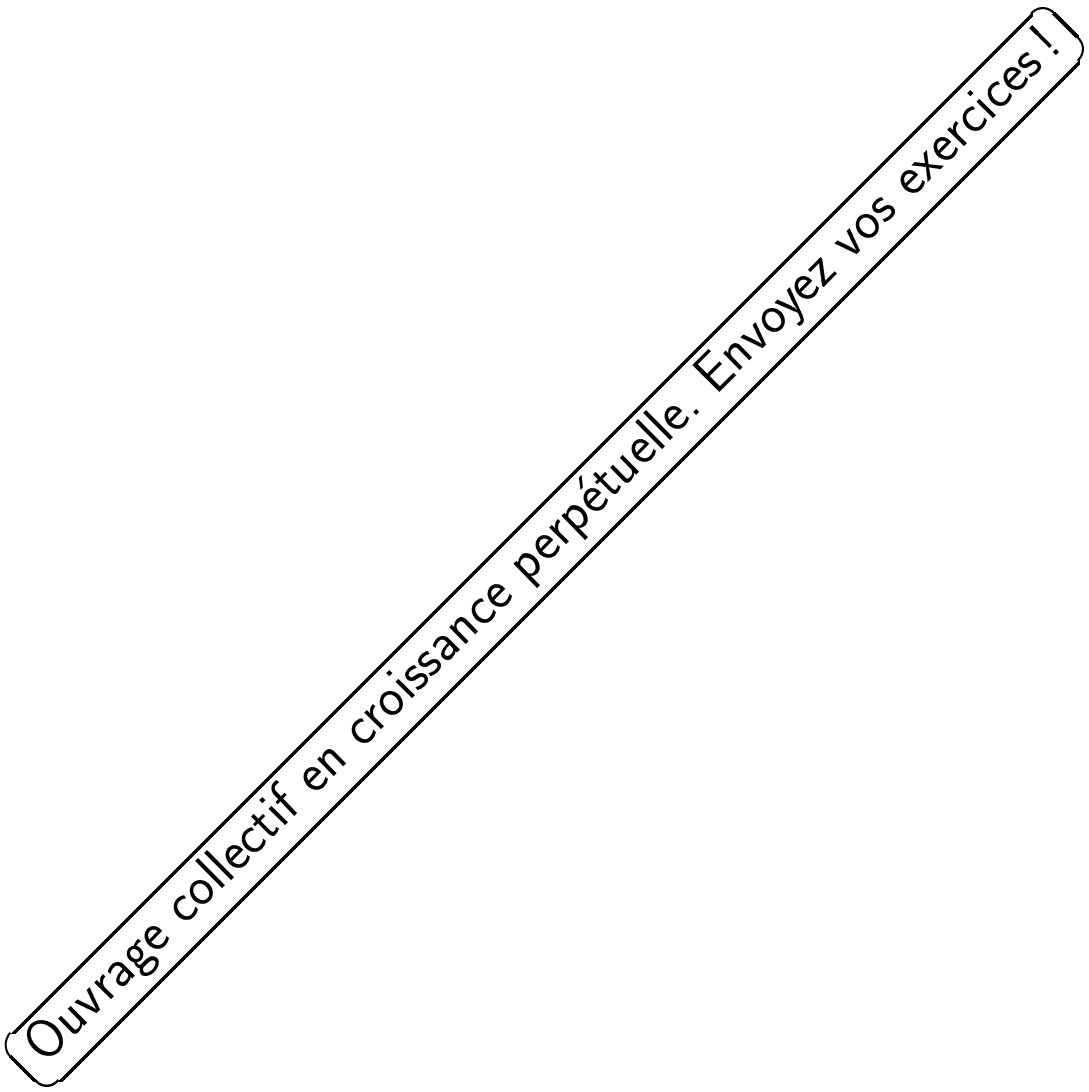
||

~

Apprentissage artificiel :

concepts et algorithmes

Recueil d'exercices



Version du 20 janvier 2003

Présentation

Ce recueil d'exercices accompagne le livre

Apprentissage artificiel, concepts et algorithmes,
par A. Cornuéjols et L. Miclet, Eyrolles 2002

Il est composé essentiellement d'envois effectués par les lecteurs depuis la sortie du livre et ne demande qu'à s'agrandir. Pour qui veut l'enrichir, il n'y a qu'à communiquer un énoncé (si possible avec la solution) à l'un des deux auteurs du livre par un courriel à l'adresse : `miclet@enssat.fr` ou `antoine@lri.fr`, de préférence en code LaTeX ou en texte.
Le recueil des solutions est destiné aux enseignants. Il n'est disponible que sur demande et expédié par courrier papier.
Merci à tous ceux et celles qui ont contribué et contribuent encore à ce recueil d'exercices ; ils sont cités ci-dessous. L'origine de l'exercice est indiquée dans le texte par les initiales du ou des auteur(s). Les exercices sans nom d'auteur sont des classiques que l'on peut considérer comme appartenant au patrimoine collectif. Les exercices « papier » sont étiquetés selon leur niveau de difficulté par un seul  (facile), ou deux  (plus difficile) ou trois  (très difficile, ou niveau recherche). Les exercices de programmation sont indiqués par  .
Pour la qualité de leurs exercices (mais pas seulement pour cela), nous recommandons vivement les livres [Duda et al., 2001], [Mitchell, 1997], [Nilsson, 1998], [Russel and Norvig, 1995]. Certains de leurs exercices ont été adaptés ici. A part le second, ces ouvrages possèdent aussi un recueil de solutions qui est envoyé à la demande par leur éditeur sur support papier.
Ce recueil est organisé en suivant les chapitres du livre. Une liste de sites web où se trouvent des logiciels sur le sujet traité est donnée après chaque chapitre d'exercices.

ACLM:	Antoine Cornuéjols et Laurent Miclet
FDRG:	François Denis et Rémi Gilleron
AB:	Abdel Belaïd
CR:	Céline Rouveirol
CHJO:	Colin de la Higuera et José Oncina
FC:	François Coste
JCJ	Jean-Christophe Janodet
FT	Franck Thollard
LL	Laurence Likforman

Chapitre 2

Première approche théorique de l'induction

Exercice 2.1  Comment se ramasser aux tests de QI

Les tests de mesure de QI font souvent appel à la faculté d'induction. Par exemple, il est fréquent d'y trouver des questions comme celle-ci :
« Donner le nombre a qui poursuit la suite : 1 2 3 5 ... »
Question. Trouver un argument pour chacunes des valeurs suivantes de a :
$a = 6$
$a = 7$
$a = 8$
$a = 2\pi$ (ou n'importe quel nombre, d'ailleurs).
Quelle est « la » bonne valeur ? Pourquoi ?

ACLM

Exercice 2.2  Comment se ramasser aux tests de QI (suite)

Les test de QI ne sont pas que des suites de chiffres à compléter. Il y a parfois des *carrés latins* à finir de remplir. Généralement, sous un graphisme plus ou moins transparent, il s'agit d'un problème comme celui-ci :

A 1	B 2	C 3
C 2	A 3	B 1
B 3	C 1	? ?

Question. Quel est le couple de valeurs manquantes que vous mettriez ? Pourquoi ?
Montrer que l'on peut construire un carré latin avec les mêmes données, en répondant avec les valeurs :

N'importe quoi N'importe quoi

ACLM

Exercice 2.3  maximum *a priori*, maximum de vraisemblance, maximum *a posteriori*

Soit une population d'individus qui consiste en un échantillon composé d'ouvriers, de médecins et d'employés des télécoms. On décrit les individus par un attribut logique *repondeur* qui vaut vrai si l'individu possède un répondeur téléphonique et faux sinon. L'espace de description est donc égal à l'ensemble

$\{repondeur, \overline{repondeur}\}$. On souhaite répartir les individus en trois classes *ouvrier*, *medecin* et *telecom*. On dispose des informations suivantes :

classe ω_i	*telecom*	*medecin*	*ouvrier*
$P(\omega_i)$	0.2	0.3	0.5
$P(repondeur/\omega_i)$	1	0.9	0.45

Une première règle de décision possible pour le choix de la règle de classification est d'attribuer à chaque description la classe majoritaire *a priori*, c'est-à-dire celle pour laquelle $P(\omega_i)$ est maximum ; c'est la règle *majoritaire* ou du maximum *a priori*. Elle ne tient compte que des probabilités *a priori* des classes et pas de l'observation..

Une seconde règle consiste à raisonner ainsi : si j'observe $\boldsymbol{x}$, je choisis la classe pour laquelle cette observation est la plus probable, c'est-à-dire celle pour laquelle $P(\boldsymbol{x}/\omega_i)$ est maximum. C'est la règle du *maximum de vraisemblance*.

La troisième, la *règle de décision bayésiene* ou du *maximum a posteriori* consiste à attribuer à une description $\boldsymbol{x}$ la classe ω_i qui maximise la probabilité *a posteriori* $P(\omega_i/\boldsymbol{x})$ qu'un élément ayant $\boldsymbol{x}$ pour description soit de classe k. La quantité $P(\omega_i/\boldsymbol{x})$ peut être estimée en utilisant la formule de Bayes, il suffit donc de choisir la classe k qui maximise le produit $P(\boldsymbol{x}/\omega_i)P(\omega_i)$.

Décrire sur cet exemple les trois règles de décision $C_{majoritaire}$, $C_{vraisemblance}$, C_{Bayes}.

On peut définir la probabilité d'erreur d'une règle de classification de la façon suivante. Soit R une règle de classification ; la probabilité d'erreur de R pour une description $\boldsymbol{x}$ est la probabilité qu'un élément de la population Π de description $\boldsymbol{x}$ soit mal classé par R ; l'erreur $E(R)$ d'une règle de classification est la moyenne pondérée des erreurs sur les descriptions d.

Calculer les erreurs pour les trois procédures de classification données précédemment.

FDRG

Chapitre 3

L'environnement méthodologique de l'apprentissage

Lois de probabilité, variance

Exercice 3.1 **Variance de la loi binomiale**

On dispose d'une pièce de monnaie biaisée. On appelle q la probabilité qu'elle tombe sur *Pile*. On lance cette pièce n fois. On obtient r *Pile*. L'expérience de lancer n fois cette pièce peut être réalisée à souhait. On s'attend à obtenir différentes valeurs de r. Soit R la variable aléatoire de valeur le nombre r de *Pile* obtenu en n lancers. On s'intéresse à la probabilité d'obtenir $R = 0$ *Pile* en n lancers, à la probabilité d'obtenir $R = 1$ *Pile* en n lancers, ..., à la probabilité d'obtenir $R = n$ *Pile* en n lancers.

1. Soit $q = 1/4$ et $n = 3$, expliciter les quantités $P(R = 0)$, $P(R = 1)$, $P(R = 2)$ et $P(R = 3)$.
2. Soit q et n quelconques, expliciter $P(R = r)$ en fonction de q et n. Une telle loi est la *loi binomiale* de paramètres n et q
3. On dispose de tables pour calculer les valeurs. Par exemple, pour $q = 1/4$ et $n = 13$, les probabilités obtenues arrondies à deux décimales sont :

0	1	2	3	4	5	6	7	8	9,13
0,02	0,11	0,20	0,25	0,21	0,13	0,06	0,01	0,01	0

   Faites une représentation graphique. Quelle est la probabilité d'obtenir un nombre de *Pile* inférieur ou égal à 5 en 13 lancers avec une pièce de proba $1/4$ pour *Pile*?
4. Vous lancez 20 fois une pièce dont la probabilité d'obtenir *Pile* est $1/4$, combien de *Pile* vous attendez vous à obtenir ? Cette notion est capturée par la notion d'espérance mathématique. L'espérance mathématique d'une variable aléatoire discrète X est définie par :

$$E(X) = \Sigma_{\omega \in \Omega} X(\omega) P(\omega)$$

   L'espérance d'une loi binomiale X de paramètres n et q est $E(X) = nq$
5. Vous lancez 20 fois une pièce dont la probabilité d'obtenir *Pile* est $1/4$, vous vous attendez à obtenir 5 *Pile* mais vous êtes conscient que des variations entre le nombre de *Pile* obtenu et le nombre attendu sont possibles. La variance et l'écart-type vont permettre de mesurer ceci. La variance d'une variable aléatoire discrète X est définie par :

$$V(X) = E((X - E(X))^2)$$

   et représente l'erreur quadratique attendue entre valeur observée et valeur attendue. L'écart-type est la racine carrée de la variance :

$$\sigma(X) = \sigma_X = \sqrt{V(X)}$$

   La variance d'une loi binomiale X de paramètres n et q est $V(X) = nq(1 - q)$ et l'écart-type est $\sigma_X = \sqrt{nq(1 - q)}$. Calculez espérance mathématique, variance et écart-type dans les cas suivants :
   - $q = 1/4$ et $n = 20$;

- $q = 1/4$ et $n = 40$;
- $q = 1/4$ et $n = 100$;
- $q = 1/4$ et $n = 400$.

FDRG

Estimation de l'erreur d'une hypothèse

Exercice 3.2 **Intuition et réalité**

Nous sommes dans un univers U, une loi de probabilité fixée mais inconnue existe sur U, on cherche à atteindre une cible f. Le système d'apprentissage a fourni une hypothèse h appartenant à un ensemble d'hypothèses H. On dispose également d'un ensemble test T de m exemples tiré indépendamment de h (et donc de l'ensemble d'apprentissage) selon la loi de probabilité sur l'univers . On souhaite estimer l'erreur réelle de h qui est la probabilité que h et f diffèrent sur un exemple tiré avec la loi de probabilité sur U.

1. Le but de cette question est de préciser et de quantifier votre « bon sens ». T contient $m = 100$ exemples, h fait $r = 25$ erreurs ; quelle est votre estimation de l'erreur réelle ? T contient $m = 500$ exemples, h fait $r = 130$ erreurs ; quelle est votre estimation de l'erreur réelle ? À laquelle de ces deux estimations faîtes vous le plus confiance ?
2. soit $p = e(h)$ l'erreur réelle de h. Soit X la variable aléatoire qui a pour valeur le nombre d'exemples mal classés lorsque l'on tire un ensemble test T de m exemples. Montrer que X suit une loi binomiale.
3. soit r le nombre d'erreurs sur T. L'erreur estimée est r/m. Cette estimation r/m de p donne-t-elle en moyenne la bonne estimation ? Pour cela, on définit : le *biais d'estimation* d'un estimateur Y de p est la quantité $E(Y) - p$. Si ce biais est 0, on dit que Y est un *estimateur sans biais de p*. Soit Y la variable aléatoire de valeur l'erreur estimée r/m. Montrer que Y est un estimateur de p [1].
4. On s'intéresse maintenant à la variance et l'écart-type de cet estimateur. En utilisant le fait que X est une loi binomiale, (on connaît la variance et l'écart-type pour une loi binomiale), et le fait que m est constant, on obtient que la variance de l'erreur estimée est $V(Y) = p(1-p)/m$ et que l'écart-type de l'erreur estimée est $\sigma_Y = \sqrt{p(1-p)/m}$. Ne connaissant pas p, nous le remplaçons par sa valeur estimée r/m, nous obtenons une estimation de l'écart-type pour l'erreur estimée :

$$\sigma_Y \simeq \sqrt{\frac{r/m(1-r/m)}{m}}$$

   Calculez l'erreur estimée et l'écart-type dans les cas suivants :
   - $m = 20$ et $r = 4$;
   - $m = 100$ et $r = 25$;
   - $m = 200$ et $r = 48$;
   - $m = 500$ et $r = 130$.
5. L'inégalité de Bienaymé-Tchebychev valable pour toute variable aléatoire X d'espérance $E(X)$ et d'écart-type σ_X s'énonce :

$$P\big[|X - E(X)| \leq k\sigma_X\big] \geq 1 - \frac{1}{k^2} \text{ pour } k \geq 1$$

   Utilisez cette inégalité pour déterminer le nombre d'exemples à tirer pour que l'erreur estimée ne s'écarte pas de plus de 5 % de l'erreur réelle avec une confiance de 99 %. On utilisera le fait que $\sigma_Y \leq 1/\sqrt{m}$. Même question avec moins de 5 % de l'erreur réelle avec une confiance de 96 %. Interprétez l'inégalité lorsque l'on fait tendre le nombre d'exemples vers l'infini.

