Algorithmes d'apprentissage

Agents qui apprennent à partir d'exemples

- La problématique : prise de décision automatisée à partir d'un ensemble d'exemples
  - Diagnostic médical
  - Réponse à une demande de prêt bancaire
- Méthodes symboliques (on produit des règles)
  - Arbres de décision
- Méthodes non symboliques ou adaptatives
  - Réseaux de neurones
  - Algorithmes génétiques

L'approche probabiliste

Exemple : Établir un diagnostic dans le domaine médical.

- Il faut être capable d'associer le nom d'une maladie à un certain nombre de symptômes présentés par des malades
- Les malades forment la population.
- Les symptômes sont les descriptions des malades.
- Les maladies sont les classes.
- On suppose qu'il y a un classement correct, c.-à-d. une application qui associe à tout malade une maladie.
- Apprendre à établir un diagnostic : associer une maladie à une liste de symptômes

Définition

- Π : la population
- D : l'ensemble des descriptions
- $K = \{1, \ldots, c\}$ l'ensemble des classes
- $X : \Pi \to D$: fonction qui associe une description à chaque élément de la population
- $Y : \Pi \to \{1, \ldots, c\}$: fonction de classement qui associe une classe à tout élément de la population.
- Une fonction $C : D \to \{1, \ldots, c\}$ est appelée fonction de classement ou procédure de classification

Le but de l'apprentissage est de rechercher une procédure de classification $C : D \to K$ telle que $C \circ X \sim Y$ ($C \circ X$ doit être une bonne approximation de Y).

Dans la pratique...

- Ensembles d'attributs logiques, symboliques ou numériques $A_1 \in D_1, \ldots A_n \in D_n$.
- $D = D_1 \times D_2 \ldots \times D_n$ l'espace de descriptions.
- Par exemple, un patient est décrit par un ensemble de symptômes (mal de tête, douleur) et une suite de mesures (tension, température).

Bonne approximation

Comment exprimer que $C \circ X$ est une bonne approximation de Y ?

- On suppose l'existence d'une distribution de probabilités sur Π et on dit que $C \circ X$ est une bonne approximation de Y, s'il est peu probable qu'elles diffèrent.
- Soient Π probabilisé, P la probabilité définie sur Π et D discret.
  - $P(d)$: probabilité qu'un élément de Π ait d pour description.
  - $P(k)$: probabilité qu'un élément de Π soit de classe k.
  - $P(d/k)$: probabilité qu'un élément de classe k ait d pour description.
  - $P(k/d)$: probabilité qu'un élément ayant d pour description soit de classe k.

– Formule de Bayes :

$$P(k/d) = \frac{P(d/k)P(k)}{P(d)}$$

Si on connaît $P(d)$, $P(k)$ et $P(d/k)$ pour tout $d \in D$ et pour toute classe $k \in \{1, \ldots, c\}$. Comment choisir la fonction C ?

Exemple

- Π : la population française
- Un attribut logique : répondeur
- L'espace de description : $\{repondeur, \overline{repondeur}\}$
- Deux classes : $\{riche, \overline{riche}\}$
- Informations :

classe k	$riche$	$\overline{riche}$
$P(k)$	0.4	0.6
$P(repondeur/k)$	0.8	0.45
$P(\overline{repondeur}/k)$	0.2	0.55

Règle de la classe majoritaire

La fonction C_{maj} attribue à chaque description la classe majoritaire, c'est à dire la classe k t.q. $P(k)$ est maximum (dans l'exemple $\overline{riche}$).

Règle du maximum de vraisemblance

- $C_{vraisemblance}$ attribue à une description d la classe pour laquelle $P(d/k)$ est maximum (la plus probable).
- Dans l'exemple : $riche$ pour propriétaire de répondeur, $\overline{riche}$ pour les autres.
- Problème : supposons
  $C = \{employe\ Telecoms, medecins, ouvriers\}$ et
  $P(repondeur/employe\ Telecoms) = 1$.

  $C_{vraisemblance}$ donne pour propriétaire d'un répondeur toujours $employe\ Telecoms$.

