

IA

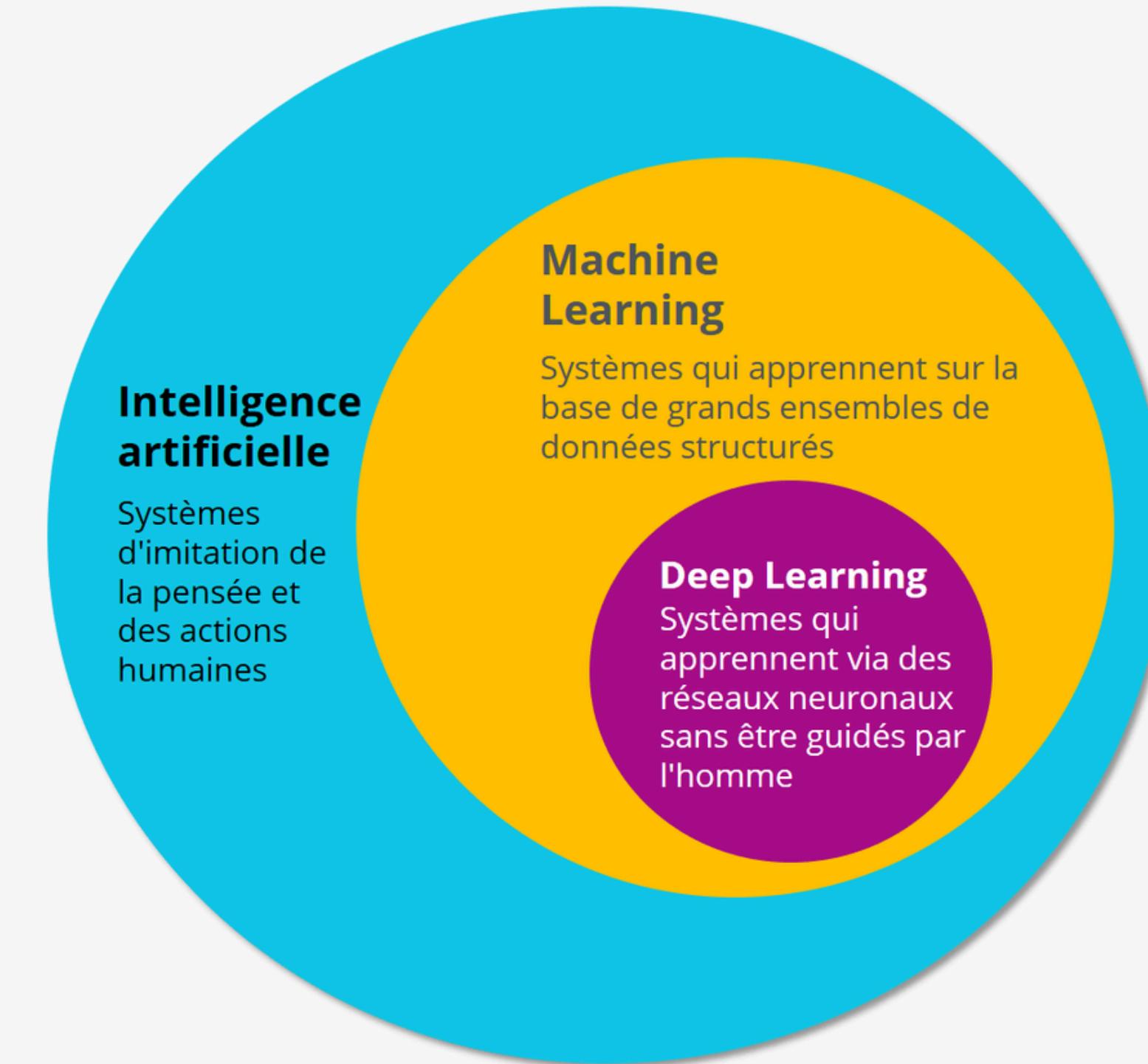
Programation avancé

Sommaire

- Shéma de l'ia : p3
- Deep learning : p4
 - Les Hyperparamètres : p11
- Machine learning : p34
- L'IA : p44



L'IA



IONOS

Deep Learning

Le **deep learning** est une branche de l'intelligence artificielle qui permet aux ordinateurs d'apprendre à partir de grandes quantités de données. Il s'appuie sur des réseaux de neurones artificiels, qui sont des modèles mathématiques inspirés du fonctionnement du cerveau humain.

Définition



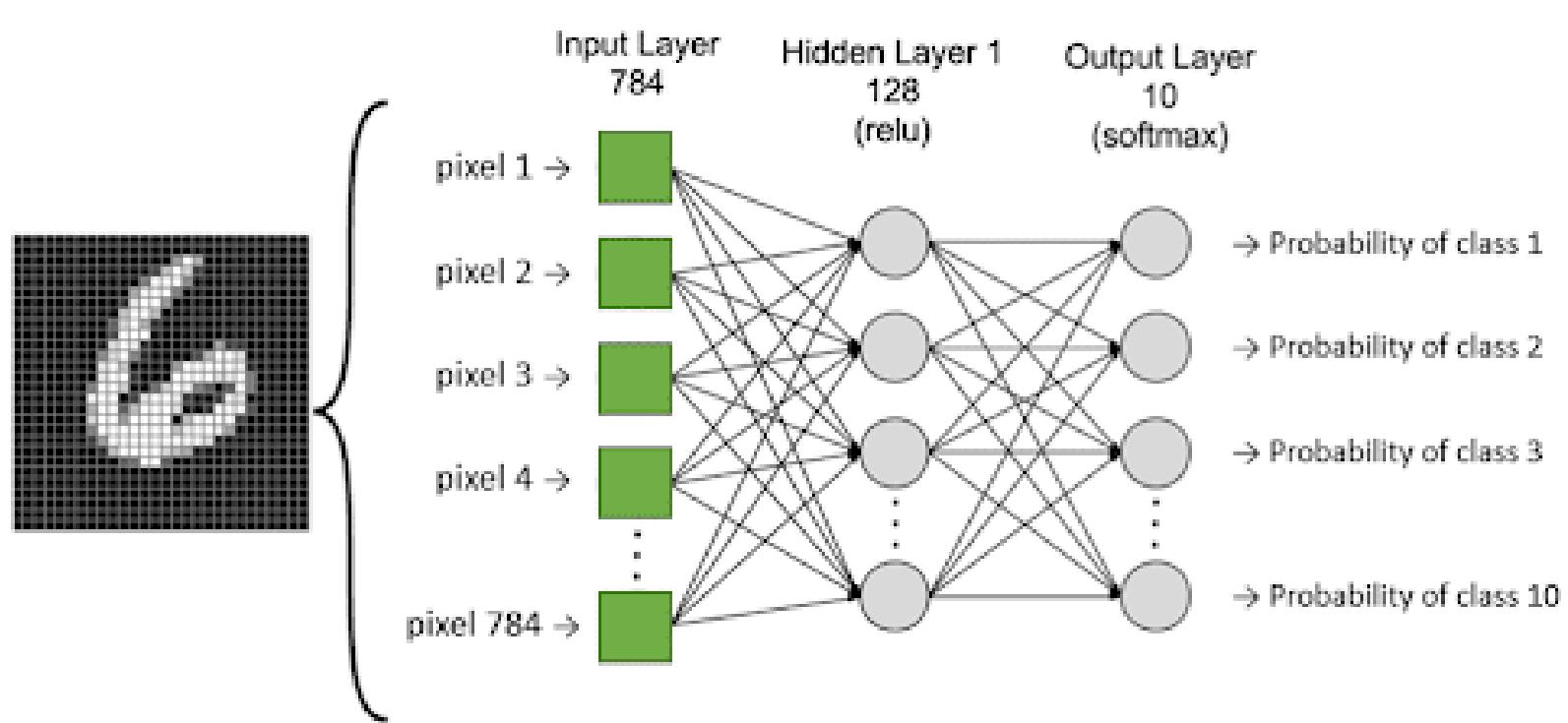
Exemple simple :

C'est un peu comme apprendre à reconnaître des lettres. Au début, vous ne savez pas comment faire, mais en voyant beaucoup d'exemples et en vous trompant plusieurs fois, vous ajustez peu à peu votre façon de regarder les lettres, jusqu'à pouvoir les reconnaître avec précision.

Deep Learning

une neurone ?

Un neurone dans le contexte du deep learning est une unité de base dans un réseau de neurones artificiels, inspirée du neurone biologique du cerveau. Il reçoit des informations, effectue un calcul simple, puis envoie un résultat à d'autres neurones.



Fonctionnement simplifié :

1. Entrées : Chaque neurone reçoit plusieurs entrées, qui sont des nombres (cela peut être des données brutes comme des pixels d'une image, ou des sorties d'autres neurones).
2. Poids : Chaque entrée est multipliée par un poids, qui est un nombre ajustable que le neurone utilise pour donner plus ou moins d'importance à cette entrée.
3. Somme : Le neurone fait la somme de toutes les entrées pondérées par leurs poids.
4. Fonction d'activation : Le résultat de cette somme passe ensuite par une fonction d'activation. Cette fonction décide si le neurone "s'active" ou non, c'est-à-dire s'il transmet un signal en sortie. (pas s'activer = fail de la neurone)
5. Sortie : Si le neurone s'active, il envoie une sortie vers les neurones de la couche suivante.

Deep Learning

une neurone ?



Imagine un neurone comme une personne qui doit décider si elle va ouvrir la porte.

- Elle regarde plusieurs signaux (bruit, son de la sonnette, etc.).
- Elle pèse l'importance de chaque signal (bruit fort = plus important, petit bruit = moins important).
- Elle combine ces signaux et, selon une règle (par exemple, si le bruit dépasse un certain niveau), elle décide d'ouvrir ou non la porte.

Deep Learning

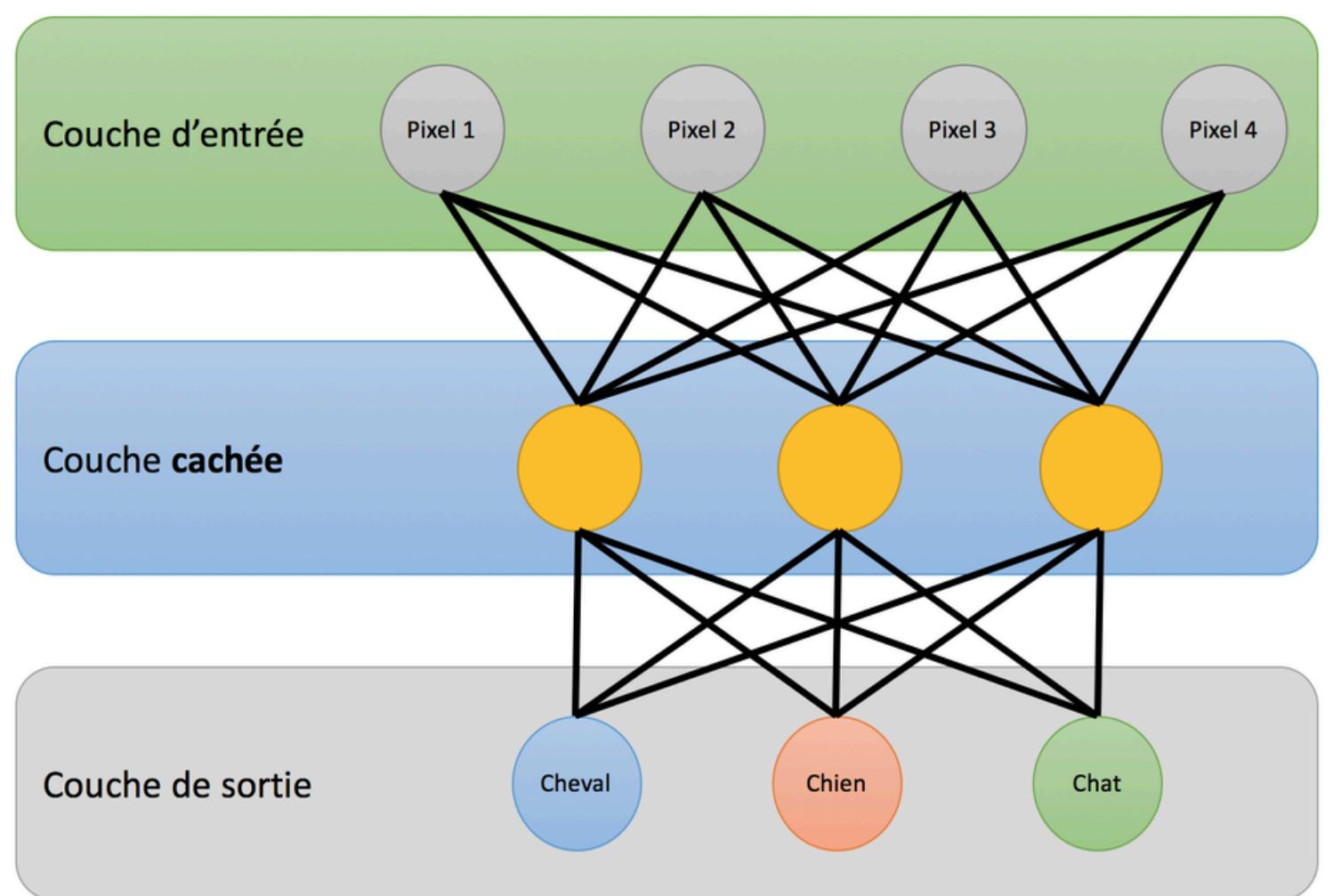
comment les utiliser ?

Le Cerveau



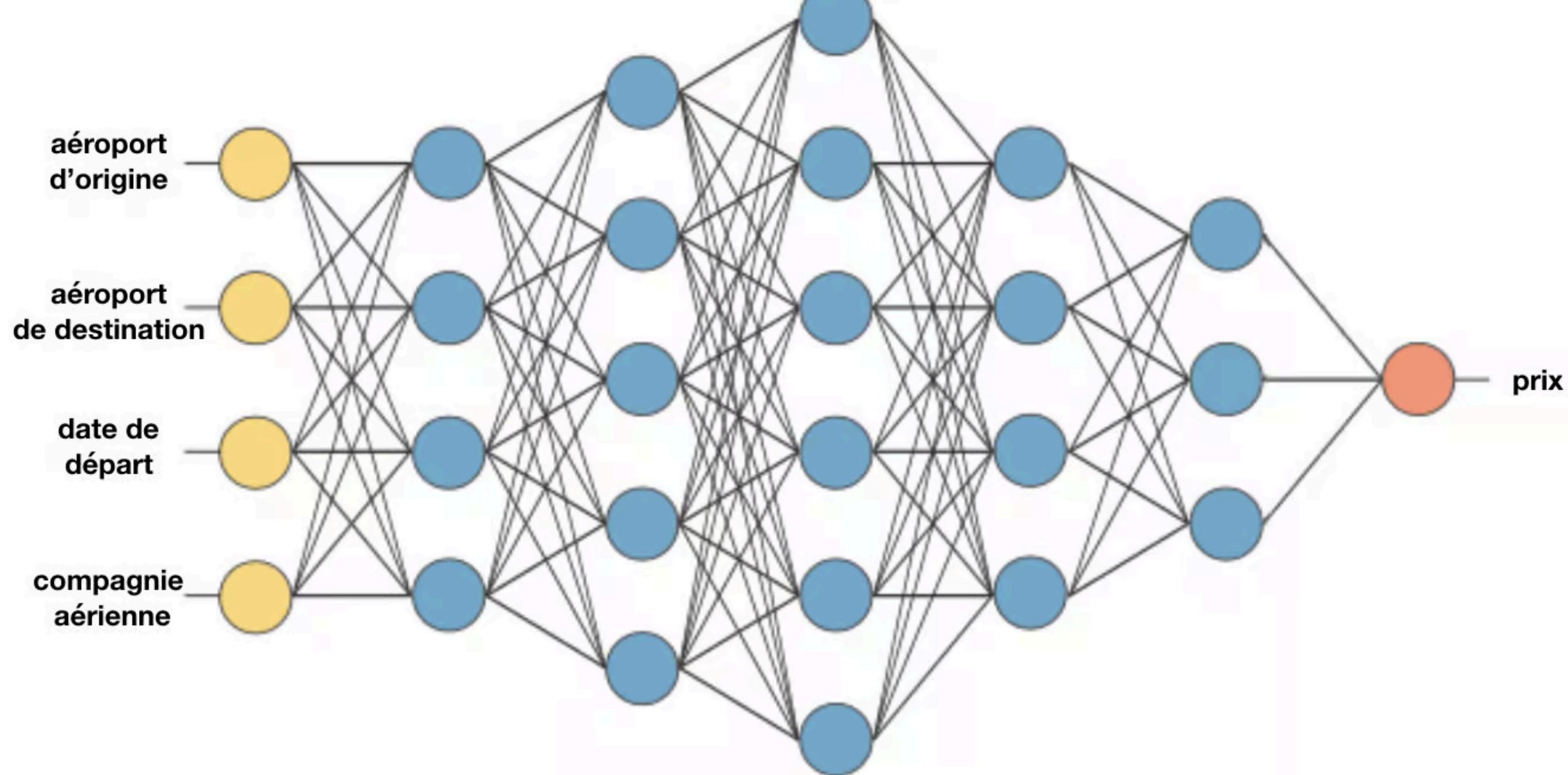
Couches :

- Couche d'entrée : La première couche qui reçoit les données brutes à traiter.
- Couches cachées : Une ou plusieurs couches intermédiaires composées de nombreux neurones. Ces couches apprennent des représentations des données d'entrée en passant par des processus de somme pondérée et d'activation.
- Couche de sortie : La dernière couche qui produit les résultats finaux du modèle, tels que des classes prédites ou des valeurs de régression.



Deep Learning

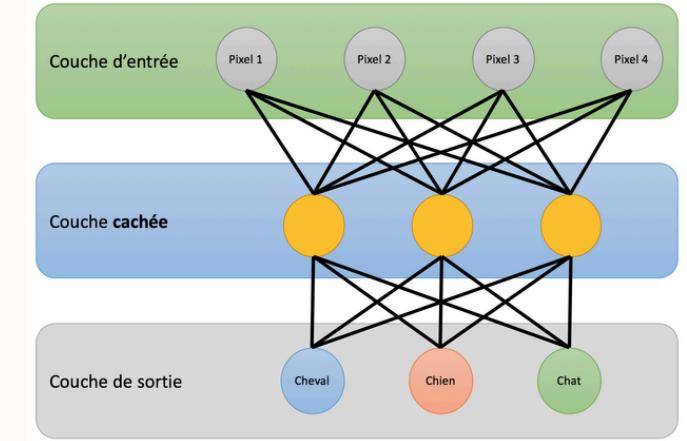
comment les utiliser ?