1. Cet estimateur est sans biais car il y a une hypothèse d'indépendance entre T et h.

FDRG

Exercice 3.3 **Intervalles de confiance. Chapitre 3, paragraphe 3.4.5**

Une façon usuelle de quantifier la précision associée à une estimation consiste à définir un intervalle dans lequel on espère trouver la valeur réelle ainsi que la probabilité que la valeur réelle soit dans cet intervalle. Un *intervalle de confiance à x % pour un paramètre p* est un intervalle qui avec une probabilité de x % contient p. Le but fixé est de déterminer des intervalles de confiance pour l'erreur réelle $p = e(h)$. Tout d'abord, nous connaissons la loi binomiale qui gouverne l'erreur estimée r/m et donc l'espérance de cette loi (p) et l'écart-type (voir exercice précédent). Par conséquent, pour trouver un intervalle de confiance à 95 % de l'erreur réelle, il suffit de trouver un intervalle de centre p qui contient r/m avec une probabilité supérieure ou égale à 95 % ou de façon équivalente un intervalle de centre r/n qui contient p avec une probabilité supérieure ou égale à 95 %. Pour une valeur donnée de x, comment calculer la taille d'un tel intervalle ? Les calculs étant trop difficiles pour une loi binomiale, nous allons utiliser des approximations par une loi normale.
La loi normale est une loi continue. La loi normale de paramètres μ et σ est définie par la densité de probabilité :

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2}$$

L'espérance d'une loi normale de paramètres μ et σ est μ et son écart-type est σ. Tracer la représentation graphique de la fonction de densité de la loi normale de paramètres 0 et 1. Le théorème central limite permet d'affirmer que, pour n assez grand, on peut approximer une loi binomiale par une loi normale de même espérance et de même écart-type.
Pour une loi normale, il existe des tables permettant de calculer le rayon des intervalles de confiance. En effet, si X suit une loi normale d'espérance μ et d'écart-type σ, alors la valeur mesurée x de X appartient avec une probabilité de x % à l'intervalle $[m - \zeta_x\sigma, m + \zeta_x\sigma]$ ou de façon équivalente, m appartient avec une probabilité de x % à l'intervalle $[x - \zeta_x\sigma, x + \zeta_x\sigma]$ où ζ_x peut être déterminé à l'aide des tables suivantes :

x %	50	68	80	90	95	98	99
ζ_x	0,67	1	1,28	1,64	1,96	2,33	2,58

En conclusion, en utilisant l'approximation faite dans le calcul de l'écart-type et en supposant m assez grand pour admettre l'approximation de la loi binomiale par une loi normale, nous obtenons le résultat suivant : avec une probabilité de x %, l'erreur réelle de h appartient à l'intervalle

$$[\, r/m - \zeta_x\sqrt{\frac{r/m \times (1 - r/m)}{n}}, r/m + \zeta_x\sqrt{\frac{r/m \times (1 - r/m)}{m}}\,]$$

Nous allons utiliser ces résultats pour les questions suivantes :

1. Déterminez les intervalles de confiance dans les cas suivants :
   - $m = 100$, $r = 25$ et $m = 90$;
   - $m = 100$, $r = 25$ et $m = 95$;
   - $m = 200$, $r = 48$ et $m = 95$;
   - $m = 500$, $r = 130$ et $m = 95$.
2. On sait que l'erreur est inférieure à 40 %. On souhaite déterminer une approximation de l'erreur réelle à 0.05 près avec une confiance de 95 %. Combien d'exemples faut-il tirer ? Même question avec une confiance de 98 %.
3. Dans certains cas, seule une borne supérieure sur l'erreur réelle est souhaitée. Déterminer l'intervalle de confiance pour $m = 100$, $r = 25$ et $m = 90$; quelle est la confiance pour l'intervalle $[0, b]$ où b est la borne supérieure de l'intervalle trouvé précédemment ? Déterminez les intervalles de confiance de la forme $[0, b]$ dans les cas suivants :
   - $m = 100$, $r = 25$ et $m = 90$;

- $m = 100$, $r = 25$ et $m = 99$;
- $m = 200$, $r = 48$ et $m = 95$;
- $m = 500$, $r = 130$ et $m = 97,5$.

Nous avons donc démontré que la loi qui gouverne le problème de l'estimation de l'erreur sur un ensemble test suit une loi binomiale. Nous pouvons alors calculer des intervalles de confiance pour l'erreur réelle. Pour cela, nous avons effectué les deux approximations suivantes :

- Remplacer l'erreur réelle par l'erreur estimée dans le calcul de l'écart-type,
- Approximer la loi binomiale par une loi normale. Cette approximation n'a de sens que lorsque : $n \geq 30$ et $np(1-p) \geq 5$.

FDRG

Exercice 3.4  Que peut-on dire ?

1. On réalise une expérience dont le résultat dépend d'une épreuve aléatoire. On observe les données suivantes :
   0,0,1,2,0,0,0,0,1,3,13,0,0,0,2,0,0,0,0,1.
   Que peut-on dire ?
2. On poursuit l'expérience, on observe les données suivantes :
   2,0,0,1,0,0,1,3,40,0,1,1,0,0,0,0,0,0,1,1.
   Que peut-on dire ?
3. Sachant que $P(n) = \frac{6}{\pi^2} \times \frac{1}{(n+1)^2}$ et que $X(n) = n + 1$, déterminer l'espérance mathématique de X.

FDRG

Exercice 3.5  Comparaison de classificateurs

On souhaite dans cet exercice comparer des hypothèses générées par différents systèmes d'apprentissage pour un même problème. La table permettant de calculer, à l'aide de la loi normale, les intervalles de confiance est :

x %	50	68	80	81	82	90	95	98	99
ζ_x	0,67	1	1,28	1,31	1,34	1,64	1,96	2,33	2,58

1. Pour un problème d'apprentissage, par un certain algorithme, on a généré une hypothèse h_1. On estime l'erreur réelle pour cette hypothèse à l'aide d'un échantillon test $\mathcal{T}_1$ de taille $m_1 = 200$. Cette hypothèse fait $r_1 = 60$ erreurs sur l'échantillon test. Déterminer un intervalle de confiance à 95 % centré autour de l'erreur estimée. Déterminer un intervalle de confiance à 95 % de la forme $[0, b]$.
2. Pour le même problème d'apprentissage, à l'aide d'un autre algorithme, on a généré une hypothèse h_2. On estime l'erreur réelle pour cette hypothèse à l'aide d'un échantillon test $\mathcal{T}_2$ de taille $m_2 = 500$. Cette hypothèse fait $r_2 = 125$ erreurs sur l'échantillon test. Déterminer un intervalle de confiance à 95 % centré autour de l'erreur estimée. Déterminer un intervalle de confiance à 95 % de la forme $[0, b]$.
3. On souhaite comparer les hypothèses h_1 et h_2. On appelle p_1 l'erreur réelle de h_1 et $\hat{p}_1$ l'erreur estimée de h_1 sur $\mathcal{T}_1$. De même, on appelle p_2 l'erreur réelle de h_2 et $\hat{p}_2$ l'erreur estimée de h_2 sur $\mathcal{T}_2$. On considère alors la quantité $d = p_1 - p_2$ et son estimateur $\hat{d} = \hat{p}_1 - \hat{p}_2$. On suppose que les échantillons $\mathcal{T}_1$ et $\mathcal{T}_2$ sont indépendants. On peut alors montrer que $\hat{d}$ est un estimateur non biaisé de d (i.e. son espérance est d) et que $\hat{d}$ peut être approximée, sous les hypothèses usuelles, par une loi normale de variance la somme des variances de $\hat{p}_1$ et $\hat{p}_2$. On rappelle que la variance pour $\hat{p}_1$ peut être approximée par $\frac{r_1/n_1(1-r_1/n_1)}{n_1}$. Calculer une approximation de la variance et de l'écart type de $\hat{d}$. Calculer un intervalle de confiance à 95 % centré autour de $\hat{d}$ pour la différence d entre les erreurs réelles.

4. Déterminer la probabilité que h_1 soit meilleure que h_2, c'est à dire la probabilité que $d > 0$.

FDRG

Quelques sites internet recommandés

Classification Toolbox for Matlab
`http://tiger.technion.ac.il/~eladyt/classification/index.htm`

Chapitre 4

Induction et relation d'ordre : l'espace des versions

Exercice 4.1  **Les rectangles, le retour**

Reprendre l'exemple du chapitre 4, paragraphe 4.3.1.1. Quels sont les autres concepts « coupés à ras » du second exemple, c'est à dire les plus généraux compatibles avec les deux premiers exemples ?

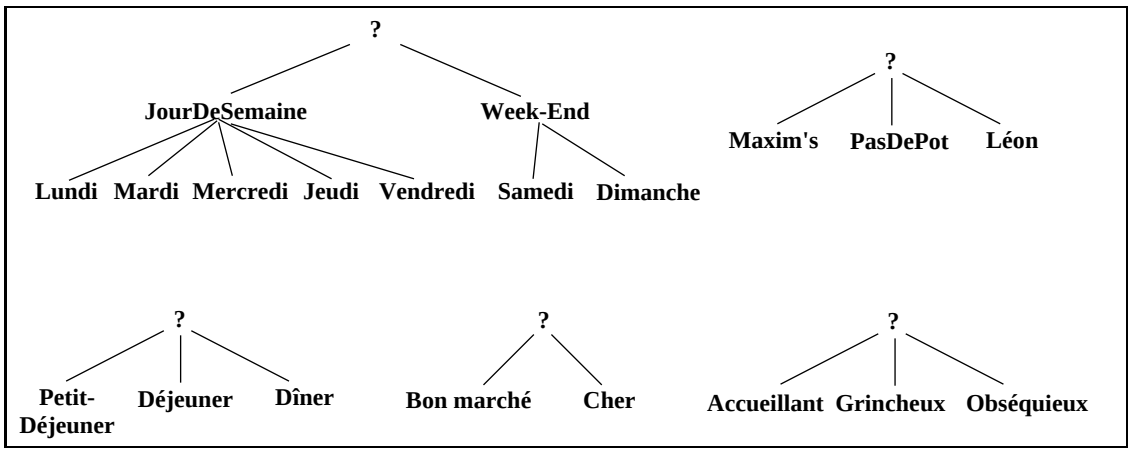
ACLM

Exercice 4.2  **Variation**

Soient des exemples correspondants à des repas pris dans le passé. Ils sont décrits par les attributs :

NomRestaurant	valeurs possibles : Léon, Maxim's, PasDePot
Repas	valeurs possibles : PetitDéjeuner, Déjeuner, Dîner
Jour	valeurs possibles : Lundi ... Dimanche
Catégorie	valeurs possibles : BonMarché, Cher
Service	valeurs possibles : Accueillant, Grincheux, Obséquieux

Ces attributs sont organisés dans des hiérarchies figurées dans la figure 4.2.



Soit l'espace des hypothèses décrites par des conjonctions de contraintes sur les attributs. Les contraintes peuvent prendre la forme :

– « ? » : toute valeur est possible

– « » : aucune valeur n'est acceptable
– une valeur spécifique (ex: *Eau* = chaude).

On cherche à trouver l'ensemble des hypothèse cohérentes avec les données d'apprentissage (l'espace des versions), c'est-à-dire telles qu'elles couvrent tous les exemples positifs et aucun exemple négatif.
Soit la séquence d'exemples suivante :

$\boldsymbol{x}_1$	Léon	PetitDéjeuner	Samedi	Bon Marché	Accueillant	+
$\boldsymbol{x}_2$	Maxim's	Déjeuner	Samedi	Cher	Accueillant	-
$\boldsymbol{x}_3$	Léon	Déjeuner	Dimanche	Bon Marché	Accueillant	+
$\boldsymbol{x}_4$	PasDePot	PetitDéjeuner	Lundi	Cher	Grincheux	-
$\boldsymbol{x}_5$	Léon	PetitDéjeuner	Dimanche	Cher	Accueillant	+

1. Expliquez pourquoi la taille de l'espace des hypothèses est de 1537. Quel est le nombre d'exemples possibles et d'hypothèses possibles si l'on ajoute l'attribut *Confort* avec les valeurs *faible*, *modéré*, *excellent*? Plus généralement, comment croissent le nombre d'exemples possibles et d'hypothèses quand on ajoute un attribut A pouvant prendre une valeur parmi k?
2. Spécifiez S et G après la prise en compte de chaque exemple par l'algorithme d'élimination des candidats.
3. Spécifiez S et G après la prise en compte de chaque exemple de la séquence prise, dans l'ordre inverse, par l'algorithme d'élimination des candidats. Bien que l'espace des versions final soit le même (expliquez pourquoi), les ensembles S et G vont, bien sûr, dépendre de cet ordre. Pouvez-vous en tirer des idées sur l'optimisation de la séquence d'apprentissage en vue de minimiser la taille de S et G?
4. Supposons maintenant que l'on considère un nouvel espace d'hypothèse $\mathcal{H}$ incluant les expressions disjonctives à deux termes, par exemple : « ?, Déjeuner, Samedi,?,?,? » $\vee$ « Maxim's,?, Mardi,?,?, obséquieux ». Calculez alors le S-set et le G-set pour la séquence donnée plus haut.
5. Supposons que l'algorithme d'apprentissage par élimination des candidats effectue un apprentissage actif, c'est-à-dire puisse choisir l'ordre des exemples à prendre en compte parmi un ensemble d'apprentissage donné, ou bien même soit capable de poser de proposer des exemples à étiqueter par l'oracle (un peu comme au Mastermind). Quelle serait selon vous une bonne stratégie de choix des exemples pour accélérer la convergence de l'apprentissage?

ACLM

Exercice 4.3 Les voitures japonaises

Faire tourner l'algorithme de l'élimination des candidats sur les exemples suivants :

Origine	Fabricant	Couleur	Decade	Type	
Japon	Honda	Bleu	1980	Economique	+
Japon	Toyota	Vert	1970	Sportive	-
Japon	Toyota	Blue	1990	Economique	+
USA	Chrysler	Rouge	1980	Economique	-
Japon	Honda	Blanc	1980	Economique	+

ACLM, d'après [Rich and Knight, 1997]

Exercice 4.4 Quelques exemples

Faire tourner l'algorithme de l'élimination des candidats sur les trois problèmes suivants (quelques connaissances en anglais sont requises). "pos" signale un exemple positif, "neg" un exemple négatif. Les attributs sont hiérarchiques (à vous de proposer les hiérarchies !).

card-attributes

pos	three	heart
neg	two	spade
pos	four	diamond
neg	jack	heart

apple-attributes-1

pos	black	small	bitter	very-few
neg	black	small	sour	very-few
pos	normal	normal	bitter	very-many

apple-attributes-2

pos	yellow	small	bitter	very-few
neg	yellow	normal	sour	very-many
pos	black	small	sweet	rather-few
neg	normal	normal	sweet	rather-few
neg	yellow	small	sour	rather-many
neg	normal	small	bitter	very-few
neg	black	normal	sour	rather-few

ACLM, d'après http://www.cs.indiana.edu/classes/b551-gass/hw4.html

Chapitre 5

La programmation logique inductive

Des exercices sur ce chapitre seront mis sur le site à la fin de l'année 2002.

Quelques sites internet recommandés

`http://www.`

Chapitre 6

Reformulation et transfert de connaissances

Chapitre 7

L'inférence grammaticale

Inférence de grammaires régulières

Exercice 7.1  **L'espace de recherche (1)**

Construire l'ensemble des automates pour lesquels l'échantillon $I^+ = \{aaa\}$ est structurellement complet.
Quel est l'ensemble frontière délimité par $I^- = \{a\}$?

ACLM

Exercice 7.2  **L'espace de recherche (2)**

Construire l'automate canonique de l'échantillon $I^- = \{aab, b\}$. Fusionner les deux états terminaux pour obtenir l'automate automate $\mathcal{A}$. Construire le treillis des automates obtenus par fusion d'états à partir de $\mathcal{A}$. Quel est le langage accepté par chacun d'entre eux?
Quel est l'ensemble frontière délimité par $I^- = \{abb\}$?

Exercice 7.3  **RPNI (1)**

Détailler les opérations faites par l'algorithme *RPNI* sur l'exemple du chapitre 7, paragraphe 7.5.4.2 après la fusion des états 0 et 2.

Exercice 7.4  **RPNI (2)**

Appliquer l'algorithme RPNI sur l'échantillon suivant :
$I^+ = \{aaa, ab, bbb, bbbab\}$, $I^- = \{ab, aa, bb\}$
Quel est le langage accepté par l'automate inféré?

CHJO

Exercice 7.5  **L'algorithme « blue fringe »**

FC

Exercice 7.6  **Un échantillon caractéristique.**

Un échantillon est dit *caractéristique* pour un langage et pour un algorithme d'inférence régulière si l'algorithme passé sur cet échantillon trouve l'automate canonique du langage.
Oncina et García ([Oncina and García, 1992]) ont démontré des conditions suffisantes pour que l'échantillon (I_+, I_-) soit caractéristique pour un langage L et pour l'algorithme $RPNI$.
Soit l'automate déterministe minimal $(\mathcal{A} = \Sigma, Q, \delta, q_0, F)$ et L le langage qu'il accepte. On a donc :

$$x \in L \iff \delta(q_0, x) \in F$$

On appelle les *préfixes* de L l'ensemble de phrases suivant :

$$Pr(L) = \{u \mid \exists x \in \Sigma^* : ux \in L\}$$

On définit les *préfixes courts* de L par :

$$Sp(L) = \{w \mid w \in Pr(L) et \nexists v : \big(\delta(q_0, v) = \delta(q_0, w)) \wedge (\mid v \mid < \mid w \mid \big)\}$$

On appelle *noyau* de L l'ensemble des phrases suivantes :

$$N(L) = \{ua \mid ua \in Pr(L) \wedge u \in Sp(L)\} \cup \lambda$$

Pour que l'échantillon (I_+, I_-) soit caractéristique pour le langage L et pour l'algorithme $RPNI$, il suffit que les deux conditions suivantes soient remplies :

$$\forall x \in N(L), \exists xu \in I_+ \text{ avec } (u = \lambda \text{ si } x \in L)$$

et

$$\forall x, y \in N(L) : \delta(q_0, x) \neq \delta(q_0, y), \exists u : (xu \in I_+ \wedge yu \in I_-) \vee (xu \in I_- \wedge yu \in I_+)$$

Questions

1. Construire l'automate à trois états sur l'alphabet $\Sigma = \{a, b\}$ qui reconnait les phrases dont le nombre de a moins le nombre de b est divisible par trois. Montrer qu'il est impossible de le minimiser.
2. Trouver un échantillon caractéristique pour ce langage et pour l'algorithme $RPNI$. Vérifier en applicant l'algorithme sur l'échantillon trouvé.
3. Avez-vous désormais une idée de la démonstration de Oncina et García?

ACLM, CHJO

Grammaires régulières stochastiques

Exercice 7.7  **Distribution de probabilités sur des séquences**

Soit k modèles probabilistes $M_1, ...M_k$, qui affectent des probabilités à des chaînes : le modèle M_i affecte la probabilité $P_{M_i}(x)$ à la chaîne x.
On veut construire une interpolation linéaire des modèles M_i : pour chaque chaîne x on affecte la probabilité

$$P_{interp}(x) = \alpha_1 P_{M_1}(x) + ...\alpha_k P_{M_k}(x) = \sum_{i=1}^{k} \alpha_i P_{M_i}(x)$$

Supposons que l'on dispose d'un ensemble E de chaînes. Exprimez les étapes d'estimation de maximisation de l'algorithme EM pour estimer au mieux (étant donné E) les valeurs des α_i ?

FT

Exercice 7.8  **Automates probabilistes**

Montrer par construction qu'un Modèle de Markov caché (Chapitre 13) dont l'alphabet des observations est discret peut être transformé en un automate probabiliste non déterministe.

FT

Protocoles d'apprentissage

Exercice 7.9  **Identification à la limite et identification par données exactes**

L'objectif de cet exercice est de montrer l'équivalence entre deux cadres d'apprentissage : l'identification à la limite et l'identification par données fixées. Il peut être abordé après l'étude du paragraphe 7.2, et plus particulièrement du 7.2.2. Nous donnons dans un premier temps les définitions des deux cadres, puis nous posons le problème avant d'en demander la solution.

