Chapitre 1

Les arbres de décision

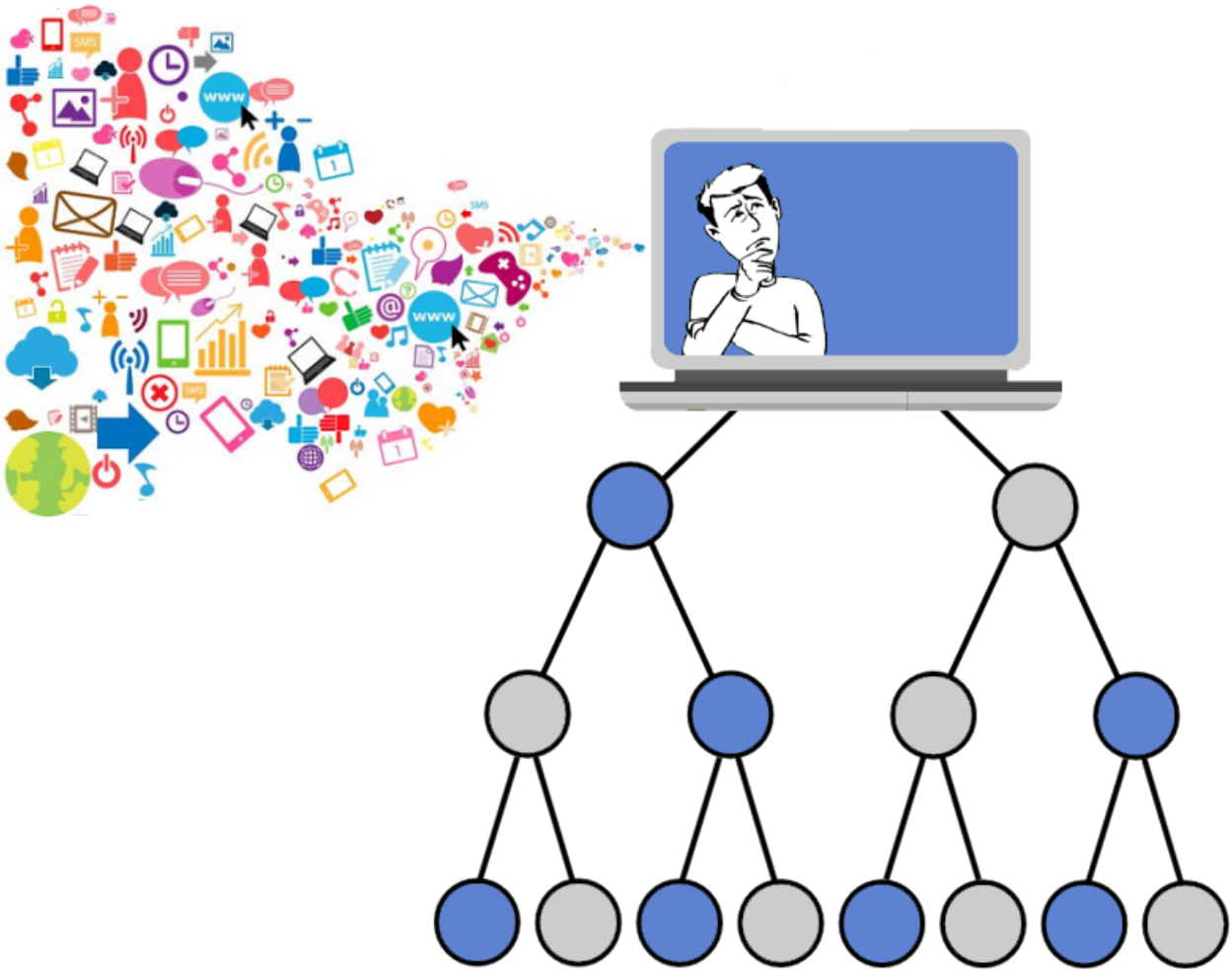

Table des matières

1.1 Introduction

De nos jours, la donnée est tellement importante que le défi n'est pas seulement de la stocker, mais aussi de la traiter, dans le but d'en extraire le maximum de connaissances, aussi bien prédictives que descriptives. Ce processus est appelé *Data Mining*, il englobe beaucoup de techniques, dont les arbres de décision, que nous allons traiter durant ce chapitre, en définissant les notions clés, en présentant quelques algorithmes très utilisés, avec en conclusion, une étude comparative.

1.2 Data Mining

Traduit en fouille ou forage de données, considéré par beaucoup de chercheur comme l'un des sujets de recherche clés dans les systèmes de base de données et d'apprentissage automatique [**553155**]. Le *Data Mining* est défini comme étant un ensemble de techniques permettant extraire de la connaissance utile, à partir d'un grand volume de données à l'état brute.

