Arbres de Décision et Sous-ensembles Flous

Christophe Marsala*

Septembre 1994

Abstract

In this paper, we highlight the problem of numerical attributes in decision trees, particularly in the ID3 family decision trees. We propose to discretize these attributes with an entropy measure during the construction of the tree. We incorporate some elements of the fuzzy sets theory to classify an object with this tree. A path from the root of the tree to a leaf is associated with a value of a membership function. The degrees of all the paths are agregated to obtain the object's class. Various agregation fuzzy operators (such as t-norms and t-conorms) are used.

Résumé

Nous mettons en valeur, dans ce rapport, le problème des attributs à valeurs numériques dans les arbres de décision en général, et en particulier dans la famille *ID3*.

Dans notre approche, les attributs sont discrétisés, durant la construction de l'arbre, en utilisant un critère d'entropie. Nous nous basons sur la théorie des sous-ensembles flous durant la classification avec l'arbre de décision construit, une décision est prise par agrégation, à l'aide des opérateurs flous, des décisions intermédiaires.

*Stage de DEA réalisé sous la direction de Bernadette Bouchon-Meunier et l'encadrement de Mohammed Ramdani

Table des matières

1 Introduction **3**

2 Etat de l'art **3**

3 Construction d'un arbre de décision **5**
3.1 Présentation 5
3.2 L'algorithme ID3 5
3.2.1 L'entropie pour décomposer . . . . . µµµµ

1 Introduction

Les arbres de décision sont très souvent utilisés dans le domaine de l'apprentissage inductif. Par contre, il leur est rarement associé les outils de la théorie des sous-ensembles flous. Pourtant, l'*imprécision* est une composante courante de notre réalité, il paraît donc intéressant d'essayer d'en tenir compte.

Dans cette optique, Mohammed Ramdani a déjà proposé de construire des arbres de décision "flous" qui permette de raisonner à partir de mesures typiquement imprécises ([Ramdani94]), le flou intervenant à partir de la base d'apprentissage .

Par contre, le système présenté ici ne fait intervenir le flou qu'une fois l'arbre de décision généré, pendant la phase de classification à partir de cet arbre.

L'algorithme de construction de l'arbre choisi est toujours l'algorithme ID3 de Quinlan, pour lequel on a dû alors prendre en considération les attributs à valeurs numériques, et l'on fait intervenir des techniques floues, pour tâcher de pallier l'imprécision qui apparaît lorsqu'on décide de discrétiser ces attributs.

Après un rapide rappel des différents travaux sur la construction des arbres de décision par l'algorithme ID3 et leur utilisation pour la classification, on détaillera notre façon de construire l'arbre et en particulier la méthode de discrétisation que l'on a choisie. Ensuite, on parlera de l'utilisation de l'arbre pour la classification, grâΔΔΔ

d'entropie qui va servir à trouver ce seuil car elle a l'avantage de tenir compte de la répartition des classes.

Par contre, il y a deux possibilités pour choisir le moment de cette discrétisation:

- soit elle est faite avant le lancement de l'algorithme ([Quinlan86]),
- soit le seuil est calculé *dynamiquement* durant le déroulement de l'algorithme; il découpera ainsi l'ensemble des valeurs de l'ensemble courant.

Dans ce dernier cas, pour réduire le coût de la recherche du meilleur point de coupure à chaque étape, [Catlett91] propose de calculer récursivement tous les futurs points de coupure d'un attribut dans une étape préliminaire de l'algorithme. Quant à [Fayyad et al.93], il propose, après avoir constaté (et démontré) que les points de coupure sont des points de séparation de classes ce qui réduit le coût de la recherche, de discrétiser en plus de 2 sous-valeurs en utilisant une mesure du gain de ces découpages supplémentaires, pour décider de les mettre en œuvre.

Une troisième solution est celle de Mohammed Ramdani ([Ramdani94]): il propose de construire un arbre de décision flou en utilisant des mesures imprécises et des nombres flous dans la base d'apprentissage, ainsi qu'une mesure d'entropie floue pendant l'algorithme de construction.