Définitions préliminaires Soit $\mathcal{C}$ une classe de langages et $\mathcal{R}$ une classe de représentations (grammaires) des langages de $\mathcal{L}$.
Rappelons tout d'abord la définition d'un premier critère de convergence, celui d'identification à la limite [Gold, 1967], qui nécessite celle de présentation complète d'un langage.

Définition 7.1 (Présentation complète d'un langage)

Soit L un langage de $\mathcal{C}$.

- *On appelle présentation de L toute séquence infinie P de paires $(x,d) \in \Sigma^\star \times \{+,-\}$ telle que $(x,+) \in P$ ssi $x \in L$, et $(x,-) \in P$ ssi $x \notin L$.*
- *On dit qu'une présentation P est complète ssi tous les mots de $\Sigma^\star$ apparaissent au moins une fois comme premier argument d'une paire de P.*

On notera P_k la séquence des k premiers éléments d'une présentation P.

Définition 7.2 (Identification à la limite)

On dit que $\mathcal{C}$ est identifiable à la limite ssi il existe un algorithme A tel que pour tout langage $L \in \mathcal{C}$ et pour toute présentation complète P de L, il existe un rang n tel que pour tout $k \geq n$, $A(P_k)$ retourne une représentation $r \in \mathcal{R}$ de L.

Intuitivement, l'identification à la limite simule un processus d'apprentissage incrémental, où un algorithme reçoit des données (des exemples et des contre-exemples) les unes à la suite des autres. Ce flot, supposé infini, est modélisé par la notion de présentation complète. A chaque nouvelle donnée reçue, l'algorithme propose un langage. Clairement, tant qu'il n'a pas reçu suffisamment de données, il peut répondre n'importe quoi. Mais il existe un moment où l'algorithme a reçu suffisamment d'information pour trouver effectivement le langage dont ces données sont issues. Et à partir de ce moment là, il ne changera plus jamais d'avis sur toutes les nouvelles données qu'il recevra.
Voyons maintenant un second critère de convergence, celui d'identification par données fixées [Gold, 1978, de la Higuera, 1997]. Ce critère modélise une situation d'apprentissage non incrémentale, où un nombre fini d'exemples et de contre-exemples est immédiatement disponibles. Bien entendu, l'information contenues dans ces données doit être suffisante pour apprendre le langage cible. Pour traduire cette notion de suffisance, on suppose que chaque langage est *caractérisé* par certains ensembles clefs d'exemples et de

contre-exemples, qu'on appelle des ensembles caractéristiques. Un algorithme doit être en mesure d'identifier le langage dès qu'un de ces ensembles caractéristiques est contenu dans les données. Et sinon, on lui demandera de retourner une représentation proche des données.

Définition 7.3 (Identification par données fixées)

On dit que $\mathcal{C}$ est identifiable par données fixées ssi il existe un algorithme B tel que pour tout langage $L \in \mathcal{C}$,

1. *pour toute paire $\langle E+, E-\rangle$ d'ensembles finis telle que $E+$ contient des mots de L (exemples) et $E-$ contient des mots n'appartenant pas à L (contre-exemples), $B(E+, E-)$ retourne une grammaire de $\mathcal{R}$ qui est cohérente avec $E+$ et $E-$ (i.e., les mots de $E+$ sont acceptés par cette grammaire, et ceux de $E-$ sont rejetés) ;*
2. *il existe une paire d'ensembles finis $\langle EC+, EC-\rangle$, appelés ensembles caractéristiques de L, tels que pour toute paire d'ensembles finis $\langle E+, E-\rangle$, si $EC+ \subseteq E+$ et $EC- \subseteq E-$, alors $B(E+, E-)$ retourne une représentation de L.*

Question Montrer que $\mathcal{C}$ est identifiable à la limite si et seulement si $\mathcal{C}$ est identifiable par données fixées.

JCJ & FT

Chapitre 8

Apprentissage par évolution simulée

Exercice 8.1  **Comment régler les paramètres de l'algorithme génétique ?**

Le taux de mutation p_{mut} et le taux de croisement p_{crois} sont les seuls paramètres à régler pour un algorithme génétique à population fixe et à sélection proportionnelle à la performance.
Peut-on envisager d'*apprendre* ces paramètres ? Sous quel protocole ? Avec quel type d'algorithme ? Décrire un « méta » algorithme génétique qui le fasse. Comment régler ses paramètres ?

ACLM

Exercice 8.2  **Un algorithme génétique de comptoir.**

Supposons que la fonction f de performance choisie soit la valeur décimale de la chaîne de bits. Par exemple $f(1001) = 9.0$.
Simuler en utilisant un dé à six faces le comportement d'un algorithme génétique à population fixe et à sélection proportionnelle à la performance pour une population de chaînes de bits de longueur $L = 4$, de taille $P_{\mathcal{G}}$, avec un taux de mutation p_{mut} de 16.6 % et un taux de croisement p_{crois} de 50 %. Faire au moins deux essais.

ACLM

Chapitre 9

L'apprentissage de surfaces séparatrices linéaires

Exercice 9.1  Comparaison de deux algorithmes

Comparer les critères que minimisent les méthodes de Ho-Kashyap et du perceptron, en supposant qu'elles convergent.

ACLM

Exercice 9.2  Et pour plus de deux classes ?

Au chapitre 9, paragraphe 9.2.6, il a été abordé la question de la séparation linéaire de C classes, avec $C > 2$.

La première méthode consiste à calculer C hyperplans, chacun séparant les exemples d'une classe de ses contre-exemples constitués de tous les autres exemples.

Questions

1. Dessiner un exemple à quatre classes et deux dimensions. Caractériser les zones délimitées par les droites séparatrices en terme de décision d'appartenance aux classes.
2. Que peut-on dire de la probabilité *a priori* des classes si elle est estimée par l'ensemble d'apprentissage?

La seconde méthode consiste à calculer $C(C-1)/2$ hyperplans chacun séparant les exemples d'une classe de ses contre-exemples constitués des exemples d'une autre classe.

Question

1. Dessiner un exemple à quatre classes et deux dimensions. Caractériser les zones délimitées par les droites séparatrices en terme de décision d'appartenance aux classes.

ACLM

Exercice 9.3  Le perceptron : apprentissage du OU

Les données d'apprentissage $\mathcal{S}$ ont deux dimensions $\boldsymbol{x}_1$ et $\boldsymbol{x}_2$. On leur rajoute une première composante $\boldsymbol{x}_0$ fixée à 1. On cherche à apprendre le vecteur $\boldsymbol{a}$ à partir de $\mathcal{S}$. La première composante du vecteur $\boldsymbol{a}$ correspond à l'attribut $\boldsymbol{x}_0 = 1$, les deux composantes suivantes correspondent aux attributs $\boldsymbol{x}_1$ et $\boldsymbol{x}_2$.

On suppose qu'à l'initialisation, les poids suivants ont été choisis : $\boldsymbol{a}_0 = 0$; $\boldsymbol{a}_1 = 1$ et $\boldsymbol{a}_2 = -1$. On suppose que les exemples sont présentés dans l'ordre lexicographique 100, 101, 110, 111. La sortie désirée vaut u, celle calculée par le perceptron vaut y.

étape	a_0	a_1	a_2	Entrée	$\Sigma_0^2 a_i x_i$	y	u	a_0	a_1	a_2
								0	1	-1
1	0	1	-1	100	0	0	0	0+0×1	1+0×0	-1+0×0
2	0	1	-1	101	-1	0	1	0+1×1	1+1× 0	-1+1×1
3	1	1	0	110	2	1	1	1	1	0
4	1	1	0	111	2	1	1	1	1	0
5	1	1	0	100	1	1	0	1+(-1)×1	1+(-1)×0	0+(-1)×0
6	0	1	0	101	0	0	1	0+1×1	1+1×0	0+1×1
7	1	1	1	110	2	1	1	1	1	1
8	1	1	1	111	3	1	1	1	1	1
9	1	1	1	100	1	1	0	1+(-1)×1	1+(-1)×0	1 +(-1)×0
10	0	1	1	101	1	1	1	0	1	1

Aucune entrée ne modifie le perceptron à partir de cette étape. Pourquoi peut-on dire que ce perceptron calcule le *OU* logique sur les entrées $\boldsymbol{x}_1$ et $\boldsymbol{x}_2$?

Question

Simuler l'algorithme d'apprentissage du perceptron (fonction seuil en sortie) de la fonction booléenne *OU* pour d'autres valeurs d'initialisation du vecteur de poids. Prendre par exemple les valeurs suivantes :

- $a_0 = 0\,;\ \boldsymbol{a}^T = (0,0)$
- $a_0 = 1\,;\ \boldsymbol{a}^T = (1,1)$
- $a_0 = 1\,;\ \boldsymbol{a}^T = (-1,1)$
- $a_0 = 0.5\,;\ \boldsymbol{a}^T = (1,1)$

FDRG

Exercice 9.4 — Le perceptron : apprentissage du *ET*

Même exercice avec la fonction *ET*.

FDRG

Exercice 9.5 — Le perceptron : apprentissage du *XOR*

Même exercice avec la fonction *XOR*. Faites les remarques qui s'imposent !

FDRG

Exercice 9.6 — Le perceptron

Reprendre l'exemple de l'exercice 9.3 avec le même échantillon d'entrée $\mathcal{S}$ et les valeurs initiales : $a_0 = 0\,;$ $\boldsymbol{a}^T = (1,0)$.
Simuler ensuite l'apprentissage en considérant l'échantillon $\mathcal{S} \cup \{((3,1),0)\}$.

FDRG

Exercice 9.7 — Algorithme de Widrow et Hoff (règle Δ)

L'algorithme d'apprentissage du perceptron par la fonction Δ donné au chapitre 10, paragraphe 10.3.1.1 (fonction identité en sortie) est aussi appelé en hommage à ses auteurs l'algorithme de Widrow et Hoff. On peut aussi simplement l'appeller « algorithme du gradient ».
Tester cet algorithme sur les données de l'exercice 9.3. Comparer le nombre d'itérations nécessaires pour différentes valeurs du paramètre α.

FDRG

Exercice 9.8 ☞ Algorithme du gradient

Implémenter l'algorithme du gradient pour une cellule à 2 entrées. Choisir une fonction linéaire de deux variables et un échantillon d'apprentissage. Comparer avec l'algorithme du perceptron, en particulier du point de vue de la vitesse de convergence.

FDRG

Exercice 9.9 ☞ Perceptron

Choisir une fonction linéaire de deux variables et un échantillon d'apprentissage de 100 exemples. Apporter du bruit de classification en changeant la classe de 10 exemples. Étudier le comportement de l'algorithme du gradient et de l'algorithme du perceptron.

FDRG

Exercice 9.10 ☞☞ Le pouvoir de séparation du perceptron

Une classe ω désigne un sous-ensemble de $\{0,1\}^d$, $\boldsymbol{x}$ un vecteur binaire de $\{0,1\}^d$. On dit qu'une classe ω est *reconnaissable par perceptron* s'il existe un perceptron $(a_0, \boldsymbol{a})$ tel que

$$a_0 + \boldsymbol{a}^T \boldsymbol{x} = \begin{cases} 1 & \text{si } \boldsymbol{x} \in \omega \\ 0 & \text{sinon} \end{cases}$$

1. Démontrer que les classes ω suivantes sont reconnaissables par perceptron.
   (a) ω est l'ensemble des vecteurs dont *au moins* m entrées sont à 1.
   (b) ω est l'ensemble des vecteurs dont *au plus* m entrées sont à 1.
   (c) ω est l'ensemble des vecteurs tels qu'il y ait plus d'entrées à 1 dans les coordonnées de rang élévé ($i > \frac{d}{2}$) que dans les coordonnées de rang bas ($i > \frac{d}{2}$).
2. Une méthode pour montrer que deux classes ne sont pas linéairement séparables est la suivante: trouver p exemples $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p$ de la première et p exemples $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_p$ de la seconde tels que $\sum_i \boldsymbol{x}_i = \sum_i \boldsymbol{y}_i$. Montrer que cette méthode est correcte. Doit-on supposer que les exemples choisis dans chacune des deux classes sont distincts? Pourquoi?
3. Montrer que les classes suivantes ne sont pas reconnaissables par perceptron.

   (a) C est l'ensemble des vecteurs tels que les entrées à 1 (s'il en existe) sont contigües. On suppose que $d \geq 4$. C est donc la classe des figures connexes.
   (b) C est l'ensemble des vecteurs symétriques par rapport au centre de la rétine (on suppose que $d > 2$).
   (c) C est l'ensemble des vecteurs dont m entrées exactement sont égales à 1 (avec $1 \leq m < d$).

FDRG

Exercice 9.11 ☞☞ Une impasse de l'évolution

Rosenblatt, Minski et Papert ([])ont étudié un modèle de perceptron un peu plus compliqué que celui qui est présenté dans le livre. Ces auteurs supposent qu'entre la « rétine » (le vecteur des entrées) et la cellule de « décision » (de sortie) se trouvent un certain nombre de cellules d'*association*. Ces cellules intermédiaires effectuent un traitement préliminaire sur certaines cellules de la rétine et transmettent le résultat de ce traitement à la cellule de décision. Les sorties des cellules d'association constituent une nouvelle rétine qui représentent les entrées de la cellule de décision.
Les cellules d'association peuvent calculer *a priori* n'importe quelle fonction booléenne. Mais si l'on ne restreint pas les traitements qu'elles peuvent effectuer, ce modèle du perceptron perd tout son intérêt puisque justement toute fonction booléenne est calculable.

Question
Relier l'argumentation précédente avec le théorème du « vilain petit canard », page 540 et plus généralement avec la nécessité de biais d'apprentissage (Chapitre 2).
Une manière naturelle de restreindre les cellules d'association est de considérer qu'elles ne peuvent dépendre que d'un petit nombre de cellules de la rétine. On peut par exemple supposer que les cellules d'association ne peuvent dépendre que d'au plus k cellules de la rétine (perceptron à "domaine borné") ou, dans un contexte géométrique, qu'elles ne peuvent dépendre que de cellules de la rétine au plus distantes de k (perceptron à "diamètre limité"). Plus précisément, dans le cas d'une rétine rectangulaire, on définira la distance de deux cellules définies par leurs numéros de lignes et de colonnes par $d((l,c),(l',c')) = |l-l'| + |c-c'|$.

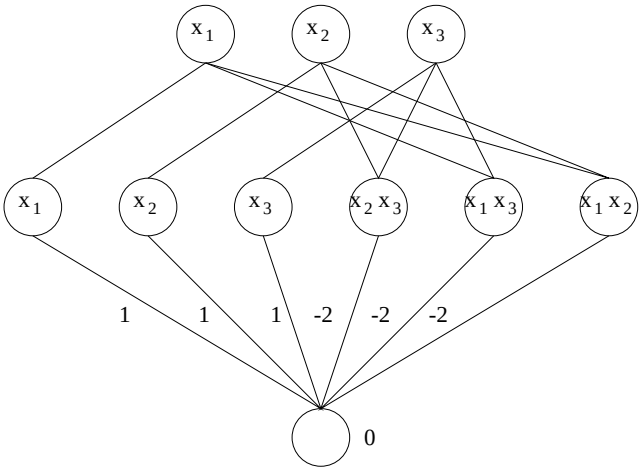


FIG. 9.1 – *Un perceptron à domaine borné (k=2) qui reconnaît si une cellule et une seule est active*

1.
   (a) Montrer qu'un perceptron à domaine borné (avec $k = 2$) peut reconnaître des figures symétriques par rapport au centre de la rétine.
   (b) Montrer qu'un perceptron à diamètre limité (avec $k = 1$) peut reconnaître des figures connexes.
   (c) Montrer qu'un perceptron à domaine borné (avec $k = 2$) peut reconnaître les entrées possédant exactement d cellules actives. On pourra s'inspirer de l'exemple de la figure 9.1.

   Cette extension semble intéressante puisque des fonctions « naturelles » qui ne sont pas calculables dans le modèle de base le deviennent avec cette variante. Malheureusement, on peut montrer que le gain n'est pas aussi important que les résultats précédents pourraient le laisser espérer.
2. Montrer qu'aucun perceptron à diamètre limité ne peut reconnaître les figures connexes (c'est-à-dire dont les entrées à 1 forment un seul morceau) sur une rétine rectangulaire.
   Indication : Supposer qu'un perceptron à diamètre limité k peut reconnaître les figures connexes et considèrer, sur une rétine rectangulaire de dimension au moins $5(k+2)$, les quatre formes de la figure 9.2, où les entrées à 1 sont figurées en noir :

FDRG

Exercice 9.12  **Une application à la reconnaissance des images**

- Montrer qu'il existe un perceptron qui différencie les chiffres pairs des chiffres impairs lorsqu'ils sont écrits sur une rétine à 7 leds (voir l'exercice 10.13).
- On veut apprendre à distinguer la classe des chiffres représentés par un système de 7 leds (allumés ou éteints) de la classe des non-chiffres. Est-ce possible avec un perceptron ?

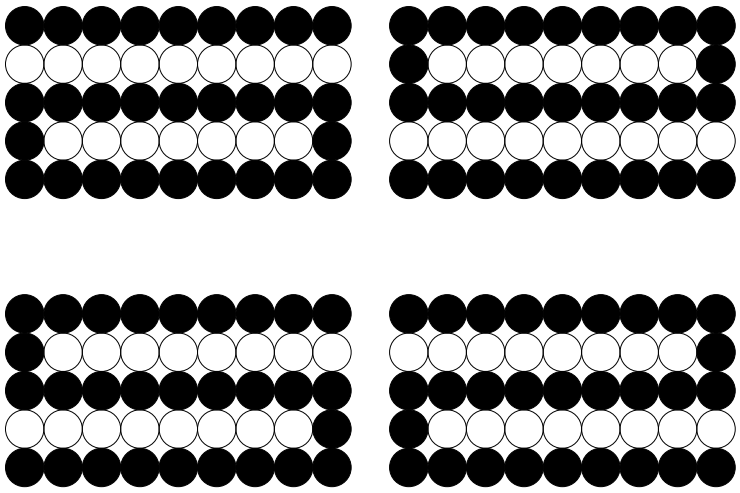


FIG. 9.2 – *Aucun perceptron à diamètre limité ne peut apprendre la connexité*

FDRG

Exercice 9.13 **Un séparateur à vaste marge**

Soit l'ensemble d'apprentissage suivant dans $\mathbb{R}^2$:

$$\mathcal{S} = \left\{ \left(\begin{pmatrix} 1 \\ 1 \end{pmatrix}, + \right), \left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, - \right), \left(\begin{pmatrix} 2 \\ 2 \end{pmatrix}, + \right), \left(\begin{pmatrix} 2 \\ 0 \end{pmatrix}, + \right), \left(\begin{pmatrix} 0 \\ 1 \end{pmatrix}, - \right) \right\}$$

1. Construire l'hyperplan optimal et calculer la marge optimale correspondante
2. Quels sont les vecteurs support ?
3. Construire la solution dans l'espace dual en calculant les multiplicateurs de Laagrange. Comparer les résultats.

