Apprentissage collaboratif supervisé

Fabien Torre

Université de Lille, LIFL, INRIA, équipe Magnet

EPAT 2014, Carry-le-Rouet

Collaboratif : qui fait appel à la collaboration de chacun.
Collaboration : action de travailler de concert,
de participer à une œuvre, avec d'autres.

Notions, notations, hypothèses

- Classification supervisée à deux classes $\mathcal{Y} = \{+1, -1\}$,
- des langages : $\mathcal{X}$ pour les exemples, $\mathcal{H}$ pour les hypothèses,
- échantillon S de n exemples étiquetés $(x_i \in \mathcal{X}, y_i \in \mathcal{Y})$,
- exemples : attributs continus, ni valeur manquante, ni bruit,
- erreur empirique (erreur_S) et erreur réelle (erreur),
- dimension VC et borne sur l'erreur [Burges, 1998] :

$$\text{erreur}(h) \leq \text{erreur}_S(h) + \sqrt{\frac{d_{VC}(\mathcal{H}).[\ln(\frac{2n}{d_{VC}(\mathcal{H})}) + 1] - \ln(\frac{\lambda}{4})}{n}}$$

Beaucoup de collaborations en vue !

On va se restreindre à la collaboration entre hypothèses et à la collaboration entre tâches.

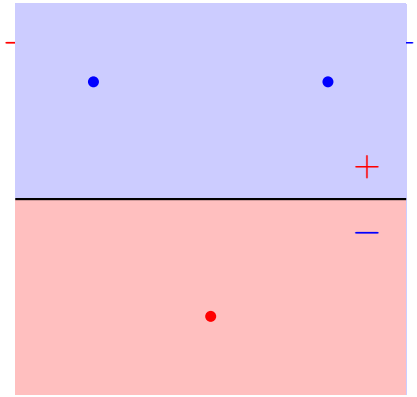
Plan du cours

Que seront les hypothèses ?

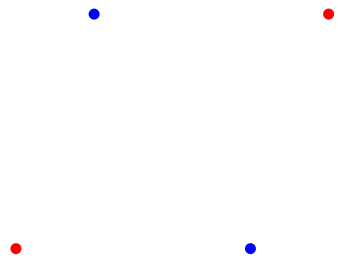
Plan du cours

$\mathcal{H}$: les droites

dimension VC pour une droite ?
au moins 3.



... mais pas 4, donc 3.

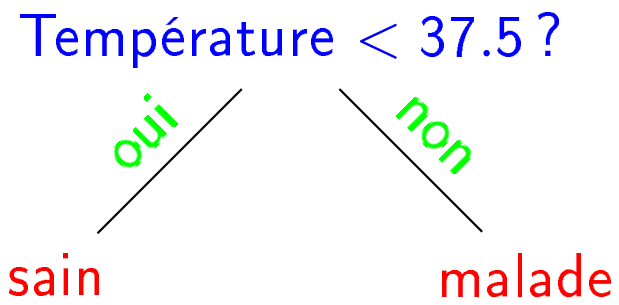


Cas particulier de ces droites : les *stumps*.

$\mathcal{H}$: les *stumps*

Définition : *stump*

Un *stump* est un arbre de décision à un seul nœud (un test et deux conclusions sur les classes).
C'est une droite perpendiculaire à l'axe du test.



Apprentissage par critère entropique, comme pour les arbres...
ou énumération de tous les *stumps*.

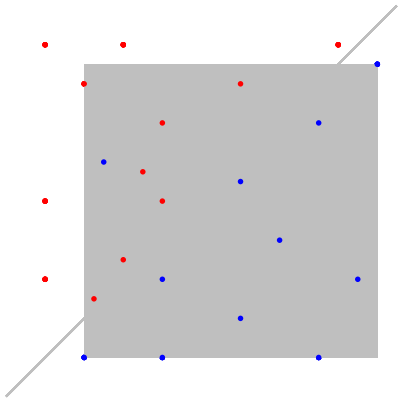
Rectangles parallèles aux axes

Un tel rectangle est défini par une valeur minimale et une valeur maximale (un intervalle) sur chaque attribut.

Généralisations d'attributs continus

5	1	2	3	1	+
5	4	7	3	2	+
[5 ;5]	[1 ;4]	[2 ;7]	[3 ;3]	[1 ;2]	+
6	8	3	3	7	+
[5 ;6]	[1 ;8]	[2 ;7]	[3 ;3]	[1 ;7]	+

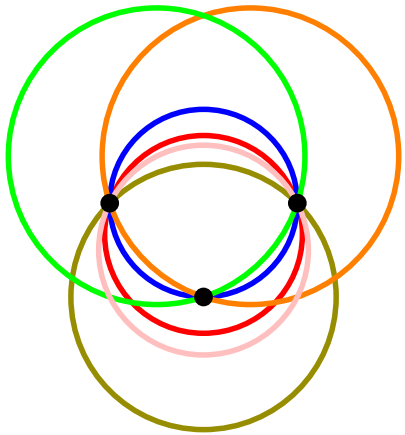
Généralisations de points dans le plan



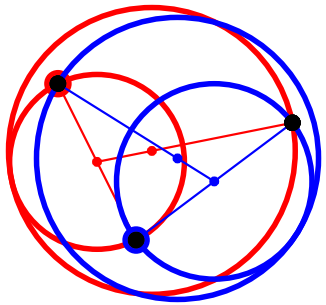
Démonstration : généralisation d'une classe en rectangle. démo

Disques

Dans le plan, il y a une infinité de disques incomparables couvrant trois points donnés.



Idée : on calcule le rayon minimal pour capturer le disque courant et le nouveau point, on en déduit le nouveau centre.



Sensibilité à l'ordre des exemples !

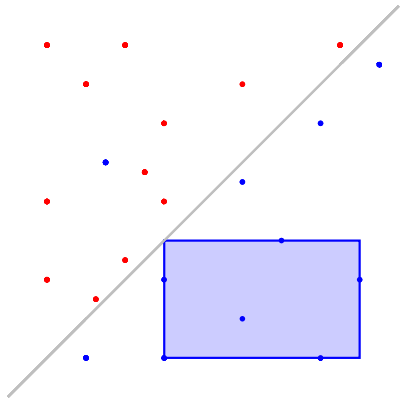
Démonstration : généralisation d'une classe en disque.

Généralisations correctes

Idée de la méthode

Pour obtenir une hypothèse correcte,

- on généralise en ajoutant un à un les exemples d'une même classe,
- on ne valide une généralisation que si aucun contre-exemple n'est couvert.



Dépendance à l'ordre des exemples.

Démonstration : rectangles/disques corrects, aléatoires, couvrants.

Algorithme de généralisation correcte

Entrées : $E = [e_1, \ldots, e_n] \subseteq \mathcal{X}$ un ensemble *ordonné* de n exemples de la même classe, N des contre-exemples.

Sortie : $h \in \mathcal{H}$ une généralisation correcte de E.

1: $h = e_1$
2: **for** $i = 2$ to n **do**
3: $h' = \mathtt{G}(h, e_i)$ { généralisation entre hypothèse+exemple. }
4: **if** $(\forall e \in N : h' \not\geq e)$ **then**
5: $h = h'$ { h', correcte, est la généralisation courante. }
6: **end if**
7: **end for**
8: **return** h

Plan du cours

Apprentissage disjonctif

Éléments d'apprentissage disjonctif :

- En pratique, il s'agit de couvrir tout exemple d'apprentissage,
- un concept est disjonctif s'il n'est pas dans $\mathcal{H}$ mais que la couverture est possible par l'union d'hypothèses de $\mathcal{H}$,
- une telle disjonction d'hypothèses est appelée une *théorie*.

