Techniques d'apprentissage

William KENGNE
Institut Camille Jordan, Université Jean Monnet

Groupe de travail sur l'apprentissage machine

Institut Camille Jordan

12 octobre, 2023

Outline

1 Des exemples pratiques· · · à la théorie

2 Fondement théorique de l'apprentissage supervisé

3 Quelques méthodes/algorithmes de l'apprentissage supervisé

Outline

1 Des exemples pratiques· · · à la théorie

2 Fondement théorique de l'apprentissage supervisé

3 Quelques méthodes/algorithmes de l'apprentissage supervisé

Naufrage du Titanic

On s'intéresse au naufrage du paquebot Titanic en 1912 avec 2201 individus à bord.

Parmi eux, 711 ont survécu au naufrage et 1490 se sont noyés.

Variables :

- Survived : survie du passager : 1 ("survie"), 0 ("non survécu");
- Pclass : 1 (meilleure classe), 2 (intermédiaire), 3 ("éco");
- Sex : sexe du passager;
- Age : âge du passager;
- Embarked : port d'embarquement : "C" (Cherbourg), "Q" (Queenstown) et "S" (Southampton);
- ...

Naufrage du Titanic

Données ayant fait l'objet de plusieurs machine learning challenge

Disponible sur Kaggle

Objectif : Prédire le profil de survie des passagers

Contrôle de qualité : chaîne de production agroalimentaire

On dispose de biscuits non cuits pour lesquels on souhaite connaître rapidement la composition en quatre ingrédients : lipides, sucres, farine et eau

Afin d'intervenir au plutôt sur les équipements si un dysfonctionnement est détecté

Les analyses chimiques sont relativement longues et coûteuses ; et ne peuvent être mise en ligne sur une chaîne de production.

On utilise alors un spectromètre qui mesure l'absorbance dans le domaine proche infrarouge, afin d'expliquer et prédire la composition par spectre.

Contrôle de qualité : chaîne de production agroalimentaire

Les données (cookies) : 72 biscuits

Variables :

- Taux de sucre, lipides, farine, eau
- 700 variables représentant le spectre sur toutes les longueurs d'ondes entre 1100 et 2498 nanomètres, régulièrement espacés de 2 nanomètres.

Objectif : Prédire le taux de sucre, lipides, farine, eau à partir de l'observe ensuite le spectre

Reconnaissance de caractères

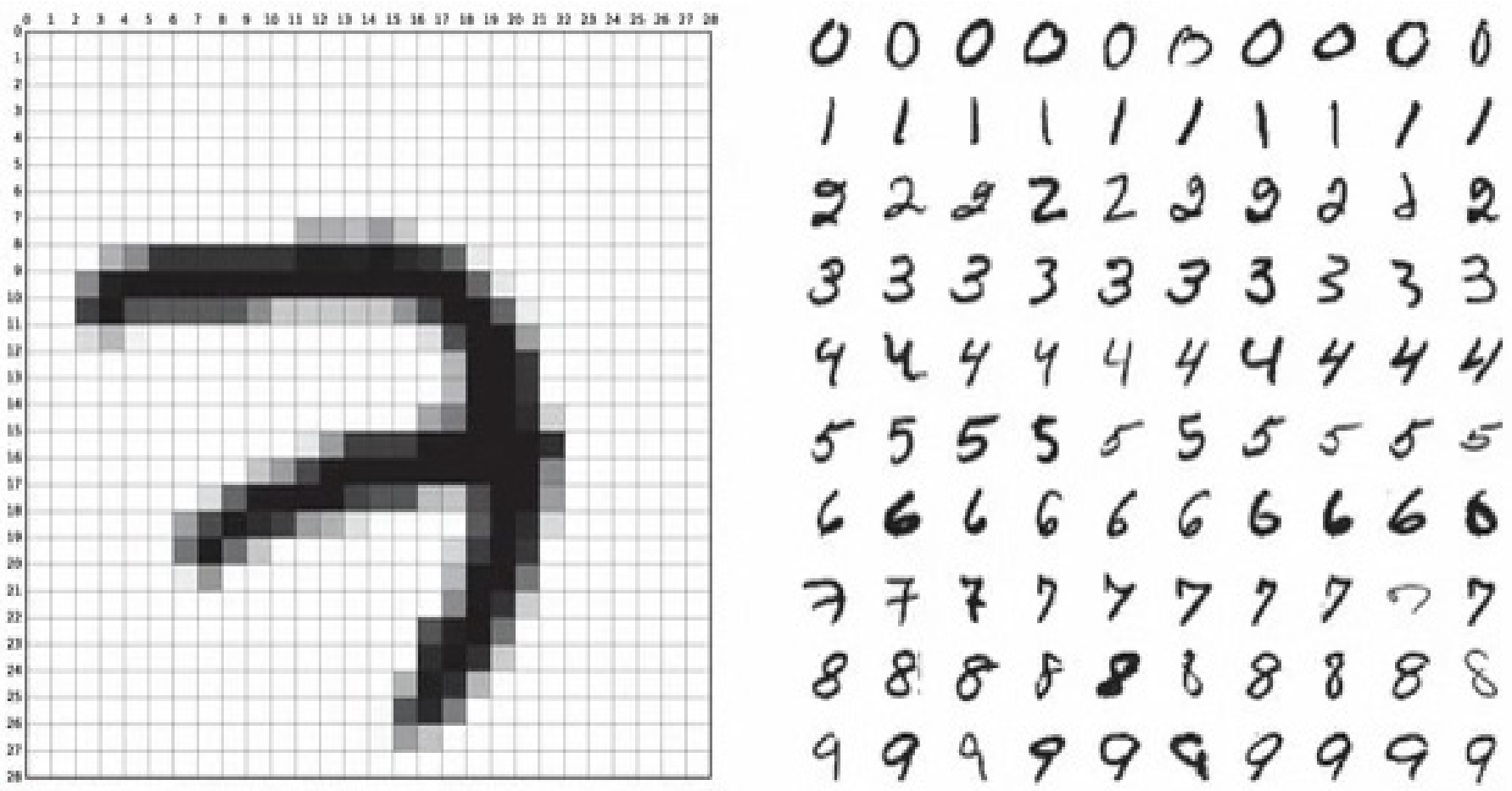


Figure: Labels données MNIST, source Baldominos *et al.* (2019)

Reconnaissance de caractères

On dispose des chiffres manuscrits décrits au format 28×28, soit 784 pixels, ainsi que les labels associées.

Données MNIST: Cette base contient 70 000 exemples, répartie en 60 000 exemples pour l'apprentissage et 10 000 pour le test.

Variables :

- Labels : 0, 1, 2,. . .,9
- 784 variables représentant les niveaux de gris des points de l'image

Objectif : Prédire les labels à partir de la descrition au format 28×28

Formalisme: Apprentissage supervisé

On suppose que $Z = (X, Y)$ où X est le vecteurs des variables explicatives et Y la variable à expliquer.

Les observations sont alors $(X_1, Y_1), \cdots, (X_n, Y_n)$.

L'espace des observations est noté $\mathcal{X} \times \mathcal{Y}$ où $\mathcal{X}$ est l'espace des variables explicatives et $\mathcal{Y}$ celui de la variable à expliquer.

