Apprentissage Automatique - L3 - KMAXPP05

Chapitre 1 : Bases de l'apprentissage

Cédric Campguilhem

Airbus Operations
cedric.campguilhem@airbus.com

2025-2026

Organisation du cours

Le cours est organisé autour de :

- Cours magistraux : 20 heures (2h x 10 sessions)
- Travaux dirigés : 16 heures (2h x 8 sessions)
- Travaux pratiques : 20 heures (2h x 10 sessions)

Les créneaux horaires suivants seront utilisés :

- Mardi 15h45 - 17h45 : TD / TP
- Mardi 18h - 20h : Contrôles
- Jeudi 13h30 - 15h30 : Cours / TP

Évaluations

Item	%	Durée	Format	Date
CC1	33%	30 min	En cours, QCM	19/03
CC2	34%	1h30	Conditions d'examen	19/05
TD & TP	33%	-	Présence, oral, projet	-
CC3	-	1h30	Conditions d'examen	11/06

Note : Le contrôle CC2 sera réalisé sur le créneau du mardi de 18h à 20h.

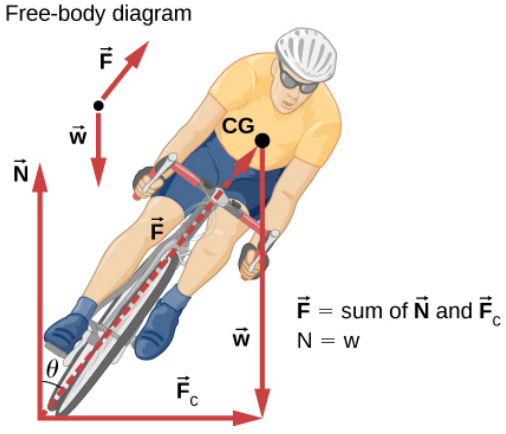
Plan

- Bases de l'apprentissage [6h]
  - Introduction
  - Typologie des problèmes d'apprentissage
  - Problématiques de l'apprentissage artificiel
  - Modèles linéaires
  - Validation d'algorithme d'apprentissage
- Algorithmes d'apprentissage [13h30]
  - Algorithme Bayésien naïf [1h30]
  - Analyse en composantes principales [2h]
  - Arbres de décision [2h]
  - Perceptrons et perceptrons multicouches [4h]
  - K moyennes [2h]
  - K plus proches voisins [2h]

Paradigmes et systèmes de pensée

Daniel Kahneman (prix Nobel d'économie 2002) a théorisé 2 modes de fonctionnement du cerveau humain :

- **La pensée lente** : consciente, logique, nécessitant un effort
- **La pensée rapide** : inconsciente, intuitive, sans effort



Approche logique

Approche intuitive

Source : Thinking, fast and slow - D. Kahneman - 2012 - Penguin

Paradigmes et systèmes de pensée

On peut établir un lien avec les paradigmes de programmation :

- La pensée lente $\rightarrow$ La programmation **déductive** : basée sur des règles logiques et explicites, issues des connaissances humaines. Le programme applique les règles aux données qui lui sont fournies.
- La pensée rapide $\rightarrow$ La programmation **inductive** : basée sur des données, l'expérience, les règles sont implicites et sont découvertes par l'algorithme. Le programme applique les règles qu'il a découvert à de nouvelles données, il généralise.

Définitions de l'apprentissage automatique

Définition 1

Champ d'étude qui donne aux ordinateurs la capacité d'apprendre sans être explicitement programmés.

— *Arthur Samuel (1959)*

Source : Some Studies in Machine Learning Using the Game of Checkers - A.L. Samuel - 1959 - IBM Journal

Définitions de l'apprentissage automatique

Définition 2

Un programme apprend d'une **expérience** E pour une **tâche** T avec une **performance** P, si P s'améliore avec E.

— *Tom Mitchell (1997)*

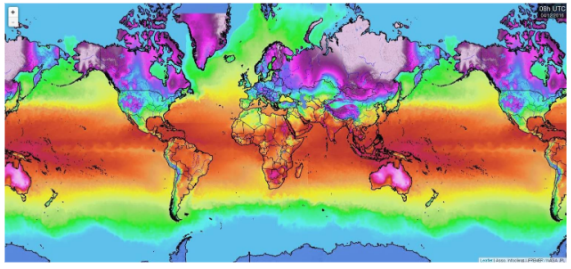
- **Expérience (E)** = Les données.
- **Tâche (T)** = Prédire, classer, regrouper, agir, générer.
- **Performance (P)** = La fonction de perte mathématique.

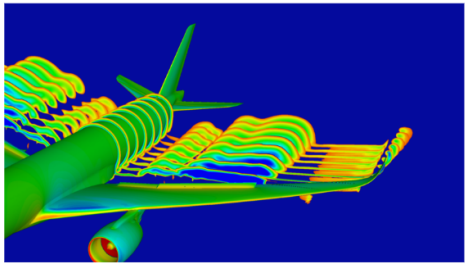
Source : Machine Learning - T.M. Mitchell - 1997 - McGraw-Hill

Tout est nombre !

Lorem Ipsum

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed eiusmod tempor incidunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquid ex ea commodi consequat. Quis aute iure reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint obcaecat cupiditat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.





Apprentissage automatique

On distingue 3 grandes familles d'algorithmes en apprentissage automatique :

- L'apprentissage **non-supervisé**
- L'apprentissage **supervisé**
- L'apprentissage par **renforcement**

Apprentissage non-supervisé

A partir d'un échantillon d'observations $\mathcal{D} = \left\{x^{(1)}, x^{(2)}, \cdots, x^{(n)}\right\}$, un algorithme d'apprentissage non-supervisé a pour objectif de découvrir la **structure interne** des données. Cela peut être réalisé :

- En regroupant ensemble des observations présentant des caractéristiques similaires : le **partitionnement**
- En trouvant une représentation simplifiée des observations avec une quantité réduite de caractéristiques : la **réduction de dimension**

Il n'y a, a priori, pas de réponse précise attendue de la part de l'algorithme.

Apprentissage non-supervisé

Voici quelques exemples d'algorithmes :

- Partitionnement :
  - Algorithme des K moyennes : `sklearn.cluster.KMeans`
  - DBSCAN (Partitionnement spatial d'applications fondé sur la densité avec gestion du bruit) : `sklearn.cluster.DBSCAN`
- Réduction de dimension :
  - Analyse en composantes principales (ACP) : `sklearn.decomposition.PCA`
  - t-SNE (plongement stochastique de voisins basé sur une loi de Student) : `sklearn.manifold.TSNE`

Exemple d'apprentissage non-supervisé

Considérons les données suivantes de joueurs de rugby :

Taille (cm)	**Poids** (kg)	**Age**	**Nombre de sélections**
190	112	29	47
172	85	22	5
186	135	37	0
180	92	30	70

- Un algorithme de partitionnement pourrait, par exemple, regrouper les joueurs les plus grands et lourds ensemble, ou ceux avec le plus de sélections.
- Un algorithme de réduction de dimension pourrait, par exemple, profiter de la corrélation entre la taille et le poids d'un côté et de l'âge et du nombre de sélections de l'autre pour proposer une représentation simplifiée qui serait basée sur seulement 2 caractéristiques, respectivement, le gabarit et l'expérience.

Apprentissage supervisé

A partir d'un échantillon d'observations $\mathcal{D} = \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \cdots, (x^{(n)}, y^{(n)})\}$, un algorithme d'apprentissage supervisé a pour objectif de découvrir la **relation** liant les caractéristiques d'entrée (x) et celles de sortie (y).

Deux familles se détachent basées sur la nature, continue ou discrète, des caractéristiques de sortie :

- Les algorithmes de **régression** : quand les caractéristiques de sortie représentent une grandeur continue
- Les algorithmes de **classification** : quand les caractéristiques de sortie représentent une grandeur discrète

Il y a, a priori, une réponse précise attendue de la part de l'algorithme.

Exemple de problème de régression

En restant dans notre exemple sur les joueurs de rugby :

Taille (cm)	**Poids** (kg)
190	112
172	85
186	135
180	92

Un algorithme d'apprentissage supervisé pourrait, par exemple, trouver une relation entre la taille et le poids d'un joueur. Ce qui permettrait de **prédire** le poids d'un joueur de rugby, non répertorié dans cette expérience, à partir de la connaissance de sa taille.

Exemple de problème de classification

Ne changeons pas une équipe qui gagne :

Taille (cm)	**Poids** (kg)	**Poste**
190	112	3ème ligne
172	85	demi d'ouverture
186	135	pilier
180	92	centre

Un algorithme d'apprentissage supervisé pourrait, par exemple, trouver une relation entre la taille et le poids d'un joueur d'un côté et le poste qu'il occupe de l'autre. Ce qui permettrait de **prédire** le poste d'un joueur de rugby, non répertorié dans cette expérience, à partir de la connaissance de sa taille et de son poids.

Apprentissage par renforcement

Contrairement aux modes précédents, l'apprentissage par renforcement ne repose pas sur un échantillon de données fixe mais sur des **interactions** dynamiques.

Un **agent** évolue dans un **environnement** et doit apprendre à prendre des décisions **séquentielles** :

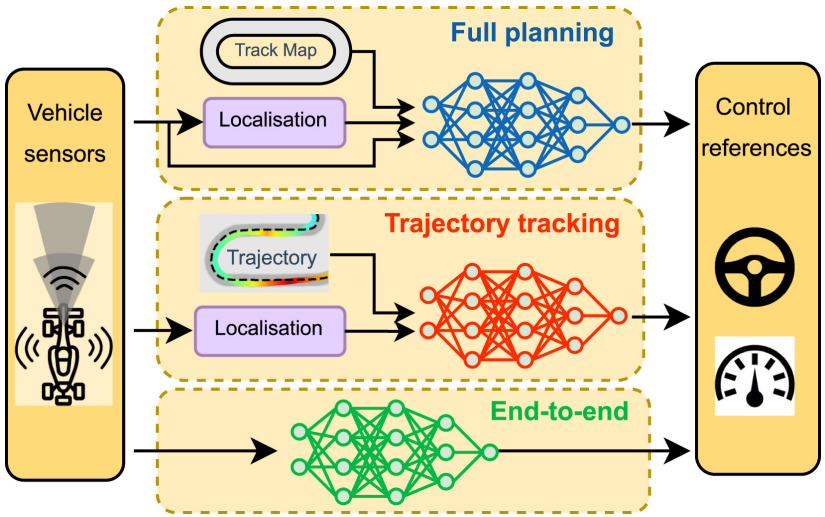
- L'agent observe un **état** (le contexte actuel)
- Il choisit une **action** à effectuer
- Il reçoit une **récompense** (positive ou négative, immédiate ou différée) en retour

L'objectif de l'algorithme est de découvrir la **stratégie optimale** permettant de maximiser la récompense totale à long terme.

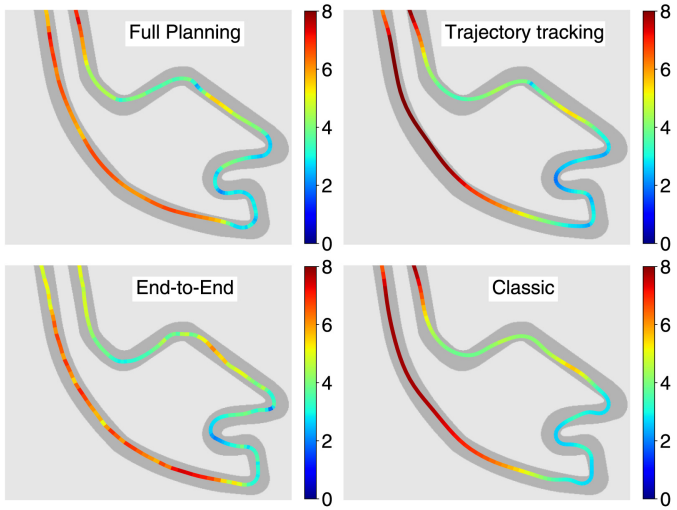
Il n'y a pas, a priori, de réponse précise attendue par l'algorithme. Mais il est possible d'évaluer, de manière précise, la qualité de sa réponse.

