

Documentation Machine learning

La science des données

La science des données est un domaine interdisciplinaire combinant les mathématiques statistiques, l'informatique, et la compréhension métier. Elle permet d'extraire des connaissances et insights à partir de données brutes.

🔍 Elle englobe : le nettoyage de données, l'analyse exploratoire, la modélisation et la visualisation.

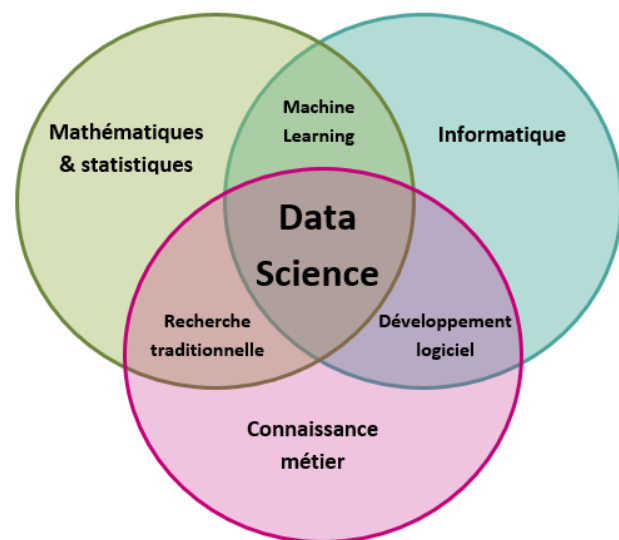
📖 Sources :

- Wikipedia

https://fr.wikipedia.org/wiki/Science_des_donn%C3%A9es

- LeDataScientist

<https://ledatascientist.com/quest-ce-que-la-data-science>

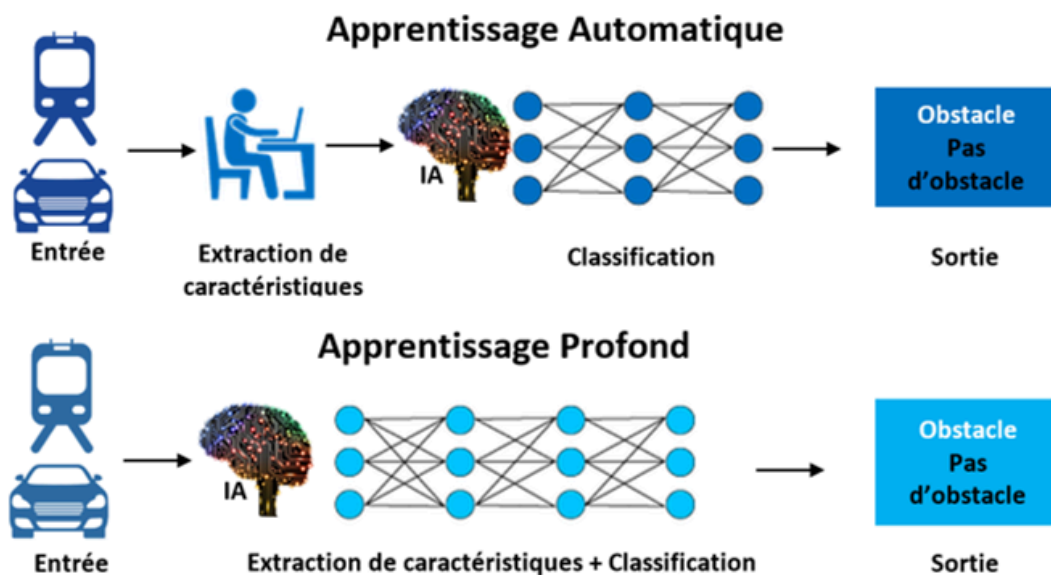


L'apprentissage automatique et/ou l'apprentissage profond

L'apprentissage automatique, Machine Learning, est une technique où les ordinateurs apprennent à partir de données, sans être explicitement

programmés pour chaque tâche mais en définissant manuellement les caractéristiques.

L'apprentissage profond, Deep Learning, est une technique utilisant des réseaux de neurones profonds pour apprendre des représentations complexes. Les caractéristiques sont trouvées automatiquement.