Deep Learning

La couche d'entrée

La première couche du réseau reçoit les données d'entrée brutes, qui peuvent être des images, des textes ou d'autres types de données.



Calculer votre IMC - Indice de masse corporelle

Votre taille : * cm

Votre poids : * kilos

CALCULER

Types de Valeurs d'Entrée et Méthodes de Traitement

1. Images

- Traitement : Redimensionnement, normalisation des pixels (valeurs entre 0 et 1), augmentation de données (rotation, translation).
- Modèle : Réseaux de neurones convolutionnels (CNN).
- Exemple : Classification d'images de chiens et de chats.

2. Texte

- Traitement : Tokenization (conversion de texte en mots ou sous-mots), encodage (par exemple, encodage One-Hot ou embeddings).
- Modèle : Réseaux de neurones récurrents (RNN) ou Transformers.
- Exemple : Analyse de sentiments à partir d'avis de clients.

3. Données Numériques/Tabulaires

- Traitement : Normalisation ou standardisation des valeurs, gestion des valeurs manquantes.
- Modèle : Réseaux de neurones fully connected.
- Exemple : Prédiction des prix des maisons en fonction de caractéristiques.

Deep Learning

Architecture

1. CNN (Convolutional Neural Network)

Un réseau de neurones pour analyser des images en utilisant des filtres pour extraire des caractéristiques (comme des bords).

Exemple : Classification d'images dans une application de détection de maladies des plantes, où le modèle identifie si une feuille est saine ou malade en analysant des photos.

2. RNN (Recurrent Neural Network)

Un réseau conçu pour traiter des séquences, comme du texte ou des séries temporelles, en utilisant des connexions qui se souviennent des informations précédentes.

Exemple : Prédiction de la prochaine parole dans une phrase pour un assistant vocal, où le modèle utilise le contexte des mots précédents pour générer une réponse.

3. Transformers

Une architecture qui utilise l'attention pour traiter des séquences de données, efficace pour le traitement du langage naturel (NLP).

Exemple : Traduction automatique de phrases d'une langue à une autre, comme dans Google Translate, où le modèle analyse le texte entier pour produire une traduction fluide.

4. Réseaux de Neurones Fully Connected (Dense)

Un réseau où chaque neurone est connecté à tous les neurones de la couche suivante, utilisé pour la classification et la régression.

Exemple : Prédiction des prix d'immobilier à partir de caractéristiques comme la superficie et le nombre de chambres, où un modèle dense combine toutes ces informations pour faire une estimation.

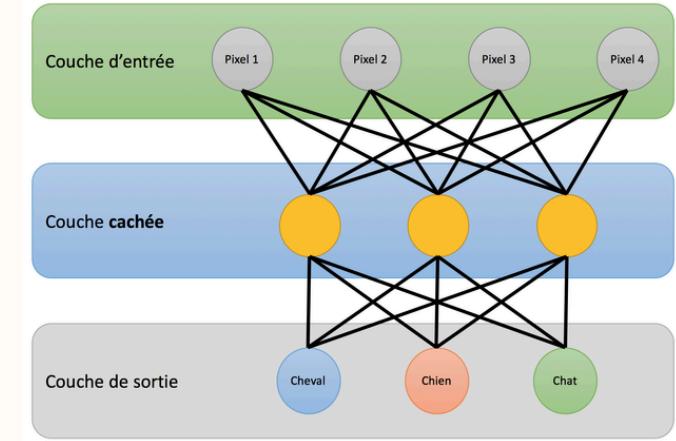
Deep Learning

La ou les couche caché

Chaque couche cachée est composée de plusieurs neurones, et chacun de ces neurones effectue le processus décrit ci-dessus. Par exemple, une couche cachée peut contenir 128 neurones, et chacun de ces neurones va recevoir les mêmes entrées de la couche précédente, mais appliquera des poids différents et aura des fonctions d'activation.

Hyperparamètres d'un Réseau de Neurones :

1. Batch Size : Taille des lots de données traités à chaque itération, influençant l'efficacité de l'entraînement.
2. Nombre de Couches et de Neurones : Profondeur du réseau et nombre de neurones par couche, ajustés selon la complexité du problème.
3. Fonction d'Activation : Choix de la fonction (ReLU, sigmoid, etc.) qui détermine l'activation des neurones.
4. Fonction de Perte : Mesure de l'erreur entre les prédictions et les vraies étiquettes, utilisée pour ajuster les poids (ex. MSE pour régression, entropie croisée pour classification).
5. Pourcentage de Validation : Proportion des données réservées pour évaluer la capacité de généralisation du modèle après l'entraînement.
6. Époques : Nombre de passages complets à travers l'ensemble des données d'entraînement, permettant d'améliorer les performances du modèle.



Calculer votre IMC - Indice de masse corporelle

Votre taille : * cm

Votre poids : * kilos

CALCULER

Deep Learning

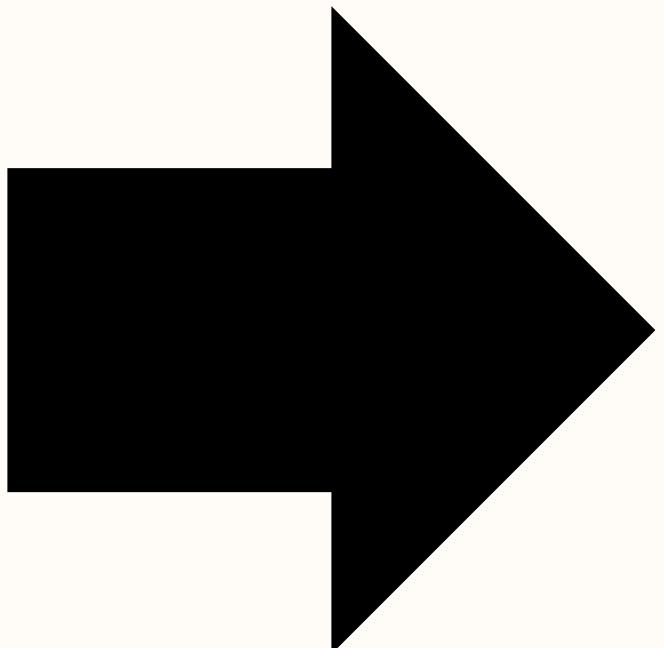
Batch size

Exemple :

Imaginons un boulanger qui doit préparer 100 viennoiseries dans la journée.

Option 1 : Faire une viennoiserie à la fois (équivalent à une taille de batch = 1)

Le boulanger prépare un seul pain, le fait cuire, puis passe au pain suivant. C'est comme si, dans un réseau de neurones, le modèle apprenait à partir d'un seul exemple de données à chaque fois (descente de gradient stochastique). Cela permet au boulanger d'adapter rapidement la recette en fonction du résultat du pain, mais cela prend beaucoup de temps car chaque pain est cuit individuellement. Le four est allumé et éteint à chaque fois, ce qui n'est pas efficace.



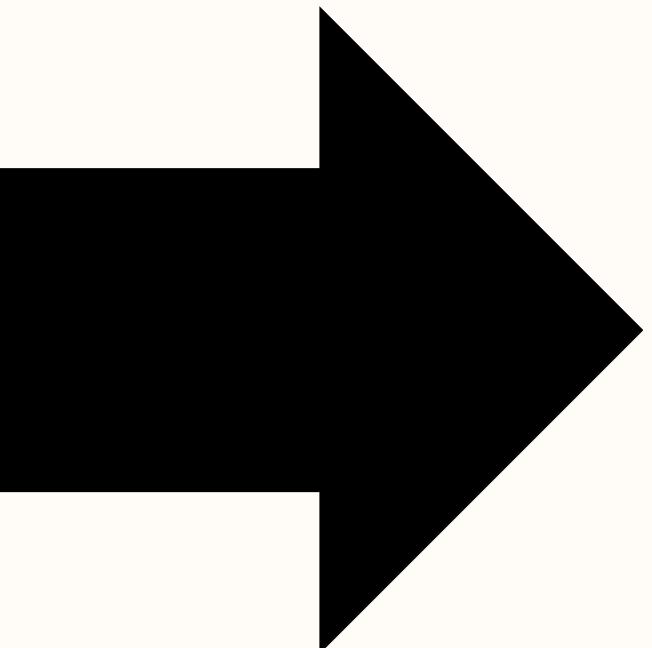
Deep Learning

Batch size

Exemple :

Option 2 : Faire tous les pains d'un coup (équivalent à une taille de batch = 100)

Le boulanger prépare les 100 viennoiseries d'un coup, puis les fait cuire ensemble. C'est l'équivalent de prendre toutes les données d'entraînement pour ajuster les poids du modèle (descente de gradient classique). Cela maximise l'utilisation du four et est très efficace pour la cuisson. Cependant, si une erreur est commise (par exemple, si la température du four était incorrecte), il y a un risque de ruiner tous les pains d'un coup, et il est difficile de corriger l'erreur avant d'avoir terminé tout le processus.



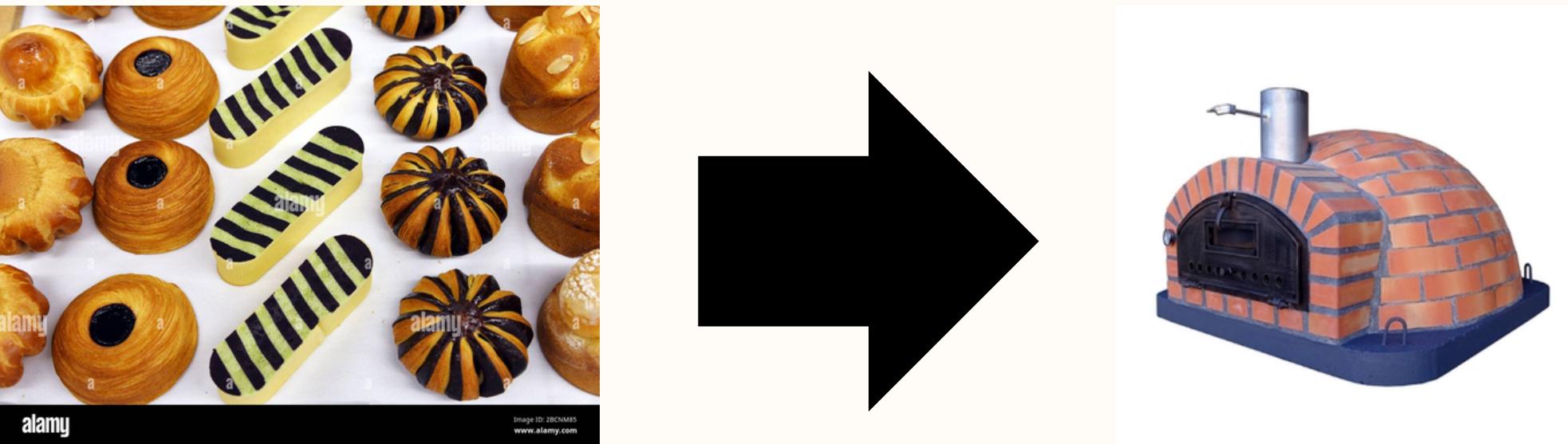
Deep Learning

Batch size

Exemple :

Option 3 : Faire des groupes de 10 viennoiseries (équivalent à une taille de batch = 10)

Le boulanger prépare 10 viennoiseries à la fois, les fait cuire ensemble, puis ajuste légèrement la recette en fonction du résultat avant de passer à la série suivante. C'est comme si le modèle apprenait à partir d'un petit sous-ensemble de données à chaque itération. Cela équilibre l'efficacité (utilisation du four en groupe) et la flexibilité (adaptation de la recette au fil du temps), tout en réduisant les erreurs majeures.



Conclusion :

- Batch de 1 pain : Apprentissage rapide mais très lent pour le processus global.
- Batch de 100 pains : Apprentissage lent mais processus global rapide.
- Batch de 10 pains : Un bon compromis entre les deux, permettant des ajustements et une efficacité raisonnable.

Dans le deep learning, choisir la taille du batch revient à décider combien de pains on fait cuire ensemble pour optimiser le processus !

Deep Learning

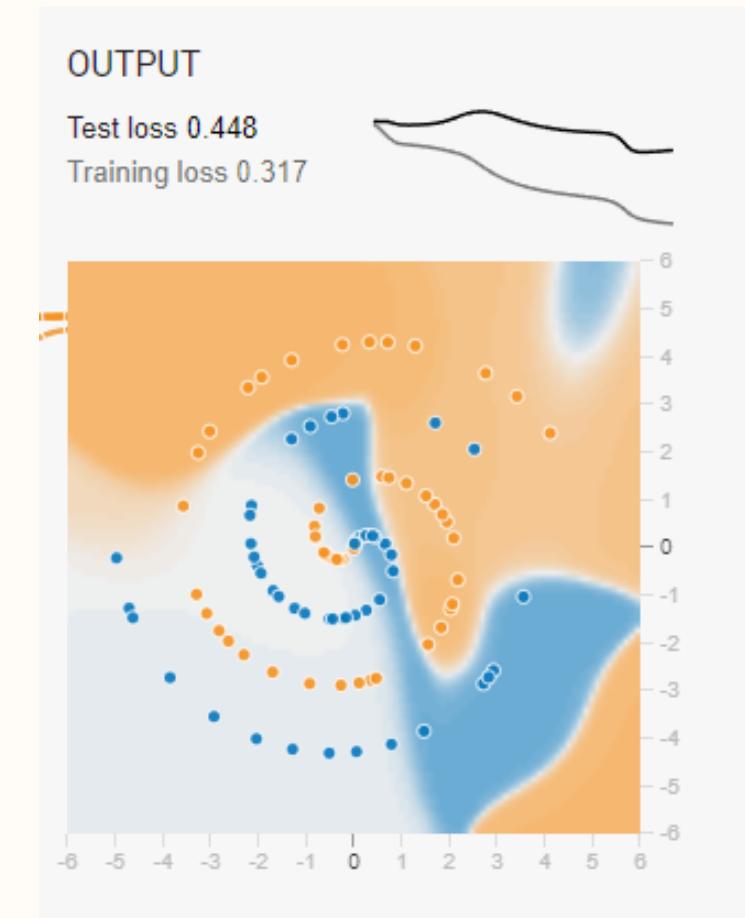
1. Profondeur du Réseau (Nombre de Couches)

- Définition : La profondeur d'un réseau de neurones fait référence au nombre de couches cachées qu'il possède. Chaque couche contient plusieurs neurones qui apprennent à partir des entrées qu'ils reçoivent.

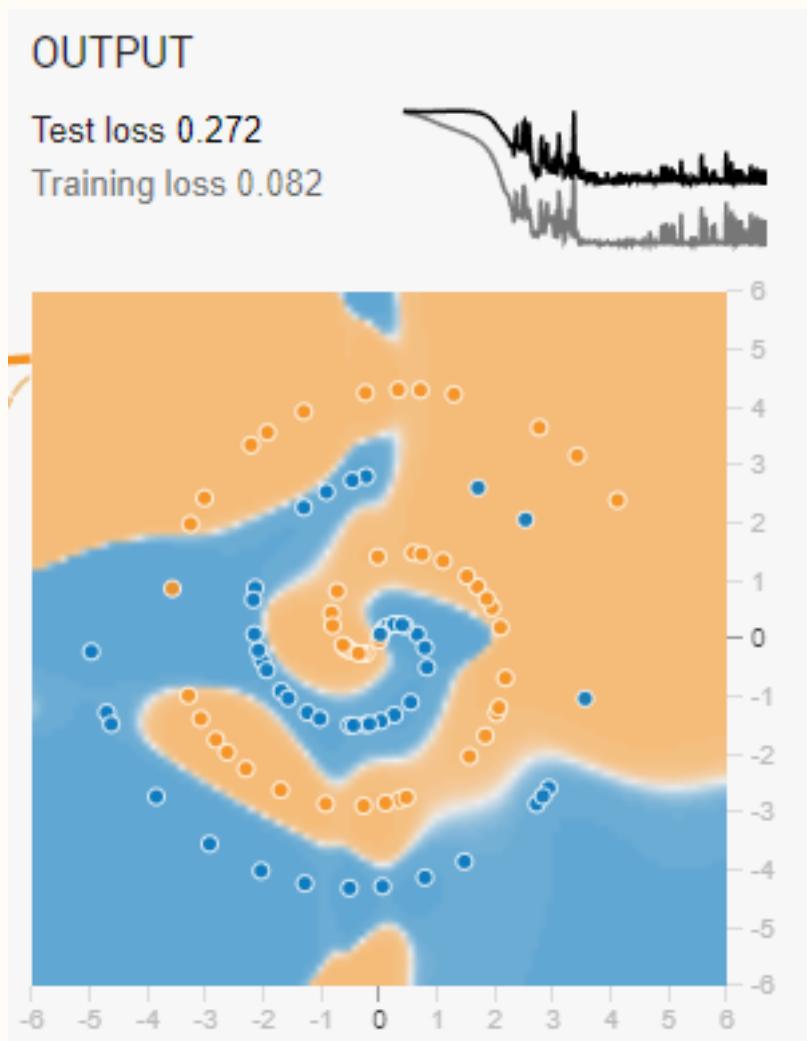
2. Nombre de Neurones par Couche

- Définition : Cela fait référence au nombre de neurones dans chaque couche cachée. Chaque neurone prend une combinaison linéaire de ses entrées, puis applique une fonction d'activation pour produire une sortie.