Différents algorithmes connus :

1. DLG [Webb and Agar, 1992], algorithme glouton pour atteindre la couverture rapidement,
2. GloBo [Torre, 1999], méthode opérant un calcul de couverture minimale pour obtenir une théorie réduite et compréhensible.

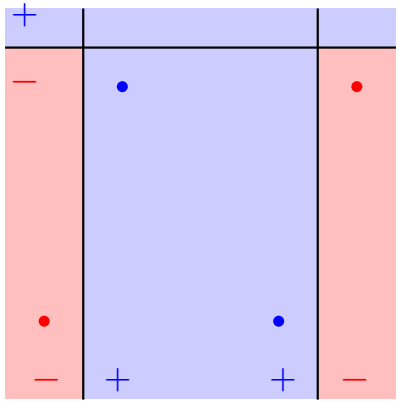
Démonstration : apprentissage rapide et théorie compréhensible.
DLG traduit la *collaboration* entre $\mathcal{H}$ et l'échantillon étiqueté.

Plan du cours

2. Apprentissage disjonctif, dimension VC et boosting
   - Collaborations d'hypothèses pour l'apprentissage disjonctif
   - Langage d'hypothèses augmenté
   - Une preuve de boosting

Combinaisons de trois droites et dimension VC

- Avec trois droites qui votent nous sortons de la classe des séparateurs linéaires,
- la dimension VC de cet espace d'hypothèses est > 4 :



Combien de points peuvent être *pulvérisés* par 3 droites ? une réponse

Plan du cours

Éléments du cadre PAC [Valiant, 1984]

- $\mathcal{D}$ une distribution sur $\mathcal{X}$, $c \in \mathcal{H}$ le concept cible,
- $\mathrm{EX}(c, \mathcal{D})$ un oracle qui tire un exemple de $\mathcal{X}$ suivant $\mathcal{D}$ et le fournit avec sa classification $\langle x, c(x) \rangle$,
- pas d'échantillon, l'apprenant L fait des appels à l'oracle,
- un apprenant L est dit fort si l'erreur de toute hypothèse apprise par L est inférieure à ϵ, quel que soit $\epsilon > 0$, quelle que soit la distribution $\mathcal{D}$ et quel que soit $c \in \mathcal{H}$,
- un apprenant L est dit faible s'il existe $\epsilon_0 < 0.5$ tel que l'erreur de toute hypothèse apprise par L est inférieure ou égale à ϵ_0, quelle que soit la distribution $\mathcal{D}$ et quel que soit $c \in \mathcal{H}$.

Remarque et question

- les rectangles sont fortement apprenables,
- *booster* un apprenant faible et avoir un apprenant fort ?

Algorithme de boosting de l'erreur

1. $h_1 = L(\mathrm{EX}(c, \mathcal{D}))$;
2. définir un nouvel oracle $\mathrm{EX}(c, \mathcal{D}_2)$ comme suit :
   1. on tire une pièce à pile ou face ;
   2. si *pile*, faire $x = \mathrm{EX}(c, \mathcal{D})$ jusqu'à $h_1(x) = c(x)$;
   3. si *face*, faire $x = \mathrm{EX}(c, \mathcal{D})$ jusqu'à $h_1(x) \neq c(x)$;
   4. renvoyer x.
3. $h_2 = L(\mathrm{EX}(c, \mathcal{D}_2))$;
4. définir un nouvel oracle $\mathrm{EX}(c, \mathcal{D}_3)$ comme suit :
   1. faire $x = \mathrm{EX}(c, \mathcal{D})$ jusqu'à $h_1(x) \neq h_2(x)$;
   2. renvoyer x.
5. $h_3 = L(\mathrm{EX}(c, \mathcal{D}_3))$;
6. renvoyer $h = \mathrm{VoteMajoritaire}(h_1, h_2, h_3)$.

Suite : preuve de *boosting* de l'erreur [Kearns and Vazirani, 1994].

Commentaires, objectif, notations

- $\mathcal{D}_2$ donne un poids global de 0.5 aux exemples bien classés par h_1 et 0.5 aux mal classés,
- il s'en suit que h_1 a une erreur de 0.5 sur $\mathcal{D}_2$ et que L, apprenant faible, ne peut pas apprendre h_1 sur $\mathcal{D}_2$,
- par construction, chaque h_i amène une information différente.

On doit montrer que l'hypothèse h fournie par L' vérifie :

$$\text{erreur}(h) < \epsilon_0$$

c'est-à-dire que L' fait strictement mieux que L.

Soient $\beta_1 = \text{erreur}_{\mathcal{D}}(h_1)$, $\beta_2 = \text{erreur}_{\mathcal{D}_2}(h_2)$ et $\beta_3 = \text{erreur}_{\mathcal{D}_3}(h_3)$.

D'une distribution à l'autre (1) : formules de passage

Considérons les exemples mal classés par h_1 :

$\mathcal{D}$	$\rightarrow$	$\mathcal{D}_2$
β_1	$\rightarrow$	$\frac{1}{2}$
$\mathcal{D}[x] = 2.\beta_1.\mathcal{D}_2[x]$	$\rightarrow$	$\mathcal{D}_2[x] = \frac{1}{2.\beta_1}.\mathcal{D}[x]$

... et les exemples bien classés par h_1 :

$\mathcal{D}$	$\rightarrow$	$\mathcal{D}_2$
$1-\beta_1$	$\rightarrow$	$\frac{1}{2}$
$\mathcal{D}[x] = 2.(1-\beta_1).\mathcal{D}_2[x]$	$\rightarrow$	$\mathcal{D}_2[x] = \frac{1}{2.(1-\beta_1)}.\mathcal{D}[x]$