ACLM

Quelques sites internet recommandés

Classification Toolbox for Matlab
`http://tiger.technion.ac.il/~eladyt/classification/index.htm`

Chapitre 10

L'apprentissage de réseaux connexionnistes

Exercice 10.1 Calculs élémentaires

Un neurone i reçoit des entrées de quatre autres neurones dont les signaux de sortie respectifs sont : 10, -20, 4 et -2. Les poids synaptiques associés à ces sorties sont respectivement : 0.8, 0.2, -1.0 et -0.9. Calculer la sortie du neurone i dans les situations suivantes (on supposera que le biais est nul) :

1. Le neurone i est linéaire.
2. Le neurone i a une fonction d'activation à seuil :

$$y_i = \begin{cases} 1 \text{ si } a_i \geq 0 \\ 0 \text{ sinon} \end{cases}$$

   où a_i dénote l'activation du neurone i (la somme pondéré de ses entrées).
3. Le neurone i est associé à une fonction d'activation logistique :

$$y_i = \frac{1}{1 + \exp(-a_i)}$$

ACLM

Exercice 10.2 Linéaire ou non linéaire ?

Soit un perceptron multicouche dont tous les neurones opèrent dans la région linéraire de la fonction logistique. Justifier l'affirmation qu'un tel réseau est équivalent à un perceptron à une couche.

ACLM

Exercice 10.3 Une autre règle d'apprentissage

Plusieurs règles d'apprentissage ont été proposées dans la littérature sur les réseaux connexionnistes, dont parmi les plus célèbres : la règle delta ou encore règle de Widrow-Hoff vue dans le chapitre **??**, et la règle dite de Hebb. Nous les rappelons ci-dessous.
Règle de Widrow-Hoff.

$$\Delta_{ij} = \eta y_i e_j$$

où y_i est la sortie du neurone amont i et e_j l'erreur au niveau du neurone aval j.

Règle de Hebb. Elle consiste à renforcer les poids des connexions entre deux neurones qui sont simultanément actifs pour une tâche donnée :

$$\Delta_{ij} = \eta y_i a_j$$

où a_j est l'activité du neurone aval j.
Faire une liste de ce qui rend ces deux règles différentes.

ACLM

Exercice 10.4 Apprentissage de la fonction parité

La *fonction parité* sur d variables booléenes $x_1, \ldots, x_d$ est définie comme suit : elle vaut 1 si le nombre d'entrées à 1 est pair.

1. La fonction parité est-elle calculable par un perceptron linéaire à seuil (on prendra les entrées comme un vecteur de $\mathbb{R}^d$, chacune valant 0. ou 1.)?
2. On considère maintenant un réseau connectionniste multicouches (RCM) où le neurone formel élémentaire est un perceptron linéaire à seuil. Donner la forme normale disjonctive de la fonction parité pour $d = 3$ et $d = 4$. Proposer une méthode qui permet de définir un RCM pour la fonction parité en utilisant la forme normale disjonctive de la fonction. Donner le RCM obtenu pour $d = 3$. Donner le nombre de couches et le nombre de neurones formels (en fonction de d) par couche obtenu par cette méthode pour la fonction parité.
3. Déterminer un réseau connectionniste multicouche (RCM) à une couche cachée calculant la fonction parité en se basant sur l'indication suivante : faire en sorte que la i-ème cellule de la couche cachée retourne 1 si au moins i cellules de la rétine sont à 1. Donnez le RCM obtenu pour $d = 3$. Donner le nombre de couches et le nombre de neurones (en fonction de d) par couche obtenu par cette méthode et comparer avec la méthode précédente.

FDRG

Exercice 10.5 Réseaux connectionnistes et fonctions booléennes

Soit la fonction booléenne définie par :

$$f(x, y, z) = x\overline{y} + xyz + \overline{xyz}$$

Déterminez un perceptron linéaire à seuil pour chacun des trois monômes de f, puis déterminez un RMC calculant f.

FDRG

Exercice 10.6 Une alternative à la sigmoïde

Réécrire l'algorithme de rétropropagation de l'erreur quand la fonction f n'est pas une sigmoïde mais la fonction *tangente hyperbolique* :

$$f = \frac{1 - \exp^{-\sigma}}{1 + \exp^{-\sigma}}$$

Démontrer d'abord que cette fonction est solution de l'équation différentielle

$$\frac{df}{d\sigma} = 1 - f^2$$

Pourquoi cette fonction est-elle une alternative raisonnable à la fonction sigmoïde?

Exercice 10.7 Un réseau connectionniste sans sigmoïde

Montrer que les surfaces séparatrices produites par un réseau connexionniste multicouche dont la fonction f est linéaire sont des hyperplans. En déduire deux bonnes raisons de choisir une sigmoïde pour f.

Exercice 10.8 Un réseau connectionniste sans offset

Que se passe-t'il si on ne met d'offset ni sur l'entrée ni sur la couche cachée d'un réseau à deux couches?

Exercice 10.9 Réseaux connectionnistes et fonctions booléennes

Montrer que toute fonction de la logique des propositions sur d variables booléennes peut être réalisée par un réseau connexionniste à $d+1$ entrées binaires possédant un certain nombre de couches. Combien de couches?

Exercice 10.10 Choix de l'architecture

Pour un problème de classification, il a été décidé d'utiliser des réseaux connexionnistes. Il a été choisi d'utiliser une architecture multi couches avec une couche cachée et l'algorithme de rétropropagation du gradient. Il a été décidé de comparer les résultats sur les trois choix suivants:

- deux neurones formels dans la couche cachée,
- quatre neurones formels dans la couche cachée,
- six neurones formels dans la couche cachée.

L'échantillon d'apprentissage $\mathcal{S}$ contient 1000 exemples, l'échantillon test $\mathcal{T}$ contient 500 exemples. Avec la première architecture, on a produit un réseau connectionniste multicouches (RCM) h_1, avec la seconde un RCM h_2 et avec la troisième un RCM h_3. D'une manière générale, les performances d'un classifieur à réponse binaire h sur un ensemble $\mathcal{R}$ sont données par une matrice de confusion de la forme:

h sur $\mathcal{R}$	0	1
0	a	b
1	c	d

où a est le nombre d'éléments de classe 0 qui sont classés 0 par h, b est le nombre d'éléments de classe 1 qui sont classés 0 par h, c est le nombre d'éléments de classe 0 qui sont classés 1 par h et d est le nombre d'éléments de classe 1 qui sont classés 1 par h.

1. En utilisant les tableaux de performance donnés en fin d'énoncé, déterminer l'erreur sur l'ensemble d'apprentissage $\mathcal{S}$ et l'erreur réelle estimée sur T pour chacun des trois RCM h_1, h_2 et h_3.
2. En utilisant la table permettant de calculer les intervalles de confiance, déterminer les intervalles de confiance à 95% centrés autour de l'erreur estimée pour chacun des trois PMC. Déterminer les intervalles de confiance à 95% de la forme $[0, b]$ pour chacun des trois RCM.
3. Laquelle des trois architectures vous semble la mieux adaptée au problème? Comment expliquer les résultats obtenus pour les erreurs réelles et estimées pour les trois architectures?

h_1 sur $\mathcal{S}$	0	1
0	592	11
1	8	389

h_1 sur $\mathcal{T}$	0	1
0	287	12
1	8	193

h_2 sur $\mathcal{S}$	0	1
0	594	5
1	6	395

h_2 sur $\mathcal{T}$	0	1
0	289	4
1	6	201

h_3 sur $\mathcal{S}$	0	1
0	597	2
1	3	398

h_3 sur $\mathcal{T}$	0	1
0	283	13
1	12	192

FDRG

Exercice 10.11

Ecrire l'algorithme de rétropropagation du gradient dans le langage de programmation de votre choix.

Exercice 10.12 **Un apprentissage plus subtil**

On utilise la fonction sigmoïde. On souhaite pénaliser les poids $w(i,j)$ élevés en utilisant donc la fonction d'erreur définie par :

$$E = \sum_{i=1}^{m} D(\boldsymbol{u}_i, \boldsymbol{y}_i) = 1/2 \sum_{i=1}^{m} \sum_{k=1}^{d} [(\boldsymbol{u}_k - \boldsymbol{y}_k)^2 + \gamma \sum_{l,j} w_{jl}^2]$$

Modifier l'algorithme de rétropropagation du gradient en conséquence.
Au fait, pourquoi pénaliser les poids élevés?

FDRG

Exercice 10.13 **Reconnaisance d'images**

Les chiffres sont souvent représentés sur les écrans par une combinaison de 7 leds (Light Emiting Diode) qui peuvent être allumés ou éteints. Nous associons à chacun de ces leds un attribut binaire en posant qu'il prend la valeur 1 si le led correspondant est allumé et 0 s'il est éteint.

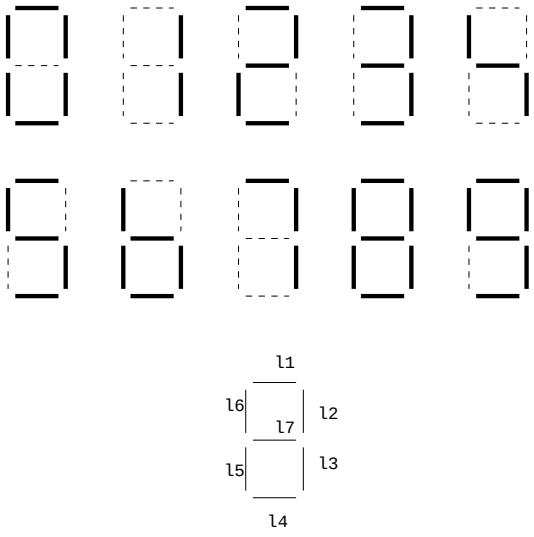


FIG. 10.1 – *Aucun perceptron à diamètre limité ne peut apprendre la connexité*

La population est l'ensemble des 10 chiffres $\Pi = \{0,1,2,3,4,5,6,7,8,9\}$, l'ensemble des descriptions est le sous ensemble de $D = \{0,1\}^7$ qui correspond aux descriptions des chiffres de Π. Nous nous intéressons au problème de classification des chiffres selon qu'ils sont premiers ou non. Les deux classes sont donc les

ensembles de descriptions associés aux deux ensembles $P = \{2, 3, 5, 7\}$ et $\overline{P} = \{0, 1, 4, 6, 8, 9\}$. La classe des chiffres premiers sera notée P ou 1, l'autre classe sera notée $\overline{P}$ ou 0.

Règles de choix des fonctions de classement On suppose que chaque chiffre est décrit par le seul led l_4. Décrire les procédures de classification associées à la règle majoritaire, à la règle du maximum de vraisemblance et à la règle de Bayes (la définition de ces règles est rappelée dans l'exercice 2.3. Calculez les erreurs réelles associées aux trois procédures trouvées.

Apprentissage par arbres de décision

1. On suppose que les chiffres sont décrits par les sept leds. Construire l'arbre de décision (Chapitre 11) produit par l'algorithme d'apprentissage par arbre de décision en utilisant la fonction entropie. Poursuivre les calculs jusqu'à obtenir un arbre parfait T_{max}.
2. On suppose que le led l_1 tombe en panne (sa valeur est donc 0 pour tous les chiffres). Comment l'arbre précédent classe t-il les 10 chiffres? Quelle est l'erreur (réelle) de ce classement.

Apprentissage par réseau connexionniste

1. A partir de l'arbre de décision trouvé précédemment, construire un réseau de neurones multicouches qui classifie les chiffres premiers.
2. Déterminer un perceptron à trois entrées correspondant aux leds l_1, l_2 et l_6 qui sépare les chiffres premiers des autres.
3. Existe-t-il un perceptron à 7 entrées (les valeurs des 7 leds) qui soit capable de classer d'une part les entrées correspondant à des chiffres, d'autre part celles qui ne correspondent pas à des chiffres?

FDRG

Quelques sites internet recommandés

Supervised Learning for Neural Networks: a tutorial with JAVA exercises
`http://diwww.epfl.ch/mantra/tutorial/french/`
Classification Toolbox for Matlab
`http://tiger.technion.ac.il/~eladyt/classification/index.htm`

Chapitre 11

Apprentissage par combinaison de décisions

Arbres de décision

Exercice 11.1  **Construction d'un arbre de décision**

Soit l'ensemble d'apprentissage composé des exemples et contre-exemples suivants :

	Attribut 1	Attribut 2	Attribut 3	Classe
1	1.	1.	FAUX	+
2	2.	2.	FAUX	+
3	3.	0.	FAUX	-
4	4.	1.	FAUX	-
5	3.	3.	VRAI	+
6	2.	4.	FAUX	+
7	4.	2.	FAUX	-
8	5.	3.	FAUX	-
9	4.	4.	VRAI	+

Les attributs 1 et 2 sont numériques, l'attribut 3 est binaire.
Construire un arbre de décision sur ces données d'apprentissage.

ACLM

Exercice 11.2  **Arbres non binaires**

Dans cet exercice, on s'intéresse aux arbres de décision non binaires sur des variables réelles. D'un nœud partent donc q branches, avec $q \geq 2$, portant des test du type : $(\boldsymbol{x}_i \leq a_1)$, $(a_1 \leq \boldsymbol{x}_i \leq a_2)$, $\ldots$, $(\boldsymbol{x}_i \geq a_{q-1})$

1. Montrer que pour tout arbre de décision de ce type, avec éventuellement un nombre de branches différent à chaque nœud, il existe un arbre de décision binaire réalisant la même classification.
2. Soit un arbre de décision non binaire ayant un seul nœud (la racine) et q feuilles, avec $q > 2$. Sa *hauteur* est définie comme la longueur maximale en nombre de nœuds entre la racine et une feuille. Quelle est la hauteur la plus grande d'un arbre binaire réalisant la même classification ? La hauteur minimale ?
3. Quel est le nombre minimal et maximal de nœuds de cet arbre binaire ?

ACLM

Exercice 11.3 Construction d'un arbre de décision

L'espace de représentation est constitué de quatre variables binaires. L'ensemble d'apprentissage est le suivant :

+	−
0110	1011
1010	0000
0011	0100
1111	1110

1. Construire un arbre de décision T_{max} (sans élagage).
2. Quelle est l'expression logique la plus simple (utilisant le moins de *ET* et de *OU*) pour caractériser chaque classe ?

ACLM

Exercice 11.4 Arbres de décision et logique booléenne

Construire un arbre de décision réalisant la fonction logique suivante sur des variables binaires :

1. x_1 *OU* (x_2 *ET* x_3)
2. (x_1 *ET* x_2) *OU* (x_3 *ET* x_4)

ACLM

Exercice 11.5 Validité de l'élagage

Relier l'élagage de l'arbre T_{max} et les méthodes de régularisation (chapitre 17, paragraphe 17.2.2).

ACLM

Exercice 11.6 Estimation de l'erreur

Calculer les intervalles de confiance à 95% des valeurs données par le logiciel `OC1` dans les exemples du chapitre 11, paragraphe 11.1.4.

ACLM

Exercice 11.7 Comparaison d'un perceptron et d'un arbre de décision

On considère des objets $(\boldsymbol{x}_1, \boldsymbol{x}_2)$ de $\mathbb{R}^2$. L'échantillon d'apprentissage est représenté graphiquement dans la figure 11.1 par des points étiquetés + pour les exemples positifs, − pour les exemples négatifs.

1. Soit l'échantillon donné par l'exemple 1 de la figure ci-dessous. Déterminer un arbre de décision binaire qui classifie correctement cet échantillon. Les tests qui étiquettent les noeuds de décision seront de la forme $\boldsymbol{x}_1 < M$ avec M entier (ou $\boldsymbol{x}_2 < M$). Peut-on trouver un perceptron à entrées réelles pour classifier cet échantillon ?
2. Même question avec l'exemple 2.
3. Quelle critique peut-on faire aux arbres de décision au vu du second exemple ? Comment pourrait-on essayer d'y remédier ? Quelle critique peut-on faire aux perceptrons au vu du premier exemple ?

Construire un réseau de neurones à couches cachées qui calcule la fonction de classification correspondant à l'arbre de décision trouvé en 1 (le procédé de construction doit pouvoir se généraliser).