L'utilisation pour la classification des arbres de décision est assez évidente: les tests aux nœuds sont respectés pour suivre un chemin vers une feuille de l'arbre, qui donne alors la classe recherchée.

En utilisant cette solution et dans le cas où il n'a pas été fait de discrétisation pendant la construction de l'arbre, garder des nœuds tests de type symbolique pour des valeurs numériques peut réduire fortement le taux de bonnes classifications. Dans ce sens, dans [Maher et al.93] on complète le système en se servant de nombres flous lors de cette classification.

Si les valeurs numériques ont été discrétisées pendant la construction, ce sont des tests par rapport à un seuiΔΔΔ

3 Construction d'un arbre de décision

3.1 Présentation

Dans le cadre de l'apprentissage inductif, on se propose de déduire un système de règles (représenté ici sous la forme d'un arbre de décision) à partir d'une *base d'apprentissage*, c'est-à-dire d'un ensemble d'*observations* d'un domaine particulier.

Dans le domaine concerné, un expert a, au préalable, déterminé une série d'*attributs* qui va servir à caractériser les objets à observer. A chaque objet, il sera donc associé une combinaison de valeurs de chacun de ces attributs. Une observation est alors l'association d'une classe à un objet.

Par exemple, on peut donner comme attribut: la couleur des cheveux, l'âge ou le fait de fumer ou non.

Un objet serait alors décrit par un triplet: (bruns, 22, oui).

Et une observation pourrait être: l'objet (bruns, 22, oui) n'est pas cardiaque. La classe de la base d'apprentissage sera alors le fait d'être cardiaque ou non.

3.2 L'algorithme ID3

L'algorithme ID3[3] de J. R. Quinlan[Quinlan86] appartient à la famille des algorithmes dits: *Top Down Induction of Decision Tree*. Il construit un arbre de décision en commençant par sa racine jusqu'aux feuilles. Pour cela, il procède de façon récursive en décomposant, en fonction d'un attribut, la base d'apprentissage en sous-ensembles pour lesquels il construit un sous-arbre de décision.

3.2.1 L'entropie pour décomposer

Pour choisir l'attribut qui va servir à décomposer la base, on utilise l'*entropie de Shannon*.

Par analogie avec la thermodynamique, l'entropie d'un ensemble mesure son *degré de prédictabilité*, à savoir: la difficulté que l'on a pour prédire la classe de l'un de ses éléments.

Soit E un ensemble d'éléments répartis en 2 classes P et N, avec p exemples de classe P et n de classe N.

L'*entropie de Shannon* de cet ensemble est définie par:

$$I(p,n) = -\frac{p}{p+n}\log_2\left(\frac{p}{p+n}\right) - \frac{n}{p+n}\log_2\left(\frac{n}{p+n}\right)$$

Un attribut a_j avec k valeurs $v_{j1}...v_{jk}$ partitionne l'ensemble d'apprentissage en k sous-ensembles E_k contenant chacun n_k et p_k exemples dans chacune des classes.

L'information attendue pour a_j est donnée par:

$$Ent(a_j) = -\sum_{i=1}^{k}\frac{p_i+n_i}{p+n}I(p_i,n_i)$$

La variation d'information obtenue en partitionnant E par a_j est:

$$\Delta(a_j) = I(p,n) - Ent(a_j)$$

L'attribut que l'on prend pour décomposer est celui qui apporte le *plus* d'information sur l'ensemble d'apprentissage, c'est-à-dire celui qui *maximise* $\Delta(a_j)$. Et par voie de conséquence, étant donné que $I(p,n)$ est constant, on choisit comme attribut celui qui *minimise* $Ent(a_j)$.

[3] *cf.* algorithme 1 en annexe

3.2.2 L'entropie pour arrêter le découpage

Comme le propose Quinlan[Quinlan86], on se sert aussi de cette mesure d'entropie pour déterminer si l'on atteint le critère d'arrêt — si tous les éléments de l'ensemble sont dans la même classe — qui se traduit par "si l'entropie de l'ensemble est nulle".