Pas beaucoup



Beaucoup



Nb couches et neurones

Deep Learning

Optimizer

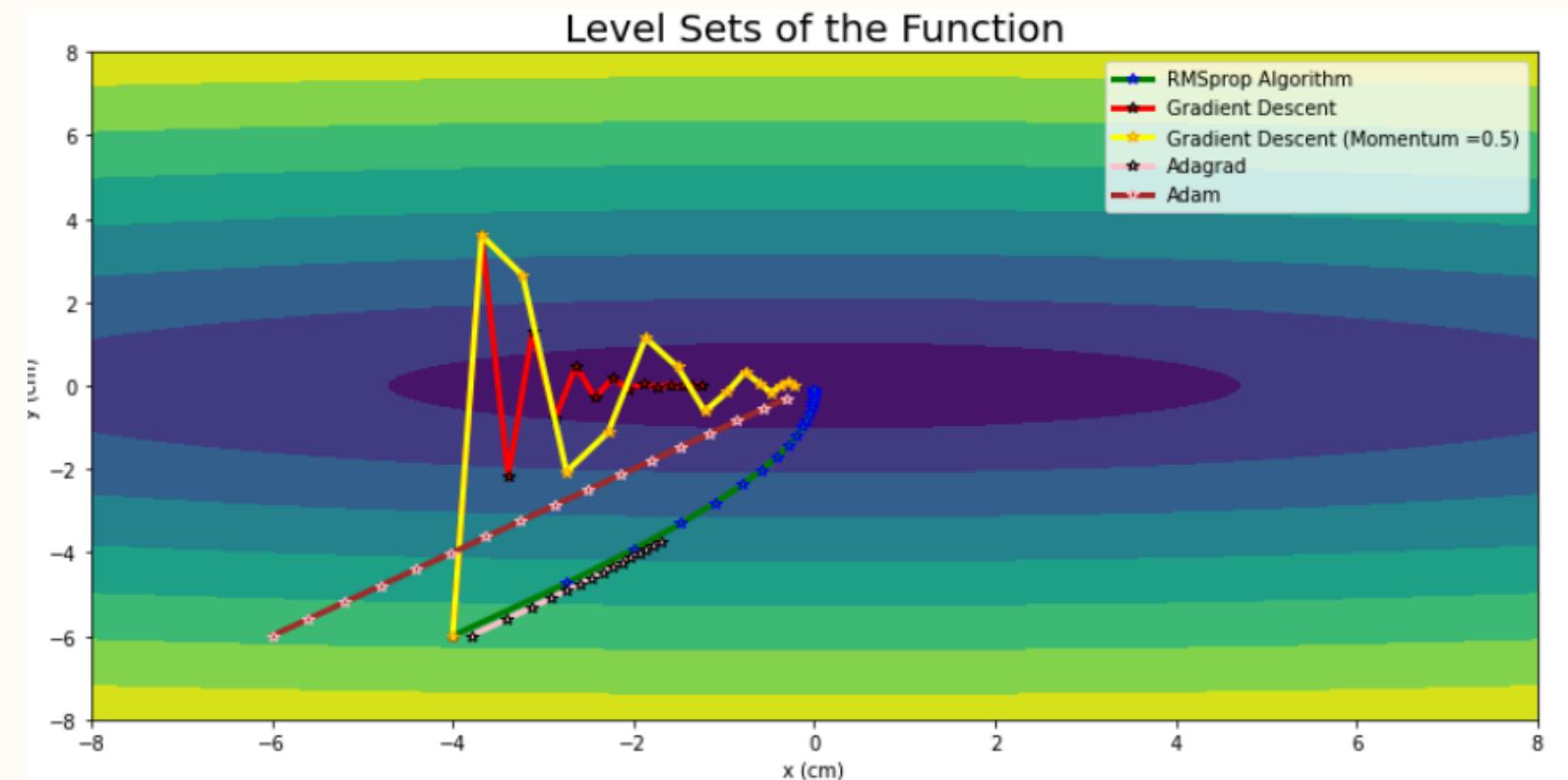
Optimizer :

- L'optimizer est l'algorithme qui ajuste les poids et les biais du réseau pendant l'entraînement. Son rôle est de minimiser la fonction de coût (ou fonction de perte) du modèle, c'est-à-dire de réduire l'erreur entre les prédictions du modèle et les valeurs réelles.
- Il agit en mettant à jour les poids du réseau via une méthode de descente de gradient ou une variante.
- Exemple d'optimizers :
 - Gradient Descent : Méthode classique pour minimiser une fonction de coût en ajustant les poids dans la direction opposée au gradient.
 - Adam : Combinaison de deux autres méthodes (RMSProp et Momentum) qui s'adapte dynamiquement à chaque paramètre du réseau.
 - RMSProp, Adagrad, SGD (Stochastic Gradient Descent) : Différentes variantes de descente de gradient.

Résumé :

- Fonction d'activation : Transforme la sortie d'un neurone.
- Optimizer : Met à jour les poids du modèle pour améliorer ses performances.

Les deux sont essentiels dans l'entraînement d'un réseau de neurones, mais ils jouent des rôles très différents dans le processus d'apprentissage.



Deep Learning

1. Sigmoid

- Description : Transforme les valeurs d'entrée dans l'intervalle (0, 1), en forme de S.
- Avantages : Bon pour la classification binaire, sortie interprétable comme une probabilité.
- Inconvénients : Susceptible au gradient évanescence et les sorties ne sont pas centrées autour de zéro, ce qui peut ralentir l'apprentissage.

2. Tanh (Tangente Hyperbolique)

- Description : Transforme les valeurs d'entrée dans l'intervalle (-1, 1), centrée autour de zéro.
- Avantages : Réduit le gradient évanescence par rapport à la sigmoid et favorise une meilleure convergence dans les réseaux profonds.
- Inconvénients : Peut toujours rencontrer des problèmes de gradient évanescence pour des valeurs d'entrée extrêmes.

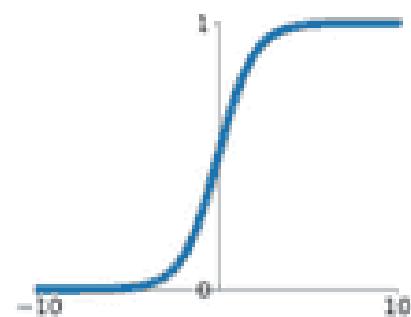
3. ReLU (Rectified Linear Unit)

- Description : Renvoie 0 pour les valeurs négatives et la valeur elle-même pour les positives.
- Avantages : Évite le gradient évanescence et est rapide à calculer.
- Inconvénients : Peut entraîner des neurones "morts" qui ne s'activent jamais.

Fonction d'activation

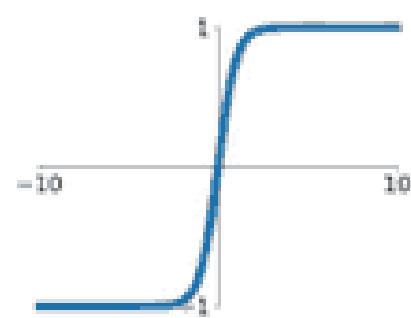
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



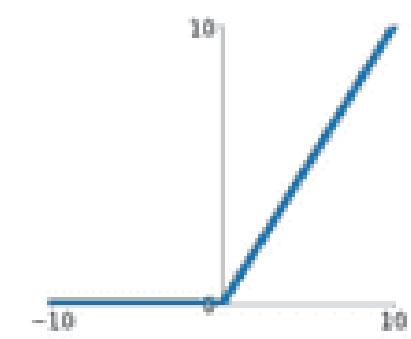
tanh

$$\tanh(x)$$



ReLU

$$\max(0, x)$$



Deep Learning

Fonction d'activation

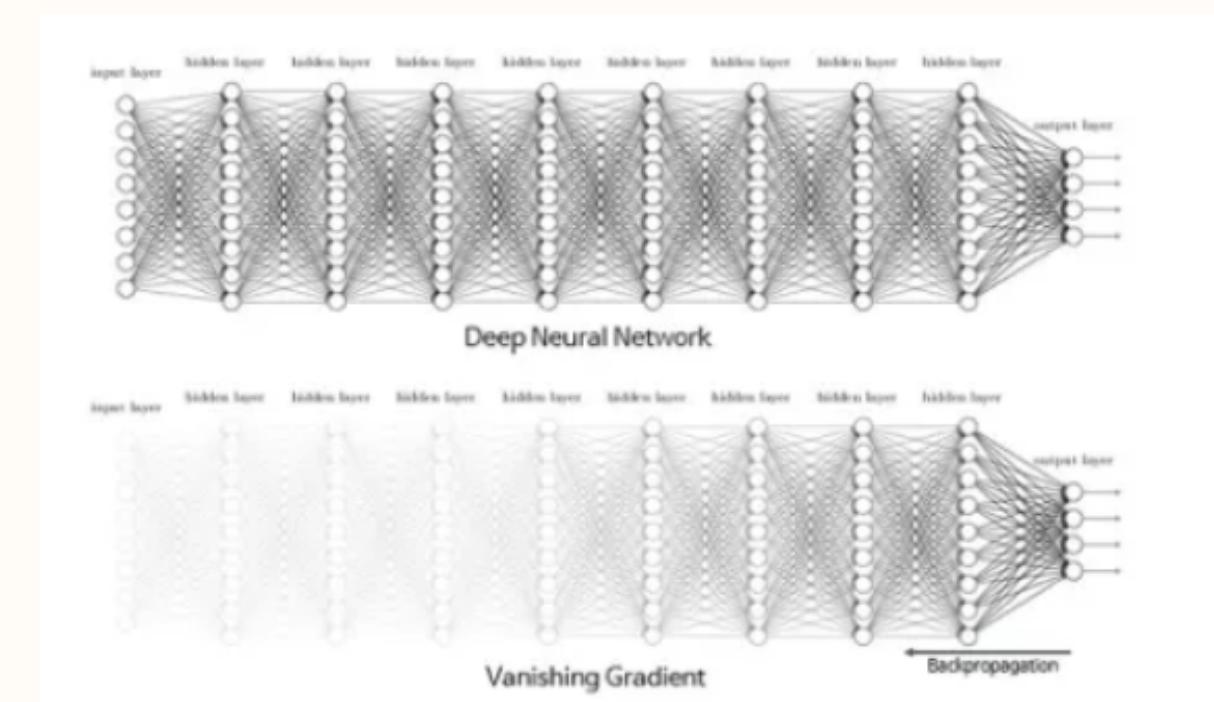
Définition Simple du Gradient Évanescence

Le gradient évanescence est un problème qui se produit lors de l'entraînement des réseaux de neurones profonds. Voici une explication claire :

- Qu'est-ce que c'est ?
- C'est lorsque les gradients (qui mesurent comment les poids du réseau doivent être ajustés) deviennent très petits à mesure qu'ils se déplacent vers les couches inférieures d'un réseau de neurones. Cela signifie que ces couches apprennent très lentement, voire pas du tout.
- Pourquoi c'est un problème ?
- Quand les gradients sont proches de zéro, les poids de ces couches ne changent presque pas. Cela rend l'apprentissage du réseau inefficace et peut l'empêcher de capturer des informations importantes à partir des données.
- Conséquence :
- Le modèle peut ne pas atteindre de bonnes performances, car il ne parvient pas à apprendre correctement à partir des données.

En résumé

Le gradient évanescence rend l'apprentissage difficile pour les couches profondes d'un réseau de neurones, car les ajustements nécessaires deviennent très petits, empêchant le modèle d'apprendre efficacement.



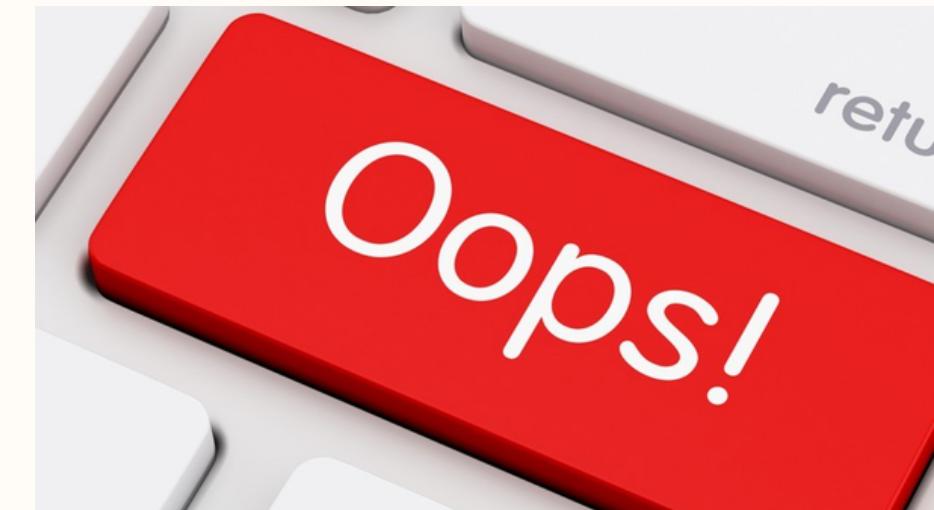
Deep Learning

Qu'est-ce que c'est ?

Une fonction de perte est une formule qui quantifie la différence entre les prédictions faites par un modèle et les résultats réels (les valeurs cibles). Son objectif est de guider l'entraînement du modèle en indiquant à quel point il se trompe.



Fonction de perte



Types de Fonctions de Perte

Il existe plusieurs types de fonctions de perte, chacune adaptée à des types de problèmes spécifiques :

- Pour la Régression :
 - Erreur Quadratique Moyenne (MSE) : Mesure la moyenne des carrés des erreurs entre les prédictions et les valeurs réelles.
 - Erreur Absolue Moyenne (MAE) : Mesure la moyenne des erreurs absolues.
 - Racine de l'Erreur Quadratique Moyenne (RMSE) : Racine carrée de l'EQM, pour ramener l'erreur à la même échelle que les valeurs réelles.
- Pour la Classification :
 - Entropie Croisée : Mesure la différence entre la distribution prédite par le modèle et la distribution réelle des classes. Très utilisée pour les problèmes de classification multi-classes.
 - Fonction de Perte Logistique : Utilisée pour les problèmes de classification binaire, elle est liée à l'entropie croisée.

Deep Learning

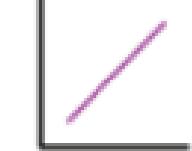
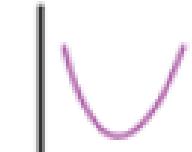
Régression / Classification ?

Régression

La régression est une tâche d'apprentissage automatique où l'objectif est de prédire une valeur continue à partir des données d'entrée.

- Exemple : Prédire le prix d'une maison en fonction de ses caractéristiques (taille, nombre de chambres, localisation).
- Résultat attendu : Une valeur numérique, comme 250 000 € pour une maison.

La régression est utilisée dans des contextes où les résultats sont des nombres sur une plage infinie, et les erreurs entre les valeurs prédites et réelles sont généralement mesurées avec des fonctions de perte comme l'erreur quadratique moyenne (MSE) ou l'erreur absolue moyenne (MAE).

	Forme allongée et diagonale	Régression simple
	Forme logarithmique	Régression logarithmique
	Forme polynomiale	Régression polynomiale
Aucune représentation graphique	Plusieurs données explicatives	Régression multiple

Deep Learning

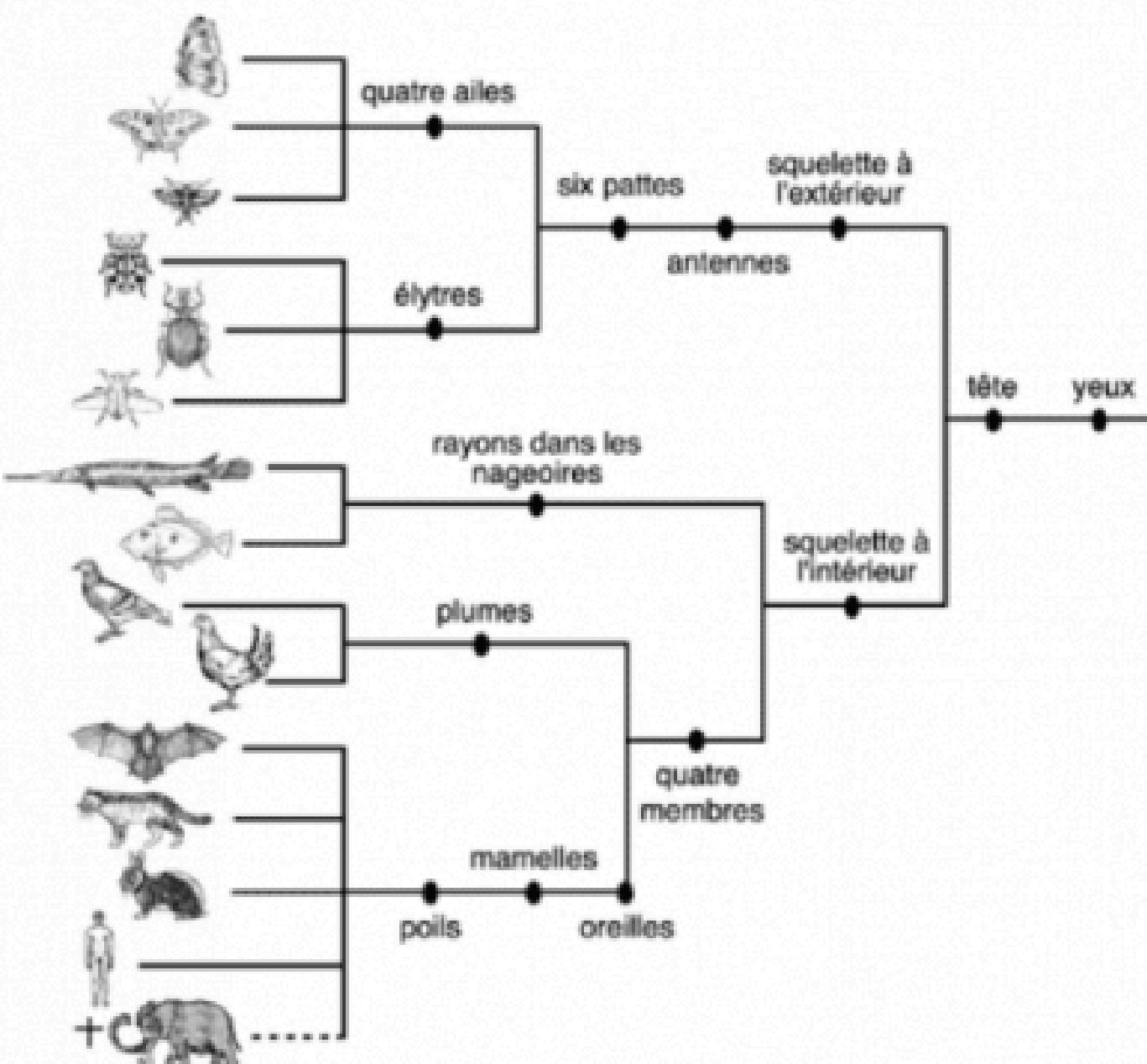
Classification

La classification, quant à elle, est une tâche où l'objectif est de prédire une catégorie ou une étiquette parmi un ensemble de classes prédefinies.

- Exemple : Prédire si un email est "spam" ou "non spam" (binaire) ou identifier le type d'animal sur une image parmi plusieurs catégories (chien, chat, oiseau, etc.).
- Résultat attendu : Une étiquette (par exemple, "spam" ou "non spam").