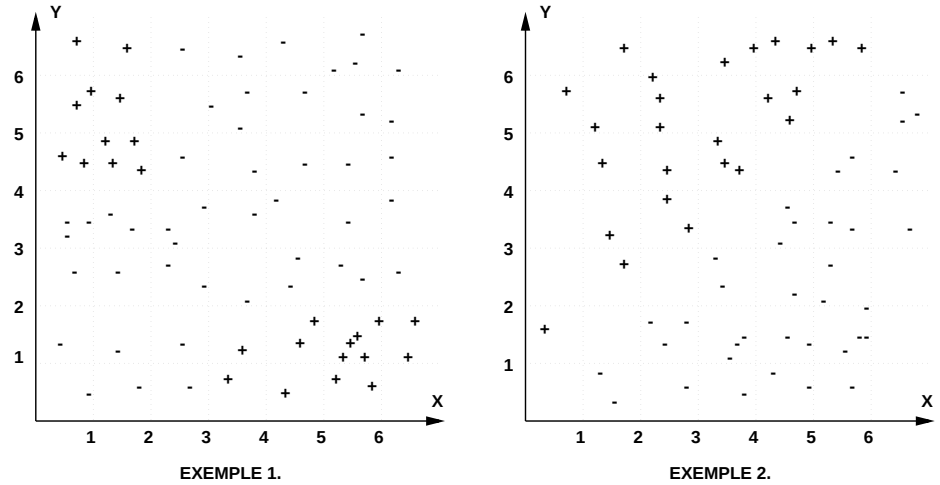


FIG. 11.1 – *Les exemples de l'exercice 11.7*

FDRG

Exercice 11.8 **Jouer au golf?**

Soit l'échantillon qui correspond à un ensemble de conditions météorologiques qui permettent (P pour Positif) ou pas (N pour Négatif) la pratique du golf:

	Outlook	*Temperature*	*Humidity*	*Windy*	Classe
1	sunny	81	78	false	N
2	sunny	80	90	true	N
3	overcast	83	80	false	P
4	rain	75	96	false	P
5	rain	69	75	false	P
6	rain	64	70	true	N
7	overcast	65	65	true	P
8	sunny	72	83	false	N
9	sunny	68	72	false	P
10	rain	71	74	false	P
11	sunny	75	69	true	P
12	overcast	70	77	true	P
13	overcast	85	70	false	P
14	rain	73	82	true	N

Soit trois arbres de décision décrits par les formules:
$t_1 = P$,
$t_2 = Humidity(N, P)$,
$t_3 = Outlook(Humidity(N, P), P, Windy(N, P))$, où les arcs sont étiquetés dans l'ordre par: « sunny, overcast et rain » pour l'attribut *Outlook*, par « > 75 et ≤ 75 » pour l'attribut *Humidity*, par « true et false » pour l'attribut *Windy*.

1. Dessiner les trois arbres;
2. Montrer que t_3 est un arbre du type T_{max};

3. Calculer l'erreur apparente sur l'échantillon pour ces trois arbres.

FDRG, d'après [Mitchell, 1997]

Exercice 11.9 Variations

Soit l'échantillon d'apprentissage suivant :

	x_1	x_2	x_3	Classe
1	0	V	N	A
2	1	V	I	A
3	0	F	O	B
4	1	V	N	A
5	1	V	O	A
6	1	F	N	A
7	0	F	O	B
8	0	V	I	A
9	0	F	N	B
10	1	V	I	B
11	1	F	O	A
12	1	F	I	A
13	0	V	O	B

1. Soit l'ensemble d'apprentissage constitué des exemples $\{1, \ldots, 9\}$. Quelle est la nature des attributs ? Construire l'arbre de décision T_{max} en choisissant les attributs dans l'ordre x_3, x_2, x_1. On l'appellera maintenant t_1.
2. Même question en utilisant l'ordre x_1, x_2, x_3. On appellera l'arbre trouvé t_2.
3. Peut-on trouver un arbre de décision sand élagage et d'erreur apparente nulle (de type T_{max}) si on considère l'ensemble d'apprentissage constitué des exemples $\{1, \ldots, 10\}$?
4. Soit l'ensemble d'apprentissage constitué des exemples $\{1, \ldots, 9\}$ et l'ensemble test constitué des exemples $\{11, 12, 13\}$. Soit l'arbre t_3 celui qui ne comporte aucun test et attribue la classe de plus forte probabilité *a priori* estimée sur l'échantillon. Soit t_4 l'arbre constitué d'un seul test sur x_1 et deux feuilles. Ces deux feuilles portent l'étiquette de la classe majoritaire dans le sous-ensemble qu'elles définissent. Calculer l'erreur apparente sur l'ensemble d'apprentissage, l'erreur apparente sur l'ensemble test et l'erreur apparente sur l'échantillon complet pour chacun des arbres $t_1, \ldots, t_4$.

FDRG

Exercice 11.10 Diagnostic médical

On dispose d'un échantillon de 200 patients. On sait que 100 sont malades (m) et 100 sont bien portants (bp). On dispose des informations suivantes :

	gorge irritée	gorge non irritée
température $< 37{,}5$	(6 bp,37 m)	(91 bp,1 m)
température $\geq 37{,}5$	(2 bp,21 m)	(1 bp,41 m)

et on considère l'arbre de décision suivant :

```
temperature<37.5 = oui
    gorge irritée = oui : malade
    gorge irritée = non : bien portant
temperature<37.5 = non : malade
```

1. Calculer pour cet arbre le critère de Gini à chaque nœud.
2. Calculer pour cet arbre toutes les valeurs du critère de l'entropie (développé dans le livre) aux nœuds de l'arbre.
3. On considère l'arbre vide. On a le choix entre l'attribut "Température $\leq 37{,}5$" et l'attribut "gorge irritée". Lequel choisit-on si on choisit l'attribut qui maximise l'entropie?

FDRG

Exercice 11.11 Retour sur le golf

On reprend l'exemple de l'exercice 11.8. On suppose que les attributs Temperature et Humidity ont été discrétisés par un expert de la façon suivante: les températures inférieures ou égales à 70 (cool), celles supérieures à 70 et inférieures ou égales à 80 (mild), celles supérieures à 80 (hot) ; le taux d'humidité inférieur ou égal à 75 (normal) et supérieur à 75 (high). On initialise l'arbre à l'arbre réduit à la seule feuille P. On utilise la fonction entropie.
Construire l'arbre de décision associé à notre échantillon.
Qu'obtient-on en laissant l'algorithme discrétiser les attributs Temperature et Humidity?

FDRG

Exercice 11.12 Réflexions sur l'entropie

On dispose d'un échantillon de 200 individus. On sait que 100 sont de classe 1 et 100 sont de classe 2. Le langage de représentation utilise deux attributs binaires A et B. On dispose des informations suivantes:

	A vrai	A faux
B vrai	(0,50)	(50,0)
B faux	(50,0)	(0,50)

Appliquer l'algorithme de base avec la fonction entropie. Quel problème rencontre-t-on? Doit-on s'arrêter? Choisir un attribut et poursuivre la construction de l'arbre, que remarque-t-on?
Supposons maintenant que le langage de description contienne non seulement les attributs A et B mais aussi d'autres attributs C, ..., Z. Que va-t-on obtenir en appliquant l'algorithme de base? Pourrait-on remédier à cette situation?

FDRG

Exercice 11.13 Classification de clients

Une banque dispose des informations suivantes sur un ensemble de clients:

client	M	A	R	E	I
1	moyen	moyen	village	oui	oui
2	élevé	moyen	bourg	non	non
3	faible	âgé	bourg	non	non
4	faible	moyen	bourg	oui	oui
5	moyen	jeune	ville	oui	oui
6	élevé	âgé	ville	oui	non
7	moyen	âgé	ville	oui	non
8	faible	moyen	village	non	non

L'attribut ternaire M décrit la moyenne des montants sur le compte client. Le second attribut ternaire A donne la tranche d'âge du client. Le troisième attribut ternaire R décrit la localité de résidence du client. Le dernier attribut binaire E a la valeur `oui` si le client a un niveau d'études supérieures. La classe

associée à chacun de ces clients correspond au contenu de la colonne I. La classe `oui` correspond à un client qui effectue une consultation de ses comptes bancaires en utilisant Internet.

1. Utiliser le classifieur naïf de Bayes (chapitre 2, Définition 2.6) et donner ses performances sur l'ensemble test T donné ci-après.
2. Construire l'arbre de décision en utilisant la fonction basée sur l'entropie. Calculer les performances sur l'ensemble test T.

test T	M	A	R	E	I
9	moyen	âgé	village	oui	oui
10	élevé	jeune	ville	non	oui
11	faible	âgé	village	non	non
12	moyen	moyen	bourg	oui	non

FDRG

Exercice 11.14  Valeurs manquantes

On considère un espace de description comprenant les trois attributs *forme*, *taille* et *couleur* prenant respectivement les valeurs *rond* et *carré*, *petit* et *grand*, *bleu*, *blanc* et *rouge*. L'attribut cible est binaire de valeurs *oui* et *non*.
Les données disponibles sont les suivantes (le «?») correspond à une valeur manquante) :

forme	**taille**	**couleur**	**classe**
rond	petit	bleu	oui
carré	grand	rouge	non
rond	?	blanc	oui
carré	petit	bleu	oui
rond	grand	bleu	oui
carré	grand	blanc	non
carré	?	blanc	oui
carré	grand	bleu	non
carré	petit	rouge	oui
rond	grand	blanc	oui

Valeur majoritaire de l'attribut On remplace les valeurs manquantes par la valeur majoritaire prise par cet attribut sur l'échantillon complet. Quelle valeur associe-t-on sur notre échantillon? Peut-on trouver un arbre de décision dont l'erreur apparente est nulle? Appliquer l'algorithme de construction d'arbre de décision en utilisant l'entropie.

On décide maintenant qu'un noeud est terminal lorsqu'il y a au plus un exemple mal classé associé à ce noeud. Détailler les calculs pour le test à choisir en racine de l'arbre.

Valeur majoritaire de l'attribut par classe Étant donné un exemple avec une valeur manquante, on remplace la valeur manquante par la valeur majoritaire prise par l'attribut correspondant pour les exemples de l'échantillon appartenant à la même classe. Quelles valeurs associe-t-on sur notre échantillon? Peut-on trouver un arbre de décision parfait (dont l'erreur apparente est nulle)? Quel arbre obtient-on en appliquant l'algorithme basé sur l'entropie?

Méthode utilisée par Quinlan dans le logiciel C4.5 Cette méthode consiste à ne plus attribuer une valeur à l'attribut mais une probabilité pour chacune des valeurs possibles. Ces probabilités sont estimées par les fréquences des valeurs possibles de cet attribut pour l'échantillon associé à une position p de l'arbre en construction. Par exemple, à la racine, la probabilité que l'attribut « taille » ait la valeur « petit » est de 3/8 car il y a 8 exemples pour lesquels la valeur de l'attibut « taille » est connue dont 3 ont la valeur « petit ». Quelles seraient les modifications à apporter à l'algorithme?

FDRG

Exercice 11.15 Bronzé?

On considère les données suivantes :

cheveux	taille	poids	crème solaire	classe ω
blond	moyenne	léger	non	1 = coup de soleil
blond	grande	moyen	oui	0 = bronzé
brun	petite	moyen	oui	0 = bronzé
blond	petite	moyen	non	1 = coup de soleil
roux	moyenne	lourd	non	1 = coup de soleil
brun	grande	lourd	non	0 = bronzé
brun	moyenne	lourd	non	0 = bronzé
blond	petite	léger	oui	0 = bronzé

1. On suppose que les individus sont décrits par le seul attribut poids. On suppose également que les probabilités *a priori* peuvent être estimées à l'aide des fréquences calculées à partir du tableau de données. Reproduire et compléter le tableau suivant :

	classe ω	
Estimations	0	1
$P(k)$		
$P(\text{léger}/\omega)$		
$P(\text{moyen}/\omega)$		
$P(\text{lourd}/\omega)$		

   Décrire les procédures de classification associées à la règle du maximum de vraisemblance et à la règle de Bayes.
2. On suppose maintenant que les individus sont décrits à l'aide des quatre attributs cheveux, taille, poids et crème solaire. Construire l'arbre de décision produit par l'algorithme d'apprentissage par arbre de décision en utilisant la fonction entropie. Détailler les calculs pour le choix de la racine.
3. Déduire de l'arbre trouvé à la question précédente un système à base de règles boolénnes (un système expert d'ordre 0). Montrer qu'une condition d'une des règles peut être supprimée tout en conservant la cohérence avec les données. Le système obtenu peut-il s'écrire sous forme d'un arbre de décision?

FDRG

Exercice 11.16 Une méthode d'élagage peu orthodoxe

La méthode classique d'apprentissage par arbres de décision consiste à construire un arbre à l'aide d'un échantillon d'apprentissage puis à l'élaguer à l'aide d'un échantillon test. L'idée sous-jacente est que la phase d'élagage doit permettre d'améliorer l'erreur réelle et que seul l'échantillon test permet de l'estimer de manière à peu près fiable.
Mais cette idée ne vaut que si l'on dispose de suffisamment d'exemples pour la première phase : un arbre dont l'erreur est catastrophique ne donnera jamais de bons résultats, quelque soit l'élagage qu'on lui fera subir.
Une alternative consiste à apprendre avec tous les exemples disponibles et à élaguer avec ces mêmes exemples. Mais cette idée ne peut fonctionner qu'à condition de ne pas se baser sur l'erreur apparente calculée sur l'ensemble d'apprentissage pour calculer l'estimation de l'erreur réelle.
Quinlan propose la méthode suivante :
On introduit un paramètre de confiance CF (par défaut, ce paramètre vaut 25%). Pour chaque feuille de l'arbre, notons N le nombre d'exemples qu'elle couvre et E le nombre d'erreurs de classification qu'elle induit dans l'échantillon. Soit p la probabilité pour qu'un nouvel exemple soit mal classé par cette feuille. La quantité E/N est donc un estimateur de p. Pour tenir compte du fait que l'arbre construit n'est pas indépendant des données, nous allons supposer que cet estimateur est très optimiste.

Plus précisément, soit E_p une variable aléatoire de loi binomiale de paramètres (N, p). C'est-à-dire que

$$Pr(E_p = k) = C_N^k p^k (1-p)^{N-k} \text{ pour } 0 \leq k \leq N$$

Nous posons $p(E, N) = max\{p | Pr(E_p \leq E) \geq CF\}$. Nous prendrons $p(E, N)$ comme valeur estimée de l'erreur réelle pour cette feuille.

Exemple : supposons qu'une feuille couvre $N = 4$ exemples et supposons qu'elle induise une erreur $(E = 1)$. On prend $CF = 25\ \%$. On a :

$$\begin{aligned} p(E, N) &= max\{p | Pr(E_p \leq 1) \geq 0,25\} \\ &= max\{p | Pr(E_p = 0) + Pr(E_p = 1) \geq 0,25\} \\ &= max\{p | (1-p)^4 + 4p(1-p)^3 \geq 0,25\} \end{aligned}$$

Pour cet exemple, on trouve $p \simeq 0,54$. Autrement dit, on estime par ce procédé l'erreur réelle pour cette feuille à 54% (au lieu des 25% fournis par l'erreur apparente).
Le reste est plus classique : on calcule l'erreur réelle estimée d'un arbre en faisant une somme pondérée des erreurs réelles estimées de ses fils.
Par exemple, si un noeud A a trois fils $A1$, $A2$ et $A3$, si le nombre d'exemples couverts par chacun de ces fils est respectivement de $N1$, $N2$ et $N3$, et si les erreurs réelles estimées pour chacun de ces fils sont $e1$, $e2$ et $e3$ alors l'erreur réelle estimée de A sera de $(N1e1 + N2e2 + N3e3)/(N1 + N2 + N3)$.
Pour élaguer un arbre, on applique l'algorithme suivant :

Tant qu'il existe un sous-arbre que l'on peut remplacer par une feuille sans faire croître l'erreur réelle estimée **alors** élaguer ce sous-arbre.

Application : On considère un espace de représentation comprenant deux attributs *adoption* et *education* pouvant prendre chacun trois valeurs : y, n et u. On suppose que la cible est binaire et que ses valeurs sont A et B.
On considère l'arbre suivant :

```
adoption = y : A (0;151)
adoption = u : A (0;1)
adoption = n :
   education = n : A (0;6)
   education = y : A (0;9)
   education = u : B (0;1)
```

Chacun des couples $(0\,;\,151)$, ... , est de la forme $(E; N)$.
On donne $p(0;6) = 0,206$; $p(0;9) = 0,143$; $p(0;1) = 0,750$; $p(1;16) = 0,159$; $p(0;151) = 0,009$; $p(1;168) = 0,016$.

1. Calculez l'erreur réelle estimée pour l'arbre
   ```
   adoption = n :
      education = n : A (0;6)
      education = y : A (0;9)
      education = u : B (0;1)
   ```
2. Calculez l'erreur réelle estimée pour l'arbre complet
3. Peut-on élaguer le sous-arbre de racine `adoption = n`?
4. Peut-on remplacer l'arbre entier par une feuille?
5. Après élagage, quelle est l'erreur réelle estimée?

D'autres algorithmes de construction de règles

Exercice 11.17  **Algorithme de couverture**

Un arbre de décision peut être vu comme une disjonction de conjonctions sur des sélecteurs sur les attributs (voir le paragraphe 11.1.5). On peut apprendre de telles formules sur des exemples autrement que par la construction d'un arbre de décision. L'algorithme de *couverture* est un exemple d'algorithme alternatif. Donnons d'abord son aspect général avant de rentrer dans les détails.
Supposons que nous ayons un algorithme `ApprendreUneConjonction`(Exemples, r) répondant aux critères suivants (nous verrons ensuite comment on peut construire un tel algorithme) :

- `ApprendreUneConjonction` prend en entrée des exemples positifs et négatifs ;
- Il calcule un concept r qui est une conjonction de sélecteurs sur les attributs ;
- Ce concept couvre beaucoup d'exemples positifs et peu d'exemples négatifs.

Rappellons qu'un sélecteur transforme un attribut en variable binaire. Par exemple, le sélecteur `jaune` = $VRAI$ sur l'attribut nominal `couleur` répond $VRAI$ si la valeur de cet attribut est `jaune`. Un sélecteur sur une variable numérique est une comparaison à un seuil.
Avec cet algorithme `ApprendreUneConjonction`, il est facile de passer à l'apprentissage d'une disjonction de conjonctions par une procédure de couverture comme celle-ci :

Procédure `Couverture`(Exemples)
RèglesApprises $\leftarrow$ {}
tant que `Qualité`(Règle, Exemples) > Seuil **faire**
`ApprendreUneConjonction`(Exemples, r)
RèglesApprises = RèglesApprises $\cup$ r
Exemples = Exemples - {les exemples correctement classés par r }
fin tant que

L'algorithme de couverture est de type glouton : il ne remet pas en question ses choix et augmente le nombre de règles en disjonction une par une. La procédure `Qualité`(Règle, Exemples) a pour rôle de mesurer la qualité apparente globale de l'ensemble de règles appris sur les exemples d'apprentissage.
Il reste à dire comment `ApprendreUneConjonction` peut fonctionner. Son problème est de construire une conjonction de sélecteurs sur l'ensemble courant d'exemples en suivant le principe ERM : minimiser le taux d'erreur sur cet ensemble d'exemples en acceptant « beaucoup » de positifs et « peu » de négatifs. Ce principe peut se quantifier par exemple par l'entropie croisée, comme dans la construction d'un arbre de décision.
Prenons par exemple le cas où il y a trois attributs, deux nominaux et un binaire, le premier prenant trois valeurs le second trois. Le nombre possible de conjonctions de sélecteurs, donc d'hypothèses, est $3 \times 3 \times 2 = 18$. Le problème se ramène donc à la recherche de la meilleure hypothèse parmi 18. D'une manière générale, il s'agit de la recherche d'un optimum dans un espace d'hypothèses de taille exponentielle en fonction du nombre d'attributs.
Pour l'exercice, prenons l'ensemble d'exemples suivants :

exemple	X	Y	Z	$Classe$
1	A	b	0	$VRAI$
2	A	b	1	$VRAI$
3	A	c	0	$FAUX$
4	B	a	0	$VRAI$
5	B	a	1	$VRAI$
6	B	c	0	$VRAI$
7	B	a	1	$FAUX$
8	C	a	0	$FAUX$

Nous imposons que l'algorithme `ApprendreUneConjonction` considère les attributs dans l'ordre du tableau : X, Y, Z et que les valeurs de chacun soient examinées dans l'ordre où elles apparaissent avec les exemples. X sera successivement testé comme A, B et C.

