LES GRANDES FAMILLES DE MACHINE LEARNING ET DEEP LEARNING

Dr . KEFFA

22 10 25 Octobre 2025

Table des matières

1. Objectifs

~ Présenter les concepts clés des principaux types de modèles ;
~ Comprendre les mécanismes qui sous-tendent les principaux types de modèles ;
~ Décrire les logiques d'apprentissages rattachées à chacun des modèles ;
~ Analyser les différents types de problèmes propres à chacune de ces approches.

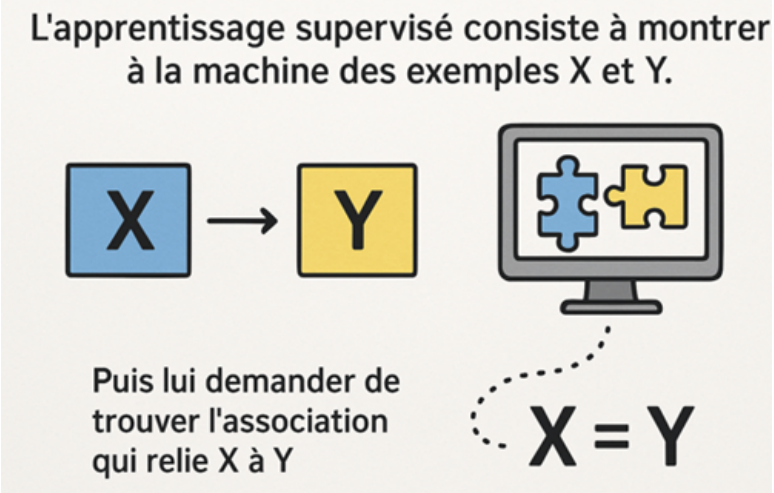
2. Introduction

Dans ce chapitre, nous abordons les **principes fondamentaux des grandes familles du Machine Learning**. L'objectif est de comprendre les **mécanismes et les logiques d'apprentissage** qui sous-tendent les principaux types de modèles : l'**apprentissage supervisé**, l'**apprentissage non supervisé**, l'**apprentissage par renforcement** et le **Deep Learning** (ou apprentissage profond). Nous mettrons en évidence les **concepts clés** et les **principes de fonctionnement** qui permettent de formaliser et de résoudre les différents types de problèmes propres à chacune de ces approches.

3. Apprentissage supervisé

3.1. 1. Définition et principe de fonctionnement

L'apprentissage supervisé est une stratégie d'apprentissage automatique qui consiste à montrer à la machine des exemples X et Y. Puis lui demander de trouver l'association qui relie X à Y.



En effet, cette association s'appelle une fonction f(X)=Y. L'ensemble constitué des X et des Y, se nomme dataset (ou **jeu de données**, échantillons ou observations en français).

3.1.1. a. La dataset :

Un **dataset** (ou **jeu de données**) est un **ensemble structuré de données** utilisé pour entraîner, valider et tester les modèles de Machine Learning.

Chaque observation (ou échantillon) d'un dataset est constitué de deux types de variables. D'un côté, on a les **targets**, labels, ou outcomes (variable à expliquées, étiquettes) et les **features** (variables explicatives, descripteurs, attributs, prédicteurs, ou caractéristiques).

· **une ou plusieurs cibles (target)** ou **labels**: La variable à expliquer ou la valeur attendue Y qui représente l'objectif, surtout dans notre contexte l'apprentissage supervisé. Il représente se qu'on veut que la machine prédire (le prix d'un appartement, la destination d'un patient deserteur…)