Ainsi, on peut se fixer un *seuil* d'entropie pour arrêter l'algorithme. Cela est particulièrement utile pour des bases d'apprentissage qui peuvent contenir des données *bruitées*. A cause d'une erreur de relevé, ou peut-être dans le cas où les attributs n'ont pas été choisis d'une manière pertinente, le critère d'arrêt peut ne jamais s'annuler et entraîner le bouclage de l'algorithme. Grâce à l'utilisation d'un seuil d'entropie, on s'arrête quand on *considère* que ce n'est plus la peine d'essayer de découper l'ensemble. C'est aussi utile, par exemple, pour éviter de décomposer des ensembles qui ont une classe fortement majoritaire mais qui n'est pas unique.

3.2.3 Choix de la version itérative

Des variantes de l'algorithme ID3 permettent de construire l'arbre de façon itérative. L'idée de base serait de construire un arbre de décision au fur et à mesure que des observations sont faites. La solution itérative proposée par Quinlan[Quinlan86] est de commencer par construire un arbre avec une partie de la base, puis de tester l'autre partie de la base avec cet arbre. Les cas qui sont alors mal classés sont alors ajoutés dans l'ensemble ayant servi à construire l'arbre pour en construire un meilleur.

D'autres solutions ont ΔΔΔ

3.3.1 La discrétisation des valeurs continues

Au départ, les valeurs continues représentent tout leur ensemble de définition ($\mathbb{R}$ ou $\mathbb{N}$) on découpe cet ensemble en sous-intervalles. Soit I, un intervalle que l'on veut découper en deux; pour trouver la *valeur de coupure*, on utilise encore une fois l'entropie ([Catlett91]):

on cherche v_m telle que $Ent(I_1) + Ent(I_2)$ soit minimale.
(avec $I_1 = \{v \in I \mid v \leq v_m\}$ et $I_2 = \{v \in I \mid v_m < v\}$, v_m n'appartenant pas obligatoirement à I).

En fait, v_m ne sera pas une valeur de l'ensemble des observations, mais une *moyenne* entre 2 valeurs v_1 et v_2 observées ([Quinlan86]).

$$v_m = \frac{v_1 + v_2}{2}$$

Mais ce n'est pas la seule manière de calculer cette valeur moyenne, pour tenir compte de la répartition des valeurs des 2 côtés de la coupure, on pourrait aussi calculer v_m comme un *centre de gravité*:

$$v_m = \frac{|I_1|.v_2 + |I_2|.v_1}{|I|}$$

(avec $|I|$ = cardinal de I ...)
v_m se trouvant alors plus proche de l'ensemble de cardinal le plus petit.

Cette méthode de découpage se généralise aisément, de façon récursive, pour découper en 3 intervalles ou plus. De plus, elle a l'avantage de tenir compte de la répartition des classes pour découper.

3.3.2 La mise en œuvre

En pratique, la discrétisation peut intervenir à deux endroits possibles:

- avant la construction de l'arbre, elle est faite une fois pour toute, et dans ce cas, du point de vue de l'algorithme, il ne traitera que des valeurs symboliques.
- ou alors, et c'est la solution que l'on a décidé d'implémenter, elle est faite au fur et à mesure de la construction de l'arbre, à l'étape *(b.1)*[6] de la construction. Un attribut pourra être discrétisé plusieurΔΔΔ

On peut cependant réduire ce coût en n'examinant comme possible point de coupure, que les valeurs qui séparent deux classes différentes; car si 2 valeurs consécutives sont dans la même classe, on ne minimisera pas la somme des entropies en coupant l'ensemble entre ces 2 valeurs ([Fayyad et al.93]).