Règle de Bayes

- C_{Bayes} attribue à une description d la classe k qui maximise la probabilité $P(k/d)$ qu'un élément ayant d pour description soit de classe k.
- $P(k/d)$ peut être estimée en utilisant la formule de Bayes, donc on choisit la classe k qui maximise le produit $P(d/k)P(k)$.
- Exemple :

$$
\begin{array}{lcll}
P(repondeur/riche)P(riche) & = & 0.8 \times 0.4 & = & 0.32 \\
P(repondeur/\overline{riche})P(\overline{riche}) & = & 0.45 \times 0.6 & = & 0.27 \\
P(\overline{repondeur}/riche)P(riche) & = & 0.2 \times 0.4 & = & 0.08 \\
P(\overline{repondeur}/\overline{riche})P(\overline{riche}) & = & 0.55 \times 0.6 & = & 0.33
\end{array}
$$

- Ici : $C_{vraisemblance} = C_{Bayes}$
- Cas spécial : classes équiprobables

Comparaison

- L'erreur pour une description $E(d)$ est la probabilité qu'un élément ayant description d soit mal classé par C, i.e. $E(d) = P(\{k|k \neq C(d)\}/d)$
- L'erreur de classification est

$$E(C) = \sum_{d \in D} E(d)P(d)$$

Erreurs dans l'exemple

$E(C_{maj}) = 0.4$

$E(C_{vraisemblance}) =_{def}$

$E(rep)P(rep) + E(\overline{rep})P(\overline{rep}) =_{def}$

$P(\overline{riche}/rep)P(rep) + P(riche/\overline{rep})P(\overline{rep}) =_{def}$

$\frac{P(rep/\overline{riche})P(\overline{riche})P(rep)}{P(rep)} + \frac{P(\overline{rep}/riche)P(riche)P(\overline{rep})}{P(\overline{rep})} =$

$P(rep/\overline{riche})P(\overline{riche}) + P(\overline{rep}/riche)P(riche) =$

$0.27 + 0.08 = 0.35$

Erreur minimale

Théorème : La règle de Bayes est celle dont l'erreur de classification est minimale.

Remarques

- Si une fonction de classement est correcte, alors $E(C_{Bayes}) = 0$
- Une fonction de classement correcte existe, ssi la probabilité que des individus appartenant à des classes différents aient des descriptions identiques est nulle.
- Dans ce cas, le problème est dit déterministe.
- Très rare en pratique :
  - Généralement les paramètres descriptifs dont on dispose ne sont pas suffisants pour classifier correctement tout.
  - Les données sont généralement inexactes.
- Il faut connaître les probabilités (difficiles à estimer)

Types de classification

- Apprentissage supervisé/non supervisé
- Choix du langage de description en fonction de la définition des attributs susceptibles d'être pertinents

On s'intéresse à l'apprentissage supervisé par un ensemble d'exemples déjà classés.

Difficultés : les lois de probabilité sont inconnues de l'apprenant, l'espace des fonctions de classement a une taille démesurée, l'échantillon disponible est de taille limitée.

Classification supervisée

- Langage de description $D = D_1 \times D_2 \times \ldots D_n$ (on note $\vec{x}, \vec{y} \in D$) et ensemble de classes $\{1, \ldots, c\}$.
- On suppose une loi de probabilité P sur D fixée mais inconnue.
- On suppose une loi de probabilité conditionnelle $P(./.)$ fixée mais inconnue.
- Échantillon S de m exemples $(\vec{x}, C(\vec{x}))$ tirés selon $P(.,.)$ définie par $P(\vec{x}, y) = P(\vec{x})P(y/\vec{x})$.
- Problème de classification supervisée : Inférer une fonction de classement dont l'erreur de classification est minimale.

Bien classer et bien prédire

– Exemple d'une procédure de classification :

1. On mémorise tous les exemples de l'échantillon d'apprentissage dans une table.
2. Lorsqu'une nouvelle description est présentée au système, on recherche dans la table.
3. Si on trouve la description, on sort le résultat correspond
4. Sinon, on choisit une classe au hasard.