Les modèles de classification produisent généralement des probabilités pour chaque classe, et des fonctions de perte comme l'entropie croisée ou la fonction de perte logistique sont utilisées pour évaluer la performance du modèle.

Régression / Classification ?



Deep Learning

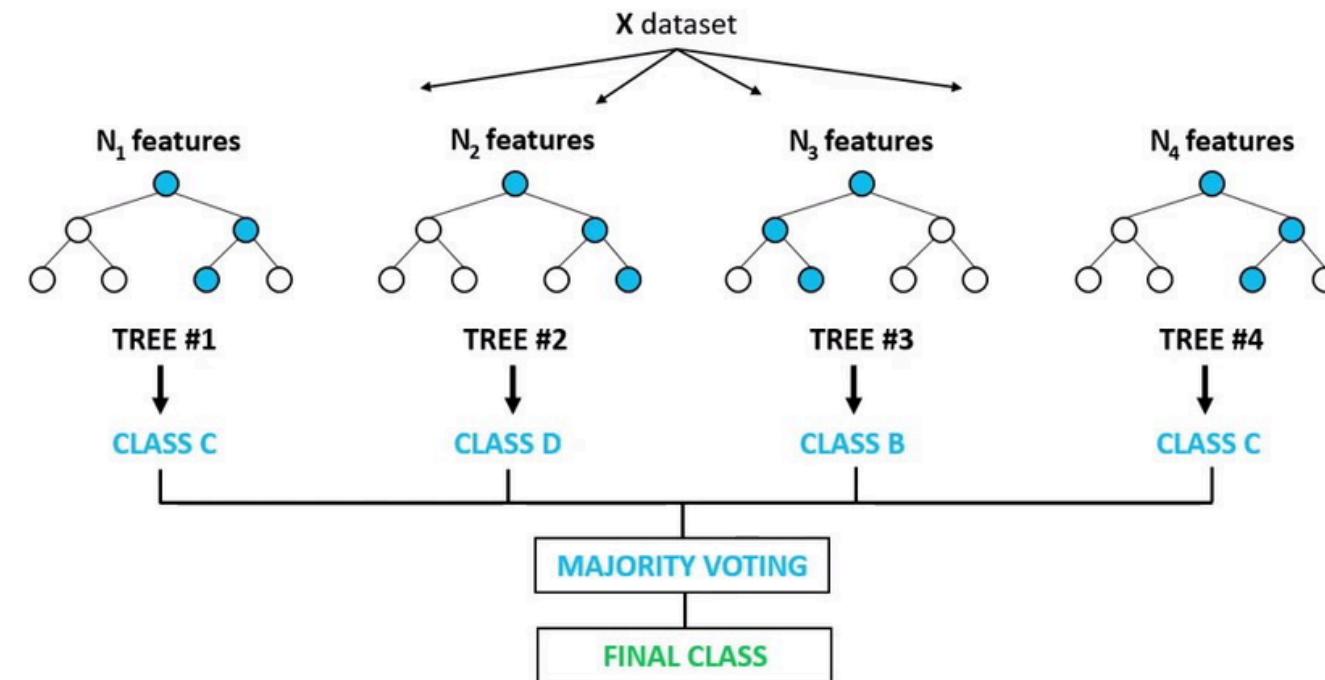
RandomClassifier

Un random classifier prédit les classes d'instances de manière aléatoire. Il ne tient pas compte des caractéristiques des données ou d'apprentissage préalable. Par exemple, si vous avez trois classes (A, B, C), il choisira A, B ou C de manière aléatoire.

L'objectif principal de l'utilisation d'un random classifier est de servir de point de référence. Si un modèle plus complexe (comme un modèle d'apprentissage automatique) ne parvient pas à surpasser la précision du random classifier, cela suggère que le modèle n'est pas efficace ou qu'il y a un problème avec les données. Pour cela on peut l'améliorer avec les hyperparamètre et l'ajuster pour l'améliorer.

Régression / Classification ?

Random Forest Classifier



Deep Learning

Erreur quadratique moyenne (MSE)

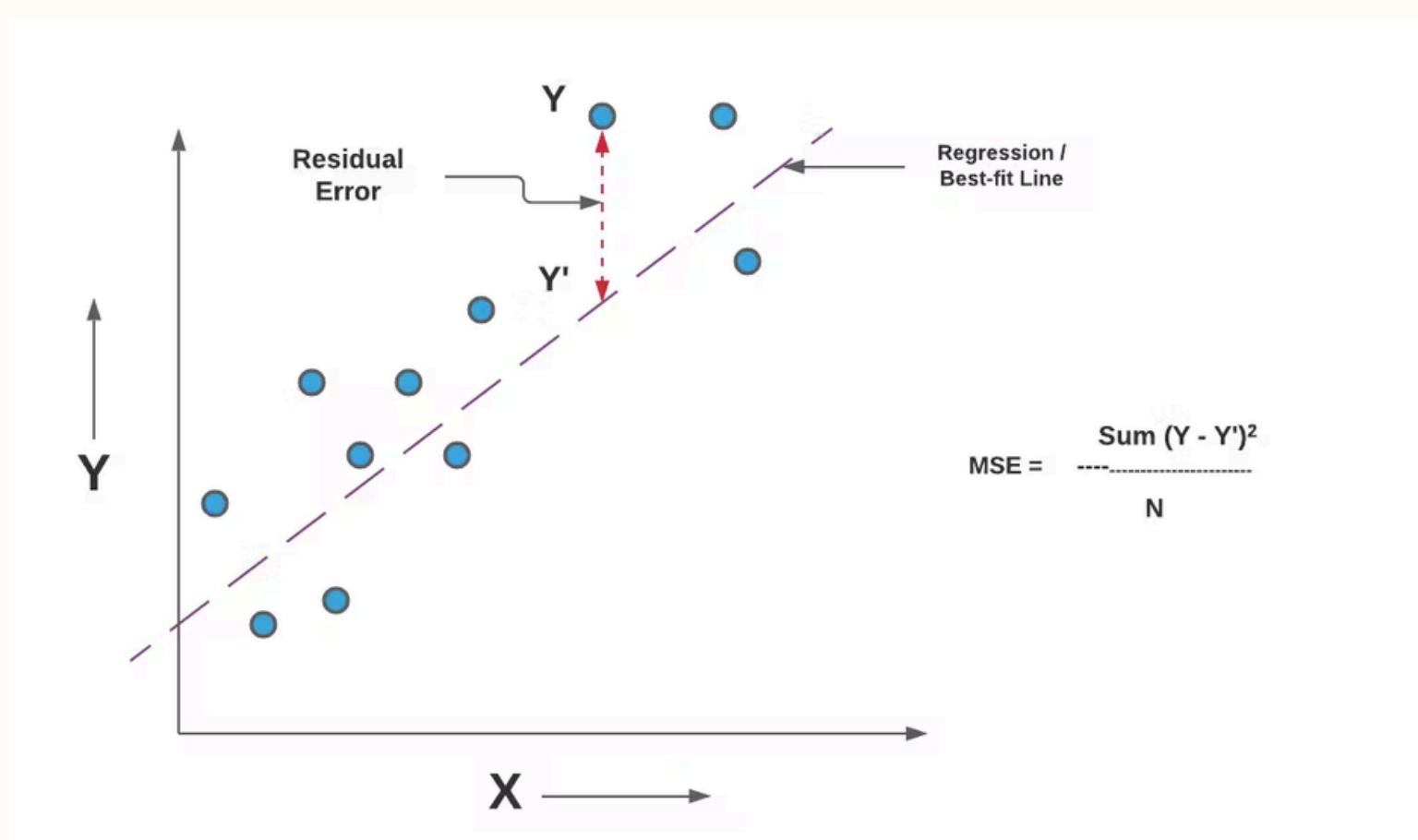
exemple :

Données (prédictions vs valeurs réelles) :

- Prédiction 1 : 20°C, Valeur réelle : 25°C
- Prédiction 2 : 22°C, Valeur réelle : 23°C
- Prédiction 3 : 30°C, Valeur réelle : 27°C

- Erreurs :**
 - Prédiction 1 : $(25 - 20)^2 = 25$
 - Prédiction 2 : $(23 - 22)^2 = 1$
 - Prédiction 3 : $(27 - 30)^2 = 9$
- MSE :**
$$\frac{25+1+9}{3} = \frac{35}{3} = 11.67$$

Fonction de perte



La MSE met l'accent sur les grandes erreurs en les élévant au carré, ce qui peut être utile pour pénaliser les prédictions très éloignées.

Deep Learning

Erreur Absolue Moyenne (MAE)

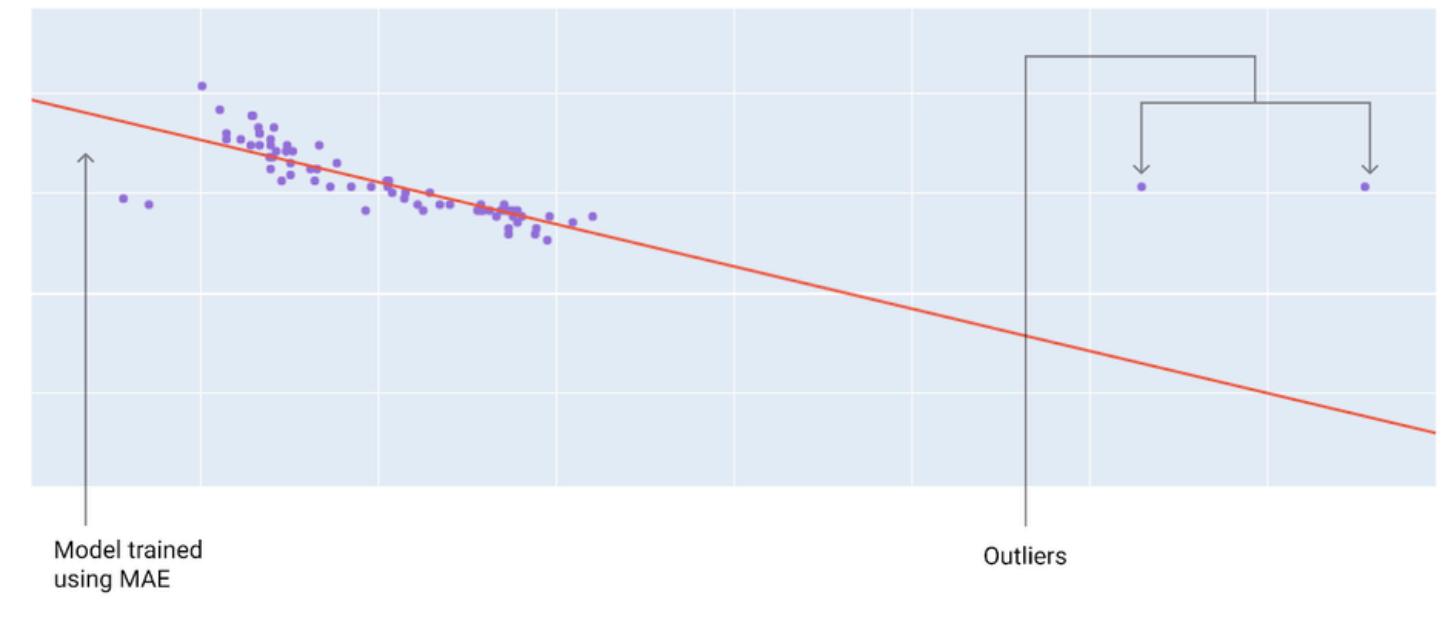
exemple :

Données (prédictions vs valeurs réelles) :

- Prédiction 1 : 20°C, Valeur réelle : 25°C
- Prédiction 2 : 22°C, Valeur réelle : 23°C
- Prédiction 3 : 30°C, Valeur réelle : 27°C

- **Erreurs absolues :**
 - **Prédiction 1 :** $|25 - 20| = 5$
 - **Prédiction 2 :** $|23 - 22| = 1$
 - **Prédiction 3 :** $|27 - 30| = 3$
- **MAE :**
$$\frac{5+1+3}{3} = \frac{9}{3} = 3$$

Fonction de perte



La MAE donne une idée plus simple et intuitive de l'erreur moyenne sans pénaliser autant les grandes erreurs que la MSE.

Deep Learning

Erreur Quadratique Moyenne Racine (RMSE)

exemple :

Données (prédictions vs valeurs réelles) :

- Prédiction 1 : 20°C, Valeur réelle : 25°C
- Prédiction 2 : 22°C, Valeur réelle : 23°C
- Prédiction 3 : 30°C, Valeur réelle : 27°C

- MSE : 11.67 (calculé précédemment)

- RMSE :

$$\sqrt{11.67} \approx 3.42$$

Fonction de perte



La RMSE est plus intuitive que la MSE car elle revient à l'unité d'origine des prédictions, ce qui facilite son interprétation.

Deep Learning

Pourcentage de validation

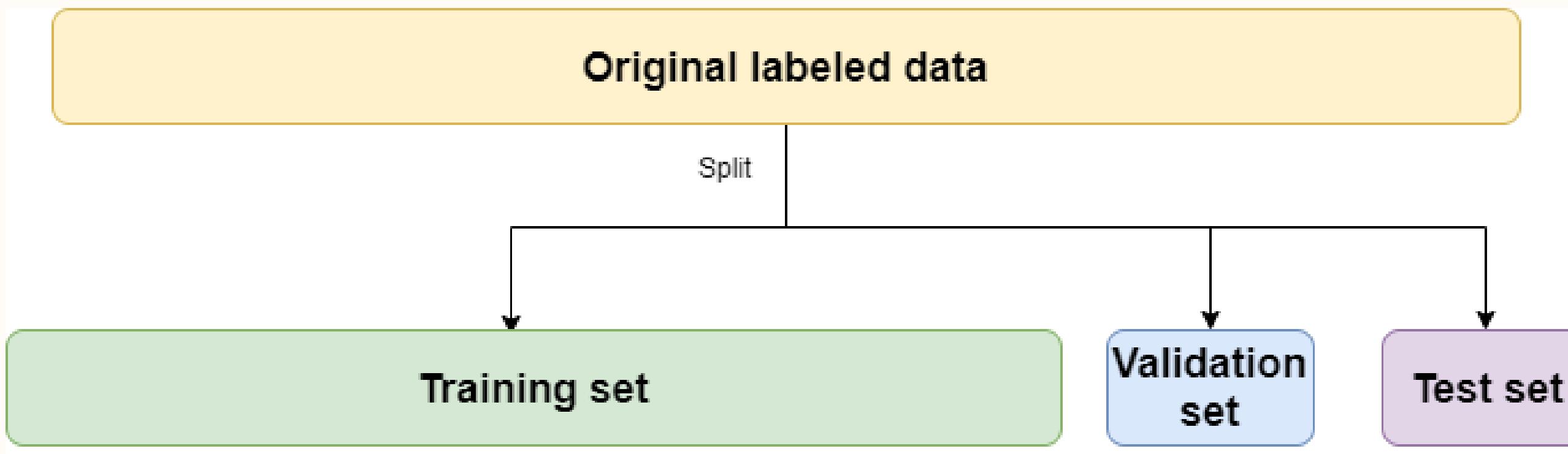
Test / entraînement :

- Ensemble d'entraînement : Pour entraîner le modèle.
- Ensemble de test : Pour évaluer les performances finales du modèle.

C'est une approche courante pour tous les modèles. Dans ce cas, les performances sont mesurées directement après l'entraînement sur l'ensemble de test.

Exemple sans validation :

- 80% des données pour l'entraînement,
- 20% des données pour le test.



Validation : L'ensemble de validation évalue la performance du modèle pendant l'entraînement pour éviter le surapprentissage et ajuster les hyperparamètres. Il sert de test intermédiaire pour vérifier la généralisation du modèle sur des données non vues.

Deep Learning

Pourcentage de validation

1. Préparation des Données : Vous avez 70,000 images de chiffres manuscrits (0 à 9).

2. Division des Données :

- Ensemble d'Entraînement : 50,000 images (71% des données).
- Ensemble de Validation : 10,000 images (14%).
- Ensemble de Test : 10,000 images (14%).

3. Entraînement : Couches de Neurones : Votre réseau comporte plusieurs couches de neurones (couche d'entrée, couches cachées, et couche de sortie). Pendant cette phase, les 50,000 images d'entraînement passent à travers le réseau.

- Propagation Avant : Chaque image est traitée par les neurones, qui appliquent des poids et des fonctions d'activation pour produire une prédiction (par exemple, un chiffre entre 0 et 9).
- Rétropropagation : Les erreurs (différence entre la prédiction et la vraie étiquette) sont calculées, et les poids des neurones sont ajustés pour minimiser cette erreur.
- Cela se répète sur plusieurs époques, permettant au modèle d'apprendre progressivement.

4. Validation : Après chaque époque d'entraînement, les performances du modèle sont évaluées à l'aide des 10,000 images de validation. Ces images passent également à travers les couches de neurones, mais elles ne sont pas utilisées pour ajuster les poids des neurones.

5. Test : Une fois l'entraînement et la validation terminés, le modèle est évalué sur les 10,000 images de test. Cela permet de mesurer la performance finale du modèle sur des données qu'il n'a jamais vues auparavant. Par exemple, le modèle pourrait obtenir une précision de 98 % sur cet ensemble, indiquant qu'il est efficace pour reconnaître les chiffres.

Deep Learning

Test / entraînement :

1. Préparation des Données : 70,000 images de chiffres manuscrits (0 à 9).
2. Division des Données :
 - Ensemble d'Entraînement : 50,000 images (71%).
 - Ensemble de Validation : 10,000 images (14%).
 - Ensemble de Test : 10,000 images (14%).
3. Entraînement : Utilisation des 50,000 images pour former le modèle en ajustant les poids en fonction des erreurs.
4. Validation : Évaluation du modèle sur les 10,000 images de validation après chaque époque pour ajuster les hyperparamètres et éviter le surapprentissage.
5. Test : Évaluation finale du modèle sur les 10,000 images de test pour mesurer la performance réelle (par exemple, 98% de précision).