📖 Sources :

- Wikipedia https://fr.wikipedia.org/wiki/Apprentissage_automatique
- Wikipedia https://fr.wikipedia.org/wiki/Apprentissage_profond

L'apprentissage supervisé

Dans l'apprentissage supervisé, notre objectif est d'apprendre une fonction qui associe une entrée à une sortie sur la base d'exemples de paires entrée-sortie, un peu comme l'apprentissage avec un enseignant. Le 'superviseur' ou 'enseignant' ici fait référence aux données étiquetées que nous donnons à l'algorithme.

Imaginons un enfant qui regarde un livre d'images avec un adulte. Quand ils tombent sur une image d'un chat, l'adulte dit : "C'est un chat". L'enfant apprend alors de cette information étiquetée que la créature particulière avec des moustaches, des oreilles pointues et une queue touffue est un chat. De même, un modèle d'apprentissage supervisé apprend à associer une image à l'étiquette "chat". Au fil du temps, le modèle devient meilleur pour reconnaître les chats dans les images, tout comme l'enfant.

Le premier type, "Apprentissage Supervisé", est représenté par une image d'un chat de dessin animé, avec la légende "C'est un chat". Il illustre comment l'apprentissage supervisé classe les données en fonction des étiquettes fournies.

🎯 Objectif : Prédire une sortie (Y) à partir d'entrées (X).

📖 Sources :

-

Apprentissage Supervisé

Données : (x, y)

x est une donnée d'entrée, y est une étiquette (par exemple, une photo avec l'étiquette "chat")

Objectif: Apprendre à associer l'entrée à la sortie, c'est-à-dire $x \rightarrow y$

Un exemple: pour classifier Ceci est un chat



Un chat

L'apprentissage non supervisé

L'apprentissage non supervisé, en revanche, traite de données non étiquetées, c'est-à-dire qu'il n'y a pas de guide et que le modèle doit trouver des modèles et des relations dans les données par lui-même.

C'est comme si l'enfant regardait le livre d'images seul et essayait de comprendre les connexions ou les similitudes entre diverses images.

Imaginons que l'enfant tombe sur deux images, l'une d'un chat et l'autre d'un lion. Bien que personne ne soit là pour le leur dire, l'enfant remarque que les deux ont des caractéristiques similaires comme des moustaches, des oreilles pointues et une queue. Ils ne connaissent peut-être pas encore les mots 'chat' et 'lion', mais ils comprennent que "Ces deux choses se ressemblent". De même, les algorithmes d'apprentissage non supervisé essaient de regrouper les choses similaires.

Le deuxième type, "Apprentissage Non Supervisé", est représenté par deux images de dessins animés qui ressemblent à un chat domestique et à un chat sauvage. La légende "Ces deux choses se ressemblent", exprime le but de l'apprentissage non supervisé de comprendre la structure

Apprentissage Non Supervisé

Données: x ,
 x est une donnée, il n'y a pas d'étiquettes !

Objectif: Apprendre la structure sous-jacente des données.

Un exemple: Comparaison, regroupement



Ces deux choses se ressemblent

sous-jacente des données et d'identifier les similitudes ou les différences.

 Sources :

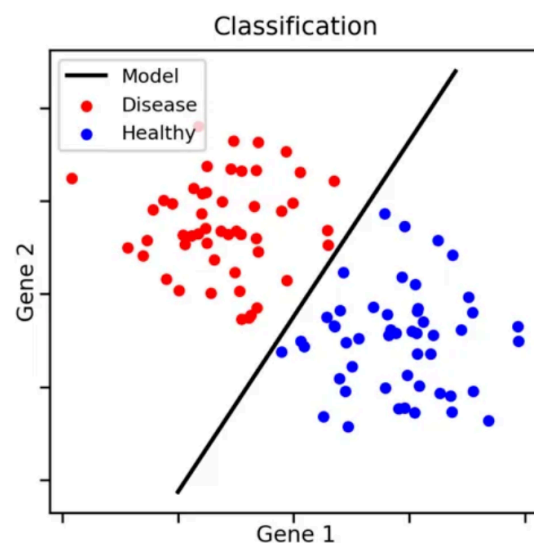
- <https://fr.linkedin.com/pulse/décodage-de-lapprentissage-automatique-apprentissage-supervisé>

La classification supervisée

La classification supervisée est une technique qui classifie, attribue une étiquette, à des données en se basant sur un ensemble d'apprentissage préalablement étiqueté.

