INTRODUCTION À LA SCIENCE DES DONNÉES

Apprentissage Supervisé

Cadre Formel & Premiers Modèles

Victor Bertret

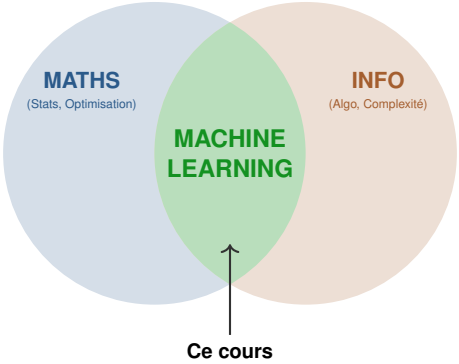
Docteur en Mathématiques

Ingénieur IA chez **Purecontrol**

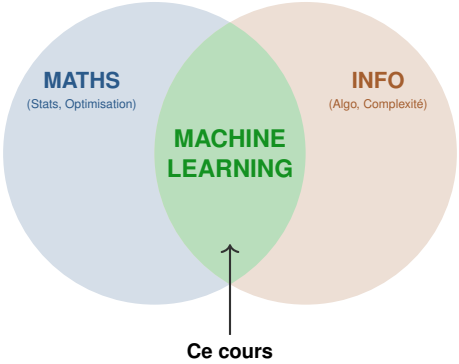
19 janvier 2026

Sommaire

Introduction : Où sommes-nous ?



Introduction : Où sommes-nous ?



*"Transformer votre culture de l'**Optimisation** (Théorie)*
*en culture de la **Généralisation** (Pratique)."*

Introduction : Le but du jeu

? Question

*"Si vous apprenez **par cœur** tous les exercices corrigés du TD...*

réussirez-vous l'examen si je change simplement les chiffres ?"

Introduction : Le but du jeu

❓ Question

*"Si vous apprenez **par cœur** tous les exercices corrigés du TD...*

réussirez-vous l'examen si je change simplement les chiffres ?"

NON.

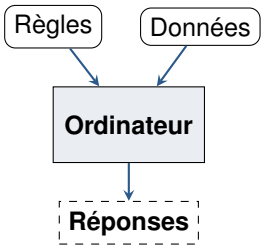
⚠ **Mémoriser ≠ Apprendre**

Apprendre = Généraliser

Le Changement de Paradigme

"En programmation classique (ex : Python, C++), vous écrivez les règles.
Mais comment écririez-vous les règles pour reconnaître un chat dans une image ?"

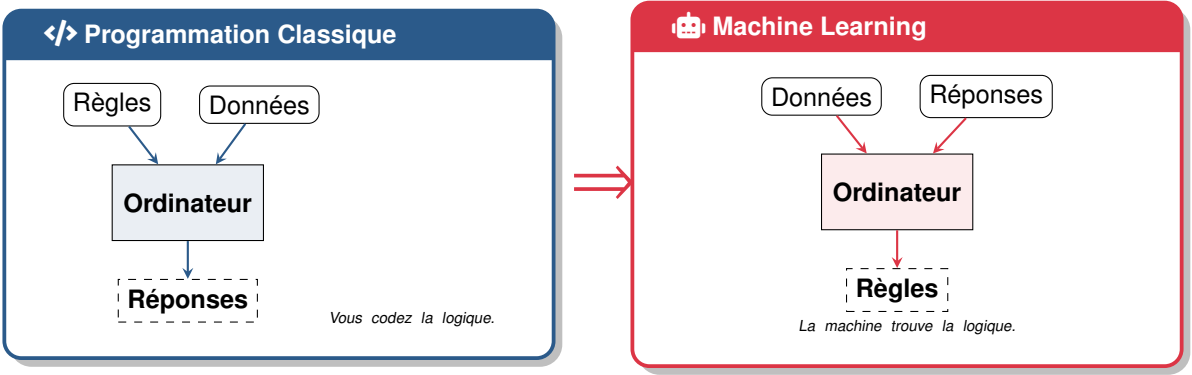
</> Programmation Classique



Vous codez la logique.

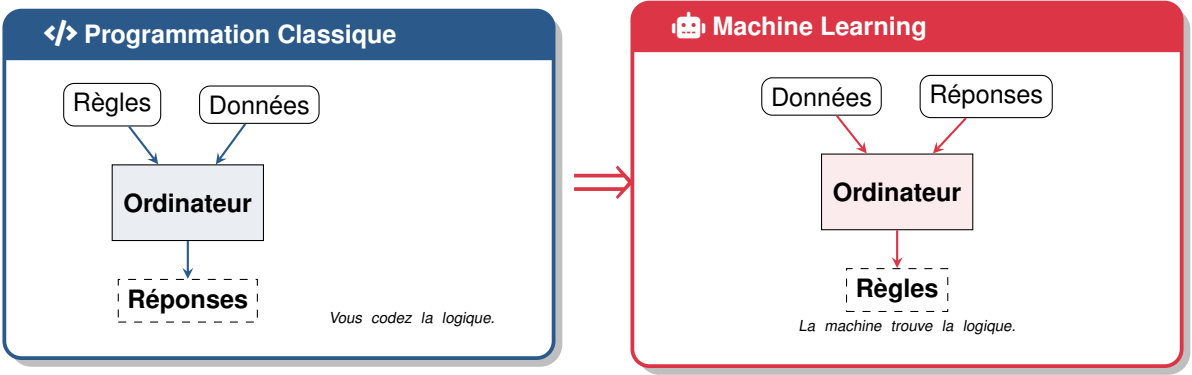
Le Changement de Paradigme

"En programmation classique (ex : Python, C++), vous écrivez les règles.
Mais comment écririez-vous les règles pour reconnaître un chat dans une image ?"



Le Changement de Paradigme

"En programmation classique (ex : Python, C++), vous écrivez les règles.
Mais comment écririez-vous les règles pour reconnaître un chat dans une image ?"



Conséquence : Le carburant n'est plus le code, mais la **Donnée** et la **Réponse**.
C'est la nature de ce carburant qui définit le type d'apprentissage (Slide suivante).

La Carte du Monde : Les 3 Paradigmes

"Il existe des milliers d'algorithmes. Pour les classer, une seule question :
Quelles données donnez-vous à la machine ?"

Supervisé

Données :
(X, Y)
(Entrée + **Réponse**)

"Le Professeur corrige les copies."

CE COURS

Non-Supervisé

Données :
(X)
(Entrée **seule**)

"L'enfant trie ses billes par couleur."

→ *Voir autre module*

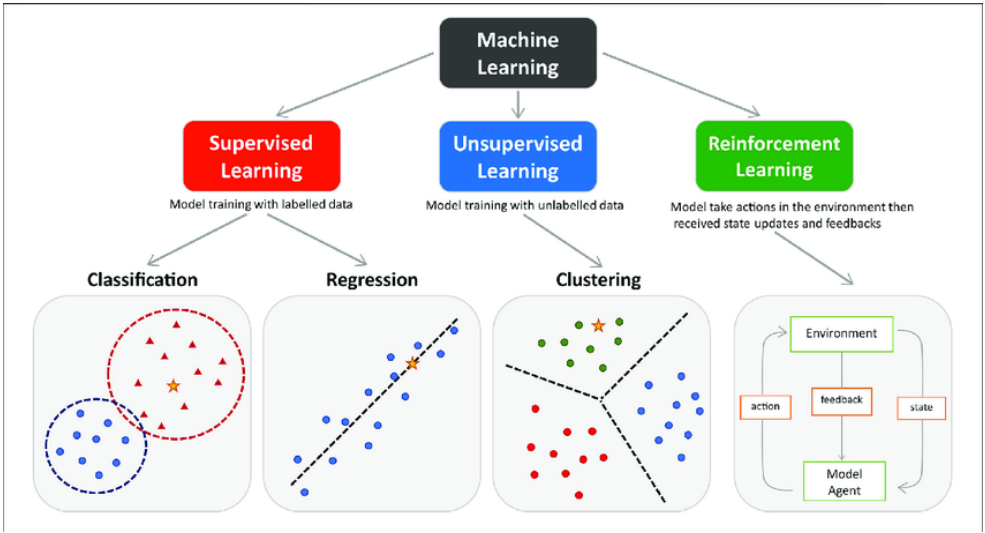
Renforcement

Données :
(A, R)
(Action/Récompense)

"Le chien et le sucre."

Hors programme

La Carte du Monde : Les 3 Paradigmes



L'Intuition : De la Donnée à la Règle

Ce qu'on observe (S)

Tasty

Tasty

Not Tasty

Exemple : Papayes (Bonnes vs Mauvaises)

$\Rightarrow$

Ce qu'on cherche (h)

Une règle de séparation claire

L'Intuition : De la Donnée à la Règle

Ce qu'on observe (S)

Tasty

Tasty

Not Tasty

Exemple : Papayes (Bonnes vs Mauvaises)

$\Rightarrow$

Ce qu'on cherche (h)

{Tasty/Not tasty}

Une règle de séparation claire

Le but du jeu : Trouver la frontière (la courbe) qui sépare le mieux les points, pour pouvoir classer les futurs points inconnus.

Formalisation : Le Dictionnaire Mathématique

"Pour que l'ordinateur apprenne, il faut traduire la réalité en vecteurs."

Monde Réel


Tasty


Tasty


Not Tasty

1. **Objet :** Une Papaye.
2. **Caractéristiques :**
   - Couleur (Verte → Orange)
   - Mollesse (Dure → Molle)
3. **Verdict :** Mangeable ?

Maths →

$\mathbf{X}^1$ Monde Formel

1. L'Entrée (Features) : $\mathcal{X} \subseteq \mathbb{R}^{n_X}$

$\mathbf{x} \in \mathcal{X}$ (Ici $n_X = 2$).

Ex : $\mathbf{x} = \begin{pmatrix} 0.8 \\ 0.2 \end{pmatrix}$ *(Très orange, dur)*

---

2. La Sortie (Target) : $\mathcal{Y} \subseteq \mathbb{R}^{n_Y}$

Ici $n_Y = 1$ (Scalaire).

$y \in \mathcal{Y} = \{0, 1\}$ (Classification).

---

3. L'Expérience : S

Ensemble de n paires observées :

$$S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\} \sim \mathcal{D}$$

Formalisation : Le Dictionnaire Mathématique

"Pour que l'ordinateur apprenne, il faut traduire la réalité en vecteurs."

Monde Réel


Tasty


Tasty


Not Tasty

1. **Objet :** Une Papaye.
2. **Caractéristiques :**
   - Couleur (Verte → Orange)
   - Mollesse (Dure → Molle)
3. **Verdict :** Mangeable ?

