[Université Sidi Mohamed Ben Abdellah](http://www.fst-usmba.ac.ma/)

Ecole supérieure de technologie

Département GE ET INFO

Statistique et Informatique Décisionnelles (STID)

**Rapport de Projet de fin d’étude :**

Les Arbres de décision

**Les méthodes : ID3, CART, CHAID et C4.5**

Réalisé par  : - Simohamed BEN YASSINE

* NACIM Salah

Encadré par : Mr BENSLIMANE Rachid

Année Universitaire 2016-2017

**Remerciement :**

Je tiens à remercier dans un premier temps, toute l’équipe pédagogique de l'école supérieure de technologie et les intervenants professionnels responsables de la formation Statistique et informatique décisionnelle.

Avant d’entamer ce rapport, nous profitons de l’occasion pour remercier tout d’abord notre professeur Monsieur Rachid BENSLIMANE qui n’a pas cessé de nous encourager pendant la durée du projet, ainsi pour sa générosité en matière de formation et d’encadrement.

Nous le remercions également pour l’aide et les conseils concernant les missions évoquées dans ce rapport, qu’il nous a apporté lors des différents suivis, et la confiance qu’il nous a témoigné.

Nous tenons à remercier nos professeurs de nous avoir incités à travailler en mettant à notre disposition leurs expériences et leurs compétences.

Notre grand respect et nos remerciements au membre du jury qui ont accepté de jugé ce travail.

On doit s’acquitter d’une dette envers tous ceux qui ont contribué à l’aboutissement de ce travail.

**Liste d’abréviations :**

AD: Arbre de décision

ID3: Induction of decision trees

Chaid: Chi-squared Automatic Interaction Detector

Cart: Classification And Regression Trees

**Liste des figures**

##### **Table des matières**

Remerciement 1

Liste d’abréviations 2

Liste des figures3

Table des matières5

Introduction générale7

# Chapitre I : Objectifs des Arbres de décision 8

1. Objectifs généraux 8
2. Principes 9

# Chapitre II : Construction d’un Arbre de décision10

1. Présentation des Arbres de décisions :10
2. Introduction sur les arbres de décision10
3. Représentation de l’arbre de décision10
4. Origines11
5. Définition d’un Arbre de décision 11
6. Avantages11
7. Inconvénients 12
8. Le but des algorithmes de construction d'arbre de décision.
9. Construction d'un arbre de décision 13
10. Les étapes qui reposent la création d’un Arbre de décision 14
11. Mesure de segmentation14
12. Gain informationnel15
13. Ratio de gain 16
14. Critère Gini 16
15. Khi-deux 17
16. Attribut discret et Attribut continu19
17. Critère d’arrêt contrôle de la profondeur de l’arbre20
18. Élagage d’arbres 21
19. Pré élagage23
20. Post élagage23
21. Élagage sur l'erreur réduit23

# Chapitre III : Les méthodes de l’arbre de décision24

1. ID324
2. Définition 24
3. Historique de la méthode ID324
4. Principe 25
5. Shannon Entropie 26
6. Le gain d’information 26
7. Exemple de l’attribut27
8. CART29
9. Presentation29
10. Impact30
11. Indice de Gini30
12. Le rôle de la normalisation30
13. Critère de construction31
14. Les divisions 31
15. Cadre de la classification 31
16. Estimations 31
17. Règle d’arrêt32
18. Propriété32
19. Elagage33
20. Critère d’élagage33
21. CHAID35
22. Introduction35
23. Présentation de CHAID 36
24. Différenciation 37
25. Arbre de décision (Weight-Based)37
26. Input37
27. Output37
28. Les paramètres de CHAID 38
29. C4.541
30. Présentation et Historique41
31. Impact 43
32. Gestion des attributs numériques6
33. Traiter les valeurs manquantes6
34. Elagage45
35. Pré élagage46
36. Post élagage46
37. Estimation des erreurs46

# Chapitre IV : Evaluation de la qualité (Comparaison)48

# Chapitre V : Etude de cas6

# Conclusion6

**Introduction générale**

Les arbres de décision sont l’une des structures de données majeures de l’apprentissage statistique.

Leur fonctionnement repose sur des heuristiques qui, tout en satisfaisant l’intuition, donnent des résultats remarquables.

Leur structure arborescente les rend également lisibles par un être humain, contrairement à d’autres approches où le prédicateur construit est une « boîte noire ».

L’introduction que nous proposons ici décrit les bases de leur fonctionnement tout en apportant quelques justifications théoriques.

Nous aborderons aussi (brièvement) l’extension aux Random Forests. On supposera le lecteur familier avec le contexte général de l’apprentissage supervisé.

Un arbre de décision modélise une hiérarchie de tests sur les valeurs d’un ensemble de variables appelées attributs*.* À l’issue de ces tests, le prédicateur produit une valeur numérique ou choisit un élément dans un ensemble discret de conclusions.

Les arbres de décision sont utilisés dans des domaines d'aide à la décision par exemple l'[informatique décisionnelle](https://fr.wikipedia.org/wiki/Informatique_d%C3%A9cisionnelle) ou l'[exploration de données](https://fr.wikipedia.org/wiki/Exploration_de_donn%C3%A9es).

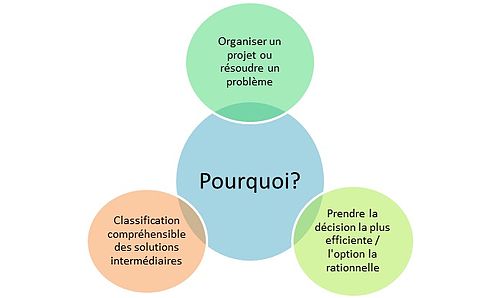
Ils décrivent comment répartir une population d'individus (clients d'une entreprise, utilisateurs d'un réseau social, …) en groupes homogènes selon un ensemble de variables discriminantes (âge, temps passé sur un site Web, catégorie socio-professionnelle, …) et en fonction d'un objectif fixé (aussi appelé « variable d'intérêt » ou « variable de sortie »

Par exemple : chiffre d'affaires, probabilité de cliquer sur une publicité, …).

**Chapitre I :** Objectif des Arbres de décision

1. **Objectifs généraux :**

* Permettre de mener une analyse de [décisions](https://fr.wikipedia.org/wiki/D%C3%A9cision) alternatives, dans le but d’explorer différentes possibilités d’actions sous formes d’hypothèses.
* Proposer les solutions envisageables et adéquates face aux décisions prises ou à prendre.
* Permettre de faire la prédiction : Déterminer la classe d’un nouvel exemple à partir des valeurs de ses attributs



* L’arbre de décision a pour but de mettre sous forme lisible une **succession de choix** afin de pouvoir choisir la solution la plus appropriée.

1. **Principes:**

Les arbres de décision constituent une méthode récente et efficace d’exploration de données, en vue de la prédiction d’une variable qualitative à l’aide de variables de tout type (qualitatives et/ou quantitatives).

Cette flexibilité constitue un avantage par rapport à certains outils de classification, prévus pour des prédicateurs d’un seul et même type.

Il s’agit d’une méthode itérative, dite de partitionnement récursif des données. En effet, la méthode construit des classes d’individus, les plus homogènes possible, en posant une succession de questions binaires (de type oui/non) sur les attributs de chaque individu.

Contrairement à beaucoup d’outils de classification (régression logistique, SVM, etc.), les arbres de décision sont extrêmement intuitifs et fournissent une représentation graphique, parlante et facile à lire, d’un protocole de classification des individus.

Cette représentation graphique est sous forme d’un arbre constitué de feuilles terminales (les classes d’individus) obtenues en suivant un chemin le long des nœuds, chaque nœud correspondant à une question binaire utilisant une variable du jeu de données.

Les arbres de décision permettent donc, d’identifier très rapidement les variables les plus discriminantes d’un jeu de données, en fonction de leur présence parfois répétée le long des nœuds.