Pourcentage de validation

0 1 2 3 4

5 6 7 8 9

Deep Learning

Surapprentissage

- Contexte : Vous créez un modèle de classification pour reconnaître des chiens et des chats à partir de 10,000 images.
- Entraînement : Vous utilisez 8,000 images pour entraîner le modèle et 2,000 images pour valider ses performances.
- Observation : Après 10 époques d'entraînement, la précision du modèle sur l'ensemble d'entraînement atteint 98 %. Cependant, la précision sur l'ensemble de validation stagne à 80 %.
- Interprétation : Bien que le modèle soit très bon sur les images d'entraînement, il ne généralise pas bien aux nouvelles images, ce qui indique qu'il a mémorisé les données d'entraînement (surapprentissage).
- Action : Vous décidez de réduire la complexité du modèle et d'ajouter une régularisation (comme le dropout) pour améliorer sa capacité à généraliser.

entraînement parfait



test moyen



Deep Learning

Sous-Apprentissage

- Contexte : Vous développez un modèle de régression pour prédire les prix des maisons à partir de plusieurs caractéristiques (superficie, nombre de chambres, etc.).
- Entraînement : Vous utilisez un petit ensemble de données contenant 1,000 maisons pour entraîner le modèle.
- Observation : Après plusieurs époques d'entraînement, la précision du modèle sur l'ensemble d'entraînement n'atteint que 60 %. La précision sur l'ensemble de validation est également faible, à environ 65 %.
- Interprétation : Le modèle est incapable d'apprendre les relations entre les caractéristiques des maisons et leurs prix. Cela signifie qu'il est trop simple pour capturer la complexité des données, ce qui indique un sous-apprentissage.
- Action : Pour remédier à cela, vous décidez d'augmenter la complexité du modèle, par exemple en ajoutant plus de neurones ou de couches, ou en utilisant un modèle plus sophistiqué.

entraînement moyen



test moyen



Deep Learning

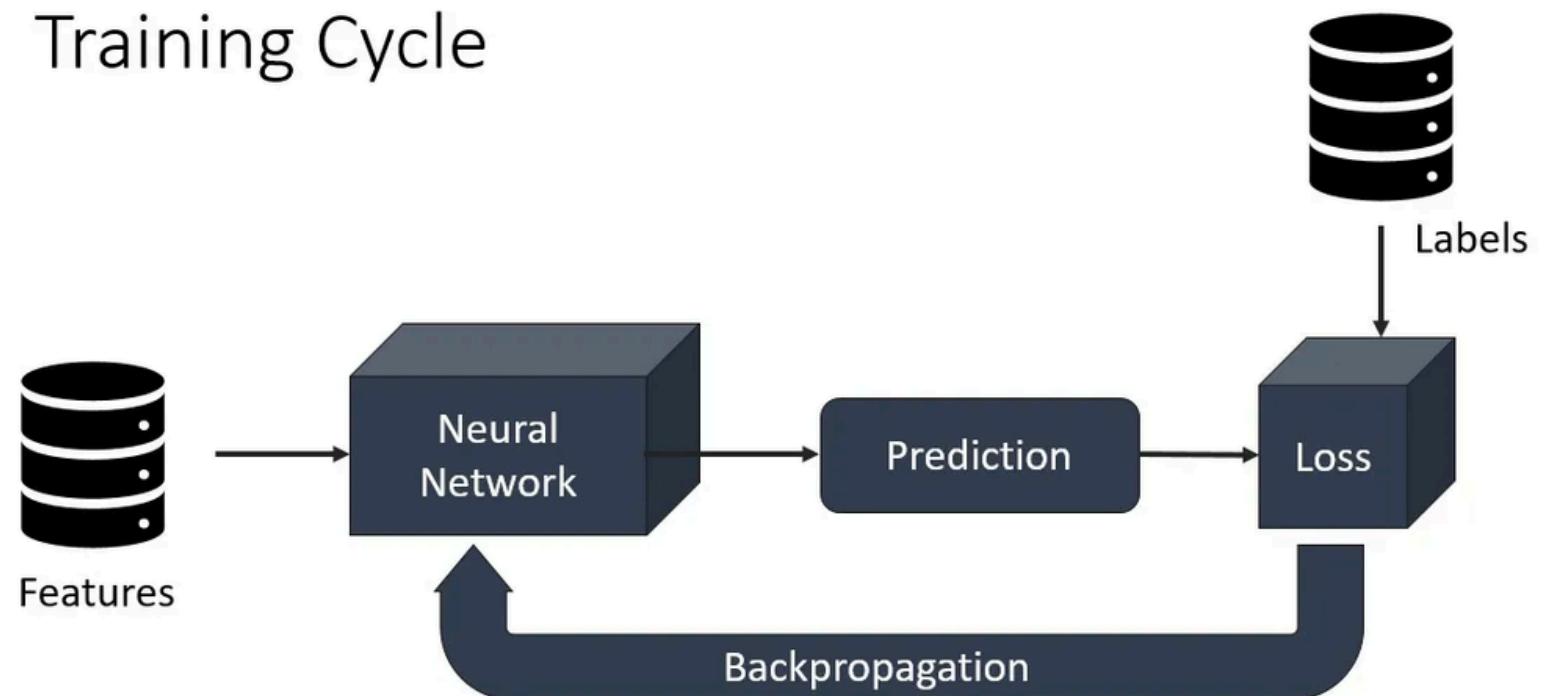
Epoque

Cela signifie que durant une époque, le modèle passe par toutes les données d'entraînement une fois et met à jour ses poids en fonction de l'erreur qu'il a commise.

Détails Importants :

- Processus : Lors d'une époque, le modèle fait des prédictions sur les données d'entraînement, calcule l'erreur (perte) et ajuste ses poids à l'aide d'une méthode d'optimisation (comme la descente de gradient).
- Nombre d'Époques : L'entraînement d'un modèle peut nécessiter plusieurs époques. Par exemple, vous pourriez entraîner un modèle sur 50, 100 ou même plus d'époques, en fonction de la complexité du modèle et de la taille de l'ensemble de données.
- Surveillance : À chaque époque, les performances du modèle (comme la précision ou la perte) sont souvent surveillées sur les ensembles de validation pour évaluer l'amélioration et détecter des problèmes comme le surapprentissage

Training Cycle

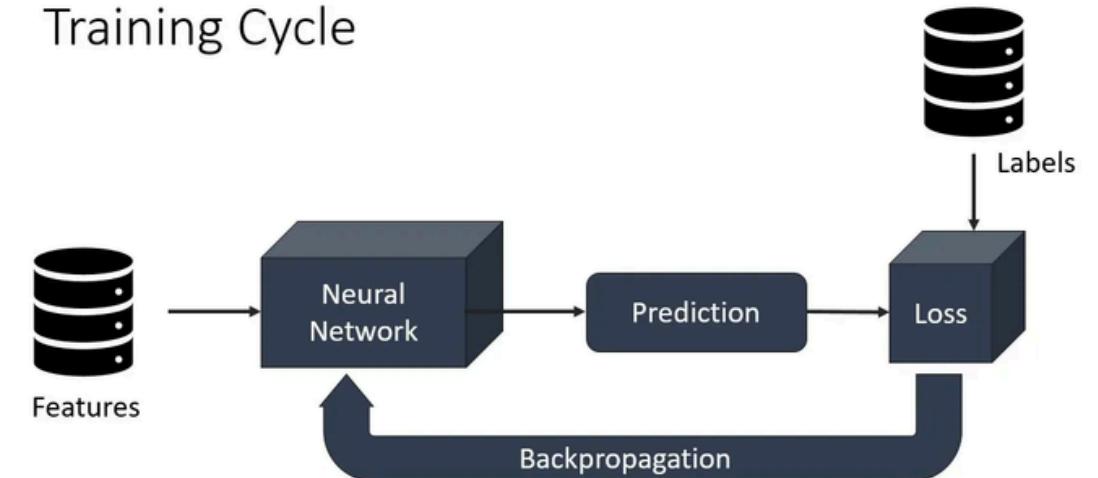


Deep Learning

- Ensemble de Données : Vous avez 60,000 images pour l'entraînement, et vous les divisez en mini-lots (batches) de 100 images.
- Entraînement sur une Époque :
- Époque 1 :
 - Le modèle traite les 60,000 images.
 - Pour chaque mini-lot de 100 images, il fait des prédictions, calcule la perte, et met à jour ses poids.
 - À la fin de l'époque, toutes les images ont été utilisées une fois pour l'apprentissage.
- Évaluation :
- À la fin de l'époque, vous évaluez la précision du modèle sur un ensemble de validation de 10,000 images. Supposons qu'il obtienne une précision de 85 %.
- Passage à l'Époque Suivante :
- Époque 2 : Vous recommencez le processus. Le modèle passe à nouveau par les 60,000 images, apprend de ses erreurs et ajuste ses poids.
- Répétition : Vous continuez ce processus pendant plusieurs époques (par exemple, 10, 20, ou plus), surveillant la précision sur l'ensemble de validation après chaque époque.
- Observation :
- Au fur et à mesure des époques, vous pouvez voir que la précision du modèle augmente, par exemple, atteignant 90 % à l'époque 5 et 95 % à l'époque 10.

Époque

Training Cycle



Deep Learning

Epoque

Processus d'Apprentissage avant le Test

1. Apprentissage :

- Lors de l'entraînement, le modèle passe plusieurs fois (époques) sur l'ensemble de données d'entraînement. Par exemple, une image peut être utilisée 20 fois au cours de 20 époques.
- Pendant chaque époque, le modèle fait des prédictions, calcule l'erreur et ajuste ses poids en conséquence.

2. Test :

- L'ensemble de test est généralement séparé de l'ensemble d'entraînement. Les images de l'ensemble de test ne sont pas utilisées pour l'entraînement.
- Par exemple, une autre image dans l'ensemble de test pourrait n'avoir été vue qu'une seule fois par le modèle lors de l'évaluation finale.

Couche et Neurones

1. Propagation Avant (Forward Propagation) :

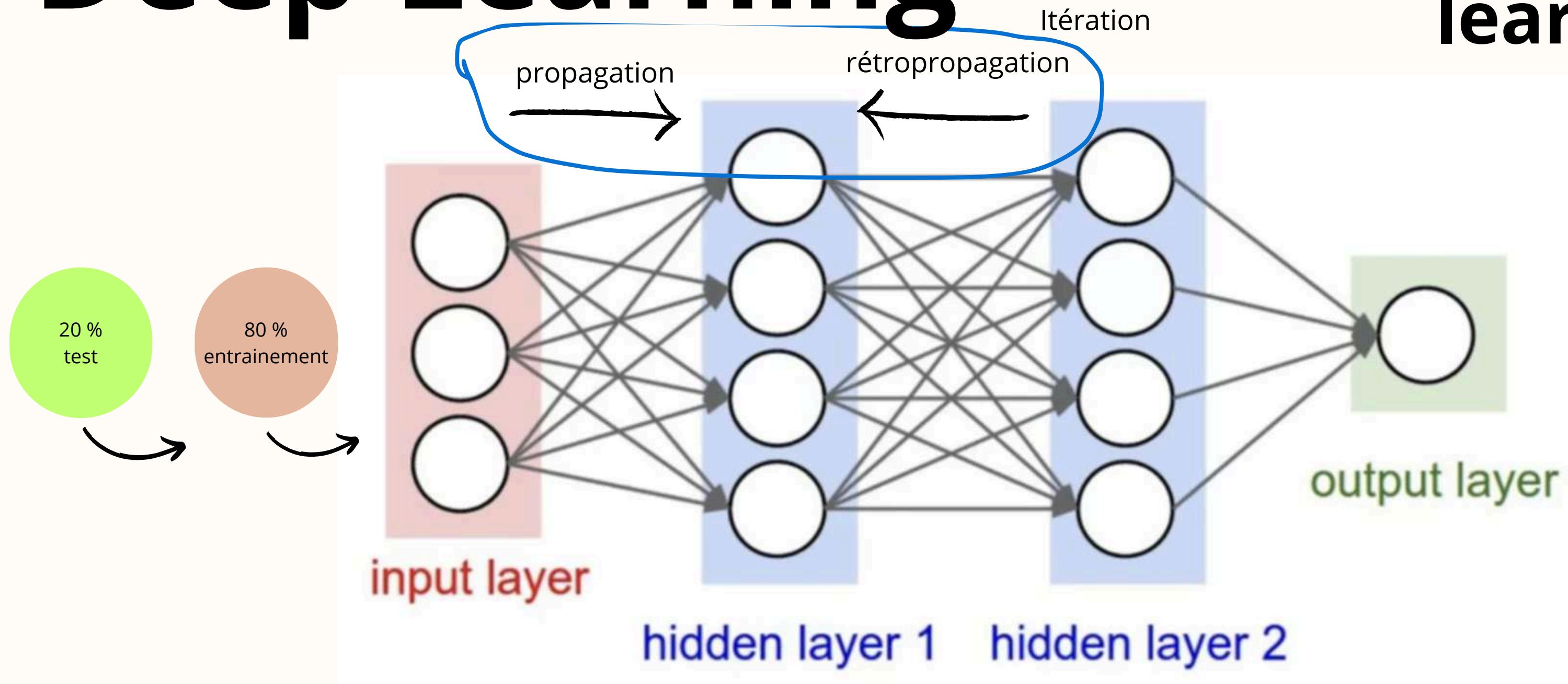
- Lors de l'entraînement, les données d'entrée passent à travers les différentes couches du réseau, des premières couches jusqu'à la dernière.
- Chaque neurone dans chaque couche effectue des calculs basés sur ses poids et envoie les résultats aux neurones de la couche suivante.

2. Rétropropagation (Backpropagation) :

- Après chaque passage (ou mini-lot) sur les données, le modèle utilise la rétropropagation pour ajuster les poids. Cela signifie que les erreurs sont propagées à l'envers, de la dernière couche à la première.
- Chaque neurone reçoit des informations sur son erreur et ajuste ses poids en conséquence, ce qui aide le modèle à s'améliorer au fil des époques.

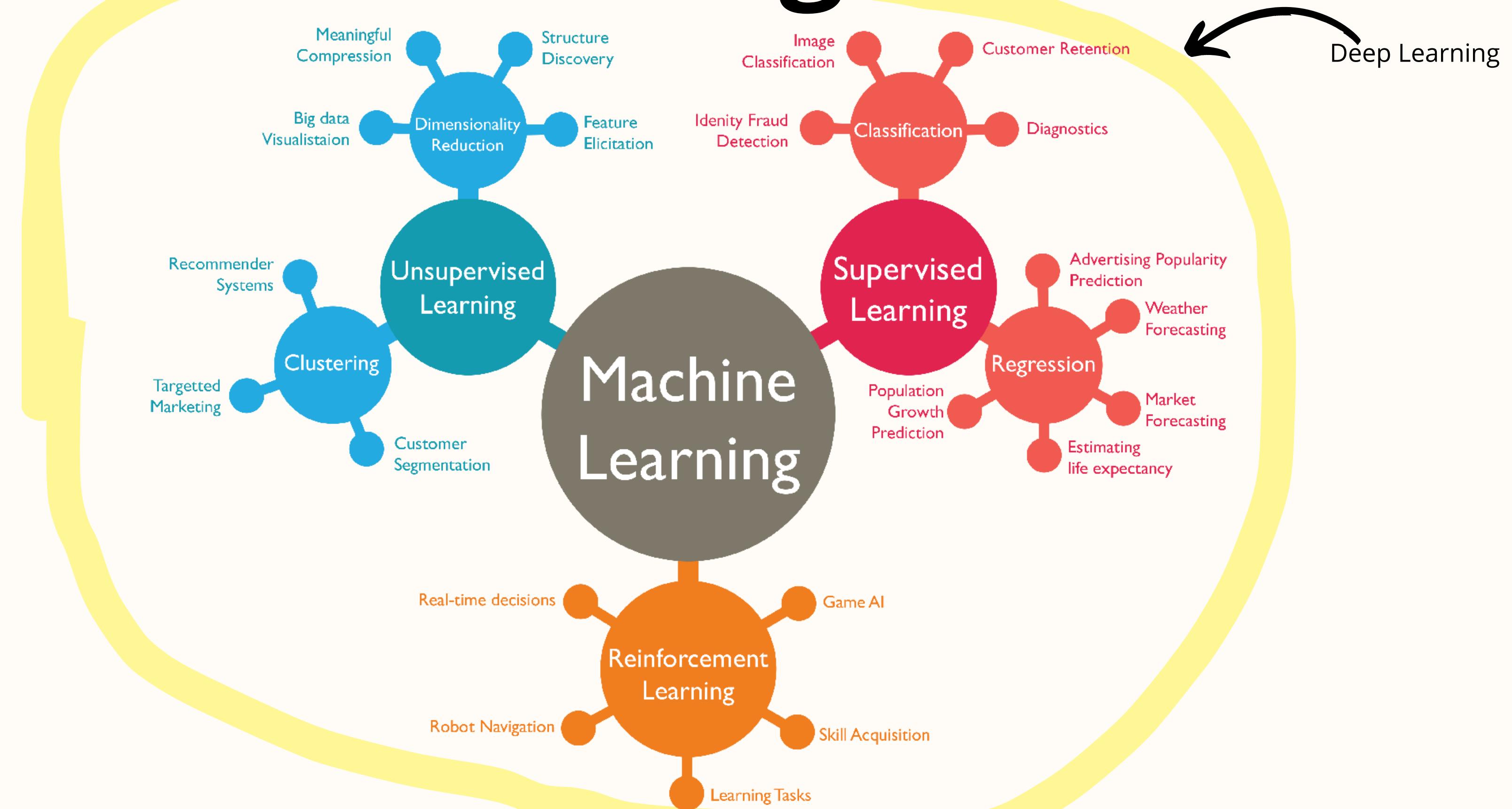
Deep Learning

Résumer de deep learning



- batch size
- pourcentage de validation
- fonction de perte
- fonction d'activation