Maths

X^1 Monde Formel

1. L'Entrée (Features) : $\mathcal{X} \subseteq \mathbb{R}^{n_X}$

$\mathbf{x} \in \mathcal{X}$ (Ici $n_X = 2$).

Ex : $\mathbf{x} = \begin{pmatrix} 0.8 \\ 0.2 \end{pmatrix}$ *(Très orange, dur)*

---

2. La Sortie (Target) : $\mathcal{Y} \subseteq \mathbb{R}^{n_Y}$

Ici $n_Y = 1$ (Scalaire).

$y \in \mathcal{Y} = \{0, 1\}$ (Classification).

---

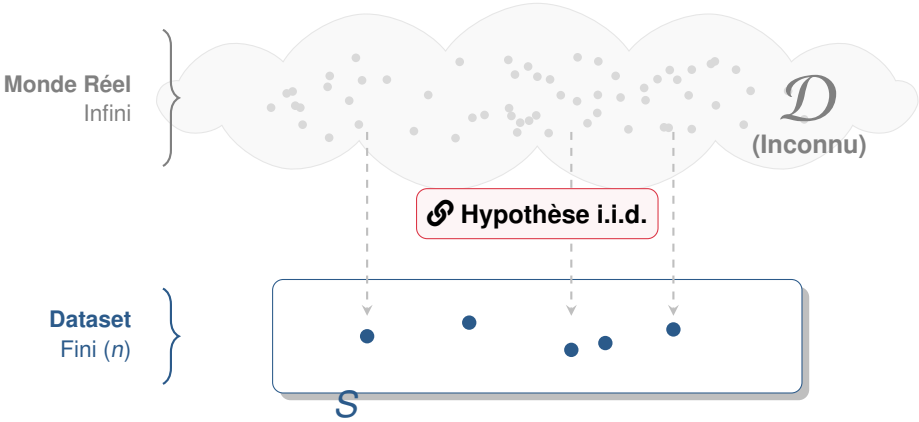
3. L'Expérience : S

Ensemble de n paires observées :

$$S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\} \sim \mathcal{D}$$

Mission : Trouver h tel que $h(\mathbf{x}) \approx y$ pour tout nouveau $\mathbf{x}$.

L'Origine de la Donnée : L'Échantillonnage



Le Pari : Si l'échantillonnage est bon (**i.i.d**), alors comprendre S permet de comprendre $\mathcal{D}$.

La Grande Bifurcation : Régression ou Classification ?

Question : Quelle est la nature du problème ?
Tout dépend de la structure mathématique de l'espace de sortie $\mathcal{Y}$.

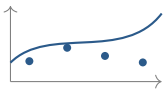
Régression

La question : "Combien ?"

Mathématiquement :

- $\mathcal{Y} \subseteq \mathbb{R}$ (Continu)
- Il y a une notion d'**ordre** et de **distance**.

Ex : Prédire un prix, une température, une durée.



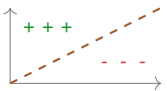
Classification

La question : "Lequel ?"

Mathématiquement :

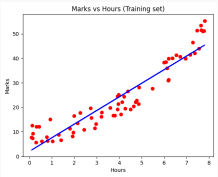
- $\mathcal{Y} = \{0, 1, \ldots, K\}$ (Discret)
- Pas de notion de distance entre les classes (chat $\neq$ 0.5 chien).

Ex : Spam, Malade/Sain, Chiffre manuscrit.

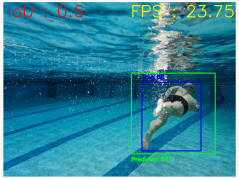


Universalité : Le même formalisme partout

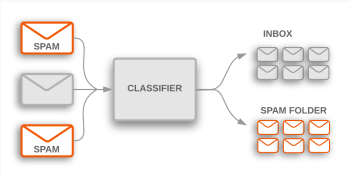
Que ce soit une papaye, un mail ou une image, c'est toujours un couple $(\mathbf{x}, y)$.



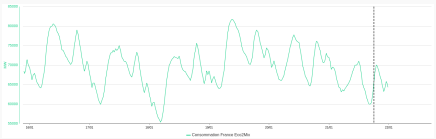
Régression ($y \in \mathbb{R}$)
Prédiction de notes



Vision ($y \in$ **Box**)
Détection d'objets



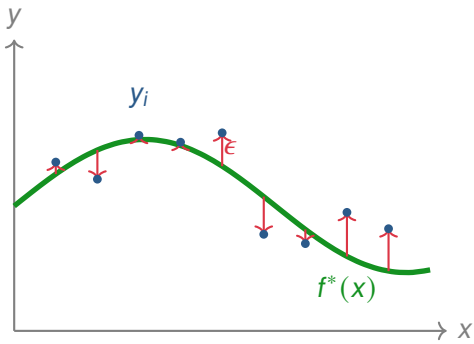
Classification ($y \in \{0, 1\}$)
Spam vs Non-Spam



Séries Tempo. ($y \in \mathbb{R}^T$)
Conso électrique

L'Hypothèse Fondamentale : Le Signal et le Bruit

"Si je mesure deux fois la même papaye, aurai-je le même taux de sucre ?"



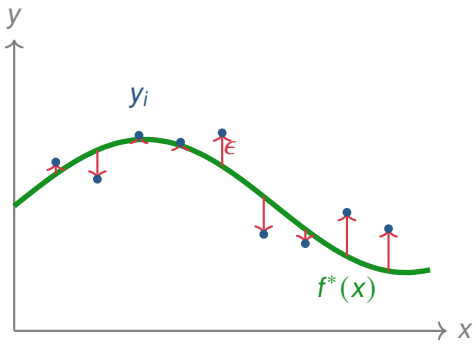
Le Modèle du Monde

$$y = f^*(x) + \epsilon$$

- **Signal (f^*) :** La loi physique (Inconnue).
- **Bruit (ϵ) :**
  1. Erreur de mesure.
  2. **Info manquante.** *(Variables cachées)*

L'Hypothèse Fondamentale : Le Signal et le Bruit

"Si je mesure deux fois la même papaye, aurai-je le même taux de sucre ?"



Le Modèle du Monde

$$y = f^*(x) + \epsilon$$

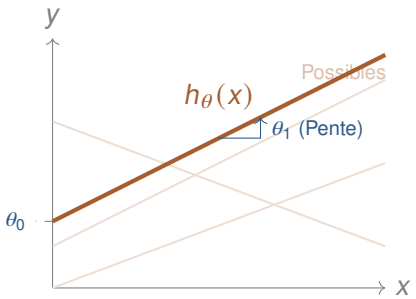
- **Signal (f^*) :** La loi physique (Inconnue).
- **Bruit (ϵ) :**
  1. Erreur de mesure.
  2. **Info manquante.** *(Variables cachées)*

Conséquence : Même avec le modèle parfait, l'erreur ne sera jamais 0.
(Il y a toujours une part d'aléa ou d'inconnu).

Sommaire

Le Mécanisme : Modèles et Paramètres (θ)

⚙ Un modèle n'est pas magique. C'est une fonction mathématique contrôlée par des **paramètres** ajustables.



Le Modèle Paramétrique :

$$h_\theta(x) = \theta_1 x + \theta_0$$

(Exemple : Régression Linéaire)

Le Vecteur de Paramètres θ :

C'est le "tableau de bord" du modèle.

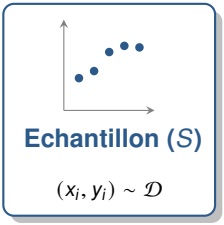
$\theta = (\theta_0, \theta_1) \in \mathbb{R}^2$

Notre Mission : La Construction de l'Estimateur ($\hat{h}$)

🔍 **Problème :** La fonction cible f^* est inconnue.
Objectif : Construire une approximation $\hat{h}$ à partir de l'échantillon S.

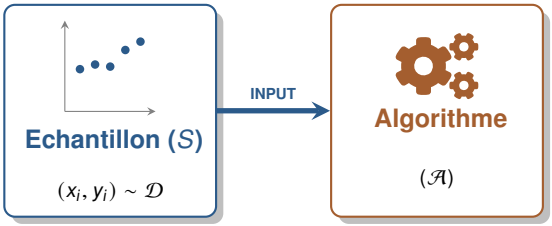
Notre Mission : La Construction de l'Estimateur ($\hat{h}$)

Problème : La fonction cible f^* est inconnue.
Objectif : Construire une approximation $\hat{h}$ à partir de l'échantillon S.



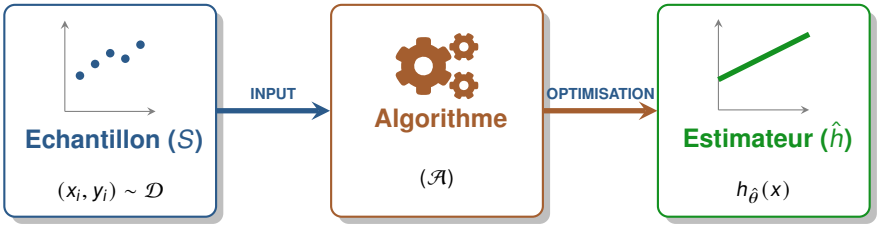
Notre Mission : La Construction de l'Estimateur ($\hat{h}$)

Problème : La fonction cible f^* est inconnue.
Objectif : Construire une approximation $\hat{h}$ à partir de l'échantillon S.



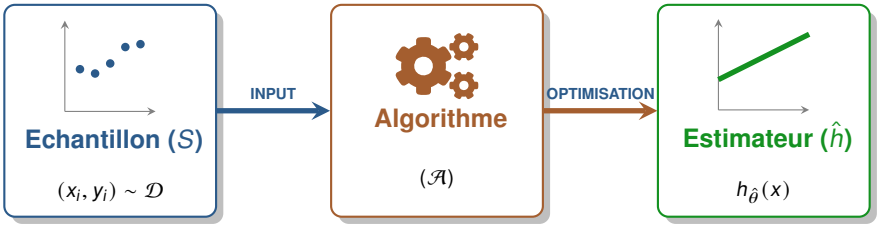
Notre Mission : La Construction de l'Estimateur ($\hat{h}$)

Problème : La fonction cible f^* est inconnue.
Objectif : Construire une approximation $\hat{h}$ à partir de l'échantillon S.



Notre Mission : La Construction de l'Estimateur ($\hat{h}$)

Problème : La fonction cible f^* est inconnue.
Objectif : Construire une approximation $\hat{h}$ à partir de l'échantillon S.



