Université de Montréal

Un système combiné de raisonnement à base de cas et de data mining : Une alternative aux techniques statistiques de marketing bancaire

par
Natalie Guay
Département d'Informatique et de Recherche Opérationnelle
Faculté des arts et des sciences

Mémoire présenté à la Faculté des Études Supérieures
en vue de l'obtention du grade de
Maître ès sciences (M.Sc)
en informatique

Juillet 2001

Université de Montréal

Faculté des Études Supérieures

Ce mémoire intitulé :

Un système combiné de raisonnement à base de cas et de data mining :

Une alternative aux techniques statistiques de marketing bancaire

Présenté par :

Natalie Guay

A été évalué par un jury composé des personnes suivantes :

Président-rapporteur :	Claude Frasson
Directrice de recherche :	Esma Aïmeur
Membre du Jury :	Guy Lapalme

Mémoire accepté le : 22 août 2001

Sommaire

Il vous est sûrement déjà arrivé de recevoir une offre de carte de crédit par courrier ou encore une proposition de plan de REER par téléphone en provenance de votre banque. Ces techniques de marketing sont communes à toutes les entreprises, mais l'efficacité de ces campagnes de publicité peut varier énormément. Si vous recevez ces offres au moment où vous vous dites « J'aurais éventuellement besoin d'une carte de crédit avec une plus grande marge. » ou « Il faudrait que je communique avec ma banque pour leur demander un plan pour mes REER. », c'est que votre banque a bien fait son travail! Pour obtenir cette efficacité, une classification doit être effectuée sur les profils des clients de l'entreprise afin de cibler les clients potentiels pour chaque produit offert. Dans la majorité des entreprises, les listes des clients à contacter pour vendre un certain produit ou un certain service sont habituellement générées à l'aide de méthodes statistiques. Ces méthodes complexes et peu intuitives ne sont à la portée que de statisticiens experts en la matière. En effet, cette façon de modéliser le comportement des clients demande des connaissances poussées en mathématiques et en statistique afin d'obtenir des modèles valables et ainsi parvenir à personnaliser les interactions entre une entreprise et ses clients.

L'objectif de ce travail est de présenter une technique permettant aux employés n'ayant pas nécessairement de base en statistique de déterminer eux-mêmes quels clients cibler, ceci dans le but d'améliorer l'efficacité du marketing de leurs produits et services. Nous démontrerons que le *Raisonnement à Base de Cas* (RBC) peut être une alternative intéressante aux modèles et méthodes statistiques actuellement utilisés pour la classification des profils de clients. De plus, nous aurons recours à une autre technique appelée *Data Mining* (DM). Les bases de données sur des clients, qu'elles soient bancaires ou autres, contiennent des quantités impressionnantes de données, ce qui en complique la manipulation et l'interprétation. Le DM nous permet justement d'analyser aisément ces très grandes quantités de données pour en faire ressortir les caractéristiques nécessaires et suffisantes à la généralisation des profils de clients.

Le principe sur lequel notre système combiné de RBC et de DM est basé est simple. Lorsqu'un nouveau client se présente, le système cherche tout d'abord à fixer la catégorie de son profil à l'aide d'un algorithme de DM, ce qui permet d'accélérer le processus de classification. Le RBC prendra la relève si la technique de DM échoue : le système cherchera dans sa *base de cas* les profils (ou *cas*) les plus similaires et, en se basant sur ces derniers, classera le profil du nouveau client de la façon la plus appropriée. Ensuite, en interprétant ce classement, l'entreprise saura quels produits et services sont les plus susceptibles d'intéresser un certain type de client et sera en mesure de concevoir des campagnes de publicité plus profitables. Du point de vue du marketing, des listes de clients potentiellement intéressés par certains produits ou services pourront être générées plus simplement et ce directement par les équipes de marketing plutôt que par des statisticiens. Ainsi, un temps précieux est gagné et certaines erreurs dues à un manque de communication et de compréhension entre les experts des différents domaines sont évitées!

Pour développer ce projet, nous avons travaillé en collaboration avec une grande banque canadienne qui nous a fourni un échantillon des profils de ses clients ainsi que le résultat de la classification statistique de ces derniers (*catégorie de marketing*). Pour des raisons de confidentialité, nous référerons tout au long de ce mémoire à cette banque sous le nom de « banqueX » et les exemples donnés seront issus de profils de clients réels légèrement modifiés.

NOTE : L'emploi du genre masculin sera utilisé tout au long de ce texte pour désigner également les genres masculin et féminin. Cette convention n'a pour but que d'alléger le texte.