Machine Learning



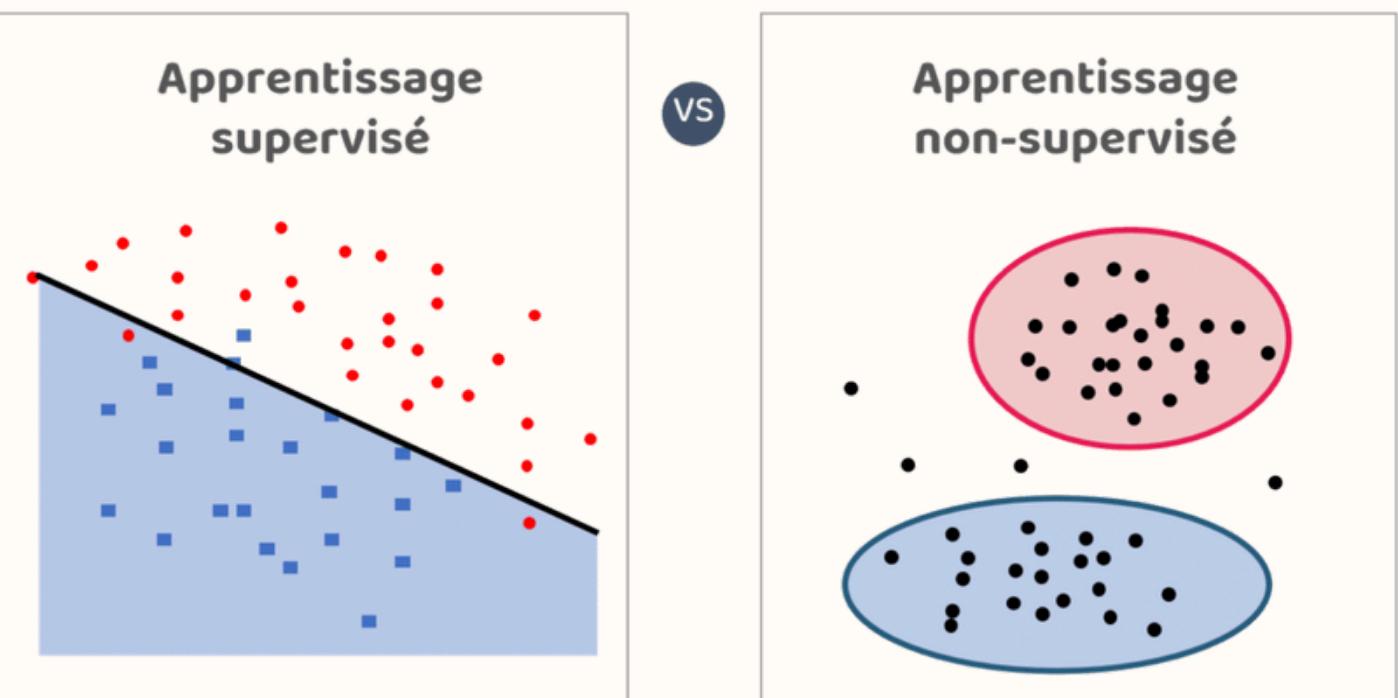
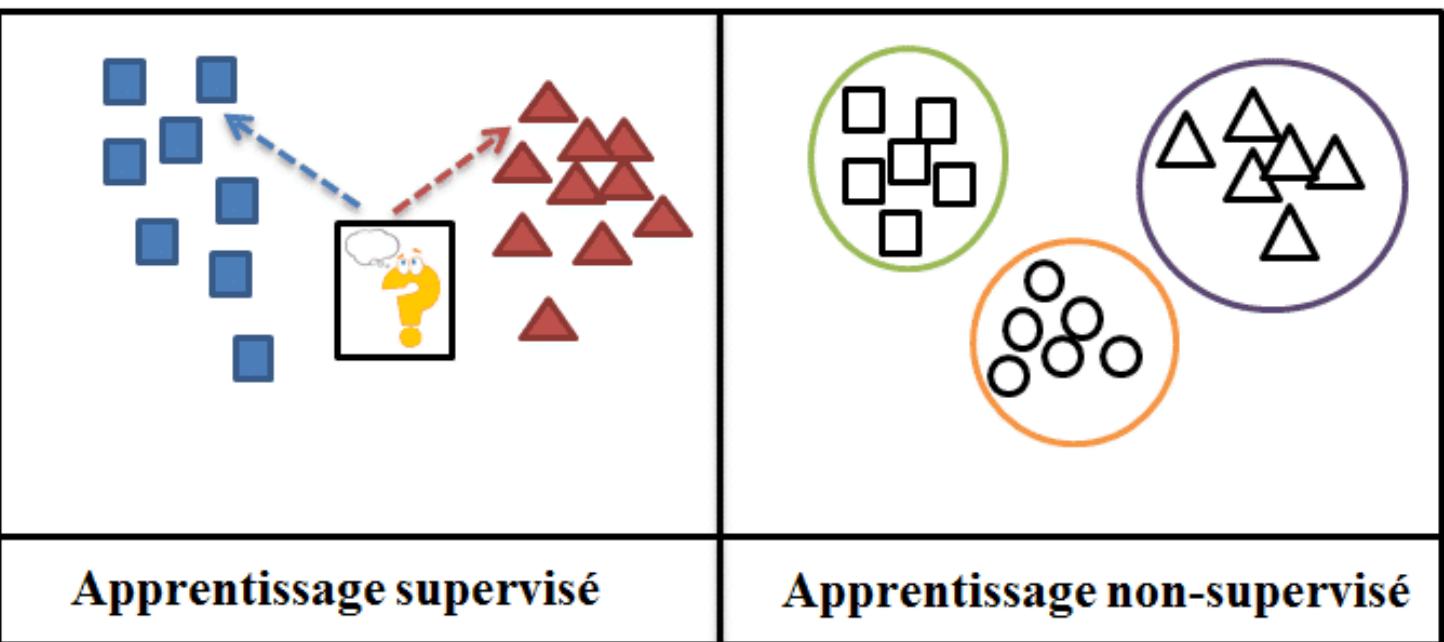
Machine Learning

1. Apprentissage Supervisé

- Définition : Dans l'apprentissage supervisé, le modèle est entraîné sur un ensemble de données étiquetées. Chaque exemple d'entraînement comprend une entrée (features) et une sortie correspondante (label).
- Objectif : Le but est de prédire la sortie à partir de nouvelles entrées.
- Exemples :
 - Classification : prédire si un e-mail est un spam ou non.
 - Régression : prédire le prix d'une maison en fonction de ses caractéristiques (surface, emplacement, etc.).

2. Apprentissage Non Supervisé

- Définition : Dans l'apprentissage non supervisé, le modèle est entraîné sur un ensemble de données sans étiquettes. L'algorithme doit trouver des structures ou des patterns sous-jacents dans les données.
- Objectif : Identifier des groupes, des clusters ou des caractéristiques dans les données.
- Exemples :
 - Clustering : regrouper des clients en fonction de leurs comportements d'achat.
 - Réduction de dimensionnalité : utiliser des techniques comme PCA pour simplifier des données complexes.



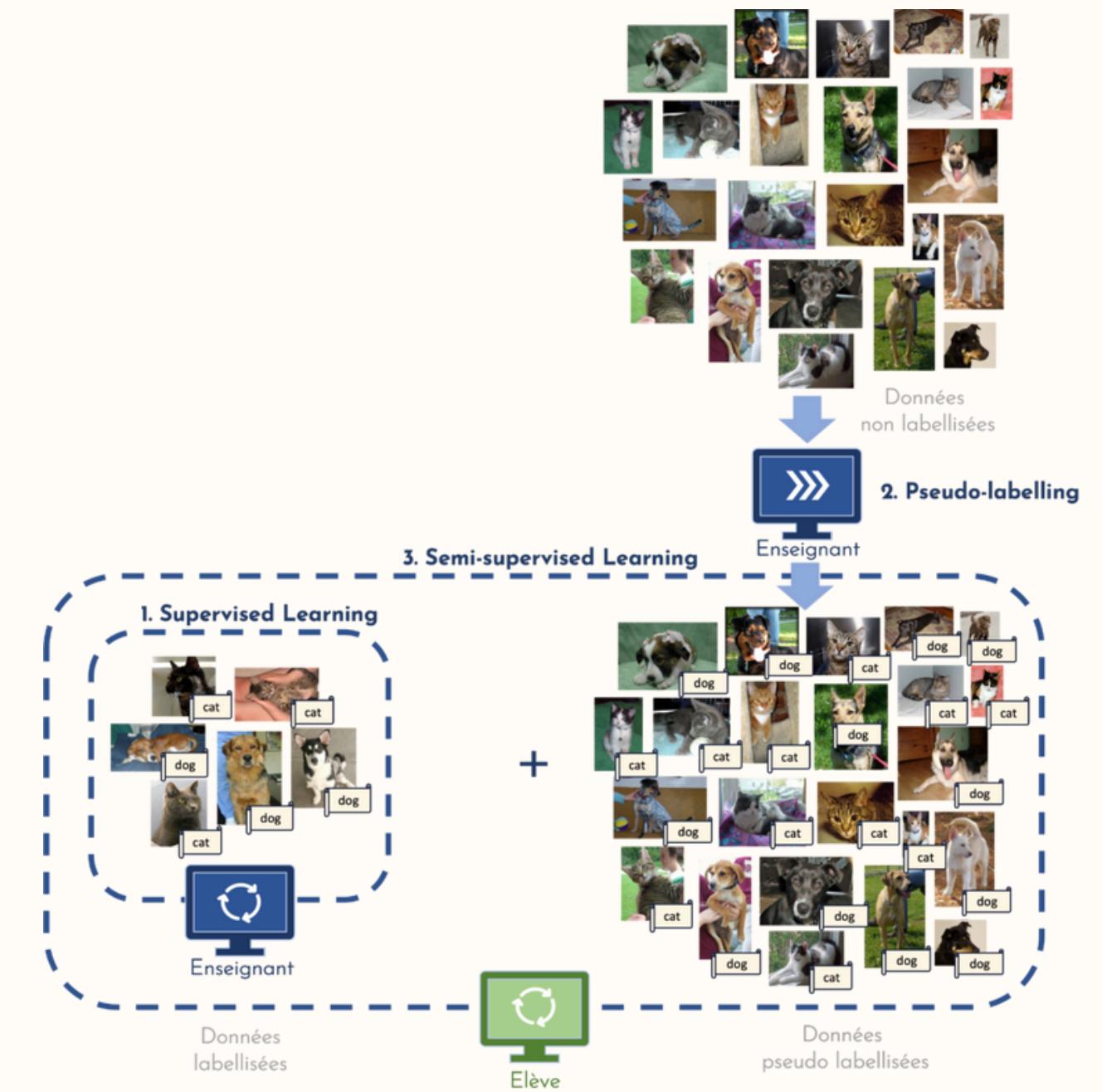
Machine Learning

3. Apprentissage Semi-Supervisé

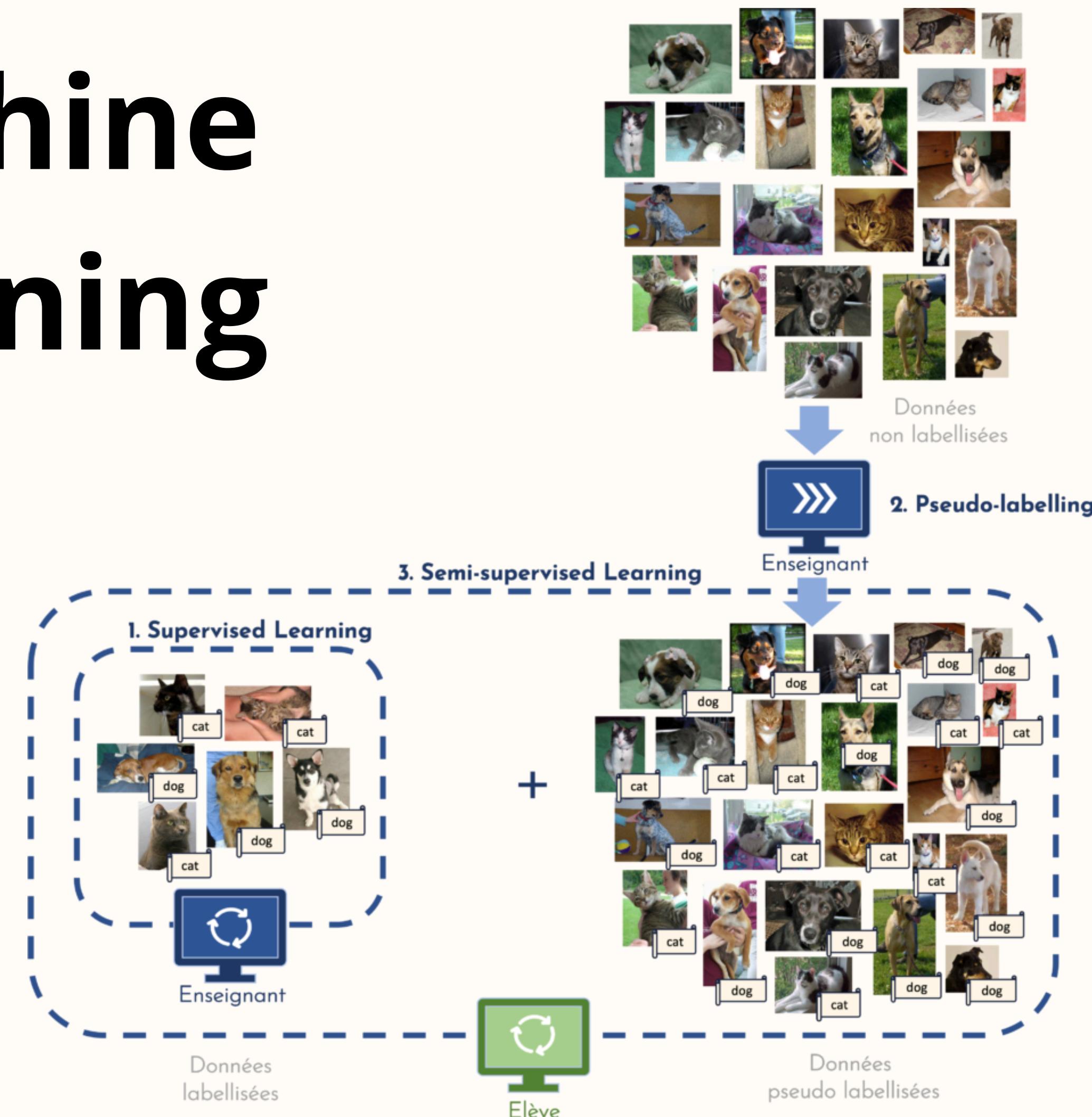
- Définition : L'apprentissage semi-supervisé utilise à la fois des données étiquetées et non étiquetées. Généralement, une petite quantité de données est étiquetée, tandis que la majorité est non étiquetée.
- Objectif : Améliorer l'apprentissage en utilisant les données non étiquetées pour mieux comprendre la structure des données tout en profitant des informations étiquetées.
- Exemples :
 - Classification d'images où un petit nombre d'images est étiqueté et le reste est non étiqueté. Le modèle utilise les images étiquetées pour apprendre et les images non étiquetées pour améliorer sa performance.

4. Apprentissage par Renforcement

- Définition : Dans l'apprentissage par renforcement, un agent apprend à prendre des décisions en interagissant avec un environnement. L'agent reçoit des récompenses ou des pénalités en fonction des actions qu'il prend.
- Objectif : Maximiser la somme des récompenses à long terme par l'apprentissage des meilleures actions à entreprendre dans chaque état.
- Exemples :
 - Un robot apprenant à naviguer dans un labyrinthe en recevant des récompenses pour atteindre la sortie.
 - Un agent jouant à un jeu vidéo, apprenant à accumuler des points en choisissant les bonnes actions.



Machine Learning



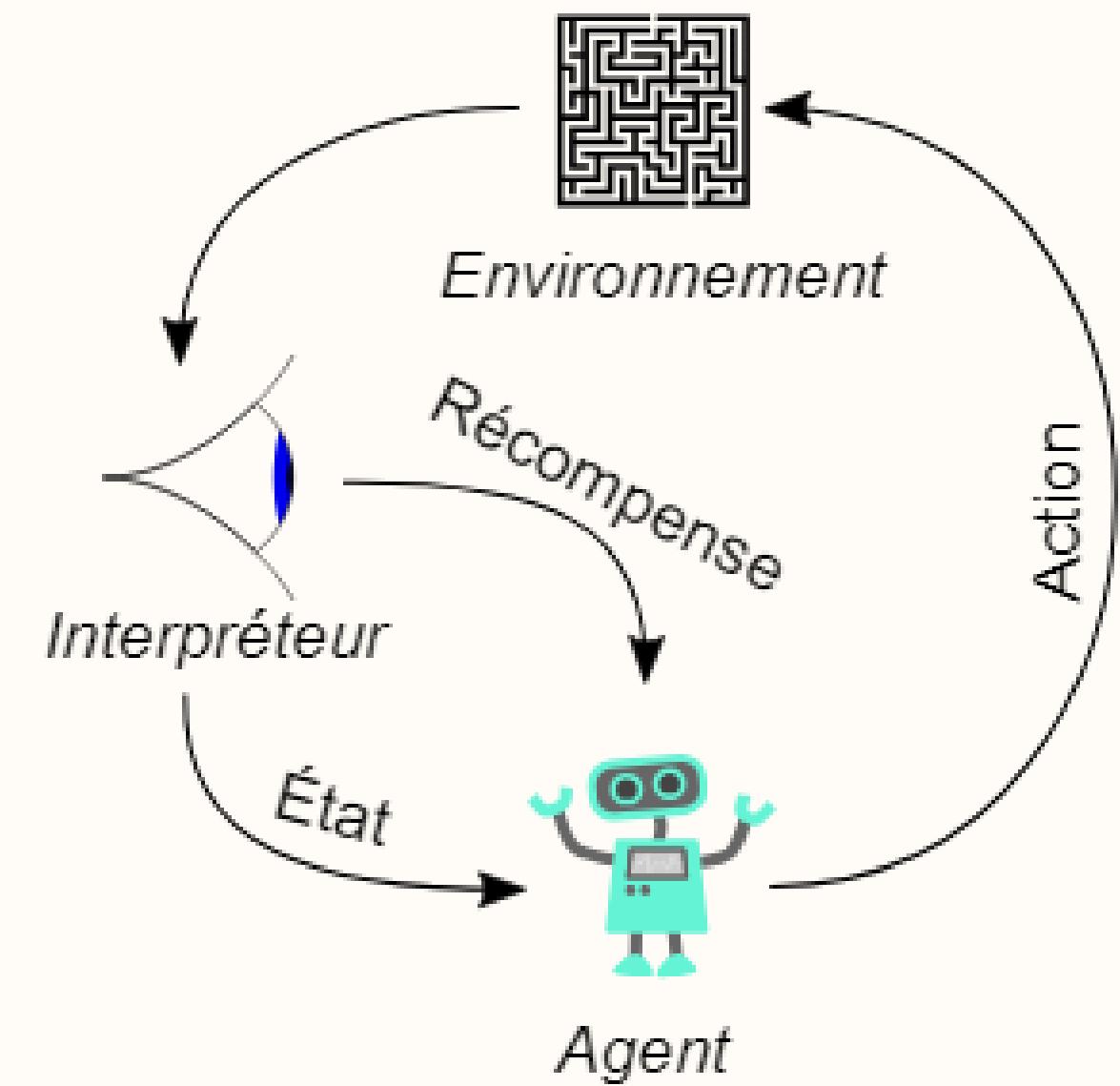
Machine Learning

Par renforcement

L'apprentissage par renforcement (Reinforcement Learning, RL) est un domaine du machine learning où un agent apprend à prendre des décisions en interagissant avec un environnement. Voici une explication simple de son fonctionnement :

Concepts Clés de l'Apprentissage par Renforcement

1. Agent : C'est l'entité qui apprend et prend des décisions. Par exemple, un robot, un programme de jeu, ou tout autre système autonome.
2. Environnement : C'est tout ce avec quoi l'agent interagit. Cela peut être un jeu, un simulateur physique, ou toute autre situation où des décisions doivent être prises.
3. État (State) : Une représentation de la situation actuelle de l'agent dans l'environnement. Par exemple, la position d'un robot ou l'état d'un jeu.
4. Action : Ce que l'agent peut faire à partir de son état actuel. Les actions modifient l'état de l'environnement.
5. Récompense (Reward) : Un retour d'information donné à l'agent après avoir effectué une action dans un état. La récompense peut être positive (bonus) ou négative (pénalité) et guide l'agent pour apprendre les actions qui mènent à de meilleurs résultats.