D'une distribution à l'autre (2) : sur un exemple

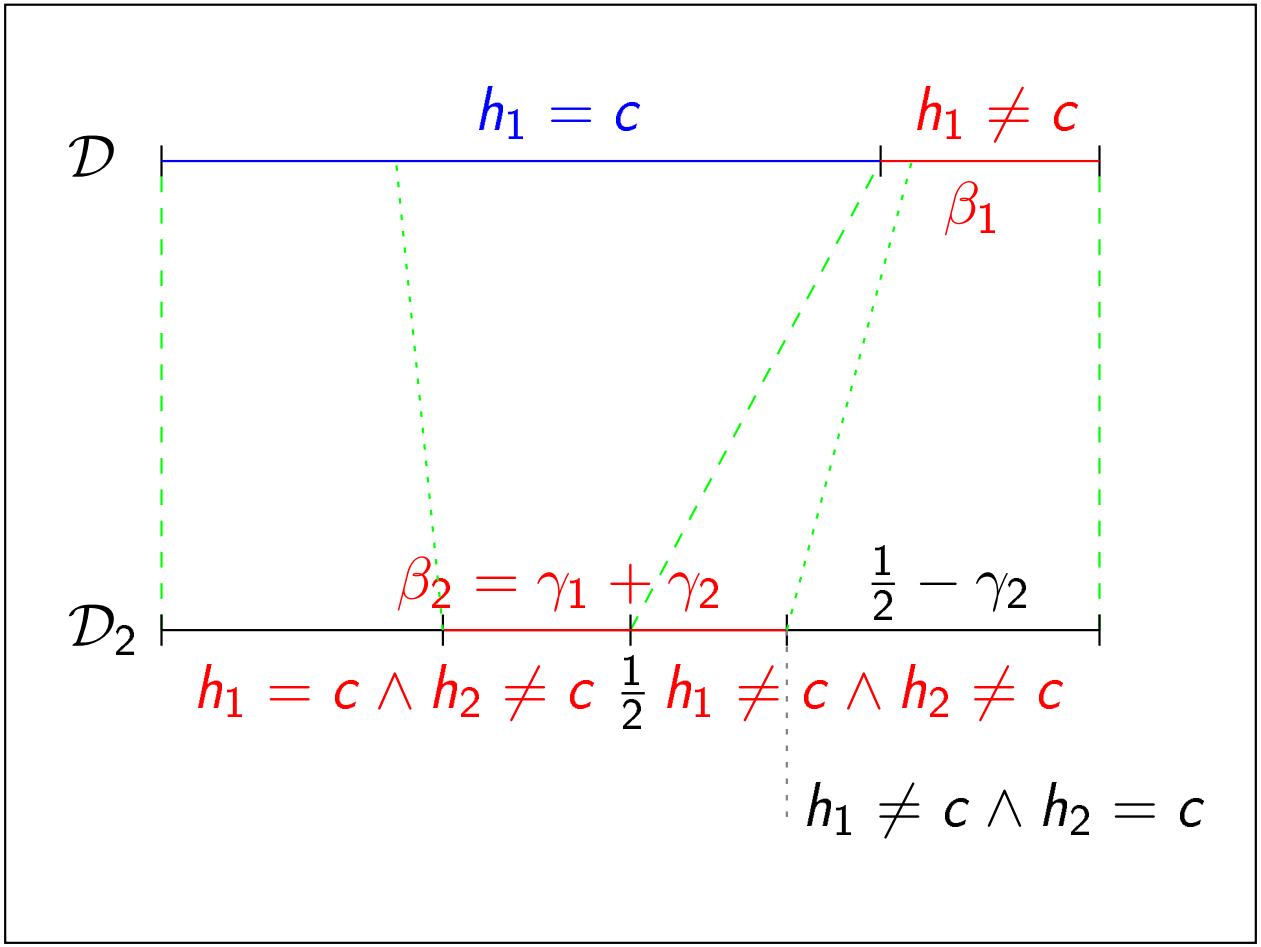
Cinq exemples, distribution uniforme

$$\mathcal{D} : \frac{1}{5} \quad \frac{1}{5} \quad \frac{1}{5} \quad \frac{1}{5} \quad \frac{1}{5}$$

- trois sont bien classés par h_1, deux en erreur : $\beta_1 = \frac{2}{5}$,
- facteur multiplicateur des mal classés : $\frac{1}{2.\beta_1} = \frac{5}{4}$,
- facteur multiplicateur des bien classés : $\frac{1}{2.(1-\beta_1)} = \frac{5}{6}$,

$$\mathcal{D}_2 : \frac{1}{6} \quad \frac{1}{6} \quad \frac{1}{6} \quad \frac{1}{4} \quad \frac{1}{4} \underbrace{\frac{1}{6} \quad \frac{1}{6} \quad \frac{1}{6}}_{=\frac{1}{2}} \underbrace{\frac{1}{4} \quad \frac{1}{4}}_{=\frac{1}{2}}$$

D'une distribution à l'autre (3) : notations complètes



Décomposition de l'erreur

Si h finale se trompe c'est que l'un des deux cas est survenu :

1. h_1 et h_2 sont d'accord mais se trompent,
2. h_1 et h_2 ne sont pas d'accord et h_3 se trompe.

ce qui se traduit formellement par :

$$\begin{aligned} \text{erreur}_{\mathcal{D}}(h) &= \Pr_{\mathcal{D}}[h_1(x) \neq c(x) \wedge h_2(x) \neq c(x)] \\ &+ \Pr_{\mathcal{D}}[h_3(x) \neq c(x) | h_1(x) \neq h_2(x)].\Pr_{\mathcal{D}}[h_1(x) \neq h_2(x)] \end{aligned}$$

On va développer explicitement, en vue de comparer à ϵ_0.

Décomposition de l'erreur : cas 1

h_1 et h_2 sont d'accord mais se trompent.

Par définition :

$$\Pr_{\mathcal{D}_2}[h_1(x) \neq c(x) \wedge h_2(x) \neq c(x)] = \gamma_2$$

et par application de la formule de passage de $\mathcal{D}_2$ vers $\mathcal{D}$:

$$\Pr_{\mathcal{D}}[h_1(x) \neq c(x) \wedge h_2(x) \neq c(x)] = 2.\beta_1.\gamma_2$$

Décomposition de l'erreur : cas 2

h_1 et h_2 ne sont pas d'accord et h_3 se trompe.

Par définition de $\mathcal{D}_3$:

$$\Pr_{\mathcal{D}}[h_3(x) \neq c(x) | h_1(x) \neq h_2(x)] = \Pr_{\mathcal{D}_3}[h_3(x) \neq c(x)] = \beta_3$$

On décompose la situation où h_1 et h_2 diffèrent :

$$\begin{aligned} \Pr_{\mathcal{D}}[h_1(x) \neq h_2(x)] &= \Pr_{\mathcal{D}}[h_1(x) = c(x) \wedge h_2(x) \neq c(x)] \\ &+ \Pr_{\mathcal{D}}[h_1(x) \neq c(x) \wedge h_2(x) = c(x)] \end{aligned}$$

Décomposition de l'erreur : cas 2.1 et cas 2.2

$$\begin{array}{rlcl} & \Pr_{\mathcal{D}_2}[h_1(x) = c(x) \wedge h_2(x) \neq c(x)] & = & \gamma_1 \\ \Rightarrow & \Pr_{\mathcal{D}}[h_1(x) = c(x) \wedge h_2(x) \neq c(x)] & = & 2.(1-\beta_1).\gamma_1 \end{array}$$

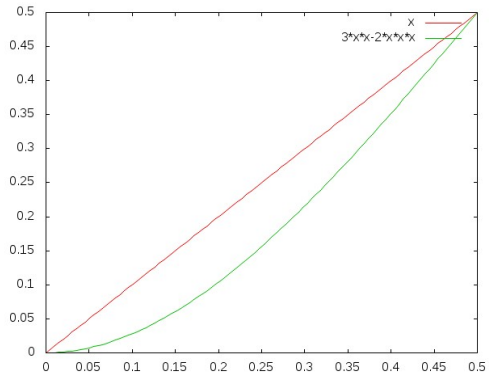
$$\begin{array}{rlcl} & \Pr_{\mathcal{D}_2}[h_1(x) \neq c(x) \wedge h_2(x) = c(x)] & = & \frac{1}{2} - \gamma_2 \\ \Rightarrow & \Pr_{\mathcal{D}}[h_1(x) \neq c(x) \wedge h_2(x) = c(x)] & = & 2.\beta_1.(\frac{1}{2} - \gamma_2) \end{array}$$