**Chapitre II :** Construction d’un Arbre de décisions

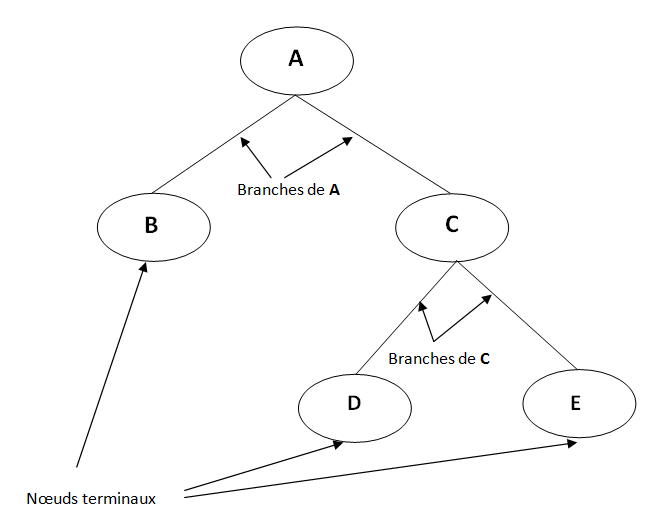
1. **Présentation des Arbres de décisions :**
2. **Introduction sur les arbres de décision**

Les arbres de décision sont la modélisation d'une classification. Ils apprennent à partir d'observations qu'on appelle des exemples. Un exemple est représenté par une série d'attributs et une classe associée, on doit connaître la classe parce que les arbres de décision travaillent sur la classification en mode supervisée.

Les arbres de décision sont un bon moyen d'illustrer le raisonnement pour distinguer les similitudes et les différences entre les attributs des exemples du jeu de données, ils sont souvent utilisés par les statisticiens pour illustrer le résultat d'une analyse.

Un arbre de décision est composé de nœuds en arborescence, le nœud à base de l'arbre est appelé la racine, chacun des nœuds sous la racine est soit une feuille ou un sous-arbre

1. **Représentation de l’arbre de décision**



**Schéma d'un arbre de décision.**

Dans l'arbre de décision de la figure 2.2, les attributs A et B ont chacun deux valeurs distinctes, lorsque les exemples avec l'attribut A égal à a], ils correspondent à une seule classe. Dans le cas, où l'attribut A est égal à a2, les exemples correspondent à deux classes différentes, on a besoin alors de prendre l'attribut B pour diviser les exemples dans leurs classes respectives.

1. **Origines**

Ces méthodes ont pris essentiellement leur essor dans le cadre des approches d'apprentissage automatique (machine learning) en Intelligence Artificielle

1. **Définition des Arbres de décisions**

Les arbres de décision sont dédiés à l’apprentissage supervisé où l’on essaie de prédire (expliquer) les valeurs prises par une variable discrète (Etre malade ou pas, Répondre positivement à une offre promotionnelle ou pas, etc.) à partir d’une série de variables discriminantes de type quelconque.

Un algorithme “arbre de décision” estime un concept cible par une représentation d’arbre, où chaque nœud interne correspond à un attribut, et chaque nœud terminal (ou feuille) correspond à une classe. ⎫

**- Il y a deux types de nœuds :**

**Nœud interne :** se déploie en différentes branches selon les différentes valeurs que l’attribue peut prendre.

**Exemple :** luminosité <= T1 or luminosité > T1. ⎫

**Nœud terminal :** décide la classe assignée à l’exemple

1. **Avantages:**

* Facilité à manipuler des données « symboliques »
* Interopérabilité
* Classification très efficace
* Prise en compte simultanée de variables qualitatives et quantitatives (discrètes ou continues);
* Pas d'hypothèse au sujet des données (modèle non-paramétrique);
* Non affecté par les problèmes d'échelles de mesure des variables quantitatives (pas de combinaison arithmétique des variables) et détermine des seuils discriminants pour ces dernières;
* Sélection des variables les plus informatives (en tenant compte des interactions);
* Peu d'influence des données erronées, SAUF aux frontières inter classes;
* Algorithmes très rapides en phase de construction des arbres et lors de la classification de nouveaux cas (1 seul chemin est parcouru);
* Règles logiques de classification aisément interprétables => extraction de connaissances explicites d'un sens de données (= "data mining") => meilleure compréhension du problème étudié.

1. **Inconvénients** :

* Sensibilité au bruit et points aberrants.
* Traitement des **variables numériques** (cf. précédemment): génération des tests (choix des seuils) ne tient pas compte des propriétés de densité (proximité) des valeurs. => Nouveaux développements avec de nouveaux critères de sélection pour les variables numériques.
* **Algorithmes séquentiels** sans remise en cause des étapes précédentes (d'où rapidité), un peu assoupli dans la production de systèmes de règles.
* **Instabilité**: sensibles aux variations (même assez faibles) de l'ensemble d'apprentissage en termes d'exemples, des variables considérées;

🡺Variations dans les arbres produits (variables sélectionnées, seuils des variables numériques, structure de l'arbre, ...) et de leurs performances.

🡺Limitations similaires aux algorithmes de sélection de variables stepwise: algorithmes rapides mais n'investiguant qu'un nombre restreint de possibilités à chaque étape, sans remise en cause des choix précédents!

🡺 Une petite variation dans les données peut entraîner un choix différent à un certain niveau => sous-arbres différents => Quel est l'arbre optimal ? (peut remettre en cause l'aspect "extraction des connaissances" !)

1. **Le but des algorithmes de construction d'arbre de décision.**

Les algorithmes de construction d'arbre de décision permettent de créer des arbres de décision avec une taille la plus petite que possible, et ce, de façon à créer des règles de décision simples. Plus un arbre de décision est grand, plus les règles sont complexes. Les algorithmes de construction d'arbres choisissent les attributs toujours par rapport aux classes.

1. **Construction d'un arbre de décision**.

Les arbres de décision sont construits à partir d'un jeu d'apprentissage, un jeu d'apprentissage est une matrice, où les lignes représentent les exemples et les colonnes représentent les caractéristiques des exemples, la dernière colonne est réservée aux classes associées aux exemples.

L'algorithme de construction a aussi besoin d'un tableau d'index qui constitue la liste de référence des attributs à traiter.

L'algorithme de construction d'arbre de décision se divise en 3 étapes. La première étape consiste à vérifier si on doit faire un nœud terminal pour représenter les exemples du jeu d'apprentissage. Pour faire un nœud terminal, on doit respecter une des conditions suivantes: Tous les exemples du jeu d'apprentissage appartiennent à la même classe ou tous les attributs ont été utilisés pour les nœuds précédents.

Cette étape permet d'arrêter l'expansion de la branche de l'arbre.

La deuxième et la troisième étape se produisent lorsqu'on ne respecte pas les critères de la première.

La deuxième consiste à trouver l'attribut pour représenter le nœud de l'arbre. Les algorithmes de construction d'arbre de décision utilisent une mesure de segmentation par rapport aux attributs à traiter. Nous allons voir en détail les différentes techniques plus tard.

La troisième étape consiste à éclater le jeu d'apprentissages pour créer les branches du nœud, chacune des branches du nœud prend une des différentes valeurs que l'attribut du nœud peut prendre. Pour chacune des branches qu'on aura créées, il faut recommencer le processus en prenant les exemples correspondants à la branche.

1. **Les étapes qui reposent la création d’un Arbre de décision :**

**II. Mesure de segmentation**

La mesure de segmentation est l'heuristique qui permet de choisir l'attribut qui permettra de répartir le mieux le jeu d'apprentissages. Cette mesure est souvent une mesure statistique.

L'objectif principal est de construire des arbres de décision relativement simple. On recherche un arbre petit et simple plutôt qu'un arbre grand qui est complexe.

Le choix des attributs à tester est une étape cruciale pour la construction d'un arbre.

Pour cela, la mesure de segmentation doit évaluer toutes les possibilités de choix pour chacun des niveaux d'un arbre de décision.

Pour choisir la variable de segmentation sur un sommet, l’algorithme teste toutes les variables potentielles et choisit celle qui maximise un critère donné. Il faut donc que le critère utilisé caractérise la pureté (ou le gain en pureté) lors du passage du sommet à segmenter vers les feuilles produites par la segmentation.