Table des matières

Table des figures

Liste des tableaux

Liste des abréviations

AD	Arbre de Décision
BD	Base de Données
BT	British Telecommunications
CALIBRE	Candidate LIBrary REtrieval
CART	Classification And Regression Trees
CHAID	Chi-squared Automatic Interaction Detection
DM	Data Mining
IA	Intelligence Artificielle
IR	Induction de Règles
JDBC	Java Data Base Connectivity
KDD	Knowledge Discovery in Databases
MDL	Minimum Description Length
MOP	Memory Organisation Packet
OLF	Office de la Langue Française
PPV	Plus Proches Voisins
PRP	Probability Ranking Principle
RBC	Raisonnement à Base de Cas
SQL	Structured Query Language

Remerciements

Je tiens tout d'abord à remercier ma directrice de recherche, le professeur Esma Aïmeur, pour sa grande confiance, ses judicieux conseils ainsi que pour la précieuse aide matérielle et intellectuelle qu'elle m'a apportée tout au long des deux années de ma maîtrise. Merci au CIRANO et tout spécialement à monsieur Jacques Robert sans qui rien de tout cela n'aurait débuté. J'ai énormément apprécié les défis intellectuels et le support financier qu'ils m'ont apportés au tout début de ma maîtrise.

Je remercie aussi les experts de la banque avec laquelle nous avons collaboré pour leur disponibilité et leur ouverture d'esprit.

J'adresse un merci bien spécial à ma mère pour avoir lu et relu ce mémoire d'innombrables fois, pour son esprit critique mais surtout pour son support inconditionnel. Merci de toujours m'avoir soutenu.

Finalement, je ne remercierai jamais assez mon copain, Simon. Merci pour ton support moral, ta simple présence et ta patience infinie avec moi.

Introduction

Nam et ipsa scientia potestas est (Knowledge is power).

- Francis Bacon

Il fut une époque où les interactions entre une entreprise et ses clients n'étaient pas très compliquées. Le meilleur exemple est certainement celui du « magasin général du village » où le propriétaire s'occupait de tout. Ce genre de magasin n'avait bien souvent aucune concurrence. Le propriétaire connaissait chacun de ses clients par son nom et savait exactement quels produits chacun préférait. Les interactions étaient alors directes et de type 1:1, c'est-à-dire qui le client avait un service personnalisé directement entre lui et le propriétaire. Aujourd'hui, nous sommes bien loin du petit magasin général! Certaines entreprises ont pris une ampleur considérable offrant bien souvent leurs produits ou services à toute une nation et parfois les exportant même vers d'autres pays. L'élargissement des marchés et la concurrence ne permettent plus aux entreprises de connaître aussi bien tous leurs clients. Ainsi le mode d'interaction personnalisé 1:1 est maintenant chose du passé. Les interactions sont devenues impersonnelles et les entreprises ne connaissent plus les besoins de chacun de leurs clients.

Les entreprises modernes se retrouvent donc aux prises avec un nouveau problème que le magasin général n'avait pas. Celui-ci peut être énoncé de la façon suivante :

Qui a besoin de quoi et quand ?

Pour répondre à cette question, les entreprises ont recueilli de très grandes quantités de données sur leurs clients et ont développé des modèles de comportements de façon statistique grâce à l'aide de statisticiens qualifiés. Cette méthode donne de bons résultats et le niveau de personnalisation vis-à-vis le client s'améliore. En revanche, depuis une dizaine d'années, les entreprises sont confrontées à certains problèmes dus entre autres aux limitations des statistiques :

- L'évolution de l'informatique a permis aux entreprises d'avoir accès à des outils de stockage de données simples et dont le coût est peu élevé. Ainsi, la quantité de données concernant les clients ne cesse de grandir. Le nombre de variables et de clients devient tellement imposant que l'on commence à se poser des questions sur les résultats obtenus via des modèles développés statistiquement. Les méthodes statistiques ont été principalement utilisées avec des bases de données relativement réduites. Nous sommes donc en droit de nous demander si ces techniques sont encore valables à de si grandes échelles [HAND 98, FRI 97].

- De plus en plus de dirigeants d'entreprises et de départements de marketing n'ayant pas ou peu de connaissances statistiques veulent comprendre les données qu'ils possèdent sur leurs clients et les fouiller eux-mêmes. Si une méthode plus simple et intuitive existait, les campagnes de publicité seraient mises sur pied beaucoup plus rapidement et directement par les experts en marketing qui s'y connaissent bien mieux en stratégies publicitaires que les statisticiens.