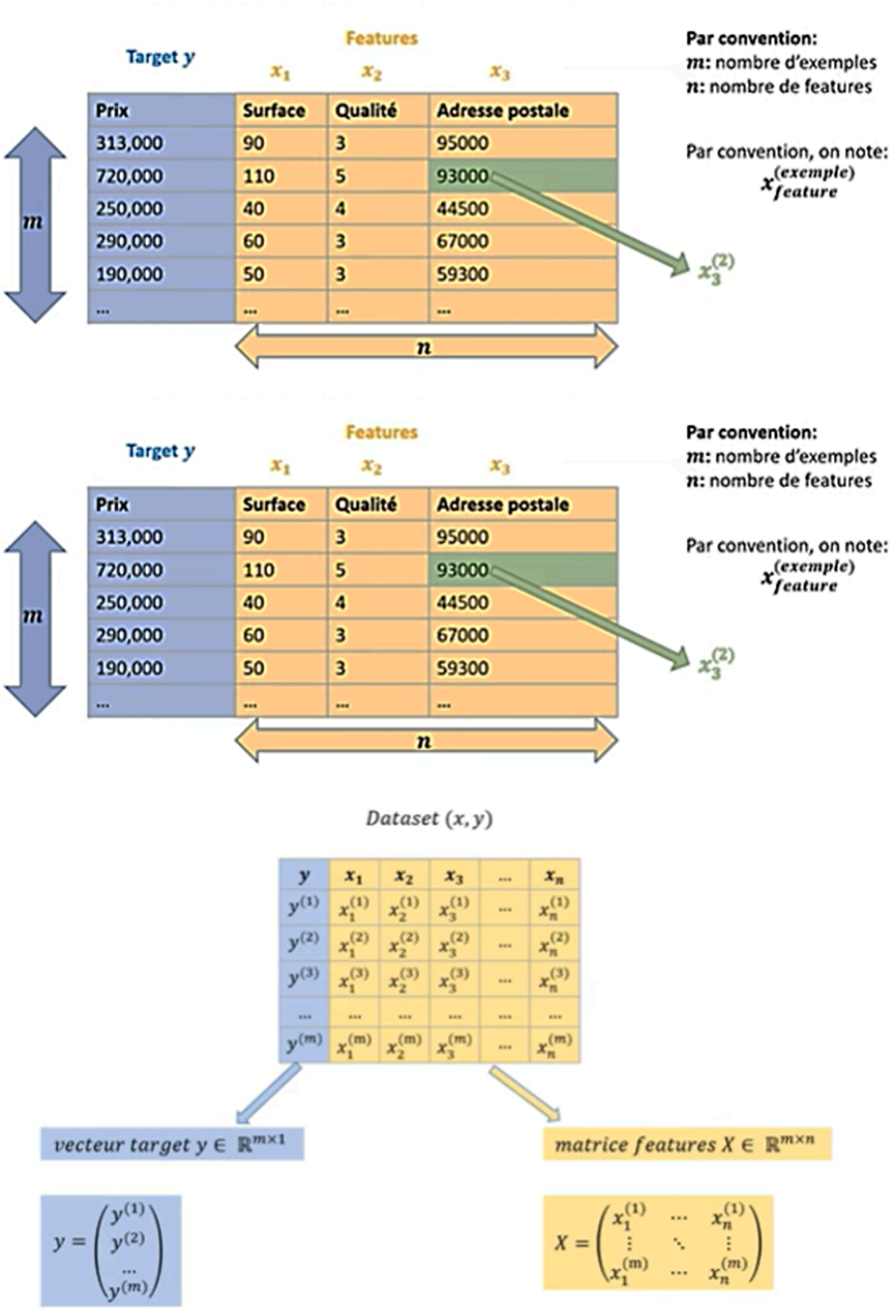
· **Les caractéristiques (features) ou variables explicatives** : Les variables d'entrée X qui servent à expliquer ou décrire la valeur de Y.

En somme, d'un côté, nous avons les features qui viennent d'influencer la valeur de Y tandis que de l'autre côté, nous avons les Y qui est une fonction des $\mathbf{X_i}$. D'où : $\boldsymbol{Y = f(X_1,X_2,\ldots,X_n)}$.

Exemple de Dataset sur des appartements

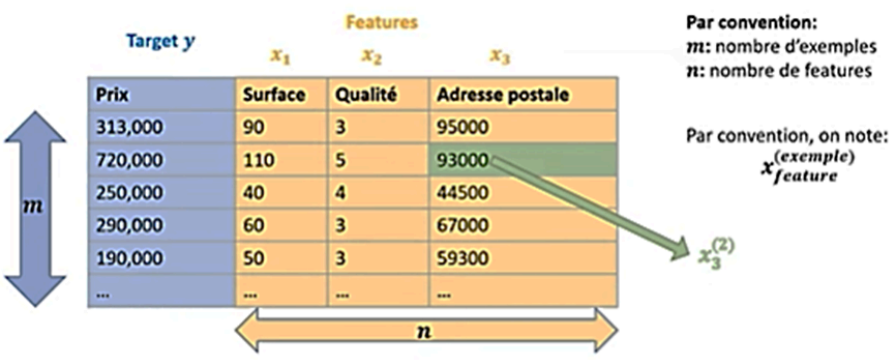
Prix	Surface	Qualité	Adresse postale
313,000	90	3	95000
720,000	110	5	93000
250,000	40	4	44500
290,000	60	3	67000

a) a. Présentation d'un dataset :

