Chapitre 1 : Généralités sur l'IA

Jordy Palafox

September 22, 2024

Intelligence Artificielle pour la Cybersécurité
CY Tech - Ing 3 CS 2024-2025

Programme

Dans ce cours, on fera :

- un rappel sur les méthodes classiques de machine learning,
- des rappels sur le deep learning avec les réseaux entièrement connectés,
- du traitement d'images appliqué à l'analyse de malwares,
- NLP ou TAL, Transformers, LLM et applications,
- Autoencodeurs, isolation forest et détection d'anomalie et nouveautés,
- Generative deep learning ?

Modalités d'évaluation

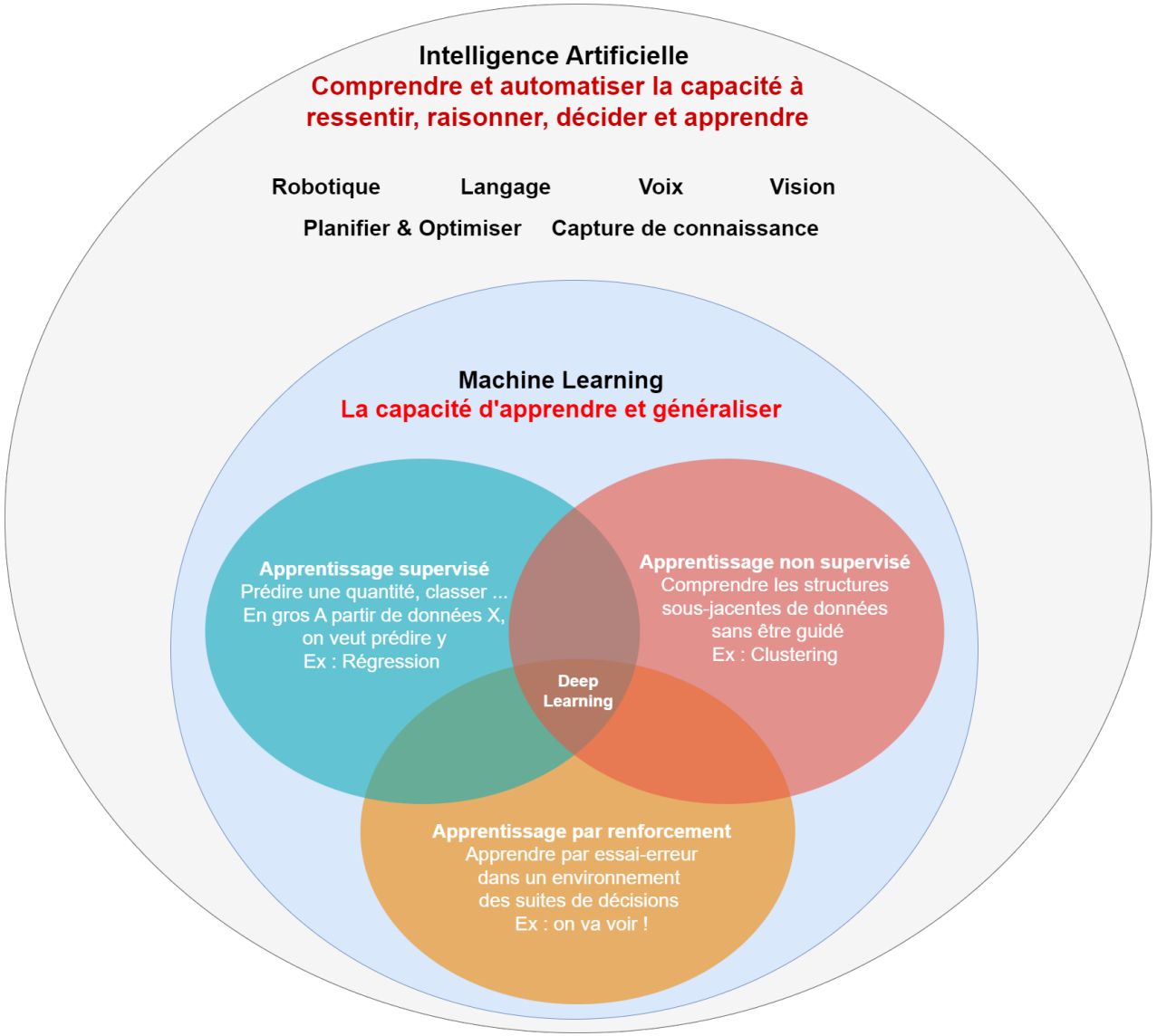
Nous disposerons de $\pm$ 10 créneaux .