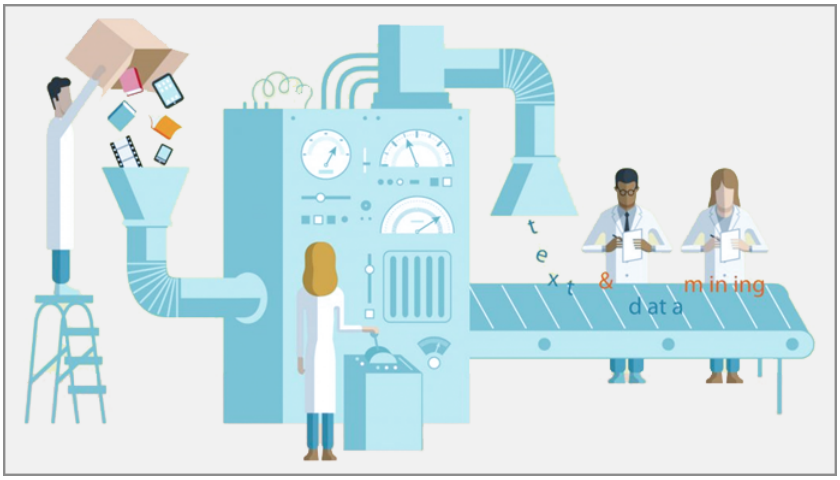

FIGURE 1.1 – Data Mining

Le *Data Mining* repose principalement sur des méthodes qui peuvent être classées en deux catégories, la caractéristique utilisée pour cette classification est le fait que la machine, dispose ou non d'un échantillon à apprendre avant la création du modèle.

— Apprentissage supervisé
L'idée consiste à se baser sur un ensemble de données d'apprentissage, afin de créer un modèle $Z = (X, Y)$, en mappant entre un ensemble de variables d'entrée X, et une variable de classe Y. L'application de ce modèle a pour but de prédire, de manière automatique, les classes de nouvelles données [**wolpert2002supervised**].

Ce type d'apprentissage concerne essentiellement :

1. La classification : ou la classe à prédire est de type discret
2. La régression : ou la classe à prédire est de type continu
Parmi les techniques les plus utilisées d'apprentissage supervisé, nous pouvons citer :
— Les réseau de neurones.
— Les arbres de décision.

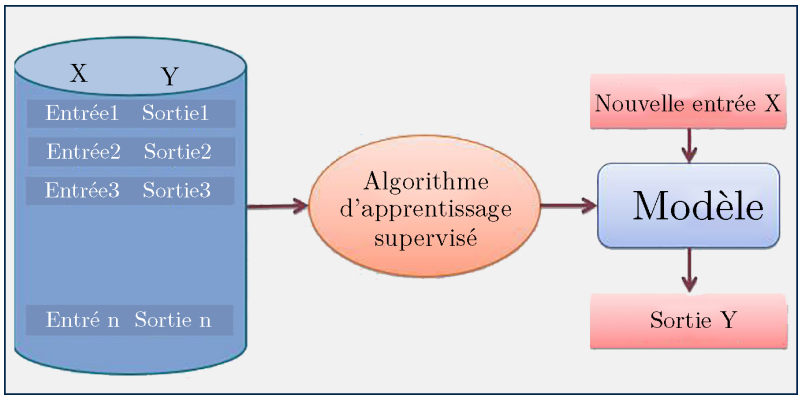

FIGURE 1.2 – Apprentissage supervisé

— Apprentissage non supervisé
L'apprentissage non supervisé étudie comment les systèmes peuvent apprendre à représenter des schémas d'entrée particuliers d'une manière qui reflète la structure statistique de la collection globale de données[**dayan1999unsupervised**], sans se baser sur un modèle crée au préalable.
Parmi les techniques d'apprentissage non supervisé, nous pouvons citer :
— La segmentation : qui, a partir d'un ensemble hétérogène, permet de créer des sous groupe homogènes.
— Les associations : qui permet de découvrir des association entre les instances d'un tres grand volume de données. Exemple : Les clients qui achètent le produit P_1 et P_2, achètent aussi P_3

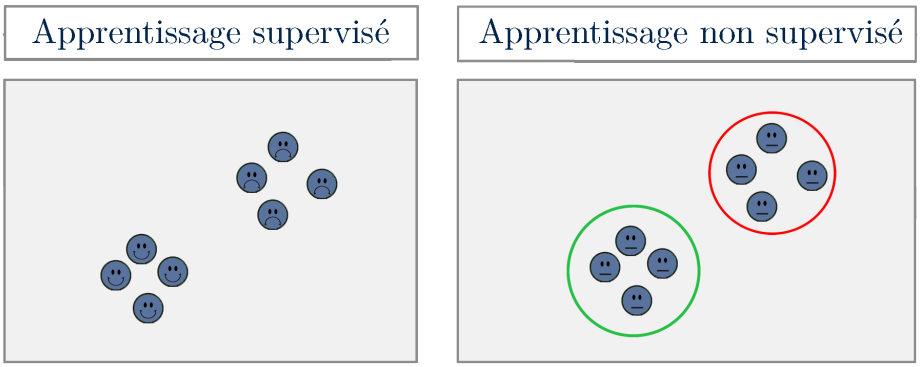

FIGURE 1.3 – La différence entre l'apprentissage supervisé et non supervisé

1.3 Définition

Faisant partie des méthodes de classification supervisées, le but des arbres de décision est aussi de créer un modèle afin de pouvoir classer automatiquement de nouvelles données. La construction de ce modèle se fait à travers des partitions récursives sur l'ensemble de données d'apprentissage, jusqu'à arriver à des sous ensembles homogènes par rapport a la variable de classe. Ainsi, l'ensemble de données d'apprentissage est transformé en un arbre, dans lequel chaque composent a une signification particulière :

1. Racine : comprend tout l'ensemble de données d'apprentissage.
2. Nœud interne : représente un test.
3. Nœud terminal (feuille) : représente une décision finale.

1.3.1 Formalisation

Soit $\mathscr{T}$ l'ensemble des tuples possibles et $\mathscr{Y}$ l'ensemble des valeurs de classes.

