Introduction au Machine Learning

Apprentissage statistique

Ludovic Darmet

18 Janvier 2023

Interface Cerveau Machine - ISAE-SUPAERO

Table des matières

Introduction

Exemples d'utilisation

Médical

- Prédire si un patient va avoir un second AVC
- Estimer l'état mental d'un sujet à partir de ses données physiologiques

Exemples d'utilisation

Médical

- Prédire si un patient va avoir un second AVC
- Estimer l'état mental d'un sujet à partir de ses données physiologiques

Industrie

- Identifier des spams dans une boite mail
- Lire le code postale sur une enveloppe
- Maintenance préventive en usine

Exemples d'utilisation

Médical

- Prédire si un patient va avoir un second AVC
- Estimer l'état mental d'un sujet à partir de ses données physiologiques

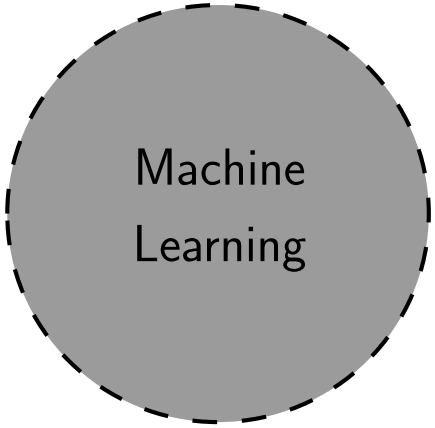
Industrie

- Identifier des spams dans une boite mail
- Lire le code postale sur une enveloppe
- Maintenance préventive en usine

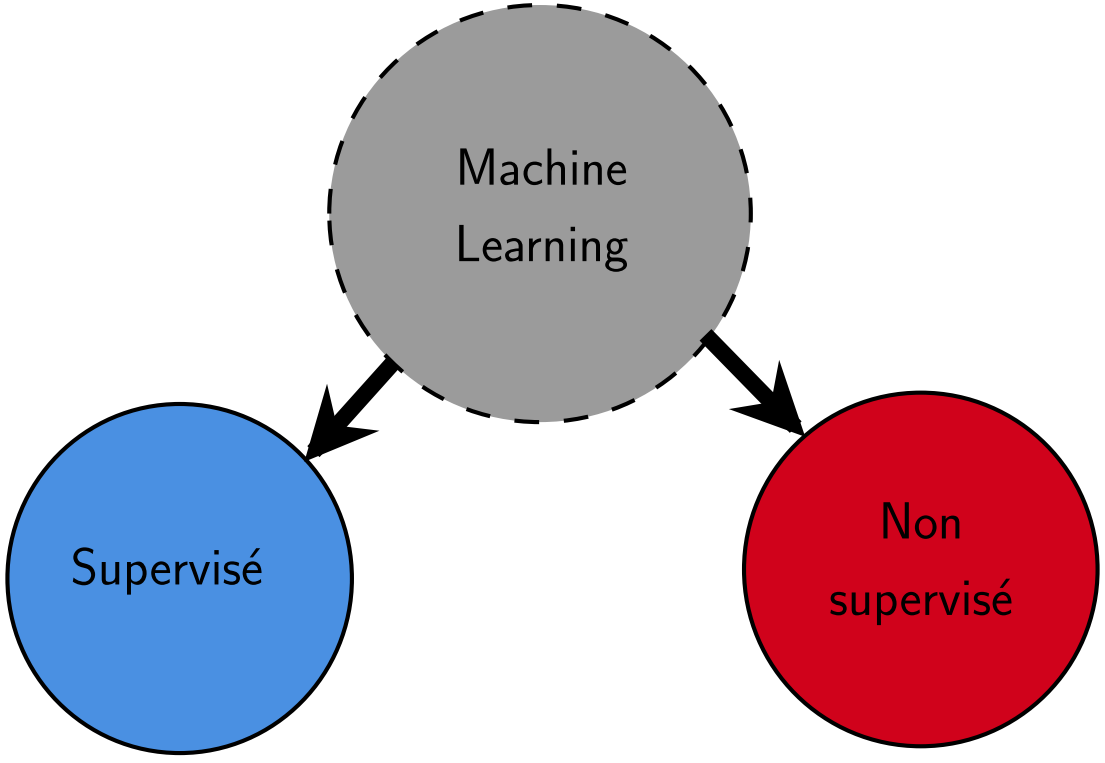
Finance

- Prédire la faillite d'une entreprise
- Estimer les variations futures du CAC-40