L'Algorithme d'Apprentissage

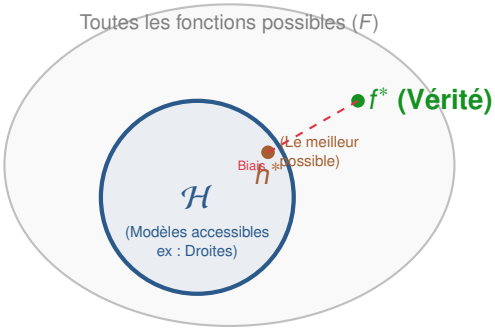
C'est la **procédure d'optimisation**. Elle cherche les meilleurs paramètres $\hat{\theta}$.
Ex : Minimisation des Moindres Carrés.

La Fonction de Décision

C'est l'**hypothèse sélectionnée**. Une fonction $h_{\hat{\theta}}$ prête à l'inférence.
Ex : La droite d'équation $y = \hat{a}x + \hat{b}$.

Où chercher ? L'Espace des Hypothèses ($\mathcal{H}$)

🔍 L'algorithme ne peut pas tout tester. Il se restreint à une **famille de fonctions** pré-définie.



L'Espace $\mathcal{H}$

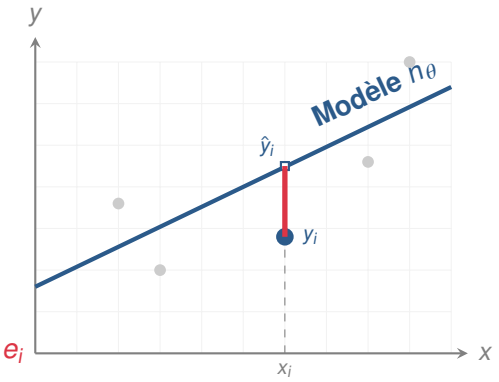
L'ensemble des fonctions que l'algorithme a le droit d'explorer.
Ex : $\mathcal{H} = \{ax + b \mid a, b \in \mathbb{R}\}$

Biais Inductif

Choisir $\mathcal{H}$, c'est faire un pari sur la forme de la solution.
Si f^* est loin de $\mathcal{H}$, on ne pourra jamais bien apprendre.

L'Arbitre : La Fonction de Coût (Loss)

⚖️ Il y a une infinité de modèles dans $\mathcal{H}$. Lequel est le meilleur ?



Les Métriques

1. La Prédiction ($\hat{y}_i$)
Valeur estimée par le modèle.

$$\hat{y}_i = h_\theta(x_i)$$

2. La Perte Locale (ℓ)
Exemple (Régression) : Norme L_2.

$$\ell(\hat{y}_i, y_i) = (\hat{y}_i - y_i)^2$$

(D'autres existent : L_1, Cross-Entropy...)

Optimisation : Comment trouver la meilleure règle ?

Problème : Comment utiliser la fonction de coût pour sélectionner le meilleur modèle $\hat{h}$?

Option A : Le Risque Réel

Minimiser l'erreur sur la **Vraie Distribution** ($\mathcal{D}$).

$$\mathcal{R}(h) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[\ell(h(x), y)]$$

(Espérance Mathématique)

Optimisation : Comment trouver la meilleure règle ?

Problème : Comment utiliser la fonction de coût pour sélectionner le meilleur modèle $\hat{h}$?

Option A : Le Risque Réel

Minimiser l'erreur sur la **Vraie Distribution** ($\mathcal{D}$).

$$\mathcal{R}(h) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[\ell(h(x), y)]$$

(Espérance Mathématique)

✕ IMPOSSIBLE

La distribution $\mathcal{D}$ est inconnue.

Optimisation : Comment trouver la meilleure règle ?

Problème : Comment utiliser la fonction de coût pour sélectionner le meilleur modèle $\hat{h}$?

Option A : Le Risque Réel

Minimiser l'erreur sur la **Vraie Distribution** ($\mathcal{D}$).

$$\mathcal{R}(h) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[\ell(h(x), y)]$$

(Espérance Mathématique)

✗ IMPOSSIBLE

La distribution $\mathcal{D}$ est inconnue.

Option B : Le Risque Empirique

Minimiser l'erreur sur les **Données Collectées** (S).

$$\hat{\mathcal{R}}_S(h) = \frac{1}{n}\sum_{i=1}^{n} \ell(h(x_i), y_i)$$

(Moyenne Empirique)

✓ FAISABLE

On possède l'échantillon S.

Synthèse : La Formulation Mathématique (ERM)

1. L'Expérience (S)

Un échantillon de n couples i.i.d. tirés selon $\mathcal{D}$:

$$S = \{(x_i, y_i)\}_{i=1}^{n}$$

avec $x \in \mathcal{X}$, $y \in \mathcal{Y}$.

2. L'Hypothèse (h_θ)

Une fonction choisie dans l'espace de recherche $\mathcal{H}$:

$$h_\theta : \mathcal{X} \to \mathcal{Y}$$

paramétrée par $\theta \in \Theta \subseteq \mathbb{R}^d$.

3. Le Risque ($\hat{\mathcal{R}}$)

L'erreur moyenne sur S (Risque Empirique) :

$$\hat{\mathcal{R}}_S(\theta) = \frac{1}{n} \sum_{i=1}^{n} \ell(h_\theta(x_i), y_i)$$

Empirical Risk Minimization (ERM)

$$\hat{\theta} \in \arg\min_{\theta \in \Theta} \left(\frac{1}{n} \sum_{i=1}^{n} \ell(h_\theta(x_i), y_i) \right)$$

✔ *"On cherche le paramètre $\hat{\theta}$ qui minimise l'erreur cumulée sur les données observées."*

Sommaire

Présentation visuelle : Le problème de classification

? Le Défi

Sachant un jeu de données S (les points connus), comment classer le nouvel individu **(?)** ?

Figure – Le jeu de données complet

Figure – L'inconnu

Présentation visuelle : Le problème de classification

? Le Défi

Sachant un jeu de données S (les points connus), comment classer le nouvel individu **(?)** ?

Intuition : Regarder les **K** exemples les plus proches.

Figure – Le jeu de données complet

Figure – L'inconnu

Intuition k-NN : Cas $k = 3$

Voisinage $k = 3$

On trace un cercle pour englober les **3 plus proches voisins**.

Intuition k-NN : Cas $k = 3$

Voisinage $k = 3$

On trace un cercle pour englober les **3 plus proches voisins**.

Résultat du vote :

- ✔ 3 Chats
- ✖ 0 Autre

→ **100% Chat**

Intuition k-NN : Cas $k = 6$

Voisinage $k = 6$

On agrandit le voisinage pour chercher **6 voisins**.

Intuition k-NN : Cas $k = 6$

Voisinage $k = 6$

On agrandit le voisinage pour chercher **6 voisins**.

Résultat du vote :

- 3 Chats
- 3 Lions

→ **50% / 50% (Incertitude)**

Conclusion : Le choix de k change la décision !

Du concept à la pratique : Formaliser le k-NN

Lazy Learning (Apprentissage Paresseux)

Pas d'entraînement → On stocke tout le dataset S en mémoire.

Question : De quoi a-t-on besoin pour coder cet algorithme ?

(Il nous faut définir 3 ingrédients mathématiques...)

Du concept à la pratique : Formaliser le k-NN

Lazy Learning (Apprentissage Paresseux)

Pas d'entraînement → On stocke tout le dataset S en mémoire.

Question : De quoi a-t-on besoin pour coder cet algorithme ?

(Il nous faut définir 3 ingrédients mathématiques...)

Du concept à la pratique : Formaliser le k-NN

Lazy Learning (Apprentissage Paresseux)

Pas d'entraînement → On stocke tout le dataset S en mémoire.

Question : De quoi a-t-on besoin pour coder cet algorithme ?

(Il nous faut définir 3 ingrédients mathématiques...)

Combien ?

Hyperparamètre k

(1, 3, 50 ?)

Du concept à la pratique : Formaliser le k-NN

Lazy Learning (Apprentissage Paresseux)

Pas d'entraînement → On stocke tout le dataset S en mémoire.

Question : De quoi a-t-on besoin pour coder cet algorithme ?

(Il nous faut définir 3 ingrédients mathématiques...)

Combien ?

Hyperparamètre k

(1, 3, 50 ?)

Où ?

Distance $d(\cdot, \cdot)$

(Euclidienne, etc.)

Du concept à la pratique : Formaliser le k-NN

Lazy Learning (Apprentissage Paresseux)

Pas d'entraînement → On stocke tout le dataset S en mémoire.

Question : De quoi a-t-on besoin pour coder cet algorithme ?

(Il nous faut définir 3 ingrédients mathématiques...)

Combien ?

Hyperparamètre k

(1, 3, 50 ?)

Où ?

Distance $d(\cdot, \cdot)$

(Euclidienne, etc.)

Quoi ?

Décision

(Vote, Moyenne...)

Du concept à la pratique : Formaliser le k-NN

Lazy Learning (Apprentissage Paresseux)

Pas d'entraînement → On stocke tout le dataset S en mémoire.

Question : De quoi a-t-on besoin pour coder cet algorithme ?

(Il nous faut définir 3 ingrédients mathématiques...)

Combien ?

Hyperparamètre k

(1, 3, 50 ?)

Où ?

Distance $d(\cdot, \cdot)$

(Euclidienne, etc.)

Quoi ?

Décision

(Vote, Moyenne...)

→ C'est la combinaison de ces 3 choix qui crée l'algorithme.

Formalisation k-NN (1/4) : Configuration & Mesure

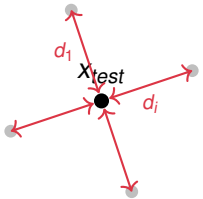
Configuration / Entrées

Pour prédire la sortie d'un nouvel individu $x_{test} \in \mathcal{X}$, l'algorithme nécessite :

Le Jeu de Données S : Ensemble de n couples $\{(x_1, y_1), \ldots, (x_n, y_n)\}$.

L'Hyperparamètre k : Nombre de voisins à considérer ($k \in \mathbb{N}^*$).

La Métrique $d(\cdot, \cdot)$: Une fonction de distance $d(x_a, x_b)$ entre deux exemples.



Étape 1 : Calcul des distances On calcule la distance entre x_{test} et **tous** les points x_i de S :

$$d_i = d(x_{test}, x_i) \quad \forall i \in \{1, \ldots, n\}$$

Formalisation k-NN (2/4) : Le choix de la métrique $d(\cdot, \cdot)$

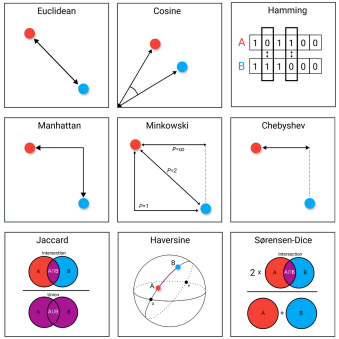


Figure – Différentes distances possibles.