Calcul exact de l'erreur

$$
\begin{aligned}
\text{erreur}_{\mathcal{D}}(h) &= \Pr_{\mathcal{D}}[h_1(x) \neq c(x) \wedge h_2(x) \neq c(x)] \\
&+ \Pr_{\mathcal{D}}[h_3(x) \neq c(x) | h_1(x) \neq h_2(x)].\Pr_{\mathcal{D}}[h_1(x) \neq h_2(x)] \\
&= 2.\beta_1.\gamma_2 + \beta_3.\left[2.(1-\beta_1).\gamma_1 + 2.\beta_1.(\tfrac{1}{2} - \gamma_2)\right] \\
&= 2.\beta_1.\gamma_2 + 2.\beta_3.\gamma_1 - 2.\beta_1.\beta_3.\gamma_1 + \beta_1.\beta_3 - 2.\beta_1.\beta_3.\gamma_2 \\
&= \beta_1.(2.\gamma_2 - 2.\beta_3.\gamma_1 + \beta_3 - 2.\beta_3.\gamma_2) + 2.\beta_3.\gamma_1 \\
&= \beta_1.[2.\gamma_2 + \beta_3.(1 - 2.\beta_2)] + 2.\beta_3.\gamma_1
\end{aligned}
$$

Borne sur l'erreur

$$
\begin{aligned}
\text{erreur}_{\mathcal{D}}(h) &= \beta_1.[2.\gamma_2 + \beta_3.(1-2.\beta_2)] + 2.\beta_3.\gamma_1 \\
&\leq \epsilon_0.[2.\gamma_2 + \beta_3.(1-2.\beta_2)] + 2.\beta_3.\gamma_1 \\
&\leq 2.\gamma_2.\epsilon_0 + \beta_3.\epsilon_0.(1-2.\beta_2) + 2.\beta_3.\gamma_1 \\
&\leq \beta_3.[\epsilon_0.(1-2.\beta_2) + 2.\gamma_1] + 2.\gamma_2.\epsilon_0 \\
&\leq \epsilon_0[\epsilon_0.(1-2.\beta_2) + 2.\gamma_1] + 2.\epsilon_0.\gamma_2 \\
&\leq {\epsilon_0}^2 - 2.{\epsilon_0}^2.\beta_2 + 2.\epsilon_0.(\gamma_1+\gamma_2) \\
&\leq {\epsilon_0}^2 - 2.{\epsilon_0}^2.\beta_2 + 2.\epsilon_0.\beta_2 \\
&\leq {\epsilon_0}^2 + 2.\beta_2.(\epsilon_0 - {\epsilon_0}^2) \\
&\leq {\epsilon_0}^2 + 2.\epsilon_0.(\epsilon_0 - {\epsilon_0}^2) \\
\text{erreur}_{\mathcal{D}}(h) &\leq 3.{\epsilon_0}^2 - 2.{\epsilon_0}^3 < \epsilon_0
\end{aligned}
$$



L' réduit effectivement l'erreur de L !

Premier algorithme de boosting [Schapire, 1990] : 3 fois récursif !

Premier algorithme de boosting [Schapire, 1990]

On note $g(\epsilon) = 3.\epsilon^2 - 2.\epsilon^3$.

$\mathrm{ApprentissageFort}(\epsilon, \mathrm{EX}(c, \mathcal{D}))$:

1. si $\epsilon \geq \epsilon_0$, retourner $L(\mathrm{EX}(c, \mathcal{D}))$;
2. $\epsilon' = g^{-1}(\epsilon)$;
3. $h_1 = \mathrm{ApprentissageFort}(\epsilon', \mathrm{EX}(c, \mathcal{D}))$;
4. définir $\mathcal{D}_2$ en fonction de $\mathcal{D}$ et de h_1, comme précédemment ;
5. $h_2 = \mathrm{ApprentissageFort}(\epsilon', \mathrm{EX}(c, \mathcal{D}_2))$;
6. définir $\mathcal{D}_3$ en fonction de $\mathcal{D}$, h_1 et h_2, comme précédemment ;
7. $h_3 = \mathrm{ApprentissageFort}(\epsilon', \mathrm{EX}(c, \mathcal{D}_3))$;
8. retourner $h = \mathrm{VoteMajoritaire}(h_1, h_2, h_3)$.

Plan du cours

3. Méthodes d'ensemble
   - Généralités sur les méthodes d'ensemble
   - Méthodes aléatoires
   - Algorithme Adaboost
   - Justifications d'AdaBoost et des méthodes d'ensemble

On laisse de côté la technique du *bagging* [Breiman, 1996] et les *Random Forests* [Breiman, 2001].

Méthodes d'ensemble

Éléments constitutifs

Une méthode d'ensemble se définit par :

- l'espace des hypothèses $\mathcal{H}$,
- un apprenant faible L (abus de langage !),
- une stratégie utilisant L pour produire les $h_{t=1...T} \in \mathcal{H}$,
- une méthode de combinaison des h_t.

Sortie

Une méthode d'ensemble fournit un classifieur H :

- $H = [(h_t, \alpha_t)]_{t=1...T} \in (\mathcal{H}, \mathbb{R}_+^*)^T$
- $H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t . h_t(x)\right)$

Plan du cours

Méthodes aléatoires

Principes

- ne pas toucher à l'échantillon (*bagging*),
- ne pas jouer sur la distribution (*boosting*),
- mais simplement rendre l'apprenant instable avec de l'aléatoire,
- produire des hypothèses par appels à l'apprenant instable,
- et enfin, faire voter ces hypothèses à égalité.

Implémentations

- arbres de décision aléatoires [Dietterich, 2000],
- rectangles aléatoires, GloBoost [Torre, 2005].

Démonstration : rectangles et cercles aléatoires.

Plan du cours

Algorithme AdaBoost [Freund, 1995] : intuitions

- AdaBoost est un algorithme de *boosting* plus simple que la version récursive de [Schapire, 1990] car itératif,
- n appels groupés à l'oracle pour constituer un échantillon : ces exemples sont fournis à L à chaque itération,
- l'instabilité est introduite en jouant sur la distribution sur S,
- à chaque étape t,
  - L apprend h_t qui classe au mieux les exemples de poids forts,
  - puis les poids des exemples mal classés par h_t sont augmentés,
  - et les poids des exemples bien classés par h_t sont diminués.

Attention aux poids initiaux des exemples...

AdaBoost, version généralisée [Freund and Schapire, 1997] I

Entrées : n exemples (x_i, y_i) et T un nombre d'itérations.
Sortie : H le classifieur final.

1: **for** $i = 1$ to n **do**
2: $\mathcal{D}_1[x_i] = 1/n$ { initialisation de la distribution initiale }
3: **end for**
4: **for** $t = 1$ to T **do**
5: $h_t = L(\mathcal{D}_t)$ { appel à l'apprenant faible }
6: *choisir* $\alpha_t \in \mathbb{R}$ *comme « poids » de* h_t
7: **for** $i = 1$ to n **do**
8: $\mathcal{D}_{t+1}[x_i] = \mathcal{D}_t[x_i].\exp(-\alpha_t.y_i.h_t(x_i))$
9: **end for**
10: $Z_t = \sum_{i=1}^{n} \mathcal{D}_{t+1}[x_i] = \sum_{i=1}^{n} \mathcal{D}_t[x_i].\exp(-\alpha_t.y_i.h_t(x_i))$
11: **end for**

AdaBoost, version généralisée [Freund and Schapire, 1997] II

12: **for** $i = 1$ to n **do**

13: $\mathcal{D}_{t+1}[x_i] = \frac{\mathcal{D}_{t+1}[x_i]}{Z_t}$

14: **end for**

15: **return** $H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t . h_t(x)\right)$

AdaBoost en pratique

- l'échantillon est donné, peu importe sa taille,
- T est choisi arbitrairement,
- α_t et L doivent encore être définis.