Exemple d'apprentissage par renforcement

Dans l'exemple ci-dessous on peut voir comment un agent interagit avec son environnement (la voiture et le circuit) par le biais de ses capteurs et des actions effectuées sur la direction du véhicule ainsi que sa vitesse. L'objectif est de trouver une trajectoire optimale (position et vitesse) afin de minimiser le temps au tour, sans sortir le véhicule du tracé (pénalité) :



Les interactions de l'agent



Les stratégies optimales

Source : Comparing deep reinforcement learning architectures for autonomous racing - B.D Evans, H.W. Jordaan, H.A Engelbrecht - Elsevier 2023

Problématique

Dans cette section nous introduirons des concepts fondamentaux de l'apprentissage supervisé :

- L'espace des hypothèses
- La hiérarchie des espaces d'apprentissage
- La formulation des problèmes de régression et classification

Espace des hypothèses

Soient $\mathcal{X}$ et $\mathcal{Y}$, respectivement l'espace des entrées et l'espace des sorties dont nous disposons pour notre problème d'apprentissage supervisé.

Hypothèse : fonction vraie f

On suppose qu'il existe une fonction **vraie** mais inconnue $f : \mathcal{X} \to \mathcal{Y}$ qui régit la relation entre les données, telle que :

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

ϵ représente un bruit aléatoire dans les caractéristiques observées (erreur de mesure), ou des caractéristiques non-observées.

Espace des hypothèses

Définition : espace des hypothèses $\mathcal{H}$

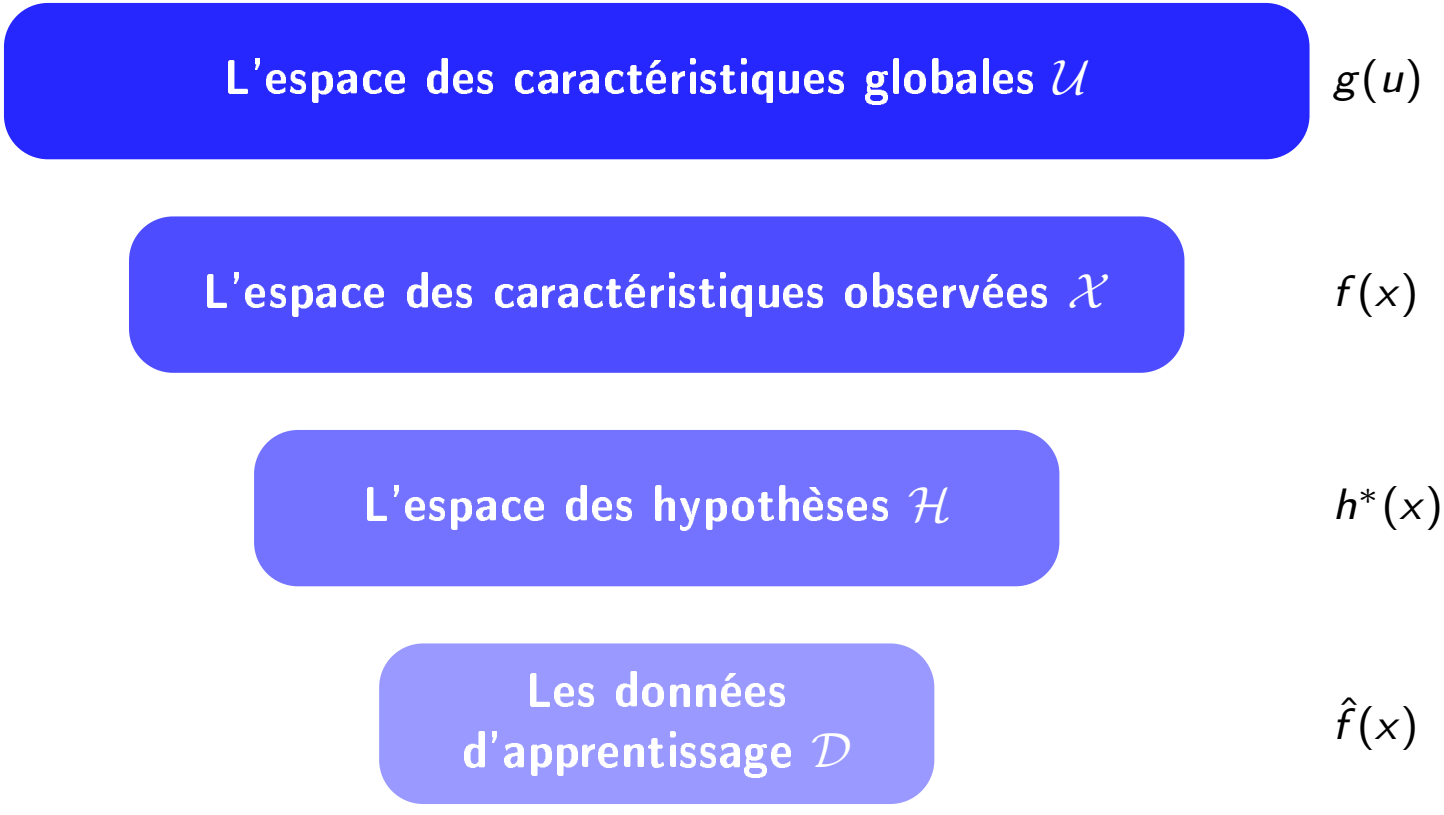
On définit l'**espace des hypothèses** $\mathcal{H}$ comme l'ensemble des fonctions candidates :

$$h : \mathcal{X} \to \mathcal{Y}$$

Le but de l'apprentissage est de trouver, à partir des données d'apprentissage $\mathcal{D}$ la *meilleure* fonction $h^* \in \mathcal{H}$.

Il nous reste à définir ce que *meilleure* signifie dans ce contexte.

Hiérarchie des espaces d'apprentissage



Hiérarchie des espaces d'apprentissage

Nous voulons prédire la vitesse d'une voiture en fonction de la consommation instantanée de carburant :

- La consommation instantanée est bien une caractéristique explicative de la vitesse.
- Mais il y a d'autres caractéristiques non-observées qui sont aussi pertinentes : poids du véhicule, vent, pente... La fonction vraie f sera donc un modèle très approché de la réalité g.
- L'espace des hypothèses $\mathcal{H}$ limite les fonctions candidates explorées, ce qui peut introduire des pertes complémentaires par rapport à la fonction vraie f.
- Enfin nous ne disposons que d'un ensemble limité de données d'apprentissage $\mathcal{D}$. Le mieux que nous puissions faire est d'obtenir une approximation de h^* que l'on note $\hat{f}$.

Espace des hypothèses et hiérarchie des espaces d'apprentissage

En résumé :

- On note g le **processus générateur** reliant les caractéristiques globales aux sorties.
- On note f la fonction **vraie** reliant l'espace $\mathcal{X}$ à l'espace $\mathcal{Y}$.
- On note h^* la **meilleure** fonction de l'espace $\mathcal{H}$ reliant ces mêmes espaces.
- On note $\hat{f} \in \mathcal{H}$ l'**estimation** de f après entraînement de l'algorithme à partir des données $\mathcal{D}$.

Apprentissage supervisé

Problématique de l'apprentissage supervisé

L'objectif de l'apprentissage supervisé est de trouver une fonction $\hat{f} : \mathcal{X} \to \mathcal{Y}$ dans l'espace des hypothèses $\mathcal{H}$ qui soit un bon **estimateur** de la fonction **vraie** f à partir des données $\mathcal{D} = \left\{(x^{(i)}, y^{(i)})\right\}_{i=1}^{n}$:

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

Où ϵ représente une incertitude liée au bruit de mesure des données observées ou à des variables explicatives non-observées.

On note $\hat{y} = \hat{f}(x)$ la prédiction réalisée par le modèle pour une entrée x.

Un bon estimateur $\hat{f}$ est tel qu'il minimise une **fonction de perte** $L(y, \hat{y})$ sur l'ensemble des données observées $\mathcal{D}$, aussi appelées données d'apprentissage.

Problème de régression

Problématique de la régression

Dans un problème de **régression**, la variable de sortie est quantitative : $\mathcal{Y} \subseteq \mathbb{R}$.

La fonction de perte L mesure une distance entre la prédiction de l'algorithme $\hat{y}$ et la valeur observée y.

Un exemple de fonction de perte est l'erreur quadratique :

$$L(y, \hat{y}) = (y - \hat{y})^2$$

Problème de classification

Problématique de la classification binaire

Dans un problème de **classification binaire**, la variable de sortie est qualitative : $\mathcal{Y} = \{0, 1\}$.

La fonction de perte L est un compteur d'erreurs de classification.

Un exemple de fonction de perte serait la fonction 0-1 :

$$\begin{cases} L(y, \hat{y}) = 0 & \text{si } y = \hat{y} \\ L(y, \hat{y}) = 1 & \text{si } y \neq \hat{y} \end{cases}$$

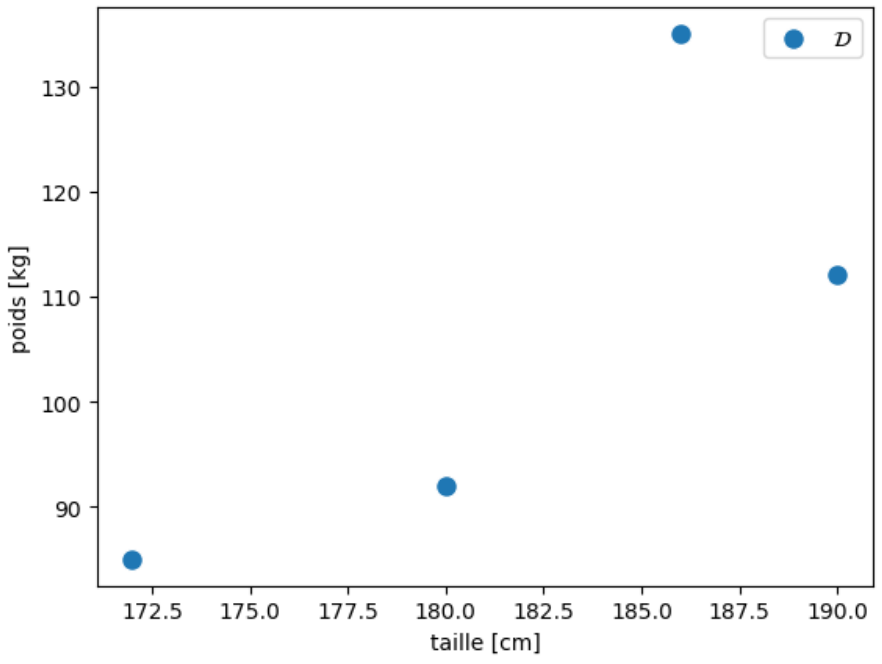
- **Note sur l'incertitude :** Ici, $\epsilon = y - f(x)$ représente une incertitude discrète prenant ses valeurs dans $\{-1, 0, 1\}$.
- **Note sur l'optimisation :** La fonction 0-1 n'est pas dérivable. En pratique, nous utiliserons des substituts (comme la log-vraisemblance).