Distances de Minkowski (L_p)

$$d_p(x, x') = \left(\sum_{j=1}^{n_X} |x_j - x'_j|^p \right)^{1/p}$$

Cas usuels :

- $p = 1$ **(Manhattan)** : Déplacement en grille (Taxi).
- $p = 2$ **(Euclidienne)** : Standard.
- $p = \infty$ **(Chebyshev)** : Max.

Algorithme k-NN (3/4) : Sélection des Voisins

Étape 2 : Tri et Sélection
On ordonne les distances calculées de la plus petite à la plus grande. On ne conserve que les indices des k voisins les plus proches.

Formalisme Mathématique : Soit π la permutation de $\{1, \ldots, n\}$ qui trie les indices par distance croissante par rapport à x_{test} :

$$d(x_{test}, x_{\pi(1)}) \leq d(x_{test}, x_{\pi(2)}) \leq \cdots \leq d(x_{test}, x_{\pi(n)})$$

Ensemble des voisins ($\mathcal{N}_k$) :

$$\mathcal{N}_k = \{\pi(1), \pi(2), \ldots, \pi(k)\}$$

Liste triée

1. **Indice** $\pi(1)$ **(**$d = 0.2$**)** ✔
2. **Indice** $\pi(2)$ **(**$d = 0.5$**)** ✔
3. **Indice** $\pi(3)$ **(**$d = 0.8$**)** ✔

---

4. Indice $\pi(4)$ $(d = 4.1)$ ✘

...

n. Indice $\pi(n)$ $(d = 9.9)$ ✘

Exemple pour $k = 3$

Algorithme k-NN (4/4) : La Règle de Décision Φ

Étape 3 : Agrégation (Fonction Φ)

La prédiction finale est le résultat d'une fonction d'agrégation Φ qui prend en entrée les valeurs des voisins **et** leurs distances :

$$h_S(x_{test}) = \Phi\Big(\underbrace{(y_{\pi_1}, d_{\pi_1}), \dots, (y_{\pi_k}, d_{\pi_k})}_{\text{Voisins et Distances}} \Big)$$

Classification

Vote Majoritaire

$$\hat{y} = \underset{c}{\operatorname{argmax}} \sum_{i=1}^{k} \mathbb{1}_{\{y_{\pi_i}=c\}}$$

Extension : Vote pondéré par la distance.

Régression

Moyenne Pondérée

$$\hat{y} = \frac{\sum w_i y_{\pi_i}}{\sum w_i}, \quad w_i = \begin{cases} 1 & \text{(Simple)} \\ \dfrac{1}{d_{\pi_i}} & \text{(Pondéré)} \end{cases}$$

Synthèse : La "Carte d'Identité" du k-NN

Philosophie

1. Lazy Learning (Paresseux)

- **Pas d'entraînement :** Le modèle EST le dataset S.
- **Mémoire :** Stockage de tous les points.

2. Approche Locale

- S'adapte aux formes complexes.
- Pas de formule globale type $y = ax + b$.

Paramètres à définir

Pour utiliser l'algo, vous devez choisir :

Synthèse : La "Carte d'Identité" du k-NN

Philosophie

1. Lazy Learning (Paresseux)

- **Pas d'entraînement :** Le modèle EST le dataset S.
- **Mémoire :** Stockage de tous les points.

2. Approche Locale

- S'adapte aux formes complexes.
- Pas de formule globale type $y = ax + b$.

Paramètres à définir

Pour utiliser l'algo, vous devez choisir :

Hyperparamètre k
Combien de voisins ? (Compromis)

Distance $d(\cdot, \cdot)$
Quelle distance ? (L_1, L_2)

Fonction Φ
Quelle règle ? (Vote/Moyenne)

Synthèse : La "Carte d'Identité" du k-NN

Philosophie

1. Lazy Learning (Paresseux)

- **Pas d'entraînement :** Le modèle EST le dataset S.
- **Mémoire :** Stockage de tous les points.

2. Approche Locale

- S'adapte aux formes complexes.
- Pas de formule globale type $y = ax + b$.

Paramètres à définir

Pour utiliser l'algo, vous devez choisir :

Hyperparamètre k
Combien de voisins ? (Compromis)

Distance $d(\cdot, \cdot)$
Quelle distance ? (L_1, L_2)

Fonction Φ
Quelle règle ? (Vote/Moyenne)

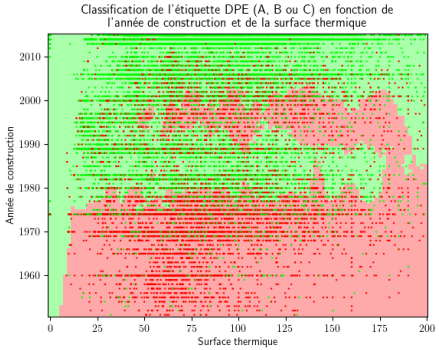
Le Flux Algorithmique : $x_{test} \xrightarrow{d(\cdot,\cdot)}$ Tri $\xrightarrow{\text{Top } k} \mathcal{N}_k \xrightarrow{\Phi} \hat{y}$

Sommaire

Application 1 (Classification) : Le Diagnostic DPE

Le Problème : Prédire la classe énergétique

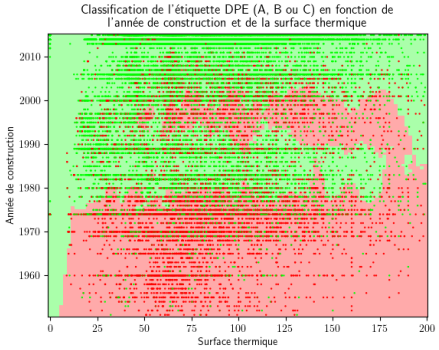
Données : Caractéristiques techniques du logement (Surface, Isolation, Année...).
Cible Y **:** Le logement est-il performant (A, B, C) ou non ?
Méthode k-NN : Règle du **Vote Majoritaire** parmi les k voisins les plus similaires.



Application 1 (Classification) : Le Diagnostic DPE

Le Problème : Prédire la classe énergétique

Données : Caractéristiques techniques du logement (Surface, Isolation, Année...).
Cible Y **:** Le logement est-il performant (A, B, C) ou non ?
Méthode k-NN : Règle du **Vote Majoritaire** parmi les k voisins les plus similaires.



Analyse

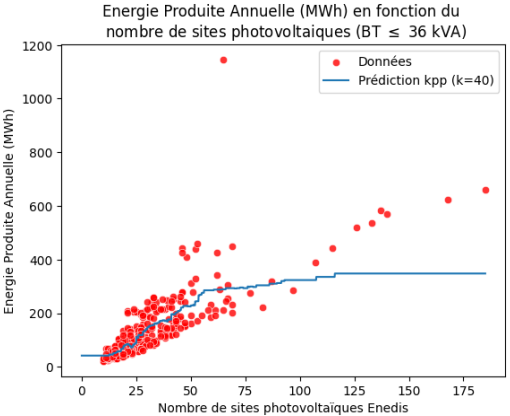
- **Non-Linéaire :** Frontières courbes, pas de ligne droite.
- **Localité :** Capture des "zones" isolées.

→ Cartographie précise du territoire.

Application 2 (Régression) : L'impact de k

⚙ Contexte : Parc Photovoltaïque

Variable X **:** Nombre de sites photovoltaïques.
Cible Y **:** Énergie produite annuelle (MWh).
Problème : Comment le choix de k influence la prédiction de production ?



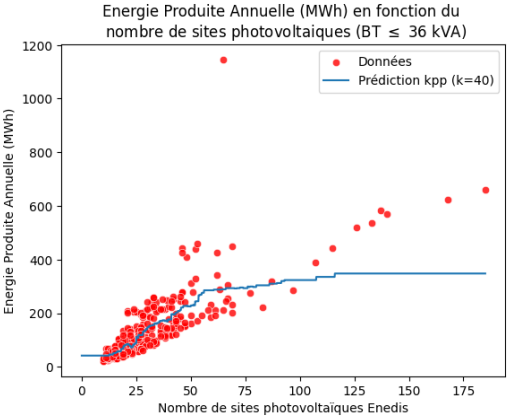
Cas A : $k = 40$

❓ *Votre avis ?*

Application 2 (Régression) : L'impact de k

⚙ Contexte : Parc Photovoltaïque

Variable X **:** Nombre de sites photovoltaïques.
Cible Y **:** Énergie produite annuelle (MWh).
Problème : Comment le choix de k influence la prédiction de production ?



Cas A : $k = 40$

❓ *Votre avis ?*

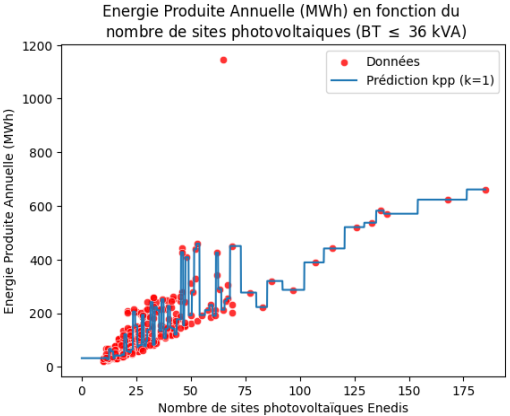
Analyse

C'est trop **lisse**.
Le modèle fait une moyenne globale et rate les pics de production.
→ **Sous-apprentissage**.

Application 2 (Régression) : L'impact de k

⚙ Contexte : Parc Photovoltaïque

Variable X **:** Nombre de sites photovoltaïques.
Cible Y **:** Énergie produite annuelle (MWh).
Problème : Comment le choix de k influence la prédiction de production ?



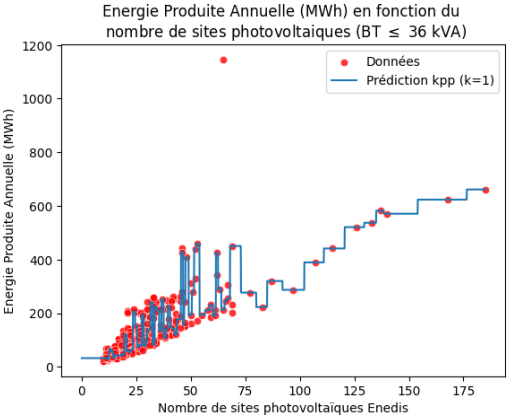
Cas B : $k = 1$

❓ *Mieux ?*

Application 2 (Régression) : L'impact de k

⚙ Contexte : Parc Photovoltaïque

Variable X **:** Nombre de sites photovoltaïques.
Cible Y **:** Énergie produite annuelle (MWh).
Problème : Comment le choix de k influence la prédiction de production ?