Exemple :

- En finance et dans le secteur bancaire pour la détection de la fraude par carte de crédit (fraude, pas fraude).
- Détection de courrier électronique indésirable (spam, pas spam).
- Dans le domaine du marketing utilisé pour l'analyse du sentiment de texte (heureux, pas heureux).
- En médecine, pour prédire si un patient a une maladie particulière ou non.



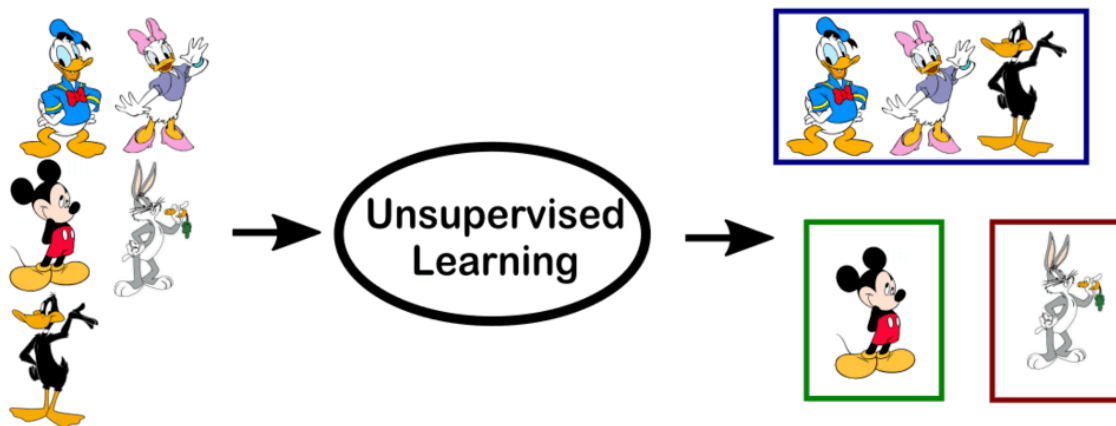
 Sources :

- <https://brightcape.co/apprentissage-supervise-vs-non-supervise/>

La classification non supervisée

La classification supervisée est la méthode de classification sans utilisation d'étiquettes.

Exemple :



 Sources :

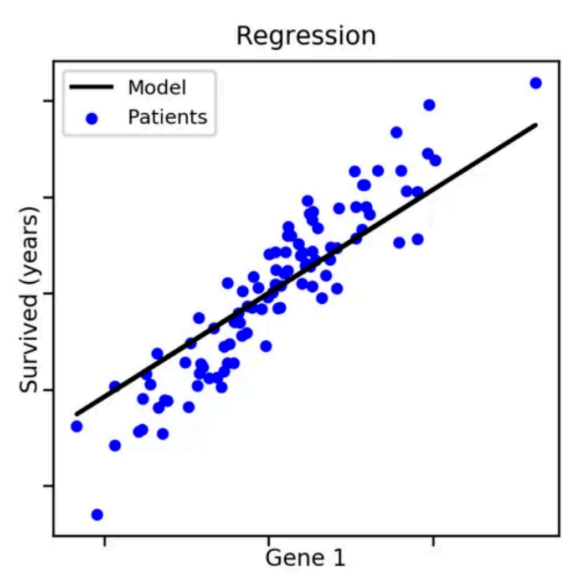
<https://brightcape.co/apprentissage-supervise-vs-non-supervise/>

La régression


En statistiques, la régression est la méthode d'analyse visant à rapprocher des variables corrélées à une ligne de régression.

En science des données, la régression est une problématique d'apprentissage automatique différente de la classification. En effet, la régression est dans une optique de prédiction d'une variable quantitative, là où la classification est d'une variable qualitative (d'un plan sémantique).

