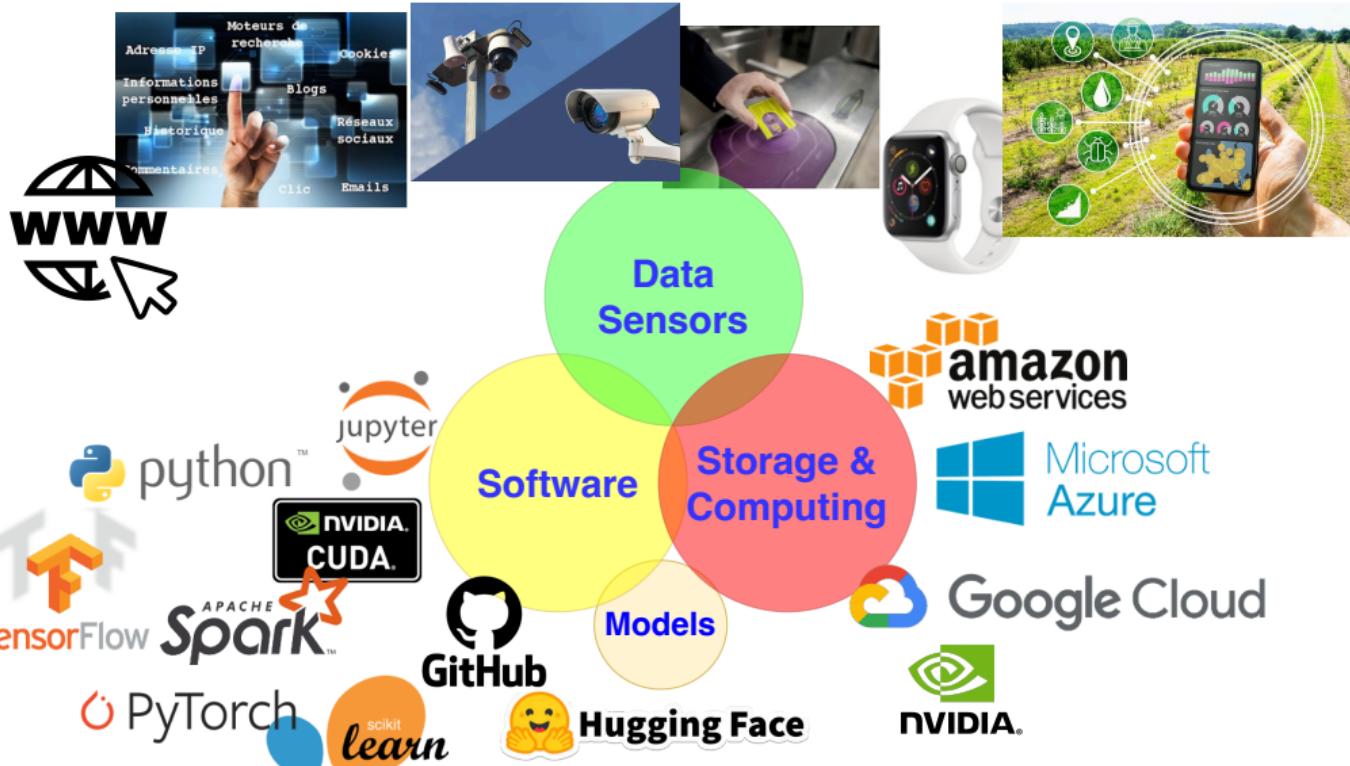
**AI:** computer programs that engage in tasks which are, for now, performed more satisfactorily by human beings because they require high-level mental processes.