Il existe un grand nombre de critères informationnels ou statistiques, les plus utilisés sont :

- [l’entropie de Shannon et le](file:///\\wiki\Entropie_de_Shannon) [coefficient de Gini et leurs variantes.](file:///\\wiki\Coefficient_de_Gini)

- [le lien du KHI-2 et ses dérivés. I](file:///\\wiki\Coefficient_de_Gini)[l permet de](file:///\\wiki\Entropie_de_Shannon) [mesurer le lien entre la variable candidate et la variable à prédire.](file:///\\wiki\Coefficient_de_Gini)

- **Choix de la variable de segmentation :**

1. **Gain informationnel :**

* **Entropie de Shannon :**

C’est une fonction mathématique qui, intuitivement, correspond à la quantité d'[information](https://fr.wikipedia.org/wiki/Information) contenue ou délivrée par une source d'information. Cette source peut être un texte écrit dans une langue donnée, un [signal électrique](https://fr.wikipedia.org/wiki/Signal_%C3%A9lectrique) ou encore un [fichier informatique](https://fr.wikipedia.org/wiki/Fichier_informatique) quelconque (collection d'octets).

**Shannon en 1949 a proposé une mesure d’entropie valable pour les distributions discrètes de probabilité.**

* Quantité d’information pour connaître les valeurs de Y :



* **Entropie Conditionnelle :**

Quantité d’information pour connaitre les valeurs de Y, Sachant les valeurs de X :



* **Gain d’entropie :**



* **Gain d’entropie normalisée :**

1. **Gain Ratio :**

Tenir compte de la distribution marginale de X :



1. **Critère Gini (Indice de concentration) :**

* **Indice de Gini :**

L’indice (ou coefficient) de Gini est une mesure, comprise entre 0 et 1, de la dispersion d’une distribution. Il est très souvent utilisé en économie ou en sociologie afin de mesurer les inégalités sociales au sein d’un pays. Dans ce contexte, plus le coefficient est proche de 1 et plus la société est inégalitaire.

On ne s’attardera pas ici sur les détails techniques concernant le coefficient de Gini.

ll est néanmoins important de comprendre l’analogie avec l’entropie probabiliste : comme cette dernière, l’indice de Gini a pour but d’évaluer la pureté de chaque feuille, et de permettre la création de feuilles aussi homogènes (donc « égalitaires ») que possible.

* **Concentration des valeurs de Y :**



* **Indice de Gini Conditionnel :**

Concentration de Y , Sachant les valeurs de X :



* **Amélioration de la concentration :**



* 1. **Indice de Gini = Entropie Quadratique :**

On peut aussi interpréter D comme un grain informationnel

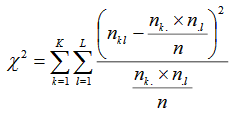
* 1. **Indice de Gini = Variance sur variables catégorielles :**

On peut aussi interpréter D comme une variance inter-classes

**Une variance inter-classes = Variance totale – Variance intra-classes**

1. **Lien du Khi-2 :**

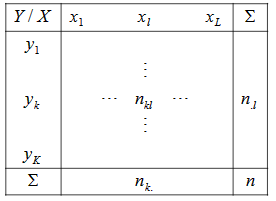
Pour évaluer la pertinence de la variable dans la segmentation, CHAID propose d’utiliser le Khi-2 d’écart à l’indépendance, bien connu en statistique, dont la formule est la suivante :



Le critère du Khi-2 varie de 0 à +∞ . Il n’est pas aisé de le manipuler car il avantage les descripteurs ayant un nombre élevé de modalités. Il est bien souvent préférable de le normaliser par le nombre de degrés de libertés, en prenant par exemple le t de Tschuprow dont le domaine de définition est [0 ; 1]

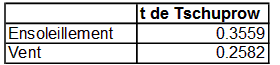


Cette variante n’est pas proposée dans le descriptif originel de Kass (1980). Elle n’a aucun effet si les descripteurs comportent le même nombre de modalités, mais elle semble de bon sens dès lors que l’on traite des descripteurs très disparates.



**Tableau : Tableau des effectifs lors du croisement de deux variables**

**Exemple :**



**Tableau : Descripteurs discrets candidats sur la racine de l'arbre**

Dans l’exemple, le calcul du t de **Tschuprow** sur les deux descripteurs candidats a produit les résultats repris dans le Tableau 4. Nous notons que la meilleure variable est bien «ensoleillement» avec un t de **Tschuprow** de 0.3559.

**Amélioration :**

La mesure du X² augmente avec :

* + - ‘n’, l’effectif sur le nœud à segmenter

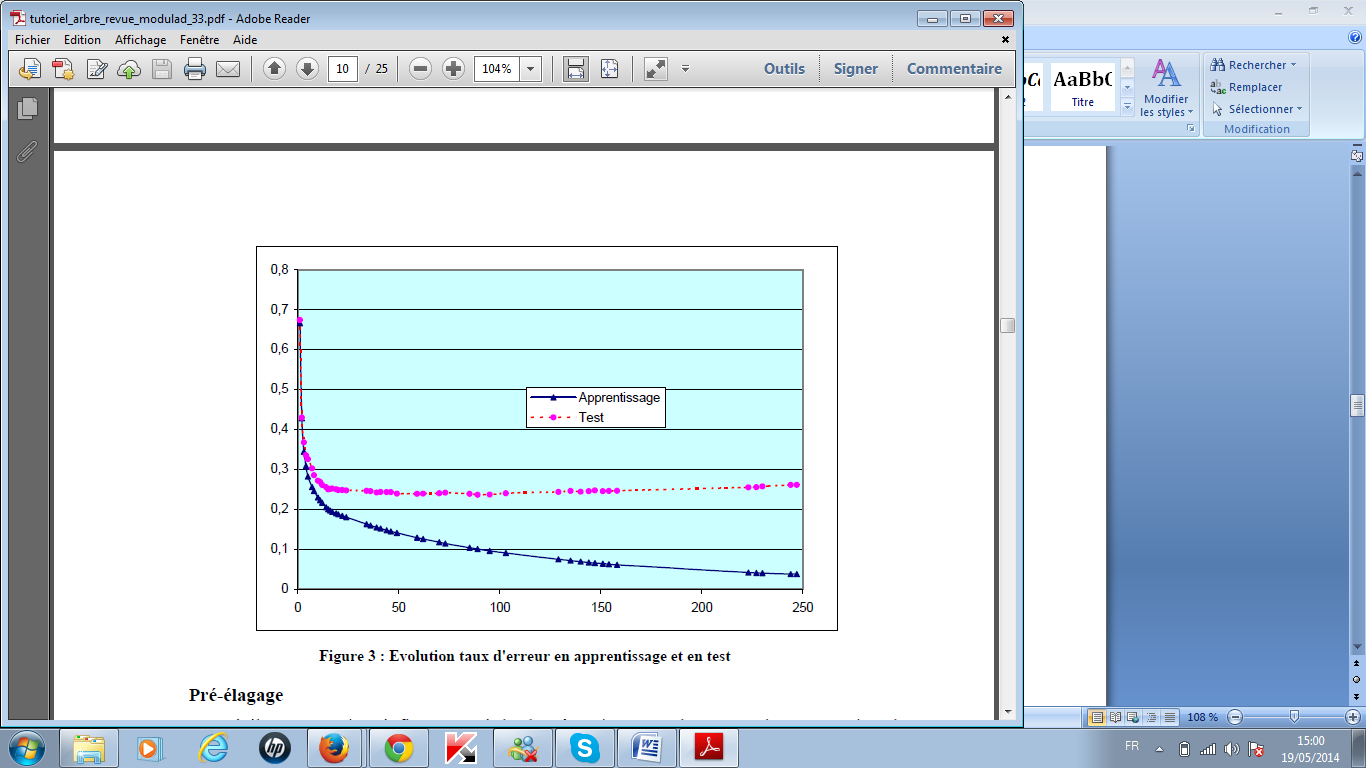
***Ces valeurs sont constantes dans les comparaisons***

* + - Le nombre de lignes
    - Le nombre de colonnes 🡺 Les variables qui ont beaucoup de modalitéssont avantagés.