Definition 1 (Condition sur un tuple) *Une condition C est une fonction $\mathscr{T} \rightarrow \{true, false\}$.*

Definition 2 (Arbre de décision) *L'ensemble des arbres de décision $\mathscr{A}$ est défini inductivement par :*

- *$Noeud(F) \in \mathscr{A}$ si $F = \{(A_1, C_1), \ldots, (A_k, C_k)\}$ tel que les A_i sont des arbres de décisions et les C_i sont des conditions telles que pour tout tuple t, il existe un unique i tel que $C_i(t) = true$.*
- *$Feuille(Y) \in \mathscr{A}$ si $Y \in \mathscr{Y}$ est une valeur de classe.*

Definition 3 (Classification locale) *Soit $A = Noeud(\{(A_1, C_1), \ldots, (A_k, C_k)\}$ et soit t un tuple. On défini $classificationLocale(A, t) = A_i$ si i est le seul indice tel que $C_i(t) = true$.*

Le rendu du modèle est très aisément interprétable, ce qui fait l'une des principales forces des arbres de décision, comme le montre la figure ci-dessous.

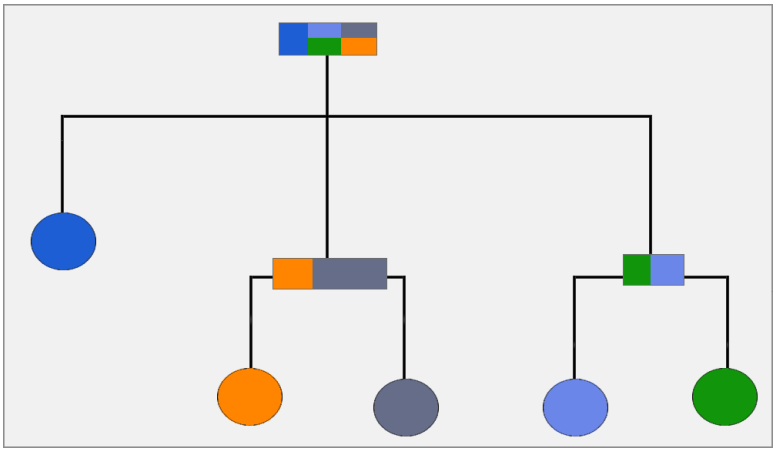

FIGURE 1.4 – Structure d'un arbre de décision

A tout arbre de décision est associé une procédure de classification, cette association est définie en parcourant l'arbre de haut en bas, passant par tous les tests menant de la racine à la feuille, qui, une fois atteinte, correspondra à la classe qui sera attribuée à la nouvelle instance.

Definition 4 (Classification par un arbre de décision) *La classification associée par un arbre de décision A à un tuple t est définie par la fonction $classifie : \mathscr{A} \times \mathscr{T} \rightarrow \mathscr{Y}$*

$$classifie(A, t) = \begin{cases} classifie(A', t) & \text{si } A = Noeud(F) \text{ et } A' = classificationLocale(A, t) \\ Y & \text{si } A = Feuille(Y) \end{cases}$$

Definition 5 (Equivalence sémantique d'abres de décision) *Soient A_1 et A_2 deux arbres de décision. A_1 est sémantiquement équivalent à A_2 (noté $A_1 \equiv A_2$) si pour tout tuple $t \in \mathscr{T}$, $classification(A_1, t) = classification(A_2, t)$.*

L'algorithme 1 représente le déroulement global de la construction d'un arbre de décision, nous remarquons à la ligne 6 qu'à chaque niveau, si le nœud n'est pas homogène, un test doit être choisi afin de diviser l'ensemble de donnée relatif a ce nœud. Le choix du test est considéré comme la partie cruciale de l'algorithme, vu son impacte directe sur l'arbre de décision résultant, ce choix est basé sur des métriques appelées *critères de division.*

Algorithm 1 : Algorithme de création d'un arbre de décision

Entrée : Échantillon S
00 : DÉBUT
01 : **RÉPÉTER**
02 : Décider si le noeud courant est terminal
03 : **SI** le nœud est terminal **ALORS**
04 : Lui affecter une classe
05 : **SINON**
06 : Sélectionner un test et créer autant de nœud fils qu'il y a de réponses possibles au test
07 : **FinSI**
08 : Passer au nœud suivant non exploré s'il existe
09 : **JUSQU'À** obtenir un arbre de décision
10 : FIN

1.4 Critères de division

L'objectif principal d'un critère de division est de réduit l'impureté d'un nœud, et ce, en choisissant l'attribut approprié sur lequel diviser l'ensemble de données.
Il existe plusieurs critères, c'est en grande partie ce qui différencie les algorithmes d'arbres de décision, nous allons détailler les plus utilisés d'entre eux, en utilisant de manière commune le jeu de données suivant :

Temps	Température	Humidité	Vent	Jouer
Ensoleillé	Haute	Haute	Faux	Non
Ensoleillé	Haute	Haute	Vrai	Non
Couvert	Haute	Haute	Faux	Oui
Pluvieux	Basse	Haute	Faux	Oui
Pluvieux	Moyenne	Normal	Faux	Oui
Pluvieux	Moyenne	Normal	Vrai	Non
Couvert	Moyenne	Normal	Vrai	Oui
Ensoleillé	Basse	Haute	Faux	Non
Ensoleillé	Moyenne	Normal	Faux	Oui
Pluvieux	Basse	Normal	Faux	Oui
Ensoleillé	Basse	Normal	Vrai	Oui
Couvert	Basse	Haute	Vrai	Oui
Couvert	Haute	Normal	Faux	Oui
Pluvieux	Moyenne	Haute	Vrai	Non

FIGURE 1.5 – Ensemble de données de départ

Gain d'information

Le gain d'information est un critère basé sur l'impureté, il utilise l'entropie comme mesure d'impureté (mesure mise au point par QUINLAN en 1987)[**singh2014comparative**]. Dans le cadre de la recherche de l'attribut de division, il correspondra à celui ayant le gain d'information le plus élevé.