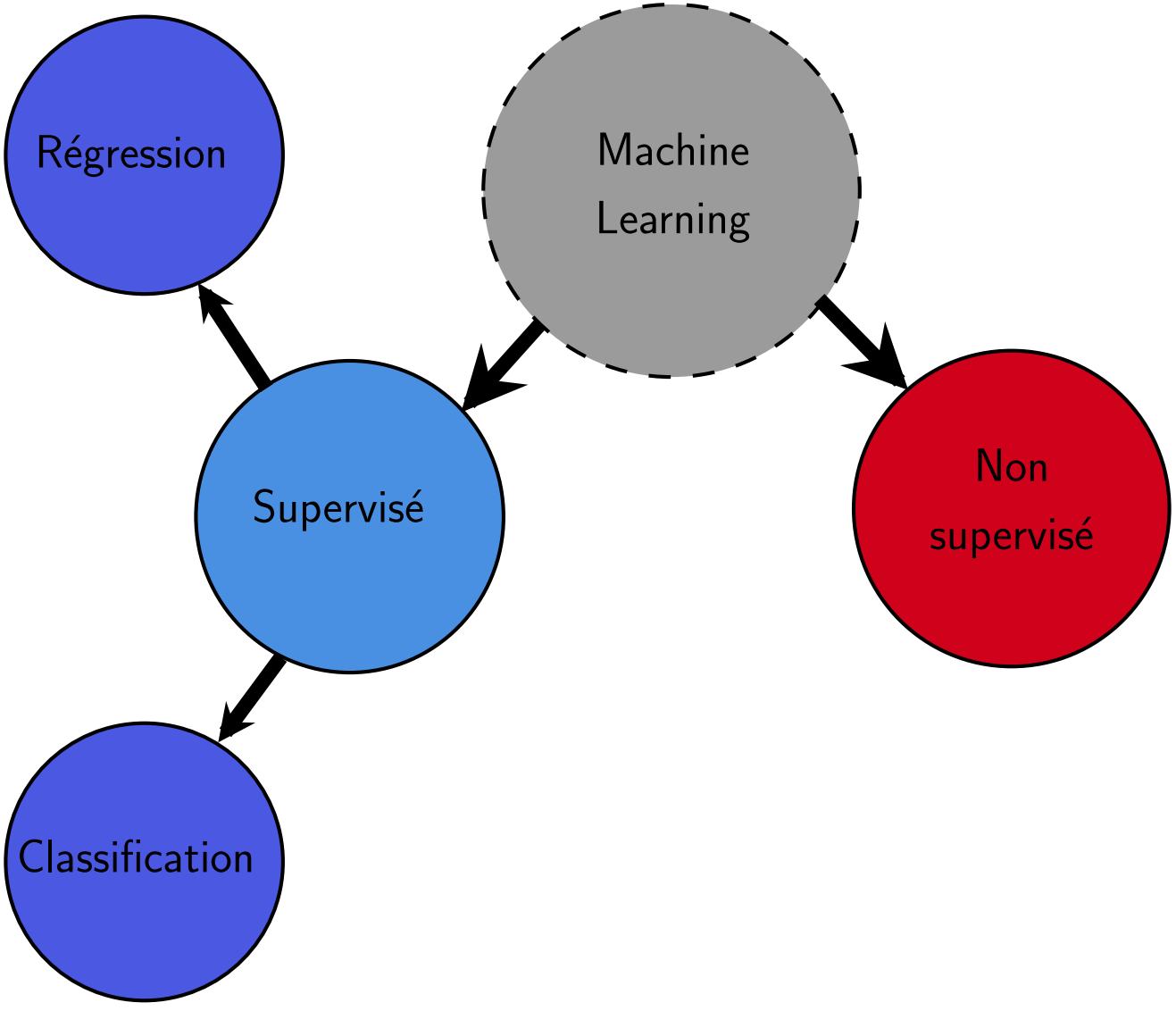
Les différents types de problèmes



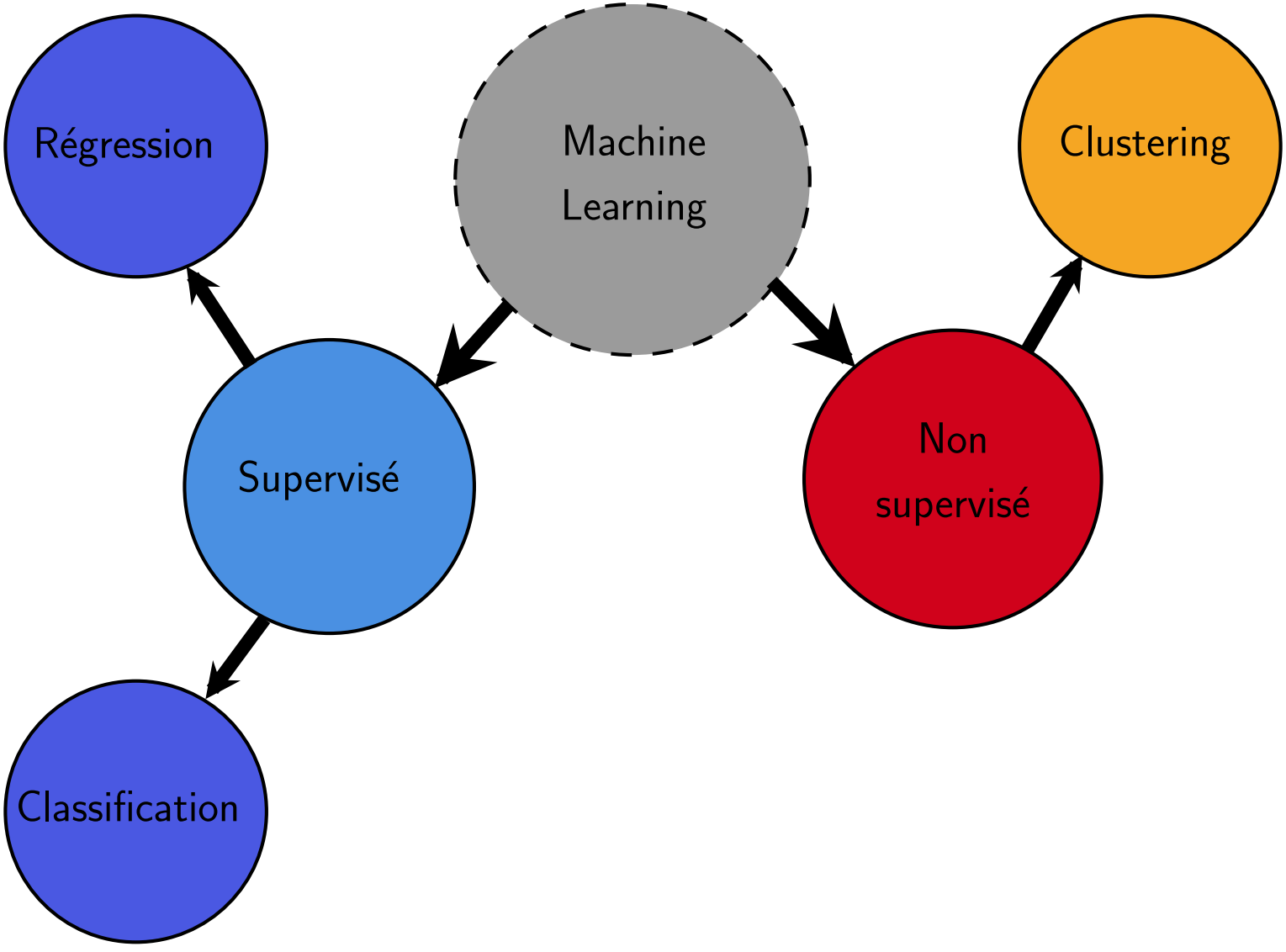
Les différents types de problèmes



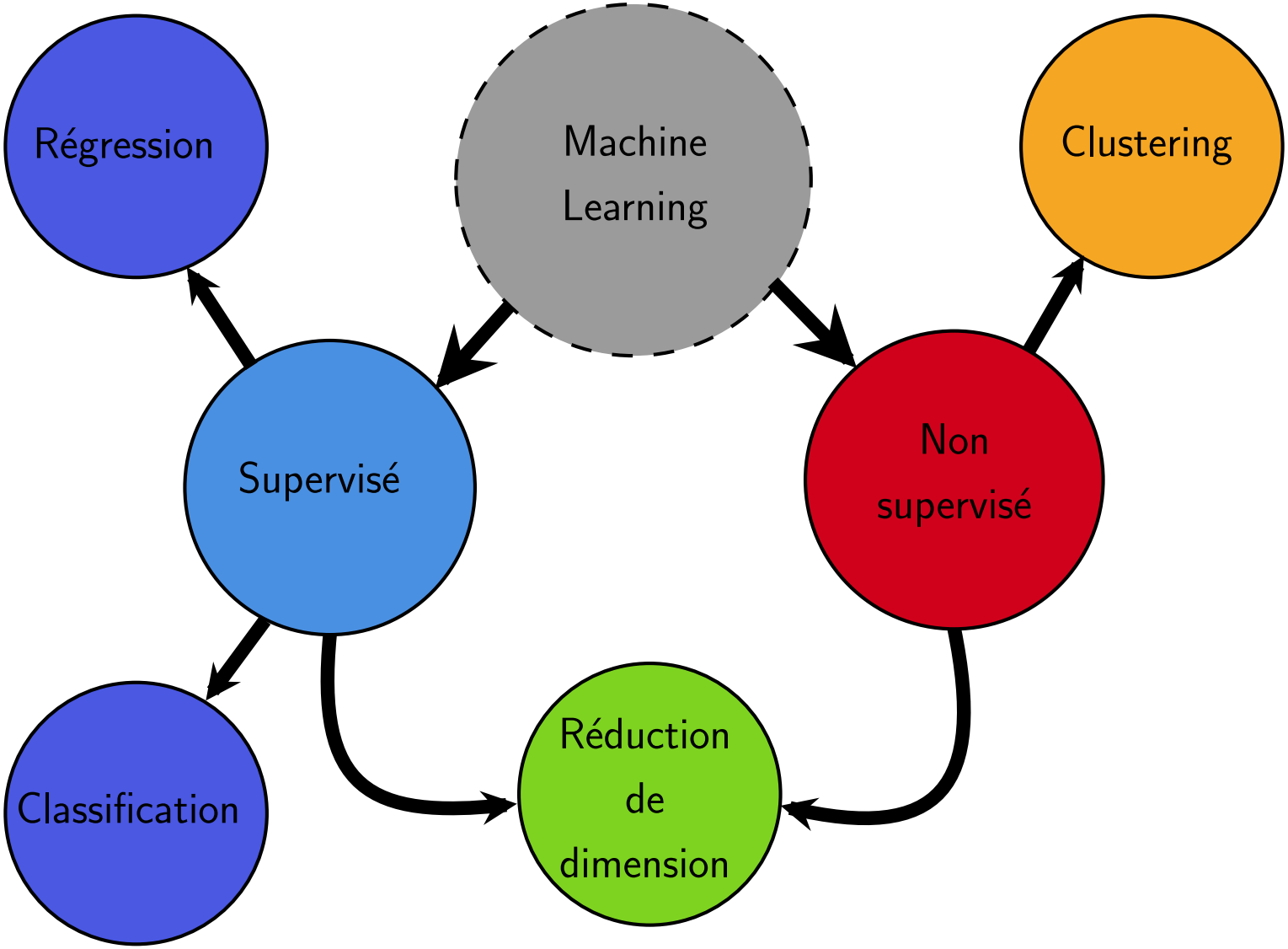
Les différents types de problèmes



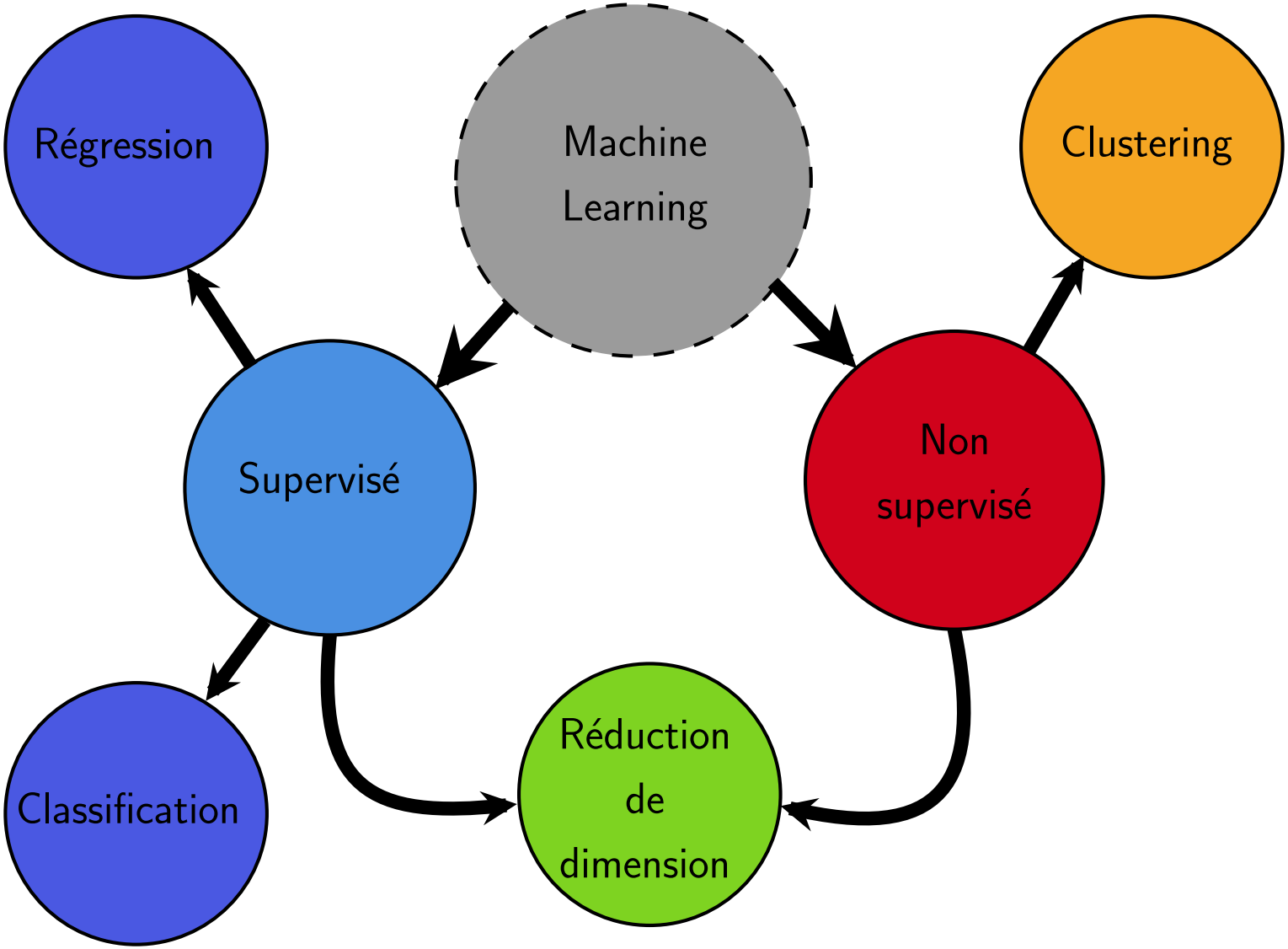
Les différents types de problèmes



Les différents types de problèmes

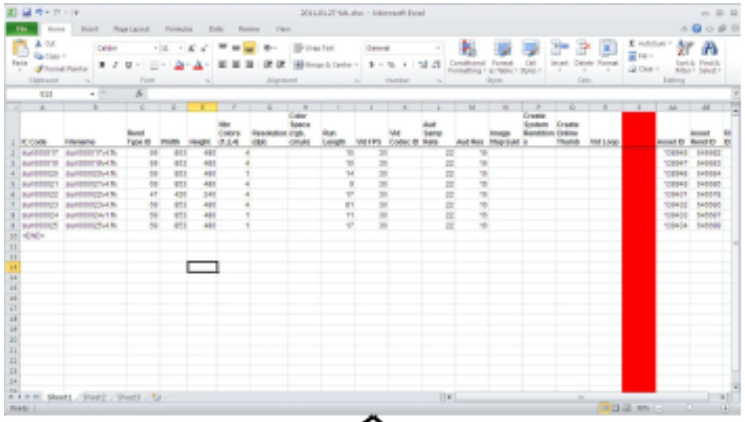


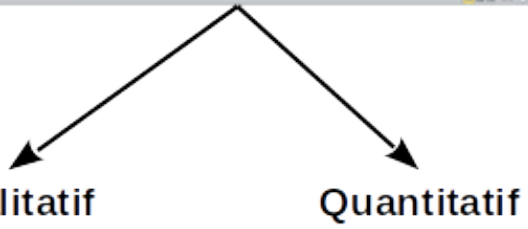
Les différents types de problèmes

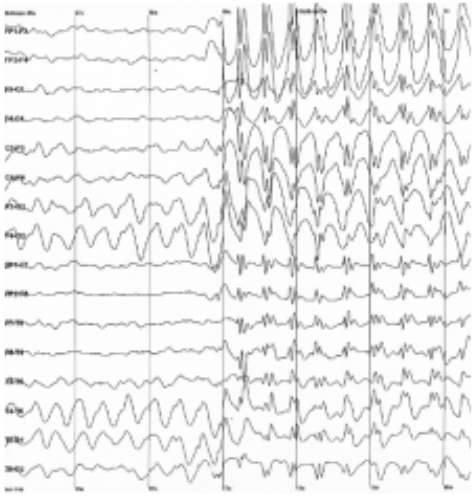


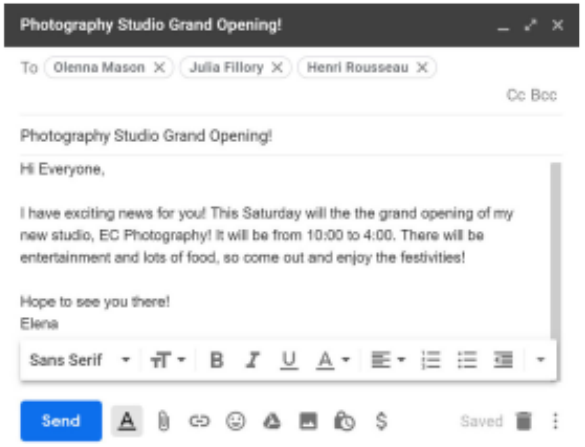
↪ Semi-supervisé, auto-supervisé, apprentissage par renforcement

Les différents types de données









1. Exploration et visualisation

- Regarder et **s'approprier** les données
- Quels sont les **biais** ? **Intuition** sur des mécanismes ?

Les étapes classiques

1. Exploration et visualisation

- Regarder et **s'approprier** les données
- Quels sont les **biais** ? **Intuition** sur des mécanismes ?

2. Data cleaning and pre-processing

- Nettoyage, **imputation** de données manquantes
- **Transformation de variables** : catégorielles en numériques, ou rendre plus descriptives
- **Sélection de variables**

Les étapes classiques

1. Exploration et visualisation

- Regarder et **s'approprier** les données
- Quels sont les **biais** ? **Intuition** sur des mécanismes ?

2. Data cleaning and pre-processing

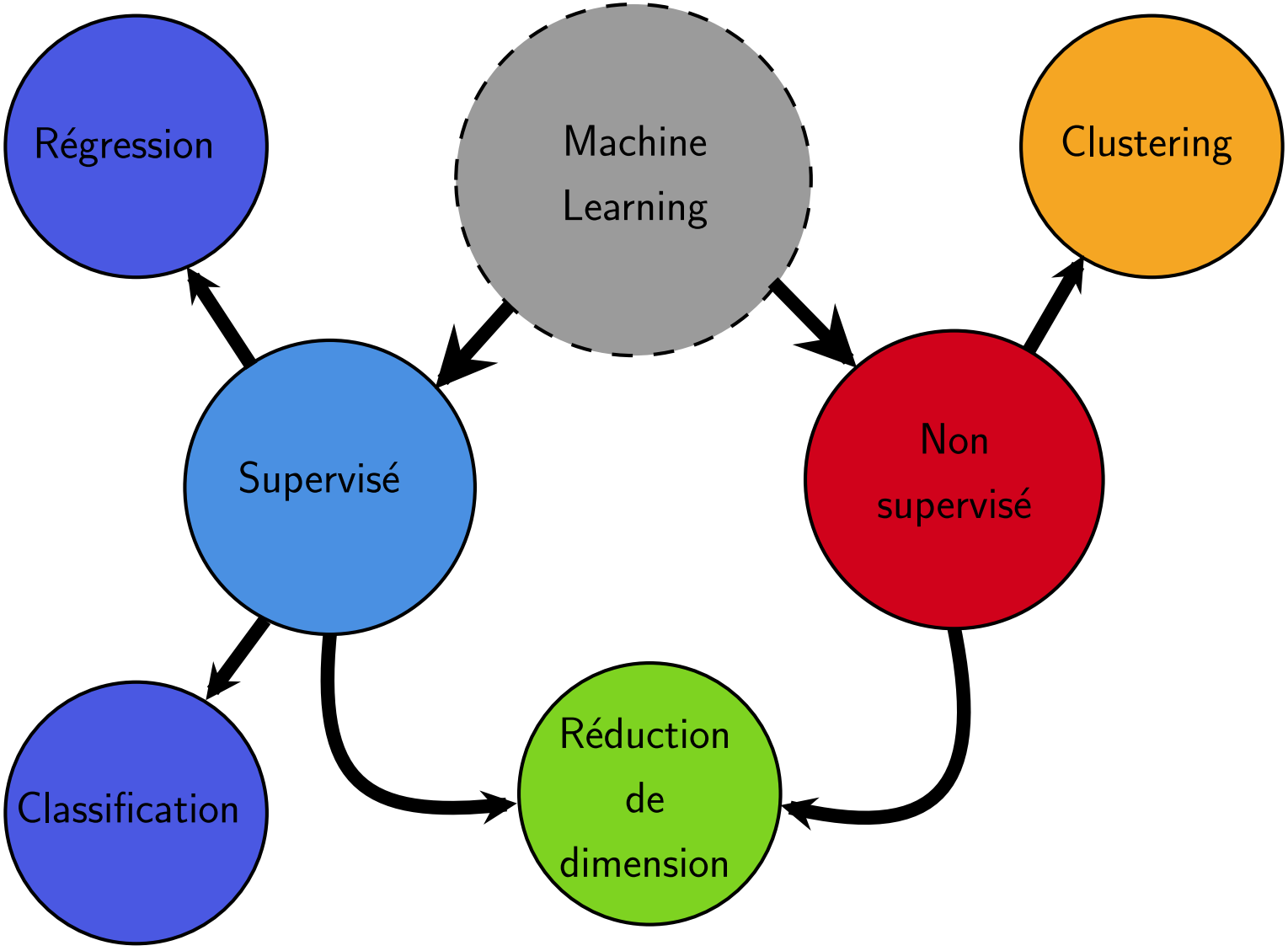
- Nettoyage, **imputation** de données manquantes
- **Transformation de variables** : catégorielles en numériques, ou rendre plus descriptives
- **Sélection de variables**