Modèles linéaires

Dans cette section, nous étudierons 2 modèles linéaires pour des problèmes de régression et de classification :

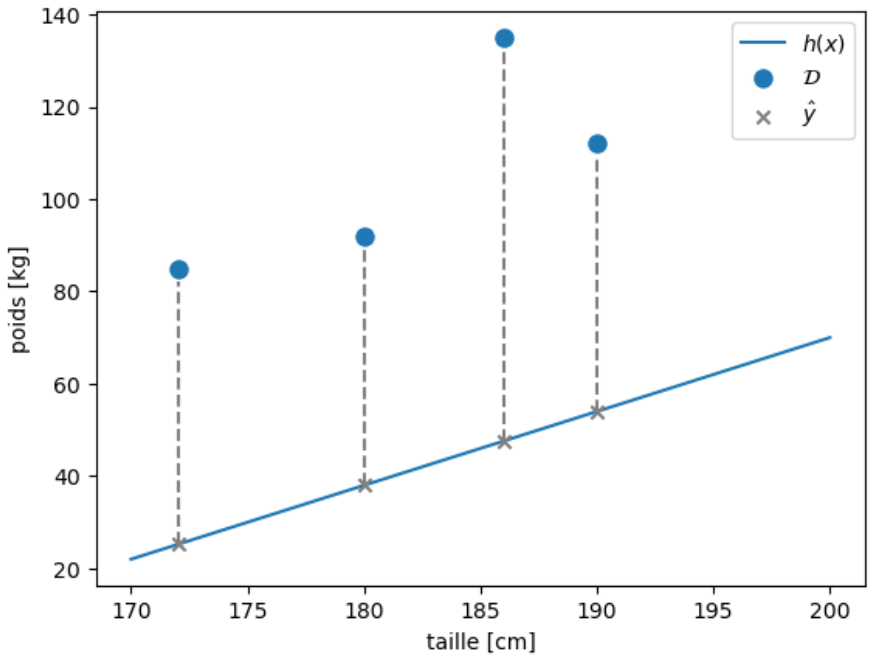
- La **régression linéaire** pour les problèmes de régression.
- La **régression logistique** pour les problèmes de classification.

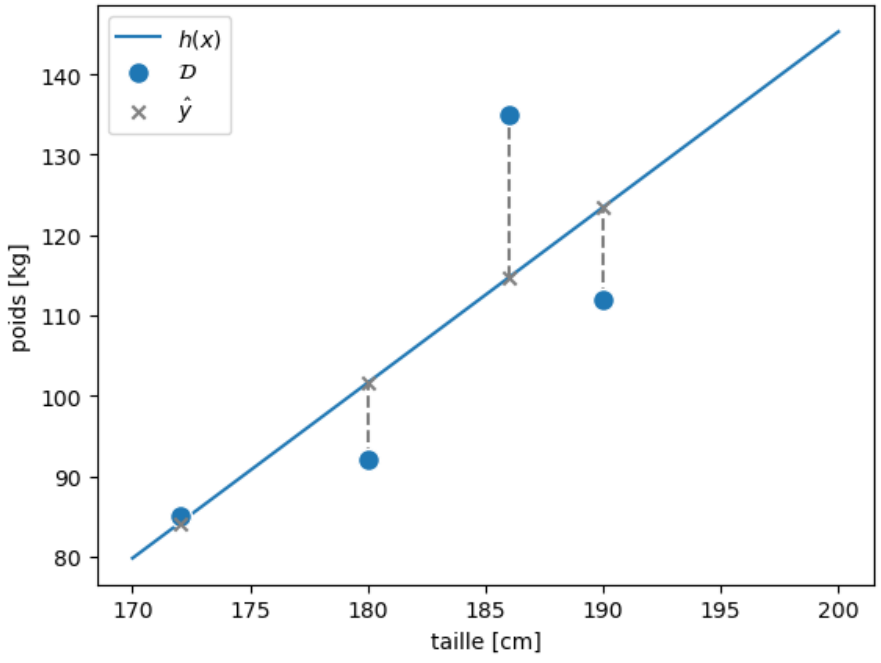
Régression linéaire



Taille (cm)	**Poids** (kg)
190	112
172	85
186	135
180	92

Exemples





L'idée est de construire une droite qui passe le plus proche possible des observations $\mathcal{D}$. L'algorithme tente de minimiser la distance totale entre les observations et les prédictions.

Modèle

Le modèle de régression linéaire peut s'écrire sous une forme simple :

$$\text{poids}^{(i)} \approx \beta_0 + \beta_1 \times \text{taille}^{(i)}$$

Une forme équivalente, vectorielle, serait :

$$\text{poids}^{(i)} \approx \begin{pmatrix} 1 & \text{taille}^{(i)} \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix}$$

Modèle

L'avantage de cette forme vectorielle, c'est qu'elle peut évoluer vers une forme matricielle nous permettant de faire plusieurs prédictions à la fois, grâce à un empilement :

$$\begin{pmatrix} \text{poids}^{(1)} \\ \text{poids}^{(2)} \\ \text{poids}^{(3)} \\ \text{poids}^{(4)} \end{pmatrix} \approx \begin{pmatrix} 1 & \text{taille}^{(1)} \\ 1 & \text{taille}^{(2)} \\ 1 & \text{taille}^{(3)} \\ 1 & \text{taille}^{(4)} \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix}$$

Et sous une forme plus condensée :

$$Y \approx \tilde{X}\beta$$

Formulation

Soient $\mathcal{X} \subseteq \mathbb{R}^d$ l'espace des caractéristiques d'entrée, $\mathcal{Y} \subseteq \mathbb{R}$ l'espace des caractéristiques de sortie, et $\mathcal{D} = \left\{\left(x^{(i)}, y^{(i)}\right)\right\}_{i=1}^{n}$ les données d'apprentissage. On définit l'application Φ qui réalise le passage à l'espace augmenté $\tilde{\mathcal{X}} \subseteq \mathbb{R}^{d+1}$:

$$\begin{aligned} \Phi : \mathcal{X} &\to \tilde{\mathcal{X}} \\ x^{(i)} &\mapsto \tilde{x}^{(i)} = \begin{pmatrix} 1 & x^{(i)} \end{pmatrix} \end{aligned}$$

L'espace des hypothèses $\mathcal{H} : \mathcal{X} \to \mathcal{Y}$ dans le cadre de la régression linéaire est :

$$\mathcal{H} = \left\{ h : x \mapsto \Phi(x)\beta \mid \beta \in \mathbb{R}^{d+1} \right\}$$

Empilement matriciel

Pour des données d'apprentissage $\mathcal{D}$, on construit une matrice de design $\tilde{X}$ par empilement vertical des observations augmentées $\tilde{x}^{(i)}$:

$$\tilde{X} = \begin{pmatrix} \tilde{x}^{(1)} \\ \tilde{x}^{(2)} \\ \vdots \\ \tilde{x}^{(n)} \end{pmatrix} \in \mathcal{M}_{n,d+1}(\mathbb{R})$$

On note Y l'empilement vertical des observations des caractéristiques de sortie :

$$Y = \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(n)} \end{pmatrix} \in \mathcal{M}_{n,1}(\mathbb{R})$$

Méthode des moindres carrés

La méthode des moindres carrés cherche à minimiser la somme des erreurs quadratiques :

$$J(\beta) = \sum_{i=1}^{n} \left(y^{(i)} - \tilde{x}^{(i)} \beta \right)^2$$
$$= \left(Y - \tilde{X}\beta \right)^T \left(Y - \tilde{X}\beta \right)$$

$\hat{\beta}$ est un estimateur de β minimisant cette somme des erreurs quadratiques :

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^{d+1}}{\arg\min} \left(\left(Y - \tilde{X}\beta \right)^T \left(Y - \tilde{X}\beta \right) \right)$$

Méthode des moindres carrés

Il existe une solution analytique à ce problème d'optimisation :

$$\hat{\beta} = \left(\tilde{X}^T \tilde{X}\right)^{-1} \tilde{X}^T Y$$

$$\hat{\beta} \in \mathcal{M}_{d+1,1}(\mathbb{R})$$

Pour exister, cette solution nécessite que la matrice $\left(\tilde{X}^T \tilde{X}\right)$ soit inversible. Ce qui signifie que les colonnes de $\tilde{X}$ sont linéairement indépendantes. La fonction d'estimation $\hat{f}$ est donc :

$$\hat{f} : \mathcal{X} \to \mathcal{Y}$$

$$x \mapsto \Phi(x)\hat{\beta}$$

Hypothèses de la méthode des moindres carrés

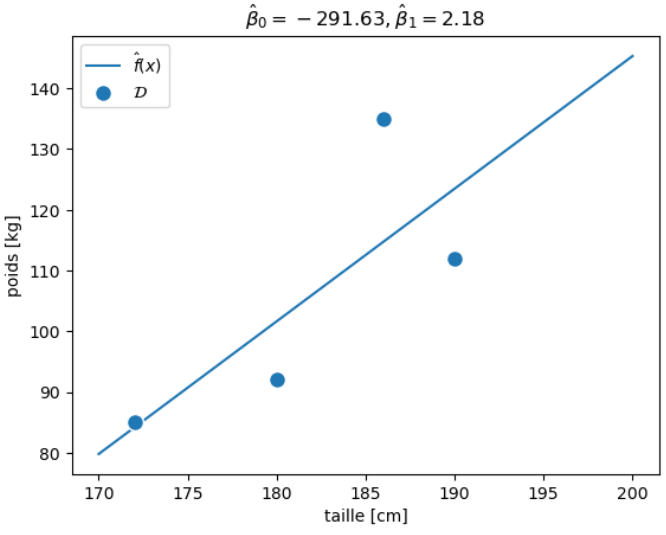
La méthode des moindres carrés prend un certain nombre d'hypothèses pour l'incertitude ϵ afin de dériver la matrice de covariance de $\hat{\beta}$. Pour rappel, en apprentissage supervisé nous cherchons une fonction estimant la fonction vraie f définie comme :

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

Notons X l'empilement matriciel des observations de caractéristiques d'entrées. ϵ doit satisfaire les conditions suivantes :

- La condition d'**exogénéité** : $\mathbb{E}[\epsilon \mid X] = 0$.
  - C'est la condition qui garantit que $\mathbb{E}[\hat{\beta}] = \beta$. Ici β est celui de h^*.
  - Il n'existe aucun lien fonctionnel entre l'espérance de ϵ et les données d'entrée.
- La condition d'**homoscédasticité** : $\mathrm{Var}[\epsilon \mid X] = \sigma_\epsilon^2$.
  - Il n'existe aucun lien fonctionnel entre la variance de ϵ et les données d'entrée.
- La condition d'**indépendance** : $\mathrm{Cov}[\epsilon^{(i)}, \epsilon^{(j)}] = 0 \quad \forall i \neq j$

Hiérarchie des espaces d'apprentissage et régression linéaire



Le modèle linéaire est loin d'être parfait :

- Soit l'hypothèse de linéarité n'est pas la bonne. L'espace des hypothèses $\mathcal{H}$ n'est pas le plus approprié.
- Soit il manque des caractéristiques, par exemple la densité, le tour de hanche, le rapport masse musculaire sur masse grasse...
- Soit il y a un bruit de mesure important dans la taille et/ou le poids.
- Soit une combinaison quelconque des éléments ci dessus !