Exemple: Cas du Titanic

- $Y =$ "survécu" ou "noyés" i.e. $\mathcal{Y} = \{$"survécu", "noyés"$\}$
- $X =$ ("Pclass", "Sex","Age","Embarked", $\cdots$)

Formalisme: Apprentissage supervisé

Objectif: Construire une fonction $f : \mathcal{X} \to \mathcal{Y}$ susceptible de reproduire "au mieux" la variable Y ayant observé X

i.e. on postule à un modèle de la forme

$$Y = f(X) + \epsilon$$

où f est une fonction inconnu et ϵ le bruit ou terme d'erreur.

L'échantillon $(X_1, Y_1), \cdots, (X_n, Y_n)$ qui servira à construire f est appelé **échantillon d'apprentissage**.

Lorsque Y est qualitative ou $\mathcal{Y}$ fini, on parle de **classification** ou **discrimination**

Lorsque Y est quantitative (continue), on parle de **régression**.

Formalisme: Apprentissage non supervisé

Apprentissage non supervisé (ou clustering): Ensemble de méthodes/algorithmes utilisées lorsque l'on dispose de données au sein desquelles on cherche à dégager une structure sous-jacente

Exemple: rechercher des classes homogènes (susceptibles de traitements différenciés).

Principe: Regrouper les objets ayant des caractéristiques similaires (homogénéité interne) et séparent les objets ayant des caractéristiques différentes (hétérogénéité externe).

Ici, $Z = X$ et les observations sont $X_1, \cdots, X_n$. On ne dispose pas de variable à expliquer Y, et ne s'intéresse donc pas à la prédiction.

Algorithmes: Centres mobiles, k-means, CAH,$\cdots$

Outline

1 Des exemples pratiques· · · à la théorie

2 Fondement théorique de l'apprentissage supervisé

3 Quelques méthodes/algorithmes de l'apprentissage supervisé

Notion de risque

- Observations $(X_1, Y_1), \cdots, (X_n, Y_n)$ issu de (X, Y) où X est le vecteur des variables explicatives et Y la variable à expliquer.
- L'espace des observations est noté $\mathcal{X} \times \mathcal{Y}$ où $\mathcal{X}$ est l'espace des variables explicatives et $\mathcal{Y}$ celui de la variable à expliquer.
- $D_n = \{(X_1, Y_1), \cdots, (X_n, Y_n)\}$ est appelé échantillon/ensemble d'apprentissage.

Une fonction/règle de prédiction (ou prédicteur), est une fonction (mesurable) $f : \mathcal{X} \to \mathcal{Y}$. Elle associe à une entrée X, la sortie $f(X)$.

L'ensemble de toutes les fonctions de prédiction est noté $\mathcal{F}(\mathcal{X}, \mathcal{Y})$.

Notion de risque

Algorithme d'apprentissage : processus qui à tout échantillon d'apprentissage, renvoie une fonction de prédiction.

Soit $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ une fonction de perte

$\ell(y, y')$ représente la perte encourue entre une sortie réelle y et une sortie prédite y'.

Exemple:

- Régression réelle : $\mathcal{Y} = \mathbb{R}$. Pertes L^p: $\ell(y, y') = |y - y'|^p$.

  $p = 1 \Rightarrow$ perte absolue

  $p = 2 \Rightarrow$ perte quadratique
- Classification : $\mathcal{Y}$ fini. $\ell(y, y') = \mathbb{1}_{y \neq y'}$: perte 0-1.
- Cas particulier de la classification binaire : $\mathcal{Y} = \{-1, 1\}$. $\ell(y, y') = \max(0, 1 - yy')$.

Notion de risque

La qualité d'un prédicteur $f \in \mathcal{F}(\mathcal{X}, \mathcal{Y})$ est mesurée par son **risque** (ou erreur de généralisation), défini par:

$$R(f) = \mathbb{E}\big[\ell(Y, f(X))\big] = \mathbb{E}_Z\big[\ell(Y, f(X))\big].$$

On dira qu'une fonction de prédiction $f^* \in \mathcal{F}(\mathcal{X}, \mathcal{Y})$ définit une **règle optimale** si

$$R(f^*) = \inf_{f \in \mathcal{F}(\mathcal{X}, \mathcal{Y})} R(f).$$

On dit aussi que f^* est une **fonction cible**.

Objectif d'un algorithme d'apprentissage: construire (à partir des données) un prédicteur $\widehat{f_n}$ de sorte que $R(\widehat{f_n})$ soit proche de $R(f^*)$.

Notion de risque

Exemple:

1. **Régression aux moindres carrés**.

$$f^* : \mathcal{X} \to \mathbb{R},\ x \mapsto \mathbb{E}(Y|X = x)$$

est une fonction cible.

2. **Classification avec perte 0-1**.

$$f^* : \mathcal{X} \to \mathcal{Y},\ x \mapsto f^*(x) = \underset{y \in \mathcal{Y}}{\text{argmax}} P(Y = y|X = x)$$

est une fonction cible.

Pour un prédicteur f, $R(f)$ ne peut être calculé en pratique, car il dépend de la loi de (X, Y) qui est inconnue.

$R(f)$ peut être estimé par sa version empirique, appelé risque empirique.

Notion de risque

Le **risque empirique** (associé à D_n) d'une fonction de prédiction f, est défini par

$$\widehat{R}_n(f) = \frac{1}{n}\sum_{i=1}^{n} \ell(Y_i, f(X_i)).$$

Par la loi des grands nombres: $\widehat{R}_n(f) \xrightarrow[n\to\infty]{\mathcal{P}} \mathbb{E}\big[\ell(Y, f(X))\big] = R(f)$

Problème:

- Ce résultat est asymptotique, on souhaiterait par exemple avoir erreur fonction de la taille des observations n ;
- La prédicteur issue de l'algorithme d'apprentissage est $\widehat{f}_n$; on aimerait plutôt comparer : $\widehat{R}_n(\widehat{f}_n)$ et $R(\widehat{f}_n)$; $R(\widehat{f}_n)$ et $R(f^*)$

Principe de minimisation du risque empirique

On souhaite construire un prédicteur ayant le plus petit risque

Comme le risque est approché par le risque empirique $\widehat{R}_n$: il est naturel de considérer un l'algorithme qui minimise le risque empirique.

Définition: Soit $\mathcal{F} \subset \mathcal{F}(\mathcal{X}, \mathcal{Y})$ (un modèle), l'algorithme de minimisation du risque empirique sur $\mathcal{F}$ est défini par

$$\widehat{f}_{\mathcal{F}} := \underset{f \in \mathcal{F}}{argmin}\ \widehat{R}_n(f).$$

Question: Comment choisir $\mathcal{F}$?

Principe de minimisation du risque empirique

Prendre $\mathcal{F} = \mathcal{F}(\mathcal{X}, \mathcal{Y})$ n'est pas une bonne idée :

Cela conduirait à l'algorithme qui attribue $Y_i = y_i$ à une entrée $X_i = x_i$ (avec $(X_i, Y_i) \in D_n$) et attribue une valeur quelconque à une entrée différente de $X_i = x_i$

C'est le phénomène de **sur-apprentissage** (ou overfitting)

En règle générale : Prendre $\mathcal{F}$ suffisamment grand pour pouvoir raisonnablement approcher la fonction cible tout en ne le prenant pas trop grand pour éviter le phénomène de sur-apprentissage.