3. Apprentissage du modèle

- **Choix du modèle** et de ses hyper-paramètres
- Estimation de la performance, **l'erreur empirique**

L'apprentissage supervisé

Les différents types de problèmes



Apprentissage

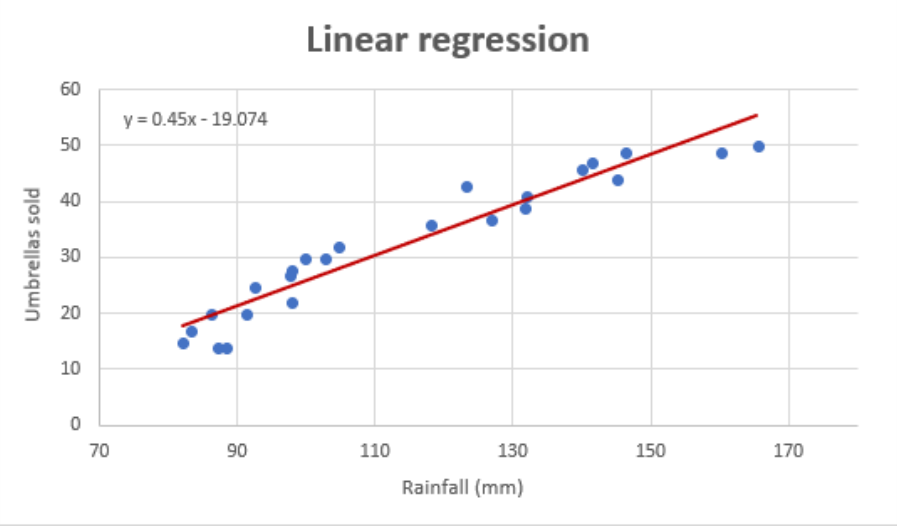
- On a X les **données** et y la **cible** que l'on veut prédire
- On cherche une fonction f telle que $y = f(X)$

Apprentissage

- On a X les **données** et y la **cible** que l'on veut prédire
- On cherche une fonction f telle que $y = f(X)$
- X et y sont des variables aléatoires dont on possède des échantillons
- Il existe une loi de probabilité jointe : $\mathbb{P}(y|X) \neq \mathbb{P}(y)$

Apprentissage

- On a X les **données** et y la **cible** que l'on veut prédire
- On cherche une fonction f telle que $y = f(X)$
- X et y sont des variables aléatoires dont on possède des échantillons
- Il existe une loi de probabilité jointe : $\mathbb{P}(y|X) \neq \mathbb{P}(y)$



Descripteurs/features

- Les lignes de X sont les différents **échantillons**
- Les colonnes sont les différents descripteurs/features

Descripteurs/features

- Les lignes de X sont les différents **échantillons**
- Les colonnes sont les différents descripteurs/features

La fonction d'erreur

- Pour entraîner le modèle : **optimisation d'une fonction d'erreur**
- Erreur quadratique par exemple : $L\left(y, f(X)\right) = \left(y - f(X)\right)^2$

Descripteurs/features

- Les lignes de X sont les différents **échantillons**
- Les colonnes sont les différents descripteurs/features

La fonction d'erreur

- Pour entraîner le modèle : **optimisation d'une fonction d'erreur**
- Erreur quadratique par exemple : $L\left(y, f(X)\right) = \left(y - f(X)\right)^2$

Erreur empirique de généralisation

- On a seulement des échantillons donc on va calculer une **espérance empirique** de l'erreur
- $f^* = \underset{f}{argmin}\ \mathbb{E}\left[L\left(y, f(X)\right)\right]$

L'apprentissage supervisé

Explorer les données

De nombreux biais dans les données !

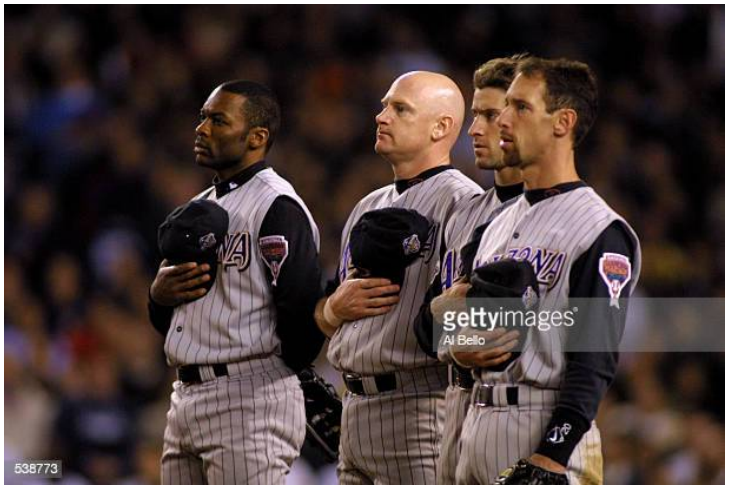
- **Corrélation ne veut pas dire causalité** : dans le Maine $R = 0.99$ entre le taux de divorce et la consommation de margarine (Spurious correlations)
- **Biais d'échantillonage** :
  * Analyse automatique de CVs chez Amazon
  * *Near-duplicates* dans la base CIFAR (jusqu'à -14% d'accuracy)
  * Image de *"mug"* sur Google Image majoritairement avec l'anse à droite
  * Base de données CelebA : toutes les femmes brunes sont souriantes $\hookrightarrow$ biais pour un détecteur de sourire

De nombreux biais dans les données !

- **Biais de sélection/du survivant** :

* Plus d'exemples de trajets aériens sans problème que d'accidents

* Identification d'un chanteur dans un stade de baseball (confusion avec *playing baseball*)



Évaluer la qualité des données

Exploration Data Analysis (EDA)

- Ordre de grandeur et variance de chaque feature
- Données **manquantes**, dupliquées : la quantité, est-ce qu'il y a un pattern ?
- Données **erronées** : en particulier si saisie de données manuelles
- Données **bruitées** et **outliers** : regarder les distributions des données
- Interaction et corrélation entre les variables
- **Erreurs de labélisation**

Évaluer la qualité des données

Exploration Data Analysis (EDA)

- Ordre de grandeur et variance de chaque feature
- Données **manquantes**, dupliquées : la quantité, est-ce qu'il y a un pattern ?
- Données **erronées** : en particulier si saisie de données manuelles
- Données **bruitées** et **outliers** : regarder les distributions des données
- Interaction et corrélation entre les variables
- **Erreurs de labélisation**

$\rightarrow$ **"No free lunch"**

L'apprentissage supervisé

Préparation des données

Transformation de features

- Transformer des **variables** **catégorielles** en variables numériques : *binarization*, *one hot encoding*
- Encoder des variables de texte : *Bag of Word*, *TF-IDF*, *tokenisation*
- **Normalisation** et **scaling** des données

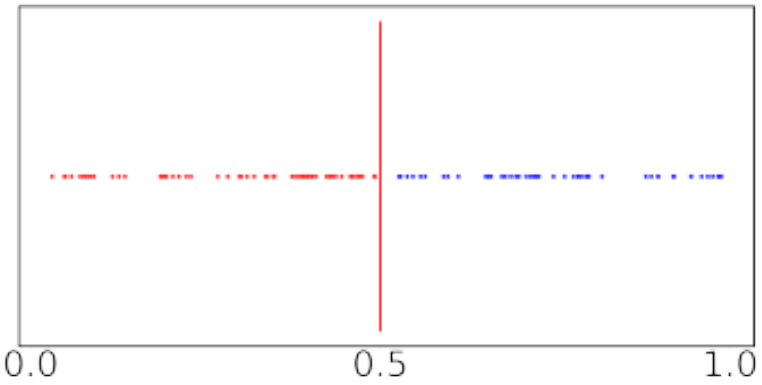
Transformation de features

- Transformer des **variables catégorielles** en variables numériques : *binarization*, *one hot encoding*
- Encoder des variables de texte : *Bag of Word*, *TF-IDF*, *tokenisation*
- **Normalisation** et **scaling** des données

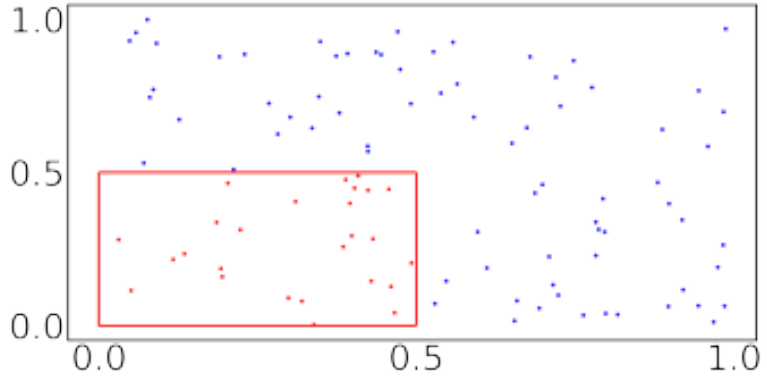
Sélection de features

- Rejeter les features inutiles ou trop **bruitées**
- Sélection **séquentielle**, *i.e.* par ajout (forward) ou suppression (backward)
- Sélection **aléatoire** de combinaisons de features