Evaluation

- Trouver un article sur les applications de l'IA pour la cyber par groupe de 3-4,
- Explorer, synthétiser, vulgariser et faire émerger l'idée principale de l'article,
- Présenter les résultats lors du dernier cours en présence de toute la promo.

Livrables : rapport résumant l'article + présentation.

Horizon IA / ML / DL



On peut rajouter l'apprentissage semi-supervisé aussi ...
`https://www.ibm.com/topics/semi-supervised-learning`

Le socle commun à l'apprentissage : les données

"Selon Statista, selon le rapport de la Commission européenne, environ 328,77 millions de téraoctets, soit 0,33 zettaoctets, de données sont créés chaque jour. Cela représente environ 2,31 zettaoctets par semaine et 120 zettaoctets par an, ce qui illustre l'immensité de la production de données."[1]

[1] `https://innowise.com/`

Quelques applications de l'IA

- Diagnostique médical via IRM, radio, etc utilisant de la Convolution pour détecter des anomalies

- Prédiction de fraudes dans les transactions bancaires sur des paiments/virements les classifiant avec Random forest, regression logistique

- Maintenance en usine de machines reposant sur des données par capteur utilisant des données type séries temporelles et des modèles LSTM ou ARIMA

- Voitures autonomes, Recommandations de produits, Tri automatique de CV et recrutement, Analyse de sentiments dans les réseaux sociaux, Optimisation des cultures grâce à la vision par ordinateur ...

Et en cyber ?

- Détection de menaces, par exemple l'entreprise DARKTRACE `https://darktrace.com/`
- Analyse de malwares, Cylance `https://www.blackberry.com/fr/fr/products/cylance-endpoint-security/cylance-ai`,
- Intrusion detection systems, Vectra AI `https://www.vectra.ai/`
- Prévention du phishing, Barracuda Sentinel `https://assets.barracuda.com/assets/docs/dms/DS_Sentinel_1-3_FR.pdf`
- Analyse des comportements des utilisateurs, Exabeam `https://www.exabeam.com/`,
- etc

Formalisation d'un problème de Machine Learning

- On a des **données** à interprêter X,
  Mesures, texte, image, enregristrement, etc
- On veut faire une **prédiction** Y,
  prendre une décision, groupe, préférence, commande, valeur, etc
- On dispose d'un **échantillon** $D = \{(x_i, y_i), i \in \{1, ..., N\}\}$,
  qui sert de base d'apprentissage, ceux sont des vraies valeurs.

Peut-on trouver une fonction (paramétrique) F_θ telle que :

$$F_\theta(X) \simeq Y$$

On introduit une **fonction de perte** L qui mesure la différence entre la valeur prédite et la valeur à prédire de sorte que :

$$\hat{\theta} = \underset{\theta \in \mathbb{R}^n}{argmin}\ L(F_\theta(X), Y)$$

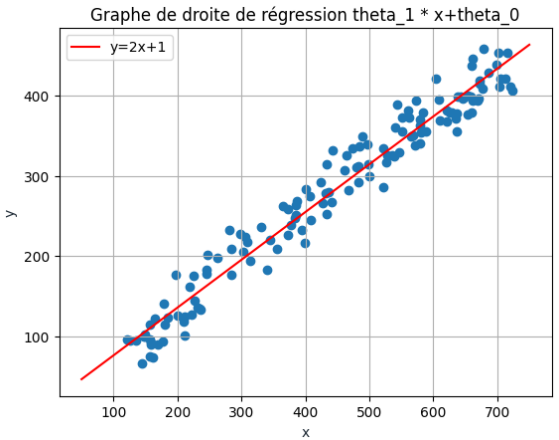
Modèles de régression

1. Régression linéaire univariée : X, Y sont à valeurs réelles, on veut une relation du type :

$$f(X) = aX + b$$

en minimisant l'erreur :

$$J(b, a) = \frac{1}{2n} \sum_{i=1}^{n} (ax_i + b - y_i)^2$$



Modèles de régression

2. Régression linéaire multivariée: $X \in \mathbb{R}^{n \times p}$, $Y \in \mathbb{R}$, on veut une relation du type :

$$f(X) = b + a_1 X_1 + \ldots + a_p X_p$$

en minimisant l'erreur

$$J(b, a_1, \ldots, a_p) = \frac{1}{2n} \sum_{i=1}^{n} (f(x_i) - y_i)^2$$

Standardisation et mélange d'échelles de valeurs

Si les X_i sont sur des échelles différents, on standardise (dès lors que les valeurs utilisent des opérations classiques $\times, +, -, ...$) :