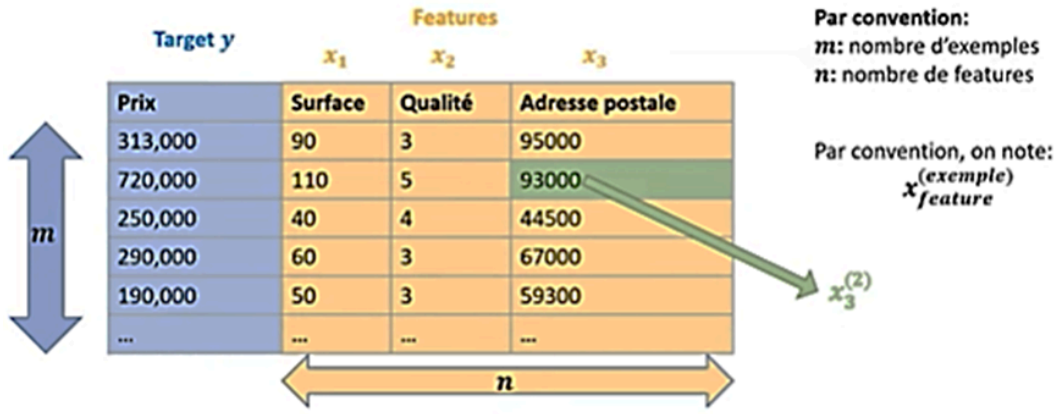


Par convention, on note **m** le nombre d'exemples, de lignes ou d'observations qu'on a dans notre dateset et **n** le nombre de features ou de colonnes qu'on a dans notre dataset. De meme, on désigne une cellule en particulier la cellule dans notre dataset , on note le numéro de l'exemple au-dessus du **x** et le numéro de la feature en indice. On obtient à titre d'exemple, se référer à la figure ci-dessous.

Exemple de Dataset sur des appartements

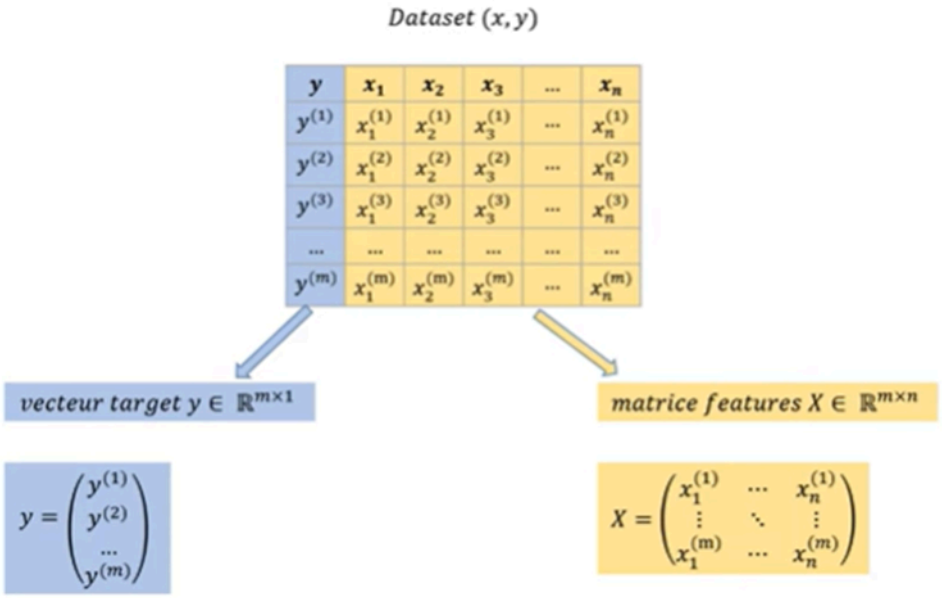


Pour finir, on obtient les dataset qui ressemblent à la figure 2 suivante



· Une fois la dataset obtenu, il est conseillé de la transformer en vecteurs. Ainsi, on aura un **vecteur target qui** sera un vecteur avec m lignes c'est-à-dire m éléments ;

· Une matrice features à m lignes et n colonnes qui sera une matrice mXn. Cette situation est illustrée par la figure 3 ci-dessous.