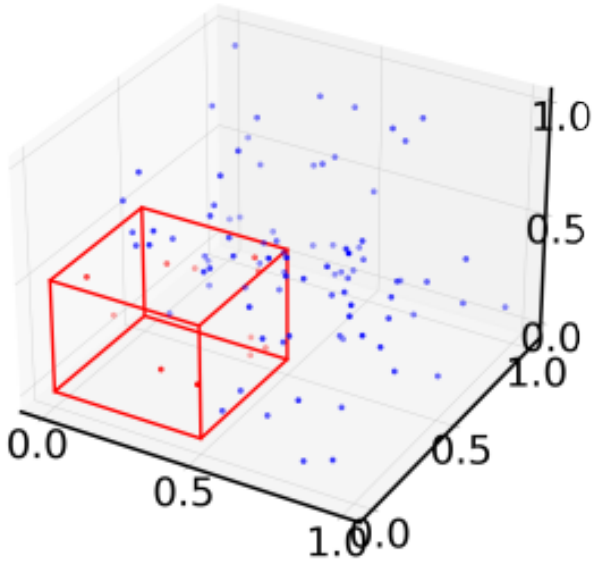
Malédiction de la dimension



1D : 53 points



2D : 26 points



3D : 11 points

Malédiction de la dimension

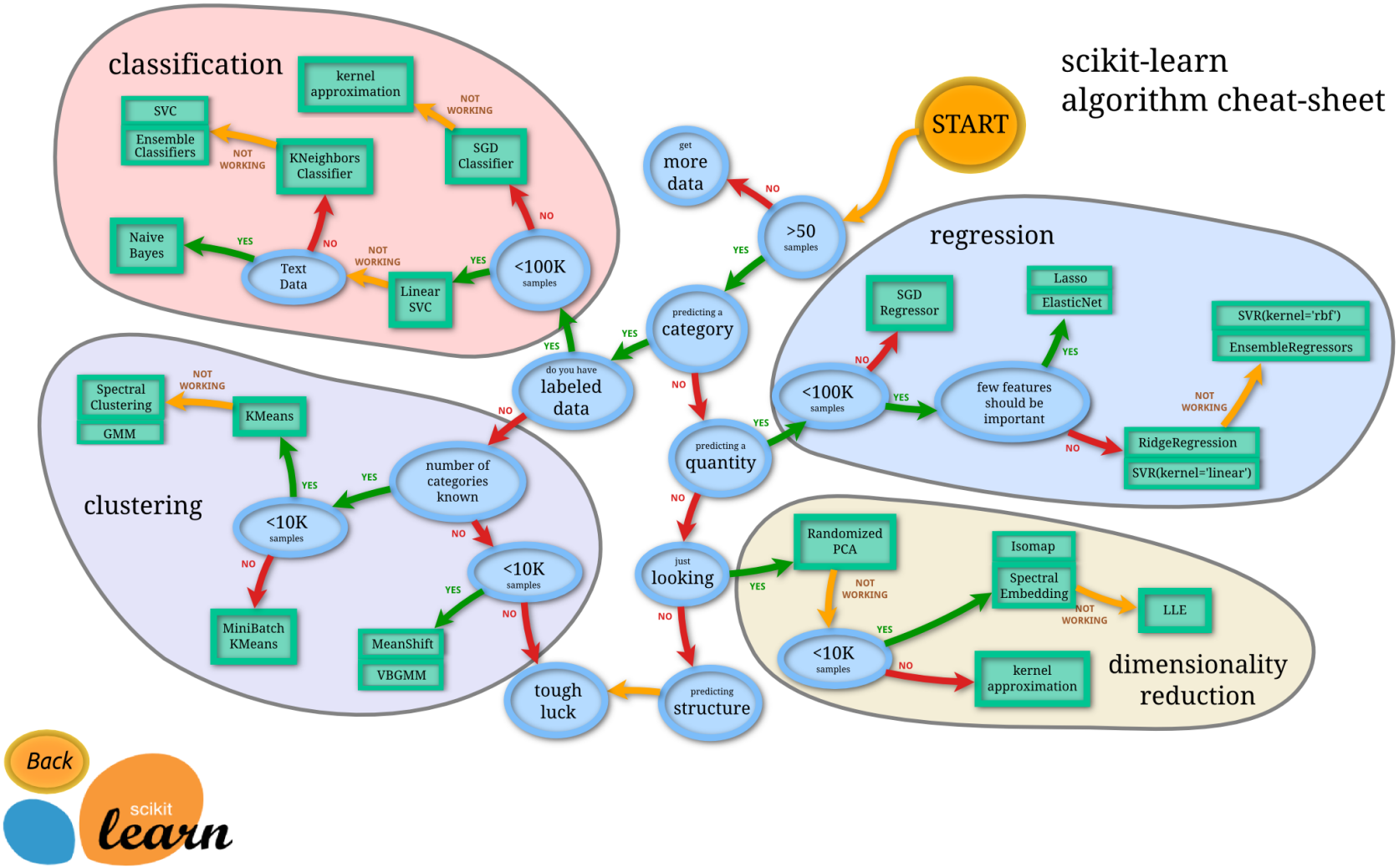
Problème de la dimensionnalité

- La **dimensionnalité** c'est le nombre de features/caractéristiques
- Plus la dimension est grande plus on va avoir besoin d'échantillons
- Plus le nombre de **classe** est grand plus on va avoir besoin d'échantillons
- Sinon faible **généralisation** $\hookrightarrow$ beaucoup de **variance** de l'erreur empirique

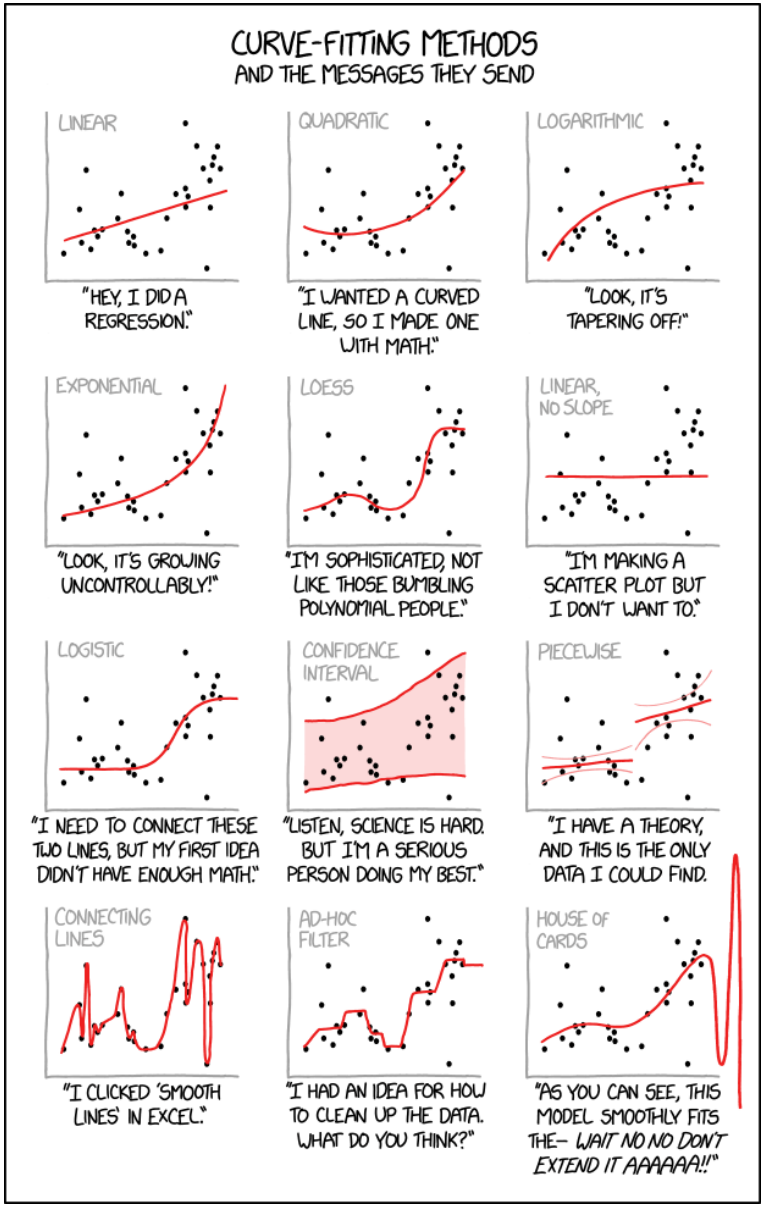
L'apprentissage supervisé

Choisir un modèle

En fonction du type de problème



Sélection de modèle : définir une métrique



Définir une métrique en lien avec notre problème

- **Classification** :
  - *Accuracy* : comment chaque classe est bien prédite
  - *F1-score* : prend en compte les erreurs de classification
  - *AUC* (ROC curve) : prend en compte la *sûreté* de la décision, adapté à des classes non-équilibrées
  - ...

Sélection de modèle : définir une métrique

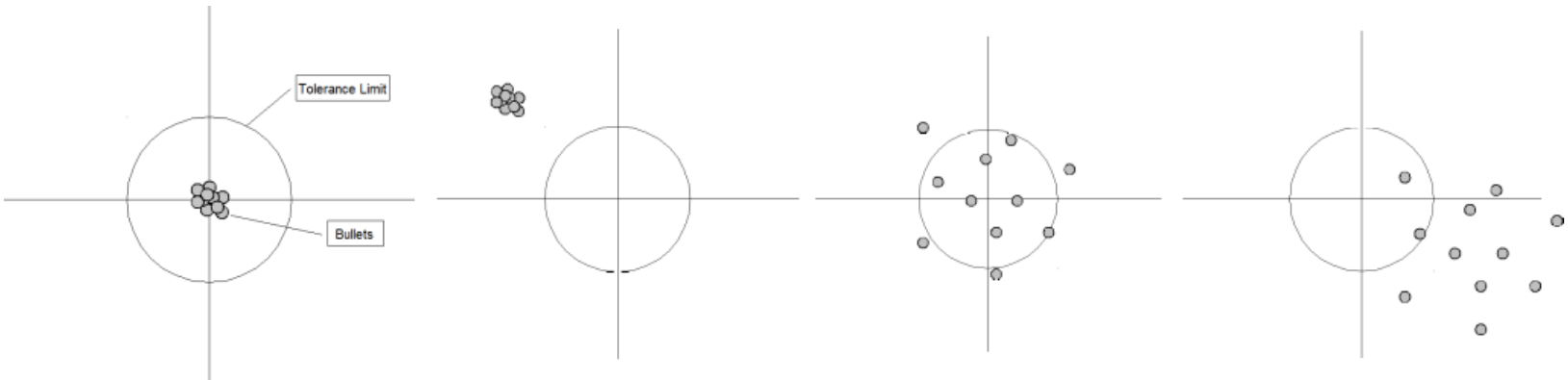
Définir une métrique en lien avec notre problème

- **Classification** :
  - *Accuracy* : comment chaque classe est bien prédite
  - *F1-score* : prend en compte les erreurs de classification
  - *AUC* (ROC curve) : prend en compte la *sûreté* de la décision, adapté à des classes non-équilibrées
  - ...
- **Régression** :
  - R^2 : la corrélation entre les prédictions et les labels
  - Erreur moyenne absolue
  - ...

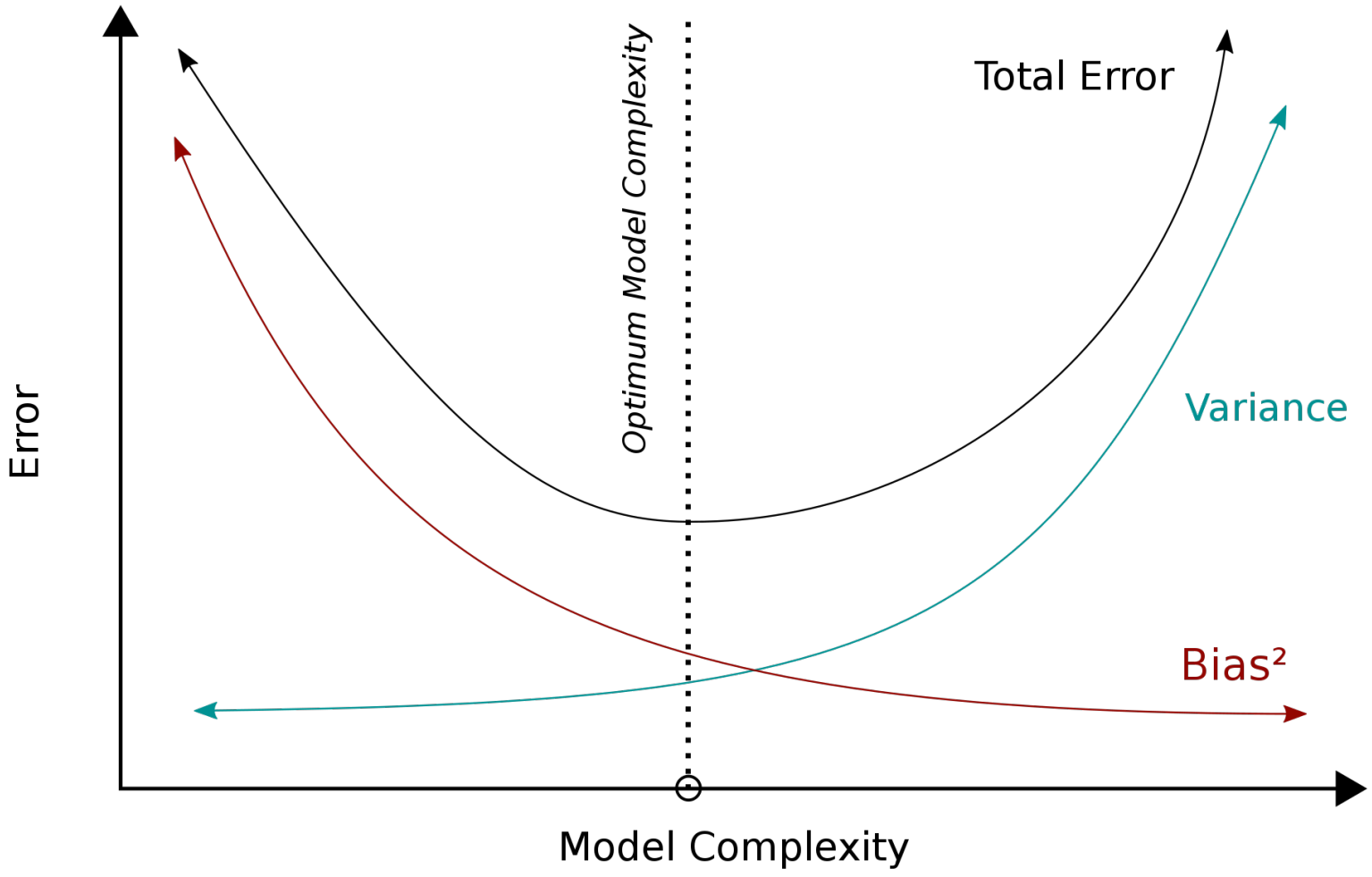
Sélection de modèle : définir une métrique

Sélectionner un modèle et régler ses paramètres

- **Comparer** les modèles entre eux : trade-off biais/variance
- **Sélectionner** les hyper-paramètres
- **Généralisation** de la décision