Formule

Le Gain d'information d'un attribut T est égal a la différence entre l'entropie pré-division (nœud père p), et l'entropie post-division (par rapport à l'attribut T), sa formule est la suivante :

$$Gain(p,T) = Entropie(P) - \sum_{j=1}^{n} (P_j * Log_2(P_j)) \tag{1.1}$$

Exemple

— Calcul de l'entropie de l'ensemble
Nombre d'instance : 14

$$Nombre\ de\ catégories\ de\ classe\ :\ 2 \begin{cases} Oui : & 9 \\ Non : & 5 \end{cases}$$

$Entropie(P) = -((\frac{9}{14})Log_2(\frac{9}{14}) + (\frac{5}{14})Log_2(\frac{5}{14})) = 0.94$

— Temps

	Oui	Non	Nb d'instance
Ensoleillé	2	3	5
Pluvieux	3	2	5
Couvert	4	0	4

TABLE 1.1 – Table de comptage de l'attribut Temps

$Entropie(Temps) = \frac{5}{14}(-(\frac{2}{5}Log_2(\frac{2}{5}) + \frac{3}{5}Log_2(\frac{3}{5}))) + \frac{5}{14}(-(\frac{3}{5}Log_2(\frac{3}{5}) + \frac{2}{5}Log_2(\frac{2}{5}))) + \frac{4}{14}(-\frac{4}{4}Log_2(\frac{4}{4}))$
$Entropie(Temps) = 0.69$

— $Gain(Temps) = 0.94 - 0.69 = 0.25$

— Après les calculs :
  — Gain(Température) = 0.05
  — Gain(Humidité) = 0.15
  — Gain(Vent) = 0.05

  L'attribut ayant le gain le plus élevé sera choisi comme attribut de division, ce qui est dans notre exemple l'attribut "Temps"

Rapport du gain d'information

Mise au point par QUINLAN, le but de ce critre normaliser le Gain d'information, afin palier à la faille de ce dernier, qui a tendance a favoriser les attributs ayant beaucoup de valeur[**salzberg1994c4**].

Formule

La mesure *SplitInfo* est utilisée afin de normaliser le Gain d'information, elle représente les informations générées suite a la division de l'ensemble de données d'apprentissage p, en n partitions correspondantes aux différentes valeurs de l'attribut T, la formule est la suivante :

$$SplitInfo(p,T) = -\sum_{j=1}^{n} P_j * log_2(P_j) \tag{1.2}$$

$$GainRatio(p,T) = \frac{Gain(p,T)}{InfoSplit(p,T)} \tag{1.3}$$

Exemple

— Calcul de la mesure InfoSplit

— Temps
Nous avons :

$$Nombre\ de\ modalité : 3 \begin{cases} Ensoleillé : & 5 \\ Pluvieux : & 5 \\ Couvert : & 4 \end{cases}$$

$InfoSplit(Temps) = -(\frac{5}{14}Log_2(\frac{5}{14}) + \frac{5}{14}Log_2(\frac{5}{14}) + \frac{4}{14}Log_2(\frac{4}{14}))$
$InfoSplit(Temps) = 1.577$

— Après les calculs :
— InfoSplit(Température) = 1.577
— InfoSplit(Humidité) = 1
— InfoSplit(Vent)= 0.98

— Calcul du Rapport de Gain

— $GainRatio(Temps) = \frac{Gain}{InfoSplit} = \frac{0.25}{1.577} = 0.15$
Après les calculs :
— $GainRatio(Température) = 0.03$
— $GainRatio(Humidité) = 0.15$
— $GainRatio(Vent) = 0.151$
Nous remarquons que pour le même jeu de données, le rapport de gain a donné des résultats différents que le gain d'information, par conséquent, l'attribut Temps, au trois valeurs, a été pénalisé, ce qui a donné deux possibilités de choix : Temps, humidité.

Le coefficient de Gini

Autre mesure de l'impureté, le coefficient de Gini mesure les divergences entre les distributions de probabilité des valeurs de l'attribut cible. Sa valeur varie entre 0 et 1, il est égal à 0 dans une situation d'égalité parfaite entre les individu de l'ensemble, et est égal à 1 dans dans la situation la plus inégalitaire possible. Le coefficient de Gini a été utilisé dans divers travaux, tels que (Breiman et al., 1984) et (Gleanedet al., 1991)[**singh2014comparative**]

Formule

$$Gini(S) = 1 - \sum_{j=0}^{n} 1 - P_j^2 \tag{1.4}$$

Ou P_j représente la fréquence relative à la classe j dans l'ensemble du nœud père.

