Jeux, synthèse et contrôle

Notes par Sarah Larroze

Introduction

Ce cours s'intéresse à l'apprentissage (Machine learning). Cette notion est liée à la complexité, à la logique, à l'algorithmique et à la statistique. Le principe de l'apprentissage est de classifier des valeurs d'un domaine X à l'aide d'un label set Y. La plupart du temps, $Y = \{0, 1\}$. Il existe différents types d'apprentissage :

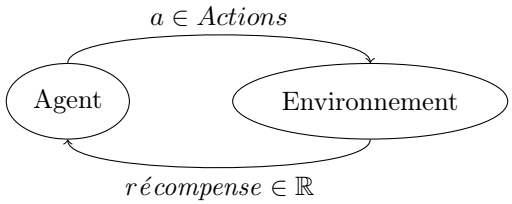
L'apprentissage supervisé
$$\begin{array}{lll} \text{Input} & : & \binom{x_1}{y_1}, \ldots, \binom{x_n}{y_n} \\ \text{Output} & : & h : X \to Y \end{array}$$

On obtient une classification si $Y = \{0, 1\}$ avec 0 qui étiquette les $x \in X$ négatifs et 1 qui étiquette les $x \in X$ positifs. Le but de l'apprentissage supervisé est de généraliser sur des entrées inconnues ce qui a été appris à partir des $\binom{x_1}{y_1}, \ldots, \binom{x_n}{y_n}$.

L'apprentissage non supervisé
$$\begin{array}{lll} \text{Input} & : & (x_1, \ldots, x_n) \\ \text{Output} & : & \text{une partition de } X \text{ en } k \text{ classes avec } k \in \mathbb{N} \end{array}$$

Le but de l'apprentissage non supervisé est de partitionner un ensemble d'entrées en fonction de similarités ou différences caractérisant cet ensemble.

Reinforcement learning (Apprentissage par récompense), noté RL



L'agent doit apprendre les actions à prendre pour maximiser la récompense en fonction de l'environnement.

Online learning (Apprentissage au fur et à mesure) Cet apprentissage correspond à une succession de phases d'apprentissage. Chaque phase d'apprentissage est constituée de la manière suivante :

$$\begin{array}{lll} \text{Input} & : & \binom{x_k}{y_k}, k \in \mathbb{N} \\ \text{Output} & : & h_k : X \to Y \end{array}$$

La fonction h_k retournée pour la phase k doit être meilleure que h_{k-1}.

1 Apprentissage supervisé

On rappelle que l'apprentissage supervisé est de la forme $\begin{array}{lll} \text{Input} & : & \binom{x_1}{y_1}, \ldots, \binom{x_n}{y_n} \\ \text{Output} & : & h : X \to Y \end{array}$.

S est appelé le "sample". Soit $\mathcal{H} \subseteq \{f : X \to Y\}$ l'ensemble des fonctions de retour acceptées pour l'apprentissage. On considère alors le problème suivant :

$$\begin{array}{lll} \text{Input} & : & S = \{(x_i, y_i) \backslash i \in [\![1, |S|]\!]\} \\ \text{Output} & : & \text{Trouver } h \in \mathcal{H} \text{ tq } h \models S \text{ (i.e } \forall i, h(x_i) = y_i) \end{array}$$

Définition Un algorithme apprend $\mathcal{H}$ si pour tout S, l'algorithme retourne $h \models S$ avec $h \in \mathcal{H}$ ou "il n'existe pas de $h \in \mathcal{H}, h \models S$". De plus, il est efficace s'il est polynomial en la dimension des x_i et le nombre de points et on note $poly(dim(x_i), |S|)$.

1.1 Formules conjonctives

Dans cette partie, $X = \{0,1\}^n, n \in \mathbb{N}$, $Y = \{0,1\}$ et $\mathcal{H} = \left\{\varphi = \bigwedge_{p \in X_1} p \wedge \bigwedge_{p \in X_2} \neg p\right\}$.
Par exemple, pour $n = 2$, $\mathcal{H} = \{p_1, p_2, p_1 \wedge p_2, p_1 \wedge \neg p_2, \neg p_1 \wedge p_2, \neg p_1 \wedge \neg p_2\}$.
On considère le sample $S = \{(x_1, y_1), \ldots, (x_5, y_5)\}$ avec $dim(x_i) = 5$ suivant :

	p_1	p_2	p_3	p_4	p_5		
x_1	0	1	0	1	1	1	y_1
x_2	0	1	1	1	0	1	y_2
x_3	0	1	1	0	1	0	y_3
x_4	1	0	1	1	0	0	y_4
x_5	0	0	0	1	1	0	y_5

On veut alors obtenir φ tq $x_1 \models \varphi$ et $x_2 \models \varphi$ et $x_3 \not\models \varphi$ et et $x_4 \not\models \varphi$ et $x_5 \not\models \varphi$. L'algorithme utilisé pour trouver une telle formule φ est de considérer la conjonction contenant tous les p_i et leur négation. On supprime au fur et à mesure les valeurs des p_i qui empêchent de satisfaire le sample. À la fin, si $\varphi \models S$ on retourne φ, sinon on retourne "il n'existe pas de φ".