Pour α_t et L, la minimisation de l'erreur empirique sert de guide.

α_t et minimisation de l'erreur empirique

$$\text{erreur}_S(H) = \frac{|\{i : H(x_i) \neq y_i\}|}{n} \leq \prod_{t=1}^{T} Z_t$$

... on cherche à minimiser Z_t, vu comme une fonction de α_t.

$$Z_t = \sum_{i=1}^{n} \mathcal{D}_t[x_i].\exp(-\alpha_t.y_i.h_t(x_i))$$
$$Z_t = \sum_{i:y_i \neq h_t(x_i)} \mathcal{D}_t[x_i].\exp(+\alpha_t) + \sum_{i:y_i = h_t(x_i)} \mathcal{D}_t[x_i].\exp(-\alpha_t)$$
$$Z_t = \epsilon_t.\exp(+\alpha_t) + (1-\epsilon_t).\exp(-\alpha_t)$$
$$Z'_t = \alpha_t.\epsilon_t.\exp(\alpha_t) - \alpha_t.(1-\epsilon_t).\exp(-\alpha_t)$$
$$Z'_t = 0 \Rightarrow \alpha_t.\epsilon_t.\exp(\alpha_t) = \alpha_t.(1-\epsilon_t).\exp(-\alpha_t)$$
$$Z'_t = 0 \Rightarrow \frac{\exp(\alpha_t)}{\exp(-\alpha_t)} = \frac{1-\epsilon_t}{\epsilon_t} \Rightarrow \exp(2\alpha_t) = \frac{1-\epsilon_t}{\epsilon_t}$$
$$Z'_t = 0 \Rightarrow 2\alpha_t = \ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$$

et on trouve $\alpha_t = \frac{1}{2}\ln(\frac{1-\epsilon_t}{\epsilon_t})$ avec $\epsilon_t = \sum_{i:y_i \neq h_t(x_i)} \mathcal{D}_t[x_i]$.

Stratégie pour l'apprenant L

Dans le même esprit, L peut chercher l'hypothèse $h \in \mathcal{H}$ qui minimise Z_t (ou qui minimise ϵ_t, ou qui maximise α_t).

Un apprenant L parfois possible

- énumérer $\mathcal{H}$ à l'avance,
- calculer à l'avance la couverture de chaque $h \in \mathcal{H}$,
- puis, à chaque étape du *boosting*, choisir la meilleure h.

Envisageable pour les *stumps* en général.

Si l'énumération exhaustive de $\mathcal{H}$ n'est pas possible, c'est à l'apprenant faible de considérer les poids des exemples.

... et pour les rectangles et les disques, quel L ? et quels α_t ? ! ?

Correction et abstentions [Torre, 2004] I

Particularités des hypothèses rectangles et disques

- Les hypothèses peuvent répondre -1, $+1$ ou 0,
- elles ne font pas d'erreur sur l'ensemble d'apprentissage.

Entrées : n exemples (x_i, y_i) et T un nombre d'itérations.
Sortie : H le classifieur final.

1: **for** $i = 1$ to n **do**
2: $\quad w_i = 1/n$ { on note maintenant $w_i = \mathcal{D}_t[x_i]$ }
3: **end for**
4: **for** $t = 1$ to T **do**
5: $\quad h_t = L([(x_i, y_i, w_i)])$
6: $\quad W_+ = \sum_{i:y_i.h_t(x_i)>0} w_i$
7: $\quad \alpha_t = \frac{1}{2} \ln \left(\frac{1 + W_+}{1 - W_+} \right)$

Correction et abstentions [Torre, 2004] II

$Z_t = \sum_{i}^{n} [w_i . \exp(-\alpha_t y_i h_t(x_i))]$

for $i = 1$ to n **do**

$w_i = \frac{w_i . \exp(-\alpha_t . y_i . h_t(x_i))}{Z_t}$

end for

end for

return $H(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t . h_t(x)\right)$

Implémentations

Boosting d'arbres

- *stumps* : *boostexter* [Schapire and Singer, 2000],
- JStumps : une implémentation Java du *boosting de stumps*,
- arbres complets : C5.0 [Quinlan, 2004], sous licence GPL.

Boosting rectangles et boules

- AdaBoost-MG [Tantini et al., 2010],
- implémentation Volata.

Démonstration : *boosting* de rectangles et disques, observation de l'erreur empirique.

Plan du cours

Explication par le cadre PAC

Conditions théoriques du *boosting*

- L est un apprenant faible,
- n est bien choisi en fonction de l'erreur à atteindre,
- T est bien choisi en fonction de l'erreur à atteindre.

Dans la vraie vie...

- on sait rarement déterminer si L est faible,
- pas d'oracle, donc pas de certitude d'avoir assez d'exemples,
- on ne sait pas comment choisir T.

Nous avons perdu la garantie de *booster* l'erreur réelle.

Explication par le compromis biais/variance

Décomposition de l'erreur d'une hypothèse h

- Entre c et $h^* \in \mathcal{H}$, on commet une erreur due au *biais*, c'est-à-dire au choix de $\mathcal{H}$,
- entre h^* et h, c'est l'erreur due à la *variance*, c'est-à-dire à l'algorithme et à l'échantillon.

- Limitation de l'erreur de biais par sortie de $\mathcal{H}$ pour un langage plus expressif : $(\mathcal{H}, \mathbb{R}_+^*)^T$,
- limitation de l'erreur de variance en évitant le choix d'une hypothèse $h \in \mathcal{H}$ particulière,
- mêmes idées que le compromis sur la dimension VC.

Arguments difficiles à formaliser, le compromis reste à trouver.
Comment choisir T ? Faut-il privilégier un $\mathcal{H}$ riche ou pauvre ?

Explication par les bornes sur les erreurs

... erreurs empirique et réelle [Freund and Schapire, 1997] :

$$\text{erreur}(H) \leq \text{erreur}_S(H) + \sqrt{\frac{T.d_{VC}(\mathcal{H})}{n}}$$
$$\text{erreur}_S(H) = \frac{|\{i:H(x_i)\neq y_i\}|}{n} \leq \prod_{t=1}^{T} Z_t < \exp\left(-2.T.(\tfrac{1}{2} - \epsilon_0)^2\right)$$

- L'erreur empirique va décroître exponentiellement vite,
- la borne sur l'erreur réelle suggère de ne pas laisser grandir T,
- arrêter dès que l'erreur empirique se stabilise ou devient nulle ?
- non, car l'erreur réelle baisse encore ensuite.

Démonstration : observation de l'erreur réelle.

L'explication par annulation de l'erreur empirique ne suffit pas.

Explication par les marges [Schapire et al., 1997]

Définition : Marge

La marge d'un exemple x est définie par : $m(x) = \frac{\sum_{t=1}^{T} \alpha_t . h_t(x)}{\sum_{t=1}^{T} \alpha_t}$

Avec une certaine confiance, on a pour tout θ :

$$\text{erreur}(H) \leq \Pr(y.m(x) \leq \theta) + \mathcal{O}\left(\sqrt{\frac{\ln(d_{VC}(\mathcal{H}))}{n.\theta^2}}\right)$$

- observation : AdaBoost fait grandir les marges des exemples d'apprentissage, après que l'erreur empirique soit à zéro,
- une interprétation géométrique.

Démonstration : observation des marges.