Un autre facteur qui élève le coût de la recherche du point de coupure est le fait que les valeurs de l'attribut en cours de traitement doivent être *triées* à chaque calcul des entropies ([Catlett91]). Mais dans le langage choisi pour l'implémentation de l'algorithme –le *langage C*– cet argument a pu être contourné[8].

3.4 Prise en compte des valeurs inconnues

C'est un type de bruitage particulier: la valeur d'un attribut pour un cas est manquante ou non-référencée.

Il y a de nombreuses façons de traiter ce type de bruitage ([Quinlan89]). Dans le cas des attributs `ΔΔΔ

4.3.2 Union des valeurs trouvées

Après avoir parcouru tout l'arbre pour chaque classe, en fonction de l'exemple, on obtient plusieurs degrés d'appartenance dépendant du chemin suivi. Il faut maintenant agréger ces différents degrés d'appartenance pour une même classe pour n'en obtenir qu'un unique.

Ce degré d'appartenance global à chacune des classes possibles, pour cet exemple, peut être réalisé en utilisant une *t-conorme*, opérateur d'union des sous-ensembles flous.

Par exemple, avec la t-conorme de Zadeh, l'opérateur est le "maximum".

4.3.3 La classe de l'exemple

Finalement, la classe de l'exemple en question sera celle qui aura le plus haut degré d'appartenance.

On peut envisager d'autres moyens d'utilisation de tous ces degrés d'appartenance qui tiendrait compte de tous lΔΔΔ

par conséquent, le degré d'appartenance de l'exemple à la classe 2 pour cette branche est l'intersection entre 1 et $0,75$. Ce qui, avec la t-norme de Zadeh, donnerait: $\min(1;\ 0,75) = 0,75$.

Pour chaque classe, on obtient alors plusieurs degrés d'appartenance dépendants du chemin suivi. Ils vont nous permettre de déduire un degré d'appartenance *global* de l'exemple examiné à chacune des classes possibles.

Ainsi, pour notre exemple, et avec la t-conorme de Zadeh, on obtient:

$$\begin{array}{lcl} \text{Classe 1} & : & \mathit{0,25} \\ \text{Classe 2} & : & \max(\mathit{0,25};\ \mathit{0};\ \mathit{0,75}) = \mathit{0,75} \\ \text{Classe 3} & : & \max(\mathit{0};\ \mathit{0,25};\ \mathit{0,1}) = \mathit{0,25} \end{array}$$

La décision finale sur la classification de l'exemple examiné se fait en considérant ces valeurs d'appartenance obtenues. Dans notre cas, on choisit comme classe pour l'exemple celle qui a le plus fort degré d'appartenance, donc ici, la classe de e est 2.

5 Résultats

Ecrit en *langage C*, le programme permet d'exécuter l'algorithme ID3 sur d'importantes bases d'apprentissage et bases de test. On l'a testé avec différents fichiers de données disponibles par liaison "ftp".

5.1 Fichiers sur les ondelettes

Ces données font partie du projet Numérique/Symbolique du P.R.C. d'Intelligence Artificielle.

On dispose de plusieurs fichiers comme base d'apprentissage (contenant chacune 300 observations), et d'une base de test de 5000 exemples. La différence entre les bases ΔΔΔ

Dans ce tableau, les colonnes représentent:

- *Seuil Entropie*: la valeur du seuil.
- *Nbre de Branches* et *Prof. Moy.*: données sur la taille de l'arbre construit.
- *Normal*, *Proba.*, *Zadeh* et *Weber*: taux de bonnes classifications suivant l'utilisation des opérateurs correspondants, *Normal* donnant le résultat sans fonction d'appartenance.

On peut remarquer que l'utilisation de fonctions d'appartenance apporte souvent un gain dans la classification. En particulier, les opérateurs de Zadeh donnent toujours de meilleurs résultats que ceux de Weber.