Les problèmes dus à l'étiquetage des données (données bruitées):

Les principaux défis rattachés à l'étiquetage des données bruitées sont les suivantes :

~ Erreur de de mesure : mauvaise saisies des données ;

~ Erreur d'étiquetage : Mauvaise correspondance des données durant l'étiquetage ;

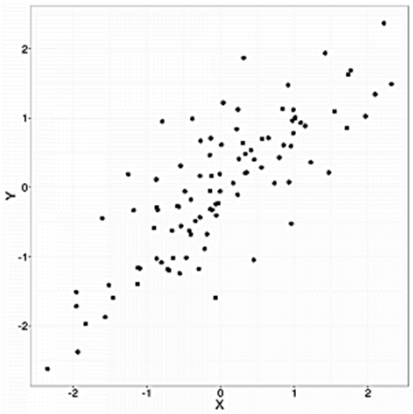
~ Erreur de choix : Les variables choisies ne correspondent pas au problème de modélisation ;

3.2. Notion de modèle

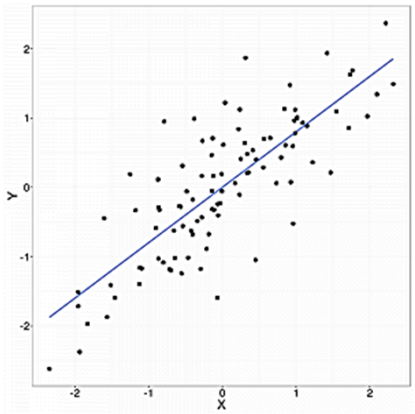
3.2.1. Définition

À partir d'une **dataset**, il est souvent possible de représenter les données sous forme d'un **nuage de points**. Ce nuage traduit visuellement la relation potentielle entre les variables d'entrée et de sortie. En observant cette dispersion de points, on cherche à identifier une **tendance générale** : par exemple, une droite ou une courbe qui résume la relation entre les données. (***voir la figure suivante***).C'est à ce stade qu'intervient la notion de **modèle** en Machine Learning. Le modèle

représente une **fonction mathématique** qui tente de décrire le comportement observé dans les données. Ainsi, si les points semblent alignés, on peut proposer un **modèle linéaire** de la forme f(x)=ax+b sont les **paramètres** du modèle.



Nuage de points du jeu de données.



Superposition d'une droite de régression (bleu).

a) Paramètres

Les paramètres du déterminent la pente et l'ordonnée à l'origine de la droite, et leur ajustement permet au modèle de mieux s'adapter aux données. Toutefois, si le nuage de points suit une courbe, on peut choisir un **modèle polynomial** comme **$f(x)=ax^2+bx+c$**.Là encore, les **paramètres** définissent la forme exacte de la courbe et sont ajustés lors de l'apprentissage pour minimiser l'écart entre les prédictions du modèle et les valeurs réelles.

Il est important de préciser que c'est nous qui décider quelle la machine doit utiliser. Dans, le cas d'espèce il revient à la machine d'apprendre des paramètres les paramètres de ce modèle.

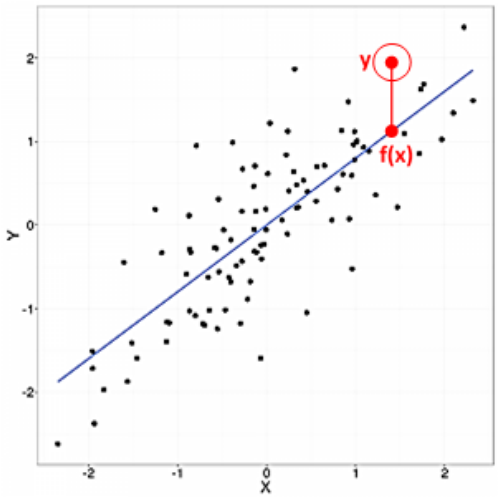
3.3. La fonction coût en Machine Learning

3.3.1. Définition

$$J(\theta) = \frac{1}{N}\sum_{i=1}^{N} \ell(y^{(i)}, \hat{y}^{(i)}(\theta))$$

$$f(x) = ax + b$$

$$J(a,b) = \frac{1}{2m}\sum_{i=1}^{m}\left(f(x^{(i)}) - y^{(i)}\right)^2$$



Superposition d'une droite de régression (bleu).

La **fonction coût** (ou **fonction de perte**, en anglais *cost function* ou *loss function*) est un concept **fondamental** du Machine Learning. Elle est une mesure mathématique qui quantifie à quel point les prédictions réalisées par un modèle sont éloignées des valeurs. On parle de l'**erreur** commise par un modèle lorsqu'il fait des prédictions sur les données.

Autrement dit, elle indique **dans quelle mesure les prédictions du modèle s'écartent des valeurs réelles**. Par ailleurs, elle représente l'outil qui guide l'apprentissage en indiquant au modèle dans quelle direction s'améliorer.