1. **Attribut discret et Attribut continu**

Les attributs discrets sont des attributs qui ont un nombre limité de valeurs possible, C4.5 traite chacune des possibilités indépendamment, si un attribut choisi comme un nœud de l'arbre a n valeurs, le nœud aura n branches; CART produit des nœuds avec seulement deux branches, si un attribut a n valeurs, on a 2n-l\_l possibilités de faire des regroupements de valeurs pour effectuer les tests [4], l'algorithme calcule l'index GINI de chacune de ces possibilités, on choisi le test qui maximisera l'index GINI.

1. **Règle d’arrêt : contrôle de la profondeur de l’arbre**

****

A mesure que le nombre de feuilles – la taille de l’arbre – augmente, le taux d’erreur calculé sur les données d’apprentissage diminue constamment. En revanche, le taux d’erreur calculé sur l’échantillon test montre d’abord une décroissance rapide, puis nous observons que le taux d’erreur reste sur un plateau avant de se dégrader lorsque l’arbre est manifestement surdimensionné.

L’enjeu de la recherche de la taille optimale consiste à stopper - pré-élagage - ou à réduire - post-élagage - l’arbre de manière à obtenir un classifieur correspondant au « coude » de la courbe sur échantillon test, lorsque le taux d’erreur commence à stagner

1. **Élagage d’arbres**

**Objectif:** supprimer les parties de l'arbre qui ne semblent pas performantes pour prédire la classe de nouveaux cas.

🡺 Remplacées par un nœud terminal (associé à la classe majoritaire).

**Processus:** généralement de type "bottom-up" (du bas vers le haut: des extrémités vers la racine), basé sur une estimation du taux d'erreur de classification: un arbre est élagué à un certain nœud si le taux d'erreur estimé à ce nœud (en y allouant la classe majoritaire) est inférieur au taux d'erreur obtenu en considérant les sous-arbres terminaux.

Supprimer les branches (parties terminales) peu représentatives pour garder de bonnes performances prédictives (généralisation) 🡺 Nécessité d'un critère pour désigner les branches à élaguer. Après élagage, les nouvelles feuilles sont labélisées sur base de la distribution des exemples d'apprentissage (classe majoritaire).

Cette phase est exécutée après la construction de l'arbre. On considère que la racine de l'arbre comprend au moins deux feuilles. Tant qu'il existe un sous arbre que l'on peut remplacer par une feuille, sans faire croître l'estimation de l'erreur réelle, alors on élague ce sous arbre. On doit savoir si les nœuds fils sont des feuilles ou des sous arbres. Si tous les nœuds sont des feuilles, on remplace la racine du sous-arbre par une feuille, si l'erreur de la nouvelle feuille est plus petite que celle de l'ancien sous arbre.

Pour créer la feuille de remplacement, on crée la liste des possibilités d'éléments des feuilles du sous-arbre et on prend la valeur la plus fréquente pour le nom de l'étiquette et le(s) autre(s) dans le tableau Etiquette des autres possibilités. On remonte ensuite jusqu'à la racine, jusqu'à ce qu'on ne puisse plus remplacer un sous arbre par une feuille.

**Arbres** **parallèles aux axes versus obliques (données numériques)?**

* + **Avantages des arbres parallèles:**
* suite de tests mono-variés;
* pas de problème de combinaison de var, d’échelle ... ;
* sélection de variables intégrée;
* rapidité;
* génération de règles logiques simples de classification.
  1. **Désavantages - limitations:**
* approximation par "escaliers" des surfaces de séparation;
* les critères usuels de sélection de tests ne tiennent pas compte des densités de points dans l’espace des données (pour sélectionner une variable et sa valeur seuil, cf. après).
* Une approche très simple consiste à fixer un critère d’arrêt local, relatif au sommet que l’on est en train de traiter, qui permet d’évaluer l’apport informationnel de la segmentation que l’on va initier.

**Problème de sur entrainement**

On coupe une branche à un nœud **‘s’** si l’erreur propagée (be) est plus grande que l’erreur statique.

**e(s)=1- (n+1)/(N+k)**

K est le nombre de classes, **‘n’**est le nombre d’individus au nœud **‘s’**, ayant la classe **‘C’**, **‘N’** est le nombre d’individu au nœud **‘s’.**

**be(s)=Ʃpie(si)**

* + 1. **Pré élagage**

Le pré élagage se produit pendant la construction de l'arbre, il agit comme un critère d'arrêt dans l'expansion de l'arbre.

Décide ou non de continuer à développer un certain nœud.

Il consiste à fixer une condition d'arrêt pour arrêter la construction.

Cette condition limite l'expansion d'une branche, c'est- à-dire que les attributs restants ne permettent plus de diviser les exemples. Si on n'effectuait pas cet arrêt exceptionnel, la branche croîtrait en prenant les attributs restants. Les exemples restants au niveau du nœud que le seuil atteint seraient les mêmes que celle au niveau du nœud terminal. Dans les algorithmes C4.5 et CART, lorsque la mesure de segmentation est égale à zéro ou à l'infini, on doit créer un nœud terminal. Il vaut mieux arrêter la phase d'expansion de la branche que de continuer sans avoir d'apport significatif dans la classification.

* + 1. **Post élagage**

Le post élagage est la méthode la plus utilisée dans la plupart des algorithmes, elle s'effectue une fois que l'algorithme d'expansion est terminé

C4.5 utilise 45 Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Une estimation de l'erreur réduite de l'arbre, cette estimation est produite à partir de l'erreur apparente de l'arbre.

* + 1. **Élagage sur l'erreur réduite réelle :**

C4.5 utilise une estimation de l'erreur pour l'élagage de l'arbre. On utilise une approche ascendante. Avec le paramètre de confiance CF. Pour chacune des feuilles de l'arbre, notons N le nombre d'exemples qu'une feuille 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 valeur de p est trouvée à l'aide de la fonction suivante, où p est une valeur entre 0 et 1 : Par la suite, on remonte jusqu'à la racine, en calculant l'estimation de l'erreur réelle (voir l'équation 2.10) de cet arbre en faisant une somme pondérée des estimations des erreurs réelles de ses fils.

**Chapitre III :** Les méthodes de l’arbre de décision

1. **ID3 :** Induction of decision trees
2. **Definition:**

L’**ID3** est un des nombreux algorithmes possibles pour générer un arbre de décision, et c’est également celui qui va être développé dans la première partie de ce cours. Cet algorithme à été développé en **1986** par Ross Quinlan. Il l’a publié dans le magazine *Machine Learning* parmi d’autres articles.

1. **Historique de la méthode ID3 :**

L’**algorithme ID3** a été développé à l’origine par Ross Quinlan. Il a tout d’abord été publié dans le livre ‘’Machine Learning’’ en 1986.

C’est un algorithme de classification supervisé, c’est-à-dire qu'il se base sur des exemples déjà classés dans un ensemble de classes pour déterminer un modèle de classification. Le modèle que produit **ID3** est un arbre de décision. Cet arbre servira à classer de nouveaux échantillons.

L'algorithme **C4.5** est une amélioration d'**ID3**, notamment du point de vue de la facilité d'implémentation.

**L’algorithme ID3 :**

A été développé par J. Ross Quinlan (1975)

Construit un arbre de décision du haut vers le bas

Utilise comme heuristique le principe du "rasoir d'Occam" pour éviter de faire des suppositions inutiles.

Minimise le nombre de "questions" à poser (l'arbre de décision le plus simple)

Se compose de deux phases principales :

L’apprentissage.

La phase de tests.

**Un arbre de décision** est une structure souvent utilisée pour modéliser le raisonnement d'experts humains.

On considère une collection d'objets de même classe, on cherche à obtenir la définition de cette classe en fonction des propriétés des objets (des valeurs des attributs de cette classe)

Dans un arbre de décision :

* Chaque nœud intermédiaire correspond à une question sur une propriété de l'objet (valeur d'un attribut)
* Chaque lien correspond à une valeur de l'attribut
* Chaque nœud terminal (feuille) correspond à une classe (une collection d'objets de même nature)

Plusieurs feuilles différentes peuvent correspondre à la même classe.