Espaces des hypothèses et régression linéaire

Il aurait été possible de considérer une autre application Φ' pour réaliser le passage de l'espace d'entrée $\mathcal{X} \subseteq \mathbb{R}$ à l'espace augmenté $\tilde{\mathcal{X}}' \subseteq \mathbb{R}^3$, par exemple :

$$\Phi' : \mathcal{X} \to \tilde{\mathcal{X}}'$$
$$x^{(i)} \mapsto \tilde{x}^{(i)} = \begin{pmatrix} 1 & x^{(i)} & \left(x^{(i)}\right)^2 \end{pmatrix}$$

La régression linéaire devient en fait une régression polynomiale de degré 2. C'est donc un nouvel espace des hypothèses $\mathcal{H}'$ qui est exploré :

$$\mathcal{H}' = \left\{ h : x \mapsto \Phi'(x)\beta \mid \beta \in \mathbb{R}^3 \right\}$$

Espaces des hypothèses et régression linéaire

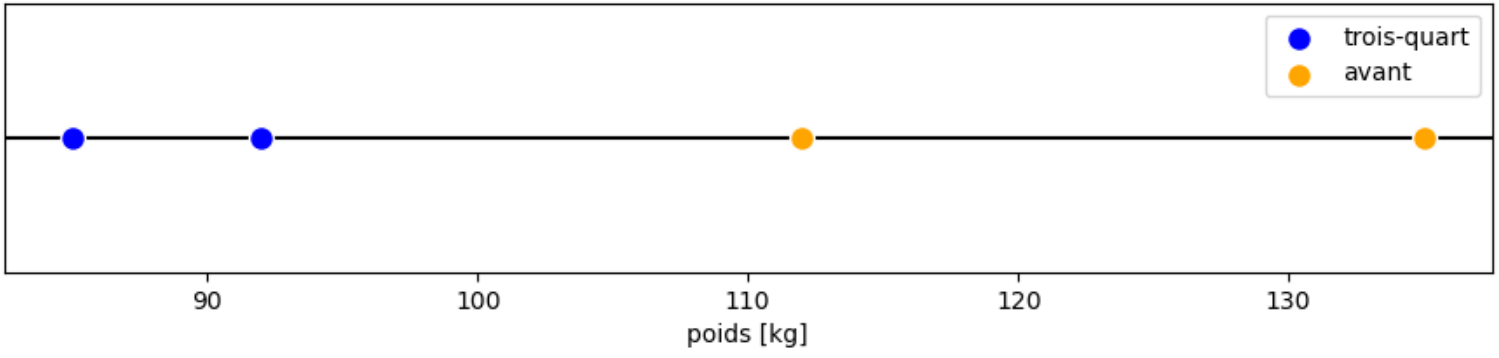
Dans le cadre de la régression linéaire, la nature de l'application Φ est un **hyper-paramètre** de l'algorithme d'apprentissage automatique.

Les coefficients β du polynôme sont les **paramètres** de l'algorithme. Ce sont les paramètres qui sont explorés par l'espace des hypothèses $\mathcal{H}$.

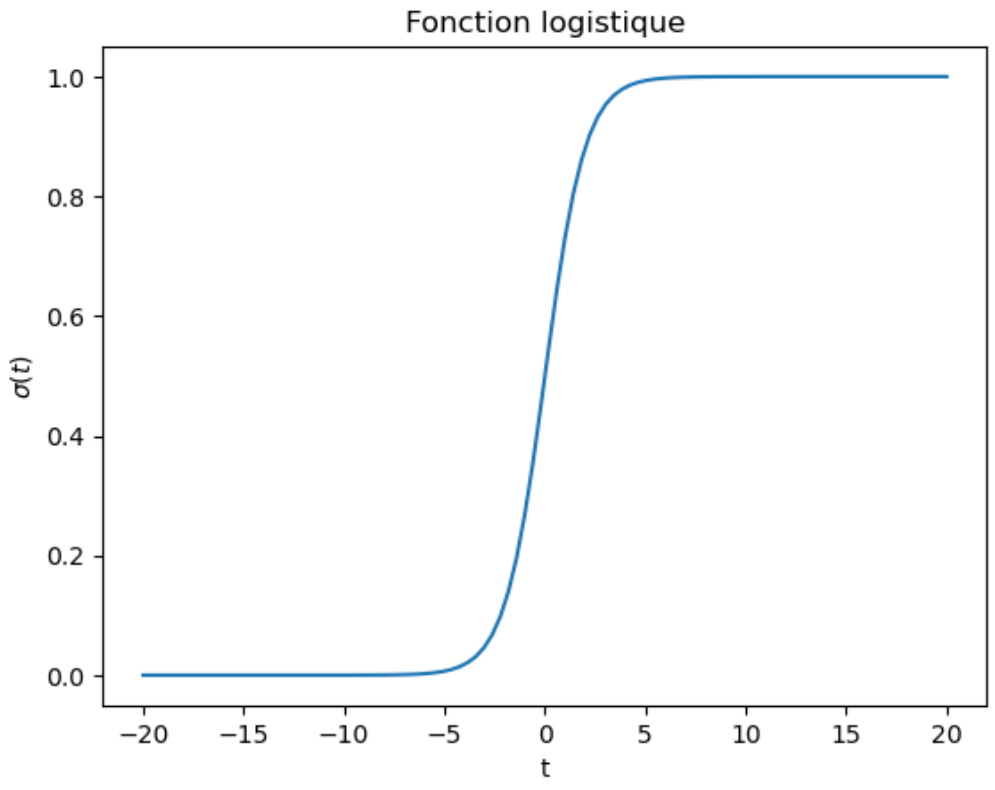
La méthode des moindres carrés permet d'obtenir une estimation $\hat{f}$ de la fonction vraie f parmi les fonctions h de l'espace $\mathcal{H}$.

Régression logistique

Poids (kg)	**Position**
112	avant = 1
85	trois-quart = 0
135	avant = 1
92	trois-quart = 0



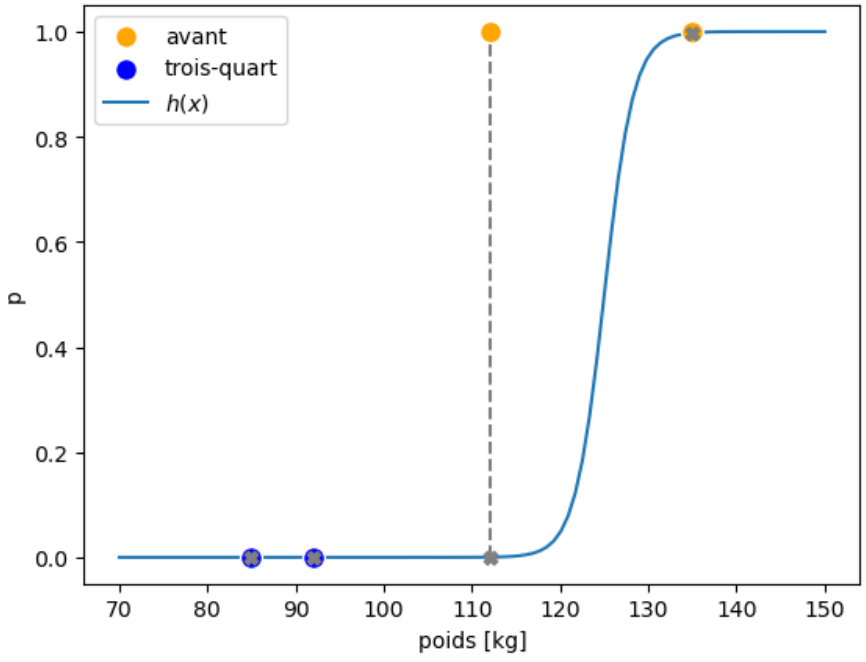
Fonction logistique

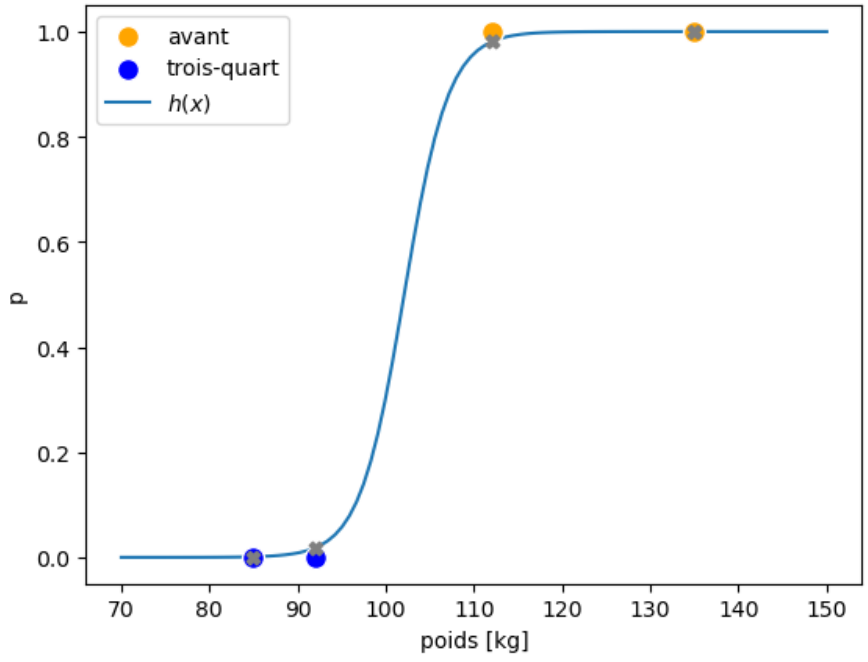


$$\sigma : \mathbb{R} \to]0, 1[$$
$$t \mapsto \frac{1}{1 + e^{-t}}$$

L'idée est d'associer les avants au plateau supérieur de cette fonction ($\sigma(t) = 1$) et d'associer les trois-quarts au plateau inférieur ($\sigma(t) = 0$).

Exemples





Cette régression logistique calcule en fait une probabilité p, plus spécifiquement la probabilité qu'un joueur d'un poids donné soit un avant.

Modèle

Le modèle de régression logistique peut s'écrire sous une forme relativement simple :

$$p^{(i)} \approx \frac{1}{1 + e^{-\left(\beta_0 + \beta_1 \times \text{poids}^{(i)}\right)}}$$

On peut le voir comme un modèle linéaire, dans l'exponentielle, combiné à une fonction qui envoie les résultats dans l'espace des probabilités $]0, 1[$.

Modèle

Et en notation matricielle :

$$p \approx \frac{1}{1 + e^{-\tilde{X}\beta}}$$

En condensant avec la fonction logistique σ :

$$p \approx \sigma\left(\tilde{X}\beta\right)$$

Formulation

Soient $\mathcal{X} \subseteq \mathbb{R}^d$ l'espace des caractéristiques d'entrée, $\mathcal{Y} = \{0, 1\}$ l'espace des caractéristiques de sortie, et $\mathcal{D} = \left\{\left(x^{(i)}, y^{(i)}\right)\right\}_{i=1}^{n}$ les données d'apprentissage. On définit l'application Φ qui réalise le passage à l'espace augmenté $\tilde{\mathcal{X}} \subseteq \mathbb{R}^{d+1}$:

$$\begin{aligned} \Phi : \mathcal{X} &\to \tilde{\mathcal{X}} \\ x^{(i)} &\mapsto \tilde{x}^{(i)} = \begin{pmatrix} 1 & x^{(i)} \end{pmatrix} \end{aligned}$$

L'espace des hypothèses $\mathcal{H} : \mathcal{X} \to [0, 1]$ dans le cadre de la régression logistique est :

$$\mathcal{H} = \left\{ h : x \mapsto \frac{1}{1 + e^{-\Phi(x)\beta}} \mid \beta \in \mathbb{R}^{d+1} \right\}$$

Hypothèses et Approche itérative

Contrairement à la régression linéaire, il n'existe pas de solution analytique au problème. Pour déterminer un $\hat{\beta}$ nous allons donc avoir besoin d'utiliser une approche **itérative**.

Loi de Bernoulli

Une approche pour résoudre le problème consiste à considérer chaque observation $y^{(i)}$ comme la réalisation d'une variable aléatoire $Y^{(i)}$ suivant une **loi de Bernoulli** de paramètre $p^{(i)}$. Les observations sont indépendantes mais ne sont pas identiquement distribuées.