Sélection de modèle : définir une métrique



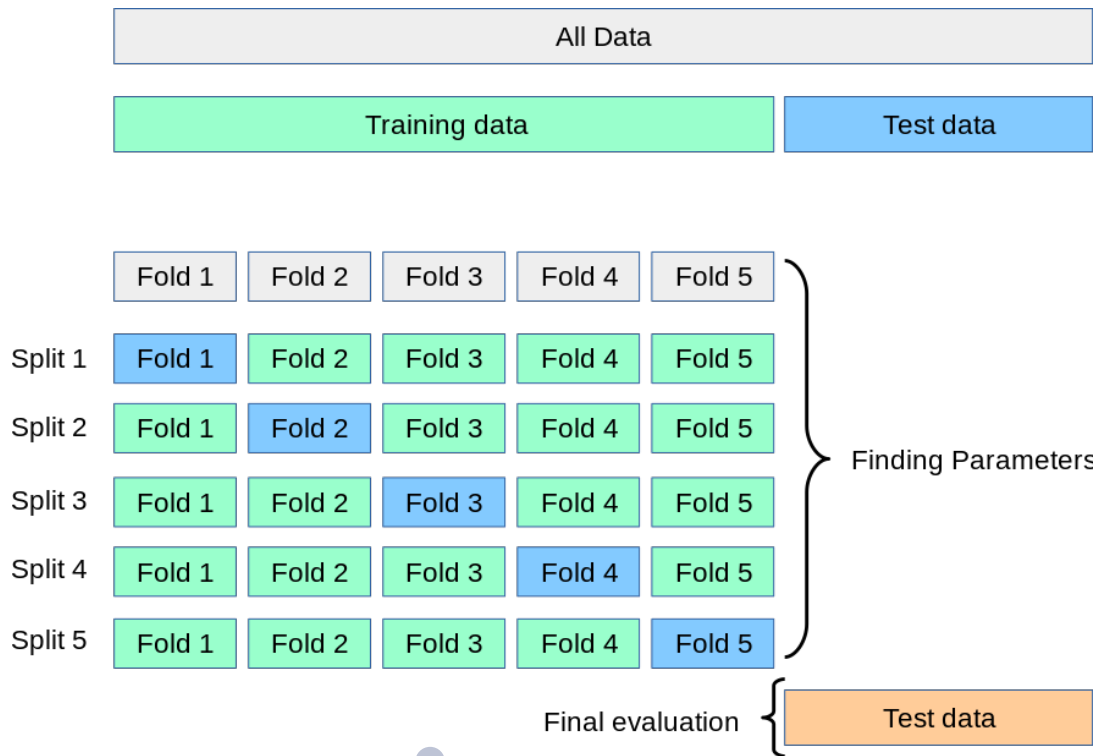
Séparation train/test et cross-validation

Train/validation/test split

- Risque d'overfitting : *sur-spécialisation* à l'échantillon de train
- Simuler l'arrivée de données nouvelles : **jeu de test**
- **Échantillon de validation** pour sélectionner les hyper-paramètres optimaux du modèle

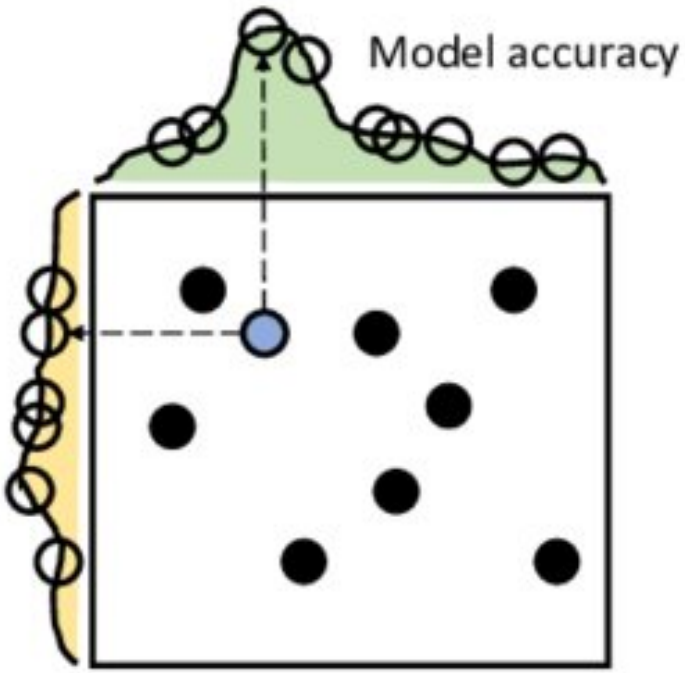
K-folds cross-validation

- Simuler un **jeu de validation plus large**
- Séparer le jeux de données en **K parties**, utiliser **K-1 parties pour le train** et **1 pour le test**, faire tourner et obtenir une performance moyenne et sa variance.



Sélectionner des hyper-paramètres : Grid Search

- **Quadriller** l'espace des paramètres du modèle : **grid**
- Recherche **exhaustive** ou **aléatoire** dans cet espace pour comparer les performances de cross-validation et trouver les valeurs optimales



L'apprentissage supervisé

Description de modèles

Régression linéaire

$$y_i = w_0 + w_1 x_{i1} + \cdots + w_p x_{ip} + \epsilon_i$$

Avec i le numéro de l'échantillon, p le nombre de features, w_p les coefficients du modèle et ϵ le terme d'erreur.

- Solution exacte dans le cas moindre carré avec des données gaussiennes
- Descente de gradient pour d'autres fonctions d'erreur.

Régularisation : réduire la complexité

- Contraindre le vecteur de poids w et imposer une parcimonie (sparse) : mettre des poids à 0
- **Ridge** : régularisation de la norme L^1 (valeur absolue) du vecteur de poids
- **Lasso** : régularisation de la norme L^2 (norme euclidienne) du vecteur de poids
- **Elastic-net** : Ridge et Lasso simultanément

Linear Discriminant Analysis

- Apprends une distribution gaussienne pour chaque classe et génère une frontière de décision linéaire suivant la **règle de Bayes**
- Une seule matrice de covariance pour toutes les classes
- Peut s'utiliser pour réduire la dimension (alors équivalent à faire une PCA par classe)

Linear Discriminant Analysis

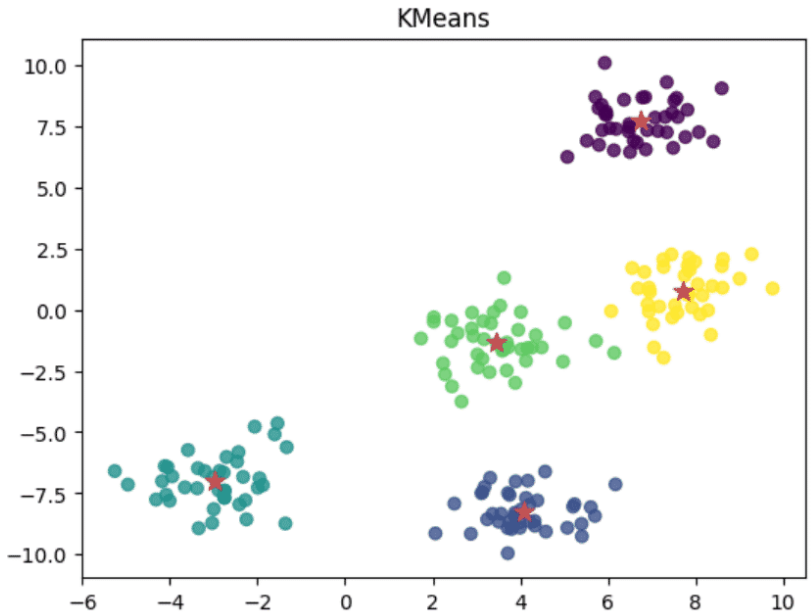
- Apprends une distribution gaussienne pour chaque classe et génère une frontière de décision linéaire suivant la **règle de Bayes**
- Une seule matrice de covariance pour toutes les classes
- Peut s'utiliser pour réduire la dimension (alors équivalent à faire une PCA par classe)

Régression logistique

- Combinaison **linéaire** de features (régression linéaire)
- Ajout d'une fonction logit en sortie : $p(X) = \frac{1}{1+e^{-(w_0+w_1 X)}}$
- Résolution par descente de gradients pour minimiser la fonction d'erreur

K-means

- Recherche de la position de K barycentres pour partitionner l'espace
- Initialisation aléatoire puis itérations avec déplacement du centre de gravité
- S'utilise aussi de manière non-supervisée



K-means

- Recherche de la position de K barycentres pour partitionner l'espace
- Initialisation aléatoire puis itérations avec déplacement du centre de gravité
- S'utilise aussi de manière non-supervisée

k-NN

- Recherche des **k plus proches voisins** de chaque exemple
- Le voisinage doit partager le même label

Arbre de décision simple

- Apprentissage automatique de règles **"if-then-else"**
- Méthode **non-paramétrique**, facile à **interpréter** et visualisation simple
- Utilisable aussi en régression
- Différentes fonctions possibles pour mesurer la qualité d'un split et donc l'optimiser : coefficient de Gini, entropy, logarithmic loss
- Rapidement **overfiting**, **beaucoup de paramètres** à régler

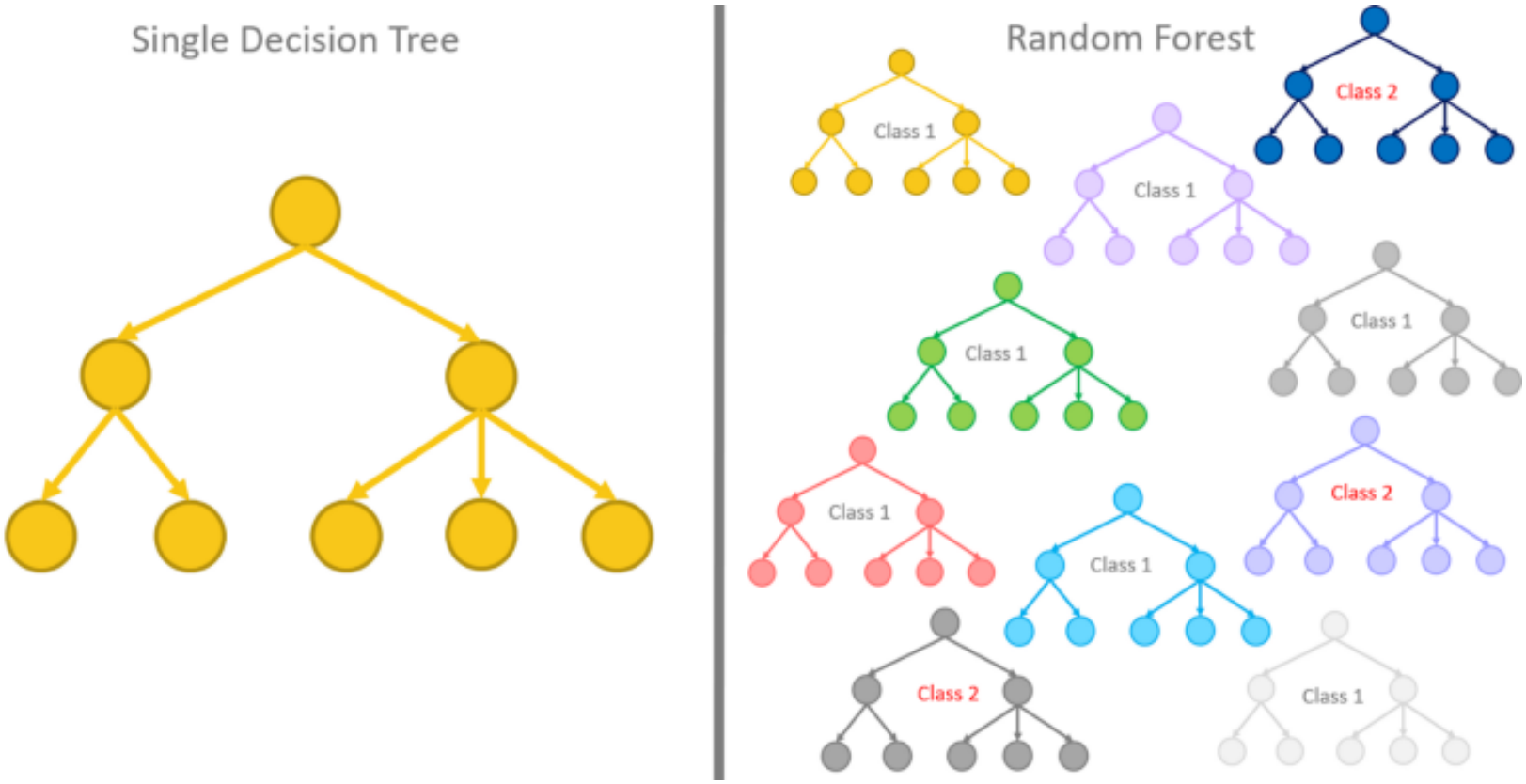
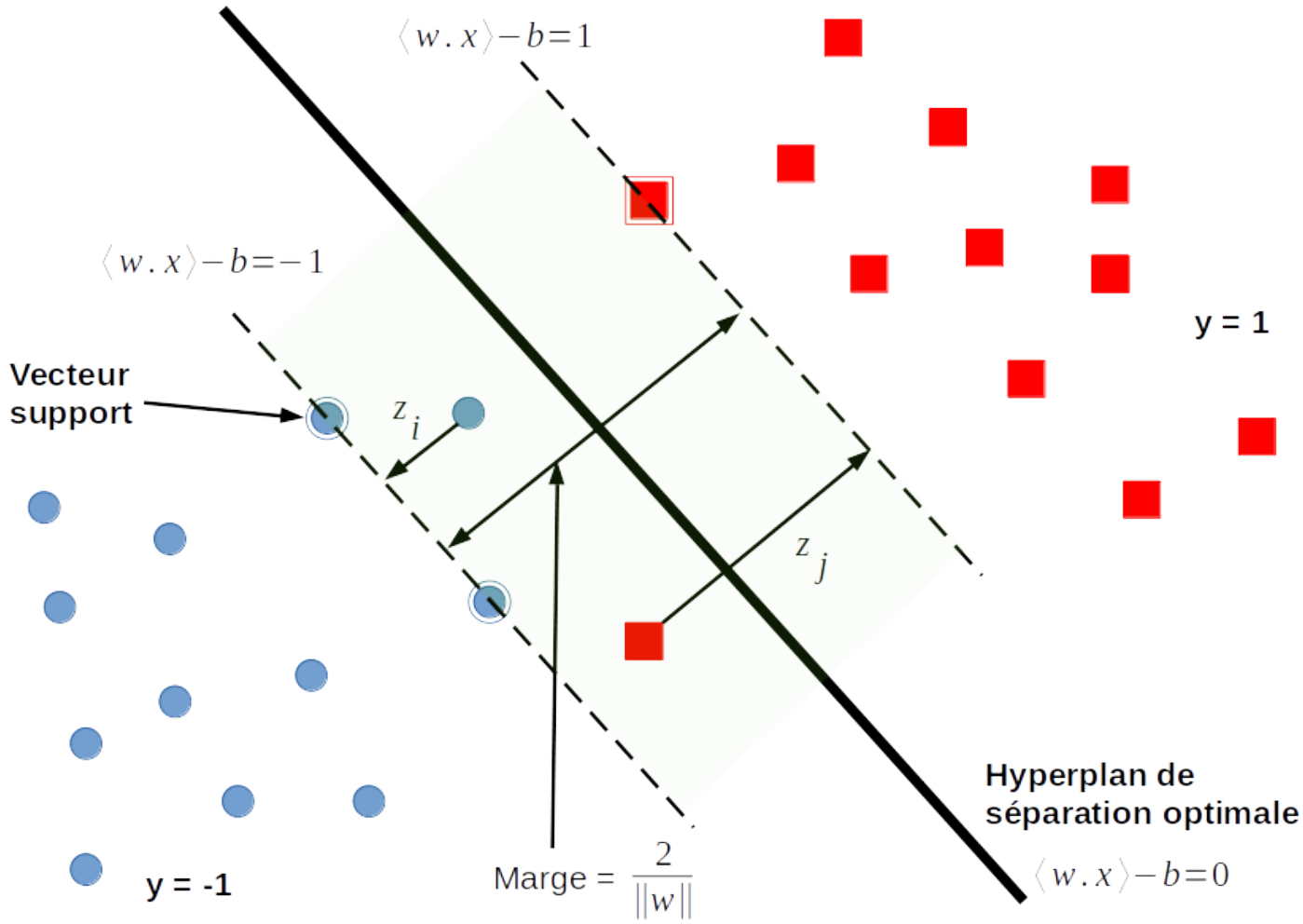