La "grandeur" de $\mathcal{F}$ est appelée complexité.

Minimisation du risque empirique

L'objectif est de déterminer un modèle $\mathcal{F}$ pour lequel $R(\widehat{f}_{\mathcal{F}})$ est proche de $R(f^*)$. On a:

$$\underbrace{R(\widehat{f}_{\mathcal{F}}) - R(f^*)}_{\text{excès de risque}} = \underbrace{\inf_{f\in\mathcal{F}} R(f) - R(f^*)}_{\text{erreur d'approximation}} + \underbrace{R(\widehat{f}_{\mathcal{F}}) - \inf_{f\in\mathcal{F}} R(f)}_{\text{erreur d'estimation}}.$$

Dans cette décomposition :

- l'**erreur d'approximation** mesure à quel point le modèle $\mathcal{F}$ permet d'approcher f^* ; elle est due au fait que $\mathcal{F}$ ne contient pas nécessairement pas la cible f^* ;
- l'**erreur d'estimation** provient de la minimisation sur $\mathcal{F}$ et mesure la qualité de l'estimation de $f_{\mathcal{F}}$ par $\widehat{f}_{\mathcal{F}}$.

Décomposition erreur d'approximation / erreur d'estimation

Plus le modèle $\mathcal{F}$ est "grand" (ou complexe), plus l'erreur d'approximation est faible,
en contrepartie, l'erreur d'estimation augmente (phénomène de sur-apprentissage).

Le phénomène inverse (sous-apprentissage) se produit lorsque $\mathcal{F}$ est "assez petit".

Nécessite à trouver un compromis (souvent appelé dilemme biais-variance) dans le choix de $\mathcal{F}$

Minimisation du risque empirique

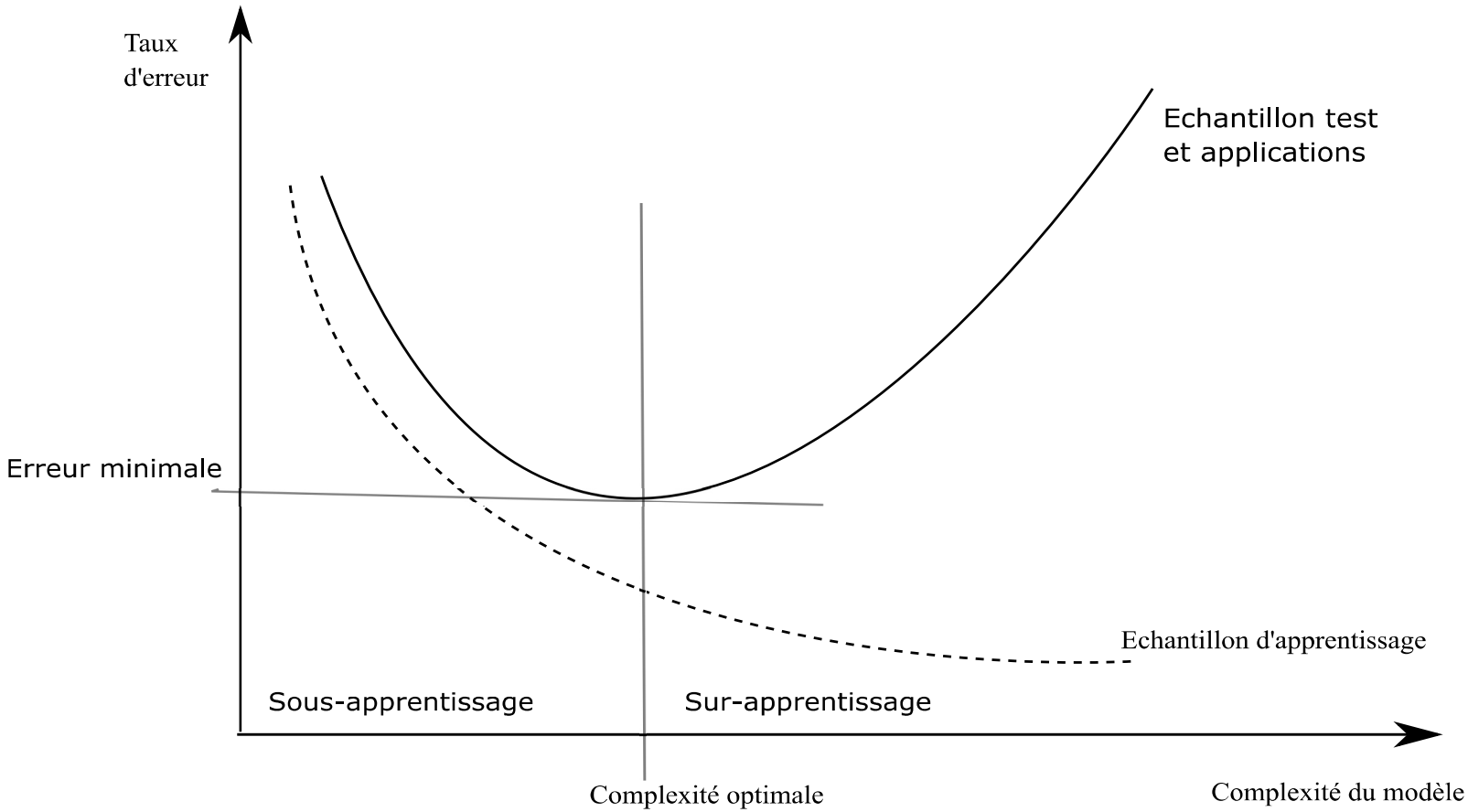


Figure: Taux d'erreur en fonction de la complexité du modèle

Minimisation du risque empirique

L'erreur d'approximation est généralement très difficile à estimer: elle dépend de f^* est inconnue.

Démarche générale de la théorie d'apprentissage :

- accepter une erreur d'approximation
- réduire considérablement l'erreur d'estimation