Machine Learning

Non supervisé

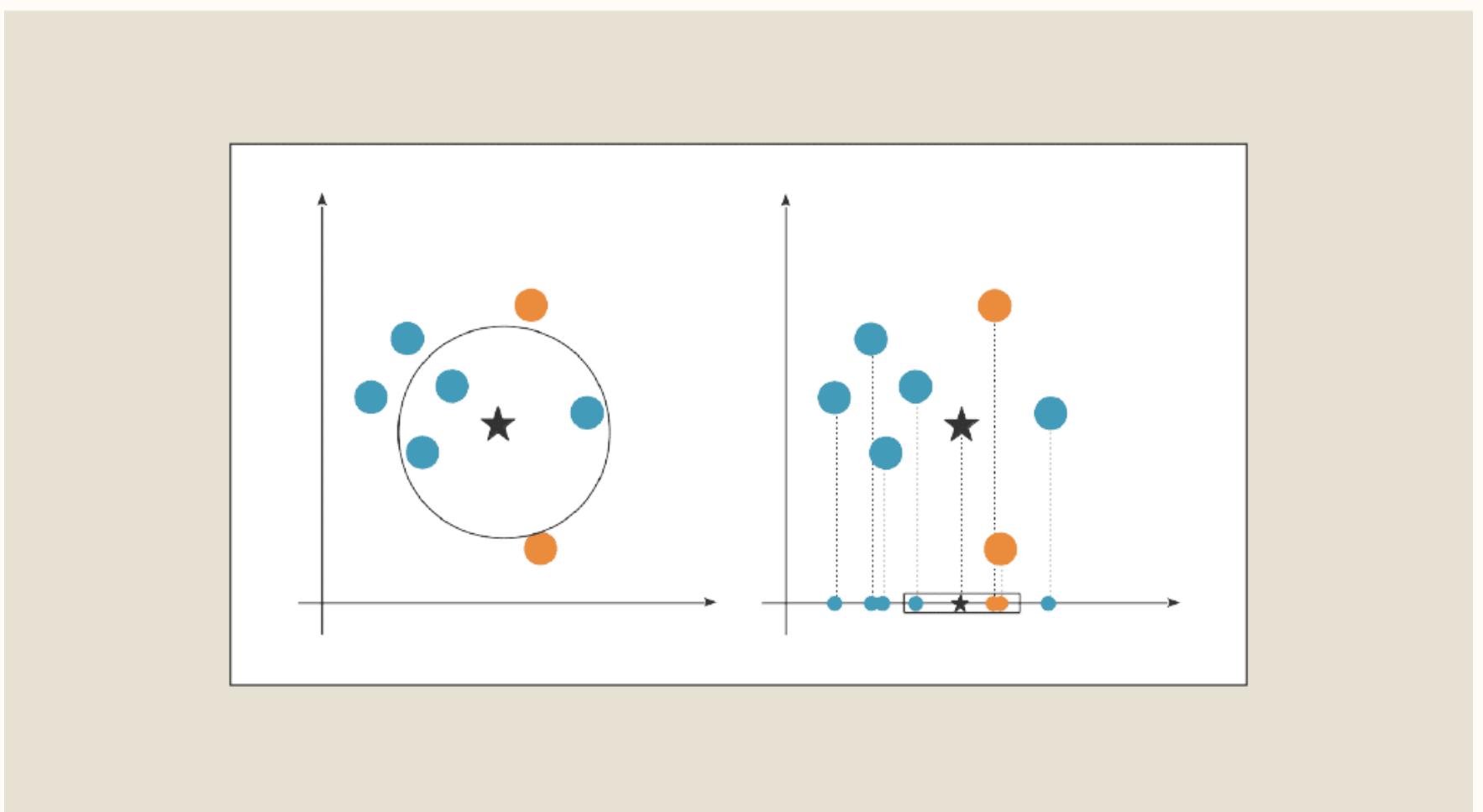
1. Réduction de Dimensionnalité

Définition

La réduction de dimensionnalité est le processus de réduire le nombre de variables (dimensions) dans un ensemble de données tout en préservant les caractéristiques essentielles de celles-ci.

Pourquoi utiliser la réduction de dimensionnalité ?

- Simplification : Rendre les données plus simples à visualiser et à interpréter.
- Performance : Améliorer la vitesse et la performance des algorithmes d'apprentissage.
- Éviter le surapprentissage : Réduire le bruit dans les données, ce qui peut aider à éviter le surapprentissage.
- Visualisation : Faciliter la visualisation des données en les projetant dans un espace à 2D ou 3D.



Machine Learning

Non supervisé

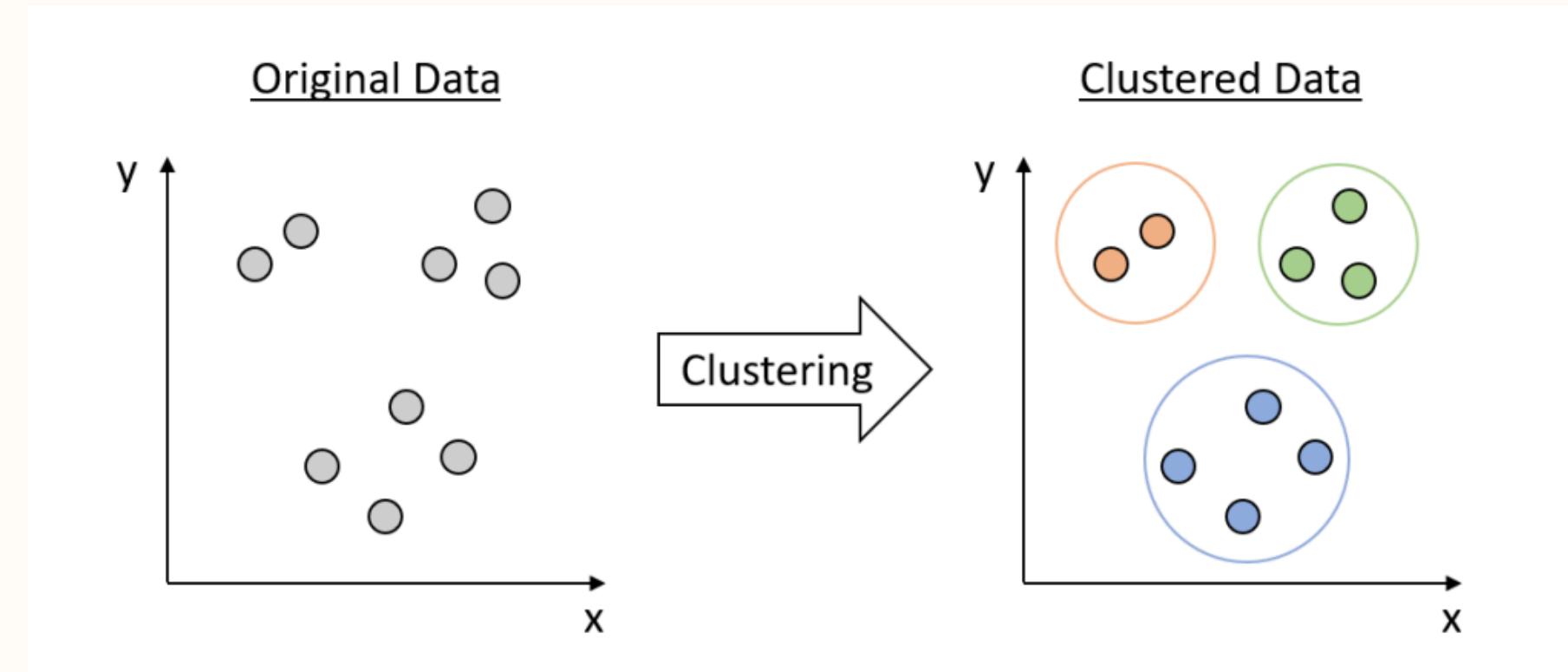
2. Clustering

Définition

Le clustering est une technique qui consiste à regrouper des données similaires ensemble dans des clusters. Les objets au sein d'un cluster sont plus similaires les uns aux autres qu'à ceux des autres clusters.

Pourquoi utiliser le clustering ?

- Segmentation : Identifier des groupes naturels dans les données pour des analyses ultérieures.
- Prétraitement : Faciliter l'analyse en réduisant la taille des ensembles de données.
- Exploration des données : Découvrir des structures cachées ou des anomalies.



Machine Learning

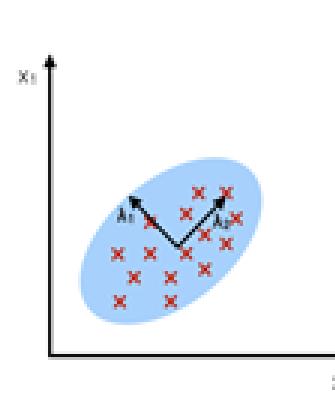
Non supervisé

Méthodes courantes

- Analyse en Composantes Principales (PCA) : Transforme les données en un nouvel ensemble de variables (composantes principales) qui sont des combinaisons linéaires des variables d'origine, tout en conservant la variance maximale.
- t-Distributed Stochastic Neighbor Embedding (t-SNE) : Technique non linéaire utilisée principalement pour la visualisation de données à haute dimensionnalité, en préservant les distances relatives entre les points proches.
- Linear Discriminant Analysis (LDA) : Utilisée principalement pour la classification, LDA peut également être utilisée pour la réduction de dimensionnalité en maximisant la séparation entre plusieurs classes.

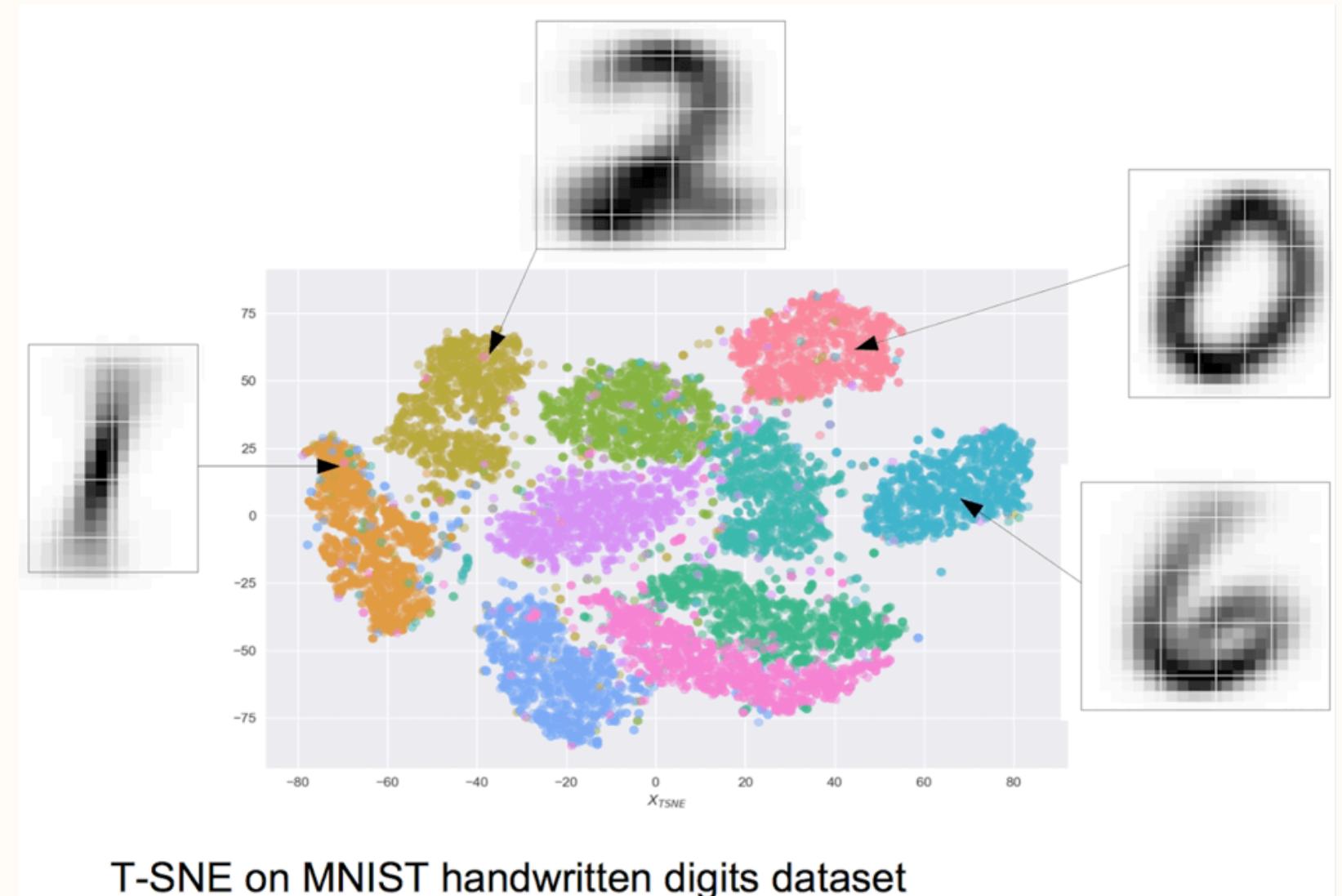
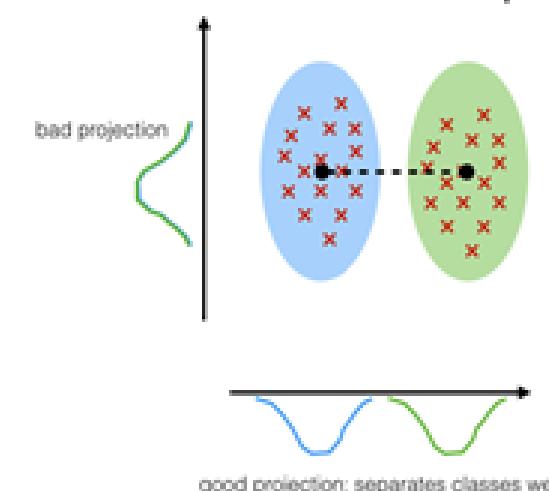
PCA:

component axes that maximize the variance



LDA:

maximizing the component axes for class-separation



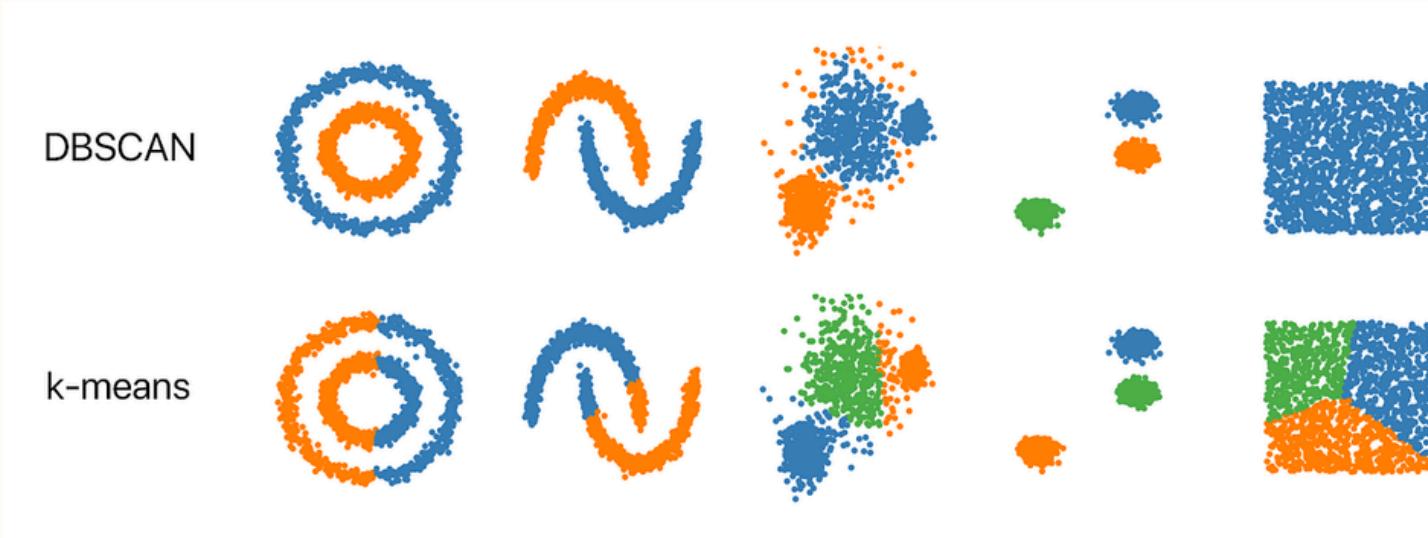
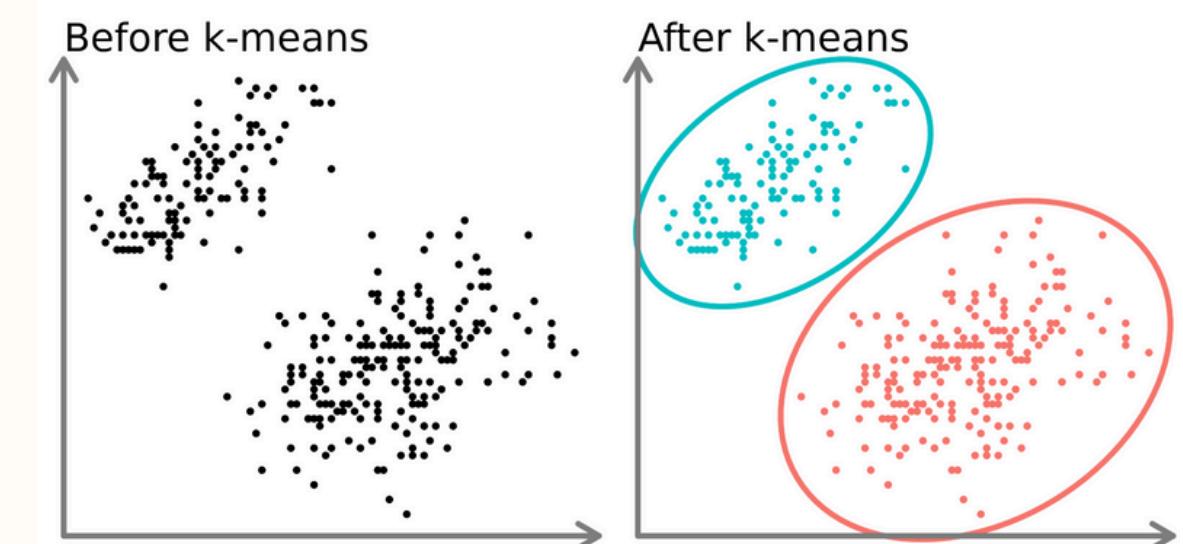
T-SNE on MNIST handwritten digits dataset