Marvin Lee Minsky, 1956

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

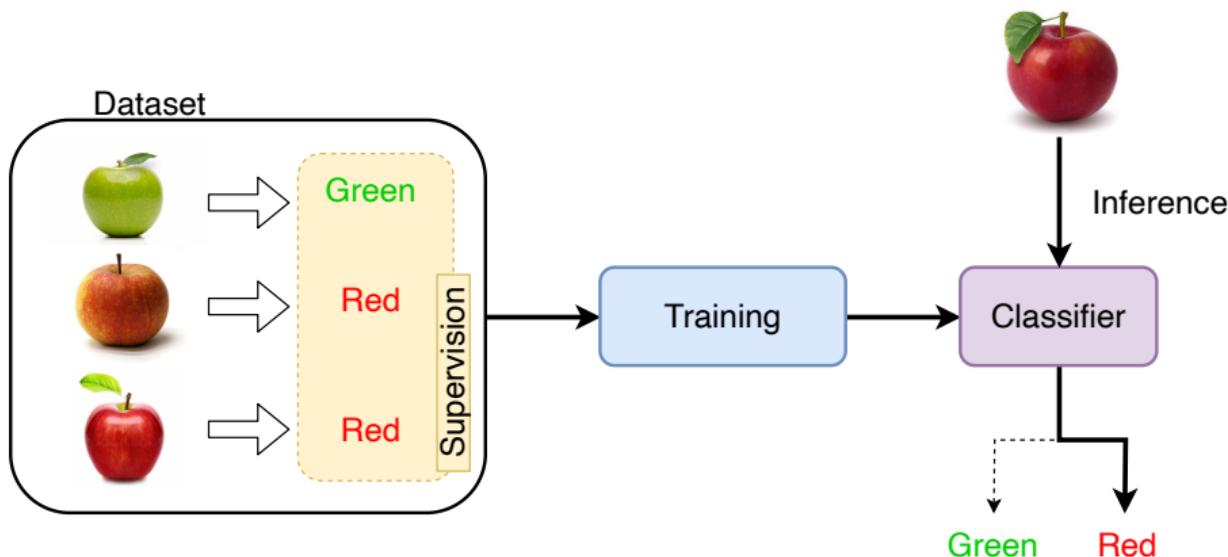
Andrew Ng, 2015

# The Ingredients of Artificial Intelligence

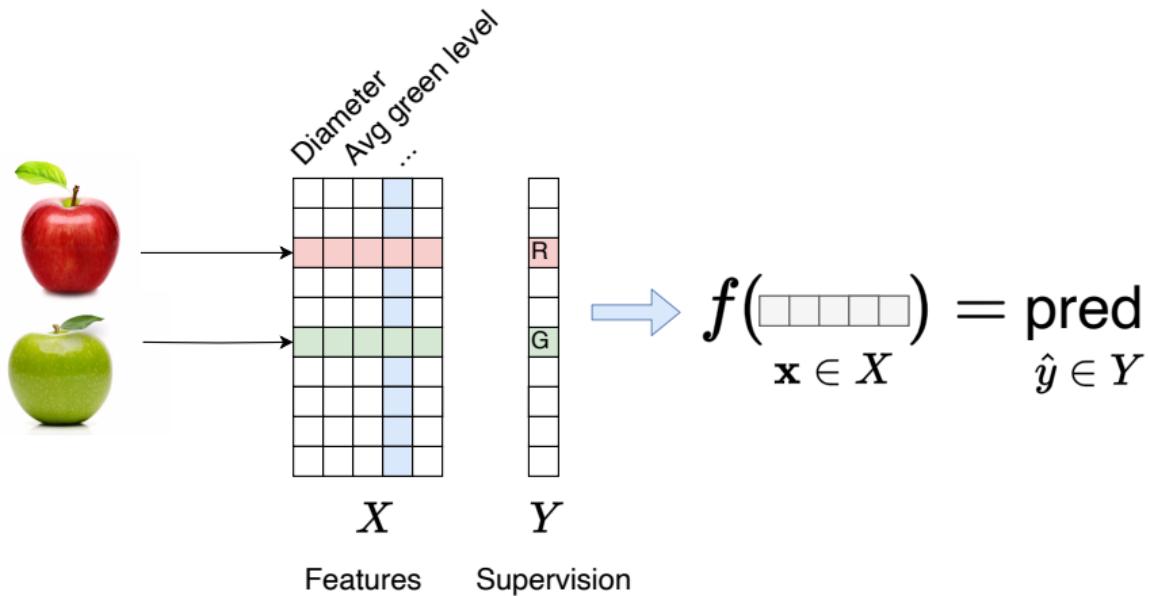


# Machine Learning Definition

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

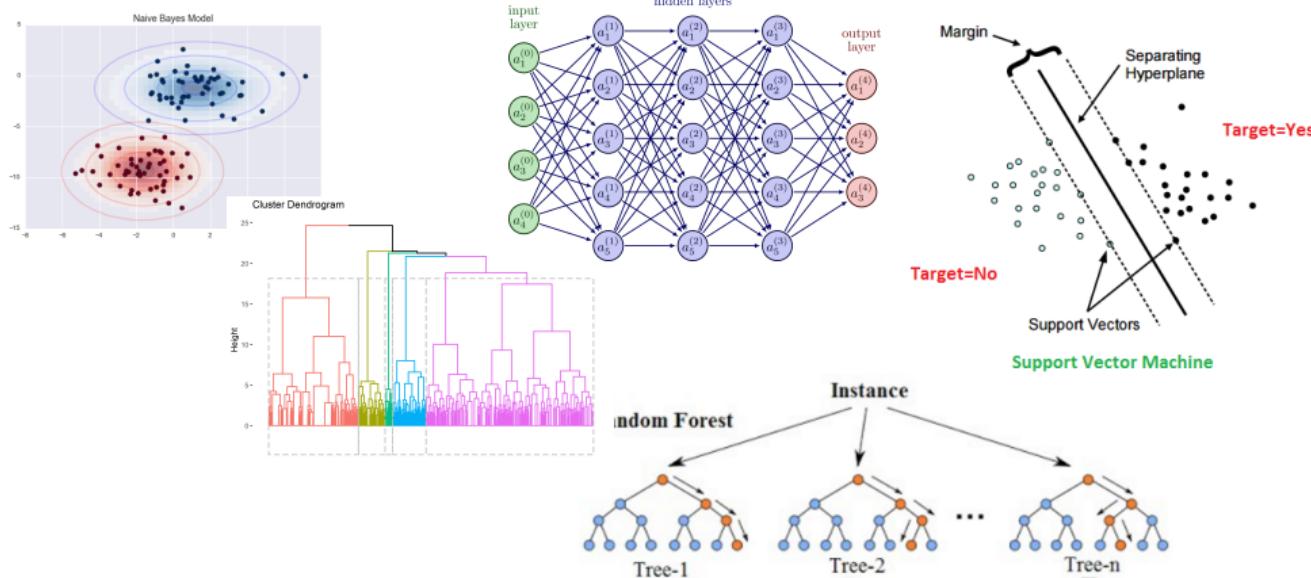


- 1 Collecting labeled **dataset**
- 2 Training **classifier**
- 3 Exploiting the model



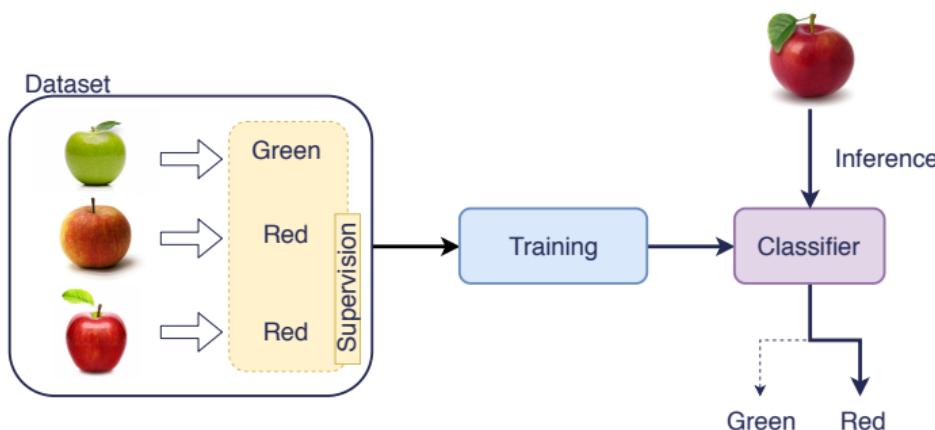
# Machine Learning Definition

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



## ■ 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 !

# Les outils nécessaires (et leurs évolutions)

- Fortran (70-90)
- C, C++ (80-...)
- Matlab
- Python (2010, ...)
  - Mais un python en évolution



Une **opportunité** incroyable pour les sciences des données !

- Outils matures & unifiés



Plusieurs séries de slides + notebooks



2 séances

Enchainement logique des briques pédagogiques et flexibilité:

- 1 Python
- 2 Numpy
- 3 (Classification bayesienn & Algorithme(s) de gradient)
- 4 Evaluation

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

⇒ Chacun doit en tirer un message personnel optimal :-)