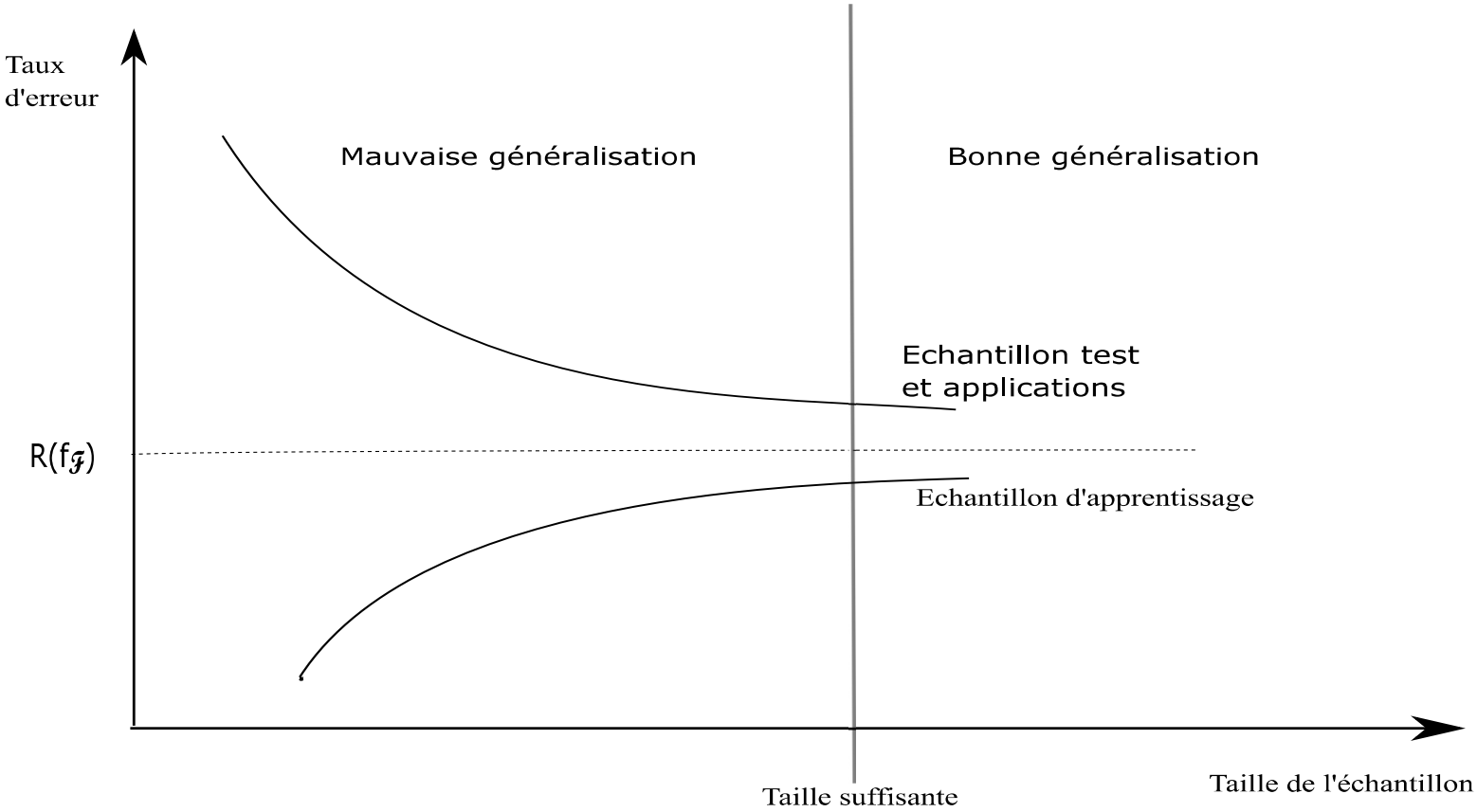
Et de répondre aux questions:

1. Sous quelles conditions le principe de minimisation du risque empirique est consistent? i.e. sous quelles conditions a t-on

$$R(\widehat{f}_{\mathcal{F}}) \xrightarrow[n\to\infty]{\mathcal{P}} R(f_{\mathcal{F}}) \quad \text{et} \quad \widehat{R}_n(\widehat{f}_{\mathcal{F}}) \xrightarrow[n\to\infty]{\mathcal{P}} R(f_{\mathcal{F}})?$$

2. A quelles vitesses converge-t-on? Cela est un indicateur de la capacité de généralisation du prédicteur

Minimisation du risque empirique



Minimisation du risque empirique

Exemple de mesure de complexité de $\mathcal{F}$:

- **Dimension VC**: Cardinal du plus grand ensemble de points qu'il est possible de pulvériser par $\mathcal{F}$ (i.e. réaliser toutes les classifications)
- **Nombre de recouvrement**: Nombre minimal de boules nécessaires pour recouvrir $\mathcal{F}$

Résultats: On montre (sous d'autres hypothèses) que si la "taille" ou "complexité" est finie, alors :

- le principe de minimisation du risque empirique est consistent
- l'erreur de généralisation est $\mathcal{O}(n^{-1/2})$.

Outline

1 Des exemples pratiques· · · à la théorie

2 Fondement théorique de l'apprentissage supervisé

3 Quelques méthodes/algorithmes de l'apprentissage supervisé

Régression ridge et LASSO

On se place dans le cadre de la régression avec par exemple $\mathcal{Y} = \mathbb{R}$.

On considère les prédicteurs obtenus par minimisation du risque empirique régularisé (pénalisé):

$$\widehat{R}_n(f) + \lambda ||f||_q^q$$

avec $\lambda > 0$ et $q \geq 1$.

- $q = 2 \Rightarrow$ régression ridge
- $q = 1 \Rightarrow$ régression LASSO

Exemple: Applicable au problème de contrôle de qualité dans les biscuits

Plus proches voisins

On se place dans le cadre de la classification avec $\mathcal{Y}$ fini.

Observation : $(y_1, x_1), \cdots, (y_n, x_n)$ une réalisation de $(Y, \underline{X})$

Objectif : Prédire la classe d'un nouvel individu i_0 sur lequel on a observé x_{i_0}

k-plus proches voisins (avec $k \in \mathbb{N}^*$) $\equiv$ k-NN (k-Nearest Neighbor)

Principe : affecter i_0 dans la classe majoritaire de ses k voisins les plus proches.

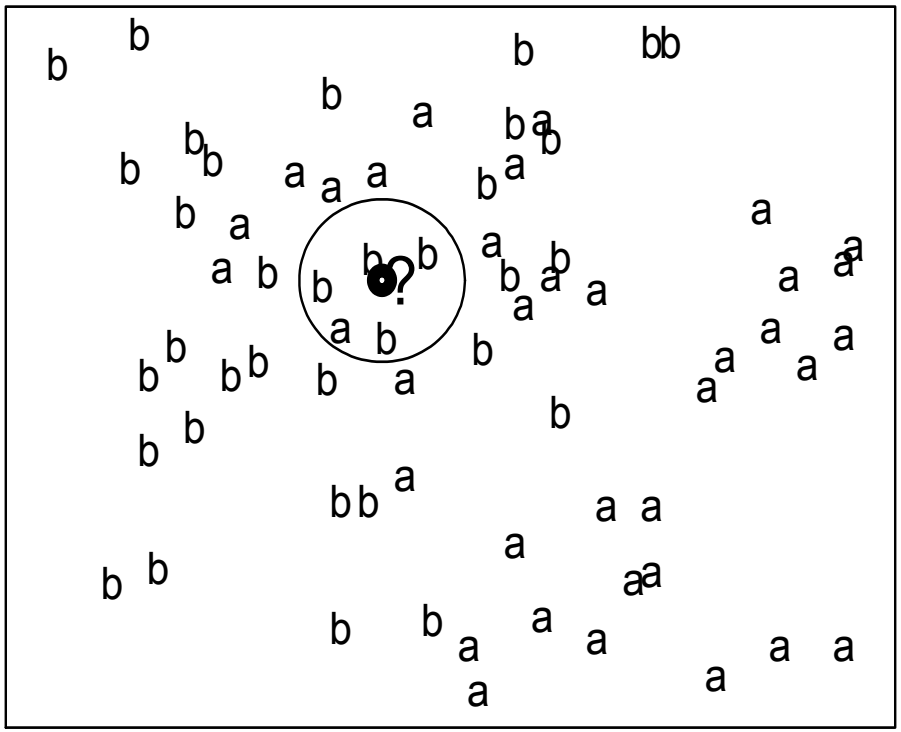
La notion de voisin ici est relative à une distance d définie sur $\mathcal{X}$ (par exemple la distance Euclidienne lorsque $\mathcal{X} \subset \mathbb{R}^p$).

