

Définition 5.1 - *apprentissage*

L'*apprentissage* en informatique permet l'approche de différents problèmes :

- la *classification*, par exemple déterminer un objet sur une image, ou un son sur un flux audio ;
- la *régression*, par exemple prévoir la valeur du cours de la bourse.

Définition 5.2 - *apprentissage supervisé*

L'*apprentissage supervisé* consiste à entraîner un modèle ou un algorithme à l'aide d'un *ensemble d'apprentissage* :

$$S = \{(x_i, y_i), i \in \llbracket 1, n \rrbracket\} \subset X \times Y \quad \text{avec} \quad \begin{cases} X & \text{l'ensemble des } \textit{objets} \text{ manipulés par le modèle ou l'algorithme} \\ Y & \text{l'ensemble des } \textit{classes} \text{ ou } \textit{valeurs} \text{ associées aux objets de } X \end{cases}$$

$(x, y) \in S$ signifie que *l'objet x est dans la classe y ou a pour valeur y .*

On cherche pour un objet inconnu x à déterminer une classe ou une valeur y convenable en s'appuyant sur l'ensemble d'apprentissage S .

Définition 5.3 - *fonction de prédiction*

À tout modèle ou algorithme d'apprentissage supervisé on peut associer une *fonction $f : X \rightarrow Y$ dite de prédiction* qui à un objet associe la *classe estimée raisonnable par le modèle ou l'algorithme*.

Définition 5.4 - *fonction de perte*

À tout modèle ou algorithme d'apprentissage supervisé on peut associer une fonction $L : Y^2 \rightarrow \mathbb{R}_+$ qui à une prédiction associe une *valeur mesurant son inexactitude*. On a :

$$\forall y \in Y, L(y, y) = 0$$

Définition 5.5 - fonction de risque empirique

Pour tout modèle ou algorithme d'apprentissage supervisé on peut associer à toute fonction f de prédiction une espérance appelée *risque* R par :

$$\begin{aligned} R(f) &= \mathbb{E}_{X,Y} \left(L(Y, f(X)) \right) \\ &= \sum_{(x,y) \in X \times Y} L(y, f(x)) \mathbb{P}_{X,Y}(x, y) \end{aligned}$$

En pratique, on n'a jamais accès à $\mathbb{P}_{X,Y}$. On définit alors le *risque empirique* comme étant la moyenne des pertes sur l'ensemble d'apprentissage. Lui est calculable :

$$R_{\text{emp}}(f) = \frac{1}{|S|} \sum_{(x,y) \in S} L(y, f(x))$$

Dès lors, un algorithme d'apprentissage supervisé mettra en œuvre des algorithmes d'optimisation afin de trouver une fonction f qui minimise le risque empirique.

Implémentation - algorithme de classification des k plus proches voisins - classique

- **Entrée :**
 - un ensemble d'apprentissage $S \subset X \times Y$ indexé sur $\llbracket 0, n \rrbracket$
 - une distance $d : X^2 \rightarrow \mathbb{R}_+$
 - un objet x de classe inconnue
 - k le nombre de voisins à considérer
- **Sortie :** la classe y du voisin majoritairement présent parmi les k plus proches de x

```
1 | file = file de priorité max vide
2 | pour tout  $i \in \llbracket 0, n \rrbracket$  :
3 |     si  $|file| < k$  :
4 |         ajouter  $x_i$  à file avec la priorité  $d(x, x_i)$ 
5 |     sinon :
6 |         si  $d(x, x_i) < file[0]$  :
7 |             supprimer le maximum de file
8 |             ajouter  $x_i$  à file avec la priorité  $d(x, x_i)$ 
9 | renvoyer la classe majoritaire parmi les éléments de file
```

Implémentation - construction d'un arbre d -dimensionnel

Cet algorithme constitue un prétraitement de l'ensemble d'apprentissage, pour "faciliter" la recherche des k plus proches voisins en aval. Attention, si d est trop grand, on fait face au fléau de la dimension qui rend la méthode trop peu efficace.