Maximisation des marges

AdaBoost fait grandir les marges... sans les maximiser.

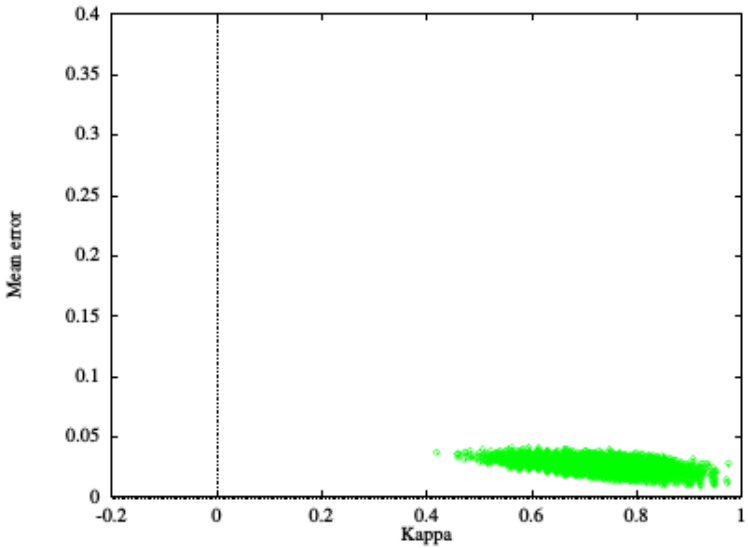
Travaux poussant l'idée plus loin :

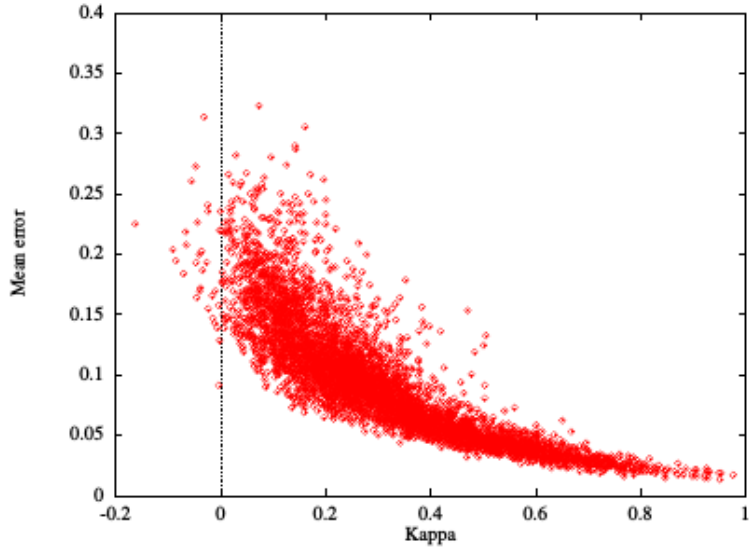
- maximiser les marges [Rätsch and Warmuth, 2003],
- mettre toutes les marges à 1 [Harries, 1999].

Ces méthodes font moins bien que AdaBoost...
la maximisation des marges n'est pas la réponse.

Explication par la diversité

AdaBoost produit des hypothèses bien plus *diverses* que les autres méthodes d'ensemble [Dietterich, 2000] :





Poursuivre sur cette piste :

- méthode favorisant la diversité [Melville and Mooney, 2004],
- mesures de la diversité [Kuncheva and Whitaker, 2003],
- faudrait-il produire des rectangles impurs ?

Plan du cours

Motivations

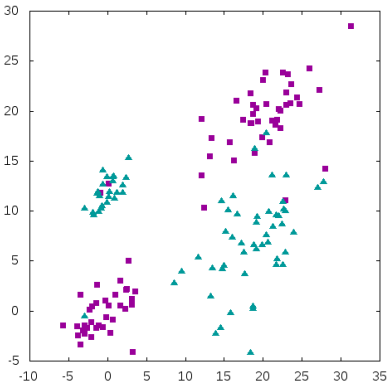
Objectifs : résoudre deux tâches simultanément,
obtenir des erreurs plus faibles qu'avec des apprentissages séparés.

Cadre choisi

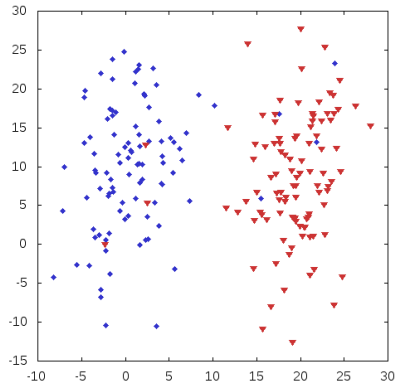
- partage de la même description des données ($\mathcal{X}$),
- tâches binaires, mais pas de correspondance entre les classes,
- un exemple d'apprentissage est étiqueté pour une tâche, pas forcément pour plus.

Hypothèse sous-jacente ? Les tâches doivent être *liées*... ?

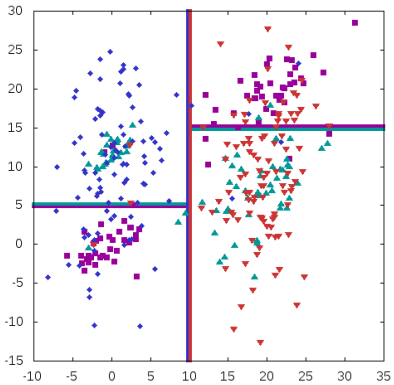
Deux tâches disjointes à traiter simultanément