Plus proches voisins

Algorithme de la procédure :

- Calculer les distances $d(x_{i_0}, x_i)$, $1 \leq i \leq n$.
- Ranger ces distances dans l'ordre $d(x_{i_0}, x_{i_1}) < d(x_{i_0}, x_{i_2}) < \cdots d(x_{i_0}, x_{i_n})$
- Retenir les individus $i_1, i_2, \cdots, i_k$ (qui sont les k plus proches voisins de i_0).
- Affecter l'individu i_0 à la classe majoritaire de ces k voisins.

Plus proches voisins



Plus proches voisins: Choix de k et validation croisée

On peut

- Choisir k en minimisant l'erreur de classement calculée par validation croisée
- Prendre $k = K + 1$ si K n'est pas "très grand" ($K < 10$, K est le nombre de classes)

Estimation de l'erreur de classement par validation croisée

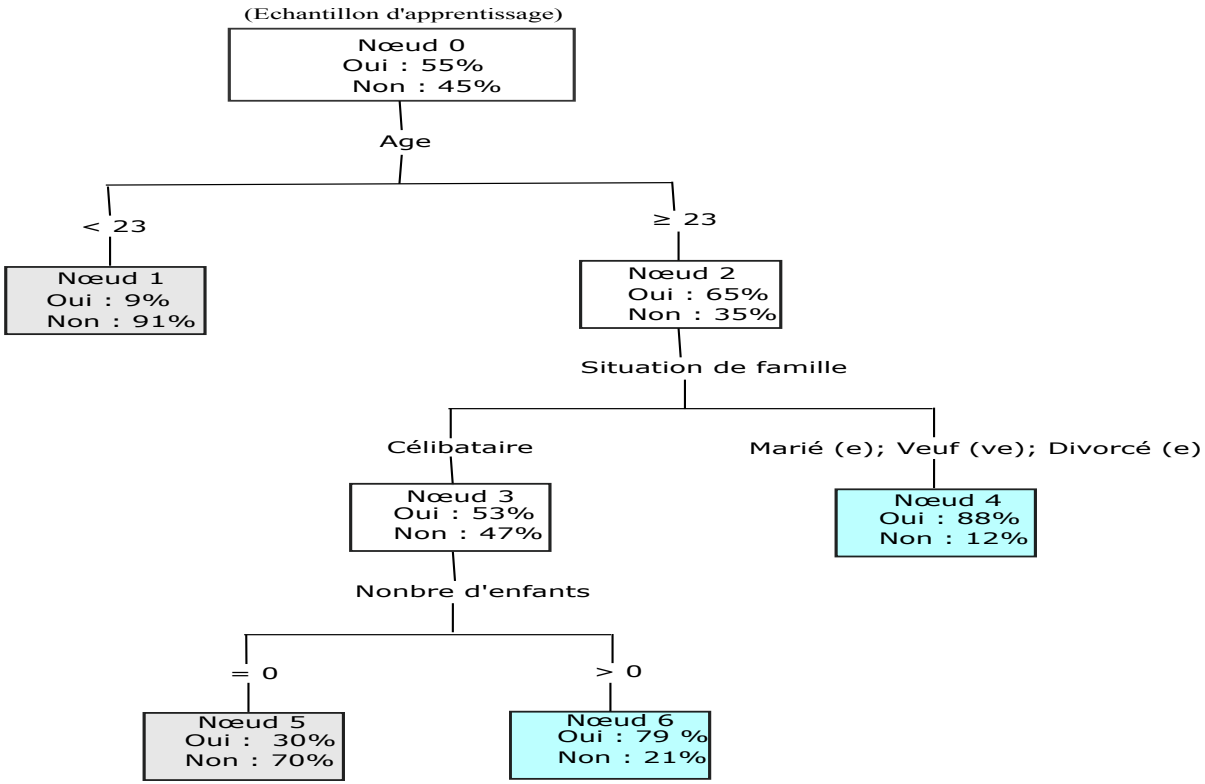
Algorithme

1. découper l'échantillon en L blocs $B_1, \cdots, B_L$ mutuellement disjoints de taille approximativement égale ;
2. pour tout $\ell = 1, \cdots, L$:
   - calculer l'erreur (notée $\widehat{\varepsilon}_\ell$) sur B_ℓ en utilisant l'ensemble $\{1, \cdots, n\} - B_\ell$ comme échantillon d'apprentissage ;
3. calculer la quantité $\widehat{\varepsilon} = \frac{1}{L} \sum_{\ell=1}^{L} \widehat{\varepsilon}_\ell$ qui est l'erreur estimée par validation croisée.

Arbre binaire de décision

Classification : Y (à K modalités) que l'on souhaite prédire à partir de $X = (X_1, \cdots, X_p)$ (quantitatives et/ou qualitatives).

Construction d'un arbre consiste à partir d'une racine (Nœud0) que l'on divise (de manière dichotomique) en une séquence de nœuds.



Arbre binaire de décision

Méthodologie générale

- On commence par la racine notée Nœud 0 que l'on partitionne en deux sous-ensembles ; on obtient ainsi des **nœuds fils**. Une division est définie par le choix conjoint d'une variable explicative et d'une condition de séparation (critère (C_1) permettant de sélectionner la "meilleur" division)
- On répète la même opération sur chaque nœud fils pour augmenter la discrimination de l'arbre jusqu'à ce qu'un critère d'arrêt (C_2) soit atteint.
- On affecte chaque nœud terminal à une classe
- On sélectionne un sous-arbre "optimal" au sens d'un critère que l'on a défini : c'est la procédure d'élagage.

Arbre binaire de décision

Critère de division C_1: Réduire l'hétérogénéité du nœud i.e. obtenir deux nœuds fils plus homogènes que le nœud père

L'hétérogénéité se mesure à l'aide d'une **fonction de pureté** (entropie, indice de Gini). Un nœud $\mathcal{N}$; posons $p_j := P(G_j/\mathcal{N})$,

$$\text{entropie}(\mathcal{N}) = -\sum_{j=1}^{K} p_j \log(p_j) \quad \text{avec la convention que } 0\log(0) = 0$$