Dans ce travail, nous suggérons une méthode simple et intuitive qui se veut soit une alternative ou encore mieux un complément aux techniques statistiques actuellement employées dans certaines entreprises. Cette méthode est capable d'analyser de grandes quantités de données et de répondre à une partie de la question énoncée ci-dessus c'est-à-dire: *Qui a besoin de quoi ?* La partie *Quand* est mise de côté pour l'instant, car elle est plus facilement prévisible « à l'œil » d'un expert. Notre projet est développé pour le

domaine bancaire, mais il serait éventuellement applicable au marketing de tout autre type d'entreprises.

En une phrase, l'objectif visé par ce travail est de:

Proposer une alternative simple, efficace et intuitive aux méthodes statistiques pour la classification de profils de clients dans le but de déterminer leurs besoins en produits et services bancaires.

Pour atteindre notre objectif nous utiliserons le *Raisonnement à Base de Cas* (RBC) [KOL 93, URL 2] couplé aux *Arbres de Décisions* (AD) [BRE *et al.* 84, HAR 75, QUI 93], une technique de *Data Mining*[1] (DM) [HAN et KAM 01, URL 3, URL 4]. Le RBC et les AD sont des techniques possédant toutes les caractéristiques nécessaires pour atteindre ce but. Leur fonctionnement est simple et intuitif et nous verrons qu'il est aussi efficace.

À l'intérieur de notre prototype, les AD seront tout d'abord utilisés pour classer de nouveaux profils de clients selon leurs caractéristiques générales. Cette technique est issue du domaine de l'*apprentissage machine* [MIC *et al.* 83, MIC *et al.* 86, MIC *et al.* 98] et permet de générer des règles nous fournissant des informations sur le schéma général des profils de clients. Les règles ainsi obtenues nous aideront à effectuer un « premier » classement des profils des clients. En effet, il se peut qu'un AD soit dans l'impossibilité de classer un profil. Dans ce cas, le RBC prend la relève et classe ce profil selon ses caractéristiques spécifiques plutôt que générales.

Pour ce faire, le RBC possède une *base de cas*, cette dernière constitue le cœur de tout système de RBC. Plus précisément, les *cas* contenus dans cette base sont en fait les « profils de clients typiques » de la banque avec laquelle nous travaillons. Cette banque sera appelée « banqueX » tout au long de ce travail pour des raisons de

[1] Ou *exploration de données* selon l'*Office de la Langue Française* (OLF) [URL 5]. Le terme Data Mining étant largement reconnu, nous l'utiliserons tout au long de ce travail.

confidentialité. Par la suite, le processus de RBC cherchera les profils typiques les plus semblables au nouveau profil du client à classer. Puis, le nouveau profil sera classé dans le groupe le plus vraisemblable selon le ou les cas les plus similaires retrouvés dans la base de cas. Tous les détails techniques du fonctionnement de notre système RBC ainsi que notre utilisation des AD seront expliqués dans les chapitres suivants.

Notre projet se divise donc en deux grandes étapes : le DM et le RBC. Le tableau suivant (Tableau 0.1) présente brièvement les différentes tâches effectuées lors de ces deux étapes. Ces tâches sont plus longuement décrites dans les chapitres deux et trois. Une fois la base de cas du RBC et l'ensemble de règles des AD bien constitués, nous serons en mesure d'identifier le « *groupe de marketing* » (ou *catégorie*) du nouveau profil d'un client et de dresser la liste des produits et des services les plus susceptibles de l'intéresser. Une autre possibilité serait de générer une liste de candidats et faire une campagne de publicité ciblée vers ce sous-ensemble de clients.

Tableau 0.1 : Principales tâches des modules de DM et de RBC

DM	▪ Prétraitement des données. ▪ Construction des règles représentant les structures importantes de la base de données des profils de clients à l'aide des arbres de décision. ▪ Classification selon les règles (si impossible voir RBC).
RBC	▪ Recherche dans la base de cas des cas similaires à un nouveau profil de client. ▪ Classification selon les cas similaires.

L'apport de ce projet se situe à deux niveaux différents. Premièrement, il suggère une approche permettant de faciliter l'utilisation de bases de cas de grandes tailles. En effet, l'utilisation d'un système de classification de profils n'employant que le raisonnement à base de cas nécessite généralement plus de temps de calcul qu'un système utilisant conjointement les arbres de décision et le raisonnement à base de cas.