Le programme énumérera donc dans l'ordre les hypothèses $(X = A)$, $(X = B)$, $(X = C)$, $(X = A) \wedge (Y = b)$, etc.
D'autre part, la procédure `Qualité`(Règle, Exemples) est égale au nombre d'exemples positifs couverts moins le nombre d'exemples négatifs couverts, le tout multiplié par le nombre de termes en conjonction dans la règle. Par conséquent :
$X = A$ a une qualité de $(2 - 1) \times 1 = 1$
$X = B$ a une qualité de $(3 - 1) \times 1 = 2$
$X = C$ a une qualité de $(0 - 1) \times 1 = -1$
$(X = A) \wedge (Y = b)$ a une qualité de $(2 - 0) \times 2 = 4$

Question 1

1. Quelle est la meilleure hypothèse? Est-il besoin de les explorer toutes pour connaître la meilleure?
2. Quelle est la meilleure si on prend la meilleure parmi les trois premières et qu'on fixe désormais la valeur du premier attribut à cette valeur, et ainsi de suite (ce qui fait que l'algorithme `ApprendreUneConjonction` est glouton, lui aussi).
3. Quelle est alors la taille de l'espace exploré?

Question 2

1. Que donne l'algorithme de couverture sur cet exemple dans le cas où l'algorithme `ApprendreUneConjonction` énumère toutes les possibilités?
2. et dans le cas où il est glouton?

Question 3
Quel arbre de décision aurait été construit sur les mêmes données?

ACLM

Exercice 11.18  **AQ**

Mitchell, page 280

ACLM

Quelques sites internet recommandés

Classification Toolbox for Matlab
`http://tiger.technion.ac.il/~eladyt/classification/index.htm`

Chapitre 12

L'apprentissage de réseaux bayésiens

Exercice 12.1  Implication et probabilité conditionnelle

Soit un réseau bayésien à deux nœuds A et B, avec B fils de A. On connaît donc $P(A)$, $P(B|A)$ et $P(B|\neg A)$.

1. Calculer $P(A \Rightarrow B)$, où « $\Rightarrow$ » représente l'implication logique.
2. Quand a-t'on $P(A \Rightarrow B) = P(B|A)$?
3. Supposons qu'au lieu de $P(B|A)$ et $P(B|\neg A)$ on connaisse $P(A \Rightarrow B)$. Que peut-on dire de $P(B)$?

d'après [Nilsson, 1998]

Exercice 12.2  Le cygne est sauvage

Avec le réseau de la figure 12.3 du livre, dire pourquoi le fait de savoir que le prix du saumon a augmenté permet de conclure que le cygne a une grande probabilité d'être sauvage.

ACLM

Exercice 12.3  Un zoo

La moitié des cygnes du zoo a faim. Les deux tiers des animaux de ce zoo sont des cygnes. Un animal sur dix du zoo a faim. Quelle est la probabilité qu'un animal tiré au hasard soit un cygne qui a faim?

ACLM, d'après [Moore, 2002]

Exercice 12.4  Un autre zoo

La moitié des animaux du zoo sont des canards. Un canard est plus souvent dans la mare que n'importe quel animal du zoo. On choisit un animal au hasard dans la mare. Quelle est la probabilité que ce soit un canard?

ACLM, d'après [Moore, 2002]

Exercice 12.5  Encore un autre zoo

La moitié des cygnes sont gris. Un cygne gris est le plus souvent âgé de moins d'un an. Les jeunes cygnes de moins d'un an sont très appréciés du public. Quelle est la probabilité qu'un cygne gris apprecié du public soit âgé de mois d'un an?

ACLM, d'après [Moore, 2002]

Exercice 12.6  Un calcul de probabilités conditionnelles

On donne un réseau bayésien à quatre nœuds : H, S, F et W et à trois flèches : de H vers S et vers F et de S vers W, avec les probabilités suivantes :

$$P(H) = 0.5 \quad P(S|H) = 0.1 \quad P(W|S) = 0.5 \quad P(F|H) = 0$$
$$P(S|\neg H) = 0.5 \quad P(W|\neg S) = 1 \quad P(F|\neg H) = 0.5$$

Question
Calculer $P(W|F)$.

ACLM, d'après [Moore, 2002]

Chapitre 13

L'apprentissage de modèles de Markov cachés

Exercice 13.1 Correspondant de guerre

Une escadrille de quatre avions effectue des missions diurnes au-dessus de l'ennemi. La probabilité qu'un avion ne revienne pas d'une mission est p. L'escadrille décolle si le nombre d'avions restants est au moins deux. D'autre part, le commandement fournit un avion neuf quand l'escadrille comporte strictement moins de trois avions.

Question 1 Donner un modèle de Markov observable à quatre états pour ce processus.

Chaque soir, les pilotes restants vont au bar de la caserne et racontent à tout le monde leurs aventures de la journée. Un journaliste cherche à avoir des informations, mais il n'a pas le droit de voir les avions ni de rentrer au bar. Il écoute à la porte, mais ne peut pas percevoir clairement ce qui se dit. Il suppose cependant que plus les conversations sont animées à l'intérieur, plus la journée a été dure et tragique, et arrosée en conséquence. Le niveau de bruit dans le bar est donc la seule observation qu'il puisse donc faire.

Question 2 Donner un modèle de Markov caché à quatre états qui relie la suite quotidienne des observations du journaliste à la suite quotidienne du nombre d'avions.

ACLM, d'après [Mari and Schott, 2001], pages 13-14

Exercice 13.2 Programmer un HMM

Programmer deux HMM

LM

Exercice 13.3 Un exemple de calculs dans les HMM

Le HMM de la figure 13.1 est défini complètement s'il on ajoute qu'il émet des observations sur l'alphabet $\{a, b, c\}$ et qu'on donne les matrices ci-dessous :

$$B = \begin{pmatrix} 0.5 & 0.5 & 0 \\ 0.5 & 0 & 0.5 \\ 0.5 & 0 & 0.5 \end{pmatrix} \quad \pi = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

On observe la séquence

$$acabbcbcc$$

Construire le tableau des valeurs $\alpha_t(i)$ pour $t = 1, 9$ et $i = 1, 3$.

LM

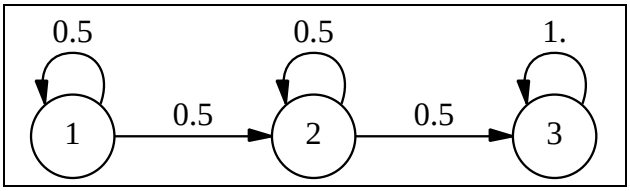


FIG. 13.1 – *Un HMM.*

Quelques sites internet recommandés

Bayes Net Toolbox for Matlab
`http://www.cs.berkeley.edu/~murphyk/Bayes/bnt.html`

Chapitre 14

L'apprentissage bayésien et son approximation

Généralités et règle de Bayes

Exercice 14.1  **Rappels**

Une *épreuve* est le résultat d'une expérience aléatoire. On note Φ l'ensemble de toutes les épreuves possibles et ϕ une épreuve ou événement élémentaire. Un *événement aléatoire* est un événement dont la réalisation ou la non réalisation dépend du résultat d'une expérience aléatoire. Un événement aléatoire sera représenté par l'ensemble des événements élémentaires qui le réalisent. Les opérations logiques sur les événements vont correspondre aux opérations ensemblistes.
Considérons, par exemple le lancer de deux dés. On a $\Phi = \{1, \ldots, 6\}^2$. L'événement A défini par : « amener un total au moins égal à 10 » est $A = \{(x, y) \mid x + y \geq 10\}$. Soit l'événement B : « amener deux dés identiques ». On a alors $A \cap B = \{(5, 5), (6, 6)\}$.
Les correspondances entre formalismes peuvent être résumées dans le tableau suivant :

terminologie probabiliste	**terminologie ensembliste**
événement certain	espace des épreuves Φ
événement impossible	ensemble vide $\emptyset$
événement contraire	complémentaire
et	intersection
ou	réunion
événements incompatibles	ensembles disjoints
système exhaustif	partition
implication	inclusion

Nous nous limitons au cas où Φ est dénombrable. Une *probabilité* sur Φ est une application P de Φ dans $[0, 1]$ telle que la somme des probabilités des événements élémentaires est égale à 1. On peut montrer les résultats suivants :

- soit $(A_i)_{i \in I}$ une famille d'événements deux à deux incompatibles, $P(\bigcup_{i \in I} A_i) = \sum_{i \in I} P(A_i)$,
- $P(A) = \sum_{\phi \in A} P(\phi)$,
- $P(A \cup B) = P(A) + P(B) - P(A \cap B)$,
- $P(\overline{A}) = 1 - P(A)$,
- si $A \subseteq B$, $P(A) \leq P(B)$.

Soit (Φ, P) un espace de probabilités et A un événement de probabilité non nulle, la probabilité conditionnelle $P(B/A)$ d'un événement B sachant A est définie par : $P(B/A) = P(A \cap B)/P(A)$.
Deux événements sont *indépendants* si ils vérifient les conditions équivalentes suivantes :

(i) $P(B/A) = P(B)$

(ii) $P(A/B) = P(A)$
(iii) $P(A \cap B) = P(A)P(B)$

Questions

Démontrez que :

- $P(./A)$ est une probabilité sur Φ et sur A,
- $P(A/B) = (P(A) \times P(B/A))/P(B)$ (formule de Bayes),
- soit $(A_i)_{i=1}^n$ un système exhaustif, $P(B) = \sum_{i=1}^n P(B/A_i)P(A_i)$.

FDRG

Exercice 14.2 Application de la règle de Bayes

On dispose d'une population Π constituée d'un ensemble de pièces qui sont équitables, biaisées avec une probabilité 1/3 pour Face, ou encore biaisées avec une probabilité de 1/4 pour Face. Une expérience consiste à jeter une pièce 20 fois de suite. Au vu du résultat d'une telle expérience, on souhaite classifier la pièce. On considère donc les trois classes $\{1, 2, 3\}$ qui correspondent à une probabilité de Face égale à respectivement 1/2, 1/3 et 1/4. On fera l'hypothèse que les classes sont *a priori* équiprobables. Une description est un mot de l'ensemble $\{P, F\}^{20}$, où P correspond à Pile et F à Face. Une procédure de classification doit associer à un mot de cet ensemble une classe. Soit la description

$$d = PPFPPFFPFPPFPPFPPPFP.$$

Trouver les formules de calcul des trois quantités $P(1/d)$, $P(2/d)$ et $P(3/d)$. Comment d serait-elle classifiée si on utilise la règle de décision de Bayes? On décide de prolonger cette expérience, on relance cette même pièce 20 fois. Indiquer un choix à faire sur les probabilités *a priori* qui serait plus intéressant que l'hypothèse initiale d'équiprobabilité.

FDRG

Exercice 14.3 Choisir un champignon par la règle de Bayes

La population Π est un ensemble de champignons. Il y a deux classes $\{1, 2\}$ où 1 est la classe des champignons vénéneux. Le langage de description est constitué de l'attribut binaire *volve*. On dispose des informations suivantes :

classe k	1 : vénéneux	2 : comestible
$P(k)$	0.05	0.95
$P(volve/k)$	0.9	0.2

1. Je ramasse les champignons si la règle de Bayes les classifie dans la classe des comestibles. Est-ce que je ramasse les champignons ayant une volve? Appliqueriez-vous cette règle si vous alliez ramasser des champignons?
2. On définit un coût pour tout couple de classes (k, i) noté $cout(k, i)$. On définit alors le coût moyen de l'affectation à la classe k d'une description d de D par :

$$coutMoyen(k/d) = \sum_{i \in \{1,\dots,c\}} cout(k, i) \times P(i/d).$$

La règle de décision du coût minimum est : *Choisir $C_{coutmin}$ qui à toute description d associe la classe k qui minimise $coutMoyen(k/d)$.*
On définit sur notre exemple les coûts suivants :

$$cout(1, 1) = cout(2, 2) = 0,\ cout(1, 2) = 2,\ cout(2, 1) = \infty.$$

J'utilise la règle du coût minimum. Est-ce que je ramasse les champignons ayant une volve?

FDRG

Exercice 14.4 La règle du *minimax*

On considère un problème de classification binaire, l'espace de description D étant constitué de deux descripteurs également binaires. On suppose que la répartition des descriptions dans chaque classe est conforme au tableau suivant :

d	00	01	10	11
classe 1	80	10	10	0
classe 2	0	5	15	80

On suppose que l'on ne dispose d'aucune information sur les poids respectifs des classes (avec lesquels on pourrait estimer leur probabilité *a priori*). Le problème de la classification ne pose aucun problème pour les descriptions 00 et 11. Mais que doit-on faire si l'on observe 01 ou 10?
La méthode du *minimax* consiste à introduire le paramètre $p = Pr(classe1)$, à calculer pour chaque valeur de p la règle de décision issue de la règle de Bayes. Pour chacune de ces règles, on calcule son erreur maximale. On choisit alors la règle qui minimise cette erreur maximale (d'où le nom de « minimax »).
Quelle est la règle de décision définie par cette procédure pour l'exemple ci-dessus?

FDRG

Exercice 14.5 Variations sur la règle de Bayes

On considère deux attributs pour déterminer la nationalité d'un individu. L'attribut *taille* qui peut prendre les valeurs *grand* ou *petit*, l'attribut *couleur des cheveux* qui peut prendre les valeurs *brun* ou *blond*. Les nationalités possibles sont *français* et *suédois*.
On suppose que les populations françaises et suédoises se répartissent selon le tableau suivant :

	petit, brun	petit, blond	grand, brun	grand, blond
Suédois	10	20	30	40
Français	25	25	25	25

1. Dans une assemblée comprenant 60 % de suédois et 40 % de français, décrire (voir par exemple l'exercice 2.3)
   (a) la règle de décision majoritaire
   (b) la règle du maximum de vraisemblance
   (c) la règle de Bayes
2. Calculez les probabilités d'erreur de chacune de ces règles
3. On suppose maintenant que l'on ne connaît plus les proportions respectives des suédois et des français. On note α la proportion des suédois ($\alpha \in [0, 1]$).
   (a) Décrire, selon les valeurs possibles de α , les règles de Bayes correspondantes.
   (b) Parmi les 5 règles que vous aurez détaillées à la question précédente, choisir celle qui **dans le pire des cas**, possède la probabilité d'erreur minimale.

FDRG

Exercice 14.6 Qui est malade?

La population consiste en un ensemble de patients. Ces patients doivent être répartis en deux classes, la classe M (pour malade) et la classe S (pour sain). Les individus sont décrits à l'aide de deux attributs logiques T et C. L'attribut T a la valeur vrai lorsque la tension artérielle d'un patient est anormale et

l'attribut C a la valeur vrai lorsque le taux de cholestérol d'un patient est anormal. On suppose que la population est un espace probabilisé et on note P la loi de probabilité. Nous allons utiliser l'approche bayésienne classique pour classer les patients au vu de leurs descriptions. Les probabilités suivantes sont connues :

classe k	S (sain)	M (malade)
$P(k)$	0.7	0.3
$P(T/k)$	0.25	0.7
$P(C/k)$	0.4	0.7

Dans ce tableau $P(k)$ représente la probabilité qu'un élément de la population soit de classe k, $P(T/k)$ représente la probabilité qu'un élément de classe k ait une tension artérielle anormale (T vaut vrai) et $P(C/k)$ représente la probabilité qu'un élément de classe k ait un taux de cholestérol anormal (C vaut vrai).
Pour pouvoir calculer les probabilités nécessaires à l'application de la règle de Bayes, nous allons faire l'hypothèse supplémentaire suivante : les deux attributs sont indépendants. Cette hypothèse permet, par exemple, d'écrire que $P(C \cap \overline{T}/S) = P(C/S) \times P(\overline{T}/S)$.

1. Déterminer la procédure de classification C_{Bayes} en utilisant la règle de décision de Bayes. La donner sous forme d'un arbre de décision.
2. Soient les procédures de classification C_1 et C_2 associées aux arbres de décision $t_1 = S$ et $t_2 = T(M, S)$. Calculer les erreurs au sens de la probabilité d'erreur pour C_1, C_2 et C_{Bayes}.
3. Plutôt que de chercher à minimiser l'erreur au sens de la probabilité d'erreur, on peut introduire des coûts pour les mauvaises classifications. On définit un coût pour tout couple de classes (k, i) noté $cout(k, i)$. On définit alors le coût moyen de l'affectation à la classe k d'une description d de D par :

$$coutMoyen(k/d) = \sum_{i \in \{1,\cdots,c\}} cout(k, i) \times P(i/d).$$

La règle de décision du coût minimum est : *Choisir $C_{coutmin}$ qui à toute description d associe la classe k qui minimise $coutMoyen(k/d)$.*
On définit sur notre exemple les coûts suivants :

$$cout(S, S) = cout(M, M) = 0,\ cout(S, M) = 2,\ cout(M, S) = 1.$$

Déterminer $C_{coutmin}$ et la donner sous forme d'un arbre de décision.