– Cette procédure ne fait pas d'erreur sur les exemples, mais son pouvoir prédictif est très faible

Objectif : Une procédure de classification devrait dépasser au moins le pouvoir prédictif de la procédure C_{maj} majoritaire.

Erreur réelle et apparente

- L'erreur réelle $E(C)$ est l'erreur de classification
- L'erreur apparente $E_{app}(C)$ est définie par :
  - S un échantillon de taille m
  - C une procédure de classification
  - $E_{app}(C) = \frac{err}{m}$ où err est le nombre d'exemples de S mal classés par C.

Objectif : minimiser $E(C)$ (mais l'apprenant ne connaît que $E_{app}(C)$ sur S).

Remarque : $E_{app}(C)$ tend vers $E(C)$ pour des échantillons de plus en plus grands.

Les méthodes de classification supervisée

- Le classifieur naïf de Bayes
- Minimiser l'erreur apparente
- Choix de l'espace des hypothèses
- Estimer l'erreur réelle
  - Utilisation d'un ensemble Test
  - Re-échantillonage

Le classifieur naïf de Bayes

- $C_{Bayes}(d) = \underset{k\in\{1,\ldots,c\}}{argmax}\ P(k/d) = \underset{k\in\{1,\ldots,c\}}{argmax}\ P(d/k)P(k)$
- $C_{Bayes}(\vec{d}) = \underset{k\in\{1,\ldots,c\}}{argmax}\ P((d_1,\ldots,d_n)/k)P(k)$
- $P(d/k)$, $P((d_1,\ldots,d_n)/k)$ et $P(k)$ ne sont pas connues.
- On estime $P(k)$ par $\widehat{P}(k)$ (% d'éléments de classe k dans S).
- L'estimation de $P((d_1,\ldots,d_n)/k)$ est difficile, donc on fait une hypothèse simplificatrice qui dit que les valeurs des attributs sont indépendants connaissant la classe :

$$P((d_1,\ldots,d_n)/k) = \Pi_{i\in\{1,\ldots,n\}} P(d_i/k)$$

- On estime $P(d_i/k)$ par $\widehat{P}(d_i/k)$ (la proportion d'éléments de classe k ayant la valeur d_i pour l'attribut i).

$$C_{NaiveBayes}(d) = \underset{k\in\{1,\ldots,c\}}{argmax}\ \Pi_{i\in\{1,\ldots,n\}}\widehat{P}(d_i/k) \times \widehat{P}(k)$$

Minimiser l'erreur apparente

- L'erreur apparente est une version très optimiste de l'erreur réelle
- Il y a beaucoup de fonctions de D dans $\{1, \ldots, c\}$. C'est impossible de toutes les explorer.
- On peut donc limiter la recherche d'une fonction de classification à un espace donné par un sous-ensemble $\mathcal{C}$ de toutes les procédures de classification.
- Exemples : programme qui traduit un texte de 200 pages de l'anglais vers le français et comporte 400 pages de code... On prefère choisir le traducteur dans un ensemble restreint de programmes (ceux de moins de 20 pages de code).

Minimiser l'erreur apparente

- On prend un espace d'hypothèses $\mathcal{C}$
- Soit C_{opt} la procédure optimale de $\mathcal{C}$ (celle qui a l'erreur de classification minimale).
- Problème difficile : approcher C_{opt}
- On ne peut pas à la fois sélectionner un classifieur à l'aide d'un ensemble d'apprentissage et juger sa qualité avec le même ensemble.
- On choisira C_{emp} qui minimise l'erreur apparente.
- Il faut bien choisir l'espace de recherche d'hypothèses :
  par exemple on a des suites d'ensembles de procédures de classification $\mathcal{C}_1 \subseteq \mathcal{C}_2 \subseteq \cdots \mathcal{C}_k \subseteq \cdots$
  où k est une mesure de complexité du système d'apprentissage

liée à la capacité.
On essaie de trouver k de sorte que $C_{k,emp}$ ait le plus faible erreur réelle possible. On peut minimiser l'erreur apparente en complexifiant de plus en plus l'espace de recherche.