- StandardScaler : $\tilde{X} = \frac{X - \mathbb{E}(X)}{\sigma}$
- MinMaxScaler : $\tilde{X} = \frac{X - min(X)}{max(X) - min(X)}$

Modèles de régression

3. Régression polynomiale : $X \in \mathbb{R}^{n \times p}$, $Y \in \mathbb{R}$, on veut une relation du type :

$$f(X) = b + a_1 X_1 + a_2 X_2 + a_3 X_3^2 + a_4 X_2^5$$

en minimisant la même erreur (mais il en existe d'autres !)

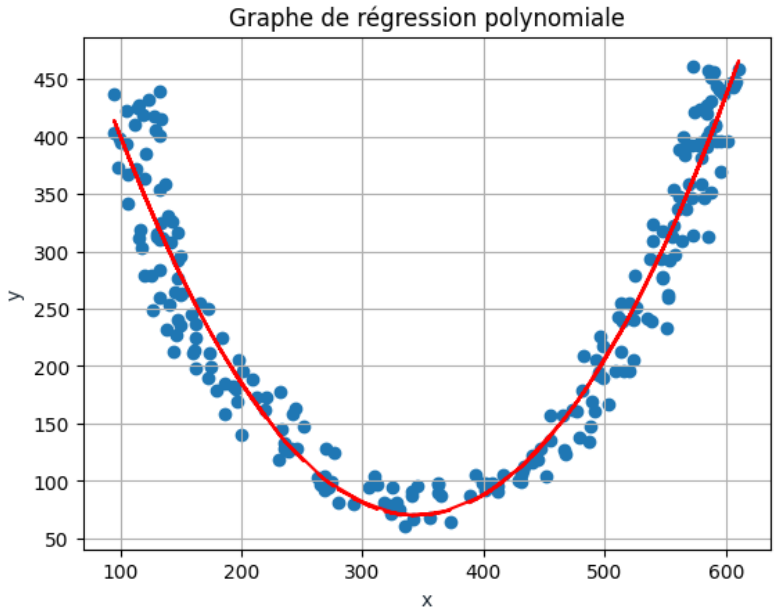
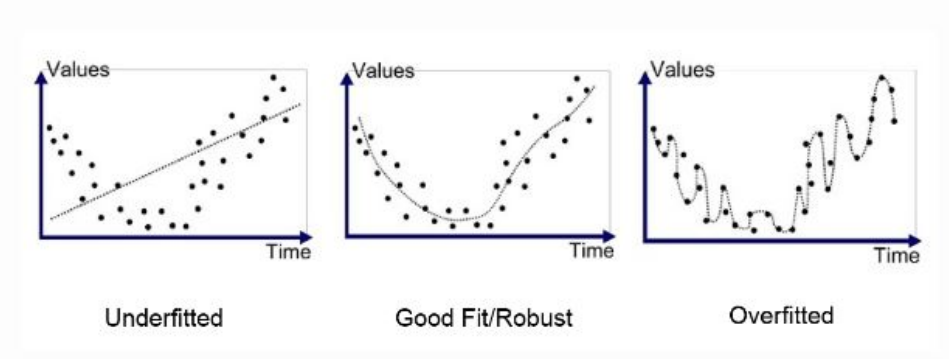
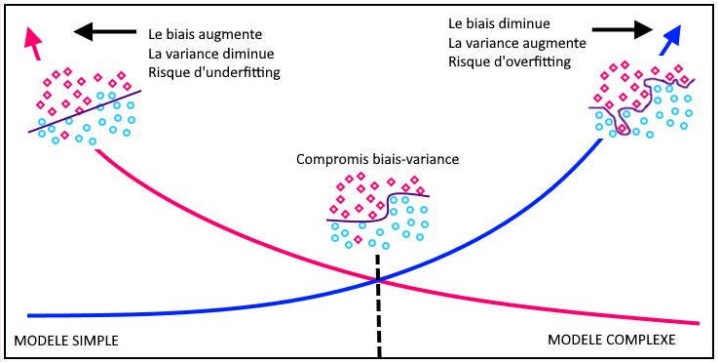


Figure: En degré 2

3. Régression polynomiale

Choix du degré, sur-apprentissage et compromis biais variance





Ce phénomène n'est pas spécifique à la regression mais à tout problème de machine learning !