Pour rappel, une variable aléatoire $Y^{(i)}$ suivant une loi de Bernoulli de paramètre $p^{(i)}$ est telle que :

$$\begin{cases} P(Y^{(i)} = 1) = p^{(i)} \\ P(Y^{(i)} = 0) = 1 - p^{(i)} \end{cases}$$

Ou de manière condensée avec $y \in \{0, 1\}$:

$$P(Y^{(i)} = y^{(i)}) = \left(p^{(i)}\right)^{y^{(i)}} \left(1 - p^{(i)}\right)^{1-y^{(i)}}$$

Maximisation de la vraisemblance

La vraisemblance de notre échantillon $\mathcal{D}$ est alors :

$$\mathcal{L}(\beta) = P(Y = \{y^{(i)}\}_{i=1}^{n} \mid \beta, \tilde{X})$$

L'idée consiste à trouver $\hat{\beta}$ tel qu'il maximise la vraisemblance des observations :

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^{d+1}}{\arg\max}\, \mathcal{L}(\beta)$$

Cette méthode de maximisation de la vraisemblance suppose que les conditions d'exogénéité et d'indépendance des $\epsilon^{(i)}$ sont satisfaites afin d'obtenir le meilleur estimateur possible.

Intuition

Prenons un "avant", à qui nous avons associé la valeur $y^{(i)} = 1$. La variable aléatoire est $Y^{(i)}$. Si la loi de Bernoulli associée est bien faite on devrait avoir $P(Y^{(i)} = 1)$ très proche de 1 et $P(Y^{(i)} = 0)$ très proche de 0.

Pour un "trois-quart", à qui nous avons associé la valeur $y^{(i)} = 0$, c'est l'inverse. On voudra avoir $P(Y^{(i)} = 1)$ très proche de 0 et $P(Y^{(i)} = 0)$ très proche de 1.

Ceci est équivalent à dire que pour un "avant" on veut que le paramètre $p^{(i)}$ de la loi de Bernoulli soit proche de 1 et que pour un arrière, on veut que ce même paramètre soit proche de 0.

Intuition

$p^{(i)}$ étant la prédiction de notre régression logistique, c'est à dire la probabilité que l'observation i soit un avant, maximiser la vraisemblance de l'ensemble des variables de Bernoulli revient à avoir une régression logistique bien calibrée !

L'ensemble de ces lois de Bernoulli est en fait contrôlé par un vecteur paramètre β **commun** à toutes les lois. Notre problème est donc sur-contraint, comme l'était la régression linéaire dans l'exemple précédent.

Log-vraisemblance négative moyenne

En pratique, on préfère minimiser une log-vraisemblance négative moyenne que maximiser une vraisemblance :

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^{d+1}}{\arg\min} \; -\frac{1}{n} \ln \left(\mathcal{L}(\beta) \right)$$

Comme les observations sont indépendantes, cette log-vraisemblance négative moyenne est en fait :

$$-\frac{1}{n} \ln \left(\mathcal{L}(\beta) \right) = -\frac{1}{n} \sum_{i=1}^{n} \ln \left(P(y^{(i)} \mid \tilde{x}^{(i)}, \beta) \right)$$

Entropie croisée moyenne

Cette log-vraisemblance négative moyenne prend une valeur très particulière et caractéristique. Pour rappel $p^{(i)} = \sigma(\tilde{x}^{(i)}\beta)$:

$$
\begin{aligned}
-\frac{1}{n}\ln\left(\mathcal{L}(\beta)\right) &= -\frac{1}{n}\sum_{i=1}^{n}\ln P(y^{(i)} \mid \tilde{x}^{(i)}, \beta) \\
&= -\frac{1}{n}\sum_{i=1}^{n}\ln\left(\left(p^{(i)}\right)^{y^{(i)}}\left(1-p^{(i)}\right)^{1-y^{(i)}}\right) \\
&= -\frac{1}{n}\sum_{i=1}^{n} y^{(i)}\ln\left(p^{(i)}\right) + (1-y^{(i)})\ln\left(1-p^{(i)}\right)
\end{aligned}
$$

Cette quantité est appelée **entropie croisée moyenne** !

Gradient d'entropie croisée

On pourrait montrer que la dérivée de la log-vraisemblance négative moyenne par rapport à β s'écrit :

$$\nabla_\beta \left(-\frac{1}{n} \ln \left(\mathcal{L}(\beta) \right) \right) = -\frac{1}{n} \sum_{i=1}^{n} \left(y^{(i)} - \sigma(\tilde{x}^{(i)} \beta) \right) \tilde{x}^{(i)T}$$

$$= -\frac{1}{n} \tilde{X}^T \left(Y - \sigma \left(\tilde{X} \beta \right) \right)$$

Ce gradient fait apparaitre 2 termes :

- $\left(Y - \sigma \left(\tilde{X} \beta \right) \right)$ qui est la différence entre la probabilité associée à la valeur observée et la probabilité prédite par la régression logistique pour un vecteur paramètre β donné.
- $\tilde{X}^T$ sont juste les caractéristiques d'entrée observées (et augmentées).

Descente de gradient

La minimisation de la log-vraisemblance négative moyenne peut être effectuée à l'aide d'une méthode de descente de gradient. La fonction à minimiser est strictement convexe (si les colonnes de $\tilde{X}$ sont linéairement indépendantes). Donc la descente de gradient est garantie de trouver un minimum **global**.

On notera $\alpha \in \mathbb{R}^+$ le taux d'apprentissage. Une valeur trop grande peut conduire à une instabilité numérique de la descente de gradient et une valeur trop faible peut conduire à une convergence trop lente. Une valeur de $\alpha = 0.001$ est typique pour débuter.

On notera $\epsilon \in \mathbb{R}^+$ la tolérance pour stopper la descente de gradient, par exemple $\epsilon = 10^{-6}$.

Descente de gradient

L'algorithme de descente de gradient peut être décrit comme suit :

1. Prendre une valeur initiale de $\hat{\beta}_{k=0} = 0_{d+1}$
2. Boucler jusqu'à convergence :
   - Calculer les n probabilités $p_k = \sigma(\tilde{X}\beta_k)$
   - Calculer le gradient $\nabla_\beta \left(-\frac{1}{n} \ln \mathcal{L}(\beta)\right)$
   - Calculer l'incrément de $\hat{\beta}$: $\delta_k = -\alpha \nabla_\beta \left(-\frac{1}{n} \ln \mathcal{L}(\beta)\right)$
   - Mettre à jour $\hat{\beta}$: $\hat{\beta}_{k+1} \leftarrow \hat{\beta}_k + \delta_k$
   - Calculer la norme Euclidienne au carré $\|\delta_k\|_2^2$
   - Si $\|\delta_k\|_2^2 < \epsilon$ alors la convergence est atteinte.

Synthèse : Linéaire vs Logistique

Caractéristique	**Régression Linéaire**	**Régression Logistique**
Nature de Y	Continue ($\mathbb{R}$)	Discrète ($\{0, 1\}$)
Modèle	$y \approx \tilde{X}\beta$	$p \approx \sigma(\tilde{X}\beta)$
Fonction de perte	Erreur Quadratique	Entropie Croisée
Objectif	Minimiser	Minimiser
Résolution	**Analytique** (directe)	**Numérique** (itérative)
Solution	$\hat{\beta} = (\tilde{X}^T\tilde{X})^{-1}\tilde{X}^T Y$	Descente de gradient

Note : Pour un problème de classification, minimiser une Entropie Croisée est équivalent à maximiser une Log-Vraisemblance. Pour un problème de régression, dans la condition où l'incertitude ϵ est Gaussienne, minimiser l'erreur quadratique moyenne est équivalent à maximiser la Log-Vraisemblance.

Sélection et Validation de modèles

Dans cette section nous discuterons de la sélection et de la validation de modèles pour un problème d'apprentissage supervisé :

- **Sélection** : quel est le meilleur modèle parmi plusieurs espaces d'hypothèses ?
- **Validation** : comment s'assurer que notre modèle a une performance suffisante ?

Décomposition de l'erreur

Pour mieux comprendre comment faire une sélection et une validation du modèle, nous devons passer un peu plus de temps sur la **décomposition de l'erreur** d'un modèle.

Risque empirique

Minimiser la fonction de perte L sur les données d'apprentissage $\mathcal{D}$ revient à minimiser le risque **empirique** défini comme suit :

Définition : Risque empirique

Le risque empirique est une fonction $R_n : \mathcal{H} \to \mathbb{R}^+$ qui mesure la performance de toute hypothèse $h \in \mathcal{H}$ sur les données d'apprentissage $\mathcal{D}$:

$$R_n = \frac{1}{n} \sum_{i=1}^{n} L\left(y^{(i)}, h\left(x^{(i)}\right)\right)$$

Risque empirique

L'apprentissage de l'algorithme supervisé permet d'obtenir l'estimateur $\hat{f}$ qui minimise le risque empirique à partir des données $\mathcal{D}$:

$$\hat{f} = \arg\min_{h \in \mathcal{H}} R_n(h)$$

Risque empirique pour la régression linéaire

Dans le cas de la régression linéaire, la fonction de perte est la fonction d'erreur quadratique. Chercher un estimateur $\hat{f}$ correspond donc à minimiser :

$$\hat{f} = \arg\min_{h \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} \left(y^{(i)} - \hat{y}^{(i)} \right)^2$$

Ce qui est équivalent à ce que l'algorithme des moindre carrés réalise (au coefficient $1/n$ près).

Risque empirique pour la régression logistique

Dans le cas de la régression logistique, la fonction de perte est l'entropie croisée moyenne (log-vraisemblance négative moyenne). Chercher un estimateur $\hat{f}$ correspond donc à minimiser :

$$\hat{f} = \arg\min_{h \in \mathcal{H}} \; -\frac{1}{n} \sum_{i=1}^{n} y^{(i)} \ln\left(p^{(i)}\right) + (1 - y^{(i)}) \ln\left(1 - p^{(i)}\right)$$

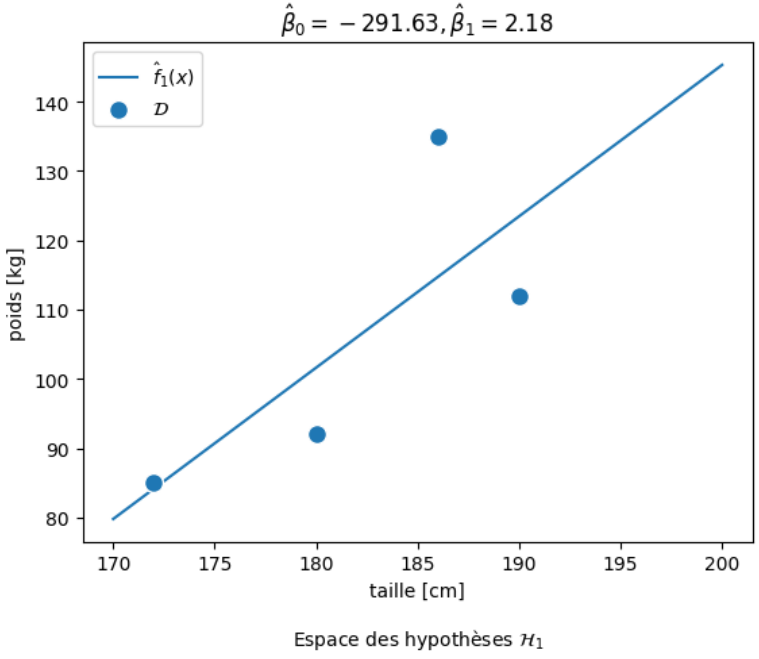
C'est la quantité que nous avons minimisé plus tôt grâce à la descente de gradient.