Exemple

— Calcul des coefficients de Gini
— Gini(Temps = ensoleillé)
Nous avons :

$$Nombre\ d'instance : 5 \begin{cases} P(vrai) : & \frac{2}{5} \\ P(faux) : & \frac{3}{5} \end{cases}$$

— $Gini(Temps = ensoleillé) = 1 - ((\frac{2}{5})^2 + (\frac{3}{5})^2) = 0.48$

— Après les calculs :
  — $Gini(Temps = pluvieux) = 1 - ((\frac{3}{5})^2 + (\frac{2}{5})^2) = 0.48$
  — $Gini(Temps = couvert) = 1 - ((\frac{5}{5})^2 = 0$
  — $Gini(Temperature = haute) = 0.5$
  — $Gini(Temperature = basse) = 0.32$
  — $Gini(Temperature = moyenne) = 0.39$
  — $Gini(Humidité = haute) = 0.408$
  — $Gini(Humidité = normal) = 0.489$
  — $Gini(Vent = vrai) = 0.5$
  — $Gini(Vent = faux) = 0.378$

  La plus faible valeur du coefficient de Gini correspond au test "Temps=couvert", il sera donc choisi comme attribut de division.

Chi-square

Le Chi-square est un test statistique, il permet, à partir d'un tableau de contingence, de comparer l'indépendance des variables, afin de savoir si elles sont liées, autrement dis, il permet de vérifier si les distribution des variables différent les unes des autres.

Formule

Le calcule du Chi-square (X^2) repose sur deux tableau, à savoir :
— Tableau d'observation
— Tableau d'estimation

$$\chi^2 = \sum_{i=0}^{n} \frac{(Observation_i - Estimation_i)^2}{Estimation_i} \tag{1.5}$$

Exemple

— Calcul du χ^2
  — Temps
    — Construction des tables

	Oui	Non	Total
Ensoleillé	2	3	5
Pluvieux	3	2	5
Couvert	4	0	4
Total	9	5	14

TABLE 1.2 – Tableau d'observation

	Oui	Non
Ensoleillé	$\frac{5*9}{14} = 3.21$	$\frac{5*5}{14} = 1.78$
Pluvieux	$\frac{5*9}{14} = 3.21$	$\frac{5*5}{14} = 1.78$
Couvert	$\frac{4*9}{14} = 2.57$	$\frac{4*5}{14} = 1.43$

TABLE 1.3 – Tableau d'estimation

— $\chi^2(Temps) = \frac{(2-3.21)^2}{3.21} + \frac{(3-1.78)^2}{1.78} + \frac{(3-3.21)^2}{3.21} + \frac{(2-1.78)^2}{1.78} + \frac{(4-2.57)^2}{2.57} + \frac{(0-1.43)^2}{1.43}$

— $\chi^2(Temps) = 3.55$
— Après les calculs :
  — $\chi^2(Température) = 0.62$
  — $\chi^2(Humidité) = 2.8$
  — $\chi^2(Vent) = 0.94$

L'attribut ayant la valeur χ^2 la plus élevée sera choisi comme attribut de division, ce qui est dans ce cas, l'attribut Temps.

1.5 Élagage

Les arbres de décision sont souvent confrontés à deux problèmes, à savoir :

— La complexité (en terme de taille)
Qui se répercute sur l'interprétabilité de l'arbre de décision.

— Le sur-apprentissage
Qui se caractérise par une bonne précision de l'arbre sur l'ensemble d'apprentissage, ainsi qu'une mauvaise sur d'autres ensembles.

Afin de palier à ces problème, l'arbre de décision doit être élagué. L'élagage peut se faire de deux manières différentes, pendant la croissance de l'arbre (pre-élagage)ou bien après sa construction (post-élagage)

— Pré-élagage
Consiste à établir une liste de règles d'arrêt, afin d'empêcher la croissance des branches qui n'améliore pas la précision prédictive de l'arbre[**esposito1997comparative**], parmi les réglés d'arrêt nous pouvons citer (entres autres) :
  — Seuil minimal d'instance
  — Profondeur maximal de l'arbre
  — Se baser sur le test d'indépendance (χ^2)
  Le χ^2 permet de tester statistiquement l'indépendance d'une variable, le calcul se fait comme suit :

    — Calcul du χ^1
    — Détermination du DDL (Degré De Liberté)
    $DDL = (Nombre\ de\ modalité\ d'attribut - 1) * (Nombre\ de\ catégorie\ de\ classe)$
    — Déterminer le seuil d'erreur α
    α est un paramètre que l'utilisateur devra choisir, généralement, $\alpha = 5\%$
    — A partir de la table statistique dite t table, récupérer la valeur du $\chi_{critique}$, qui correspond à l'intersection des deux valeur, DDL et α.
    — Comparer $\chi^2 avec \chi_{critique}$
      — Si $\chi^2 > \chi_{critique}$
      La distribution de l'attribut est statistiquement non significative.