Un chemin de la racine à un nœud correspond à une série de questions et réponses.

Les questions correspondent aux attributs et les réponses à leurs valeurs.

1. **Principes :**
   * Si tous les éléments du nœud sont dans la même classe, alors quitter
   * Autrement, faire les étapes suivantes :
   * Choisir un attribut et l'assigner au nœud.
   * Partitionner les exemples dans des sous-ensembles selon les valeurs de cet attribut.
   * Créer des nouveaux nœuds pour chaque sous-ensemble non-vide de la partition.
   * La liste de nouveaux nœuds devient les enfants du nœud.
   * Appliquer la procédure récursivement sur les nouveaux nœuds.
   * Les théories de Shannon sont à la base de l'algorithme ID3 et Donc C4.5.
   * Entropie Shannon est la plus connue et la plus appliqué.
   * Il définit d'abord la quantité d'informations fournies par
   * Un événement: plus la probabilité d'un événement est faible (il est Rare), plus il fournit d'informations est grande.
2. **Shannon Entropie**

En général, si on nous donne une distribution de probabilité P = (p1, p2, ..., pn) et un échantillon S alors les informations portées par cette distribution, aussi appelée entropie de P, est donnée par:

**Entropie (P) = -**

1. **Le gain d’information G (p, T)**

Nous avons des fonctions qui nous permettent de mesurer le degré de mélange de classes pour tous les échantillons et donc toute position de l'arbre en construction. Il reste à définir une fonction pour sélectionner Le test qui doit étiqueter le nœud actuel.

Il définit le gain pour un test T et une position p

**Gain (p,T)= Entropie (P) –**

pj est l'ensemble de toutes les valeurs possibles, nous pouvons utiliser cette mesure pour classer les attributs et construire l'arbre de décision, où à chaque nœud est situé l'attribut avec le gain d'information le plus élevé parmi les attributs pas encore considéré dans le chemin à partir de la racine.

**GainRatio(p,T) =**

L’entropie est la mesure de l'incertitude sur la classification d'un objet.

**ID3** utilise l'entropie pour calculer l'arbre de décision le plus petit, ne conservant alors que les informations absolument nécessaires pour classer un objet.

Pour partitionner les exemples, **ID3** choisit l'attribut qui donne la plus petite entropie de classification .

L'entropie **H(C|A),** ou l'entropie de la classification après avoir choisi de partitionner les exemples suivant la valeur de l'attribut A :

**H(C | A) =**

Où aj désigne une valeur de l'attribut A, M le nombre total de valeurs pour l'attribut A et p(aj).

La probabilité que la valeur de l'attribut A vaut aj.

**H(C | aj) = -**

Où N est le nombre de classes différentes, **P (ci | aj)** la probabilité conditionnelle qu'un objet, soit de la classe **ci** étant donné que l'attribut A vaut aj.

1. **Exemple de l’attribut**

**ID3** n'est pas un algorithme séquentiel exemple après exemple.

Pour obtenir un arbre de décision concis et suffisant, **ID3** choisit les attributs d'éclatement pour aboutir, par le chemin le plus court et nécessaire, au plus grand nombre d'exemples de la même classe, **ID3** recherche le meilleur attribut (celui qui maximise le gain d'information):

1. Recherche de l'entropie de la collection d'exemples à traiter :

**I (P, N) = log² - log²**

Où :

**N** : nombre d'exemples de la classe 1

**P** : nombre d'exemples de la classe 2

1. Recherche du gain d'information pour chaque valeur vi de l'attribut A calcul de E(A) :

**E(A) =**

1. calcul du gain gagné pour chaque attribut :

**Gain (A) = I (P, N) –E(A)**

1. choix de l'attribut : celui qui maximise le gain :

L'attribut choisi constitue un nœud et chacune de ses valeurs constitue une branche identifiée par la valeur et aboutit à un ensemble d'exemples.

Si les exemples de cette sous-population sont tous de la même classe, alors la branche est terminée.

La feuille qui en découle indique la valeur de la classe.

Sinon, il faudra choisir à nouveau l'attribut qui maximise le gain d'information sur les exemples restant au nœud.

Le chemin allant de la racine jusqu'à une feuille représente une règle.

Chaque chemin de cet arbre est de longueur variable et aboutit obligatoirement à une et une seule classe.

1. **CART :** (Classification And Regression Trees)
2. **Presentation**

L'algorithme construit un arbre de décision d'une manière analogue à l'[algorithme ID3](https://fr.wikipedia.org/wiki/Algorithme_ID3). Contrairement à ce dernier, l'arbre de décision généré par CART est binaire (un nœud ne peut avoir que 2 fils)

Algorithme développé par Breiman, Freidman, Olshen et Stone (1984)

- Estimateurs par histogramme de la fonction Cible s

- Partitionnement récursif et dyadique de l’espace des observations (X)

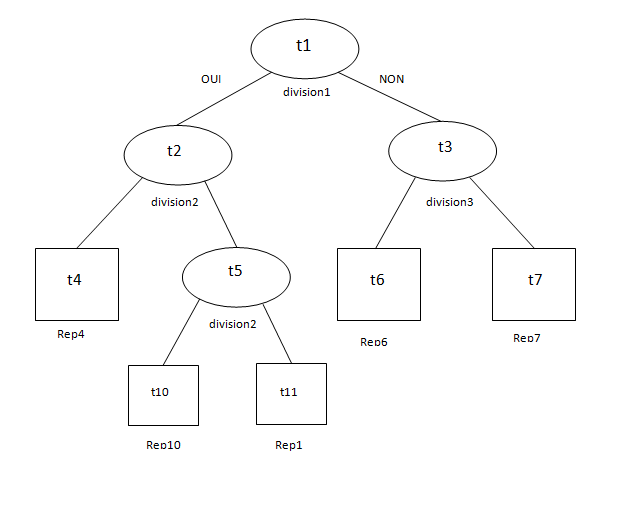
* Algorithme de construction d’un arbre **CART** : **3 étapes successives**

- Construction de l’arbre

- Elagage

- Sélection finale

Voici l’aspect d’un arbre **CART**

: 

1. **Impact**

**Indice de concentration (CART)**

1. **Indice de Gini**

Concentration des valeurs de Y

**I(Y)= -**

1. **Le rôle de la normalisation**

Éviter la fragmentation des données

– La propriété de Fusion des mesures

• Le t de Tschuprow normalise le CHI-2

• Le Gain Ratio normalise le gain informationnel

• Le Gain de Gini n’est pas normalisé

(Mais on s’affranchit autrement de cette limitation dans CART)

**Quelques questions se posent :**

* Comment définir les divisions successives ? ⇒ Critère de construction
* Quand arrêter le principe de division ? ⇒ Règle d’arrêt
* Comment définir les réponses ? ⇒ Règle d’assignation

Une difficulté supplémentaire : les réponses peuvent être différentes selon le cadre d’étude.

1. **Critère de construction**
2. **Les divisions :**

* Question binaire, de la forme Xi ≤ a ou Xi ∈ C

🡺 Détermination de i et de a ou C

Idée : tester toutes les divisions possibles et retenir la meilleure selon un critère

1. **Cadre de la classification :**

* Notations : J = {1,…, J}
* Pj probabilités à priori de la classe j. Peut être estimée par
  + avec Nj = Card{(xk , yk )|yk = j}.
* Soit t un nœud de l’arbre, N(t) = Card{(xk , yk )|xk ∈ t} nombre d’observations de L dans t.
* soit t un nœud de l’arbre et j ∈ J ,
  + Nj (t) = Card{(xk , yk )|xk ∈ t et yk = j} nombre d’observations de L dans t et de classe j.