Cas B : $k = 1$

❓ *Mieux ?*

Analyse

C'est trop **nerveux**.
Le modèle apprend le "bruit" et les erreurs de mesure.
→ **Sur-apprentissage**.

Synthèse Régression : La fonction en escalier

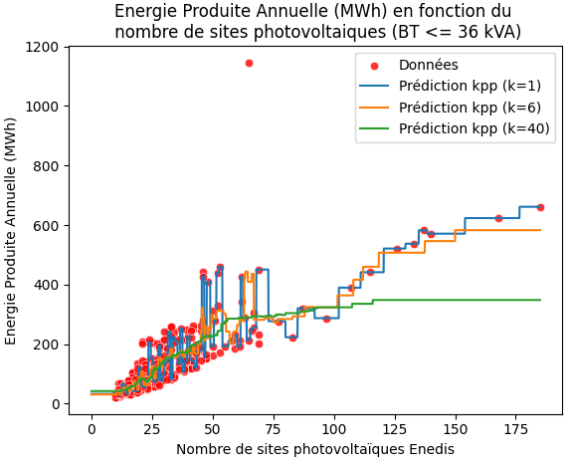


Figure – Prédiction Production (Y) vs Nbr Sites (X)

⚖ Le Compromis

Il faut trouver un k intermédiaire (ex : courbe verte) pour capter la tendance sans le bruit.

🔍 Observation Mathématique :

La régression k-NN produit une fonction **constante par morceaux** ("en escalier").

Tant que l'ensemble des k voisins (sites similaires) ne change pas, la moyenne de production prédite reste identique.

En Pratique : Implémentation Scikit-Learn

</> Séquence Type (Python)

```
from sklearn.neighbors import KNeighborsRegressor

X = production["nb_sites"].values
y = production["energie_mwh"].values.reshape(-1, 1)

# 2. Modèle : On fixe k ici
neigh = KNeighborsRegressor(n_neighbors=5)

# 3. Fit : Stockage des données (Lazy)
neigh.fit(X, y)

# 4. Predict : Calculs de distance + Moyenne
y_pred = neigh.predict(x_new)
```

Mécanique Interne

.fit()	Stocke seulement S (Rapide).
predict()	Fait tout le travail (Lent).

</> Voir le Code Complet

(Notebook Google Colab)

Limitation 1 : Le Piège des Unités

⚖ Problème

Le k-NN calcule des distances. Si une variable a de **grandes valeurs**, elle écrase les autres.

Exemple Immo :

Surface : $100 m^2$ (Écart type ≈ 20)

Année : 2000 (Écart type ≈ 10)

Prix : 300 000 € (Écart type $\approx 50\,000$)

$\rightarrow$ La distance ne voit que le Prix !

Solution

Normalisation (Standard Scaler)

$$\tilde{x} = \frac{x - \mu}{\sigma}$$

Toutes les variables ont le même poids.

Limitation 2 : Le Fléau de la Dimension

Le Paradoxe

Intuitivement, plus on a d'infos (variables), mieux c'est.
En k-NN, c'est l'inverse ! Plus la dimension d augmente, plus la notion de "voisin" perd son sens.

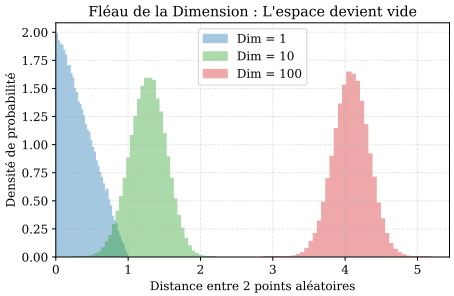


Figure – Distribution des distances.

L'Espace est vide

- **Dim 1 (Bleu) :** Les points sont proches.
- **Dim 100 (Rouge) :** La distance moyenne explose.

Tout le monde devient loin de tout le monde.

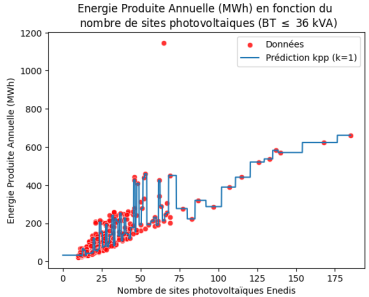
Conséquence

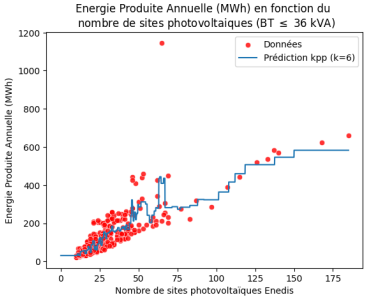
Le k-NN échoue sur les données brutes de grande dimension (Images, Texte).

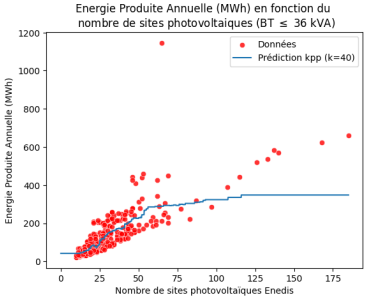
Sommaire

Le Challenge : Automatiser l'Intuition

Ce que nous voyons (L'œil humain) :







Trop Complexe

"Nerveux" (Bruit)

Bon Compromis

"Juste ce qu'il faut"

Trop Simple

"Mou" (Moyenne)

? Question : Comment définir mathématiquement le "meilleur" modèle ?

Formalisons le problème : Idéal vs Réalité

L'Objectif Réel (Généralisation)

Nous cherchons le modèle h_k qui minimise l'erreur sur la distribution réelle $\mathcal{D}$.

$$\min_k \mathcal{R}(h_k)$$

(Être performant sur les futures données, pas juste le passé.)

Le Problème

Réalité : La distribution $\mathcal{D}$ est **inconnue**.

Nous n'avons accès qu'à un échantillon partiel : le jeu d'entraînement S.

Formalisons le problème : Idéal vs Réalité

L'Objectif Réel (Généralisation)

Nous cherchons le modèle h_k qui minimise l'erreur sur la distribution réelle $\mathcal{D}$.

$$\min_k \mathcal{R}(h_k)$$

(Être performant sur les futures données, pas juste le passé.)

Le Problème

Réalité : La distribution $\mathcal{D}$ est **inconnue**.

Nous n'avons accès qu'à un échantillon partiel : le jeu d'entraînement S.

Conséquence Logique : L'Approche ERM

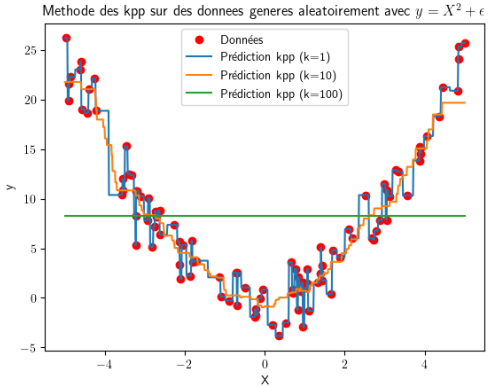
On ne peut pas calculer $\mathcal{R}(h_k)$, donc on minimise ce qu'on voit : **l'Erreur Empirique sur** S.

$$\text{ERM} : \min_k \hat{\mathcal{R}}_S(h_k)$$

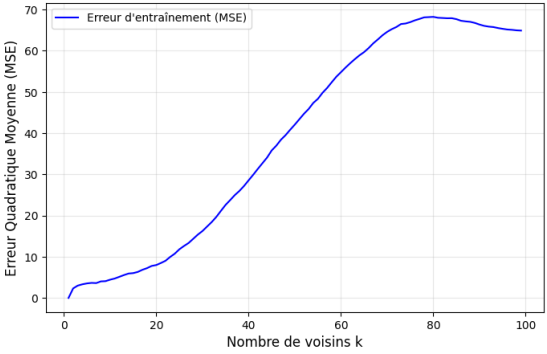
"On espère que si l'erreur est faible sur S, elle sera faible sur $\mathcal{D}$..."

Expérience 1 : L'approche ERM (Test sur S)

Protocole : On teste tous les k et on cherche le minimum d'erreur sur S.



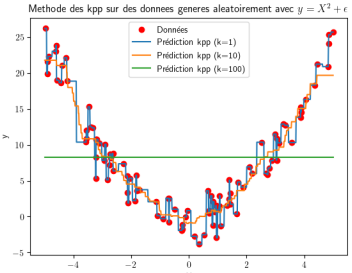
Prédiction Visuelle



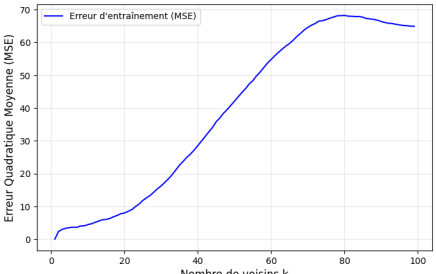
Risque Empirique ($\hat{\mathcal{R}}_S$)

Expérience 1 : L'approche ERM (Test sur S)

Protocole : On teste tous les k et on cherche le minimum d'erreur sur S.



Prédiction Visuelle



Risque Empirique ($\hat{\mathcal{R}}_S$)

Verdict de l'Ordinateur vs Réalité

L'Ordi : "L'erreur est nulle (0) pour $k = 1$, donc c'est le meilleur !"

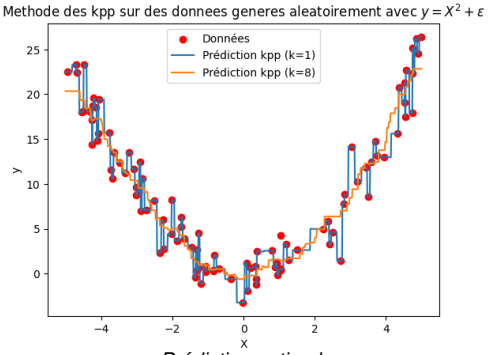
Le Piège : Regardez à gauche → C'est du bruit pur (Overfitting).

Conclusion : Minimiser l'erreur sur S ne suffit pas.