Le data mining pourrait par conséquent permettre l'utilisation du RBC même sur de larges bases de données, apportant ainsi la possibilité d'augmenter le nombre d'attributs dans les cas et le nombre de cas formant la base.

Deuxièmement, l'utilité du raisonnement à base de cas dans le domaine du marketing, que ce soit pour une banque ou pour d'autres types d'entreprises, a été très peu explorée. Ainsi, notre travail lance le RBC dans de nouvelles avenues encore aujourd'hui pratiquement inexplorées.

Ce travail est divisé en quatre chapitres. Le premier chapitre présente l'état de l'art en ce qui a trait à ce genre de système. Le deuxième chapitre donne plus de détails sur les méthodes de DM. Il introduit également la technique de data mining des arbres de décision que nous privilégions dans ce travail. Puis, le fonctionnement du processus de RBC est ensuite présenté dans le chapitre trois.

Le prototype hybride que nous avons réalisé est présenté au chapitre quatre. Nous y décrivons les étapes de son développement ainsi que ses deux modules principaux : le module de data mining et celui de raisonnement à base de cas. De plus, nous exposons les procédures d'évaluation de notre prototype ainsi que les résultats obtenus. Les améliorations futures et les extensions possibles au prototype développé forment la partie finale de ce quatrième chapitre.

Finalement, la conclusion résume le fonctionnement, les objectifs et les résultats obtenus par le prototype développé dans le cadre de ce travail.

Chapitre 1 : État de l'art

Quot homines, tot sententiae (Autant d'hommes, autant d'avis).

- Térence

Les systèmes de RBC sont souvent utilisés dans un but de classification. Par exemple, un expert en évaluation du risque de crédit d'une banque qui veut classer le nouveau profil d'un client pour savoir s'il est risqué de lui accorder un prêt, pourrait bénéficier d'un système de RBC pour effectuer sa classification. Un autre exemple de classification à l'aide de systèmes de RBC serait celui d'un médecin qui classerait un patient selon ses symptômes pour découvrir quel traitement lui convient le mieux. De notre côté, notre but est de classer les clients d'une banque afin de déterminer quels produits et services ont la plus grande probabilité de les intéresser.

Dans ce chapitre, nous présentons ce qui a été développé à ce jour en matière de systèmes combinant l'utilisation du data mining [HAN et KAM 01, URL 3, URL 4] et du raisonnement à base de cas [KOL 93, URL 2] dans le domaine qui nous concerne : la classification.

Deux systèmes implémentant une approche combinée de data mining et de

raisonnement à base de cas sont présentés dans ce chapitre. Le système CALIBRE [FAG et BLO 99] utilise ces deux techniques de façon séquentielle. Il construit la base de cas initiale à l'aide de techniques de data mining et ensuite il n'effectue que du raisonnement à base de cas à partir de sa base initiale. Le système ELEM2-CBR [CER *et al.* 99] quant à lui intègre complètement les deux techniques l'une dans l'autre. En effet, ELEM2-CBR utilise chaque fois les deux techniques de façon conjointe. Présentons sans plus tarder ces deux techniques.

1.1 CALIBRE

L'utilisation du RBC dans le domaine du marketing en est à ses premiers balbutiements. Après une recherche exhaustive, les seuls travaux ayant été publiés et appliqués au domaine du marketing sont ceux de M. Fagan et K. Bloor des laboratoires de la BT (*British Telecommunications*) effectués en 1999 [FAG et BLO 99]. La raison pour laquelle il est difficile de trouver des publications sur ce genre de système est bien simple : le domaine du marketing est un milieu plutôt fermé et contrôlé par des ententes de confidentialité. Les entreprises ne dévoilent donc que très peu d'informations sur leurs stratégies de marketing afin de conserver leur compétitivité. C'est pourquoi nous ne présentons qu'un système appliqué au domaine du marketing dans cette section sur l'état de l'art.

Les deux chercheurs de la BT, Fagan et Bloor, ont développé un système de RBC appelé CALIBRE (Candidate LIBrary REtrieval). Ce système est utilisé afin de générer des listes de candidats cibles pour des campagnes de promotion. Dans leur papier, les auteurs précisent que ce système a été développé et implémenté de façon à ce que ses utilisateurs soient des experts en marketing. Ainsi des employés n'ayant pas ou peu de connaissances en statistique ou en mathématiques peuvent aisément générer des listes de marketing à l'aide de CALIBRE.