4. Régression logistique

Ici $Y \in \{0, 1\}$, c'est de la **classification binaire**. Le modèle est :

$$f(X\theta) = g(b + a_1X_1 + \ldots + a_pX_p)$$

où $\theta = (b, a_1, ..., a_p)$ et g est la fonction logistique définie par :

$$g(t) = \frac{1}{1 + e^{-t}}$$

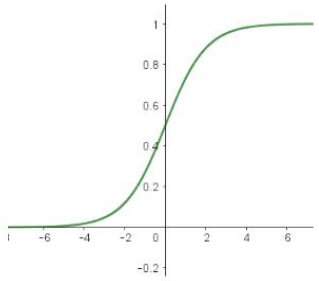


Figure: Fonction logistique

On prédit 1 si $g(X\theta) \geq 0$ et 0 si $g(X\theta) \leq 0$ ce qui équivaut à 1 si $X\theta \geq 1$ et 0 si $X\theta \leq 0$

4. Régression logistique
   Ici la fonction d'erreur est l'**entropie croisée** :
   $$j(f(X), y) = -ylog(f(x)) - (1-y)log(1-f(x))$$

   La fonction de perte est :
   $$J(\theta) = \frac{1}{n}\sum_{i=1}^{n} j(f(x_i), y_i).$$

   L'optimisation (minimisation) se fait par l'algorithme du gradient :

Algorithme du gradient

1. Initialiser $(\theta_0, \theta_1, ..., \theta_p)$
2. Itérer jusqu'à convergence de :
   $$\theta_j := \theta_j - \alpha \frac{\partial J}{\partial \theta_j}(\theta),\ j = 0, ..., p$$

Il existe d'autres méthodes d'optimisation (BFGS par exemple). Le gradient s'applique à tous les modèles introduits précédemment.

5. Régression régularisée

A la fonction de perte L, on ajoute un terme de pénalisation :

$$J(\theta) = \frac{1}{2m}\sum(f(x_i) - y_i)^2 + P(\lambda, \theta)$$

- Régression Ridge si $P(\lambda, \theta) = \lambda \parallel \theta \parallel_2 = \lambda \sum \theta^2$
- Régression Lasso si $P(\lambda, \theta) = \lambda \parallel \theta \parallel_1 = \lambda \sum |\theta|$
- Régression ElasticNet si si $P(\lambda, \theta) = \lambda \sum \left(\frac{1}{2}(1-\alpha)\theta_j^2 + \alpha|\theta_j|\right)$

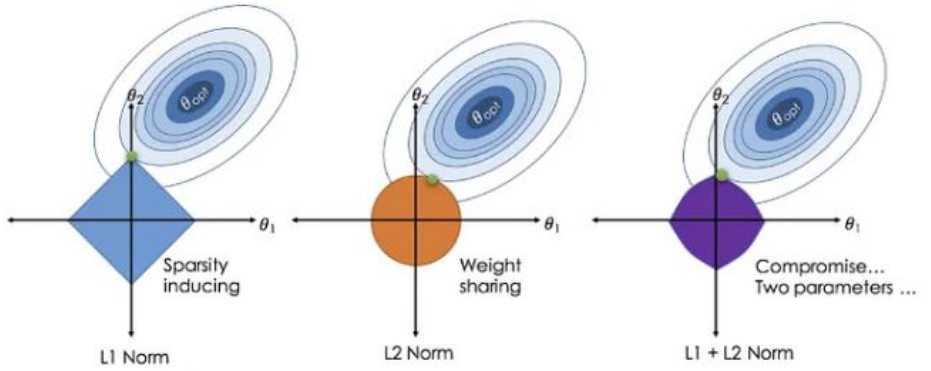


Figure 1: An image visualising how ordinary regression compares to the Lasso, the Ridge and the Elastic Net Regressors. Image Citation: Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net.

Ridge si les variables sont indépendantes et corrélées, Lasso pour réduire le nombre de variables.

Soit C le nombre de classes souhaitées.