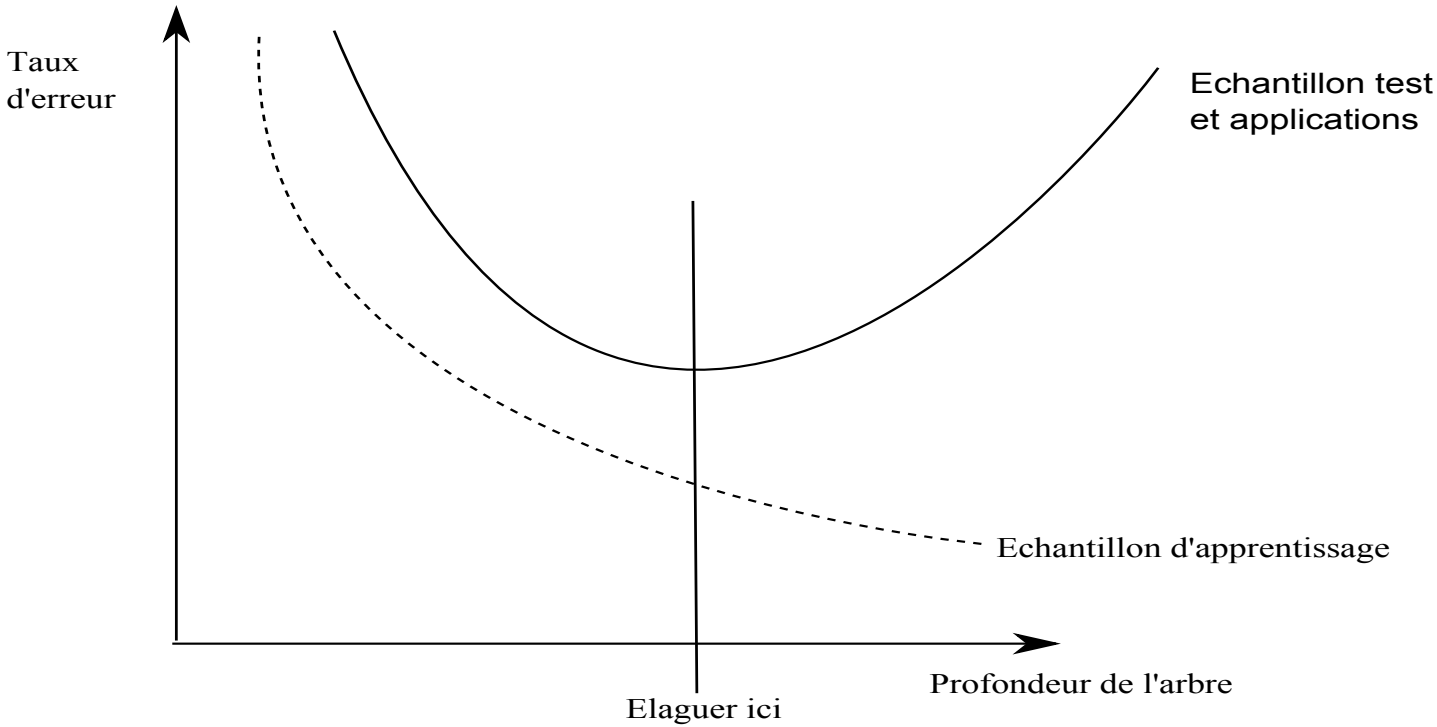
Critère d'arrêt C_2:

- la profondeur de l'arbre et/ou le nombre de feuilles ont atteint des limites fixées
- l'effectif de chaque nœud est inférieur à une valeur fixée
- la qualité de l'arbre est suffisante ou n'augmente plus de façon significative.

Arbre binaire de décision

Affectation: Par exemple, affecter chaque feuille à la classe majoritaire de la feuille.

Elagage: Algorithmes CART (Classification And Regression Tree)



Exemple: Titanic

Agrégation de modèles

Classification: $\mathcal{Y}$, fini, on dispose de K classes. On pose :

$$D_n = \{(Y_i, \underline{X}_i); i = 1, \cdots, n\} \text{ et } d_n = \{(x_i, \underline{x}_i); i = 1, \cdots, n\}.$$

Principe:

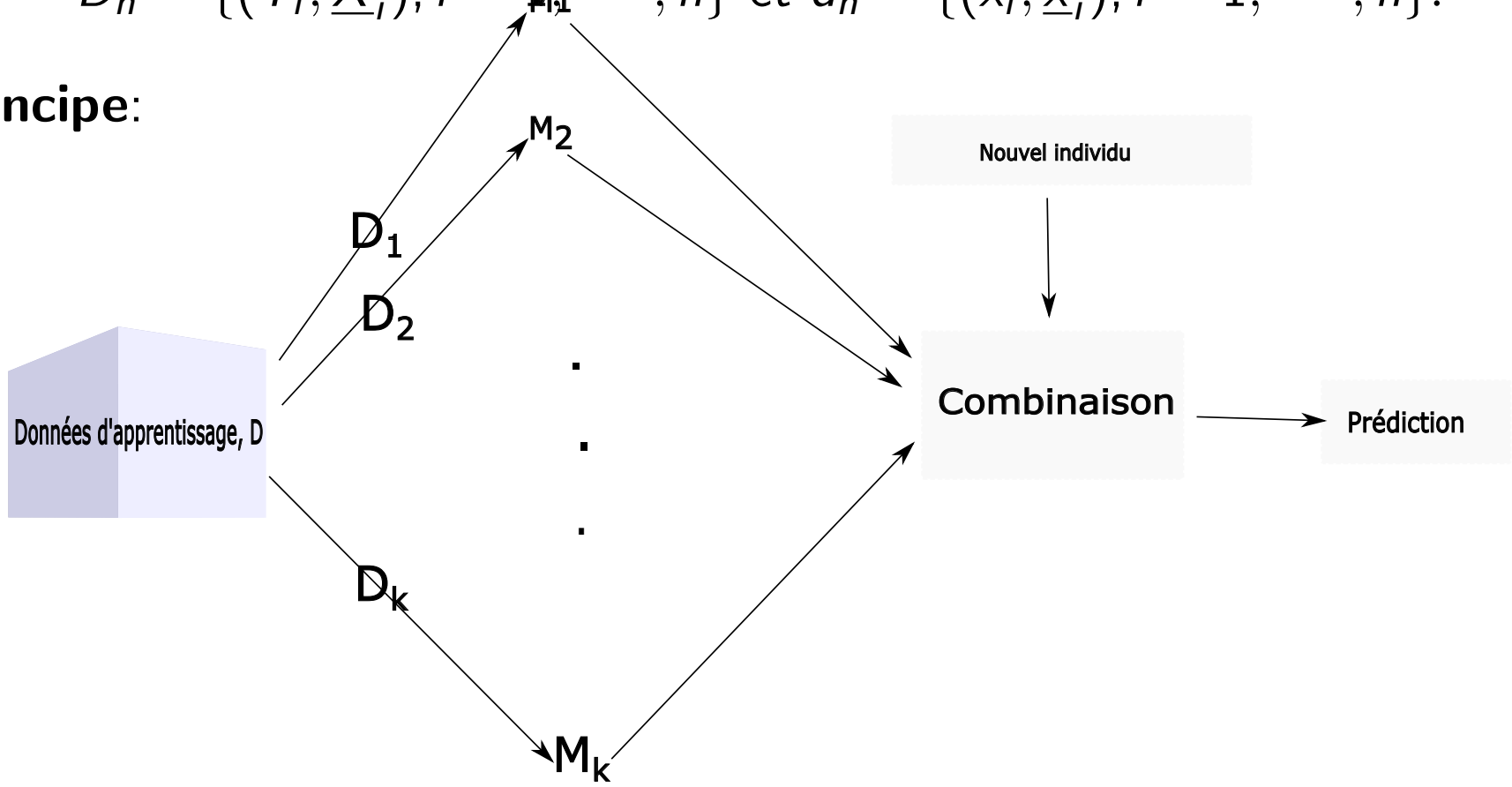


Figure:

Bagging

Bagging (pour **B**oostrap **Agg**regat**ing**), Breiman (1996)

Il consiste à faire coopérer des prédicteurs construits sur des échantillons bootstrap.