## ■ 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](https://github.com/vguigue/tuto_numpy)

# L'outil notebook : un environnement convivial

- + Mêler du texte (énoncé) + blocs de code exécutable
- + Flexibilité: (1) navigateur web, (2) intégration dans Visual Studio Code, (3) Interface web (colab, github.dev, ...)
- + Tester du code, apprendre la syntaxe, développer un prototype
  - ⇒ Ajouter des boîtes, modifier le code
- ≈ Développer des modèles complexes
  - Risque sur l'organisation, la fiabilité

The screenshot shows a Jupyter Notebook window titled "jupyter 1\_Tuto\_python". The main content area displays a "Tutorial d'introduction à Python" with sections on variables, functions, and types. Below the tutorial, there are three code cells:

```
Entrée [1]: a = 1 # python sait qu'il s'agit d'un entier, la valeur est stockée dans a
print(a) # affichage
# a = 4 # la valeur précédente est perdue, a vaut maintenant 45
print(a)

Entrée [2]: b = 18.5 # création d'un réel
b = b + a # récupération, manipulation, etc...
print(b)

Entrée [3]: # chaîne de caractères: on peut utiliser les " ou les '
s = "Bonjour et bienvenue à la formation continue"
s2 = "une autre chaîne"
```

⇒ Petite démonstration des bons réflexes sur l'usage des notebooks

Le mot de la fin sur l'usage de chatGPT et consorts