En bref, cette fonction coût mesure l'erreur moyenne entre :

~ Les **prédictions du modèle** (y)

~ et les valeurs réelles (y)

Exemple pour notre modèle de régression linéaire :

On a :

Ainsi, la fonction coût est :

En effet, lorsqu'on obverse notre modèle de régression linéaire sous ci-dessous, on constate clairement que la droite représentative de notre modèle ne passe pas par tous les points de notre nuage de points car soit il prédire des valeurs bien supérieures ou en dessous des valeurs réelles. On dit qu'il y a une erreur par rapport à cette donnée dans le dataset qui est ***y*** et notre prédiction qui est ***f(x)***. On peut trouver ces différentes erreurs sur tous les points de notre dataset. Quand on assemble tous ces erreurs, cela, nous donne la fonction coût.

Remarque : Il existe de nombreuses fonctions de coût. Le choix d'une fonction de coût dépend d'une part du problème en lui-même, autrement dit de ce que l'on trouve pertinent pour le cas pratique considéré, et d'autre part de considérations pratiques.

a) Rôles de la fonction coût

· **Mesurer l'erreur entre la prédiction et la réalité**

Le rôle principal de la fonction coût est de **guider** à **faire de meilleures prédictions**.

Sans fonction coût, un modèle ne saurait pas **ce qui est bon ou mauvais**, ni **dans quelle direction** modifier ses paramètres. Ainsi, il est à mesurer l'écart entre :

~ la **sortie prédite** par le modèle y ;

~ et la **valeur réelle** attendue y.

Exemple applicatif :

Considérons notre dataset sur les appartements. Si l'on prédit le prix d'un appartenant par rapport à sa surface :

~ Vrai *y* = 720.000

~ Prédit y = 700.000

~ Erreur = 20

La fonction coût va transformer cet écart en une valeur numérique. A titre d'exemple :

J = (y-*y*)^2 = 25) qui indique la qualité de la prédiction. IL ressort que plus cette valeur est grande, plus le modèle **s'éloigne de la vérité** ; plus est petite, plus le modèle **est précis**.

· **Guider l'apprentissage du modèle**

Le deuxième rôle fondamental de la fonction coût est de **guider l'optimisation** du modèle **le modèle** tout en minimiser ses erreurs.

En effet, l'apprentissage consiste à **trouver les paramètres** (poids et biais) qui **minimisent** cette fonction. Cela passe par la recherche des paramètres du modèle noté Θ^* tels la fonction coût $J(\Theta^*)$ avec soit minimal. En machine Learning, on va développer une stratégie qui cherche à trouver quels sont les paramètres a, b, c…qui minimisent la fonction coût, c'est-à-dire qui minimise l'ensemble de nos erreurs dans une situation de prédiction. Il convient de noter qu'il existe beaucoup d'algorithmes qui minimisent notre fonction d'apprentissage. L'algorithme le plus représentatif est la **descente de gradient.** Celui-ci utilise la dérivée de la fonction coût pour savoir :

~ dans quelle direction ajuster les paramètres ;

~ et de quelle **ampleur**.

Par conséquent, la fonction coût agit comme une **boussole mathématique**

:

elle indique au modèle **comment corriger ses erreurs** à chaque itération. En somme, elle se prend son essence par le fait qu'à l'issue de de la construction de notre modèles, les nouvelles données étiquetés obtenues soient très proches des premières données collectées (avant sa construction).

3.4. Les algorithmes de l'apprentissage supervisé :

3.4.1. La classification binaire :

Exemple

Voici quelques exemples de problèmes de classification binaire :

- Identifier si un patient déserteur est allé sur un rebouteur ou dans un autre centre de santé ;
- Identifier si un email est un spam ou non ;
- Identifier si un tableau a été peint par Picasso ou non;
- Identifier si une image contient ou non une girafe;
- Identifier si une molécule peut ou non traiter la dépression;
- Identifier si une transaction financière est frauduleuse ou non.

Exemple

Voici quelques exemples de problèmes de classification multi-classe :

- Identifier si un patient déserteur est allé sur un rebouteur, en clinique, hors du pays, à domicile ou dans un autre centre de santé public.
- Identifier en quelle langue un texte est écrit;
- Identifier lequel des 10 chiffres arabes est un chiffre manuscrit ;
- Identifier l'expression d'un visage parmi une liste prédéfinie de possibilités (colère, tristesse, joie, etc.) ;
- Identifier à quelle espèce appartient une plante;
- Identifier les objets présents sur une photographie.

Dans le cas où les étiquettes sont **binaires**, elles indiquent l'appartenance à une classe. On parle alors de classification binaire.

· **Définition :**

Un problème d'apprentissage supervisé dans lequel l'espace des étiquettes est binaire, autrement dit Y = {0,1} est appelé un problème de ***classification binaire***.

Dans le cas où les étiquettes sont **discrètes**, et correspondent donc à plusieurs classes, on parle de classification multi-classe.