En ce qui concerne le seuil d'entropie, il permet de ne pas décomposer des ensembles *globalement* prévisibles. La signification de ce terme "globalement", étant fonction de la valeur fixée pour le seuil: plus ce seuil augmente, et plus l'arbre perd en spécialisation et gagne en généralité. L'avantage de ce type d'*élagage* (on regroupe des branches qui concernent une faible partie de la population étudiée, avec celles qui représentent une plus importante partie) dépend d'une valeur du seuil: celle qui n'autorise des regroupements que dans une certaine mesure. Au dessus de cette valeur, les élagages sont trop nombreux, et en dessous de cette valeur, ils ne sont pas suffisants.

5.1.2 Discrétisation en 2 ou en 3 valeurs

On va maintenant examiner la différence de résultats en fonction du seuil d'entropie dans le cas où l'on discrétise en 2 ou en 3 valeurs.

Comme dans la section précédente, on ne rapporte pas les résultats pour les autres opérateurs car ils se situent, en règle générale, entre les résultats de ceux mentionnés.

Seuil Entropie	*Nbre de Branches*	*Prof. Moy.*	*Normal*	*Proba.*	*Zadeh*	*Weber*
0.0	59	5.25	67.64	67.68	67.58	67.84
0.1	57	5.28	67.92	68.00	67.90	68.16
0.2	51	5.22	68.16	68.26	68.16	68.46
0.24	45	5.09	68.56	68.62	68.56	68.60
0.25	25	5.04	71.28	71.30	71.26	71.32
0.30	21	4.86	71.12	71.14	71.08	71.20
0.40	15	4.33	71.10	71.08	71.02	71.10
0.50	5	3.57	68.40	68.40	68.44	68.38

Les résultats sont toujours moins bons que dans le cas de la discrétisation en 2 valeurs. On peut toujours noter que les opérateurs d'agrégation sont un peu meilleurs que la méthode habituelle, mais cette fois, ce sont les opérateurs de Weber qui donnent de meilleurs résultats que ceux de Zadeh.

Enfin, la discrétisation en 2 ou 3 valeurs donne des arbres plus larges mais moins profonds que dans le cas de la discrétisation en 2 valeurs.

5.1.3 Itérativité et discrétisation en 2 valeurs

On va maintenant tester l'apport de l'algorithme itératif en faisant varier la taille de la fenêtre. On garde comme seuil d'entropie, la valeur 0. 30, et l'on ne discrétise qu'en 2 valeurs.

Fenêtre	*étapes*	*Utilisées*	*Branches*	*Prof. Moy*	*Normal*	*Proba.*	*Zadeh*	*Weber*
30	14	258	30	6.27	72.94	72,90	72.98	72.46
50	8	235	24	5.12	72.82	72.82	72.86	72.54
100	7	247	26	5.46	73.54	73.52	73.56	73.26

Avec:

- *Fenêtre*: nombre d'observations pour construire le premier arbre, c'est aussi le nombre de mauvaises classifications nécessaires avant de reconstruire l'arbre.

- *étapes*: nombre d'arbres construits en tout.
- *Utilisées*: le nombre d'observations effectivement utilisées pour construire l'arbre de décision.

En fait, le fenêtrage permet d'extraire de la base d'apprentissage un sous-ensemble d'observations caractéristiques de toute cette base d'apprentissage.

La taille de la fenêtre d'itération influe donc sur les résultats, et de façon positive: cela s'explique toujours par la perte en spécialisation de l'arbre construit. Cette perte n'apporte que dans une certaine mesure des améliorations en classifications, la taille de la fenêtre doit être choisie avec soin pour que l'itérativité puisse apporter un avantage optimal pour la classification.

5.2 Fichiers sur les maladies cardiaques

Moins de tests ont été réalisé sur les fichiers qui suivent, les résultats sont donnés à titre indicatif.

La base sur les maladies cardiaques est une base qui mélange des attributs numériques et des attributs symboliques. Les données viennent de la "Cleveland Clinic Foundation"[14].

Les observations sont définies par 13 attributs (dont l'âge, le type de douleur ressentie à la poitrine etc...).