Exemple :



- Prédire le prix de l'immobilier
- Prédire le cours de bourse

 Sources :

[-https://brightcape.co/apprentissage-supervise-vs-non-supervise/](https://brightcape.co/apprentissage-supervise-vs-non-supervise/)

La validation croisée

La validation croisée est une technique essentielle en machine learning pour évaluer la performance d'un modèle de manière rigoureuse. Elle est particulièrement utile lorsque les données sont limitées ou lorsqu'on souhaite éviter que la performance mesurée soit biaisée par une unique séparation aléatoire des données.

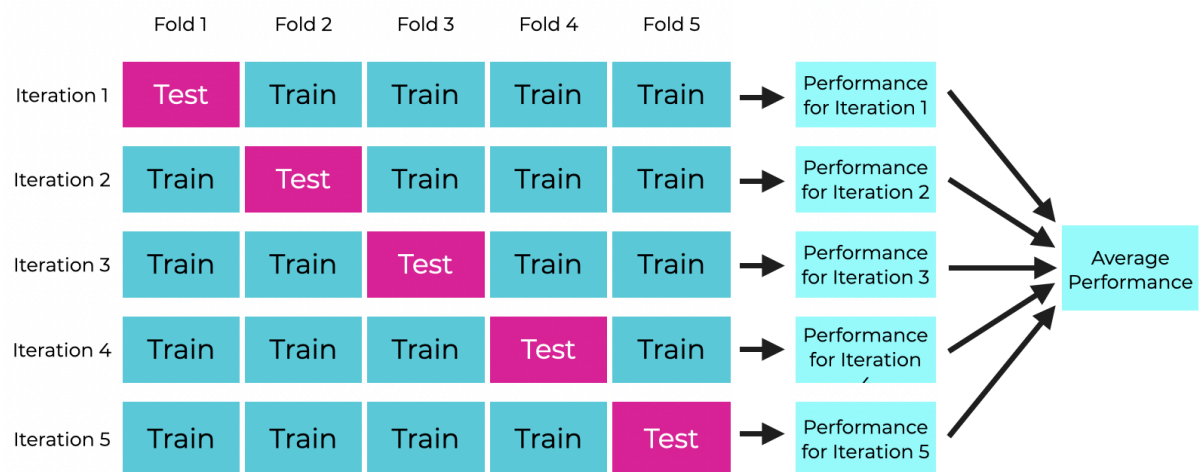
Dans sa version la plus utilisée, appelée **validation croisée k-fold**, l'ensemble des données est divisé en k sous-ensembles de taille égale appelés *folds*. Le modèle est alors entraîné k fois, à chaque fois sur $k-1$ folds, en réservant le fold restant pour tester le modèle. À chaque itération, une mesure de performance (comme la précision, l'erreur quadratique moyenne, etc.) est calculée. Une fois les k itérations terminées, on fait la **moyenne des scores obtenus**, ce qui fournit une évaluation plus stable et plus représentative des performances du modèle sur des données inconnues.

Cette méthode présente un **grand avantage** : elle permet d'utiliser **chaque observation à la fois pour l'entraînement et pour le test**, ce qui réduit la variance liée à une unique division des données. C'est pourquoi la validation croisée est considérée comme une méthode fiable pour comparer plusieurs modèles ou ajuster leurs paramètres, notamment lorsque les données sont rares.

En revanche, elle peut être **plus coûteuse en calcul**, puisqu'elle nécessite d'entraîner le modèle k fois. Cela peut poser problème pour des algorithmes très complexes ou sur des datasets volumineux. Néanmoins, cet investissement en temps de calcul est souvent justifié par la **qualité des estimations** qu'elle permet.

En pratique, un choix courant est **$k = 5$ ou 10** , qui offre un bon compromis entre précision de l'évaluation et coût computationnel.