Définition :

Un problème d'apprentissage supervisé dans lequel l'es pace des étiquettes est discret et fini, autrement dit Y = {1,2,…,C} est appelé un problème de classification multi-classe. ***C*** représente le nombre de classes.

a) Régression :

Exemple

Voici quelques exemples de problèmes de régression :

- Prédire le nombre de clics sur un lien;
- Prédire le nombre d'utilisateurs et utilisatrices d'un service en ligne à un moment donné;
- Prédire le prix d'une action en bourse;
- Prédire l'affinité de liaison entre deux molécules;
- Prédire le rendement d'un plant de maïs

· **Définition :**

Un problème d'apprentissage supervisé dans lequel l'espace des étiquettes est **Y =IR** est appelé un problème de régression.

Régression structurée :

Dans le cas où l'espace des étiquettes est un espace structuré plus complexe que ceux évoqués précédemment, on parle de régression structurée en anglais, structured regression, ou structured out put prediction. Il peut par exemple s'agir de prédire des vecteurs, des images, des graphes, ou des séquences. La régression structurée permet de formaliser de nombreux problèmes, comme ceux de la traduction automatique ou de la reconnaissance vocale. Ce cas dépasse cependant le cadre du présent ouvrage, et nous nous concentrerons sur les problèmes de classification binaire et multi-classe, ainsi que de régression classique.

L'apprentissage supervisé face à Généralisation et sur-apprentissage

Exemple

Imagine un étudiant qui **mémorise** par cœur les réponses d'un vieux devoir plutôt que de comprendre les concepts. Le jour de l'examen, si les questions sont légèrement différentes, il échoue. C'est exactement ça le **sur-apprentissage** :

Exemple

Le **sous-apprentissage**, c'est quand un modèle est **un mauvais élève qui n'a pas assez écouté** : il n'a pas appris la leçon (les données d'entraînement), donc il échoue aussi bien sur les questions qu'il a déjà vues que sur les nouvelles.

Les questions de généralisation et sur-apprentissage demeurent les deux questions essentielles pour les algorithmes d'apprentissage supervisés.

¨ **Généralisation :**

On appelle **généralisation** la capacité d'un modèle à faire des prédictions correctes sur de nouvelles données, qui n'ont pas été utilisées pour le construire.

¨ **Sur-apprentissage (overfitting)**

On dit d'un modèle qu'il sur apprend lorsqu'il qu'il n'est pas capable de se généraliser c'est-à-dire le modèle a « **mémorisé** » le bruit et les détails spécifiques aux données d'entraînement au lieu d'apprendre la structure générale qui permet de bien généraliser à de nouvelles données.

Autrement dit, est un modèle qui sur apprend est un modèle est **trop bon élève** : il apprend par cœur sa leçon (les données d'entraînement) au lieu de la comprendre, et il échoue dès que la question est posée un peu différemment (sur les nouvelles données).

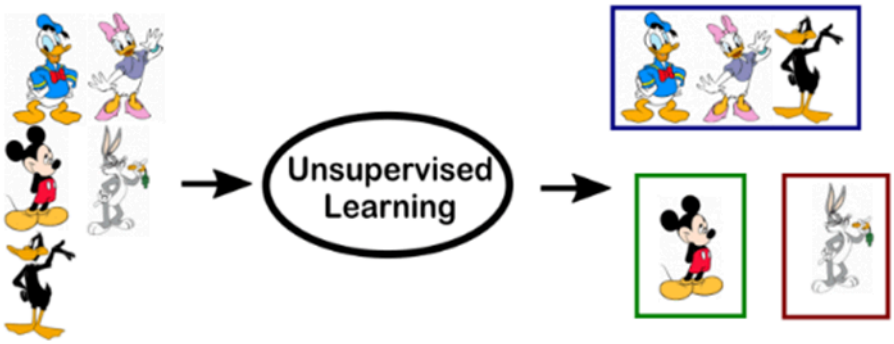
- **Sous-apprentissage (overfitting)**

On dit d'un modèle qui est trop simple pour avoir de bonnes performances même sur les données utilisées pour le construire qu'il sous-apprend. En anglais, on parle **d'underfitting.**

4. II Les algorithmes de l'apprentissage non supervisé :

Dans le cas de l'apprentissage non supervisé, les données ne sont pas étiquetées. Il s'agit alors de modéliser les observations pour mieux les comprendre.

4.1. Définition



L'**apprentissage non supervisé** est une branche du *Machine Learning* qui s'occupe des situations où **le modèle apprend à partir de données sans réponses connues**.

Autrement dit, on dispose d'un ensemble de ***n* observations** (appelées souvent x_i) qui représentent les différents exemples de nos données.