Le but premier du prototype de Fagan et Bloor est de réduire le temps de

génération des listes de clients cibles et d'augmenter la « rétroaction » entre les gens de marketing et la liste générée. Pour atteindre cet objectif, le prototype CALIBRE utilise conjointement le RBC et le DM et est implanté sous la forme de pages Web. Les deux figures suivantes (Figure 1.1 et Figure 1.2) adaptées du papier de Fagan et Bloor [FAG et BLO 99] représentent les étapes nécessaires pour générer la liste de clients avant l'utilisation du système CALIBRE et après. Nous remarquons que l'utilisation du système CALIBRE réduit énormément le nombre d'intermédiaires entre le département de marketing et la liste elle-même. De plus, un lien entre la liste de candidats et le système a été rajouté pour effectuer une « rétroaction ». Regardons ces deux processus d'un peu plus près.

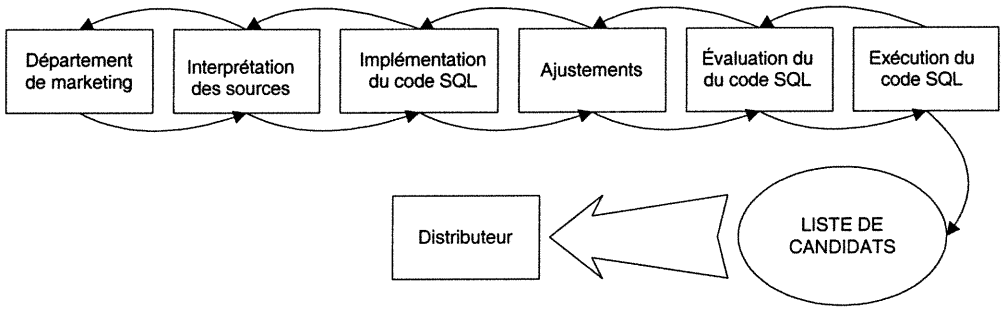


Figure 1.1 : Ancien processus de génération de listes de candidats

L'ancien processus consiste à produire du code SQL à partir de différentes sources de discussion entre plusieurs intervenants (Ex. : courriel). Le langage SQL est utilisé pour formuler différentes requêtes dans des bases de données. Ce processus de génération des listes de clients est long et peu fiable. Les besoins et les informations transmises entre les membres du département de marketing vont probablement être modifiés entre les étapes d'interprétation des sources et d'exécution du code comme dans le jeu du « téléphone arabe ». Il se peut donc que la liste finale ne corresponde plus aux exigences de départ. Ce phénomène est en partie dû au fait que ce ne sont pas des experts du domaine du marketing qui effectuent toutes les étapes intermédiaires mais que cette tâche est plutôt donnée à des experts en SQL.

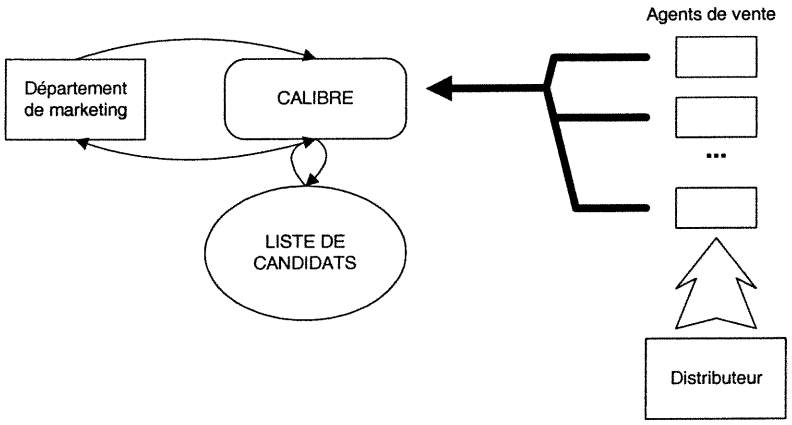


Figure 1.2 : Processus de génération de listes de candidats avec CALIBRE

Le nouveau processus suggéré dans ce papier présente l'avantage de réduire le nombre d'intermédiaires entre le département de marketing et la liste des candidats. De plus, le processus bénéficie maintenant d'une « rétroaction » provenant des agents de vente (flèche en gras). Cet ajout permet d'ajuster le système CALIBRE pour qu'il tienne compte de ses réussites et de ses échecs dans la génération de nouvelles listes. Cet « apprentissage » n'était pas présent dans l'ancien processus; on y repartait à zéro chaque fois que l'on générait une nouvelle liste au risque de répéter d'anciennes erreurs! CALIBRE permet en plus de construire la liste de candidats de façon itérative, c'est-à-dire que les experts en marketing peuvent expérimenter différentes idées jusqu'à ce qu'ils soient satisfaits de la liste générée.