Figure 1. Random Forest

Gradient Boosting : ensemble learning

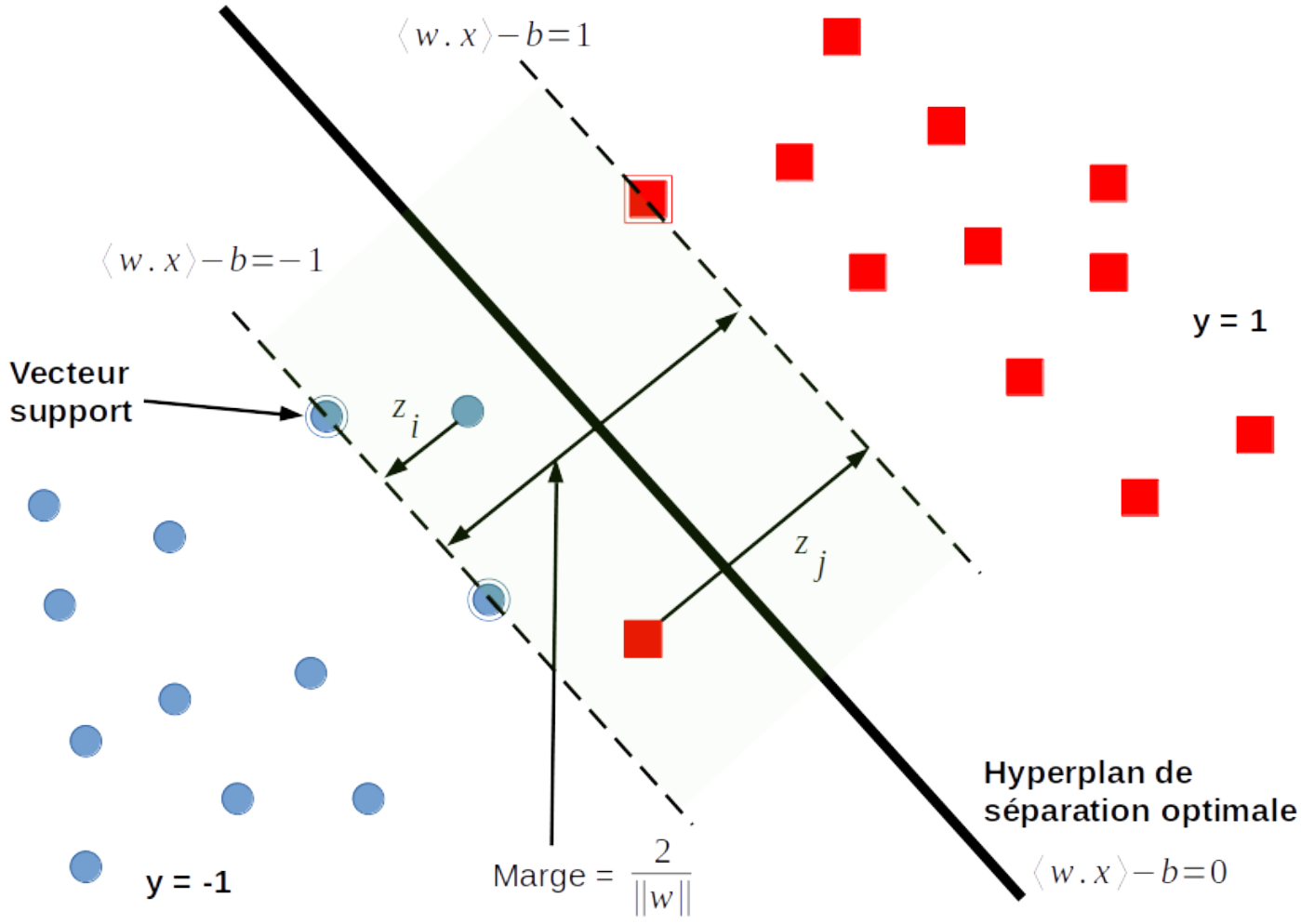
- Apprentissage **itératif** :
  * Un ensemble de **modèle "faibles"** (arbres de décision) pour faire un **modèle "fort"**
  * On cherche à chaque itération à re-classer les données mal classées : $F_{m+1}(x_i) = F_m(x_i) + h_m(x_i) = y_i$ avec h_m le nouvel estimateur "faible".
  * Dans les **résidus** : $h_m(x_i) = y_i - F_m(x_i)$, les données mal classées ont plus de poids $\hookrightarrow$ **boosting**
  * Souvent combiné avec du **bootstrap aggregating** (*bagging*) : XGBoost

Machine à support de vecteur pour la classification (SVM)



Hinge Loss : $\lambda||\mathbf{w}||^2 + \left[\frac{1}{n}\sum max(0, 1 - y_i\left(\mathbf{w}^T\mathbf{x}_i - b\right))\right]$

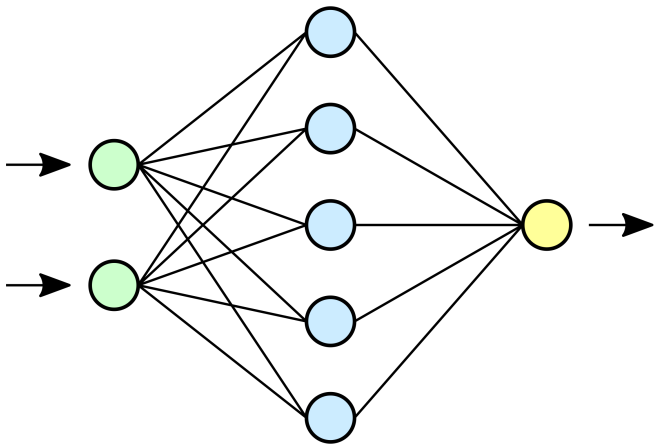
Machine à support de vecteur pour la classification (SVM)



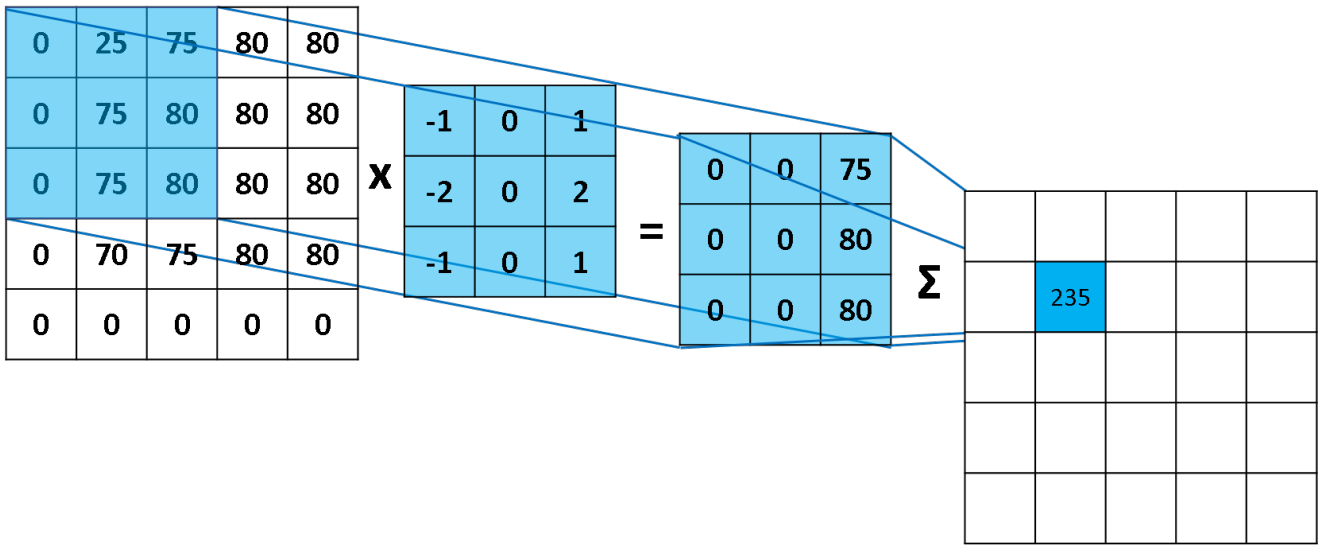
Kernel trick : projeter les données dans un autre espace dont on connaît le produit scalaire

Réseaux de neurones : multi-layers perceptron

- **Perceptron** : équivalent d'une régression logisitque
- Optimisation **couche par couche** en partant de la fin : backpropagation des erreurs
- Descente de gradient **stochastique**
- Problème de *l'évanouissement du gradient*

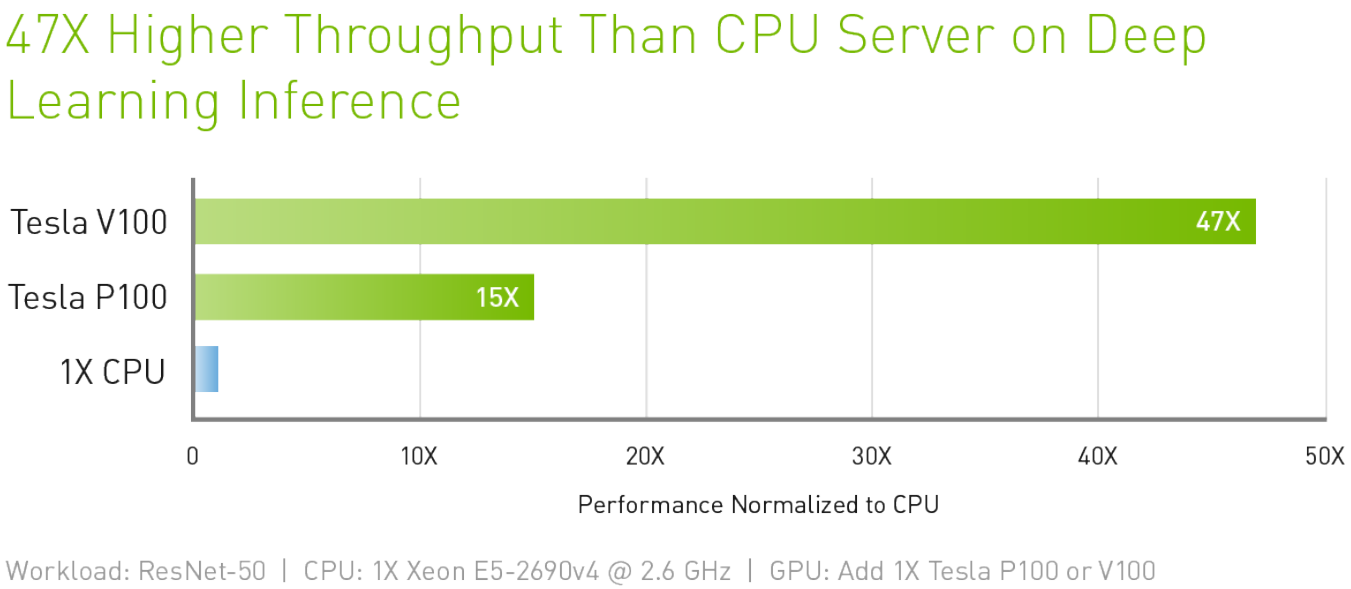


Hypothèse de **d'invariance par translation** : convolution par un même noyau de toute l'image