tâche T_1



tâche T_2



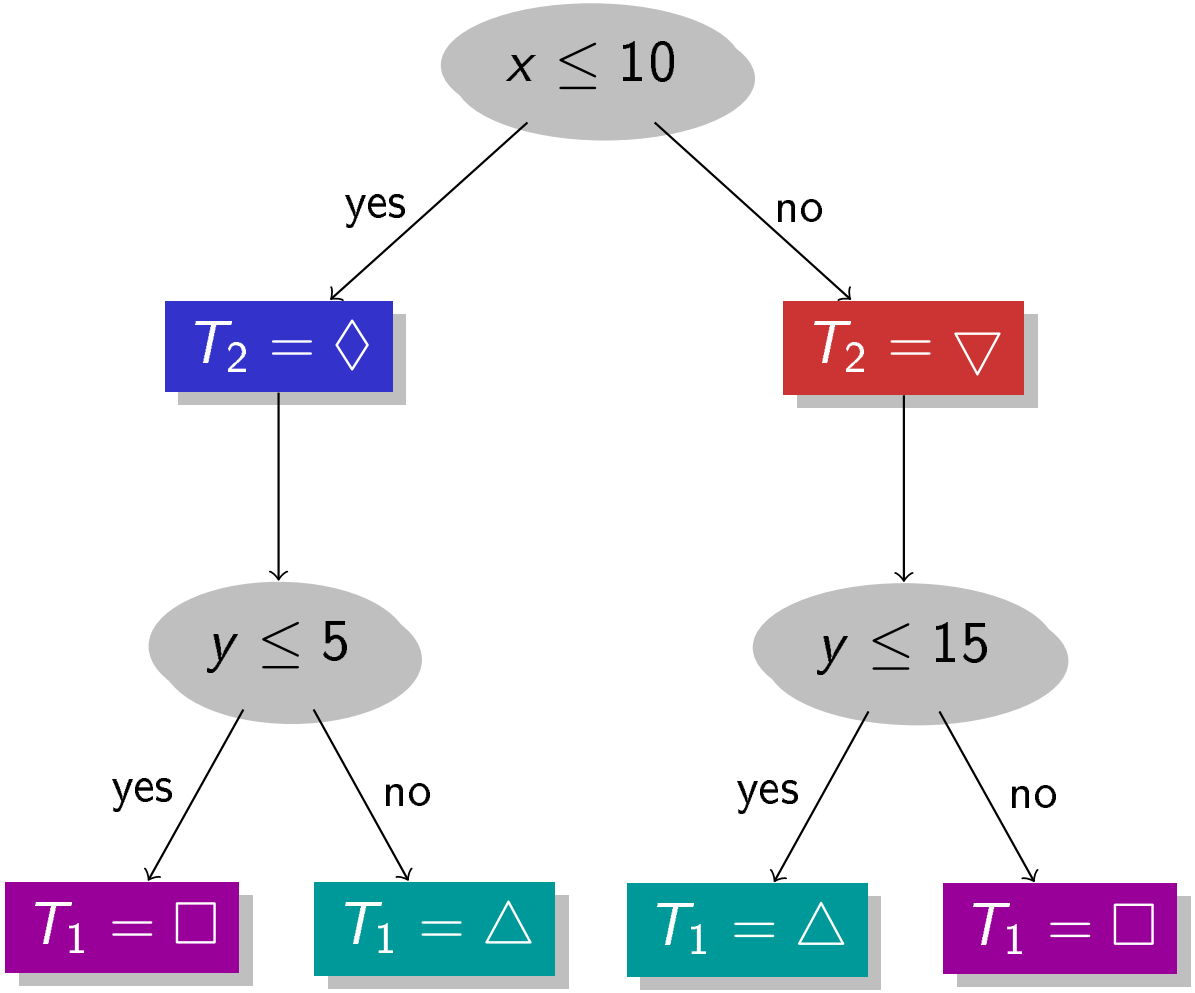
le séparateur pour T_2 permet de résoudre T_1

Une tâche réelle : la classification de *mails* (ENRON, *spam*, etc.).

Plan du cours

Hypothèse faible : le *bi-stump* [Faddoul et al., 2010]

- Un test, puis conclusion sur les classes d'une tâche,
- nouveaux tests aux feuilles et conclusion sur les classes de l'autre tâche.



AdaBoost pour le 2-tâches [Faddoul et al., 2010]

Quelques adaptations nécessaires :

- outre sa description et sa classe, on ajoute à chaque exemple la tâche concernée : (x_i, j_i, y_i) avec $x_i \in \mathcal{X}$, $j_i \in \{1, 2\}$, $y_i \in \mathcal{Y}_j$,
- si un point est étiqueté pour les deux tâches, on le duplique pour obtenir deux exemples,
- distribution $\mathcal{D}_1$ équilibrée par tâche (0.5 pour chacune), puisque l'on veut à la fin mesurer une erreur par tâche,
- projection des h_t sur une tâche j : $h_t(x, j)$,
- des *stumps* simples dans la compétition au cas où.

Plan du cours

Boosting et apprentissage N-tâches [Faddoul et al., 2011]

- N tâches alors que nos bi-stumps ne s'expriment que sur deux,
- nécessite une version d'AdaBoost avec abstentions et erreurs en apprentissage [Schapire and Singer, 1999],
- sommes des poids des exemples : W_+, W_- et W_0,
- condition d'apprenabilité faible : $W_+ > W_-$,
- L doit maximiser W_+ - W_-,
- règle habituelle pour la mise à jour des poids,
- la minimisation de Z_t amène à $\alpha_t = \frac{1}{2} \ln \left(\frac{W_+}{W_-} \right)$,
- on retrouve l'équivalent des bornes déjà vues.

Alternative pour le *boosting* N-tâches

- Changement de la règle de mise à jour des poids pour les exemples en abstention, multipliés par $\frac{\exp(\alpha_t)+\exp(-\alpha_t)}{2}$,
- cela conduit à $\alpha_t = \frac{1}{2} \ln \left(\frac{W_+ + \frac{1}{2} W_0}{W_- + \frac{1}{2} W_0} \right)$.

Comparaison

- on retrouve les mêmes bornes dans les deux cas,
- on regarde le nombre d'étapes de boosting nécessaires pour avoir $\prod_{t=1}^{T} Z_t < \epsilon$ pour un ϵ donné,
- on trouve respectivement $\mathcal{O}(-N \ln(\epsilon))$ et $\mathcal{O}(-N^2 \ln(\epsilon))$,
- expérimentalement on observe des comportements différents vis-à-vis de la borne (commune).

On garde la première instanciation de AdaBoost.

AdaBoost pour N tâches I

Entrées : n exemples (x_i, j_i, y_i) et T un nombre d'itérations.
Sortie : H le classifieur final.

1: **for** $i = 1$ to n **do**
2: $w_i = \frac{1}{N.S_{j_i}}$ { S_{j_i} : ensemble des exemples de la tâche j_i }
3: **end for**
4: **for** $t = 1$ to T **do**
5: $h_t = L([(x_i, j_i, y_i, w_i)])$
6: **for** $b \in \{-, 0, +\}$ **do**
7: $W_b = \sum_{i:\text{sign}(y_i h_t(x_i, j_i))=b} w_i$
8: **end for**
9: $\alpha_t = \frac{1}{2} \ln \left(\frac{W_+}{W_-} \right)$
10: $Z_t = \sum_i^n [w_i . \exp(-\alpha_t y_i h_t(x_i, j_i))] = W_0 + 2\sqrt{W_- W_+}$

AdaBoost pour N tâches II

11: **for** $i = 1$ to n **do**

12: $w_i = \frac{w_i . \exp(-\alpha_t . y_i . h_t(x_i, j_i))}{Z_t}$

13: **end for**

14: **end for**

15: **return** j classifiers $H_j(x) = \text{sign}\left(\sum_{i=1}^{T} \alpha_t h_t(x, j)\right)$

Expérimentalement : on a effectivement des erreurs plus faibles en apprenant simultanément [Faddoul et al., 2011].

Plan du cours

Bilan

Points saillants

- diversité entre hypothèses,
- instabilité des apprenants,
- aléatoire en apprentissage,
- faire les choses en même temps,
- le *boosting* comme méthode.

Mise en œuvre du *boosting*

- choix des hypothèses faibles,
- définition d'un apprenant faible,
- choix de la mesure d'erreur,
- définition de la distribution initiale,
- définition des coefficients α_t,
- réécriture des bornes et preuves,
- instanciation de l'algorithme AdaBoost.

Ouvertures

- *Online learning* [Littlestone and Warmuth, 1994],
- multi-agents [Bourgne et al., 2010],
- vie artificielle : les vraizamis (démonstration)
- apprentissage supervisé à base de fourmis démonstration

Plan du cours

Collaborations autour de l'étiquetage

À travers le type de dialogue installé entre l'enseignant et l'apprenant, l'étiquetage des exemples, le retour sur les prédictions.

1. récompense avec chaque prédiction de l'élève : *apprentissage par renforcement* [Sutton and Barto, 1998],
2. partage entre enseignants de leurs étiquetages : *apprentissage collaboratif*, *recommandation* [Candillier et al., 2007].

Collaborations avec les exemples

1. Exemples étiquetés et exemples non étiquetés : *apprentissage semi-supervisé*,
2. plusieurs représentations d'un même exemple : *apprentissage multi-vues*, *co-training*,
3. pour les deux : [Blum and Mitchell, 1998],
4. *apprentissage multi-instances* [Dietterich et al., 1997].