La Figure 1.3 représente l'architecture du système CALIBRE. On remarque trois modules différents : celui de data mining, celui contenant les bases de données et celui du serveur RBC. Ce système utilise différentes techniques de DM sur différents entrepôts de données. L'extraction d'informations est effectuée avec les outils de DM du SAS Institute [URL 11]. Le processus de segmentation à l'aide du DM permet de diviser les données en groupes. Ces différents groupes formeront les cas des différentes bases de cas (bases de données ORACLE) utilisées pour le RBC.

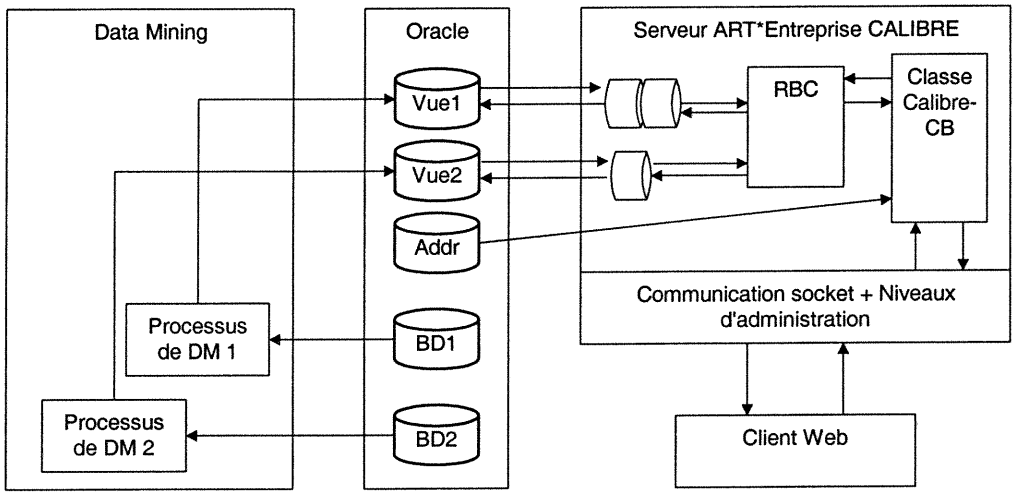


Figure 1.3: Architecture du système CALIBRE

Le module du serveur RBC comporte plusieurs composantes. Le processus de RBC a été construit avec les outils fournis dans un logiciel d'aide au développement de RBC appelé ART*Entreprise[2]. La classe « Calibre-CB » est un niveau d'abstraction que les auteurs ont ajouté à l'outil ART*Entreprise. Cette classe de haut niveau permet principalement de créer plusieurs systèmes de RBC de ART*Entreprise en les gérant comme un seul grand système. Les composantes de communication sont dirigées par des sockets de type Berkeley [STE 90] et les niveaux d'administration Web offrent la possibilité de branchements multiples et gèrent toutes les questions de sécurité.

La composante principale du module du serveur RBC est le processus de raisonnement à base de cas. La mesure de similarité utilisée pour retrouver les cas les plus semblables à un nouveau cas est calculée par la fonction standard fournie avec l'outil ART*Entreprise [BRI 96]. Deux formules sont utilisées, une pour les données textuelles et une pour les données numériques :

[2] ART*Entreprise est un outil qui est produit par la compagnie californienne Brightware [URL 8].

- Données textuelles :

$$score\_attribut_{f,i} = mmw_{f,i} + \frac{msf_{f,i}}{tsf_f} * \left(mw_{f,i} - mmw_{f,i}\right)$$

- Données numériques :

$$score\_attribut_{f,i} = mw_{f,i} - \frac{\left|cv_{f,i} - pcv_f\right|}{mdev_{f,i}} * \left(mw_{f,i} - mmw_{f,i}\right)$$

où

- $mw_{f,i}$ représente le poids accordé si l'attribut f du cas i est identique à celui du nouveau cas.
- $mmw_{f,i}$ représente le poids accordé si l'attribut f du cas i n'est pas identique à celui du nouveau cas.
- $msf_{f,i}$ représente le nombre de sous-attributs de l'attribut f du cas i qui sont identiques à ceux du nouveau cas.
- tsf_f représente le nombre total de sous-attributs de l'attribut f du nouveau cas.
- $cv_{f,i}$ représente la valeur numérique de l'attribut f du cas i.
- pcv_f représente la valeur numérique de l'attribut f du nouveau cas.
- $mdev_{f,i}$ représente la déviation de l'attribut f du cas i.