	α	0.20	0.15	0.1	0.05	0.025	0.01	0.005	0.001	0.0005
Degré de liberté	1	1.376	1.965	3.078	6.314	12.71	31.82	63.66	318.3	636.6
	2	1.061	1.386	1.886	2.920	4.303	6.965	9.925	22.33	31.60
	3	0.978	1.250	1.638	2.353	3.182	4.541	5.841	10.21	12.92
	4	0.941	1.190	1.533	2.132	2.776	3.747	4.604	7.173	8.610
	5	0.920	1.156	1.476	2.015	2.571	3.365	4.032	5.893	6.869
	6	0.906	1.134	1.440	1.943	2.447	3.143	3.707	5.208	5.959
	7	0.896	1.119	1.415	1.895	2.365	2.998	3.499	4.785	5.408
	8	0.889	1.101	1.397	1.860	2.306	2.896	3.355	4.501	5.041
	3	0.883	1.100	1.383	1.833	2.262	2.821	3.250	4.297	4.781

FIGURE 1.6 – T table

— Post-élagage
Dans ce cas, l'élagage est appliqué à un arbre de décision complètement construit, en supprimant certain sous arabes, dans le but de réduire son taux d'erreur.
L'effet bénéfique de l'élagage a attiré l'attention de nombreux chercheurs, qui ont proposés un certain nombre de méthodes[**esposito1997comparative**], parmi ces méthodes nous pouvons citer :

— *REP* : *Reduced Error Prunning*
Proposée par QUINLAN, et globalement la plus simple a implémenté[**quinlan1987simplifying**].
elle consiste a découpé le jeu de données en trois parties, deux tiers servirons à la construction de l'arbre, et le tiers restant, appelé "ensemble de test" sera utilisé pour l'élagage.
Une fois l'arbre construit, l'élagage se fait comme suit :
Au niveau de chaque sous arbre T', l'algorithme compare l'estimation de l'erreur réelle avant et après sa suppression, si la précision de l'arbre ne réduit pas apres suppression de T', ce dernier sera élagué et remplacé par la feuille au nombre d'instance le plus élevé de sa hiérarchie.
A noter que l'estimation de l'erreur réelle est calculée avec la formule suivante :

$$Erreur(T) = \frac{Nombre\ d'instance\ mal\ classer}{Nombre\ total\ d'instance} \tag{1.6}$$

1.6 Les algorithmes d'arbre de décision

Il existe une variété d'algorithme d'arbre de décision, aussi diverse que les critères de division que nous avons citer, nous allons voir avec plus de détails les plus utilisés d'entre eux.

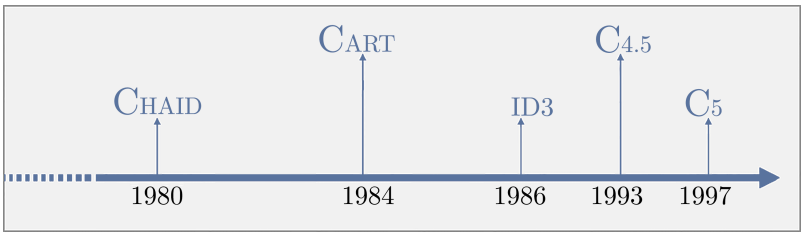

FIGURE 1.7 – Apparition des algorithmes d'arbre de décision par ordre chronologique

1.6.1 *CHAID : Chi-Squared Automatic Interaction Detection*

Un des plus anciens algorithmes d'arbres de décision, proposés en 1980 par KASS.
Comme son nom l'indique, cet algorithme utilise le test du χ^2 comme critère de division[**wilkinson1992tree**].
En dépit de la puissance et de la robustesse du χ^2, son aspect statistique rend la qualité des résultats relative au volume de données d'apprentissage, en d'autres termes, CHAID est beaucoup plus adapté aux grands qu'aux petits volume de données.
Il utilise le pré-élagage en se basant sur le test d'indépendance χ^2, afin de déterminer si le test est statistiquement signifiant ou pas.

1.6.2 *CART : Classification and Regression Trees*

Mis au point par le statisticien LEO BREIMAN en 1984, il est considéré comme une des références incontournable des arbres de décision. *CART* utilise *le coefficient de Gini* comme critère de division, cette dernière est de type binaire, chaque test partitionne l'ensemble de données en deux sous ensemble, ceux pour lesquels la réponse est positive et négative. L'arbre obtenu est ensuite élagué en utilisant la méthode post-élagage *CCP* (*Cost–Complexity Pruning*.
CART peut gérer tout types d'attribut (continu ou discret) ainsi que des instances a valeur manquante ou aberrantes[**Hssina2014ACS**].