Le but de l'apprentissage non supervisé est de **trouver une structure cachée ou des relations** dans ces données, sans qu'on lui dise à l'avance quelle est la bonne réponse (voir figure suivante).

En somme, le modèle apprend **seul à organiser ou à regrouper** les données selon leurs ressemblances ou leurs différences.

4.2. Les familles de problèmes d'apprentissage non supervisés

4.2.1. Clustering

Exemple

Voici quelques exemples de problèmes de partitionnement :

- La segmentation de marché consiste à identifier des groupes d'usagers ou de clients ayant un comportement similaire. Cela permet de mieux comprendre leur profil, et cibler une campagne de publicité, des contenus ou des actions spécifiquement vers certains groupes ;
- Identifier des groupes de documents ayant un sujet similaire, sans les avoir au préalable étiquetés par sujet. Cela permet d'organiser de larges banques de textes.
- La compression d'image peut être formulée comme un problème de partitionnement consistant à regrouper des pixels similaires pour ensuite les représenter plus efficacement ;
- La segmentation d'image consiste à identifier les pixels d'une image appartenant à la même région ;
- Identifier des groupes parmi les patients présentant les mêmes symptômes permet d'identifier des sous-types d'une maladie, qui pourront être traités différemment.

Le **clustering**, ou **partitionnement**, est une méthode d'**apprentissage non supervisé**.

Son objectif est de **regrouper automatiquement les données** en plusieurs ensembles appelés **groupes** ou **clusters**.

Chaque groupe contient des données qui se **ressemblent entre elles**, tandis que les données appartenant à des groupes différents sont **aussi différentes que possible**.

Par exemple, si l'on dispose de **n observations** (ou points de données), le clustering consiste à les répartir en **K groupes** de manière cohérente selon certains **critères de similarité** (comme la distance ou les caractéristiques communes).

Ainsi, le clustering permet de **découvrir la structure cachée des données**, sans qu'on ait besoin de connaître les bonnes réponses à l'avance (voir figue ci-dessus).

a) Réduction de dimension

La **réduction de dimension** est une méthode d'**apprentissage non supervisé** qui consiste à **représenter les données dans un espace plus simple et plus petit**.

Autrement dit, au lieu de travailler dans un espace très grand (appelé **X**) où les données sont décrites par de nombreuses caractéristiques, on cherche à les **projeter dans un nouvel espace** (appelé **Z**) ayant **moins de dimensions**.

Chaque donnée de départ, notée *xi*, est donc transformée en une version plus compacte *zi* dans ce nouvel espace (voir la figure ci-dessous).

Le but est que ces nouvelles représentations *zi* **gardent les informations importantes** des données d'origine tout en éliminant les détails inutiles.

Ainsi, la réduction de dimension permet de **simplifier les données** tout en **préservant leur sens essentiel**.

Remarque :

~ Certaines méthodes de réduction de dimension sont supervisées : il s'agit alors de trouver la représentation la plus pertinente pour prédire une étiquette donnée.

~ Il existe plusieurs techniques de réduction de dimension, dont la plus populaire est la méthode **PCA**, **Analyse en Composantes Principales**.

Estimation de densité

Une autre grande catégorie de problèmes d'**apprentissage non supervisé** concerne l'**estimation de la loi de probabilité** des données.

Cela signifie que l'on cherche à **comprendre comment les données sont réparties** ou **quelle forme a leur distribution**.

En d'autres termes, on suppose que les données que l'on possède ne sont qu'un **échantillon aléatoire** d'une population plus grande, et on veut **trouver la règle mathématique (la loi de probabilité)** qui décrit comment ces données ont été générées.

Cette approche vient des **statistiques traditionnelles** et permet d'**analyser et de modéliser les données** même lorsqu'on ne connaît pas les bonnes réponses à l'avance.

4.3. Les algorithmes d'apprentissage semi-supervisé :

Exemples

Voici quelques exemples de problèmes d'apprentissage semi- supervisées :

- **La classification d'images médicales** consiste à utiliser un petit nombre d'images déjà étiquetées (par exemple, "tumeur bénigne" ou "tumeur maligne") et un grand nombre d'images non étiquetées pour **améliorer la précision du diagnostic automatique** tout en réduisant le besoin de marquage manuel par les médecins ;
- **~ La détection de spams dans les e-mails** peut s'appuyer sur quelques milliers de messages déjà classés comme *spam* ou *non spam*, et sur un grand volume de messages non étiquetés, afin d'**entraîner un modèle plus robuste** capable de mieux reconnaître les spams ;
- **La reconnaissance d'objets dans les vidéos** peut utiliser quelques images annotées manuellement (indiquant les objets visibles) et de nombreuses autres non annotées pour **apprendre à identifier les objets dans toutes les séquences vidéo**.
- **L'analyse des sentiments dans les réseaux sociaux** consiste à utiliser un petit ensemble de messages ou de commentaires étiquetés comme *positif*, *négatif* ou *neutre*, combiné à un grand nombre de messages non étiquetés, pour **mieux comprendre l'opinion générale des utilisateurs**.
- **La classification de documents** repose souvent sur quelques textes classés par thème (politique, sport, économie…) et un grand nombre d'autres non classés ; l'apprentissage semi-supervisé permet alors **d'améliorer la catégorisation automatique** de tous les documents.