- **Entrée :**
 - l'ensemble $X_S \subset X \subset \mathbb{R}^D$ pour lequel on connaît la classe de chaque objet
 - $i \in \llbracket 0, D-1 \rrbracket$ la coordonnée selon laquelle on trie
- **Sortie finale :** un arbre D -dimensionnel décrivant les positions relatives de chaque objet de X_S

```
1 | Construit( $i, X_S$ ) :  
2 |      $n = |X_S|$   
3 |     si  $n == 0$  :  
4 |         renvoyer Vide  
5 |     sinon :  
6 |          $X_S = \text{tri de } X_S \text{ dans l'ordre croissant de}$   
7 |              $\text{la } i\text{-ème coordonnée}$   
8 |          $\text{val} = X_S[\lfloor \frac{n}{2} \rfloor]$   
9 |          $X_g, X_d = \text{séparer strictement } X_S \text{ en l'indice } \lfloor \frac{n}{2} \rfloor$   
10 |         $g = \text{Construit}(d, i+1 \bmod D, X_g)$   
11 |         $d = \text{Construit}(d, i+1 \bmod D, X_d)$   
12 |        renvoyer Noeud(val, g, d)
```

Implémentation - *algorithme de classification des k plus proches voisins - avec arbre D -dimensionnel (1/2)*

- **Entrée :**
 - un arbre D -dimensionnel décrivant les positions relatives de chaque objet de $S \subset X \times Y$, avec $X \subset \mathbb{R}^D$
 - $i \in \llbracket 0, D - 1 \rrbracket$ la coordonnée courante
 - un objet x de classe inconnue
- **Sortie :** une pile contenant le chemin de la racine (en queue) vers une feuille de \mathcal{A} modélisant le plus proche voisin de x (en tête).

```
1 | explore( $\mathcal{A}$ ,  $i$ ,  $x$ ) :  
2 |     si  $\mathcal{A}$  est Vide :  
3 |         renvoyer []  
4 |     sinon :  
5 |         on identifie  $\mathcal{A} = \text{Noeud}(\text{val}, \text{g}, \text{d})$   
6 |          $x_i = i$ -ème coordonnée de  $x$   
7 |          $v_i = i$ -ème coordonnée de  $\text{val}$   
8 |         si  $x_i \leq v_i$  :  
9 |             renvoyer  $\mathcal{A} :: \text{explore}(\text{g}, i+1 \bmod D, x)$   
10 |        sinon :  
11 |            renvoyer  $\mathcal{A} :: \text{explore}(\text{d}, i+1 \bmod D, x)$ 
```

Implémentation - *algorithme de classification des k plus proches voisins - avec arbre D -dimensionnel (2/2)*

On munit $X \subset \mathbb{R}^D$ d'une distance $d : X^2 \rightarrow \mathbb{R}_+$

• **Entrée initiale :**

- une pile contenant le chemin de la racine (en queue) vers une feuille de \mathcal{A} modélisant le plus proche voisin de x (en tête)
- $i \in \llbracket 0, D - 1 \rrbracket$ la coordonnée courante
- un objet x de classe inconnue
- k le nombre de voisins à considérer

• **Sortie :** la classe y du voisin majoritairement présent parmi les k plus proches de x

```
1 file = file de priorité max vide
2 kPPV(1, i, x, k) :
3   Noeud(val, g, d), new_l = depiler l
4    $x_i$  =  $i$ -ème coordonnée de  $x$ 
5    $v_i$  =  $i$ -ème coordonnée de  $val$ 
6   si file[0].prio < d( $x_i E_i, v_i E_i$ ) // distance unidimensionnelle :
7     // dépend de  $d$  mais souvent égal à  $|x_i - v_i|$ 
8     si |file| < k :
9       ajouter  $x$  à la file avec la priorité d( $x, val$ )
10    sinon :
11      si file[0].prio > d( $x, val$ ) :
12        supprimer l'élément de priorité maximale de file
13        ajouter  $x$  à la file avec la priorité d( $x, val$ )
14
15    sinon : // risque de voisins plus proches,
16      // par lesquels on ne passerait pas : on va regarder
17      chemin = explore(Noeud(val, g, d),  $i \bmod D$ ,  $x$ )
18      kPPV(chemin, |chemin| +  $i \bmod D$ ,  $x$ ,  $k$ )
19
20    si |file| < k : // besoin d'encore des voisins ?
21      kPPV(new_l,  $i - 1 \bmod D$ ,  $x$ ,  $k$ ) // on remonte l'arbre
22
23    renvoyer la classe majoritaire parmi les éléments de file
```