Estimer l'erreur réelle

- Utilisation d'un ensemble Test : on partitionne l'échantillon en un ensemble d'apprentissage S et un ensemble test T
- La répartition est faite aléatoirement.
- On génère une procédure de classification C en utilisant S.
- On estime l'erreur réelle $E(C)$ avec

$$\widehat{E}(C) = \frac{\sharp malclasses(T)}{\sharp T}$$

- L'estimation est faire sur un ensemble indépendant de celui qui a servi à l'aprentissage
- Méthode bien adaptée lorsque la taille de S est suffisamment grande

Estimer l'erreur réelle

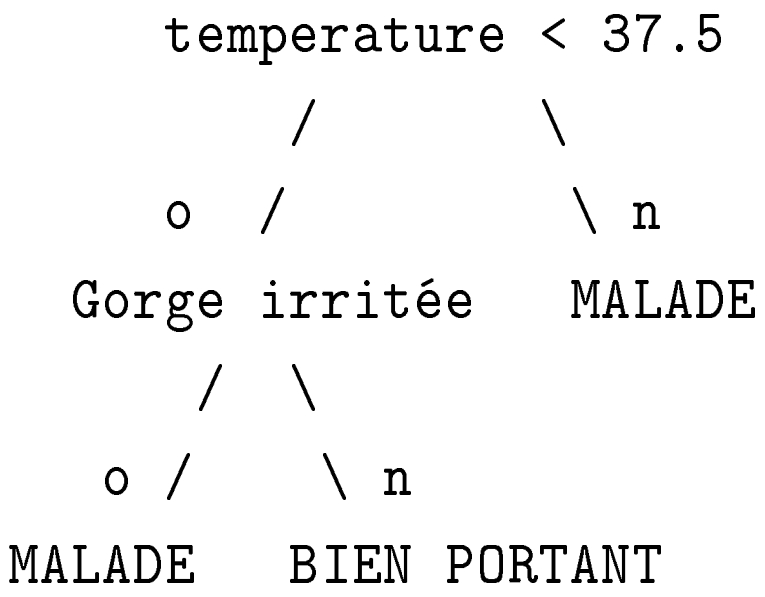
Lorsque l'échantillon S est trop petit on ne peut pas sacrifier des éléments pour tester le classifieur.

On utilise des techniques de re-échantillonage.

On divise l'échantillon S en k parties. On fait k sessions d'apprentissage en utilisant à chaque fois $k-1$ parties pour apprendre et une partie pour tester.

Les arbres de décision

- C'est une méthode symbolique
- On cherche une procédures de classification compréhensible
- Exemple :



Les arbres de décision

- Un arbre de décision est un arbre au sens informatique
- Les noeuds sont repérés par des positions $\in \{1, \ldots, p\}^*$, où p est l'arité maximale des noeuds.
- Les noeuds internes sont les noeuds de décision.
- Un noeud de décision est étiqueté par un test qui peut être appliqué à chaque description d'un individu d'une population.
- Chaque test examine la valeur d'un unique attribut.
- Dans les arbres de décision binaires on omet les labels des arcs.
- Les feuilles sont étiquetées par une classe.

Les arbres de décision

- À chaque arbre, on associe naturellement une procédure de classification.
- À chaque description complète est associée une seule feuille de l'arbre.
- La procédure de classification représenté par un arbre correspond à des règles de décision.
- Exemple :

  Si Température $<$ 37,5 ET gorge irritée ALORS malade

  Si Température $<$ 37,5 ET gorge non irritée ALORS bien portant

  Si Température $\geq$ 37,5 ALORS malade

Construire des arbres de décision

- Étant donné un échantillon S et des classes $\{1, \ldots, c\}$, on veut construire un arbre t
- À chaque position p de t correspond un sous-ensemble de S qui contient les éléments de S qui satisfont les tests de la racine jusqu'à p.
- On définit pour chaque p :
  - $N(p)$ = le cardinal de l'ensemble des exemples associé à p
  - $N(k/p)$ = le cardinal de l'ensemble des exemples associé à p de classe k
  - $P(k/p) = N(k/p)/N(p)$ = la proportion d'éléments de classe k à la position p