CROSS VALIDATION, EXPLAINED



📖 Sources :

[-https://help.glik.com/fr-FR/cloud-services/Subsystems/Hub/Content/Sense_Hub/AutoML/holdout-crossvalidation.htm](https://help.glik.com/fr-FR/cloud-services/Subsystems/Hub/Content/Sense_Hub/AutoML/holdout-crossvalidation.htm)

https://scikit-learn.org/stable/modules/cross_validation.html

<https://www.sharpsightlabs.com/blog/cross-validation-explained/>

Les données d'entraînement, les données test et/ou de validation

Dans l'entraînement d'un modèle d'apprentissage automatique, il est courant de séparer le jeu de données en trois parties.



Les données d'entraînement sont utilisées pour entraîner le modèle, c'est-à-dire pour ajuster les poids et les paramètres internes.

Les données de validation permettent d'ajuster les *hyperparamètres* du modèle et de prévenir le surapprentissage (overfitting). Elles sont utilisées régulièrement durant l'entraînement, sans être directement exploitées pour la mise à jour du modèle.

Les données de test servent à évaluer la performance finale du modèle sur des données encore jamais vues, afin d'estimer sa capacité à généraliser.

Qu'est-ce qu'un hyperparamètre ?

Un hyperparamètre est un paramètre fixé à l'avance avant l'entraînement du modèle. Contrairement aux paramètres internes (comme les poids dans un réseau de neurones) qui sont appris automatiquement à partir des données, les hyperparamètres sont des choix faits par l'utilisateur.

Voici quelques exemples d'hyperparamètres :

Hyperparamètre	Description
Taux d'apprentissage (<i>learning rate</i>)	Détermine la vitesse à laquelle le modèle ajuste ses paramètres.
Nombre d'arbres (Random Forest)	Contrôle la complexité du modèle en forêt aléatoire.
Nombre d'époques (epochs)	Nombre de fois où le modèle parcourt l'ensemble des données d'entraînement.
Taille du batch (<i>batch size</i>)	Nombre d'exemples vus en même temps avant de mettre à jour les paramètres.
Profondeur maximale d'un arbre (Decision Tree)	Limite la complexité de l'arbre pour éviter le surapprentissage.

Ces hyperparamètres doivent être réglés avec soin, souvent en testant plusieurs combinaisons grâce aux données de validation.

Qu'est-ce que le surapprentissage (overfitting) ?

Le surapprentissage se produit lorsqu'un modèle apprend trop bien les détails et les particularités du jeu de données d'entraînement, y compris le bruit ou les anomalies, au lieu de saisir les tendances générales.

Exemple :

Imagine un élève qui mémorise par cœur les réponses d'un examen blanc, sans comprendre les concepts. Il réussira ce test blanc, mais échouera à l'examen final si les questions changent un peu.

📖 Sources :

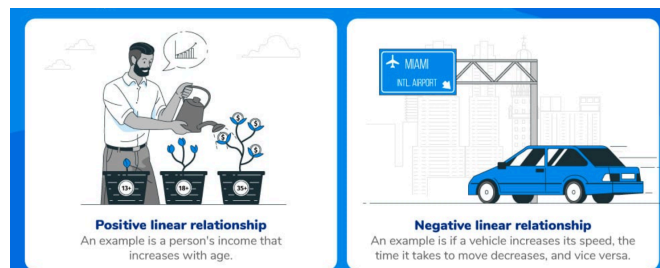
- https://fr.wikipedia.org/wiki/Jeux_d'entrainement,_de_validation_et_de_test

- <https://scikit-learn.org/stable/glossary.html#term-hyperparameter>

Corrélation linéaire de Pearson entre deux variables

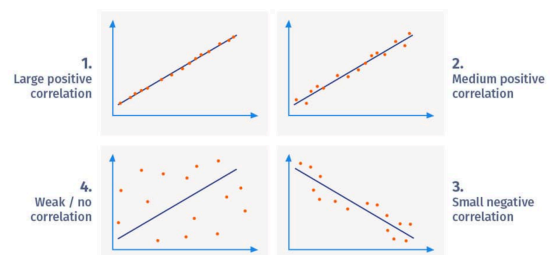
La corrélation de Pearson est un indicateur statistique qui mesure l'intensité et le sens de la relation linéaire entre deux variables numériques. Ce coefficient, noté r , varie entre -1 et 1 :

Valeur de r	Interprétation
$r \approx +1$	Forte corrélation positive : les deux variables augmentent ensemble.
$r \approx -1$	Forte corrélation négative : l'une augmente quand l'autre diminue.
$r \approx 0$	Aucune corrélation linéaire détectée.