Comparaison de modèles et d'hypothèses

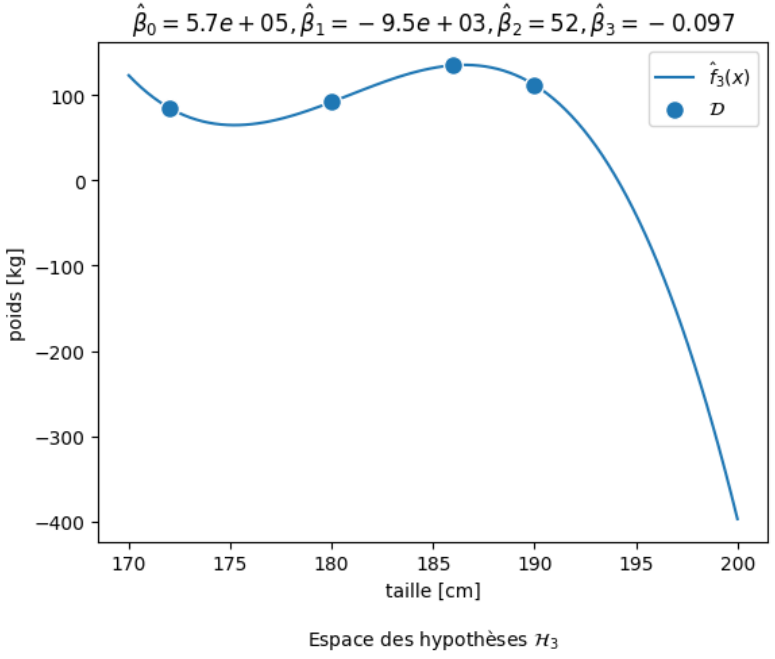
Reprenons le problème de régression linéaire pour prédire le poids d'un joueur à partir de sa taille et considérons deux espaces des hypothèses :

- $\mathcal{H}_1$: l'espace des hypothèses pour des polynômes mono-variés de degré 1.
- $\mathcal{H}_3$: l'espace des hypothèses pour des polynômes mono-variés de degré 3.

Quel est le meilleur modèle ? ou la meilleure hypothèse ?



Espace des hypothèses $\mathcal{H}_1$



Espace des hypothèses $\mathcal{H}_3$

Quel est le meilleur modèle ? ou la meilleure hypothèse ?

- Pour une régression linéaire on cherche à minimiser l'erreur quadratique...
- ...mais ce n'est pas le **but**, c'est un **moyen**. Le but, c'est de trouver une **relation** entre les caractéristiques d'entrées et de sortie qui soit la plus proche possible de la fonction vraie $f(x)$ et du processus générateur $g(u)$.
- Pour l'espace des hypothèses $\mathcal{H}_1$, $\hat{\beta}_0$ n'a pas d'interprétation tangible : un joueur qui mesurerait 0 cm pèserait -291 kg ? En revanche $\hat{\beta}_1$ nous informe qu'en prenant 1 cm, le poids augmente en moyenne de 2 kg. Ce qui ne paraît pas absurde.
- Pour l'espace des hypothèses $\mathcal{H}_3$, le modèle montre qu'on perd du poids jusqu'à 175 cm, puis on gagne du poids jusqu'à 185 cm, avant, à nouveau, de reperdre du poids au-delà de 185 cm. Cela semble peu plausible !

Quel est le meilleur modèle ? ou la meilleure hypothèse ?

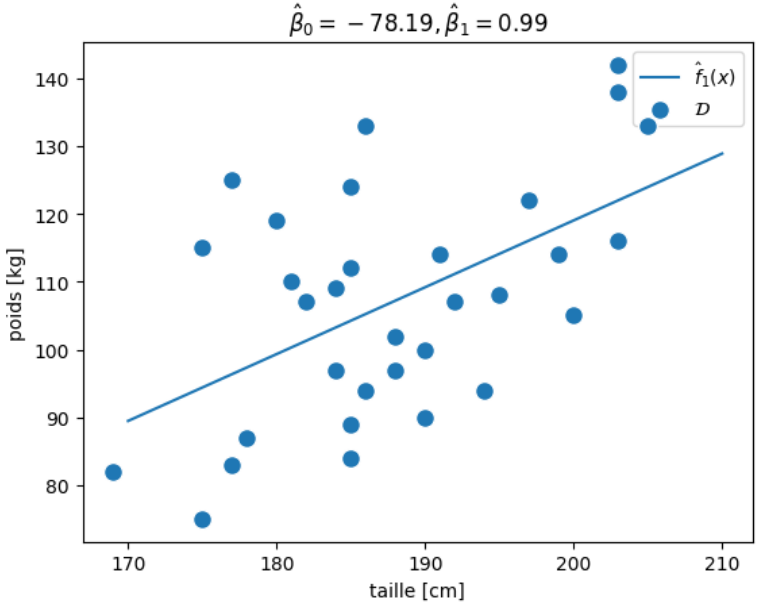
- L'hypothèse $\mathcal{H}_1$ est meilleure ici.
- Notre **connaissance a priori** du problème nous permet de discriminer la meilleure hypothèse. Que faire quand nous n'avons pas de connaissances a priori ?
- On échange avec des **experts** du problème.
- Mais on ne peut vraiment *rien* faire avec les mathématiques ?
- *Si*, on peut ! (cela ne dispense pas de s'entourer des experts du domaine).

Quel est le meilleur modèle ? ou la meilleure hypothèse ?

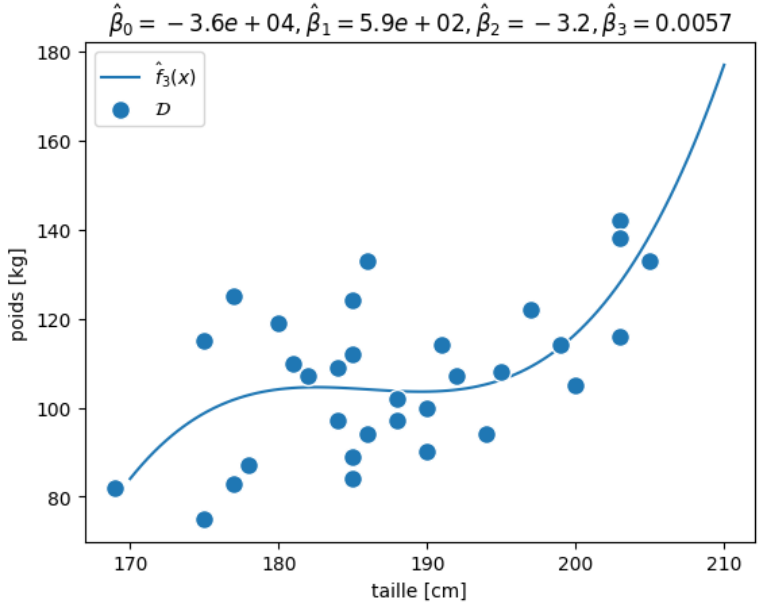
Que se passe-t-il avec notre hypothèse $\mathcal{H}_3$?

- Il est probable que nous n'ayons pas accès à toutes les variables explicatives dans nos données. Donc il est impossible d'obtenir un modèle qui explique correctement le problème.
- C'est pourtant l'hypothèse prise en travaillant avec $\mathcal{H}_3$. Le modèle a la **capacité** à expliquer entièrement le poids avec la taille, en lui donnant une importance démesurée par rapport à l'importance qu'elle a réellement.
- C'est ce que l'on appelle le **sur-apprentissage** : on accorde trop d'importance à certaines variables observées.
- On dit aussi que le modèle issu de $\mathcal{H}_3$ ne **généralise** pas correctement : la règle qui est **induite** lors de l'apprentissage ne correspond pas à la réalité.

Quel est le meilleur modèle ? ou la meilleure hypothèse ?



Espace des hypothèses $\mathcal{H}_1$



Espace des hypothèses $\mathcal{H}_3$

Quel est le meilleur modèle ? ou la meilleure hypothèse ?

Dans l'exemple précédent, en utilisant un jeu de données plus conséquent :

- Le polynôme de degré 3 semble désormais monotone mais reste peu plausible.
- La droite présente une pente $\hat{\beta}_1$ différente du modèle précédent.

Le sur-apprentissage n'est pas qu'une conséquence d'une complexité (ou capacité) de modèle trop importante, mais plutôt d'un rapport entre la complexité du modèle et la quantité de données observées pour l'apprentissage.

Le modèle entrainé $\hat{f}$ **varie** avec des données d'apprentissage $\mathcal{D}$ différentes.

Quel est le meilleur modèle ? ou la meilleure hypothèse ?

Que peut-on faire en tant que mathématiciens ?

- On doit donner un **contre-pouvoir** au risque empirique.
- Ce contre-pouvoir est une nouvelle métrique : le **risque vrai**.
- L'idée, très simple, consiste à **vérifier** le modèle avec des observations qui n'ont pas été utilisées pendant l'apprentissage et à **évaluer** cette métrique sur cet **échantillon de test**.

Risque vrai

Le risque **vrai** généralise la notion de risque empirique aux espaces $\mathcal{X}$ et $\mathcal{Y}$:

Définition : Risque vrai

Le risque vrai est une fonction $R : \mathcal{H} \to \mathbb{R}^+$ qui mesure la performance de toute hypothèse h sur les espaces $\mathcal{X}$ et $\mathcal{Y}$:

$$R = \mathbb{E}_{(\mathcal{X},\mathcal{Y})\sim\mathcal{P}}\left[L\left(y^{(i)}, h\left(x^{(i)}\right)\right)\right]$$

Au lieu de moyenner la fonction de perte sur les données explorées, le risque vrai moyenne la fonction de perte sur l'ensemble complet des observations possibles.
Dans la pratique, il est difficile voire impossible de calculer le risque vrai.
On peut obtenir une estimation grâce à un **échantillon de test**.

Écart de généralisation

Définition : Écart de généralisation

L'écart de généralisation est une fonction $\mathcal{H} \to \mathbb{R}$ qui mesure la différence entre le risque vrai et le risque empirique :

$$\text{Écart} = R - R_n$$

Un écart de généralisation positif signifie que l'algorithme est moins performant sur son espace d'application que sur l'espace sur lequel il a été entrainé. C'est un marqueur de sur-apprentissage, ou d'un défaut de généralisation de l'algorithme.

Risque vrai ponctuel

Définition : Risque vrai ponctuel

Le **risque vrai ponctuel** est une fonction $R_{xy} : \mathcal{H} \to \mathbb{R}^+$ qui mesure la performance de toute hypothèse h en **un point** des espaces $\mathcal{X}$ et $\mathcal{Y}$ tout en considérant **plusieurs** jeux de données d'apprentissage :

$$R_{xy} = \mathbb{E}_{\mathcal{D}} \left[L \left(y^{(i)}, h \left(x^{(i)} \right) \right) \right]$$

Des jeux de données différents peuvent conduire à des estimateurs h différents. Il est donc intéressant de voir, pour un espace des hypothèses $\mathcal{H}$ donné, comment se comportent **en moyenne** les estimateurs h en un point des espaces $\mathcal{X}$ et $\mathcal{Y}$.