Appliquons cet algorithme que le sample S précédent :
Initialement, $\varphi = p_1 \wedge \neg p_1 \wedge p_2 \wedge \neg p_2 \wedge p_3 \wedge \neg p_3 \wedge p_4 \wedge \neg p_4 \wedge p_5 \wedge \neg p_5$
$x_1 \models \varphi : \varphi = \cancel{p_1} \wedge \neg p_1 \wedge p_2 \wedge \cancel{\neg p_2} \wedge \cancel{p_3} \wedge \neg p_3 \wedge p_4 \wedge \cancel{\neg p_4} \wedge p_5 \wedge \cancel{\neg p_5}$
$x_2 \models \varphi : \varphi = \cancel{p_1} \wedge \neg p_1 \wedge p_2 \wedge \cancel{\neg p_2} \wedge \cancel{p_3} \wedge \cancel{\neg p_3} \wedge p_4 \wedge \cancel{\neg p_4} \cancel{\wedge p_5} \wedge \cancel{\neg p_5}$
On vérifie ensuite que : $x_3 \not\models \varphi$, $x_4 \not\models \varphi$, $x_5 \not\models \varphi$. C'est bien le cas ici donc on retourne $\varphi = \neg p_1 \wedge p_2 \wedge p_4$.

On note $n = dim(x_i)$, cet algorithme s'effectue en $O(n|S|)$ donc $\mathcal{H}$ est apprenable efficacement.

1.2 DNF (Disjonctive normal form)

On va à présent considérer le cas où $\mathcal{H}$ est l'ensemble des formules sous forme normale disjonctive. Par définition, φ est sous forme normale disjonctive si $\varphi = \bigvee_{i=1}^{k} C_i, C_i = \bigwedge_{p \in X_1} p \wedge \bigwedge_{p \in X_2} \neg p$. Dans cette partie, $X = \{0,1\}^n, n \in \mathbb{N}$ et $Y = \{0,1\}$.

On reprend l'exemple précédent et on cherche à nouveau à savoir s'il est possible d'obtenir une formule φ tq $\varphi \models S$.

Pour cela, il suffit de prendre :

$$\varphi = \underbrace{(\neg p_1 \wedge p_2 \wedge \neg p_3 \wedge p_4 \wedge p_5)}_{x_1 \models \varphi} \vee \underbrace{(\neg p_1 \wedge p_2 \wedge p_3 \wedge p_4 \wedge \neg p_5)}_{x_2 \models \varphi} \tag{1}$$

On peut donc toujours construire φ pour qu'elle ne satisfasse que les entrées à vérifier. Il faut ensuite vérifier qu'il n'y a pas d'incohérence dans le sample (i.e $\nexists x_i, x_j, i \neq j$ avec $y_i = y_j$). On remarque alors que $\mathcal{H}$ est trivialement apprenable. En revanche, le consistency model ne prend pas en compte la généralisation car il se contente d'apprendre les exemples positifs et ne met donc pas certaines propriétés en avant.

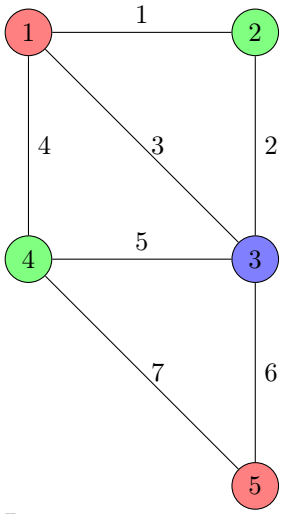
1.3 3-DNF

Dans cette section, $\mathcal{H}_3 = \varphi = C_1 \vee C_2 \vee C_3$. On s'intéresse au problème de la 3-coloration qui se résume au problème suivant :

Input	:	$G = (V, E)$ un graphe non-orienté
Output	:	$\exists c : V \to \{R, G, B\}\, tq\, \forall e = (u, v) \in E, c(u) \neq c(v)$

Étant donné un graphe G, on cherche à construire un sample $S = \{(x_i, y_i)\}$ tq G est 3-coloriable ssi $\exists \varphi \in \mathcal{H}_3 \backslash \varphi \models S$.

L'exemple suivant présente un graphe 3-coloriable (voir couleur des sommets) et le sample le représentant. La partie supérieure du sample représente les sommets du graphes et la partie inférieure représente ses arêtes.



Sommets

	p_1	p_2	p_3	p_4	p_5	y
x_1	1	0	0	0	0	1
x_2	0	1	0	0	0	1
x_3	0	0	1	0	0	1
x_4	0	0	0	1	0	1
x_5	0	0	0	0	1	1

Arêtes

	p_1	p_2	p_3	p_4	p_5	y
x_1	1	1	0	0	0	0
x_2	0	1	1	0	0	0
x_3	1	0	1	0	0	0
x_4	1	0	0	1	0	0
x_5	0	0	1	1	0	0
x_6	1	0	1	0	1	0
x_7	1	0	0	1	1	0

Prouvons que pour tout graphe G il est possible de construire un samlple $S = \{(x_i, y_i)\}$ tq G est 3-coloriable ssi $\exists \varphi \in \mathcal{H}_3 \backslash \varphi \models S$:

— Supposons que G est 3-coloriable, construisons alors φ tq $\varphi \models S$ en illustrant sur l'exemple précédent.
On construit d'abord une formule représentant la couleur rouge (ici les états 1 et 5) : $C_R = \neg p_2 \wedge \neg p_3 \wedge \neg p_4$. On procède de la même manière pour le vert et le bleu.
$C_v = \neg p_1 \wedge \neg p_4 \wedge \neg p_5$
$C_b = \neg p_1 \wedge \neg p_2 \wedge \neg p_4 \wedge \neg p_5$
On obtient alors $\varphi = C_r \vee C_v \vee C_b$. À partir d'un graphe 3-colorié, on peut donc toujours construire φ tq $\varphi \models S$.

— Supposons maintenant qu'on dispose d'une formule $\varphi = C_r \vee C_v \vee C_b$ tq $\varphi \models S$. Les sommets qui satisfont respectivement C_r, C_v ou C_b donnent la 3-coloration de G.