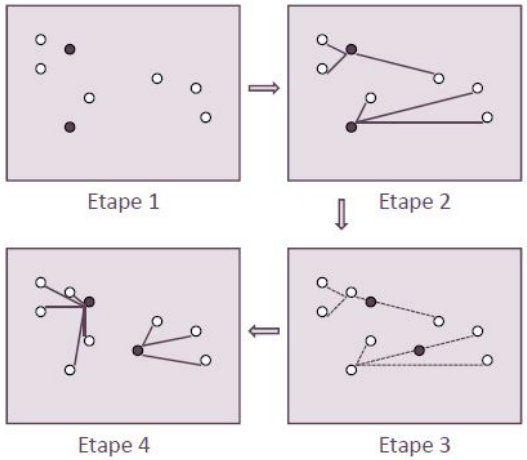
Algorithme

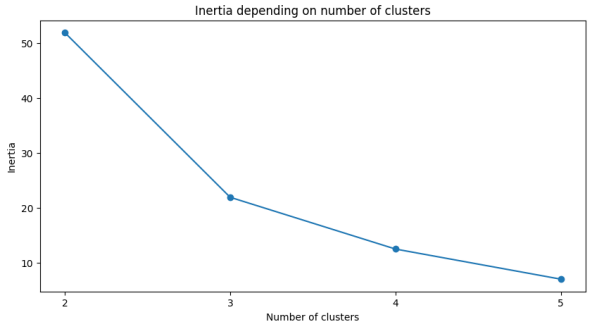
Etape 1 : Choisir C individus au hasard comme centres initiaux des classes

Etape 2 : On calcule les distances entre chaque individu et chaque centre de classe, et on affecte l'individu à la classe la plus proche

Etape 3 : On remplace les centres des classes par les C barycentres des classes définies à l'étape 2