Les corrélations peuvent être visualisées à l'aide de **nuages de points (scatter plots)** :

Type de corrélation	Illustration
Corrélation positive forte	Les points suivent une ligne ascendante.
Corrélation négative forte	Les points suivent une ligne descendante.
Corrélation faible ou nulle	Les points sont dispersés sans structure.



Les visualisations sont souvent accompagnées d'une droite de régression, qui permet d'illustrer la tendance centrale de la relation.

Attention : corrélation \neq causalité

Même si deux variables sont corrélées, cela ne signifie pas que l'une cause l'autre. Il peut y avoir des variables cachées ou des coïncidences.

 Sources :

-<https://www.questionpro.com/blog/fr/coefficient-de-correlation-de-pearson/>

Une fonction de coût

La fonction de coût (aussi appelée *cost function* ou *loss function* en anglais) est un élément central de l'entraînement d'un modèle de machine learning. Elle permet de quantifier l'erreur entre les prédictions du modèle et les valeurs réelles attendues.

Autrement dit, la fonction de coût indique à quel point le modèle se trompe : plus sa valeur est élevée, plus le modèle est loin de la vérité ; plus elle est faible, plus le modèle est précis.

Fonctionnement

1. Le modèle effectue une prédiction à partir d'une entrée.
2. On compare cette prédiction à la valeur réelle.
3. On calcule l'écart entre les deux à l'aide de la fonction de coût.
4. Ce score est ensuite utilisé pour ajuster les paramètres du modèle (par exemple via la descente de gradient).

Ce processus est répété sur l'ensemble des données d'entraînement, et l'objectif est de minimiser cette fonction pour que le modèle s'améliore.

La fonction de coût est **le guide du modèle** : elle lui indique dans quelle mesure il fait des erreurs, et elle est essentielle pour que les algorithmes d'optimisation comme la descente de gradient puissent ajuster les paramètres du modèle dans le bon sens.

 Sources :

-<https://datascientest.com/fonction-de-cout-tout-savoir>

-https://rtavenar.github.io/deep_book/fr/content/fr/loss.html

La descente de gradient

La descente de gradient est un algorithme d'optimisation utilisé pour entraîner les modèles de machine learning. Son objectif est de minimiser la fonction de coût, c'est-à-dire de réduire l'erreur entre les prédictions du modèle et les valeurs réelles.

Fonctionnement

1. Le modèle commence avec des paramètres initiaux (souvent choisis aléatoirement).
2. Il effectue des prédictions sur les données d'entraînement.
3. Il calcule l'erreur entre les prédictions et les vraies valeurs via une fonction de coût.
4. Il calcule le gradient de cette fonction : cela indique la direction dans laquelle il faut ajuster les paramètres pour réduire l'erreur.
5. Il met à jour les paramètres en suivant cette direction, avec une certaine amplitude (le taux d'apprentissage).
6. Ce processus est répété à chaque itération jusqu'à atteindre un minimum (idéalement global) de la fonction de coût.

Le taux d'apprentissage (learning rate)

Le taux d'apprentissage détermine la vitesse de mise à jour des paramètres à chaque itération :

- Un taux trop élevé peut faire osciller ou diverger le modèle.
- Un taux trop faible entraîne une convergence très lente, voire un blocage avant d'atteindre le minimum.

Le choix du taux d'apprentissage est donc crucial pour la convergence rapide et stable de l'algorithme.



Métaphore technique

La descente de gradient peut être comparée à un algorithme qui cherche à atteindre le point le plus bas d'une surface (représentant l'erreur) en se déplaçant progressivement dans la direction où la pente est la plus raide.



Sources :

[-https://www.ibm.com/fr-fr/think/topics/gradient-descent](https://www.ibm.com/fr-fr/think/topics/gradient-descent)