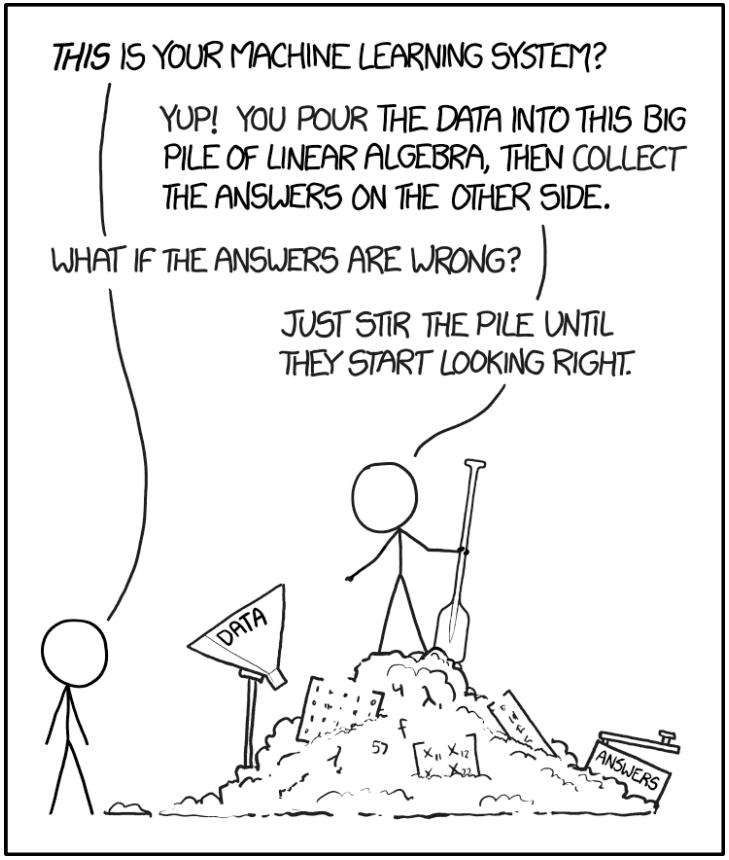


Révolution depuis 2012 :

- Volumes de données bien plus **larges**
- **GPU** plus performants
- Backpropagation : 1960s, CNNs : 1995

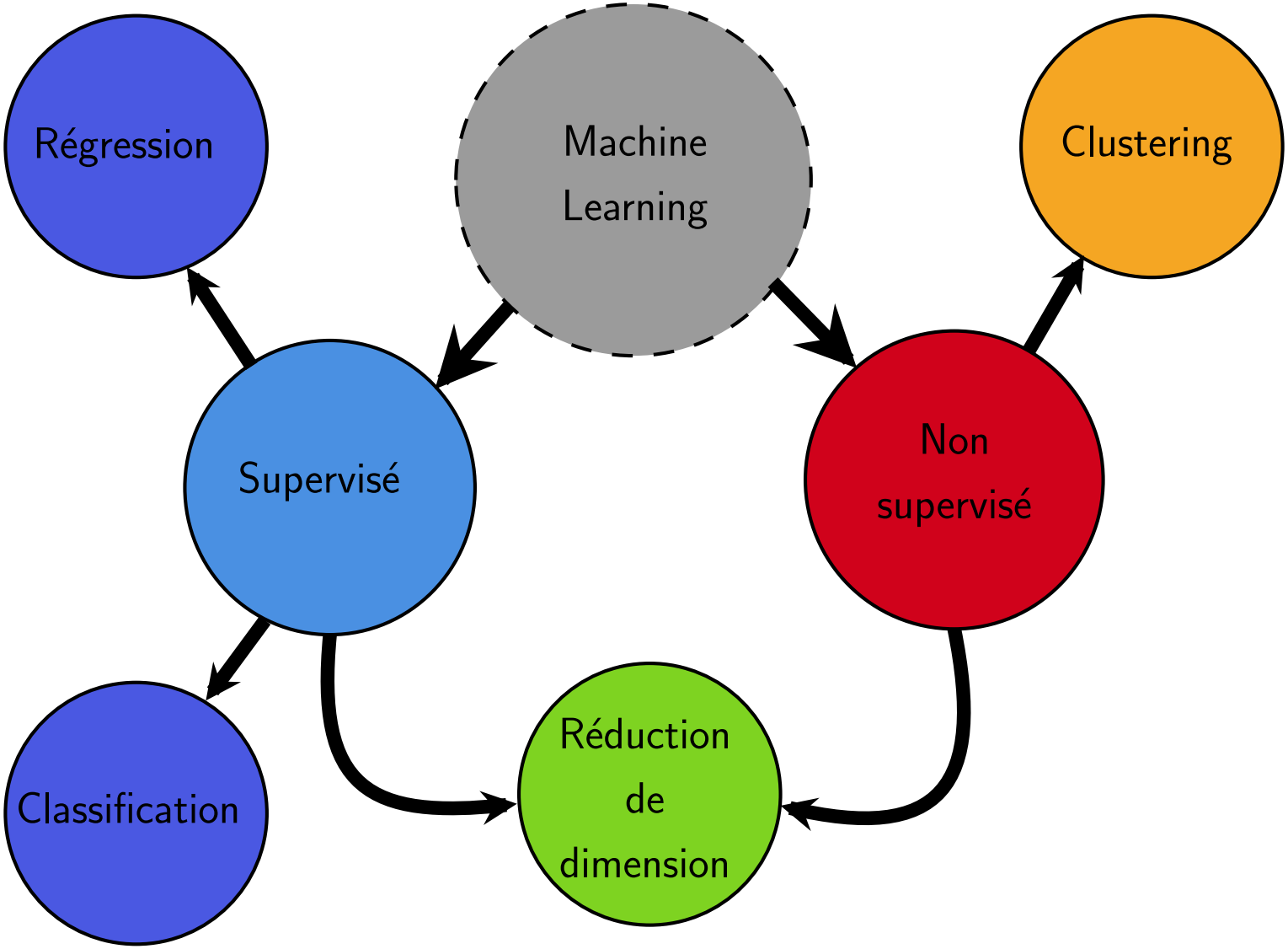


Beaucoup *d'ingénierie* pour avoir un réseau performant : évanouissement du gradient, choix de l'optimizer, pooling, dropout, residual and inception layers,...



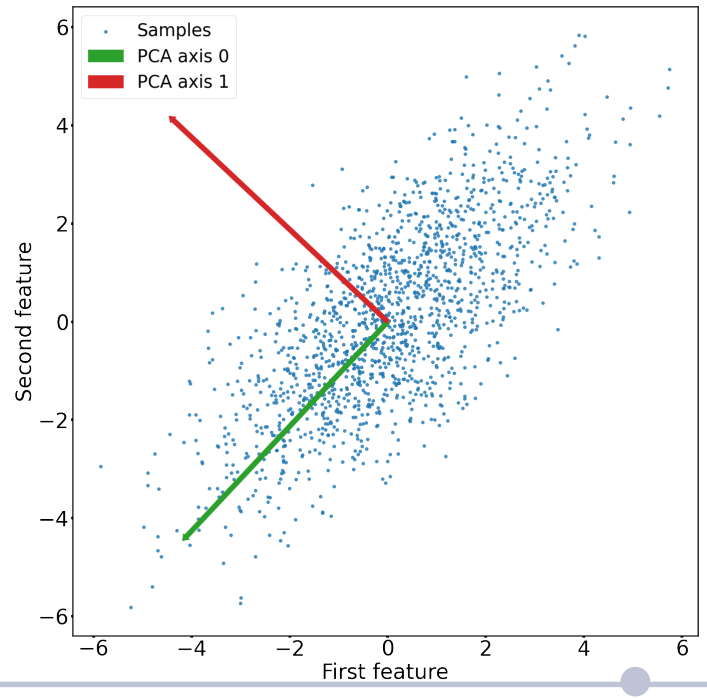
Réduction de dimensions

Les différents types de problèmes



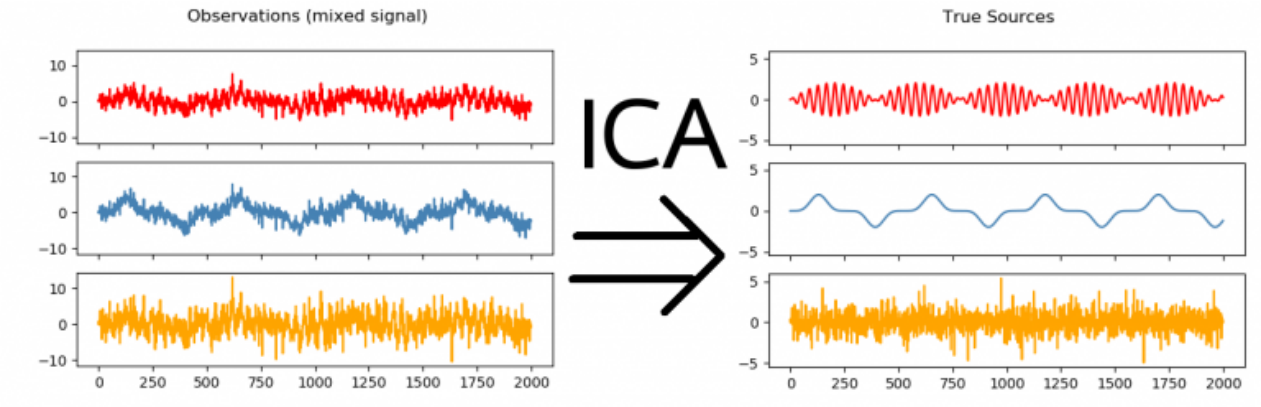
Principal Components Analysis (PCA)

- Hypothèse de *séparation linéaire* des données
- Cherche des axes de projections orthogonaux qui maximise la variance des données
- En pratique on cherche donc une **matrice de rotation** U



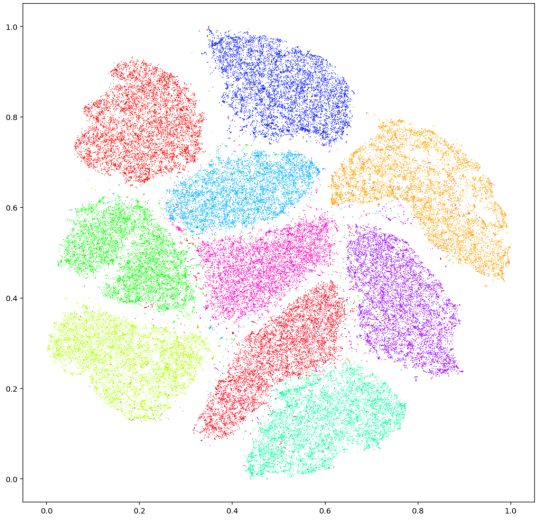
Independent Component Analysis (ICA)

- Hypothèse de n *composantes indépendantes* dans les données
- $X = AS$ avec S un set de n sources indépendantes
- **Changement de base** $\rightarrow$ composantes indépendantes
- Proche du *sparse dictionnary learning* où la contrainte est alors la parcimonie des sources (en avoir le moins possible)