Revenons à l'exemple

- Échantillon de 200 patients : 100 malades et 100 bien portants.
- Répartition (M malades, S bien portants) :

	gorge irrité	gorge non irrité
température $< 37{,}5$	(6 S, 37 M)	(91 S, 1 M)
température $\geq 37{,}5$	(2 S, 21 M)	(1 S, 41 M)

- $N(11) = 43$,
- $N(S/11) = 6$, $N(M/11) = 37$,
- $P(S/11) = \frac{6}{43}$, $P(M/11) = \frac{37}{43}$

Exemple : Banque

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

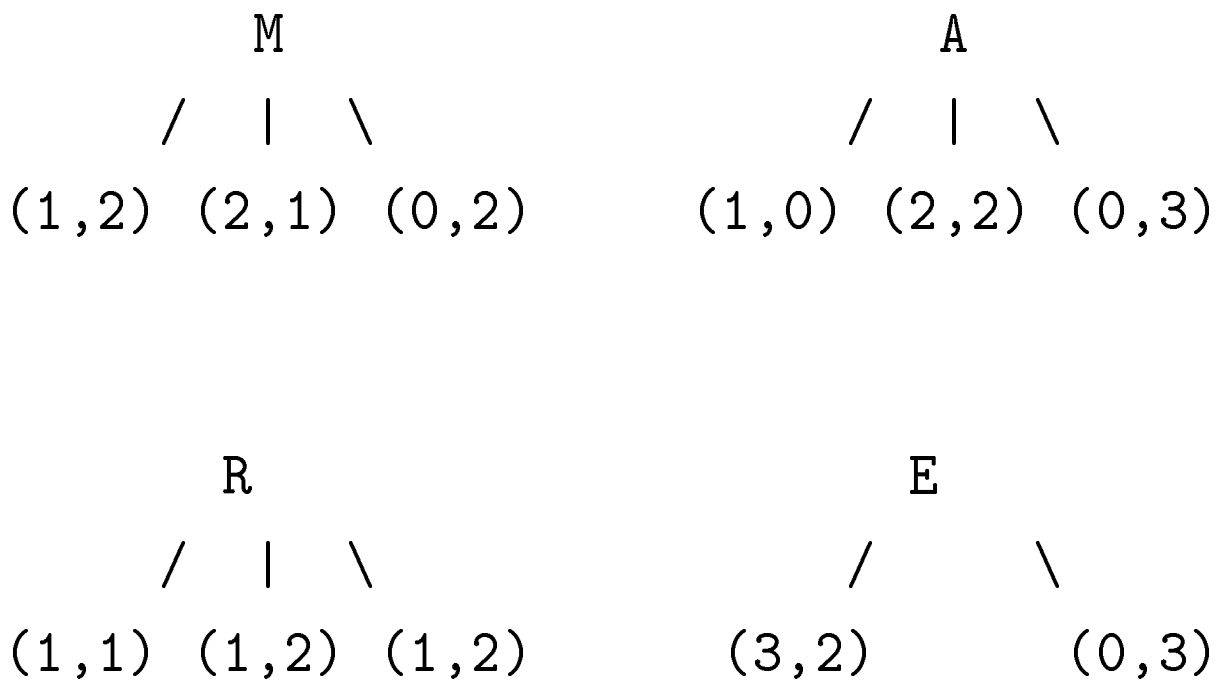
Les attributs

- M : La moyenne des montants sur le compte
- A : Tranche d'âge du client
- R : Localité de résidence du client
- E : Études supérieures
- I : Consultation du compte par Internet
- On veut construire un arbre de décision pour savoir si un client consulte son compte par Internet.