FDRG

Exercice 14.7 ☞ Programmer le classifieur naïf de Bayes

Programmer le classifieur naïf de Bayes à partir de données structurées. On utilisera les conventions du site internet `http://www.ics.uci.edu/ mlearn/MLRepository.html` : le programme prendra en entrée deux fichiers d'extensions `data` et `names` ; le fichier d'extension `data` contient autant de lignes que d'exemples, les valeurs d'attribut sont séparées par des virgules, le dernier attribut est la classe ; le fichier d'extension `names` contient sur la première ligne terminée par un point les valeurs possibles des classes, puis une ligne par attribut formée du nom de l'attribut, le symbole :, la liste des valeurs pour cet attribut, terminée par un point. Le résultat sera un fichier d'extension `bayes` contenant toutes les estimations nécessaires. Enfin, un programme `naivebayes` prenant en entrée une description et les fichiers utiles permettra de classer la description avec le classifieur naïf.

FDRG

Exercice 14.8 Classification bayésienne naïve de textes

On dispose d'un corpus de textes classés : par exemple, des courriers électroniques classés dans des dossiers, des textes de journaux classés par thème. On souhaite classer un texte en utilisant les informations du corpus. Une méthode possible est d'utiliser le classifieur de Bayes naïf. Il nous faut, auparavant, préciser le langage de description et le mode d'estimation des probabilités.
À partir du corpus de textes, on constitue le vocabulaire $\mathcal{V}$ qui est l'ensemble de tous les mots apparaissant dans le corpus. Le langage de description est alors l'ensemble de tous les textes possibles sur ce vocabulaire $\mathcal{V}$. Par souci de simplification, un texte sera condensé en un « sac de mots » qui est l'ensemble, avec répétition, des mots du texte. On a oublié la notion de position dans le texte et la notion relative de position d'un mot par rapport à un autre. Soit $\{1, ..., c\}$ l'ensemble des classes, soit t un texte (un sac de mots) à classer, le classifieur naïf de Bayes s'écrit :

$$C_{NaiveBayes}(t) = \underset{k \in \{1,...,c\}}{\operatorname{argmax}} \prod_{m \in t \cap \mathcal{V}} \hat{P}(m/k) \times \hat{P}(k) \tag{14.1}$$

L'estimation $\hat{P}(k)$ est faite par :

$$\hat{P}(k) = \frac{Card(T(k))}{Card(T)} \tag{14.2}$$

où $Card(T(k))$ est le nombre de textes de classe k et $Card(T)$ le nombre total de textes. On estime $P(m/k)$ par :

$$\hat{P}(m/k) = \frac{\sharp m, T(k)}{\sharp T(k)} \tag{14.3}$$

où $\sharp m, T(k)$ est le nombre d'occurrences du mot m de $\mathcal{V}$ dans l'ensemble des textes de classe k et $\sharp T(k)$ le nombre total d'occurrences de mots dans l'ensemble des textes de classe k.
Il suffit alors d'utiliser l'équation 14.1 pour classer un texte t. On est cependant confronté au problème suivant : si le texte t à classer contient un mot m n'apparaissant dans aucun document de classe k, la quantité $\hat{P}(m/k)$ vaut alors 0, de même que la quantité $\prod_{m \in t \cap \mathcal{V}} \hat{P}(m/k) \times \hat{P}(k)$. Le classifieur de Bayes ne pourra lui attribuer la classe k, même si, par ailleurs, le texte t contient d'autres mots très fréquents dans les textes de classe k. Pour éviter ce biais, on modifie l'estimateur utilisé par :

$$\hat{P}(m/k) = \frac{1 + \sharp m, T(k)}{Card(\mathcal{V}) + \sharp T(k)} \tag{14.4}$$

Enfin, revenons sur le choix du vocabulaire $\mathcal{V}$. Plusieurs choix sont possibles:
- $\mathcal{V}$ est l'ensemble des mots de la langue (environ 50 000 mots en anglais) ;
- $\mathcal{V}$ est l'ensemble des mots du corpus d'apprentissage ;
- $\mathcal{V}$ est l'ensemble des mots du corpus d'apprentissage dont on retire
  1. les mots les plus fréquents qui sont souvent des mots peu signifiants pour la classification : et, le, la, ...
  2. les mots rares
- un algorithme détermine $\mathcal{V}$ en calculant l'ensemble des mots pertinents pour la classification (en général, en utilisant des critères basés sur le contenu en information tels que l'entropie).

Pour conclure, l'algorithme d'apprentissage pour le classifieur de Bayes naïf consiste en un calcul du dictionnaire $\mathcal{V}$ et en un calcul des estimations :

Phase d'apprentissage pour Bayes naïf
donnée : un ensemble T de textes classés dans des classes $1, \ldots, c$
1. calculer le vocabulaire $\mathcal{V}$ à partir de T
2. pour toute classe k, calculer $\hat{P}(k)$ en utilisant l'équation 14.2
3. pour toute classe k et tout mot m de $\mathcal{V}$, calculer $\hat{P}(m/k)$ en utilisant l'équation 14.4

Et la classification d'un texte t se fait par l'algorithme suivant :

classifieur de Bayes naïf
donnée : $\mathcal{V}$, les estimations $\hat{P}(k)$ et $\hat{P}(m/k)$ pour tout m et tout k
entrée : un texte t
pour toute classe k, calculer $\prod_{m \in t \cap \mathcal{V}} \hat{P}(m/k) \times \hat{P}(k)$
sortie : la classe k qui maximise le calcul précédent

Illustration : Le but est de classer des phrases dans deux classes, selon leur thème : la radio ou la télévision. Étant donné l'échantillon :

Exemples de la classe TV :
Le programme TV n'est pas intéressant. La TV m'ennuie.
Les enfants aiment la TV
On reçoit la TV par onde radio
Exemples de la classe Radio :
Il est intéressant d'écouter la radio
Sur les ondes, les programmes pour enfants sont rares
Les enfants vont écouter la radio ; c'est rare!

et le vocabulaire pour mettre en oeuvre cette procédure de classification étant :$\mathcal{V}$= *TV, programme, intéressant, enfants, radio, onde, écouter, rare*. Comment serait classée la phrase : *J'ai vu la radio de mes poumons à la TV!*?

FDRG

Exercice 14.9  **Programmer un classificateur de textes**

Programmer le classifieur naïf de Bayes à partir de textes. On permettra à l'utilisateur de définir des seuils de fréquence pour la constitution du vocabulaire V. Pour constituer des bases d'exemples, vous n'avez que l'embarras du choix. Prenez par exemple des news issues de différents groupes ou recherchez sur le web la base d'exemples REUTERS.

FDRG

Règle des $k - ppv$.

Exercice 14.10  **Courbes d'erreur**

Pour un ensemble d'apprentissage à deux classes équiprobables, avec m éléments (m est supposé assez grand, par exemple 10^3) et une décision par k plus proches voisins, donner l'allure des courbes de l'erreur apparente et de l'erreur réelle en fonction de k. Comment choisir au mieux k?

ACLM

Exercice 14.11  **Condensation**

L'algorithme de condensation donne un résultat qui varie en fonction de l'ordre dans lequel les exemples d'apprentissage sont rangés. Donner un exemple de cette propriété sur un ensemble simple d'apprentissage. Proposer une façon de l'ordonner pour rendre l'algorithme le plus efficace possible (le critère d'efficacité est le nombre de points sélectionnés).

ACLM

Exercice 14.12 **Exercices de condensation**

On donne l'ensemble d'apprentissage suivant :

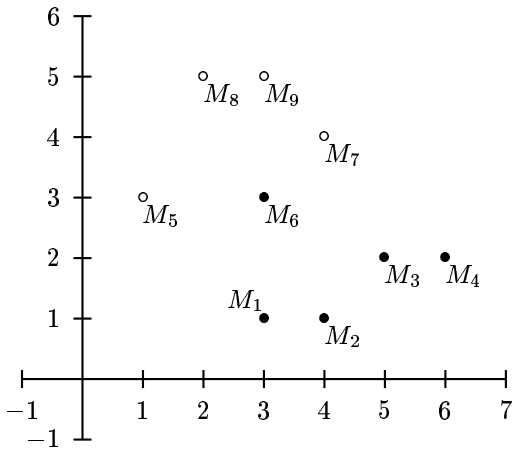


FIG. 14.1 – *L'ensemble d'apprentissage.*

La distance choisie est la distance euclidienne. Par exemple, $\delta_E(M_5, M_8) = \sqrt{(3-5)^2 + (4-5)^2} = \sqrt{5}$.
Les points $\{M_1, M_2, M_3, M_4, M_6\}$ sont les éléments d'apprentissage d'une classe ω_1 et les points $\{M_5, M_7, M_8, M_9\}$ sont ceux d'une classe ω_2.

Question 1 Donner l'équation d'une séparatrice linéaire dont l'erreur apparente est nulle. La tracer.

Question 2 Construire un arbre de décision donnant une erreur apparente nulle. NB : il n'y a pas besoin de faire de calculs de logarithmes ; il suffit de raisonner en comparant les tableaux (2×2) qui caractérisent chacun la distribution conjointe d'un attribut et du test sur les données d'apprentissage.

Question 3 On considère seulement les points M_5, M_6 et M_7. Quelles sont les surfaces de Voronoï correspondantes ? Quelle est la surface séparatrice par la règle du 1-PPV ? Quelle erreur fait cette séparatrice sur l'ensemble des points ?

Question 4 On considère de nouveau tous les points. Quelles sont les surfaces de Voronoï correspondantes ? Quelle est la surface séparatrice par la règle du 1-PPV ? Y a-t'il des points qui n'interviennent pas dans son tracé ?

Question 5 Que donne l'algorithme de condensation en prenant les points dans l'ordre M_1, M_2, ... M_9. Quelle est la surface séparatrice par la règle du 1-PPV ?

Question 6 Que donne l'algorithme de condensation en prenant les points dans l'ordre M_9, M_4, M_6, M_1, M_5, M_7, M_2, M_3 et M_8. Quelle est la surface séparatrice par la règle du 1-PPV ?

ACLM

Exercice 14.13  **Influence de d**

Les algorithmes d'édition et de condensation sont d'autant moins efficaces que la dimension d de l'espace de représentation est grande. Pourquoi ?

ACLM

Exercice 14.14  **L'algorithme AESA en version complète**

L'algorithme AESA [1], imaginé et développé par Vidal [Vidal, 1994], améliore les performances de l'algorithme classique de recherche par plus proche voisin.

Principe général Le processus de recherche classique par plus proche voisin est simple. Si $\Delta : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ est la fonction utilisée pour le calcul de la distance entre deux objets de $\mathcal{X}$, et si $\mathcal{S} \subseteq \mathcal{X}$ est l'ensemble d'apprentissage, alors l'algorithme de base pour rechercher l'exemple de $\mathcal{S}$ le plus proche de x s'exprime de la façon suivante :

PROCÉDURE : Recherche classique du plus proche voisin
début
  $s :=$ un élément de $\mathcal{S}$ choisi arbitrairement ;
  $pp := s$;
  pour tous les $p \in \mathcal{S} - \{s\}$ **faire**
    si $\Delta(p, x) < \Delta(pp, x)$ **alors**
      $pp := p$;
    fin si
  fin pour
  retourne pp ;
fin procédure

Au final, pp est le plus proche voisin de x. Le temps de calcul dépend donc directement de la complexité temporelle du calcul de distance et du nombre d'exemples. Afin d'améliorer la performance du système, on peut modifier la recherche du plus proche voisin en diminuant le nombre de calculs de distance, en suivant l'algorithme d'accélération AESA dont une version simplifiée a été donné au chapitre 14, P 442. Cet algorithme ne pose aucune contrainte sur l'organisation des données. Ses deux principes majeurs sont le non-examen d'éléments de $\mathcal{S}$ dont on sait par avance qu'ils ne pourront pas être solution, et la sélection à chaque étape du meilleur candidat possible *a priori*. Ils reposent uniquement sur les propriétés des distances. Rappellons que Δ est une telle distance métrique sur $\mathcal{X}$ si pour tous x, y, et z de $\mathcal{X}$:

$$\Delta(x, y) \geq 0 \text{ et } \Delta(x, y) = 0 \text{ ssi } x = y \tag{14.5}$$

$$\Delta(x, y) = \Delta(y, x) \tag{14.6}$$

$$\Delta(x, z) + \Delta(z, y) \geq \Delta(x, y) \tag{14.7}$$

Pour comprendre comment ces propriétés sont utilisées par l'algorithme lors de sa phase d'élimination, supposons qu'à une étape donnée le sous-ensemble U de $\mathcal{S}$ [2] est constitué par les éléments dont la distance à l'objet x a déjà été calculée. Si $s \in U$ est l'un de ces exemples, et si $pp \in U$ est pour le moment l'exemple le plus proche de x (i. e. $\Delta(pp, x) \leq \Delta(u, x)$ pour tout $u \in U$), alors l'inégalité triangulaire exprimée par l'équation 14.7 fournit les conditions nécessaires pour que les exemples p restant (ni éliminés, ni utilisés) soient plus proches de x que ne l'est pp (i. e. $\Delta(p, x) < \Delta(pp, x)$) :

$$\Delta(p, s) \leq \Delta(s, x) + \Delta(pp, x) \tag{14.8}$$

$$\Delta(p, s) \geq \Delta(s, x) - \Delta(pp, x) \tag{14.9}$$

1. **A**pproximating and **E**limination **S**earch **A**lgorithm.
2. $\mathcal{S}$, l'ensemble des exemples, est appelé « ensemble des prototypes » dans la terminologie originale de l'algorithme.

Prenons l'exemple de la figure 14.2. Soit x le point à classer. L'ensemble des exemples est en train d'être

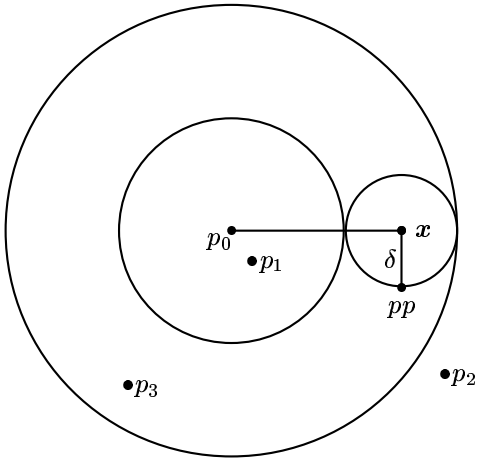


FIG. 14.2 – *Élimination d'exemples dans l'algorithme* AESA. *Le plus proche voisin courant de x est pp; il est situé à la distance δ. L'exemple suivant, p_0, n'est pas à situé à une distance inférieure à δ de x. Aucun point du type p_1 ou p_2 ne peut plus être le plus proche voisin de x. En revanche, il faudra calculer la distance $D(p_3, x)$.*

parcouru, et le plus proche voisin de x est pour le moment un certain point d'apprentissage pp situé à la distance $\Delta(pp, x) = \delta$. Soit p_0 l'exemple suivant. Si $\Delta(p_0, x) \leq \delta$, on actualise δ et pp. Sinon, il est possible d'affirmer que parmi tous les points d'apprentissage restant à examiner, deux catégories doivent être définitivement éliminées :

- ceux qui sont situés à l'intérieur du cercle de centre p_0 et de rayon $\Delta(p_0, x) - \delta$;
- ceux qui sont à l'extérieur du cercle de centre p_0 et de rayon $\Delta(p_0, x) + \delta$.

Le critère d'élimination de l'algorithme s'exprime alors par :

Critère d'élimination : Les exemples $p \in \mathcal{S}$ ne satisfaisant pas les inéquations 14.8 et 14.9 n'ont pas besoin d'être considérés pour le calcul de la distance avec x et peuvent être éliminés de $\mathcal{S}$, c'est-à-dire ajoutés à l'ensemble E des exemples éliminés[3].

Cette procédure peut être poursuivie en sélectionnant comme candidat au tour suivant l'un des exemples non encore éliminés, et en appliquant de nouveau ce critère pour éliminer un nouveau sous-ensemble de exemples. La condition d'arrêt est atteinte lorsque tous les exemples ont été utilisés ou éliminés.

En choisissant un candidat en suivant un ordre arbitraire initial[4], et en supposant une distribution aléatoire des exemples dans $\mathcal{S}$, cette procédure permet de réduire la quantité de calcul de distance par rapport à la recherche classique. Cependant, plus le candidat sélectionné est proche de l'objet x à approcher, plus le critère d'élimination est efficace. Afin de réduire encore le nombre de calculs de distance, il convient donc également de choisir dès les premières itérations les candidats les plus proches possibles de x. Aussi l'algorithme AESA propose-t-il une stratégie de sélection basée sur le critère d'approximation suivant :

Critère d'approximation : Sélectionner l'exemple s non éliminé ($s \in \mathcal{S} - E$) le plus proche de l'intersection de toutes les hypersphères[5] de rayons $\Delta(x, u)$, centrées sur chaque exemple u de l'ensemble U des exemples utilisés.

Bien que ni cette intersection[6] ni sa distance avec chacun des exemples ne puissent être obtenues explicitement, le critère d'approximation peut être approché numériquement en sélectionnant l'exemple $s \in \mathcal{S}$ qui minimise la somme des différences de distances entre s et les exemples utilisés et entre x et les

3. Il faut noter que $U \subseteq E$; les exemples effectivement utilisés sont également placés dans E.
4. Comme pour l'algorithme classique.
5. Projetées dans un espace à deux dimensions, les hypersphères considérées sont des cercles.
6. Quand suffisamment d'hypersphères sont impliquées, l'intersection est l'élément x cherché.

exemples utilisés :

$$s := \text{argmin}_{p \in \mathcal{S}-E} \left(\sum_{u \in U} |\Delta(p,u) - \Delta(x,u)| \right) \tag{14.10}$$

Ainsi, les principales étapes de l'algorithme AESA sont les suivantes :

1. **Initialisation**. L'ensemble U des exemples utilisés et l'ensemble E des exemples éliminés sont initialement vides.
2. **Approximation**. Sélectionner l'exemple s à l'aide du critère d'approximation (équation 14.10), et l'ajouter à U.
3. **Calcul de distance**. Calculer et stocker la distance entre s et l'élément de test x.
4. **Mise à jour du résultat courant**. Trouver le plus proche voisin parmi les exemples utilisés ($p \in U$).
5. **Élimination**. Éliminer les exemples ne répondant pas au critère d'élimination (équations 14.8 et 14.9), et les ajouter à E.
6. **Répéter** les points 2 à 5 jusqu'à ce que tous les exemples soient éliminés.