Décomposition biais variance

Pour un problème de régression, utiliser l'erreur quadratique comme fonction de perte permet de décomposer le risque vrai ponctuel de la manière suivante :

$$R_{xy} = \left(\mathbb{E}_{\mathcal{D}}\left[f(x) - \hat{f}(x)\right]\right)^2 + \mathbb{E}_{\mathcal{D}}\left[\left(\hat{f}(x) - \mathbb{E}_{\mathcal{D}}\left[\hat{f}(x)\right]\right)^2\right] + \sigma_\epsilon^2$$

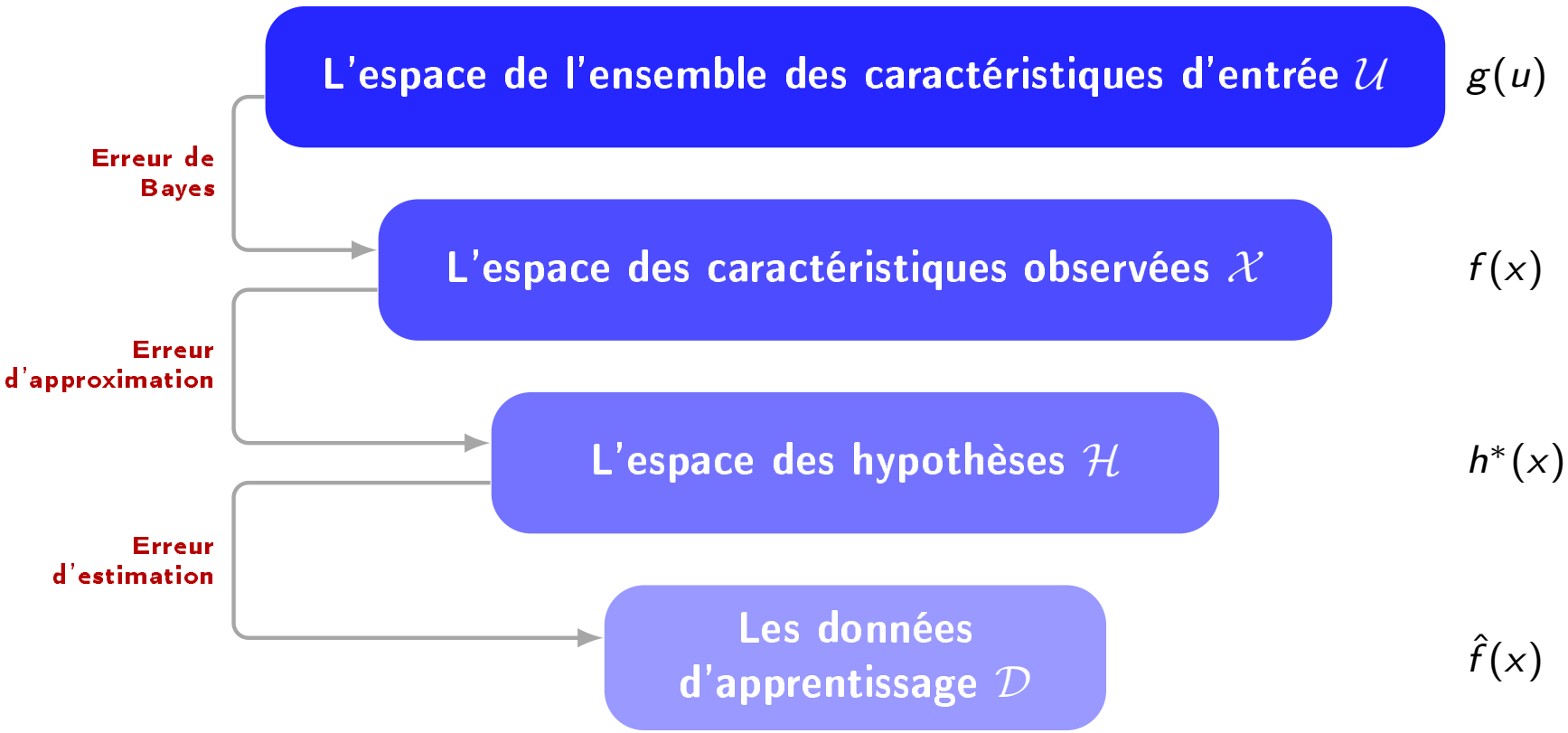
Cette décomposition est nommée **décomposition biais variance** de l'erreur.

Décomposition biais variance

- $\mathbb{E}_{\mathcal{D}}\left[f(x) - \hat{f}(x)\right]$ est appelé **biais** du modèle. C'est une erreur d'approximation.
- $\mathbb{E}_{\mathcal{D}}\left[\left(\hat{f}(x) - \mathbb{E}_{\mathcal{D}}\left[\hat{f}(x)\right]\right)^2\right]$ est appelé **variance** du modèle. C'est une erreur d'estimation.
- σ_ϵ^2 est la variance des incertitudes sur les données observées. C'est une erreur **irréductible**, appelée erreur de Bayes.

Le modèle a un risque ponctuel faible, à la condition qu'il dispose, à la fois d'un faible biais et d'une faible variance.
Le biais mesure la différence moyenne entre le modèle et la fonction vraie. La variance mesure comment la prédiction du modèle change quand les données d'apprentissage changent.

Décomposition de l'erreur



Le rasoir d'Ockham

L'espace des hypothèses $\mathcal{H}$ doit représenter des fonctions suffisamment complexes (faible biais) pour représenter les relations entre les variables d'entrées et de sortie. Mais pas trop complexe pour ne pas laisser la possibilité à l'algorithme d'accorder une importance trop grande à certaines variables d'entrée (faible variance).

Le risque empirique doit être minimisé mais en maintenant un écart de généralisation faible.

Si deux espaces d'hypothèses permettent d'arriver à des modèles de performance semblable, privilégier l'hypothèse la plus simple.

Le rasoir d'Ockham

Les deux chemins mènent au même endroit. Le chemin bleu est plus rapide. Mais le chemin rouge est beaucoup plus facile à expliquer à un touriste. Et le touriste aura moins de chance de se perdre.

Métriques de performance

Dans cette section, nous introduirons les métriques de performance essentielles pour les problèmes de régression et de classification. Il en existe bien plus que ce qui est mentionné ici !

Métriques de régression : le résidu

La première métrique, centrale au problème de régression, est le **résidu**. Littéralement, cette métrique mesure ce qui n'a pas pu être expliqué par le modèle, ce qui **reste** à expliquer :

Définition : Résidu

Dans un problème de régression, le **résidu** pour une obsservation i est la différence entre les caractéristiques de sorties observées et celles estimées par le modèle $\hat{f} : \mathcal{X} \to \mathcal{Y}$:

$$r^{(i)} = y^{(i)} - \hat{f}(x^{(i)})$$

Le résidu r ne doit pas être confondu, ni avec l'incertitude $\epsilon^{(i)} = y^{(i)} - f(x^{(i)})$, ni avec l'erreur de modèle $f(x^{(i)}) - \hat{f}(x^{(i)})$ qui combine à la fois l'erreur d'approximation et l'erreur d'estimation. Comme l'incertitude $\epsilon^{(i)}$ ne peut en général pas être mesurée, $r^{(i)}$ en est la meilleure estimation possible.

Métriques de régression : le coefficient de détermination

Le coefficient de détermination est défini en fonction de la somme des carrés des résidus, notée $\mathrm{SS}_{\mathrm{res}}$ et de la somme des carrés totale, notée $\mathrm{SS}_{\mathrm{tot}}$

Définition : Coefficient de détermination

Dans un problème de régression, le **coefficient de détermination** R^2 est une métrique mesurant la fraction de variance expliquée :

$$R^2 = 1 - \frac{\mathrm{SS}_{\mathrm{res}}}{\mathrm{SS}_{\mathrm{tot}}} = 1 - \frac{\sum_{i=1}^{n} \left(y^{(i)} - \hat{y}^{(i)}\right)^2}{\sum_{i=1}^{n} \left(y^{(i)} - \bar{y}\right)^2}$$

Note : Dans le contexte de la régression linéaire, quelle que soit la fonction d'augmentation Φ disposant d'une constante β_0, le coefficient de détermination est aussi le carré du coefficient de corrélation de Pearson entre les caractéristiques de sortie estimées $\hat{y}$ et les caractéristiques de sorties observées y.

Métriques de régression : le coefficient de détermination

- Si le modèle prédit $\bar{y}$ pour toute entrée x, alors $R^2 = 0$.
- Si le modèle prédit $y^{(i)}$ pour l'entrée $x^{(i)}$ pour toute observation i, alors $R^2 = 1$.
- Si les résidus $r^{(i)}$ sont nuls pour toute observation i, alors $R^2 = 1$.
- Attention ! Un coefficient de détermination proche de 1 peut masquer un problème de sur-apprentissage. On évalue donc cette métrique à la fois sur les données d'apprentissage et sur des données de test.

Métrique de classification : la matrice de confusion

Pour un problème de classification à m classes ($\mathcal{Y} = \{0, 1, \cdots, m-1\}$), la matrice de confusion permet de **compter** les prédictions réalisées par le modèle par rapport à la classe réelle pour chaque observation k :

Définition : Matrice de confusion

En classification à m classes, la matrice de confusion $C \in \mathcal{M}_{m,m}$ est définie comme suit en fonction des classes **réelles** $y^{(k)}$ et des classes **prédites** $\hat{y}^{(k)}$ pour chaque observation k :

$$C_{i,j} = \text{card}\left(\{k \in \{1, 2, \cdots, n\}\} \mid y^{(k)} = i \text{ et } \hat{y}^{(k)} = j\right)$$

Note : Attention ! Il n'existe pas de convention globale pour la disposition de la matrice. Dans la définition ci-dessus, les classes réelles sont représentées en ligne et les classes prédites en colonnes (convention de `sklearn.metrics.confusion_matrix`). La matrice porte bien son nom !

Métrique de classification : la matrice de confusion

C'est beaucoup plus simple qu'il n'y paraît. Illustrons avec un problème de classification binaire ($m = 2$) où nous cherchons à prédire la position (avant = 1 ou trois-quart = 0) d'un joueur de rugby :

$$C = \begin{array}{c} \begin{matrix} \text{prédiction : trois-quart} & \text{prédiction : avant} \end{matrix} \\ \begin{bmatrix} 5 & 2 \\ 1 & 7 \end{bmatrix} \end{array} \begin{array}{l} \text{réelle : trois-quart} \\ \text{réelle : avant} \end{array}$$

- Il y a $5 + 7$ prédictions correctes et $1 + 2$ prédictions incorrectes.
- Une matrice de confusion diagonale ne comporte que des prédictions correctes.
- Les prédictions erronées se retrouvent dans les termes extra-diagonaux.
- Comme avant = 1, 7 est le nombre de **vrai positifs**(VP).
- 5 est le nombre de **vrai négatifs**(VN), 2 le nombre de **faux positifs**(FP) et 1 le nombre de **faux négatifs**(FN).

Métrique de classification : la matrice de confusion

Il y a d'autres dénominations associées à la matrice de confusion qui peuvent être rencontrées :

$$\text{Précision} = \frac{\text{VP}}{\text{VP} + \text{FP}}$$

$$\text{Rappel} = \frac{\text{VP}}{\text{VP} + \text{FN}}$$

$$\text{Valeur prédictive négative} = \frac{\text{VN}}{\text{VN} + \text{FN}}$$

$$\text{Spécificité} = \frac{\text{VN}}{\text{VN} + \text{FP}}$$

La précision (la valeur prédictive négative) répond à la question spécifique suivante : est-ce que le joueur que j'ai classé comme avant (trois-quart) en était bien un ?
Le rappel (la spécificité) répond à l'autre question spécifique suivante : ai-je bien réussi à classer tous les avants (trois-quart) de mes données en tant que tel ?

Méthodes de validation croisée

Nous avons vu précédemment que :

- Pour vérifier la capacité du modèle à **généraliser** il est nécessaire d'évaluer sa performance sur des données différentes de celles utilisées pour l'apprentissage.
- L'estimation $\hat{f}$ dépend des données d'apprentissage. Des données différentes peuvent conduire à un modèle différent. C'est la **variance** du modèle.

En apprentissage automatique il est commun d'utiliser plusieurs échantillons de données :

- Données d'**apprentissage** : pour l'entraînement du modèle.
- Données de **validation** : pour tester différents espaces des hypothèses $\mathcal{H}$.
- Données de **test** : pour l'évaluation finale et *indépendante* du modèle.

Méthodes de validation croisée

Dans la pratique, une équipe développant un modèle d'apprentissage automatique, ne *devrait* pas avoir accès aux données de test. Un échantillon de **validation** *fixe* présente des avantages dès qu'il s'agit de comparer des estimateurs différents sur les mêmes données. Mais il est aussi courant de **générer** les données de validations à partir des données d'apprentissage.