Exemple de la Banque (suite)

- On construit l'arbre d'une façon descendante
- On choisit un test, on divise l'ensemble d'apprentissage S et on réapplique récursivement l'algorithme.
- On initialise avec l'arbre vide
- Des 8 éléments de S, 3 sont de classe oui et 5 de classe non.
- La racine de l'arbre n'est étiquetée par aucun test mais elle est caractérisée par $(3, 5)$.
- Si le noeud n'est pas terminal déjà, alors on choisit un test.
- Ici il y a 4 choix (M,A,R,E)

Les 4 choix



Quel test choisir ? Intuitivement `A est un bon test`

Mesurer le degré de mélange des exemples

- Un test est intéressant s'il permet une bonne discrimination
- Une fonction qui mesure le degré de mélange des exemples doit
  - prendre son maximum lorsque les exemples sont équirépartis (ici par exemple (4,4)).
  - prendre son minimum lorsque les exemples sont dans une même classe ((0,8) ou (8,0)).
- On veut une fonction qui minimise le degré de mélange

Des fonctions pour mesurer le degré de mélange

Les fonctions suivantes minimisent le degré de mélange

$$Entropie(p) = -\sum_{k=1}^{c} P(k/p) \times log(P(k/p))$$

$$\begin{aligned} Gini(p) &= 1 - \sum_{k=1}^{c} P(k/p)^2 \\ &= 2 \sum_{k<k'} P(k/p)P(k'/p) \end{aligned}$$

Exemple pour l'attribut E

- $Entropie(\epsilon) = -\frac{3}{8}log\frac{3}{8} - \frac{5}{8}log\frac{5}{8} \simeq 0,954$
- $Entropie(1) = -\frac{3}{5}log\frac{3}{5} - \frac{2}{5}log\frac{2}{5} \simeq 0,970$
- $Entropie(2) = -\frac{0}{3}log\frac{0}{3} - \frac{3}{3}log\frac{3}{3} = 0$
- $Gini(\epsilon) = 2 \times \frac{3}{8} \times \frac{5}{8} \simeq 0,469$
- $Gini(1) = 2 \times \frac{3}{5} \times \frac{2}{5} = 0,480$
- $Gini(2) = 2 \times \frac{0}{3} \times \frac{3}{3} = 0$

Choisir un test

- Soit f la fonction choisie (Gini ou Entropie par exemple)
- On définit le gain pour un test T et un position p

$$Gain(p,T) = f(p) - \sum_{j=1}^{n}(P_j \times f(pj))$$

où n est l'arité du test T et P_j la proportion d'éléments de S à la position p qui vont en position pj (qui satisfont la i-ème branche du Test)

- On cherche pour chaque position p le gain maximum

Exemple

- $Gain(\epsilon, M) = Entropie(\epsilon)$
  $-(\frac{3}{8}Entropie(1) + \frac{3}{8}Entropie(2) + \frac{2}{8}Entropie(3)) =$
  $Entropie(\epsilon) - 0,620 = 0.334$
- $Gain(\epsilon, A) = Entropie(\epsilon)$
  $-(\frac{1}{8}Entropie(1) + \frac{4}{8}Entropie(2) + \frac{3}{8}Entropie(3)) =$
  $Entropie(\epsilon) - 0,500 = 0.454$
- $Gain(\epsilon, R) = Entropie(\epsilon)$
  $-(\frac{2}{8}Entropie(1) + \frac{3}{8}Entropie(2) + \frac{3}{8}Entropie(3)) =$
  $Entropie(\epsilon) - 0,870 = 0.084$
- $Gain(\epsilon, E) = Entropie(\epsilon)$
  $-(\frac{5}{8}Entropie(1) + \frac{3}{8}Entropie(2)) =$
  $Entropie(\epsilon) - 0,607 = 0.347.$

Généralités

- Idée : Diviser récursivement et le plus efficacement possible les exemples de l'ensemble d'apprentissage par des tests définis à l'aide d'attributs, jusqu'à ce qu'on obtienne des sous-ensemble ne contenant (presque) que des exemples appartenant à une même classe.
- On a besoin des trois opérations :
  - Décider si un noeud est terminal.
  - Sélectionner un test à associer à un noeud.
  - Affecter une classe à une feuille.