1. **Estimations :**

* P (j, t) : probabilité qu’une observation soit dans le nœud t et de classe j
  + Estimée par p (j, t) = Pj
* P (t) : probabilité qu’une observation soit dans le nœud t
  + Estimée parp (t)= ∑ p (j, t).
* p (j| t) : probabilité a posteriori dans t de la classe j
  + Estimée par

1. **Règle d’arrêt :**

Par récursivité, la construction de l’arbre devient évidente.

Un nœud t d’un arbre est déclare terminal si :

* Une seule observation dans le nœud t
* Que des observations dans t avec un même label

🡺 Partition minimale de l’espace X.

**Règle d’assignation :**

**Définition :**

Soit t un nœud dont la réponse associée est j(t).

**J (t) = argmax p(j |t)**

La probabilité de mauvais classement du nœud t, évaluée par substitution, est définie par :

**R (t) =**

1. **Propriété :**

Cette définition de j(t) est telle que r (t) est minimale.

1. **Elagage :**

Sous suite de Breiman

Notations :

* Soit T un arbre et t un nœud non terminal de T. Elaguer T à partir de t consiste à créer un nouvel arbre T\* qui n’est autre que T privé de tous les descendants de t.

* Tout arbre T′ obtenu par élagage de T est un sous-arbre de T, ce que l’on note T′ ≺ T.

1. **Critère d’élagage :**

Crit α (T) = R (T) + α | Ť |

Proposition :

Pour chaque valeur de α, il existe un unique sous-arbre de T max, noté T α ,

tel que :

* + **T α = argmin crit α (T) T < Tmax**
  + Si **crit α T α = crit α (T),** alors **T α < T.**

Soit α1 et α2 deux réels positifs avec **α1 ≤ α2** 🡺Alors **T α2 < T α1**.

Détermination du premier élément de la suite : Soit T0 le premier élément de la suite associé à **α = 0.**

* T0 est le sous-arbre de T max obtenu en élaguant tous les nœuds t pour lesquels:

**R(t) = R (tg) + R(td )**.

* Ainsi T0 satisfait :
  + **crit0(T0) = 0**
  + Pour tout nœud t de T0 :

**R(t) > R(td ) + R(tg )**.

**Pourquoi :**

A la fin de la phase d’élagage, nous disposons de plusieurs sous-arbres et donc de plusieurs estimateurs.

🡺En sélectionnant deux méthodes :

* Echantillon test.
* Validation croisée

1. **CHAID :** Chi-squared Automatic Interaction Detector
2. **Introduction**

L'acronyme **CHAID** signifie en fait "Chi-squared Automatic Interaction Detector". Il s'agit de l'une des méthodes d'arbres de classification les plus anciennes, initialement proposée par Kass (1980 ; d'après Ripley, 1996, l'algorithme **CHAID** est une version modifiée de l'algorithme **THAID** développé par Morgan et Messenger, 1973).

L'opérateur de l'arbre de décision **CHAID** fonctionne exactement comme l'opérateur de l'arborescence de décision avec une exception: il utilise un critère basé sur le chi-carré au lieu des critères de gain d'information ou de rapport de gain. De plus, cet opérateur ne peut pas être appliqué à des ensembles d'exemples avec des attributs numériques. Il est recommandé d'étudier la documentation de l'opérateur de l'arbre de décision pour une compréhension de base des arbres de décision.

Contrairement aux algorithmes du module [Modèles d'Arbres de Classification et de Régression (GCART)](http://www.statsoft.fr/concepts-statistiques/modeles-arbres-de-classification-et-regression/modeles-arbres-de-classification-et-regression.htm), la méthode **CHAID** va "construire" des arbres de décision non-binaires (c'est-à-dire des arbres de décision dans lesquels nous pouvons avoir plus de deux branches connectées à un même nœud (racine), en utilisant un algorithme assez simple qui est particulièrement bien adapté à l'analyse des fichiers de données les plus volumineux.

En outre, dans la mesure où l'algorithme **CHAID** va généralement produire de nombreuses tables de fréquences d'ordre multiple (par exemple, pour la classification d'une variable de réponse catégorielle avec de nombreuses catégories, sur la base de prédicateurs catégoriels avec de nombreuses modalités), il est devenu extrêmement populaire en marketing, pour des études de segmentation du marché.

1. **Présentation de CHAID**

**CHAID** est l'un des trois grands types d'algorithmes disponibles dans STATISTICA pour la construction d'arbres de décision ; les autres méthodes disponibles reposent sur les algorithmes **CART** (voir le module [Modèles d'Arbres de Classification et de Régression](http://www.statsoft.fr/concepts-statistiques/modeles-arbres-de-classification-et-regression/modeles-arbres-de-classification-et-regression.htm) ; voir aussi Breiman, et al., 1984) et QUEST (Quick, Unbiased, Efficient Statistical Trees ; voir le module Arbres de Classification ; voir aussi Loh et Shih, 1997).

En particulier, la plupart des questions évoquées dans la rubrique [GCART - Introduction - Principes Fondamentaux](http://www.statsoft.fr/concepts-statistiques/modeles-arbres-de-classification-et-regression/modeles-arbres-de-classification-et-regression.htm#introduction) s'appliquent également à ***GCHAID***: ces deux techniques permettent de construire des arbres de décision, dans lesquels chaque nœud (non-terminal) identifie une condition de division, afin de produire une prévision (des variables dépendantes continues) ou une classification (des variables dépendantes catégorielles) optimale.

Par conséquent, ces deux types d'algorithmes peuvent s'appliquer aussi bien à des problématiques de régression qu'à des problématiques de classification .

**Remarque :**

* **Valeurs manquantes**

 Les valeurs manquantes des variables prédictives sont traitées différemment dans le module **CHAID** et dans le module [Arbres de Décision Interactifs](http://www.statsoft.fr/concepts-statistiques/arbres-de-decision-interactifs/arbres-de-decision-interactifs.php). Dans la mesure où le module Arbres de Décision Interactifs ne reconnaît pas les matrices du modèle de type **ANCOVA**, il est plus souple pour gérer les valeurs manquantes.

Plus précisément, dans le module **CHAID**, les observations avec des valeurs manquantes dans au moins une des variables prédictives sont exclues du processus de construction de l'arbre de décision (même si vous demandez des substituts ; ces substituts ne sont utilisés que pour calculer les valeurs ou classifications prévues) ; dans le module [Arbres de Décision Interactifs](http://www.statsoft.fr/concepts-statistiques/arbres-de-decision-interactifs/arbres-de-decision-interactifs.php), les variables (et les valeurs manquantes pour ces variables) sont considérées une à une, et les observations possédant des valeurs manquantes sur les prédicateurs ne sont exclues du processus de construction de l'arbre de décision que si ces variables sont utilisées pour les divisions et qu'aucun substitut pertinent n'a été demandé ni sélectionné.

1. **Différenciation**

L'opérateur **CHAID** fonctionne exactement comme l'opérateur Décision Tree avec une exception: il utilise un critère basé sur le chi-deux au lieu des critères de gain d'information ou de rapport de gain.

De plus, cet opérateur ne peut pas être appliqué à des ensembles d'exemples avec des attributs numériques.

1. **Arbre de décision (Weight-Based)**

Si l'opérateur Poids par **Chi 2 Statistic** est appliqué pour la pondération d'attribut dans le sous-processus de l'opérateur Décision (Weight-Based), il fonctionne exactement comme l'opérateur **CHAID**.

1. **Input**

Ensemble de formation (tableau de données) :

Ce port d'entrée attend un ExempleSet. Il s'agit de la sortie de l'opérateur Generate Nominal Data dans le processus d'exemple joint. La sortie d'autres opérateurs peut également être utilisée comme entrée.

Cet opérateur ne peut pas gérer les données numériques, par conséquent, l'exemple ne doit pas avoir des attributs numériques.

1. **Output**

Modèle (arbre de décision)

L'arbre de décision **CHAID** est fourni à partir de ce port de sortie. Ce modèle de classification peut maintenant être appliqué sur des ensembles de données invisibles pour la prédiction de l'attribut label.