Nous explorerons dans cette section différentes méthodes des **validation croisée** qui permettent de générer ces données de validation :

- Leave-One-Out
- Validation croisée à K plis
- Bootstrapping

Leave-One-Out

Pour des données d'apprentissage $\mathcal{D} = \{(x^{(i)}, y^{(i)}) \mid i = 1, 2, \cdots, n\}$, la méthode de **Leave-One-Out** consiste à créer n échantillons de taille $n - 1$ tel que :

$$\mathcal{D}^{(-k)} = \{(x^{(i)}, y^{(i)}) \quad \forall i \neq k\} \quad \forall k = 1, 2, \cdots, n$$

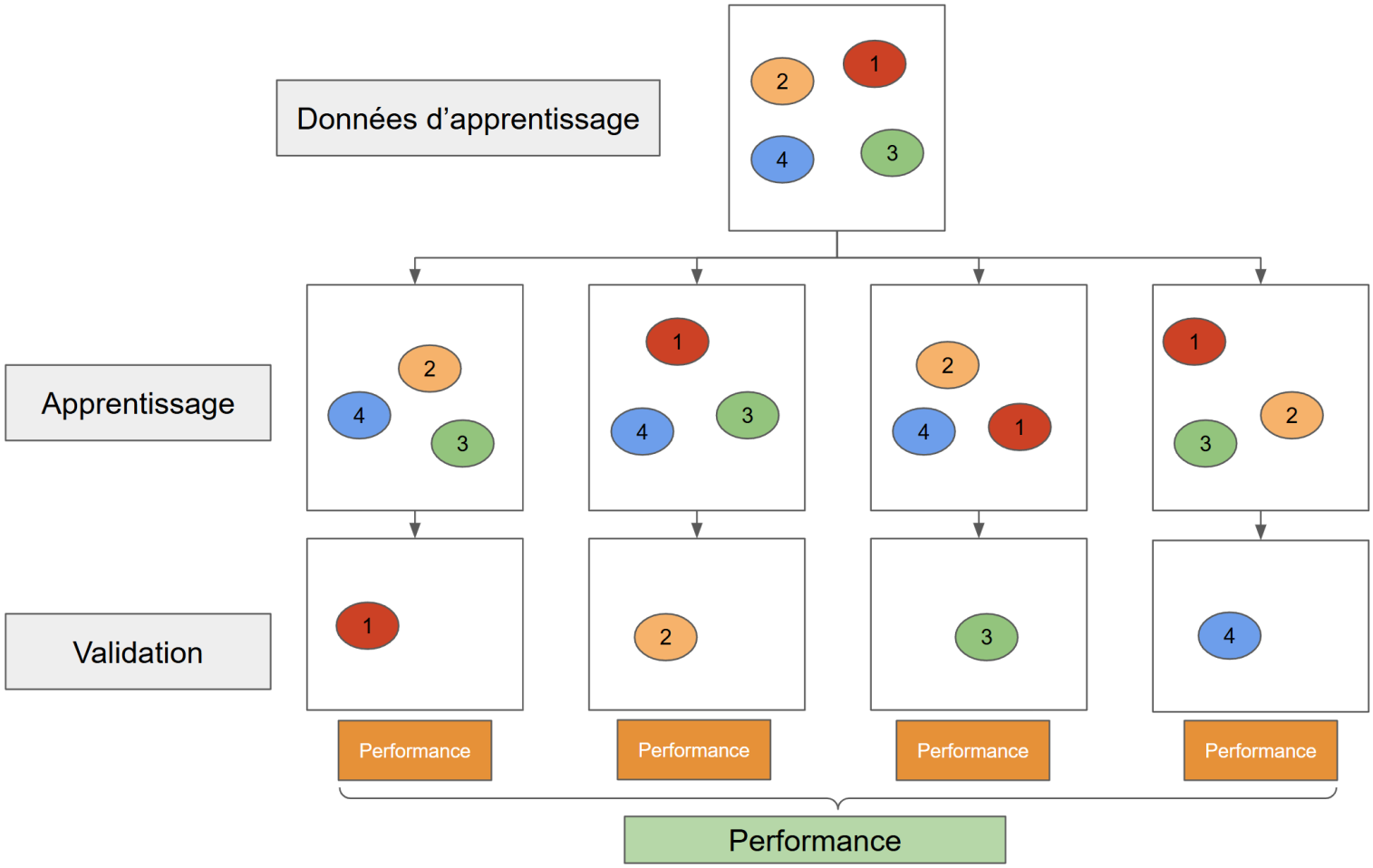
L'estimateur $\hat{f}^{(-k)}$ est donc entraîné à partir des données $\mathcal{D}^{(-k)}$ et une métrique d'erreur peut être calculé sur le point $(x^{(k)}, y^{(k)})$. On calcule ensuite des métriques moyennes sur les n différents modèles entraînés.

Cette méthode est principalement utilisée pour des données d'apprentissage avec très peu d'observations ($n \sim 10 - 100$).

Voir : `sklearn.model_selection.LeaveOneOut`.

Leave-One-Out

Un exemple de Leave-One-Out :



Validation croisée à K plis (K-Fold)

La méthode de **validation croisée à K plis** consiste à partitionner aléatoirement les données $\mathcal{D}$ en K sous-ensembles (appelés **plis** ou **blocs**) de tailles égales.

Soit $\mathcal{I} = \{1, \ldots, n\}$ l'ensemble des indices de $\mathcal{D}$. On définit une **partition** de $\mathcal{I}$ en K sous-ensembles disjoints $\mathcal{I}_1, \ldots, \mathcal{I}_K$ tels que :

$$\bigcup_{k=1}^{K} \mathcal{I}_k = \mathcal{I} \quad \text{et} \quad \mathcal{I}_j \cap \mathcal{I}_m = \emptyset \text{ pour } j \neq m$$

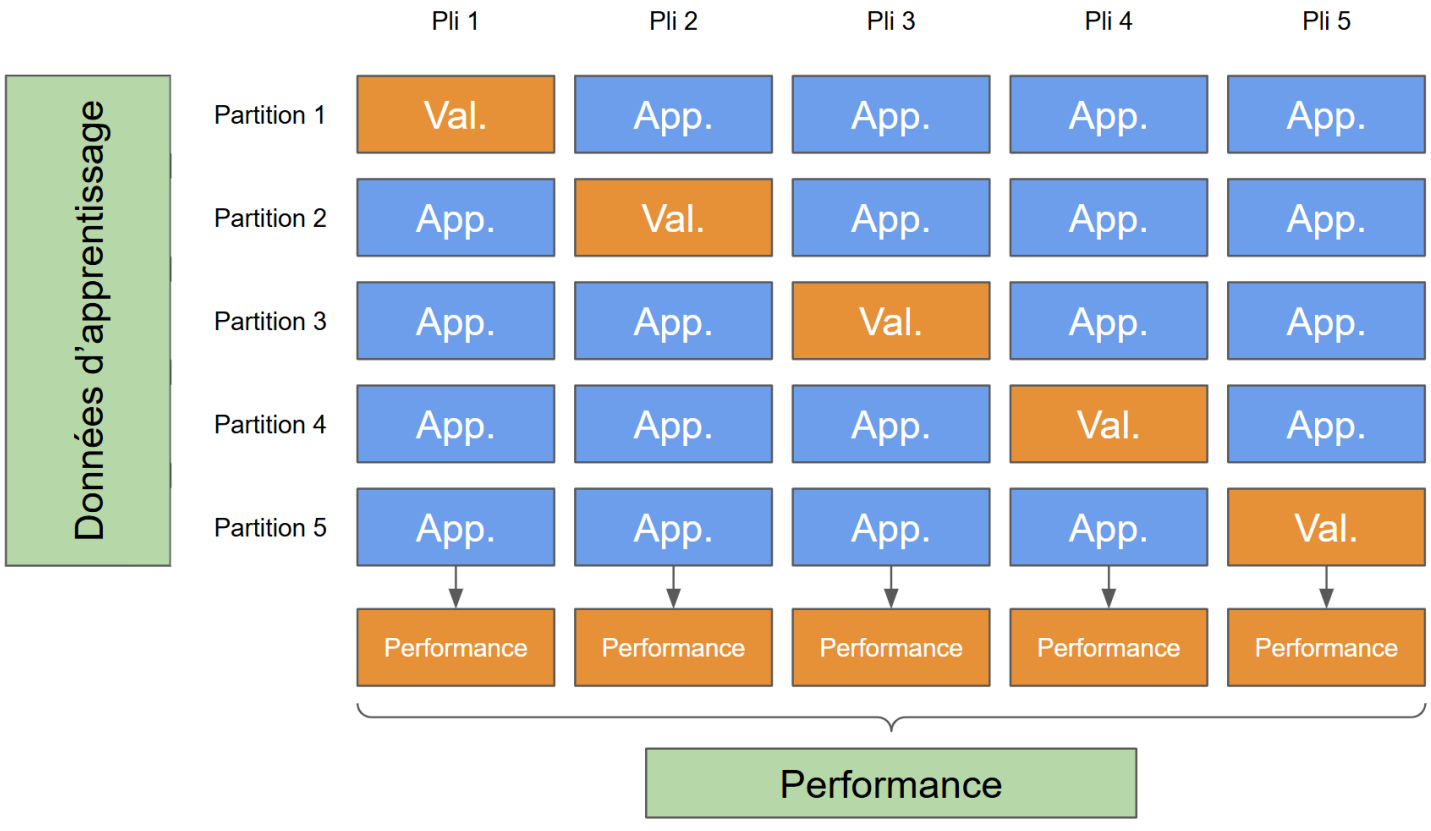
Pour chaque pli k, on définit l'estimateur $\hat{f}^{(-k)}$ entraîné sur $\mathcal{I} \setminus \mathcal{I}_k$. Les métriques de performance étant alors évaluées sur $\mathcal{I}_k$. On moyenne ensuite ces métriques pour l'ensemble des K plis.

Typiquement, on choisit $K = 5$ ou $K = 10$. Pour $K = n$, on retrouve la méthode Leave-One-Out.

Voir : `sklearn.model_selection.KFold`.

Validation croisée à K plis (K-Fold)

Un exemple de validation croisée à 5 plis :



Bootstrapping

La méthode du **bootstrapping** consiste à ré-échantillonner, avec remise, n observations aléatoires de l'ensemble des données $\mathcal{D}$. On notera $\mathcal{D}^{(b)}$ un échantillon ainsi créé. On génère B échantillons de la sorte.

Soit $\mathcal{I} = \{1, \ldots, n\}$ l'ensemble des indices de $\mathcal{D}$. Un échantillon bootstrap $\mathcal{D}^{(b)}$ est une suite de n variables aléatoires $I_1^*, I_2^*, \cdots, I_n^*$ indépendantes et identiquement distribuées. Chaque variable aléatoire suit une loi uniforme discrète sur $\mathcal{I}$:

$$P(I_j^* = k) = \frac{1}{n}, \quad \forall k \in \mathcal{I}, \quad \forall j = 1, 2, \cdots, n$$

Bootstrapping

Un échantillon bootstrap est donc défini comme :

$$\mathcal{D}^{(b)} = \left\{ \left(x^{(l_j^*)}, y^{(l_j^*)} \right) \right\}_{j=1}^{n}$$

On peut alors en déduire un échantillon pour la validation (out-of-bag) :

$$\mathcal{D}_{\text{oob}}^{(b)} = \mathcal{D} \setminus \mathcal{D}^{(b)}$$

Typiquement, on choisit $B \sim 10 - 1000$ et on moyenne les métriques sur les B échantillons de validation.

Bootstrapping

Un exemple de bootstrapping :