Entre classifieurs finaux

Typiquement deux critères à optimiser. Exemples :

- erreur à minimiser et compréhensibilité maximale,
- précision et rappel,
- etc.

L'apprenant doit proposer plusieurs classifieurs sur le front de Pareto (i.e. incomparables entre eux du point de vue multi-critères).

L'utilisateur en choisira un comme il l'entend.

Entre langages d'hypothèses

1. Union de deux langages d'hypothèses,
   démonstration : rectangles et disques dans le même $\mathcal{H}$,
2. *méthodes hybrides*, par exemple :
   des SVM placés à certaines feuilles d'un arbre de décision
   [Piltaver et al., 2014].

Là aussi une notion de diversité est centrale.

Entre méthodes d'apprentissage

1. Les *classes prédites* par différentes méthodes de premier niveau constituent une nouvelle description des exemples, utilisée par une méthode de second niveau : *stacked generalization* [Wolpert, 1992],
2. les *confiances dans chaque classe*, obtenues par une première méthode, enrichissent la description des exemples pour une seconde méthode : *apprentissage en cascade* [Gama and Brazdil, 2000].

Dans le même esprit, des numéros de *clusters* obtenus par une méthode non-supervisée peuvent enrichir la description d'exemples et aider une méthode supervisée.

Encore une fois, et pour finir : la diversité est importante.

Bibliographie I


Blum, A. and Mitchell, T. M. (1998).
Combining labeled and unlabeled sata with co-training.
In Bartlett, P. L. and Mansour, Y., editors, *COLT*, pages 92–100. ACM.


Bourgne, G., Soldano, H., and Fallah-Seghrouchni, A. E. (2010).
Learning better together.
In Coelho, H., Studer, R., and Wooldridge, M., editors, *ECAI*, volume 215 of *Frontiers in Artificial Intelligence and Applications*, pages 85–90. IOS Press.


Breiman, L. (1996).
Bagging predictors.
Machine Learning, 24(2) :123–140.

Bibliographie II


Breiman, L. (2001).
Random forests.
Machine Learning, 45(1) :5–32.


Burges, C. J. (1998).
A tutorial on support vector machines for pattern recognition.
Data Mining and Knowledge Discovery, 2 :121–167.


Candillier, L., Meyer, F., and Boullé, M. (2007).
Comparing state-of-the-art collaborative filtering systems.
In Perner, P., editor, *MLDM*, volume 4571 of *Lecture Notes in Computer Science*, pages 548–562. Springer.

Bibliographie III


Dietterich, T. G. (2000).
An experimental comparison of three methods for constructing ensembles of decision trees : Bagging, boosting, and randomization.
Machine Learning, 40(2) :139–158.


Dietterich, T. G., Lathrop, R. H., and Lozano-Pérez, T. (1997).
Solving the multiple instance problem with axis-parallel rectangles.
Artif. Intell., 89(1-2) :31–71.

Bibliographie IV


Faddoul, J. B., Chidlovskii, B., Gilleron, R., and Torre, F. (2011).
Learning many tasks with adaboost.
Technical Report -, inria.


Faddoul, J.-B., Chidlovskii, B., Torre, F., and Gilleron, R. (2010).
Boosting multi-task weak learners with applications to textual and social data.
In Draghici, S., Khoshgoftaar, T. M., Palade, V., Pedrycz, W., Wani, M. A., and Zhu, X., editors, *The Ninth International Conference on Machine Learning and Applications (ICMLA 2010)*, pages 367–372, Washington, DC, USA. IEEE Computer Society.

Bibliographie V


Freund, Y. (1995).
Boosting a weak learning algorithm by majority.
Information and Computation, 121(2) :256–285.


Freund, Y. and Schapire, R. E. (1997).
A decision-theoretic generalization of on-line learning and an application to boosting.
Journal of Computer and System Sciences, 55(1) :119–139.


Gama, J. and Brazdil, P. (2000).
Cascade generalization.
Machine Learning, 41(3) :315–343.

Bibliographie VI


Harries, M. (1999).
Boosting a strong learner : evidence against the minimum margin.
In *Proc. 16th International Conf. on Machine Learning*, pages 171–180. Morgan Kaufmann, San Francisco, CA.


Kearns, M. J. and Vazirani, U. V. (1994).
An Introduction to Computational Learning Theory.
MIT Press.


Kuncheva, L. I. and Whitaker, C. J. (2003).
Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy.
Machine Learning, 51(2) :181–207.

Bibliographie VII


Littlestone, N. and Warmuth, M. K. (1994).
The weighted majority algorithm.
Inf. Comput., 108(2) :212–261.


Melville, P. and Mooney, R. J. (2004).
Creating diversity in ensembles using artificial data.
Information Fusion : Special Issue on Diversity in Multiclassifier Systems.


Piltaver, R., Lustrek, M., Zupančič, J., Džeroski, S., and Gams, M. (2014).
Multi-objective learning of hybrid classifiers.
to be published in ECAI proceedings.

Bibliographie VIII


Quinlan, R. (2004).
Data mining tools see5 and c5.0.


Rätsch, G. and Warmuth, M. K. (2003).
Efficient margin maximizing with boosting.
submitted to Journal of Machine Learning Research (JMLR).


Schapire, R. E. (1990).
The strength of weak learnability.
Machine Learning, 5 :197–227.

Bibliographie IX


Schapire, R. E., Freund, Y., Bartlett, P., and Lee, W. S. (1997).
Boosting the margin : a new explanation for the effectiveness of voting methods.
In *Proc. 14th International Conference on Machine Learning (ICML)*, pages 322–330. Morgan Kaufmann.


Schapire, R. E. and Singer, Y. (1999).
Improved boosting algorithms using confidence-rated predictions.
Machine Learning, 37(3) :297–336.


Schapire, R. E. and Singer, Y. (2000).
Boostexter : A boosting-based system for text categorization.
Machine Learning, 39(2/3) :135–168.

Bibliographie X


Sutton, R. S. and Barto, A. G. (1998).
Reinforcement learning : An introduction.
IEEE Transactions on Neural Networks, 9(5) :1054–1054.


Tantini, F., Terlutte, A., and Torre, F. (2010).
Sequences classification by least general generalisations.
In Sempere, J. M., editor, *10th International Colloquium on Grammatical Inference (ICGI 2010)*, pages 189–202, Valencia (Spain). Springer-Verlag.


Torre, F. (1999).
GloBo : un algorithme stochastique pour l'apprentissage supervisé et non-supervisé.
In Sebag, M., editor, *Actes de la Première Conférence d'Apprentissage*, pages 161–168.

Bibliographie XI


Torre, F. (2004).
Boosting correct least general generalizations.
Technical Report 0104, GRAppA.


Torre, F. (2005).
Globoost : Combinaisons de moindres généralisés.
Revue d'Intelligence Artificielle, 19(4-5) :769–797.


Valiant, L. G. (1984).
A theory of the learnable.
Communications of the ACM, 27 :1134–1142.

Bibliographie XII


Webb, G. I. and Agar, J. W. M. (1992).
Inducing diagnostic rules for glomerular disease with the DLG machine learning algorithm.
Artificial Intelligence in Medicine, 4 :419–430.


Wolpert, D. H. (1992).
Stacked generalization.
Neural Networks, 5(2) :241–259.