De plus, le système CALIBRE offre la possibilité de générer des rapports et présente une interface Web simple et intuitive.

Précisons que les auteurs du papier sur le système CALIBRE n'ont eux-mêmes trouvé qu'une seule autre référence concernant l'utilisation du RBC dans le domaine du marketing. Cette référence nous est malheureusement inaccessible, car elle est directement reliée aux documents concernant le produit ART*Entreprise de la compagnie Brightware.

En résumé, le but des chercheurs de la British Telecommunications était d'améliorer le processus de génération de liste de candidats dans le domaine du marketing. Ils suggèrent dans leur papier l'utilisation d'une combinaison des techniques de DM et de RBC pour créer un système appelé CALIBRE qui réduit le nombre d'intermédiaires entre le département de marketing et la production de la liste. Ce projet en est encore au stade de développement et aucun test n'a encore été effectué avec CALIBRE.

1.2 ELEM2-CBR

Le système dont nous allons discuter dans cette section n'est pas appliqué au domaine du marketing. En revanche, il présente une approche intéressante à la classification de cas. Le système ELEM2-CBR a été développé par trois chercheurs canadiens et offre une architecture hybride combinant le raisonnement à base de cas et la technique d'*Induction de Règles* (*IR*) [CER *et al.* 99]. Ils suggèrent cette approche combinée afin d'améliorer les performances de systèmes n'appliquant qu'une de ces deux méthodes. En effet, les auteurs expliquent comment les méthodes de RBC et d'IR se supportent mutuellement dans leurs faiblesses grâce à leurs propriétés complémentaires. Le RBC fournit des connaissances spécifiques sur les cas tandis que l'IR fournit un savoir plus général ce qui donne un équilibre renforcé au système. Le système ELEM2-CBR peut effectuer deux tâches : la classification et la prédiction de valeurs numériques. Nous ne discuterons que de la tâche de classification dans cette section, la tâche de prédiction ne nous concernant pas dans ce travail.

ELEM2-CBR effectue une classification en utilisant, comme nous venons de le mentionner, un mélange de deux techniques : l'induction de règles et le raisonnement à base de cas. La première action effectuée par ce système est de générer un ensemble de règles de classification à partir d'un ensemble de données en utilisant une méthode d'IR nommé ELEM2 [AN *et al.* 95, AN 97, AN et CER 98]. Cet ensemble de règles sera utilisé comme point de départ pour la classification de nouveau cas. Plusieurs situations

peuvent survenir lors de cette étape. Un nouveau cas peut être couvert par une ou plusieurs règles. S'il n'est couvert que par une seule règle, le cas sera classé dans la catégorie suggérée par cette dernière (Figure 1.4, a) . Lorsque le nouveau cas est couvert par plusieurs règles et que toutes ces règles suggèrent la même catégorie, le cas y est classé (Figure 1.4, b). Par contre, si les règles suggèrent différentes catégories (Figure 1.4, c), le processus de RBC prendra la relève et classera le nouveau cas à l'aide de deux formules différentes calculant la similarité entre le nouveau cas et les cas de la base de cas : *l'affectation des poids* et la *mesure de similarité*.

Nouveau cas (5 attributs) : $a_1 = 2, a_2 = 46, a_3 = 12, a_4 = 1, a_5 = 0$

	Nombre de règles couvrant ce cas :	
(a)	1	règle #1: $a_1 = 2, a_2 > 10$ alors catégorie = A
(b)	2	règle #1: $a_1 = 2, a_2 > 10$ alors catégorie = A règle #2: $a_1 = 2, a_3 > 11, a_4 >= 1, a_5 < 6$ alors catégorie = A
(c)	2	règle #1: $a_1 = 2, a_2 > 10$ alors catégorie = A règle #2: $a_1 = 2, a_3 > 1, a_4 >= 1, a_5 < 9$ alors catégorie = B

Figure 1.4 : Exemple de classification à l'aide de règles.