Définition 5.6 - entropie de Shannon

Soit $S = \{(x_i, y_i), i \in \llbracket 1, n \rrbracket\} \subset X \times Y$ un ensemble d'apprentissage. Pour $y \in Y$, on définit $C_y = \{x \in X, (x, y) \in S\}$ l'ensemble des objets de classe y .

L'entropie de Shannon de S est :

$$H(S) = - \sum_{\substack{y \in Y \\ C_y \neq \emptyset}} \frac{|C_y|}{n} \ln \left(\frac{|C_y|}{n} \right)$$

L'entropie de Shannon mesure le désordre dans la distribution des classes de S .

Définition 5.7 - gain d'information

Soit $S \subset X \times Y$ un ensemble d'apprentissage, avec $X \subset A_1 \times \dots \times A_m \times X'$. Pour $i \in \llbracket 1, m \rrbracket$, On considère le critère A_i pouvant prendre les valeurs v_1, \dots, v_k . On écrit alors :

$$\begin{aligned} S &= \bigsqcup_{j=1}^k \left\{ (x, y) \in S, \underbrace{x_i}_{\substack{\text{valeur du critère} \\ A_i \text{ pour } x}} = v_j \right\} \\ &= \bigsqcup_{j=1}^k S_j \quad (\text{notation}) \end{aligned}$$

On définit le gain d'information du critère A_i comme :

$$G(S, A_i) = H(S) - \sum_{j=1}^k \frac{|S_j|}{|S|} H(S_j)$$

Le gain d'information renseigne sur la pertinence du critère A_i pour discriminer les objets de S .

Définition 5.8 - apprentissage non supervisé

L'apprentissage non supervisé repose des méthodes de *clustering* ou (ou *méthode de classification non supervisée*). Étant donné un ensemble $z = \{x_i, i \in \llbracket 1, n \rrbracket\}$, on en cherche une partition :

$$z = \bigsqcup_{i=1}^k P_i$$

Les $(P_i)_{i \in \llbracket 1, k \rrbracket}$ sont appelés *classes* ou *clusters*. L'objectif est que ces clusters vérifient :

- petite variabilité intra-classe
- grande variabilité inter-classe

Définition 5.9 - relation de finesse

Soit z un ensemble d'objets, de partitions en clusters $(P_i)_{i \in \llbracket 1, k \rrbracket}$ et $(P'_j)_{j \in \llbracket 1, k' \rrbracket}$. On dit que $(P_i)_{i \in \llbracket 1, k \rrbracket}$ est plus fine que $(P'_j)_{j \in \llbracket 1, k' \rrbracket}$ lorsque tout cluster P'_j est l'union de clusters P_i .

Définition 5.10 - variance d'une partition

Soit z un ensemble d'objets d'un espace métrique (E, d) , de partition en clusters $(P_i)_{i \in \llbracket 1, k \rrbracket}$. La variance de la partition $(P_i)_{i \in \llbracket 1, k \rrbracket}$ est :

$$V\left((P_i)_{i \in \llbracket 1, k \rrbracket}\right) = \sum_{i=1}^k \sum_{x \in P_i} d(x, g_i)$$

où, pour tout $i \in \llbracket 1, k \rrbracket$, g_i est le barycentre du cluster P_i :

$$g_i = \frac{1}{|P_i|} \sum_{x \in P_i} x$$

Définition 5.11 - hiérarchie d'un ensemble d'objets

Soit z un ensemble d'objets. On appelle hiérarchie de z toute famille finie $(\pi_i)_{i \in \llbracket 1, n \rrbracket}$ de partitions de z telle que :

1. $\forall i \in \llbracket 1, n-1 \rrbracket$, π_i est plus fine que π_{i+1}
2. π_1 est celle des singletons de z
3. $\pi_n = \{z\}$

On représente une hiérarchie à l'aide d'un *dendrogramme*.