Etape 4 : On itère à partir de l'étape 2 jusqu'à convergence





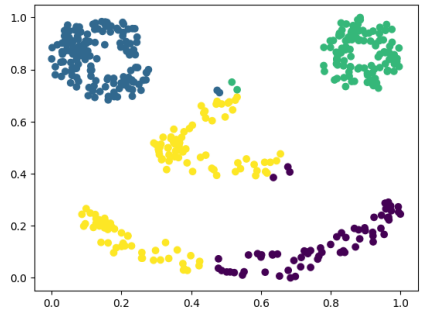


Figure: KMeans le classique

Clustering

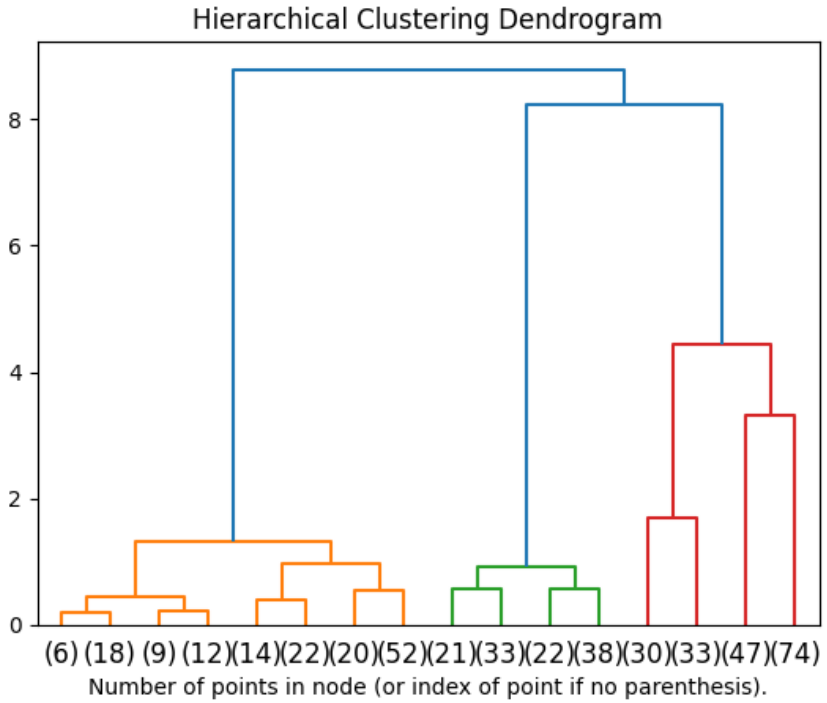


Figure: Agglomerative clustering

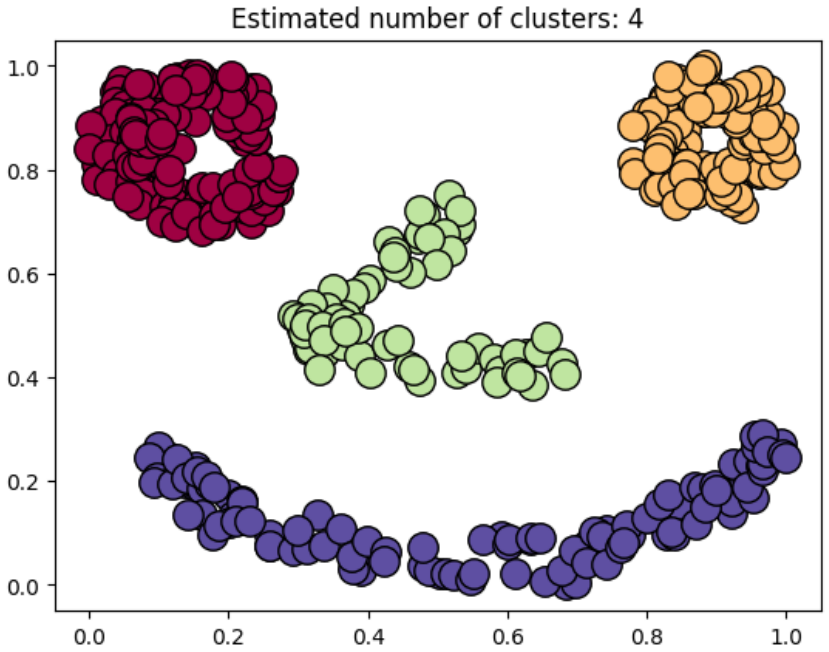


Figure: DBscan

Réduction de dimension et visualisation

Déjà rencontrée et étudiée en ING1, l'**Analyse en composante principale ou ACP** pour projeter dans un nouvelle espace maximisant la variance des données sur chaque nouveau axe.

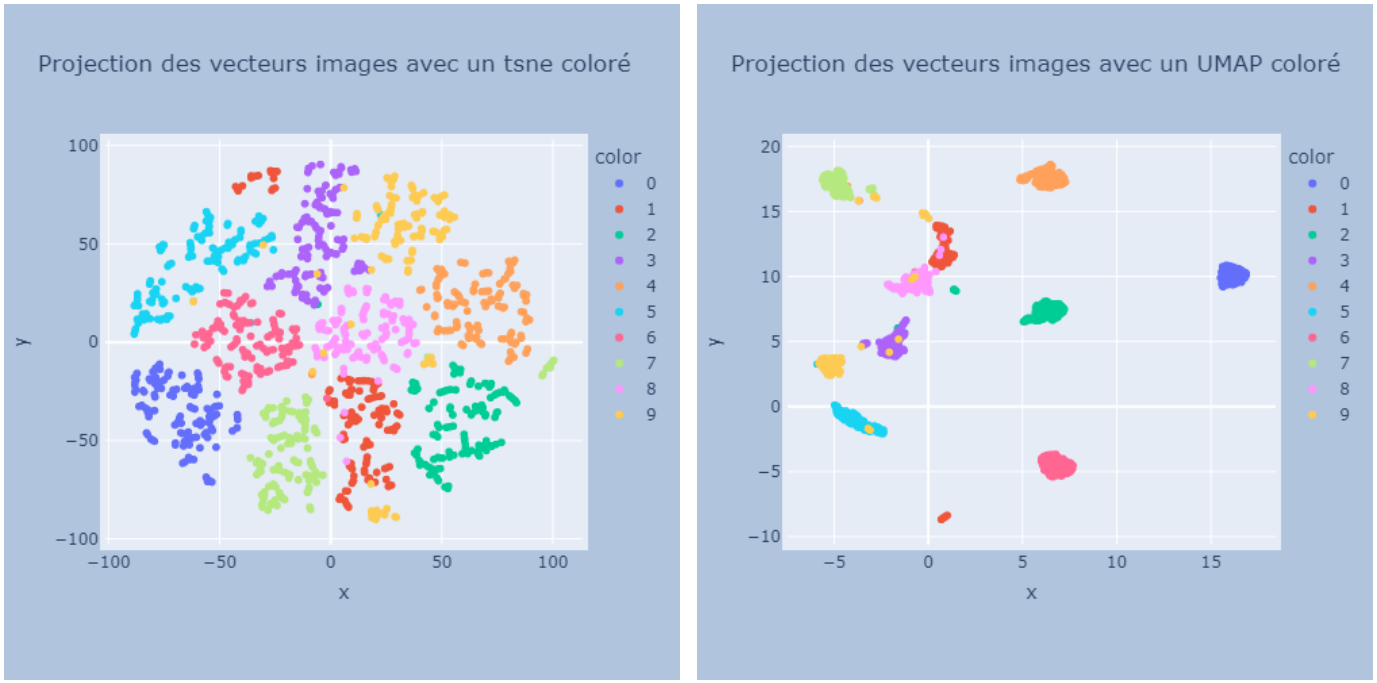


Figure: Projection d'images de chiffres écrits à la main avec les algos t-SNE et UMAP