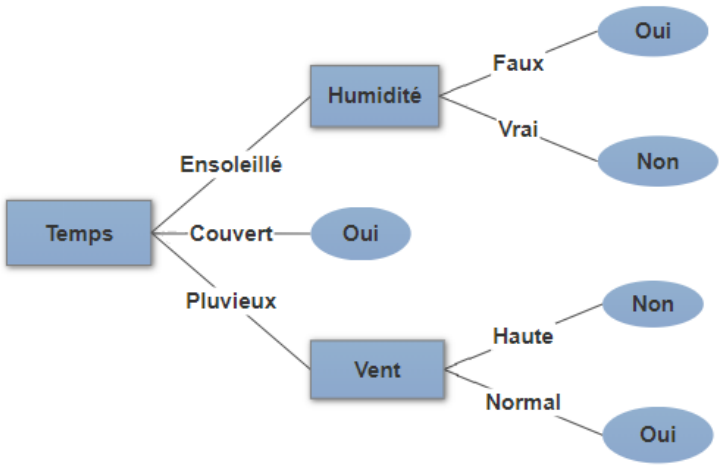

FIGURE 1.8 – Exemple d'arbre de décision crée en utilisant CHAID

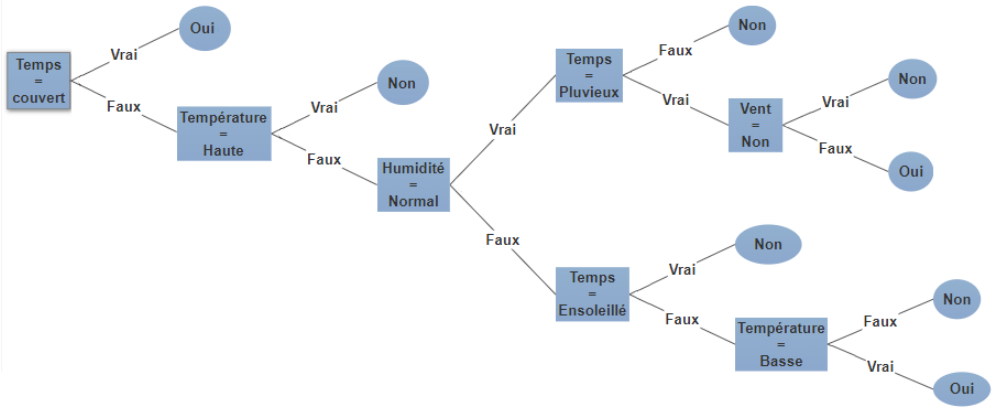

FIGURE 1.9 – Exemple d'arbre de décision crée en utilisant CART

1.6.3 *ID3 : Iterative DiChaudomiser 3*

Mise au point par ROSS QUINLAN en 1986, après l'avoir publié en 1975 dans un libre intitulé « Machine Learning ». Parmi les plus simples des algorithmes d'arbres de décision.
Il utilise le Gain d'information comme critère de division, il ne gère que les attributs discrets ainsi que des instances à valeurs connues[**Hssina2014ACS**].
ID3 utilise la méthode pre-élagage du test d'indépendance χ^2, avec comme condition que l'attribut soit statistiquement significatif afin d'être sélectionné comme attribut de division.

1.6.4 C4.5

Proposé en 1993 par le même ROSS QUINLAN, comme amélioration de son premier algorithme (ID3). C4.5 se base sur le Rapport de Gain comme critère de division. L'arbre de décision crée sera ensuite élagué en utilisant la méthode *EBP (Error-Based Pruning)*[**quinlan2014c4**]. Les améliorations apportées (par rapport à ID3) sont les suivantes[**:**]

— Gestion des attribut continus
Tâche réalisée en quatre temps :

1. Trier les valeurs de l'attribut par ordre croissant.
2. Créer un seuil pour chaque couple de valeurs consécutives ($A = \frac{Val1+Val2}{2}$).

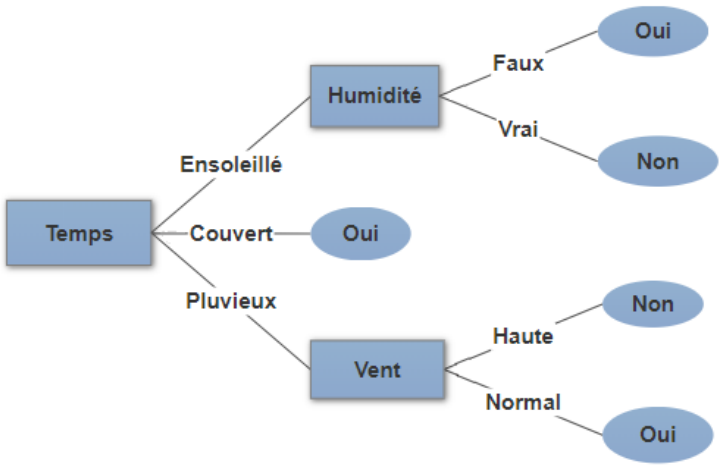

FIGURE 1.10 – Exemple d'arbre de décision crée en utilisant ID3

3. Pour chaque seuil :
   — Rendre la liste des attributs ayant une valeur supérieur et inférieur ou égal au seuil
   — Calculer l'information apportée par l'attribut.
4. Le seuil qui apportera le maximum d'information sera choisi.

— Gestion des attributs aux valeurs manquantes

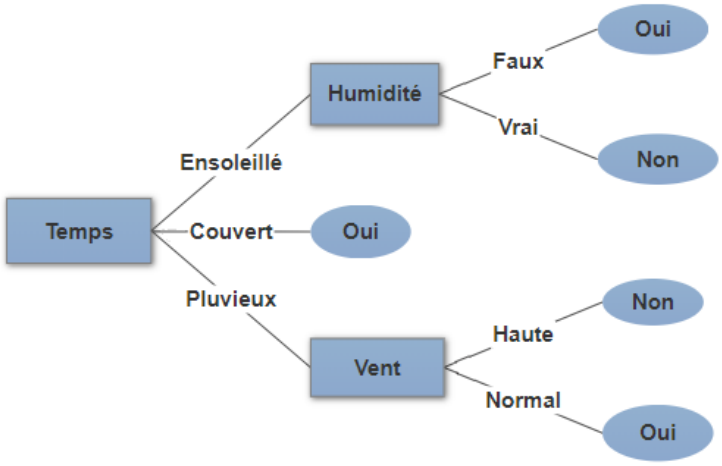

FIGURE 1.11 – Exemple d'arbre de décision crée en utilisant C4.5

1.6.5 C5

Une amélioration de l'alorithme C4.5, mis au point en 1997 par le même ROSS QUINLAN.
Etant une version commerciale, les détails de l'implémentation n'ont pas tous été révélés, neaumoins, à travers ce qui à été publié, nous pouvons énumérer les améliorations suivantes[**wu2008top**] :