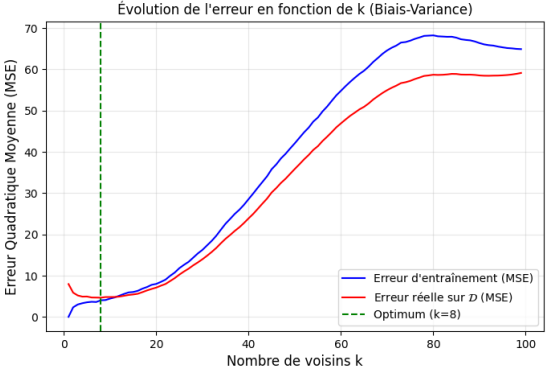
Expérience 2 : La Vérité (Test sur $\mathcal{D}$)

Vue "Oracle" (Simulation)

Nous connaissons la vraie loi. Calculons l'erreur sur de **nouvelles données**.



Prédiction optimale

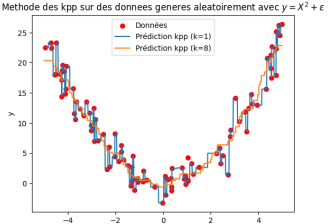


Risque Réel ($\mathcal{R}$)

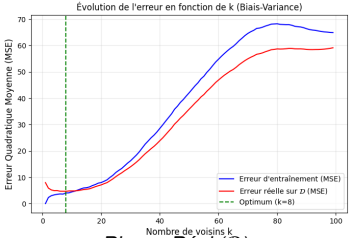
Expérience 2 : La Vérité (Test sur $\mathcal{D}$)

Vue "Oracle" (Simulation)

Nous connaissons la vraie loi. Calculons l'erreur sur de **nouvelles données**.



Prédiction optimale



Risque Réel ($\mathcal{R}$)

Le Verdict

Observation :
L'erreur réelle remonte si k est trop petit.

Gagnant :
$k = 8$ (et non $k = 1$)

Le Conflit : $\hat{\mathcal{R}}_S$ vs $\mathcal{R}$

❓ Que vient-il de se passer ?

Sur l'Entraînement (S)

Quand la complexité augmente ($k \to 1$)...

... l'erreur **diminue** continuellement.

↓ Tend vers 0 (Apprentissage par cœur).

Sur la Réalité ($\mathcal{D}$)

Quand la complexité augmente ($k \to 1$)...

1. L'erreur diminue (apprentissage).
2. Puis elle **REMONTE** (sur-apprentissage).

↺ **Forme en "U"**.

Le Conflit : $\hat{\mathcal{R}}_S$ vs $\mathcal{R}$

❓ Que vient-il de se passer ?

Sur l'Entraînement (S)

Quand la complexité augmente ($k \to 1$)...

... l'erreur **diminue** continuellement.

↓ Tend vers 0 (Apprentissage par cœur).

Sur la Réalité ($\mathcal{D}$)

Quand la complexité augmente ($k \to 1$)...

1. L'erreur diminue (apprentissage).
2. Puis elle **REMONTE** (sur-apprentissage).

↺ **Forme en "U"**.

⚠ Conclusion Fondamentale

Une bonne performance sur S ne garantit rien sur $\mathcal{D}$.

$$\hat{\mathcal{R}}_S(h) \approx 0 \quad \not\Rightarrow \quad \mathcal{R}(h) \text{ est faible}$$

Le Cas Pathologique de l'ERM

Règle de décision "Par Cœur"

Considérons la fonction h_S qui mémorise tout :

$$h_S(x) = \begin{cases} y_i & \text{si } x = x_i \in S \text{ (Connu)} \\ C & \text{sinon (Inconnu)} \end{cases}$$

Analyse des Erreurs :

- ✔ **Sur S :** $\hat{\mathcal{R}}_S(h_S) = 0$
  (Risque empirique nul).
- ✘ **Sur $\mathcal{D}$:** $\mathcal{R}(h_S) \approx 0.5$
  (Risque réel élevé, équivalent au hasard).

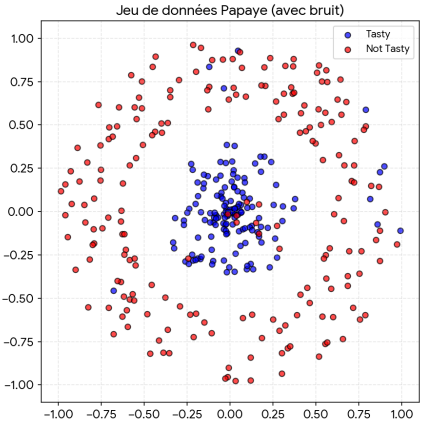


Figure – h_S apprend le bruit (rouge dans bleu).

Application directe : Le cas $k = 1$

L'algorithme 1-NN est une fonction de type h_S.
(C'est une machine à mémoriser)

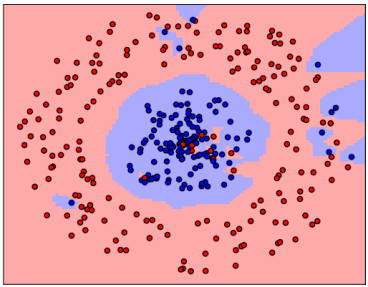
Mécanisme sur le jeu d'entraînement :

Pour tout point x_i connu...

... son plus proche voisin est **lui-même**.

→ Distance = 0.

→ Erreur = 0.



Les "îlots" de décision autour du bruit.

Résultat

Le modèle capture le **bruit** et les anomalies sans distinction.

Sommaire

La Solution : Restreindre la Recherche

Le Biais Inductif

Pour empêcher le "par cœur", nous devons **interdire** certaines fonctions trop complexes.
→ On restreint la recherche à une **Classe d'Hypothèses** spécifique.

Exemple 1 : Modèle Linéaire

On force la fonction à être une droite : $y = ax + b$.

- ⊘ Impossible d'entourer chaque point de bruit.
- ✔ Force la généralisation.

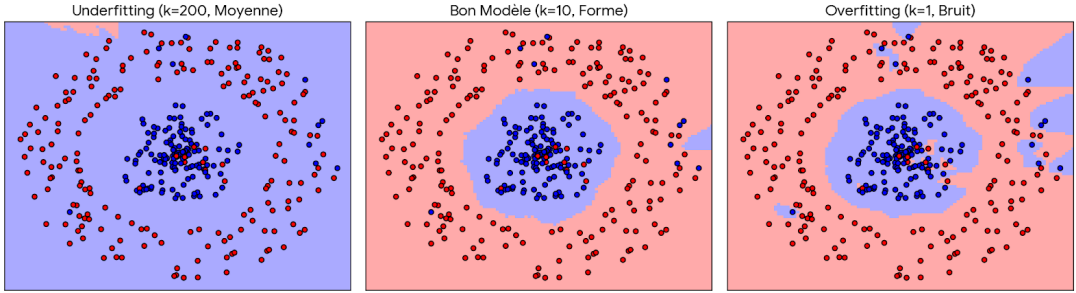
Exemple 2 : k-NN (k fixé)

On force la décision à dépendre de k voisins.

- Si on fixe $k = 10$, on interdit d'isoler un point seul.
- C'est notre garantie contre le sur-apprentissage.

*Le choix de la classe (Linéaire, k-NN...) est ce qu'on appelle le **Biais Inductif**.*

Trouver le "Juste Milieu" (Compromis)



Sous-ajustement
(Underfitting)
k **très grand**

Bon Modèle
(Généralisation)
k **optimal**

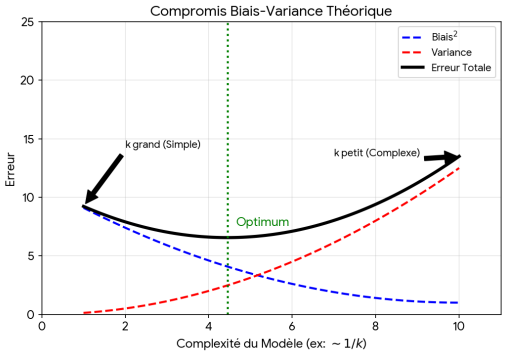
Sur-ajustement
(Overfitting)
$k = 1$

Interprétation par la Classe d'Hypothèse :

Classe trop restrictive
(Modèle trop rigide / Biais fort)

Classe trop permissive
(Modèle trop libre / Variance forte)

Le Compromis Biais-Variance



Le Biais (Sous-apprentissage)

Cause : Hypothèses trop simplificatrices.
Le modèle est trop **rigide**.
→ Il échoue à capturer la structure des données (Underfitting).

La Variance (Sur-apprentissage)

Cause : Sensibilité excessive à l'échantillon S.
Le modèle est trop **flexible**.
→ Il modélise le **bruit** aléatoire au lieu du signal (Overfitting).

$$\textbf{Erreur} \approx \text{Biais}^2 + \text{Variance} + \epsilon$$

Paramètre k	Complexité	Biais	Variance	Diagnostic
Petit ($k \to 1$)	Haute	Faible	**Haute**	**Overfitting**
Grand ($k \to N$)	Faible	**Haut**	Faible	**Underfitting**

Conclusion : Vers la Généralisation

Bilan Théorique

- **Estimateur k-NN :**
  Méthode non-paramétrique locale, très sensible au choix de k et à la dimension d.
- **Échec de l'ERM :**
  Minimiser $\hat{\mathcal{R}}_S$ sans contrainte mène à $k = 1$ (Variance max).
- **Décomposition :**

$$\text{Err} = \text{Biais}^2 + \text{Var} + \text{Bruit}$$

Il faut arbitrer entre rigidité et flexibilité.

Le Problème d'Optimisation

On cherche à minimiser le **Risque Réel** :

$$k^* = \underset{k}{\operatorname{argmin}}\ \mathcal{R}(h_k)$$

Contrainte : $\mathcal{D}$ est inconnue.
Le Risque Empirique $\hat{\mathcal{R}}_S$ est un estimateur *biaisé* (optimiste).

Conclusion : Vers la Généralisation

Bilan Théorique

- **Estimateur k-NN :**
  Méthode non-paramétrique locale, très sensible au choix de k et à la dimension d.
- **Échec de l'ERM :**
  Minimiser $\hat{\mathcal{R}}_S$ sans contrainte mène à $k = 1$ (Variance max).
- **Décomposition :**

$$\text{Err} = \text{Biais}^2 + \text{Var} + \text{Bruit}$$

Il faut arbitrer entre rigidité et flexibilité.

Le Problème d'Optimisation

On cherche à minimiser le **Risque Réel** :

$$k^* = \underset{k}{\operatorname{argmin}}\ \mathcal{R}(h_k)$$

Contrainte : $\mathcal{D}$ est inconnue.
Le Risque Empirique $\hat{\mathcal{R}}_S$ est un estimateur *biaisé* (optimiste).