Arbres de décision et méthodes ensemblistes

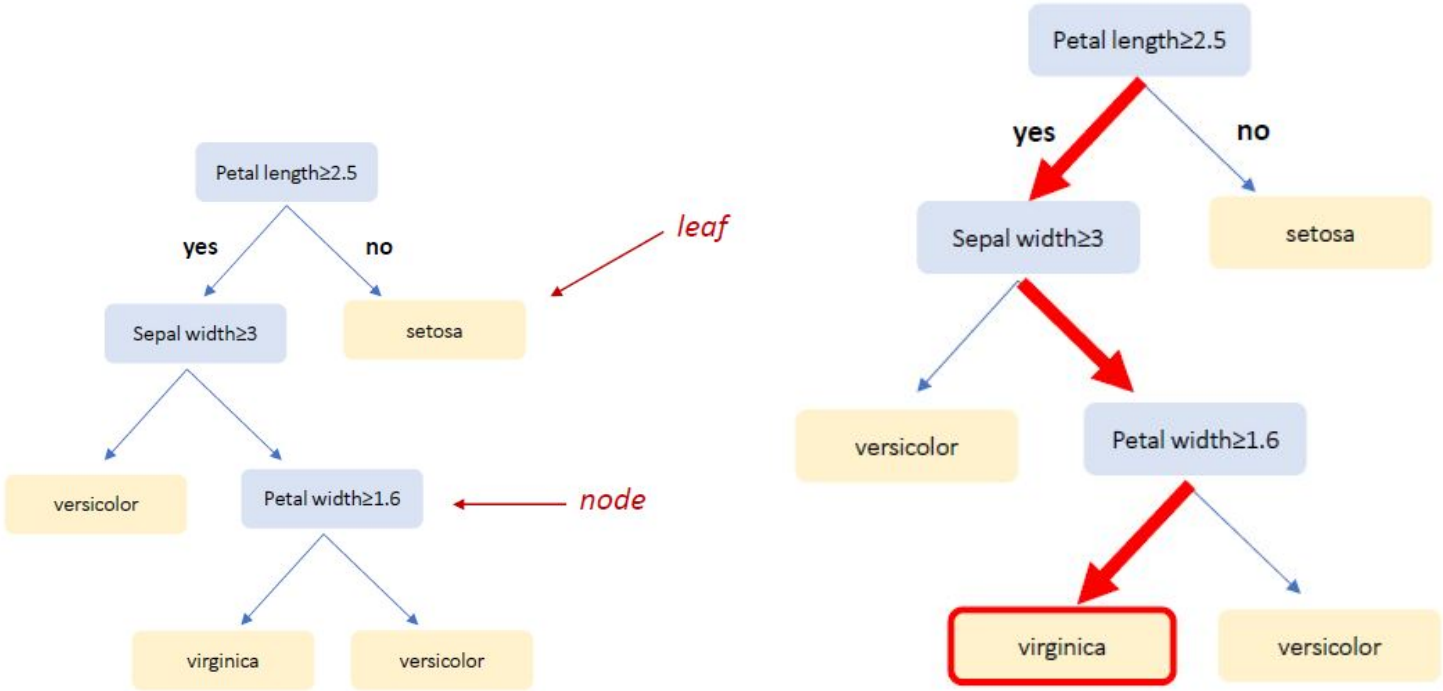


Figure: Decision tree

Attention au surapprentissage ...

Comme le nom l'indique, une forêt aléatoire est une agrégation d'arbres de décision. L'objectif est de rendre la méthode moins sensible au bruit et aux points aberrants de la base d'apprentissage.

L'idée est simple. Il s'agit de construire plusieurs arbres sur des échantillons bootstrap de la base d'apprentissage.