- Utilise l'apprentissage ensembliste
  Qui consiste à partitionner les données d'apprentissage et de créer un modèle relatif à chaque partie. L'ensemble des modèles sera utilisé lors de la classification, et se, en faisant l'agrégation de leurs résultat.
- Nouveau type de données (heure, date)

Le coté technique a aussi été amélioré, parmi les points fort du C5 (par rapport au C4.5)[**pandya2015c5**] :

- Rapidité
- Faible utilisation de la mémoire
- Meilleur taux de précision

1.6.6 Récapitulatif

	CHAID	CART	ID3	C4.5	C5
Critère de division	χ^2 ($Chi-square$)	Coefficient de Gini	Gain d'information	Rapport du gain	Rapport du gain
Type de division	Multiple	Binaire	Multiple	Multiple	Multiple
Type d'attribut	Discrets et continus	Discrets et continus	Discrets	Discrets et continus	Discrets et continus
Élagage	Pre-élagage : test du χ^2	Post-élagage : Cost Complexity Pruning	Pre-élagage : test du χ^2	Post-élagage : Error-Based Pruning	Binomial Confidence Limit
Valeur manquante	Oui	Oui	Non	Oui	Oui

TABLE 1.4 – Récapitulatif des fonctionnalités de chacun des algorithmes

1.6.7 Comparaison

La recherche faite nous emmène à déduire que la force de chacun de ces algorithmes est relative à un contexte particulier, qui est le jeu de données d'apprentissage, la comparaison entre algorithmes ne peut donc se faire qu'en fonction de ce contexte.
Rappelons que dans les arbres de décision, le modèle est crée qu'en faisant la correspondance entre les entrées et les sorties des instances d'apprentissage, ce modèle aura donc +/- la même qualité que celle de l'ensemble de données utilisé. Cet ensemble peut être défini par le biais d'un ensemble de paramètres, tel quel :

— Taille
  Qui est exprimée en nombre d'instance, elle varie d'un type de traitement à un autres, selon l'ampleur de ce dernier.
— Attribut
  Plusieurs critères peuvent être associés aux attributs :
  — Nombre d'attribut

    - Nombre de modalité d'attribut
    - Type d'attribut (continu ou discret)
- Classe
  Qui correspond à l'attribut à prédire, elle est défini par :
    - Un nombre de catégorie
      Qui peut être binaire ou N-aire.
    - Type
      Qui, dans le cas d'une classification, sera de type discret, et dans le cas d'une régression, sera de type continu.

C'est à travers ces caractéristiques que les performances des algorithme peuvent être mesurées, à l'aide de plusieurs métriques, telles que :

- Précision
  Représente un ratio entre le nombre de prédiction bien faite, et le nombre total de prédiction, ce ratio est estimé à partir de l'ensemble de test, qui n'à pas été utilisé lors de l'apprentissage.
- Complexité
  Représente la taille de l'arbre de décision conçu, il peut être mesure en nombre de nœud, ou en profondeur de l'arbre.
- Temps d'exécution
  Représente le temps écoulé lors de la création du modèle ou lors de la classification.

Lors de la création d'un modèle, il est très important d'utiliser les courbes, dites "courbes d'apprentissage", afin de suivre ces performances. et ce, dans le but de mieux estimer les caractéristiques qui lui seront adéquates.

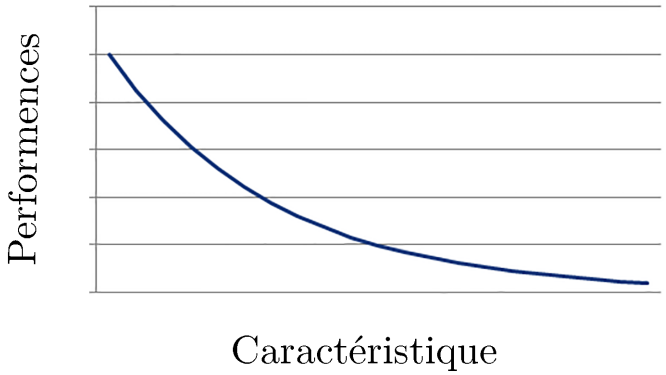

FIGURE 1.12 – Courbe d'apprentissage

Nous allons voir, à travers le tableau ci-dessous, quels sont les conditions appropriées pour chacun des algorithmes vu précédemment.

	Taille	Attribut	Classe
CHAID	Précision ✓	Temps ×	Temps ×
	Adapté au grand volume de données	Non adapté au grand nombre de modalité d'attribut, car : ça fera augmenter le nombre de ligne de la table de contingence	Non adapté au grand nombre de catégorie de classe, car : ça fera augmenter le nombre de colonne de la table de contingence
CART	✓	Complexité ×	Complexité✓
	Indifférent à la variation de taille, car : Sa formule est légère	Non adapté au grand nombre de modalité d'attribut, Car : Son critère divise par valeur d'attribut et non par attribut	Adapté aux classes à catégorie binaire
ID3	Temps ×	Complexité ×	✓
	Non adapté au très grand volume, à cause : Des logarithmes utilisés dans le calcule du gain	Non adapté au nombre réduit de modalité, car : il utilise le Gain d'information comme critère de division Non adapté aux attributs de type continu	Indifférent à la variation de la classe
C4.5	Temps ×	Complexité ✓	✓
	Non adapté au très grand volume, du au calcul : — Gain — InfoSplit	Adapté au nombre élevé de modalité (grâce du Rapport de gain)	Indifférent à la variation de la classe

1.7 Conclusion

.....
.....
.....