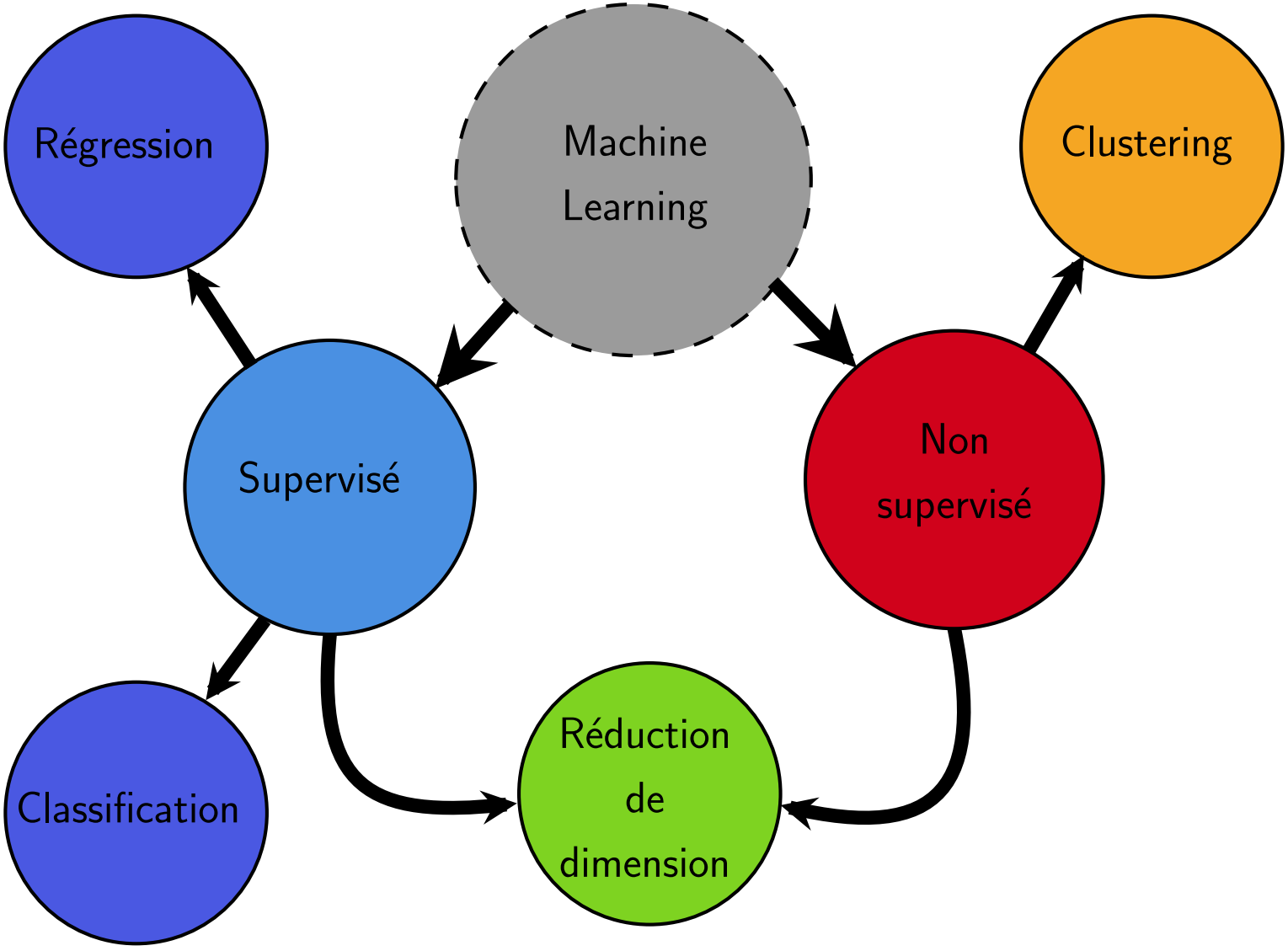
t-distributed Stochastic Neighbor Embedding (t-SNE)

- Transformation non-linéaire
- Visualisation de données en 2D : des points avec des features similaires doivent être proches
- Beaucoup de paramètres à régler, distances entre groupes non-interprétables



Apprentissage non-supervisé

Les différents types de problèmes



$\hookrightarrow$ Associer des exemples **proches dans l'espace**

Affinity propagation

- **Matrice de similarité** : par exemple distance euclidienne entre 2 points
- Un *"échantillon"* sert de point de départ puis un *"message"* est transmis entre noeuds du graph en fonction de la similarité

$\hookrightarrow$ Associer des exemples **proches dans l'espace**

Affinity propagation

- **Matrice de similarité** : par exemple distance euclidienne entre 2 points
- Un *"échantillon"* sert de point de départ puis un *"message"* est transmis entre noeuds du graph en fonction de la similarité

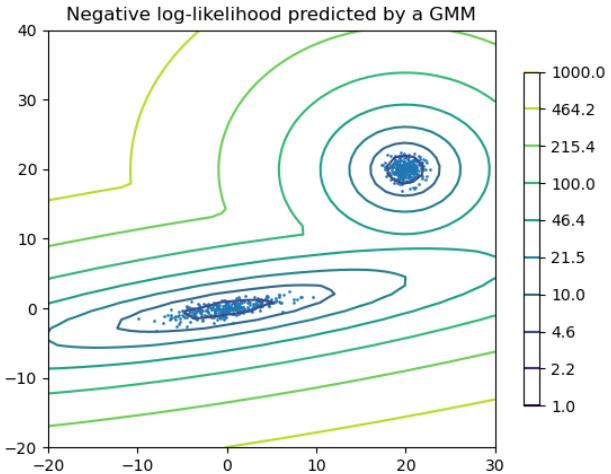
Density-based spatial clustering (DBSCAN)

- Cherche quels points sont *"accessibles"* entre eux : rayon autour de chaque point
- Permet d'exclure des outliers : points seuls et non-accessibles

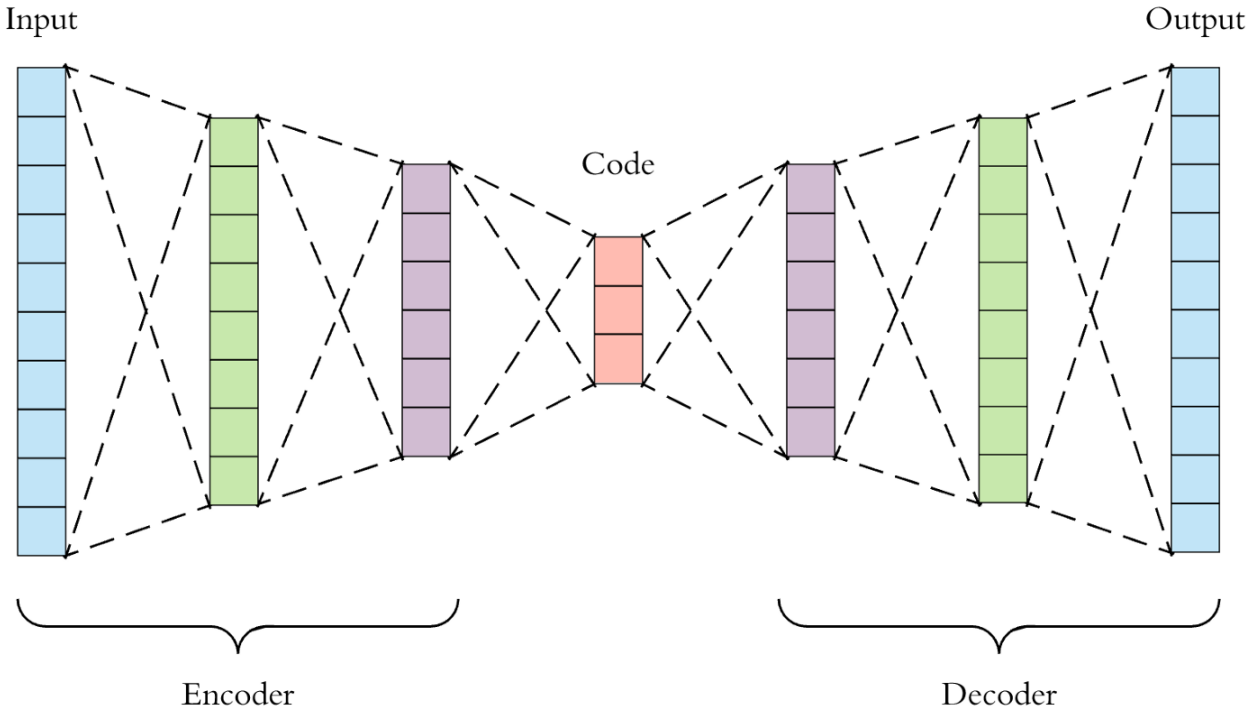
$\hookrightarrow$ K-means, k-NN, ICA déjà présentés

Gaussian Mixture Model (GMM)

- Similarité avec k-means mais covariances non sphériques
- Expectation-Maximization (EM) pour l'apprentissage



Auto-encoders, Variational Auto-Encoders (VAE), GAN



Les outils pour le machine learning

L'environnement

TensorFlow

matplotlib

PyTorch

Alternatives

- Mais aussi historiquement R : communauté des statisticiens mais possibilité hors machine learning plus limitées (hormis *R Shiny*)

Alternatives

- Mais aussi historiquement R : communauté des statisticiens mais possibilité hors machine learning plus limitées (hormis *R Shiny*)
- Python reste un langage avec des **performances moyennes** (pas d'allocation, dynamic typing, for loop, etc)

Alternatives

- Mais aussi historiquement R : communauté des statisticiens mais possibilité hors machine learning plus limitées (hormis *R Shiny*)
- Python reste un langage avec des **performances moyennes** (pas d'allocation, dynamic typing, for loop, etc)
- Julia tente de faire mieux en gardant les avantages

- **Visual Studio Code**, PyCharm, Spyder, Jupyter Notebook : environnements de développement

Outils de programmation

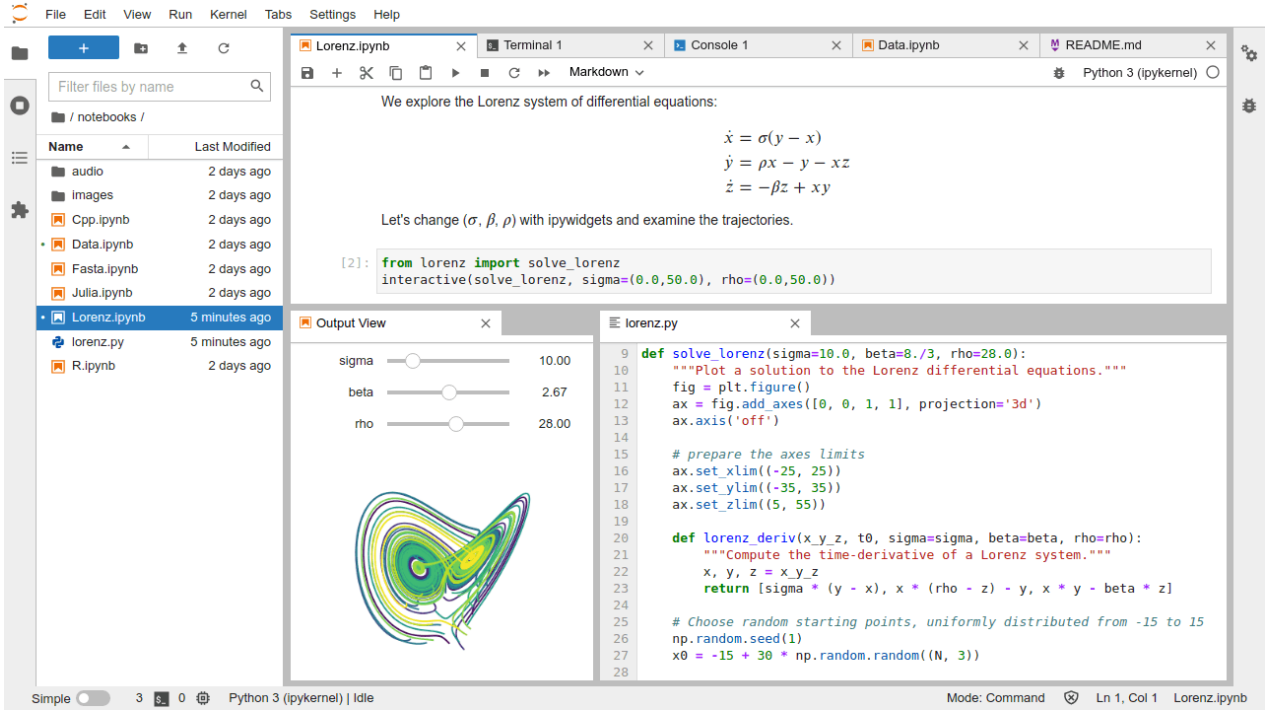
- **Visual Studio Code**, PyCharm, Spyder, Jupyter Notebook : environnements de développement
- Git, **GitHub**, GitLab : contrôle de version, partager du code

Outils de programmation

- **Visual Studio Code**, PyCharm, Spyder, Jupyter Notebook : environnements de développement
- Git, **GitHub**, GitLab : contrôle de version, partager du code
- Anaconda : gestion de packages et environnement virtuels

Visual Studio Code

Jupyter Notebook



Exécuter le code **par blocs**, mêler des figures au milieu, **éditer des rapports**, construire une présentation

Online computing : effectuer les calculs sur une machine à distance

- **Google Collab**, MyBinder : notebook en ligne (avec GPU)

Online computing : effectuer les calculs sur une machine à distance

- **Google Collab**, MyBinder : notebook en ligne (avec GPU)
- AWS EC2, Microsoft Azure, Google Cloud : cloud computing, particulièrement intéressant pour accéder à des gros GPU

Online computing : effectuer les calculs sur une machine à distance

- **Google Collab**, MyBinder : notebook en ligne (avec GPU)
- AWS EC2, Microsoft Azure, Google Cloud : cloud computing, particulièrement intéressant pour accéder à des gros GPU
- **AutoML**, Cloud Vision API, Amazon Sage Maker : API de ML automatiques, souvent sans code (mais cher)

Online computing : effectuer les calculs sur une machine à distance

- **Google Collab**, MyBinder : notebook en ligne (avec GPU)
- AWS EC2, Microsoft Azure, Google Cloud : cloud computing, particulièrement intéressant pour accéder à des gros GPU
- **AutoML**, Cloud Vision API, Amazon Sage Maker : API de ML automatiques, souvent sans code (mais cher)
- Labelisation manuelle de données : Amazon Mechanical Turk, ClickWorker, Appen, Tellus international

Pour aller plus loin

Livre et cours en ligne pour approfondir la théorie

- *The Element of Statistical Learning*, **Hastie** et al. et son MOOC associé : `https://www.edx.org/course/statistical-learning`
- *Pattern Recognition and Machine Learning*, Chirstopher M. **Bishop**
- Cours en ligne *Coursera* de **Andrew Ng**

Mettre en pratique sur des données

- Participer à des compétitions sur **Kaggle**
- Explorer des notebooks et datasets publiques de **Kaggle**

Des questions ?