Machine Learning

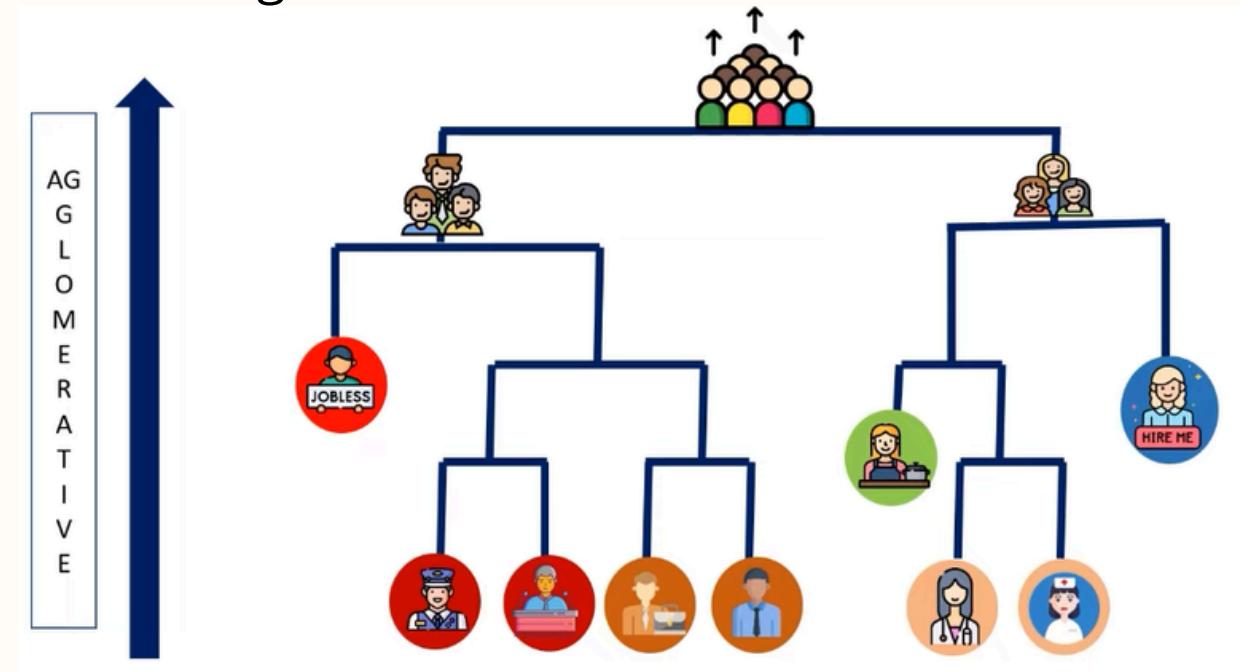
Non supervisé

Méthodes courantes

- K-Means : Une des méthodes les plus populaires qui divise les données en k clusters en minimisant la distance intra-cluster. L'algorithme nécessite de spécifier le nombre de clusters à l'avance.
- Hierarchical Clustering : Crée une hiérarchie de clusters à partir des données. Peut être agglomératif (fusion des clusters) ou divisif (division des clusters). Ne nécessite pas de spécifier le nombre de clusters à l'avance.
- DBSCAN (Density-Based Spatial Clustering of Applications with Noise) : Identifie des clusters basés sur la densité de points dans l'espace. Il est efficace pour détecter des clusters de forme arbitraire et peut gérer les bruits.



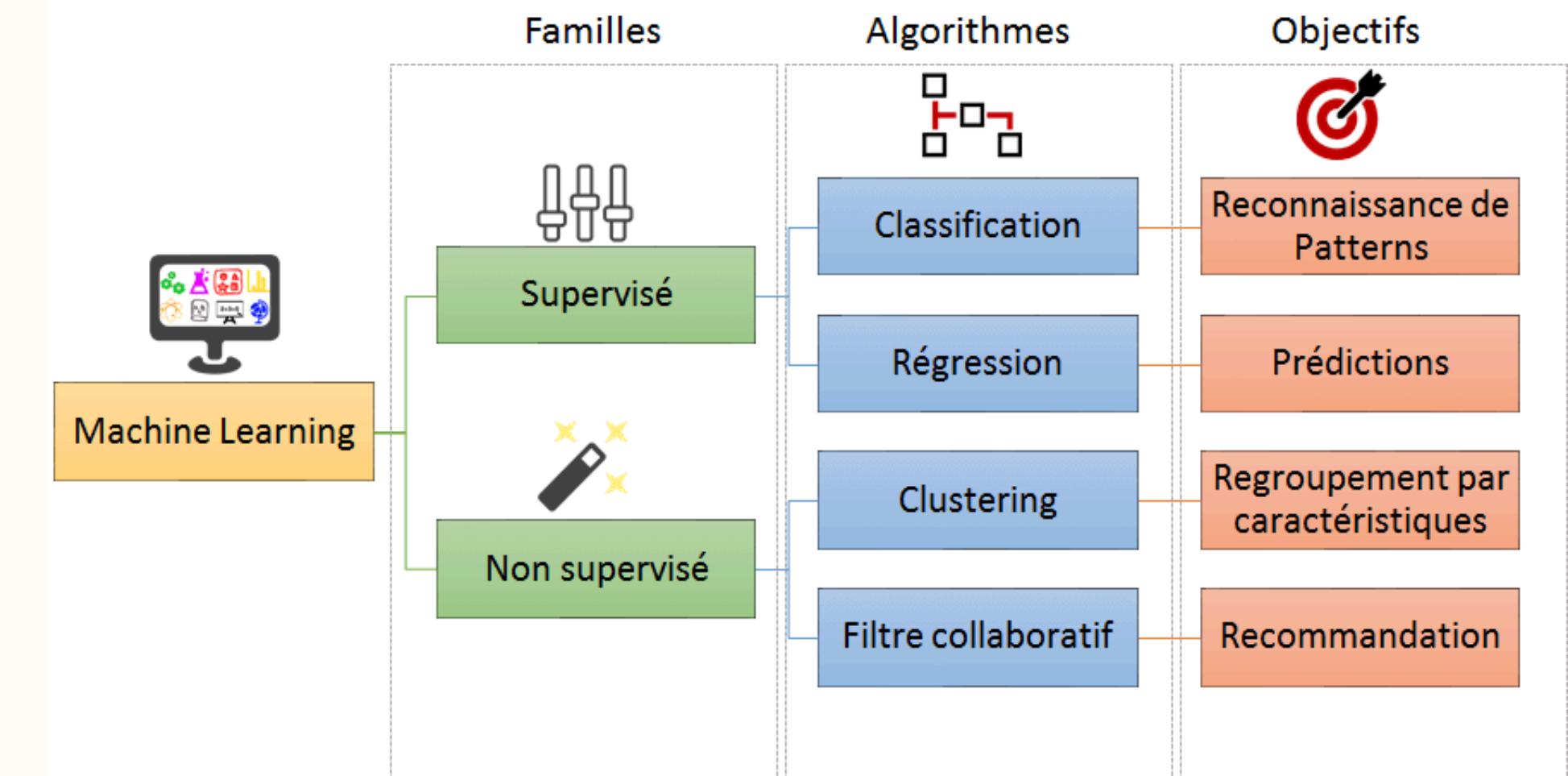
clustering hierarchical



Machine Learning

Conclusion

Le machine learning est un puissant outil qui permet de créer des modèles capables d'apprendre à partir de données. En combinant divers composants tels que des données, des modèles, des hyperparamètres et des fonctions de perte, il offre une flexibilité pour résoudre une large gamme de problèmes dans différents domaines, allant de la finance à la santé en passant par le marketing.



L'IA

Définition de l'Intelligence Artificielle (IA)

L'intelligence artificielle (IA) désigne un domaine de l'informatique qui vise à créer des systèmes capables d'effectuer des tâches qui nécessiteraient normalement l'intelligence humaine. Cela inclut des capacités telles que la compréhension du langage, la perception visuelle, la prise de décision, l'apprentissage et la résolution de problèmes.

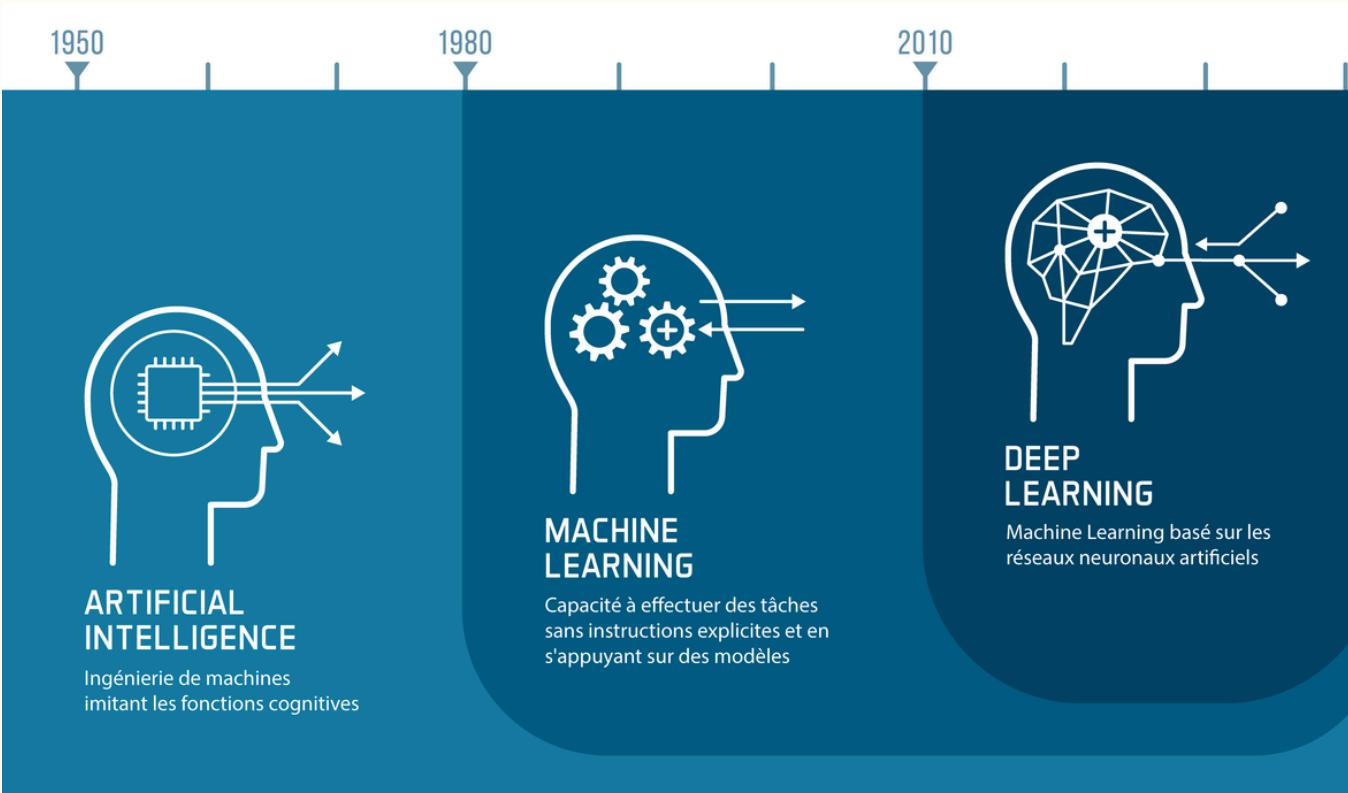
Composantes Clés de l'IA

1. Apprentissage Automatique (Machine Learning):

- Sous-domaine de l'IA qui permet aux machines d'apprendre à partir des données. Les algorithmes d'apprentissage automatique identifient des patterns dans les données et font des prédictions ou des décisions basées sur ces patterns.

2. Apprentissage Profond (Deep Learning):

- Sous-catégorie de l'apprentissage automatique qui utilise des réseaux de neurones profonds pour traiter des données complexes, comme les images et le langage naturel. C'est particulièrement efficace pour des tâches telles que la reconnaissance d'image et la traduction automatique.



L'IA

Types d'IA

1. IA Faible (Narrow AI):

- Systèmes conçus pour effectuer une tâche spécifique, comme la recommandation de films ou l'assistance vocale. Ils ne possèdent pas une véritable compréhension ou conscience.

2. IA Forte (General AI):

- Hypothétique forme d'IA qui pourrait comprendre, apprendre et appliquer des connaissances dans divers domaines de manière autonome, comme un humain. Ce type d'IA n'existe pas encore.



L'IA

Applications de l'IA

- Santé: Diagnostic médical assisté par IA, analyse d'images médicales, et personnalisation des traitements.
- Finance: Prévisions de marché, détection de fraudes, et automatisation des processus financiers.
- Transport: Véhicules autonomes, optimisation des itinéraires, et gestion du trafic.
- Marketing: Personnalisation des recommandations, analyse des comportements des consommateurs, et ciblage publicitaire.
- ...



L'IA

Conclusion

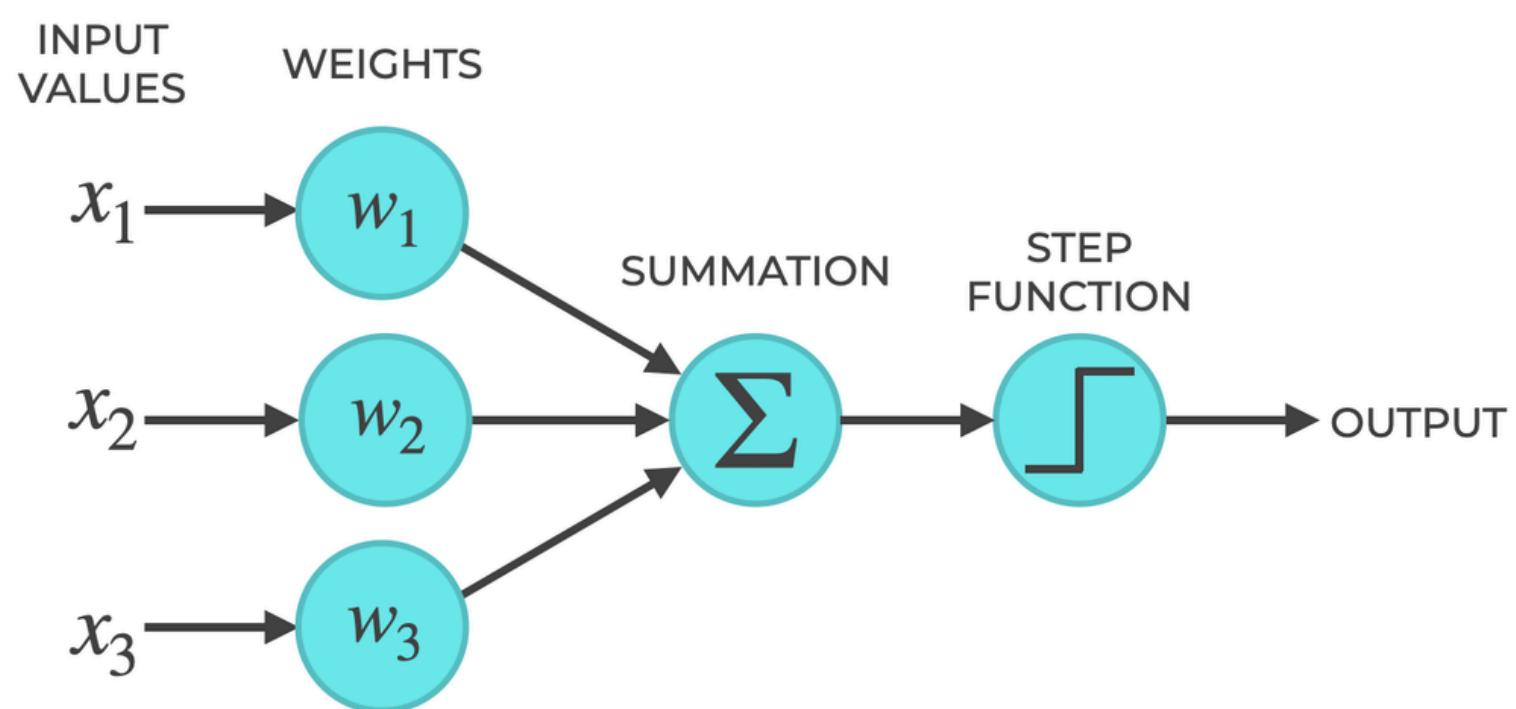
L'intelligence artificielle est un domaine en pleine expansion qui a le potentiel de transformer de nombreux aspects de notre vie quotidienne et des industries. En imitant certaines fonctions cognitives humaines, l'IA peut améliorer l'efficacité, réduire les erreurs et offrir de nouvelles solutions à des problèmes complexes.

Cependant, elle soulève également des questions éthiques et sociétales qui nécessitent une attention particulière à mesure que la technologie continue d'évoluer.



<https://youtu.be/ZP7T6WAK3Ow>

PERCEPTRONS, EXPLAINED



Formats à ajouter :

- Bruits def
- mesure de perf avec la précision, rappel, f1 score
- dummy classifier
- différence de régression, gradient ou classique
- Différent models
- support vector machine
- bias
- Perceptron

faire mindmap