Prochaine étape :
(Comment maximiser $\mathcal{R}(h_k)$ sans "tricher" ?)

Introduction : Loss Function vs Métrique

Question : Si mon modèle a une erreur moyenne de 10^{-3}, est-il bon ?

Introduction : Loss Function vs Métrique

Question : Si mon modèle a une erreur moyenne de 10^{-3}, est-il bon ?
Réponse : Impossible à dire sans connaître le contexte métier.

La Fonction de Coût (Loss)

Pour qui ? L'algorithme d'optimisation (Gradient Descent).

Propriétés requises :

- Dérivable (lisse).
- Convexe (si possible).

Ex : MSE (L_2), LogLoss.

La Métrique d'Évaluation

Pour qui ? Vous et le client.

Propriétés requises :

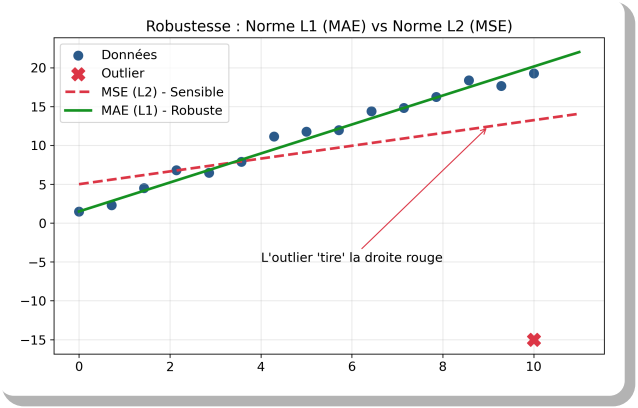
- Interprétable (Unité réelle).
- Robuste.

Ex : MAE (Euros), Accuracy (%).

→ *Parfois ce sont les mêmes (MSE), parfois non (Classification).*

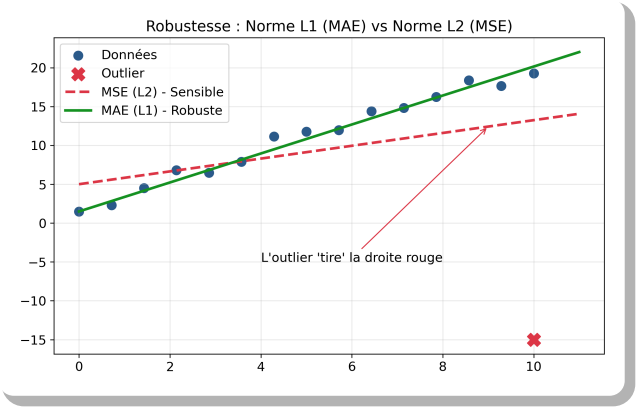
Sommaire

Régression : Le duel des normes (L_1 vs L_2)



👁 *L'outlier rouge "attire" la prédiction L2.*

Régression : Le duel des normes (L_1 vs L_2)



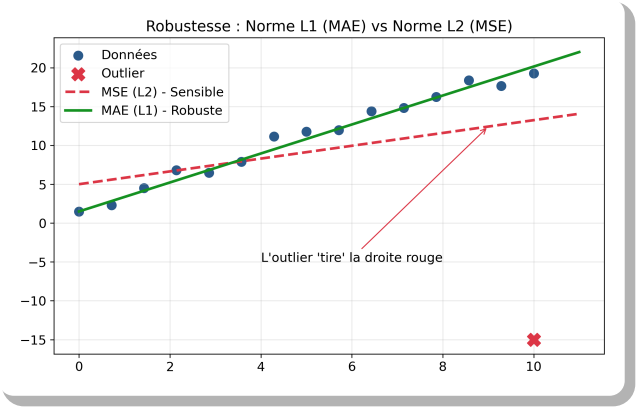
L'outlier rouge "attire" la prédiction L2.

Norme L_2 (Sensible)

Le carré amplifie les erreurs extrêmes.

$$\textbf{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Régression : Le duel des normes (L_1 vs L_2)



L'outlier rouge "attire" la prédiction L2.

Norme L_2 (Sensible)

Le carré amplifie les erreurs extrêmes.

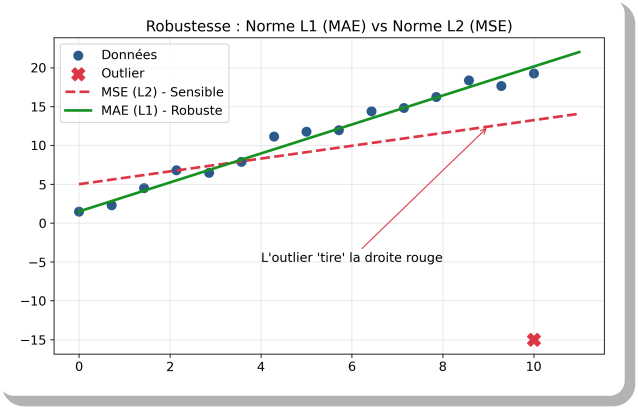
$$\textbf{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Norme L_1 (Robuste)

L'erreur est linéaire (Médiane).

$$\textbf{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Régression : Le duel des normes (L_1 vs L_2)



L'outlier rouge "attire" la prédiction L2.

Norme L_2 (Sensible)

Le carré amplifie les erreurs extrêmes.

$$\textbf{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Norme L_1 (Robuste)

L'erreur est linéaire (Médiane).

$$\textbf{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Note : La **MSE** (Mean Squared Error) est le carré de la RMSE.

$$\text{MSE} = \frac{1}{n}\sum(y_i - \hat{y}_i)^2 \quad \text{(Optimisée par l'algo)}$$

Régression : Le Coefficient de Détermination (R^2)

Comparaison avec le modèle "naïf" (la moyenne $\bar{y}$)

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2} = 1 - \frac{\text{Erreur de mon modèle}}{\text{Variance totale des données}}$$

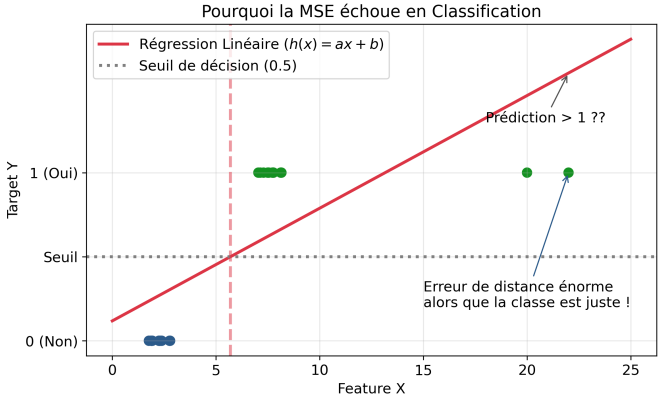
Comment lire votre note ?

Attention : Un R^2 élevé ne garantit pas qu'il n'y a pas de sur-apprentissage !

Sommaire

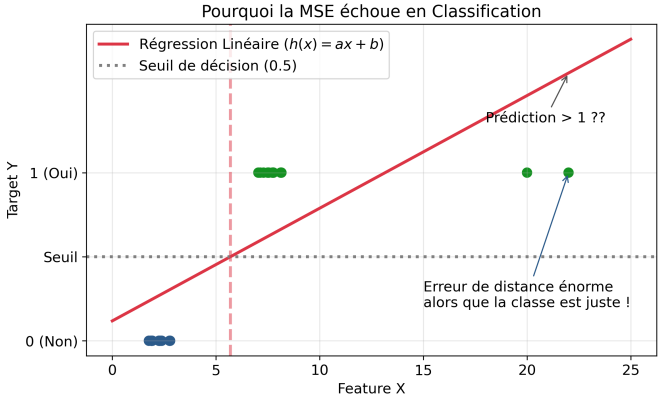
Le Pivot : Pourquoi la MSE échoue en Classification ?

Question : Peut-on évaluer des classes 0 et 1 avec une distance au carré (MSE) ?



Le Pivot : Pourquoi la MSE échoue en Classification ?

Question : Peut-on évaluer des classes 0 et 1 avec une distance au carré (MSE) ?



PROBLÈME 1

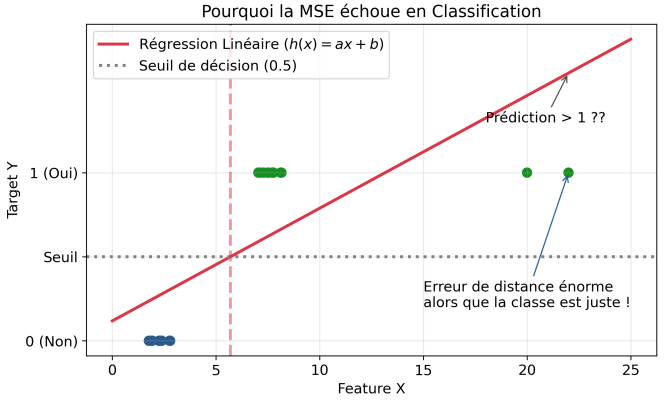
Prédiction hors bornes

$\hat{y} > 1$ ou < 0 ?

Impossible pour une proba.

Le Pivot : Pourquoi la MSE échoue en Classification ?

Question : Peut-on évaluer des classes 0 et 1 avec une distance au carré (MSE) ?



PROBLÈME 1

Prédiction hors bornes

$\hat{y} > 1$ ou < 0 ?

Impossible pour une proba.

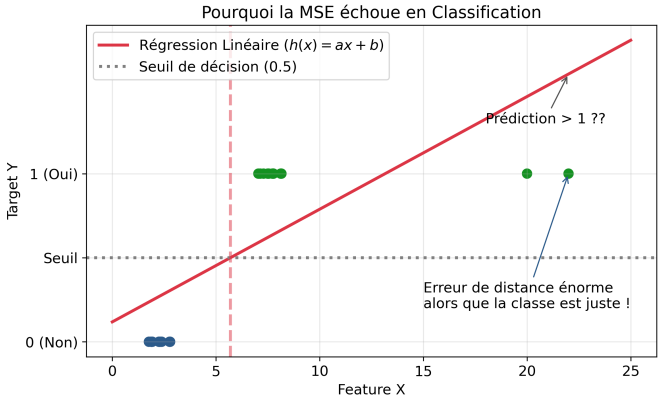
PROBLÈME 2

Instabilité du seuil

L'outlier "tire" la droite et déplace la frontière de décision au mauvais endroit.