Ensemble d'exemples (tableau de données)

Le jeu d'exemple qui a été donné en tant qu'entrée est transmis sans passer à la sortie via ce port. Il est généralement utilisé pour réutiliser le même exemple dans d'autres opérateurs ou pour afficher l'exemple dans l'espace de travail Résultats.

1. **Paramètre :**

* **Minimal\_size\_for\_split :**

La taille d'un nœud est le nombre d'exemples dans son sous-ensemble. La taille du nœud racine est égale au nombre total d'exemples dans l'exemple. Seuls les nœuds sont divisés dont la taille est supérieure ou égale à la taille minimale pour le paramètre divisé.

* **Minimal\_leaf\_size :**

La taille d'un nœud feuille est le nombre d'exemples dans son sous-ensemble. L'arbre est généré de telle manière que chaque sous-ensemble de nœud feuille ait au moins le nombre minimal d'occurrences de feuilles.

* **Minimum\_gain :**

Le gain d'un nœud est calculé avant de le fractionner. Le nœud est divisé si son gain est supérieur au gain minimal. Des valeurs plus élevées de gain minimal entraînent moins de fentes et donc un arbre plus petit. Une valeur trop élevée empêche complètement le fractionnement et un arbre avec un seul nœud est généré.

* **Maximal\_depth**

La profondeur d'un arbre varie en fonction de la taille et de la nature de l'exemple. Ce paramètre est utilisé pour limiter la taille de l'arbre de décision. Le processus de génération d'arbres n'est pas poursuivi lorsque la profondeur de l'arbre est égale à la profondeur maximale.

Si sa valeur est définie sur '-1', le paramètre de profondeur maximale ne met aucune limite sur la profondeur de l'arbre, un arbre de profondeur maximale est généré.

1. Si sa valeur est définie sur '1', un Arbre avec un seul nœud est généré.

* **Confidence :**

Ce paramètre spécifie le niveau de confiance utilisé pour le calcul d'erreur pessimiste de l'élagage.

* **Number\_of\_prepruning\_alternatives :**

Comme la pré-pruning s'exécute parallèlement au processus de génération d'arbre, elle peut empêcher le fractionnement à certains nœuds lorsque le fractionnement à ce nœud n'ajoute pas à la puissance discriminative de l'arbre entier. Dans ce cas, des nœuds alternatifs sont essayés pour le fractionnement.

Ce paramètre ajuste le nombre de nœuds alternatifs essayés pour le fractionnement quand il est empêché par pré-élagage à un certain nœud.

* **No\_prepruning(no pré élagage)**

Par défaut, l'arbre de décision est généré avec pré-affranchissement. Si vous définissez ce paramètre sur true désactive le pré-affranchissement et délivre un arbre sans pré-affranchissement.

* **NO\_pruning( no élagage)**

Par défaut, l'arbre de décision est généré avec l'élagage. Si vous définissez ce paramètre sur true désactive l'élagage et délivre un arbre non pré-comprimé.

Un Algorithme Simple pour Construire des Arbres de Décision .

**L’algorithme procède de la manière suivante :**

**Préparation des prédicateurs**

Tout d'abord, Tanagra va créer des prédicteurs catégoriels à partir de chacun des prédicteurs continus, en répartissant la distribution des différents prédicteurs continus en un certain nombre de catégories d'effectifs sensiblement égaux. Pour les prédicteurs catégoriels, les catégories (classes) sont définies "naturellement".

**Regroupement de catégories**

Tanagra va ensuite examiner les prédicteurs afin de déterminer pour chacun, le couple de catégories (du prédicteur) les plus semblables (c'est-à-dire significativement moins différentes) par rapport à la variable dépendante ; pour les problèmes de classification (où la variable est également catégorielle), le programme va calculer un test du **Chi²** (Chi-deux de Pearson) ; pour les problèmes de régression (où la variable dépendante est continue), le programme va calculer des tests F.

Si le test respectif, pour un couple donné de catégories du prédicteur, ne peut être considéré comme significatif eu égard à une valeur alpha-de-fusion, le programme va alors regrouper les catégories correspondantes du prédicteur et répéter ce processus (c'est-à-dire, rechercher le couple suivant de catégories, qui à présent peuvent être des catégories précédemment fusionnées).

Si le couple respectif de catégories du prédicteur est statistiquement significatif (inférieur à la valeur correspondante du alpha-de-fusion), le programme va alors calculer (éventuellement) une valeur-p ajustée de Bonferroni pour l'ensemble des catégories du prédicteur respectif.

**Sélection d'une variable de division**

Tanagra va ensuite choisir pour la division, la variable prédictive qui possède la plus faible valeur-p ajustée, c'est-à-dire la variable prédictive qui permet de produire la division la plus significative ; si la plus petite valeur-p ajustée (Bonferroni) des prédicteurs est supérieure à une certaine valeur alpha-de-division, le processus de division prend fin, et le nœud respectif est un nœud terminal.

Ce processus se poursuit jusqu'à ce qu'il ne soit plus possible de réaliser d'autres divisions (compte tenu des valeurs alpha-de-fusion et alpha-de-division).

1. **L’algorithme C4.5**
2. **Présentation et historique**

L’**algorithme C4.5** est un [algorithme](https://fr.wikipedia.org/wiki/Algorithmique) de [classification](https://fr.wikipedia.org/wiki/Classification) supervisé, publié par [Ross Quinlan](https://fr.wikipedia.org/wiki/Ross_Quinlan). Il est basé sur l'[algorithme ID3](https://fr.wikipedia.org/wiki/Algorithme_ID3) auquel il apporte plusieurs améliorations.

À partir d'un échantillon d'apprentissage composé d'une variable objectif ou variable prédite **Y** {\displaystyle Y}et d'au moins une variable d'apprentissage ou variables prédictives **{x1, x2,…,xn}= X**{\displaystyle \{x\_{1},x\_{2},\ldots ,x\_{n}\}=X}, **C4.5** produit un modèle de type [arbre de décision](https://fr.wikipedia.org/wiki/Arbre_de_d%C3%A9cision).

Ce modèle permet de prédire pour un individu {\displaystyle i}la valeur estimée {\displaystyle {\hat {y\_{i}}}}de la variable objectif en fonction des valeurs prise par les variables "prédictives"{\displaystyle x\_{i}}.

L'algorithme **C4.5** se base sur une mesure de l'[entropie](https://fr.wikipedia.org/wiki/Entropie_(math%C3%A9matiques)) dans l'échantillon d'apprentissage pour produire le modèle ([graphe d'induction](https://fr.wikipedia.org/w/index.php?title=Graphe_d%27induction&action=edit&redlink=1)).

* L'avantage du recours à l'entropie est que l'algorithme opère sur des données symboliques que ce soient des variables catégorielles (comme des couleurs) ou numériques discrètes.
* Le désavantage de la méthode est que pour préserver l'efficacité de l'apprentissage et la pertinence du modèle produit, les variables continues doivent être discrétisées avant la mise en œuvre de l'algorithme.

**Quinlan** continua avec les versions **C5.0** et **See5** (**C5.0** pour les systèmes **UNIX** et **See5** pour Windows) qu'il commercialisa. **C5.0** améliore **C4.5** sur plusieurs points dont :

* la rapidité
* l'utilisation de la mémoire
* des arbres de décision plus petits

**C5.0** est un produit commercial dont le code source est disponible gratuitement pour l'interprétation et l'utilisation des arbres de décision et l'ensemble des règles qu'il produit.

**Algorithmes de force industrielle**

* Pour qu'un algorithme soit utile dans un large éventail d'applications réelles, il doit:
* Autoriser les attributs numériques
* Autoriser les valeurs manquantes
* Être robuste en présence de bruit
* Être en mesure de rapprocher les descriptions de concepts arbitraires (au moins en principe)
* Les systèmes de base doivent être étendus pour répondre à ces exigences

**C4.5 contient des mécanismes pour proposer trois types de tests:**

* Le test "standard" sur un attribut discret, avec un résultat et une branche pour chaque valeur possible de cet attribut.
* Si l'attribut Y a des valeurs numériques continues, un test binaire avec des résultats Y< Z et Y> Z pourrait être défini, basé sur la comparaison de la valeur de l'attribut avec une valeur de seuil Z

**L’algorithme C4.5**

Un test plus complexe basé aussi sur un attribut discret, dans lequel les valeurs possibles sont attribuées à un nombre variable de groupes avec un résultat et une branche pour chaque groupe.