L'**apprentissage semi-supervisé** est une méthode de *Machine Learning* qui se situe **entre l'apprentissage supervisé et non supervisé**.

Il consiste à **apprendre à partir d'un jeu de données dont seule une partie est étiquetée**, c'est-à-dire que certaines données ont une réponse connue, mais d'autres non.

Cette approche est très utile lorsqu'il est **facile de collecter beaucoup de données**, mais **difficile ou coûteux de leur attribuer des étiquettes**. Par exemple, dans la **classification d'images**, on peut rassembler facilement des milliers de photos, mais **indiquer pour chacune ce qu'elle représente** (chat, voiture, maison, etc.) demande beaucoup de **travail humain**.

De plus, comme les étiquettes sont données par des personnes, elles peuvent parfois **contenir des erreurs ou des biais humains**, que le modèle risquerait ensuite de reproduire.

L'apprentissage semi-supervisé permet **d'utiliser les données non étiquetées pour améliorer le modèle**, tout en **réduisant le besoin d'un étiquetage complet**.

C'est une **méthode plus avancée**, mais très prometteuse, que l'on étudie généralement **dans des niveaux plus approfondis du Machine Learning**

4.4. Les algorithmes par renforcement :

Exemples

Voici quelques exemples de problèmes d'apprentissage semi- supervisées :

~ **L'entraînement d'un agent à jouer à un jeu vidéo** consiste à apprendre quelles actions effectuer (avancer, sauter, attaquer, etc.) pour **obtenir le meilleur score possible**. L'agent reçoit des récompenses lorsqu'il progresse dans le jeu ou atteint des objectifs, et apprend ainsi à **développer sa propre stratégie de jeu**.

~ **L'apprentissage d'un robot mobile** consiste à permettre à un robot d'**apprendre à se déplacer dans un environnement inconnu** sans heurter d'obstacles. À chaque bonne action (éviter un mur, atteindre une destination), il reçoit une récompense positive, et une pénalité lorsqu'il fait une erreur.

~ **La gestion automatique d'un feu de circulation** peut être formulée comme un problème d'apprentissage par renforcement : le système apprend à **réguler la durée des feux rouges et verts** pour minimiser les embouteillages, en recevant une récompense lorsqu'il améliore la fluidité du trafic.

~ **L'optimisation de la consommation d'énergie dans un bâtiment intelligent** consiste à apprendre comment ajuster la climatisation, le chauffage ou l'éclairage afin de **réduire la consommation tout en maintenant le confort des occupants**. Les récompenses dépendent du niveau d'économie d'énergie réalisé.

~ **L'entraînement d'un agent financier virtuel** permet d'apprendre à **acheter ou vendre des actions** en fonction des variations du marché, dans le but d'obtenir un **rendement maximal**. L'agent reçoit une récompense quand ses décisions génèrent des profits.

L'**apprentissage par renforcement** est une méthode du *Machine Learning* où un **système apprend en interagissant avec un environnement**.

Concrètement, il **effectue des actions** et reçoit ensuite une **récompense** :

~ la récompense est **positive** si l'action était bonne,

~ et **négative** si l'action était mauvaise.

Le but du système est donc d'**apprendre, au fil du temps, quelles actions mènent aux meilleures récompenses**.

Parfois, cette récompense n'arrive **qu'après plusieurs actions**, comme lorsqu'un programme apprend à **jouer aux échecs ou au jeu de go** : il ne sait s'il a bien joué qu'à la **fin de la partie**.

L'objectif de ce type d'apprentissage est de **trouver une stratégie**, appelée **politique**, qui permet de **prendre les meilleures décisions possibles** pour obtenir la récompense la plus élevée.

L'**apprentissage par renforcement** est surtout utilisé dans des domaines comme **les jeux** (échecs, go, jeux vidéo) et **la robotique**, où les systèmes doivent **apprendre à agir de manière autonome**.

C'est un **sujet avancé**, souvent étudié après avoir bien compris les bases du Machine Learning.