Soit $(d^*_{n,b})_{b=1,\cdots,B}$ B échantillons boostrap tirés de d_n ; et $(\widehat{f}_{d^*_{n,b}})_{b=1,\cdots,B}$ une famille de prédicteurs, où $\widehat{f}_{d^*_{n,d}}$ est construit à partir de $d^*_{n,b}$. Une prédiction par Bagging est:

$$\widehat{f}_B(\cdot) = \underset{y\in\mathcal{Y}}{\text{argmax}}\ \text{Card}\{b \mid \widehat{f}_{d^*_{n,b}}(\cdot) = y\}.$$

Il s'agit de constituer un comité pour voter et d'élire la prédiction la plus probable ; la prédiction obtenue correspond à un vote à la majorité simple.

Bagging

Algorithme du bagging :

Soit $d_n = \{(y_i, \underline{x}_i); i = 1, \cdots, n\}$ les observations de D_n et

i_0 un individu sur lequel on a fait l'observation $\underline{X}_{i_0} = \underline{x}_{i_0}$ et on souhaite prédire $Y_{i_0} = y_{i_0}$

Pour b=1 à B faire

Tirer (avec remise) un échantillon bootstrap de taille n, $d^*_{n,b}$

Estimer $\widehat{f}_{d^*_{n,b}}$ sur l'échantillon boostrap

Fin pour

Prédire y_{i_0} par $\underset{y \in \mathcal{Y}}{\text{argmax}}\ \text{Card}\{b \mid \widehat{f}_{d^*_{n,b}}(\underline{x}_{i_0}) = y\}$.

Bagging

Erreur out-of-bag:

- L'estimation out-of-bag (OOB) de l'erreur de prédiction se fait directement durant l'apprentissage.
- Pour chaque $(y_i, \underline{x}_i)$, considérer les modèles estimés sur un échantillon ne contenant pas cette observation (environ 37%).
- Prédire y_i comme précédemment et calculer l'erreur de prédiction associée

Coopération: Faire coopérer des modèles n'aura d'intérêt que si ces derniers ne classent pas tous de la même manière (si le vote est systématiquement unanime autant n'avoir qu'un seul modèle)

Inconvénient : Le bagging ne s'adapte pas efficacement avec les prédicteurs faibles (weak classifier)

Forêts aléatoires

Les forêts aléatoires (ou Random forests) sont une amélioration du bagging pour les arbres binaires

Le bagging est d'autant plus performant que les modèles sous-jacents (ici, les arbres binaires) sont moins corrélés.

Afin de diminuer cette corrélation, Breiman propose en 2001 de rajouter une couche aléatoire dans la construction des arbres

Le principe consiste à ajouter un tirage aléatoire de m variables parmi les p (avant chaque division) et de sélectionner la meilleure division uniquement sur ces m variables.

On suppose $\underline{X} = (X_1, \cdots, X_p)$

Forêts aléatoires

Algorithme des forêts aléatoires:

Soit $d_n = \{(y_i, \underline{x}_i); i = 1, \cdots, n\}$, i_0 un individu sur lequel on a observé $\underline{X}_{i_0} = \underline{x}_{i_0}$ et on souhaite prédire $Y_{i_0} = y_{i_0}$

Pour b=1 à B faire

Tirer (avec remise) un échantillon bootstrap de taille n, $d^*_{n,b}$

Estimer un arbre binaire sur $d^*_{n,b}$ avec randomisation i.e.

Pour chaque division : tirer m variables au hasard parmi les p et rechercher la division optimale dans ces m variables

On obtient alors le prédicteur $\widehat{f}_{d^*_{n,b}}$

Fin pour

Prédire y_{i_0} par $\underset{y \in \mathcal{Y}}{\text{argmax}}\ \text{Card}\{b \mid \widehat{f}_{d^*_{n,b}}(\underline{x}_{i_0}) = y\}$.

Forêts aléatoires

La prédiction obtenue correspond à un vote à la majorité simple.

Chaque modèle de base est alors sous-optimal, mais (comme l'union fait la force!), l'agrégation conduit finalement à de bons prédicteurs.

On peut choisir $m = \sqrt{p}$.

Forêts aléatoires

Importance des variables

- Un inconvénients des méthodes d'agrégation est que les modèles construits ne sont pas facilement interprétables, et sont qualifiés de "boîte noire".
- Dans beaucoup de situations pratique, il y a de grands intérêts à déterminer les variables explicatives qui jouent un rôle important dans la prédiction

Importance par l'erreur OOB :

- Permutation aléatoire des valeurs de la variable.
- Plus l'erreur OOB est dégradée par la permutation des valeurs de la j-ème variable, plus celle-ci est importante.

Boosting

Le Boosting:

- Le principe du boosting est de combiner plusieurs prédicteurs faibles (weak classifier) pour en obtenir un plus performant.
- Le prédicteur de base doit être meilleur que l'aléatoire i.e. taux d'erreurs inférieur à 0.5 pour une classification binaire.
- Chaque prédicteur faible est pondéré par la qualité de sa classification : mieux il classe, plus il sera important.
- Les observations mal classées auront un poids plus important (on dit qu'ils sont boostés) vis-à-vis de l'apprenant faible à l'étape suivante.

Boosting

On se place dans le cadre de la discrimination binaire avec par exemple $\mathcal{Y} = \{0, 1\}$ ou $\mathcal{Y} = \{-1, 1\}$.

L'un des algorithmes les plus utilisés en boosting est *AdaBoost* (adaptative boosting), proposé dans Freund et Schapire (1996).

Algorithme:

Soit $d_n = \{(y_i, \underline{x}_i); i = 1, \cdots, n\}$ les observations de D_n et i_0 un individu sur lequel on a fait l'observation $\underline{X}_{i_0} = \underline{x}_{i_0}$ et on souhaite prédire $Y_{i_0} = y_{i_0}$

Initialiser les poids : $\mathbf{w} = \{\omega_i = \frac{1}{n} \ ; \ i = 1, \cdots, n\}$

Boosting

Pour m=1 à M faire

Estimer le prédicteur $\widehat{f}_m$ sur l'échantillon pondéré par **w**

Calculer le taux d'erreur :

$$\epsilon_m = \frac{\sum_{i=1}^n \omega_i \mathbb{1}_{y_i \neq \widehat{f}_m(\underline{x}_i)}}{\sum_{i=1}^n \omega_i}$$

Calculer les logit : $\alpha_m = \log((1-\epsilon_m)/\epsilon_m)$

Calculer les nouvelles pondérations :

$$\omega_i \leftarrow \omega_i \exp\left(\alpha_m \mathbb{1}_{y_i \neq \widehat{f}_m(\underline{x}_i)}\right) \; ; \; i = 1, \cdots, n$$

Fin pour

Prédire y_{i_0} par $\underset{y \in \mathcal{Y}}{\text{argmax}} \sum_{m=1}^M \alpha_m \mathbb{1}_{\widehat{f}_m(\underline{x}_i)=y}$.

Cela correspond à un vote pondéré.