1. **Impact :**

**Théorie de l’information Le gain informationnel (C4.5) :**

**Entropie de Shannon**

Quantité d’information pour connaître les valeurs de Y

**E (Y) =**

**Valeurs d'attribut inconnues**

Dans C4.5, on accepte le principe selon lequel les échantillons avec les valeurs inconnues **sont distribués de façon probabiliste en fonction de la fréquence relative des valeurs connues.**

Le nouveau critère de gain aura la forme suivante:

**Gain (x) = F (Info (T) - Infox (T))**

F = nombre d'échantillons dans la base de données avec une valeur connue pour un attribut donné / nombre total d'échantillons dans un ensemble de données

1. **Valeurs d’attributs numériques**

Méthode standard: divisions binaires

Par exemple. Temp <45

Contrairement aux attributs nominaux, chaque attribut comporte de nombreux points de partage possibles

La solution est simple:

Évaluer le gain d'information (ou autre mesure) pour chaque point de partage possible de l'attribut

Choisissez le meilleur point de partage

Le gain d'information pour le meilleur point de partage est le gain d'info pour l'attribut

**Éviter le tri répété**

* Trier les instances par les valeurs de l'attribut numérique
* Complexité du temps pour le tri: O (n log n)

**Q : Cela doit-il être répété à chaque nœud de l'arbre?**

**R: Non ! L'ordre de tri pour les enfants peut être dérivé de l'ordre de tri pour le parent**

* Complexité temporelle de la dérivation: O (n)
* **Inconvénient:** nécessité de créer et stocker un tableau d'indices triés pour chaque attribut numérique

1. **Manquant en tant que valeur distincte :**

Idée simple: traiter la valeur manquante comme une valeur distincte

**- Q: Quand ce n'est pas approprié?**

**- R: Lorsque des valeurs manquent pour des raisons différentes.**

- **Exemple 1:** l'expression génique pourrait être manquante quand elle est très haute ou très faible.

- **Exemple 2:** le champ est enceinte = manquant pour un patient masculin doit être traité différemment (non) que pour une patiente de 25 ans (inconnu).

**Valeurs manquantes - avancées**

Diviser les instances avec des valeurs manquantes en morceaux :

* Une pièce descendant une branche reçoit un poids proportionnel à la popularité de la branche.
* La somme des poids à 1.
* Info gain fonctionne avec des instances fractionnaires
* Utiliser des sommes de poids au lieu de compter
* Pendant la classification, diviser l'instance en morceaux de la même manière
* Fusionner la distribution de probabilité en utilisant des poids

1. **Elagage**

**Objectif:**

Empêcher le surmenage du bruit dans les données.

* Deux stratégies pour «tailler» l'arbre de décision:
  + Post élagage : prendre un arbre de décision pleinement développé et éliminer les pièces non fiables.
  + Pré élagage: cessé de cultiver une succursale lorsque l'information devient peu fiable.
* Post élagage préféré en pratique pré élagage peut "arrêter trop tôt"

1. **Pré- élagage**

* Basé sur un test de signification statistique :

Arrêtez de cultiver l'arbre quand il n'y a pas d'association statistiquement significative entre n'importe quel attribut et la classe à un nœud particulier

* **Test le plus populaire:** test chi-deux

- ID3 a utilisé le test chi-carré en plus du gain d'information

- Seuls les attributs statistiquement significatifs ont été autorisés à être sélectionnés par la procédure de gain d'information

1. **Post-élagage**

- Tout d'abord, construire l'arbre complet

- Ensuite, prune-le

- Arbre pleinement cultivé montre toutes les interactions d'attributs

- Problème: certains sous-arbres peuvent être dus à des effets fortuits

- Deux opérations d'élagage:

- Remplacement des sous-arbres

- Soulever des sous-arbres

- La stratégie possible:

- Estimation d'erreur

- Tests de signification

- Principe MDL

1. **Estimation des taux d'erreur**

* Taillez seulement si elle réduit l'erreur estimée.
* L'erreur sur les données de formation n'est-elle PAS un estimateur utile?

Q: Pourquoi cela entraînerait-il très peu d'élagage?

- Utiliser un ensemble de retenue pour l'élagage («élagage à erreur réduite»).

**Méthode C4.5**

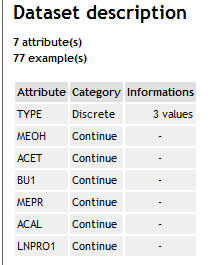
* Dériver l'intervalle de confiance des données d'entraînement.
* Utilisez une limite heuristique, dérivée de celle-ci, pour l'élagage.
* Méthode basée sur le processus de Bernoulli.
* Hypothèses statistiques timides (basées sur les données d'entraînement).

**Chapitre IV :** L’évaluation de la qualité (Comparaison)

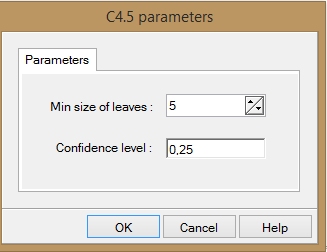
|  |  |  |  |
| --- | --- | --- | --- |
|  | **Choix de variable** | **Critère de segmentation** | **Profondeur** |
| **ID3** |  | Gain d’entropie |  |
| **CART** | Quantitatif | Indice de Gini | Post-élagage avec échantillion d’élagage |
| **CHAID** |  | Liendu Khi-2 / T de Tschuprow | Pré –élagage avec Test du Khi-2 |
| C4.5 |  | Entropie de Shannon  Gain informationnel  (Gain Ratio) | Post-élagage avec estimation pessimiste de l’erreur |

La méthode C4.5 :

En utilisant les mêmes données du fichier d’alcool, on retenons la variable à prédire dans Target et les variables prédictives dans INPUT.

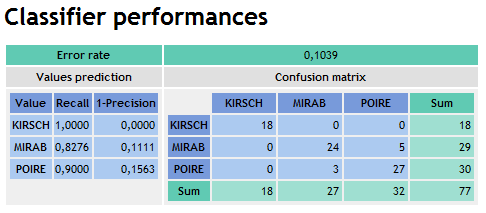


Paramaitre :

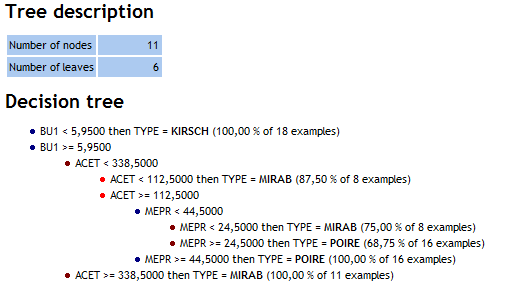


On fixe le nombre de feuille minimum en 5 (Post-élagage) pour éviter la complexité des résultats.

Puis on valide le menu View pour obtenir les resultats. La matrice de confusion en resubstitution, calculée sur les données en apprentissage (77 observations), est affichée en premier, puis le taux d’erreur apparent est 10.39% .



Nous nous rendons compte que les arbres 7 ou 6 feuilles sont assez proches de la solution retenue.



**Conclusion**

Dans cet article, nous avons considéré deux algorithmes de classification, à savoir, CART, et C4.5. Nous avons comparé leur performance à la fois en termes de classification et de généralisation

Sur un exemple étudiée à propos les composants d’alcool. On constate que

Le C4.5 une fois convergé, en général, a une meilleure classification et généralisation de précision par rapport à CART. D'autre part, il est également noté que les erreurs de prédiction faites par chaque algorithme sont différentes. Cela indique qu’il peut y avoir une possibilité de combiner ces algorithmes de telle sorte que leur exactitude des prédictions pourrait être améliorée. Ceci est présenté comme un défi pour la future recherche.