Le Pivot : Pourquoi la MSE échoue en Classification ?

Question : Peut-on évaluer des classes 0 et 1 avec une distance au carré (MSE) ?



PROBLÈME 1

Prédiction hors bornes

$\hat{y} > 1$ ou < 0 ?

Impossible pour une proba.

PROBLÈME 2

Instabilité du seuil

L'outlier "tire" la droite et déplace la frontière de décision au mauvais endroit.

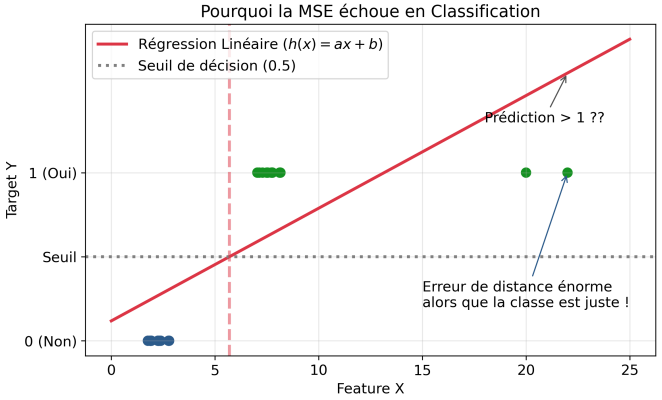
PROBLÈME 3

Pénalisation injuste

La MSE punit une réponse "trop vraie" (ex : $\hat{y} = 2$ pour $y = 1$).

Le Pivot : Pourquoi la MSE échoue en Classification ?

Question : Peut-on évaluer des classes 0 et 1 avec une distance au carré (MSE) ?



PROBLÈME 1

Prédiction hors bornes

$\hat{y} > 1$ ou < 0 ?

Impossible pour une proba.

PROBLÈME 2

Instabilité du seuil

L'outlier "tire" la droite et déplace la frontière de décision au mauvais endroit.

PROBLÈME 3

Pénalisation injuste

La MSE punit une réponse "trop vraie" (ex : $\hat{y} = 2$ pour $y = 1$).

Conclusion : En classification, la distance ne compte pas. **Seule la décision compte.**

Classification : Comment noter une décision binaire ?

Scénario : Vous devez évaluer un filtre Anti-Spam.
J'ai 100 emails (90 normaux, 10 spams). Mon modèle en classe 95 correctement.

Votre premier réflexe ?

Classification : Comment noter une décision binaire ?

Scénario : Vous devez évaluer un filtre Anti-Spam.
J'ai 100 emails (90 normaux, 10 spams). Mon modèle en classe 95 correctement.

Votre premier réflexe ?

"Il a 95/100."

"Il fait 5% d'erreur."

C'est ce qu'on appelle l'**Exactitude** (Accuracy).

$$\text{Accuracy} = \frac{\text{Vrai Pos} + \text{Vrai Neg}}{\text{Total}}$$

Classification : Comment noter une décision binaire ?

Scénario : Vous devez évaluer un filtre Anti-Spam.
J'ai 100 emails (90 normaux, 10 spams). Mon modèle en classe 95 correctement.

Votre premier réflexe ?

"Il a 95/100."

"Il fait 5% d'erreur."

$$\text{Accuracy} = \frac{\text{Vrai Pos} + \text{Vrai Neg}}{\text{Total}}$$

C'est ce qu'on appelle l'**Exactitude** (Accuracy).

Mais est-ce suffisant ?

Imaginez maintenant que j'ai 99 emails normaux et 1 seul spam...

Le Piège de l'Accuracy (Classes Déséquilibrées)

Le Modèle "Paresseux" :
Il prédit **TOUJOURS** "Non Spam" (Classe 0), quoi qu'il arrive.

Performance sur 100 mails (dont 1 seul spam) :

99 Normaux → Prédits Normaux (✔ OK)

1 Spam → Prédit Normal (✖ Erreur)

SCORE

99%

d'exactitude

Conclusion : L'accuracy est aveugle aux classes minoritaires.
Il faut arrêter de compter juste "Juste/Faux", il faut regarder **quel type d'erreur** on fait.

Le Vocabulaire : Les 4 cas possibles

En binaire, il y a deux façons d'avoir raison, et deux façons d'avoir tort.

Classe Positive (1) : Ce qu'on cherche (Spam, Maladie).

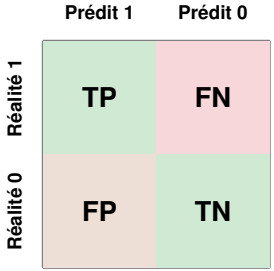
Vrai Positif (TP) : J'ai prédit 1, c'est 1.
"Alerte justifiée"

Vrai Négatif (TN) : J'ai prédit 0, c'est 0.
"Silence justifié"

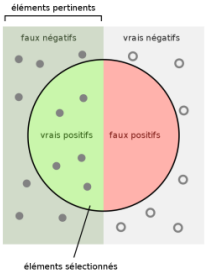
Faux Positif (FP) : J'ai prédit 1, c'est 0.
"Fausse Alarme" (Type I)

Faux Négatif (FN) : J'ai prédit 0, c'est 1.
"Loupé !" (Type II)

Matrice de Confusion



Le Dilemme : Précision vs Rappel



Combien de candidats sélectionnés sont pertinents ?

$$\text{Précision} = \frac{\text{vrais positifs}}{\text{éléments sélectionnés}}$$

Combien d'éléments pertinents sont sélectionnés ?

$$\text{Rappel} = \frac{\text{vrais positifs}}{\text{éléments pertinents}}$$

Précision (Qualité)

"Je ne veux pas de fausses alarmes."

$$\text{Precision} = \frac{TP}{TP + FP}$$

Rappel (Quantité)

"Je ne veux rien rater."

$$\text{Recall} = \frac{TP}{TP + FN}$$

La Synthèse : Le F1-Score

Problème : Si j'ai 100% de Précision mais 1% de Rappel, est-ce un bon modèle ?
La moyenne arithmétique donnerait 50.5%. C'est trompeur.

La Moyenne Harmonique

$$F1 = 2 \times \frac{\text{Précision} \times \text{Rappel}}{\text{Précision} + \text{Rappel}}$$

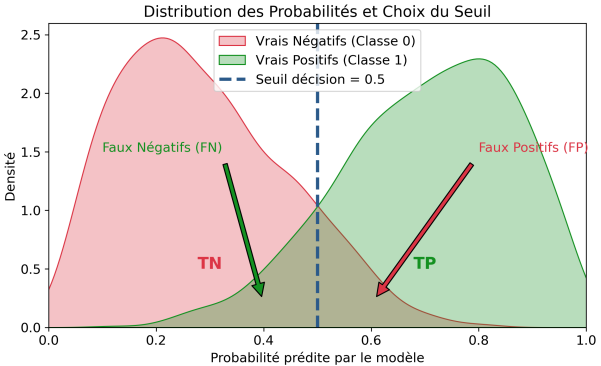
Comportement :
Le F1-Score est "sévère".
Il est proche de la **plus mauvaise** des deux notes.

→

✔ Pour avoir un bon F1, il faut être bon **partout**.

La Nuance : Probabilités et Seuils

⚙ Le modèle ne sort pas "0" ou "1", mais un **score** (ex : $P(y = 1|x) = 0.7$).
C'est nous qui fixons la barre (Seuil) pour décider.



Si je déplace le seuil vers la droite (0.9) :

Je deviens exigeant.

↑ Précision augmente.

↓ Rappel diminue.

Si je déplace le seuil vers la gauche (0.1) :

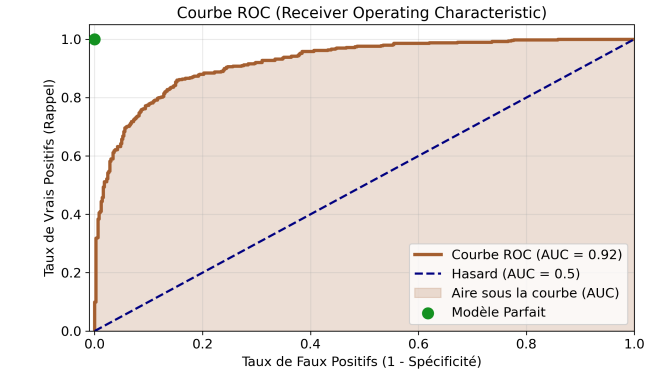
Je deviens paranoïaque.

↓ Précision diminue.

↑ Rappel augmente.

Performance Globale : Courbe ROC et AUC

Objectif : Évaluer le pouvoir discriminant du modèle, **tous seuils confondus**.



Analyse Mathématique

Les Axes (Compromis)

Y : Taux Vrais Positifs (Rappel).

X : Taux Faux Positifs (Fausses Alarmes).

↑ *On veut coller au coin Haut-Gauche.*

L'Aire sous la Courbe (AUC)

$$\text{AUC} = \int_0^1 \text{ROC}(x)\,dx$$

0.5 : Hasard (Diagonale).

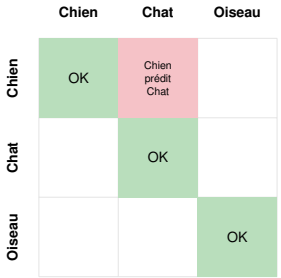
1.0 : Séparation Parfaite.

Probabilité de bien classer une paire (+/-).

Extension : Le défi du Multi-classes ($K > 2$)

Scénario : Classer des images en **Chien**, **Chat** ou **Oiseau**. (3×3).

Matrice de Confusion $K \times K$



1. Approche "One-vs-Rest"

On calcule la Précision/Rappel pour **chaque classe** séparément.
Ex : Précision "Chat" (contre tout le reste).

2. Agrégation (La Note Finale)

Comment résumer 3 scores ?

Micro-Avg : Moyenne globale (biaisé par la classe majoritaire).

Macro-Avg : Moyenne des scores de classes (donne du poids aux petites classes).

✔ **On préfère souvent la Macro-Avg.**

Références Bibliographiques I

T. Hastie, R. Tibshirani, J. Friedman (2009).
The Elements of Statistical Learning : Data Mining, Inference, and Prediction.
Springer Series in Statistics. 2nd Edition.

S. Shalev-Shwartz, S. Ben-David (2014).
Understanding Machine Learning : From Theory to Algorithms.
Cambridge University Press.

Scikit-Learn Developers.
User Guide : Machine Learning in Python.
Documentation officielle : `https://scikit-learn.org`

L. Breiman, J. Friedman, R. Olshen, C. Stone (1984).
Classification and Regression Trees.
Wadsworth International Group.