On dispose de 9 bases d'apprentissage, chacune de 30 observations, et l'on a une base de test de 240 exemples (différents des observations). Il y a 2 classes à reconnaître: cardiaque ou non.

Les résultats qui suivent concernent la même base d'apprentissage. On a fixé, le seuil d'entropie à 0.50, valeur qui pour ce type de base a donné les meilleurs résultats en classification.

Discrétisation	*Branches*	*Prof. Moy*	*Normal*	*Zadeh*
2	10	3.20	76.25	76.25
2 ou 3	8	2.50	73.75	73.75

5.3 Fichiers sur les reconnaissances de véhicules

C'est le même type de base que pour les données précédentes.

On a 9 bases d'apprentissage de 94 observations et une base de test de 752 exemples.

On dispose de 18 attributs numériques qui représentent des mesures effectuées sur le profil d'une voiture (cœfficient de compacité...). A partir de ces attributs, il faut déterminer la marque de la voiture. Il y a en tout 4 classes a reconnaître.

Les résultats qui suivent concernent la même base d'apprentissage. On a fixé, arbitrairement, le seuil d'entropie à 0.0

Discrétisation	*Branches*	*Prof. Moy*	*Normal*	*Zadeh*
2	22	4.82	66.62	66.76
2 ou 3	27	4.41	61.30	61.84

5.4 Fichiers sur les reconnaissances de champignons

C'est une importante base qui comporte 8416 exemples. Pour les tests, cette base a été répartie aléatoirement en deux fichiers: la base d'apprentissage qui a été faite en prenant 10% de l'ensemble des données et la base de test qui comprend les autrΔΔΔ

Les paliers de changement de classification en fonction du seuil d'entropie sont:

Seuil	*Branches*	*Prof. Moy*	*Normal*
0.0	16	2.81	99.89
0.15	14	2.29	99.03
0.19	11	1.45	98.66
0.2	9	1.00	98.38

La version itérative a été lancée en prenant comme taille de la base d'apprentissage initiale 80 observations, et une reconstruction est faite toutes les 20 erreurs. Le seuil d'entropie a été fixé a 0.1.

Dans ce cas, le nombre d'observations total utilisées pour construire un arbre qui représente la base d'apprentissage, a été de 119 observations, et le taux de bonnes classifications: 96.39%.

En fait, les résultats dépendent du sous-ensemble de la base donnée pris comme base d'apprentissage. On a pu ainsi trouver une base d'apprentissage de 833 observations qui, pour un seuil d'entropie nul, permet de construire un arbre de décision de 17 branches, de profondeur moyenne égale à 3.12 nœuds, qui classifie les 7583 exemples restants sans aucune erreur (*ie* à 100%).

6 Conclusion

A travers cette implémentation de l'algorithme ID3 en *langage C* et les tests effectués sur les différentes bases d'exemples, on peut remarquer le très bon comportement[15] de cet algorithme dans le cas d'attributs symboliques.

Pour les données numériques, en introduisant une discrétisation durant la construction de l'arbre, et des sous-ensembles flous durant la classification, on peut notablement améliorer les résultats.

Les moyens de discrétisation sont encore perfectibΔΔΔ

A Annexes

A.1 Algorithme ID3

Algorithme 1 (ID3) Construction de $AD(E)$:

a) (Critère d'arrêt)
si tous les $e_i \in E$ *sont dans la même classe* c_k,
alors $AD(E) =$ *[Feuille* c_k*].*

b) *sinon*

b.1) *choisir un attribut* $a_j \in A$, *de valeurs* v_{jk}.
On peut décomposer E *en* k *sous-ensembles disjoints:*
$E_k = \{e_i \in E$ *et la valeur de* a_j *pour* e_i *est* $v_{jk}\}$

b.2) $AD(E)$ *a pour racine le nœud test sur* a_i, *et pour fils les sous arbres* $AD(E_k)$.