Boosting

Important : Dans la pratique, ajouter une procédure de contrôle pour vérifier que chaque prédicteur de base a un taux d'erreur inférieur à 0.5 ; cela garantie que les α_m restent positifs.

Dans cet algorithme, plus un prédicteur est performant (i.e. ϵ_m petit), plus son poids sera élevé (α_m grand), plus il pèsera dans le processus de décision.

Réseaux de neurones profonds

On se place dans le cadre **classification/régression** avec:

- $D_n = \{(X_1, Y_1), \cdots, (X_n, Y_n)\}$ un échantillon issu de (X, Y)
- $\mathcal{X} \subset \mathbb{R}^{d_x}$ l'espace des entrées, $\mathcal{Y} \subset \mathbb{R}^{d_y}$ l'espace des sorties
- $\ell : \mathbb{R}^{d_y} \times \mathcal{Y} \to [0, \infty)$ une fonction de perte
- $\mathcal{F} := \mathcal{F}(\mathcal{X}, \mathcal{Y})$ et $\mathcal{H} \subset \mathcal{F}$ un espace de prédicteurs
- $R(h) = \mathbb{E}\big[\ell\big(h(X), Y\big)\big]$, $\widehat{R}_n(h) = \dfrac{1}{n}\displaystyle\sum_{i=1}^{n} \ell\big(h(X_i), Y_i\big)$ risque et risque empirique de $h \in \mathcal{F}$
- Un oracle $h^* \in \underset{h \in \mathcal{F}}{\operatorname{argmin}} R(h)$, une cible sur $\mathcal{H}$ $h_{\mathcal{H}} \in \underset{h \in \mathcal{H}}{\operatorname{argmin}} R(h)$

Rappel objectif : Construire un prédicteur $\widehat{h}_n$ tel que $R(\widehat{h}_n)$ est proche de $R(h_{\mathcal{H}})$

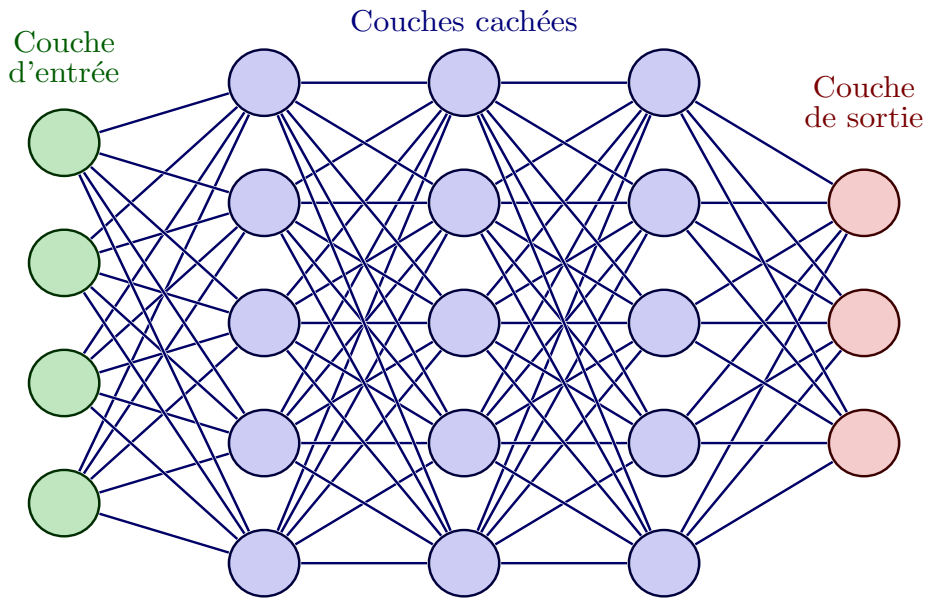
Réseaux de neurones profonds

Soit $L \in \mathbb{N}$, $\mathbf{p} = (p_0, p_1, \cdots, p_{L+1}) \in \mathbb{N}^{L+2}$, $\sigma : \mathbb{R} \to \mathbb{R}$ une fonction d'activation

Un **réseaux de neurones profond** (RNP) avec architecture $(L, \mathbf{p})$ est toute fonction de la forme,

$$h : \mathbb{R}^{p_0} \to \mathbb{R}^{p_{L+1}},\ x \mapsto h(x) = A_{L+1} \circ \sigma_L \circ A_L \circ \sigma_{L-1} \circ A_{L-1} \circ \cdots \circ \sigma_1 \circ A_1(x),$$

- A_ℓ : fonction affine,
- $\sigma_\ell : \mathbb{R}^{p_\ell} \to \mathbb{R}^{p_\ell}$, de la forme; $\sigma_\ell(z) = (\sigma(z_1), \cdots, \sigma(z_{p_\ell}))^T$,
- $p_0 = d_x$: dimension entrées
- $p_{L+1} = d_y$: dimension sorties



Exemple de fonction d'activation, ReLU: $\sigma(z) = \max(z, 0)$

Réseaux de neurones profonds

Approximation de fonctions par des réseaux de neurones profonds

Pour un RNP h d'architecture $(L, \mathbf{p})$, posons $\text{depth}(h) = L$ et $\text{width}(h) = \max_{1 \leq \ell \leq L} p_\ell$; et pour $L, N > 0$, considérons :

$$\mathcal{H}_{\sigma, d_x, d_y}(L, N) = \left\{\text{RNP } h,\ \text{depth}(h) \leq L,\ \text{width}(h) \leq N\right\}$$

Résultat (K. 2023): Soient $s > 0$ et l'espace de Hölder $\mathcal{C}^s(\mathcal{X})$

Pour tout $h \in \mathcal{C}^s(\mathcal{X})$, $\epsilon > 0$, $\exists h_{\epsilon,s} \in \mathcal{H}_{\sigma, d_x, d_y}\left(L_\epsilon, N_{\epsilon,s}\right)$ tel que,

$$\|h - h_{\epsilon,s}\|_\infty \leq \epsilon.$$

De plus, $N_{\epsilon,s}$ décroit quand s augmente.

Rappelons que :

$$\underbrace{R(\widehat{h}_n) - R(h^*)}_{\text{excès de risque}} = \underbrace{R(h_{\mathcal{H}}) - R(h^*)}_{\text{erreur d'approximation}} + \underbrace{R(\widehat{h}_n) - R(h_{\mathcal{H}})}_{\text{erreur d'estimation}}.$$

Réseaux de neurones profonds

Conséquence: Si l'oracle $h^* \in \mathcal{C}^s(\mathcal{X})$, alors on souhaite construire un prédicteur $\widehat{h}_n$ tel que $R(\widehat{h}_n) - R(h^*) \to 0$.

Minimisation du risque empirique:

Le prédicteur RNP satisfait

$$\widehat{h}_n = \underset{h \in \mathcal{H}_\sigma}{\operatorname{argmin}}\ \widehat{R}_n(h),$$

avec $\mathcal{H}_\sigma = \mathcal{H}_{\sigma,d_x,d_y}(L, N)$

Problème d'optimisation complexe : Car, grande dimension et non convexe.

Algorithme : ADAM (Adaptive moment estimation), SGD (Descente de Gradient Stochastique),$\cdots$

Merci pour votre attention.