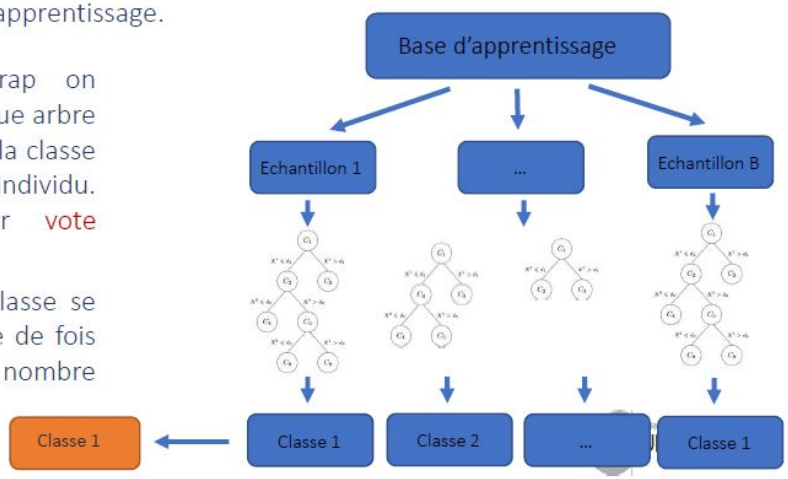
Un échantillon bootstrap est un tirage aléatoire d'éléments de la base d'apprentissage :

- soit de n éléments avec remise
- soit de $k<n$ éléments (avec ou sans remise)

parmi le n observations de la base d'apprentissage.

Pour chaque échantillon bootstrap on construit un arbre de décision. Chaque arbre permet d'obtenir une estimation de la classe de la variable cible pour un nouvel individu. L'estimation finale se fait par vote majoritaire.

La probabilité d'appartenir à une classe se calcule en comptabilisant le nombre de fois où un arbre a prédit la classe sur le nombre d'arbres total.



Algorithme des random forests

E={$(x_1,y_1),\ldots,(x_n,y_n)$} base d'apprentissage avec p variables explicatives
Pour b=1,…,B (boucle sur B arbres)
tirer un échantillon bootstrap E_b
construire un arbre sur l'échantillon E_b tel que :
À chaque nœud de l'arbre choisir le meilleur split
sur un nombre restreint de variables tirées aléatoirement
parmi les p variables
Fin pour b
Pour un nouvel individu défini par x,
estimer sa classe pour chacun des B arbres
choisir la classe majoritaire

Figure: Random Forest

XGBoost

Modèle qui a bouleversé les compétitions Kaggle sur les jeux de données tabulaires ! Repose sur 3 axes (dont certains communs avec les précédents) :

- **Bagging** : entraîner des modèles (arbres) sur des sous-échantillons aléatoires, on retient la classe la plus fréquentes (classification) ou moyenne des résultats (régression),
- **Boosting** :
  1. on initialise toutes les données aux mêmes poids,
  2. on entraine un premier modèle F_0,
  3. augmentation du poids des données où erreur,
  4. entraînement d'un second modèle F_1 sur les nouvelles données,
  5. itération jusqu'à critère d'arrêt (toutes les données utilisées ou nombre max de modèles)
  6. Prédiction finale : prédiction pondéré des anciens modèles

XGBoost

- **Gradient boosting** : hypothèse : la fonction de perte est différentiable
  - Initialisation d'un classifieur faible f_0,
  - on fait une première étape du boosting,
  - on a un nouveau modèle f_1 tel que $f_1(x) = f_0(x) + h_0(x)$ où h est le résidu obtenu par : $r_{i1} = -\frac{\partial L(y_i, f_0(x_i))}{\partial f(x_i)}$
  - On définit le poids de f_0 par $\gamma_0 = \underset{\gamma}{argmin} \sum_{i=1}^{n} L(y_i; f_0(x)) + \gamma h_0(x_i)$
  - le nouveau modèle f_1 est alors donné par $f_1(x) = f_0(x) + \eta\gamma_0 h_0(x)$
  - On itère jusqu'à critère d'arrêt

D'autres modèles ?

Une liste non exhaustive :

1. Classifieur Bayesien (basé sur la formule de Bayes $\mathbb{P}(A|B) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)}$
2. Support vector machine (voir `https://helios2.mi.parisdescartes.fr/~lomn/Cours/DM/Material/ComplementsCours/SVM.pdf`)
3. etc

Un peu de litterature : `https://arxiv.org/abs/2106.03253` "Tabular Data: Deep Learning is Not All You Need"
On va dans la suite comparer les modèles que l'on vient de voir et un réseau de neuronnes classique.