A.2 Options de programmation pour la recherche du point de coupure

Comme on l'a vu en *3.3*, on n'a pas besoin, pour la recherche du point de coupure, de retrier à chaque étape l'ensemble des valeurs. En effet, les valeurs d'un attribut sont stockées sous la forme d'une liste chaînée, triée au fur et à mesure de l'entrée des observations. Ce tri est une obligation pour pouvoir ensuite calculer les cardinaux des attributs et la fréquence de chaque valeur pour le calcul des entropies.

Durant toutes les coupures par attribut, la structure des listes de valeurs n'est jamais modifiée, seul le nombre d'occurrence de ces valeurs est modifié par ces coupures, et s'annule si la valeur ne fΔΔΔΔΔΔΔΔΔΔΔΔΔ
pas partie de l'ensemble en cours de traitement. En conséquence, les données de l'ensemble en cours de discrétisation n'ont jamais besoin d'être re-triées.

A.3 Calcul du Gain-Ratio

L'idée du *gain ratio* de Quinlan est d'éviter d'utiliser comme nœud de l'arbre, un attribut qui possède un trop grand nombre de valeurs.

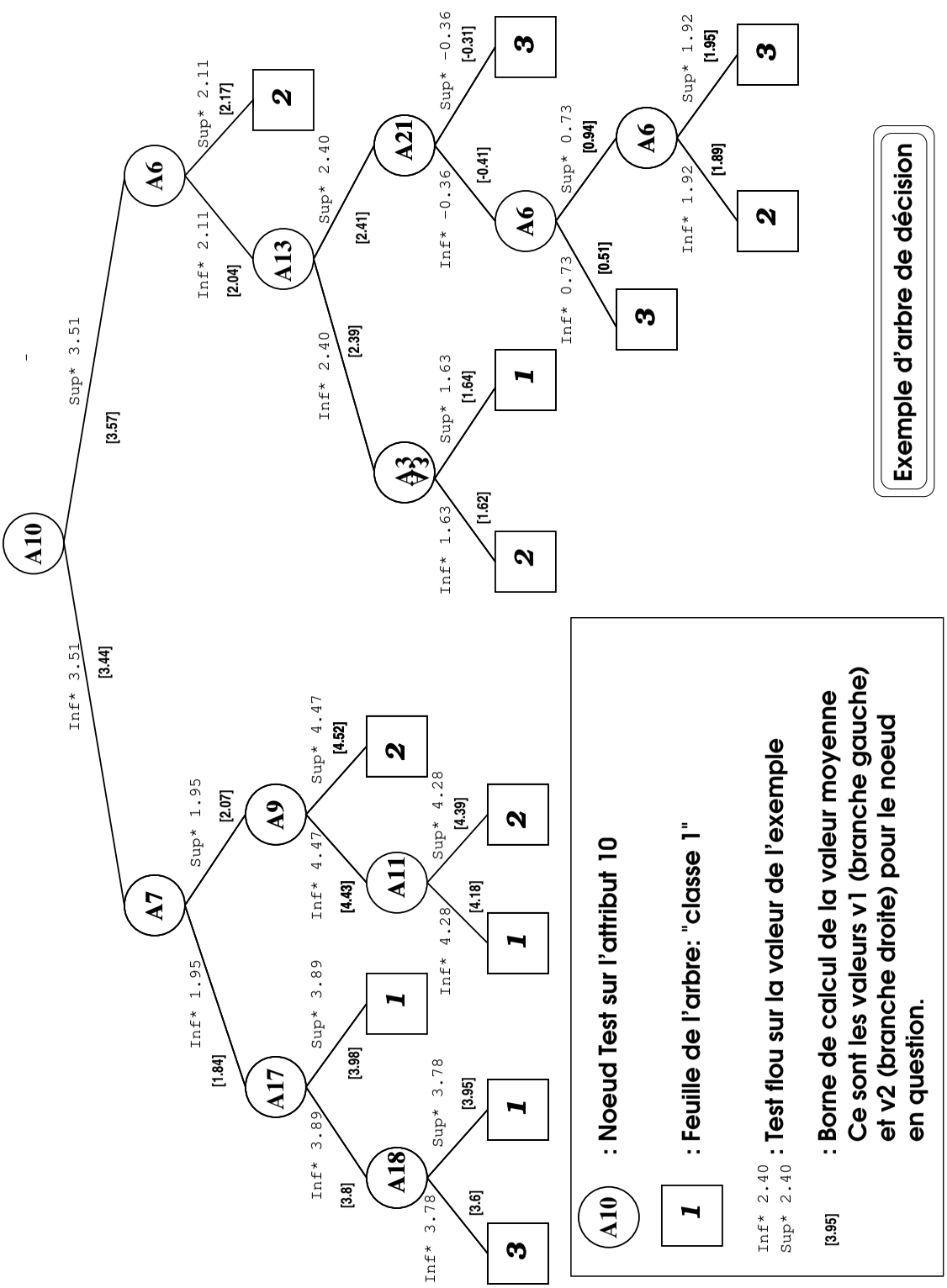
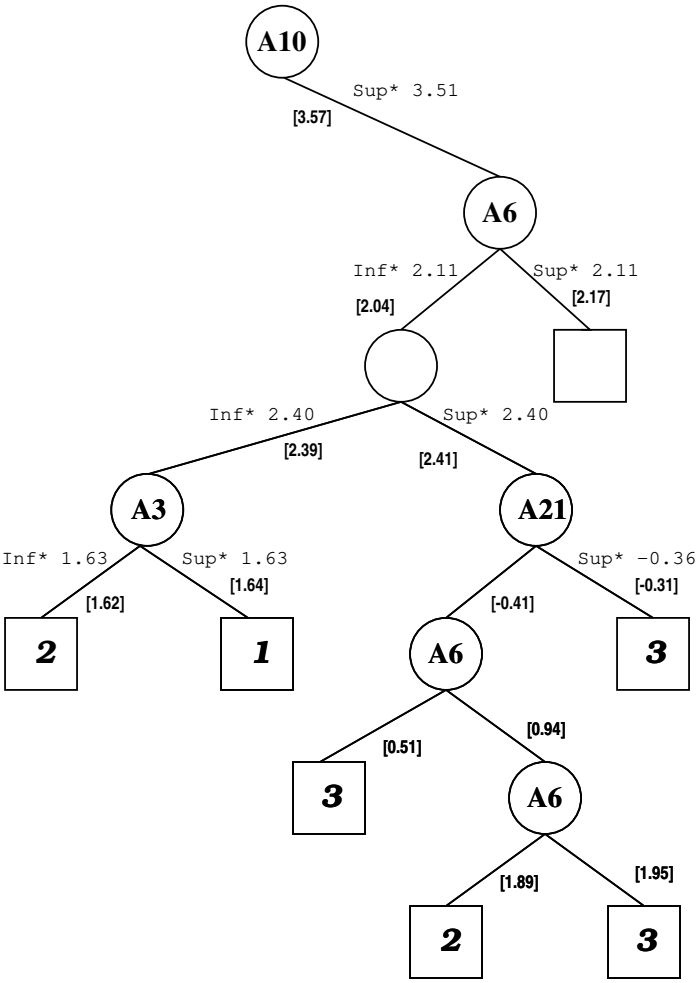
Il utilise encore pour cela, une mesure d'entropie, la mesure de l'information nécessaire pour répondre à la question: "Quelle est la valeur de l'attribut a_j ?", qui est donnée par:

$$IV(a_j) = -\sum_{i=1}^{k} \frac{p_i + n_i}{p+n} \log_2 \frac{p_i + n_i}{p+n}$$

Le *gain ratio* est alors le rapport de la variation d'information apportée par l'attribut a_j et la valeur de $IV(a_j)$:

$$\frac{\Delta(a_j)}{IV(a_j)}$$

A.4 Figure 2: Exemple d'arbre de décision pour attributs numériques



A.5 Figure 3: Exemple de parcours de l'arbre