Algorithme générique

```
entrée: langage de description, échantillon S
sortie: un arbre de décision
Début
   Initialiser à l'arbre vide
   répéter
      Si noeud courant  est terminal
         alors Affecter une classe
         sinon Selectionner test et créer sous-arbres
      Passer au noeud suivant non-exploré
   jusqu'à obtenir un arbre de décision
Fin
```

Généralités

- Arbre de décision parfait (tous les exemples sont bien classifiés) n'existe pas toujours.
- Le meilleur arbre est l'arbre le plus petit parfait.
- L'algorithme précédent ne remet jamais en cause un choix effectué.
- L'erreur réelle peut être importante.
- En pratique, on construit l'arbre et ensuite on élague.
- Deux algorithmes : CART et C4.5

L'algorithme CART

- Génère un arbre de décision binaire
- Langage de représentation : On suppose prédéfini un ensemble de tests binaires.
- Nous disposons d'un échantillon S découpé en un ensemble d'apprentissage A et un ensemble de test T.

La phase d'expansion

- Entrée : ensemble d'apprentissage A
- On utilise la fonction $Gini$
- Décider si un noeud est terminal :

  Un noeud à la position p est terminal si $Gini(p) \leq i_0$ ou $N(p) \leq n_0$, où i_0 et n_0 sont des paramètres à fixer.
- Sélectionner un test à associer à un noeud :

  On choisit le test qui maximise $\Delta(p, T)$, où p est une position, T un test, P_g (resp. P_d) la proportion d'éléments qui vont sur la position $p1$ (resp. $p2$).

$$\Delta(p, T) = Gini(p) - (P_g \times Gini(p1) + P_d \times Gini(p2))$$

- Affecter une classe à une feuille : on choisit la classe majoritaire.
- Sortie : un arbre de décision.

La phase d'élagage

- Entrée : l'arbre de décision obtenu dans la phase d'expansion
- On construit une suite d'arbres $t_0 t_1 \ldots t_k$.
- On calcule pour chaque t_j l'erreur apparente sur l'ensemble T.
- La suite est donné par :
  1. t_0 est l'arbre obtenu dans la phase d'expansion
  2. t_k est une feuille
  3. À l'étape t_i : pour toute position p de t_i on calcule $g(p)$ et on choisit la position p qui minimise $g(p)$. L'arbre t_{i+1} est un élagué de t_i en position p.
- Sortie : L'arbre de la suite dont l'erreur apparente est minimale.

La fonction g

Calcul de $g(p)$: soit u_p le sous-arbre de t_i à la position p et

$$g(p) = \frac{\Delta_{app}(p)}{|u_p| - 1}, \text{ où } \Delta_{app}(p) = \frac{MC(p) - MC(u_p)}{N(p)}$$

$N(p)$ est le nombre d'exemples de A associés à p, $MC(p)$ est le nombre d'exemples de A mal-classés à p si on élague t_i en position p et $MC(u_p)$ est le nombre d'exemples de A associés à p de t_i mal classés par u_p.

Alternatives à CART

Lorsque la taille de l'échantillon S ne permet pas le découpage en A et T.

L'algorithme C4.5 (phase d'expansion)

- Entrée : ensemble d'apprentissage A et langage de représentation avec des ensembles de tests n-aires.
- On utilise la fonction *Entropie*
- Décider si un noeud est terminal :

  p est terminal si tous les éléments associés à ce noeud sont dans une même classe où si on ne peut sélectionner aucun test.
- Sélectionner un test à associer à un noeud :
  - On envisage seulement les tests qui ont au moins deux branches contenant au moins deux éléments (ces paramètres peuvent être modifiés).
  - On choisit le test qui maximise le gain en utilisant la fonction entropie.

– La fonction $Gain$ privilégie les attributs ayant un grand nombre de valeurs. On modifie $Gain$ comme suit :

$$GainRatio(p,T) = \frac{Gain(p,T)}{Splitinfo(p,T)}$$

avec

$$Splitinfo(p,T) = -\sum_{j=1}^{n} P'(j/p) \times log(P'(j/p))$$

où n est l'arité du test T et $P'(j/p)$ la proportion d'exemples présents à p prenant la j-ème valeur/classe du test T.

– Affecter une classe à une feuille :
On attribue la classe majoritaire. S'il n'y a pas d'exemples on attribue la classe majoritaire du père.

– Sortie : Un arbre de décision

L'algorithme C4.5 (phase d'élagage)

La phase d'élagage est basée sur une heuristique.

Améliorations :

- Attributs discrets
- Attributs continus
- Valeurs manquantes