L'exemple le plus proche de x est au final l'exemple identifié comme tel au cours de la dernière itération de l'algorithme.

Algorithme détaillé Soit donc $\mathcal{X}$ l'espace de représentation, $\Delta : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ la fonction de calcul de distance entre deux individus de $\mathcal{X}$, et $\mathcal{S}$ l'ensemble des exemples. Soit encore x le nouvel objet dont le représentant le plus proche dans $\mathcal{S}$ est recherché.
Afin de profiter au maximum des capacités d'accélération de la recherche, une phase préliminaire de calcul est nécessaire pour évaluer les distances entre tous les couples (p_1, p_2) de exemples de $\mathcal{S}$. Soit Δ_P le tableau bidimensionnel contenant les résultats de ce calcul. Un deuxième tableau Δ_x (mono-dimensionnel) est introduit afin de stocker au fur et à mesure les distances entre les exemples de $\mathcal{S}$ utilisés et x.
Notons encore s l'exemple courant sélectionné à chaque étape de la recherche, U l'ensemble des exemples utilisés, E l'ensemble des exemples éliminés (les éléments de U et ceux éliminés sans être utilisés) et pp le meilleur exemple (le plus proche de x) rencontré jusqu'à présent.

PROCÉDURE Initialisation : déclaration et préparation des données pour la recherche principale
début
$c :=$ un premier exemple de $\mathcal{S}$ sélectionné de façon arbitraire ;
$s := c$;
$U := \{c\}$;
$pp := c$;
$\Delta_x[c] := \Delta(x,c)$;
$E := \{c\} \bigcup \{p \mid p \in \mathcal{S} - \{c\} : \Delta[p,c] \geq 2 \times \Delta_x[c]\}$;
fin procédure

La phase d'élimination initiale concerne les exemples dont la distance à c est deux fois plus grande que celle de c à x (qui ne seront donc jamais plus proches de x que ne l'est c).

PROCÉDURE Recherche AESA principale : boucle d'approximation et d'élimination sur les exemples jusqu'à trouver l'exemple le plus proche
début
exécute Initialisation ;
tant que $E \neq \mathcal{S}$ **faire**
$s := \text{argmin}_{p \in \mathcal{S}-E} \left(\sum_{u \in U} |\Delta_{\mathcal{S}}[p,u] - \Delta_x[u]| \right)$;
$\Delta_x[s] := \Delta(x,s)$;
$U := U \bigcup \{s\}$;
$E := E \bigcup \{s\}$;

$pp := \text{argmin}_{p \in \{pp, s\}} (\Delta_x[p])$;
si $pp = s$ **alors**
$Q := U$;
sinon
$Q := \{s\}$;
fin si
pour tous les $q \in Q$ **faire**
$E := E + \{p \mid p \in \mathcal{S} - E$ et $(\Delta_{\mathcal{S}}[p, q] \geq \Delta_x[q] + \Delta_x[pp]$
ou $\Delta_{\mathcal{S}}[p, q] \leq \Delta_x[q] - \Delta_x[pp])\}$;
fin pour
fin tant que
retourne pp ;
fin procédure

La boucle principale de l'algorithme sélectionne à chaque étape un exemple parmi ceux qui restent en suivant le critère d'approximation (équation 14.10). L'exemple pp est mis à jour, et un ensemble Q correspondant à l'ensemble U est créé en conséquence pour la phase d'élimination. Si l'exemple s sélectionné est le nouveau meilleur exemple, Q est affecté à U, et tous les exemples utilisés entrent normalement en jeu. En revanche, si l'exemple s n'est pas meilleur que pp, Q est uniquement affecté au singleton $\{s\}$. Puisque pp est inchangé, il est en effet inutile de procéder à une nouvelle élimination en utilisant les éléments de U qui ont déjà servi à l'étape précédente. Seul l'utilisation d'un nouvel exemple peut donc permettre d'éliminer des éléments à cette étape.
Le processus de recherche se termine lorsque tous les exemples ont été utilisés ou éliminés. L'exemple le plus proche de x est alors pp et la distance correspondante est $\Delta_x[pp]$.

Exemple Plaçons nous dans $\mathbb{R}^2$, et considérons l'ensemble d'apprentissage représenté par la table 14.1. La demi-matrice des distances entre les exemples est calculée dans un premier temps. La table 14.2 présente le carré de ces distances. Soit le point $x = (4, 4)$ dont on cherche le plus proche voisin dans $\mathcal{S}$.

$\mathcal{S}$	abs_{p_i}	ord_{p_i}
p_1	7	5
p_2	12	12
p_3	15	6
p_4	5	10
p_5	2	11
p_6	3	4
p_7	8	3
p_8	5	-3
p_9	8	2
p_{10}	2	8
p_{11}	6	-1
p_{12}	12	6
p_{13}	14	1

TAB. 14.1 – *Coordonnées des points de l'ensemble d'apprentissage.*

Nous allons commencer de dérouler l'algorithme sur ces données. Dans la figure suivante, les exemples sont initialement représentés par un point noir ($\bullet p_i$) et le meilleur candidat à l'instant courant est noté par un point blanc ($\circ p_i$). Les exemples éliminés à l'étape courante sont rayés ($\bullet \not{p_i}$), tandis que ceux éliminés aux étapes précédentes apparaissent grisés ($\bullet p_i$).
La première étape de l'algorithme est la suivante :

1. **Sélection.** L'exemple p_1 est (arbitrairement) sélectionné en premier, et p_1 devient le plus proche voisin courant : $\Delta_x[p_1] = \sqrt{10}$, $pp = p_1$, $U = \{p_1\}$.

$\mathcal{S}$	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}	p_{11}	p_{12}	p_{13}
p_1	0	97	10	29	61	17	5	68	65	34	37	26	65
p_2	.	0	153	52	90	164	130	325	80	117	250	65	178
p_3	.	.	0	73	117	29	1	34	65	72	13	32	37
p_4	.	.	.	0	10	40	58	169	116	13	122	65	162
p_5	.	.	.	.	0	50	100	205	194	9	160	125	244
p_6	.	.	.	.	.	0	26	53	148	17	34	85	137
p_7	.	.	.	.	.	.	0	45	58	61	20	25	40
p_8	.	.	.	.	.	.	.	0	181	130	5	130	97
p_9	.	.	.	.	.	.	.	.	0	173	130	9	26
p_{10}	.	.	.	.	.	.	.	.	.	0	97	104	193
p_{11}	.	.	.	.	.	.	.	.	.	.	0	85	68
p_{12}	.	.	.	.	.	.	.	.	.	.	.	0	29
p_{13}	.	.	.	.	.	.	.	.	.	.	.	.	0

TAB. 14.2 – *Carré des distances entre les points de l'ensemble d'apprentissage.*

2. **Élimination**. Les exemples à l'extérieur du cercle de centre p_1 et de rayon $2\Delta_x[p_1]$ sont éliminés : les points p_2, p_3, p_5, p_8 et p_{13} rejoignent l'ensemble E. $E = \{p_1,\ p_2,\ p_3,\ p_5,\ p_8,\ p_{13}\}$. La figure 14.3 présente cette élimination.

Question
Donner les étapes suivantes. Combien de distances calcule-t'on ?

Raffinements Il existe des raffinements de cet algorithme, notamment pour limiter les pré-calculs de distance entre tous les exemples disponibles. L'algorithme LAESA [Micó et al., 1994] divise l'ensemble d'apprentissage en sous-ensembles par classification automatique[7], chacun étant rassemblé sous la bannière d'un exemple particulier. Ces exemples sont sélectionnés de façon à ce qu'ils soient le plus uniformément répartis sur l'ensemble d'apprentissage. La recherche du plus proche voisin d'un nouvel exemple s'effectue alors dans un premier temps parmi ces représentants, et dans un second temps à l'intérieur du sous-ensemble d'exemples gouvernés par le représentant sélectionné. Cela nécessite alors uniquement le calcul des distances croisées entre les représentants et les autres exemples, et entre exemples d'un même sous-ensemble. Le calcul des distances entre exemples de sous-ensembles différents n'est alors plus nécessaire, ce qui permet ainsi une baisse substentielle des coûts de pré-calcul, tant en durée qu'en quantité de mémoire nécessaire.

ACLM

Exercice 14.15  **Décision avec coût**

On suppose que l'on dispose d'un paramètre x obtenu par un examen médical qui permet de détecter la présence (classe ω_1) ou l'absence (classe ω_2) d'une maladie. Une valeur de x faible est plutôt signe de la maladie tandis qu'une valeur de x élevée est plutôt signe de son absence. On considère que de ne pas détecter la maladie sur un patient atteint est plus grave que de diagnostiquer la maladie sur un patient sain. On distingue donc deux types d'erreurs définies dans le tableau ci dessous :

classe réelle :	ω_1	ω_2
décision ω_1	décision correcte	erreur de type II
décision ω_2	erreur de type I	décision correcte

En cas d'erreur de diagnostic, on distingue les probabilités suivantes :

$$P_{ND} = \text{probabilité de fausse alarme} = P(\text{décider}(\omega_1, \omega_2)$$

$$P_{FA} = \text{probabilité de non détection} = P(\text{décider}(\omega_2, \omega_1)$$

7. Ou *clustering* en anglais. Voir le chapitre 15.

Le traitement médical étant coûteux, il faut faire un compromis pour maintenir le taux de fausse alarme à un niveau raisonnable.

Question 1

Pour modéliser le problème , on introduit deux coûts : $\lambda(\omega_2|\omega_1)$, le coût de non détection et $\lambda(\omega_1|\omega_2)$, le coût de fausse alarme

Donner l'expression du risque conditionnel associé à chaque décision. Donner la règle de décision bayésienne en utilisant le rapport de vraisemblance $P(x|\omega_1)/P(x|\omega_2)$

Lequel des deux coûts λ doit être supérieur à l'autre pour ce problème?

Question 2

On fixe les valeurs des coûts $\lambda(\omega_2|\omega_1)$ et $\lambda(\omega_1|\omega_2)$. On exprime la règle de décision bayésienne en fonction du paramètre x et d'un seuil x_0 tel que :

si $x < x_0$ on décide ω_1

si $x > x_0$ on décide ω_2

Donner les expressions des probabilités de fausse alarme (P_{FA}) et de non détection (P_{ND}) en fonction des lois de densité conditionnelles $p(x|\omega_i)$, $i = 1, 2$ et du seuil x_0. Comment évoluent ces probabilités en fonction de x_0 ?

Question 3

On considère maintenant que les lois sont gaussiennes de même variance :

$$p(x|\omega_1) = \mathcal{N}(m_1, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-m_1)^2}{2\sigma^2}}$$

$$p(x|\omega_2) = \mathcal{N}(m_2, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-m_2)^2}{2\sigma^2}}$$

En déduire l'expression du point frontière x_0 en fonction de m_1, m_2, σ et des probabilités *a priori*. Donner les expressions de P_{FA}, P_{ND} en fonction de σ, x_0 en utilisant la fonction $erfc$ définie par

$$erfc(x) = \frac{2}{\pi} \int_x^{+\infty} e^{-t^2} dt$$

Application numérique

Prendre $\sigma = 0.5$, l'un des coûts égal à 1 , l'autre à 4 (cf. question 2) et calculer x_0.

Question 4

On représente $P_{ND}/P(\omega_1)$ en fonction de $P_{FA}/P(\omega_2)$ pour différentes valeurs du point frontière x_0. La courbe obtenue est appelée courbe *ROC*. Donner son allure.

On cherche généralement à obtenir une probabilité de non détection faible en acceptant un taux de fausse alarme raisonnable. Dans quelle partie de la courbe *ROC* se situe x_0 ? Comment varie x_0 le long de la courbe *ROC*? Peut-on rendre aussi faibles que possible à la fois P_{ND} et P_{FA} ?

LL

Exercice 14.16  Apprentissage par PPV pour des données nominales.

Soient des objets décrits par deux attributs nominaux, leur *couleur* et leur *forme*. Le premier attribut peut prendre une valeur dans l'ensemble $\{Bleu, Vert, Jaune\}$ et le second une valeur dans l'ensemble $\{Cercle, Rectangle, Ovale, Losange\}$.

On écrit les éléments de l'ensemble d'apprentissage de manière raccourcie : $((V, O), +)$ signifie que l'objet décrit par les attributs ($couleur = Vert$) et ($forme = Ovale$) appartient au concept (est un exemple positif) et $((J, C), -)$ signifie que l'objet décrit par les attributs ($couleur = Jaune$) et ($forme = Cercle$) n'appartient pas au concept (est un contre-exemple).

Soit l'ensemble d'apprentissage :

1. ((V, O), +)
2. ((V, R), +)
3. ((B, L), +)
4. ((B, R), -)

5. ((J, L), -)
6. ((J, C), -)

Question 1 Transformer la description des objets en les représentant par des attributs binaires.
Montrer que chaque exemple a désormais exactement deux attributs valant VRAI et cinq valant FAUX.
Construire un arbre de décision sur ces attributs binaires. Il y a plusieurs solutions équivalentes. Pourquoi ?
NB : il n'y a pas besoin de faire de calculs de logarithmes ; il suffit de raisonner en comparant les tableaux (2×2) qui caractérisent chacun la distribution conjointe d'un attribut et du concept sur les données d'apprentissage.
Comment sont classés les objets (V, C) et (J, R) par cet arbre ?

Question 2 On reprend la description nominale des objets. On définit la distance entre deux objets comme la somme sur l'ensemble des deux attributs des "distances par attribut". Dans cette question, la valeur de la distance par attribut entre deux objets vaut simplement 0 si que ces objets ont la même valeur pour cet attribut, 1 sinon.
Comment sont classés l'objets (J, R) sur l'ensemble d'apprentissage par la règle du K-PPV pour $K = 4$?
Et (V, C) pour $K = 3$?
Comment prendre une décision pour (J, R) pour $K = 3$?

Question 3 On va définir une distance par attribut moins triviale que la précédente. Pour chaque attribut, on calcule à partir de l'ensemble d'apprentissage une matrice de distances pour affecter à tout couple de valeurs d'un attribut une distance non binaire. La distance entre deux objets sera la somme sur tous les attributs de ces distances par attributs.
Pour un attribut pouvant prendre les valeurs X et Y, par exemple, on définira la distance entre ces deux valeurs comme :

$$\left|\frac{N(X,+)}{N(X)} - \frac{N(Y,+)}{N(Y)}\right| + \left|\frac{N(X,-)}{N(X)} - \frac{N(Y,-)}{N(Y)}\right|$$

où :

- N(X,+) est le nombre de fois que, dans l'ensemble d'apprentissage, un exemple positif prend la valeur X sur l'attribut considéré,
- N(X,-) est le nombre de fois que, dans l'ensemble d'apprentissage, un contre-exemple prend la valeur X sur l'attribut considéré,
- N(X) est le nombre de fois qu'un exemple ou un contre-exemple prend cette valeur.

Par exemple, pour l'attribut *Couleur*, la distance entre les valeurs *Bleu* et *Vert* se calcule comme :

$$\delta_{Couleur}(B, V) = \left|\frac{1}{2} - \frac{2}{2}\right| + \left|\frac{1}{2} - \frac{0}{2}\right| = 1$$

Calculer les matrices de distances pour les deux attributs.
Avec cette nouvelle distance, comment sont classés les objets (V, C) et (J, R) sur l'ensemble d'apprentissage par la règle du K-PPV pour $K = 1$?

ACLM

Quelques sites internet recommandés

Classification Toolbox for Matlab
`http://tiger.technion.ac.il/~eladyt/classification/index.htm`

Chapitre 15

La classification non supervisée et la découverte automatique

Exercice 15.1  **Protocoles d'apprentisage non supervisé**

On peut réaliser de l'apprentissage non supervisé de manière incrémentale ou de manière « batch ». Quelles sont selon vous les conséquences de ces deux choix?

ACLM

Quelques sites internet recommandés

Classification Toolbox for Matlab
`http://tiger.technion.ac.il/~eladyt/classification/index.htm`

Bibliographie

[de la Higuera, 1997] de la Higuera, C. (1997). Characteristic sets for polynomial grammatical inference. *Machine Learning*, 27:125–138.

[Duda et al., 2001] Duda, R., Hart, P., and Stork, D. (2001). *Pattern classification.* Wiley.

[Gold, 1967] Gold, M. (1967). Language identification in the limit. *Information and Control*, 10:447–474.

[Gold, 1978] Gold, M. (1978). Complexity of automaton identification from given data. *Information and Control*, 37:302–320.

[Mari and Schott, 2001] Mari, J.-F. and Schott, R. (2001). *Probabilistic methods in computer science.* Kluwer.

[Micó et al., 1994] Micó, L., Oncina, J., and Vidal, E. (1994). A new version of the nearest-neighbour approximating and eliminating search algorithm (aesa) with linear preprocessing-time and memory requirements. *Pattern Recognition Letters*, 15:9–17.

[Mitchell, 1997] Mitchell, T. (1997). *Machine Learning.* McGraw Hill.

[Moore, 2002] Moore, A. (2002). `http://www-2.cs.cmu.edu/~awm/`.

[Nilsson, 1998] Nilsson, N. (1998). *Artificial Intelligence : a new synthesis.* Morgan-Kaufmann.

[Oncina and Garcia, 1992] Oncina, J. and Garcia, P. (1992). *Identifying regular languages in polynomial time.*, volume 5, pages 99–108. World Scientific, Singapore.

[Oncina and García, 1992] Oncina, J. and García, P. (1992). Inferring regular languages in polynomial updated time. In *Pattern Recognition and Image Analysis : Selected papers from the IVth Spanish Symposium*, pages 49–61. World Scientific.

[Rich and Knight, 1997] Rich, E. and Knight, K. (1997). *Artificial Intelligence (second edition).* McGraw Hill.

[Russel and Norvig, 1995] Russel, S. and Norvig, P. (1995). *Artificial Intelligence : a modern approach.* Prentice-Hall.

[Vidal, 1994] Vidal, E. (1994). New formulation and improvements of the nearest-neighbour approximation and elimating search algorithm (asea). *Pattern Recognition Letters*, pages 1–7.