L'*affectation des poids* aux attributs d'un cas est une étape très importante en RBC. Ces poids permettent au système de retrouver le bon cas dans le bon contexte. Les auteurs utilisent une adaptation d'une méthode appelée *Probability Ranking Principle* (*PRP*) [COO 73]. Les poids du nouveau cas sont donc fixés selon le calcul suivant, adapté du PRP:

$$w(av_i) = \log \frac{r(N-n-R+r)}{(n-r)(R-r)}$$

Dans cette formule, N représente le nombre de cas dans la base de cas. Parmi ces N cas, R sont représentatifs, c'est-à-dire que ces cas représentent bien le nouveau cas à classer. La variable n représente le nombre de fois où la paire *attribut-valeur* av_i (attribut : a_1 et sa valeur : 2) est présente dans N et la variable r le nombre de fois où elle est présente dans R. Le nombre de cas dits représentatifs est obtenu avec ELEM2 [AN *et al.* 95, AN 97, AN et CER 98].

Une fois les poids et les règles obtenus, la recherche des cas les plus similaires peut être entreprise. Les auteurs utilisent une *mesure de similarité* qui leur permet de déterminer quels cas de la base sont les plus similaires au nouveau cas :

$$Similarité(x,q) = \sum_{i=1}^{n} w_i \times Simil(x_i, q_i)$$

Dans cette formule, n représente le nombre d'attributs, x_i représente la valeur du i ème attribut du cas x et q_i représente la valeur du i ème attribut du nouveau cas à classer. La valeur de w_i représente le poids du i ème attribut du nouveau cas selon la fonction d'affectation des poids. La valeur de la fonction $Simil(x_i,q_i)$ est déterminée selon l'équation suivante :

$$Simil(x_i, q_i) = \begin{cases} 0 & \text{si } x_i \neq q_i \text{ et que la valeur de l'attribut est qualitative;} \\ 1 & \text{si } x_i = q_i \text{ et que la valeur de l'attribut est qualitative;} \\ 1 - |norm(x_i) - norm(q_i)| & \text{si la valeur de l'attribut est quantitative.} \end{cases}$$

norm signifie que les valeurs de x_i et q_i sont normalisées.

Voici les différentes étapes à suivre afin d'effectuer la classification avec ELEM2-CBR :

1. Trouver dans l'ensemble de règles générées par ELEM2 la ou les règles compatibles avec le nouveau cas entré dans le système.

2. Si une seule règle est compatible, alors le nouveau cas est classé dans la catégorie C indiquée par la règle.

3. Si plusieurs règles sont compatibles et qu'elles sont toutes dans la même catégorie, alors le nouveau cas est classé dans la catégorie indiquée par ces règles.

4. Si plusieurs règles sont compatibles et qu'elles sont dans des catégories différentes ou qu'aucune règle n'est totalement compatible avec le nouveau cas (les règles partiellement compatibles sont utilisées dans ce cas) alors le processus de RBC intervient. Les cas de la base de cas sont ordonnés selon les méthodes d'affectation des poids et de mesure de similarité décrites ci-dessus. Si des règles compatibles ou partiellement compatibles sont retrouvées, aller à l'étape 6.

5. Si aucune règle n'est partiellement compatible alors les cas sont aussi ordonnés à cette étape, selon les méthodes d'affectation des poids et de mesure de similarité. Par contre, on fixera $R = r = 0$ dans la fonction d'affectation des poids.

6. Sélectionner les k cas les plus intéressants (qui ont le plus grand degré de similarité) parmi les cas ordonnés. La valeur de k est déterminée par l'usager. Ces cas formeront l'ensemble S.

7. Si tous les cas dans S sont dans la même catégorie, alors le cas est classé dans cette dernière.

8. Sinon, pour chaque catégorie C_i présente dans l'ensemble de règles S, il faut effectuer le calcul suivant (*Decision Score*):

$$DS(C_i) = \sum_{j=1}^{m} Similarité(c_j, q)$$

où c_i est un des m cas dans S qui représente la catégorie C_i et la variable q représente le nouveau cas.

9. Classer le nouveau cas dans la catégorie ayant obtenu le meilleur résultat (valeur la plus grande) à l'étape 8.

Les étapes 1 à 3 utilisent les règles et résultats fournis par la méthode d'IR pour tenter de classifier le nouveau cas. Si la classification est impossible seulement avec l'IR alors le RBC intervient. En fait, la tâche du RBC est de résoudre les conflits présents dans l'ensemble de règles généré par le processus d'IR. Les étapes 4 à 6 effectuent la recherche des cas les plus similaires au nouveau cas et les étapes 7 à 9 en déterminent la classification finale. Les auteurs nomment les étapes 7 à 9 : « étapes d'adaptation du nouveau cas ». La Figure 1.5 présente le schéma de l'intégration du RBC et de l'IR dans le système ELEM2-CBR.

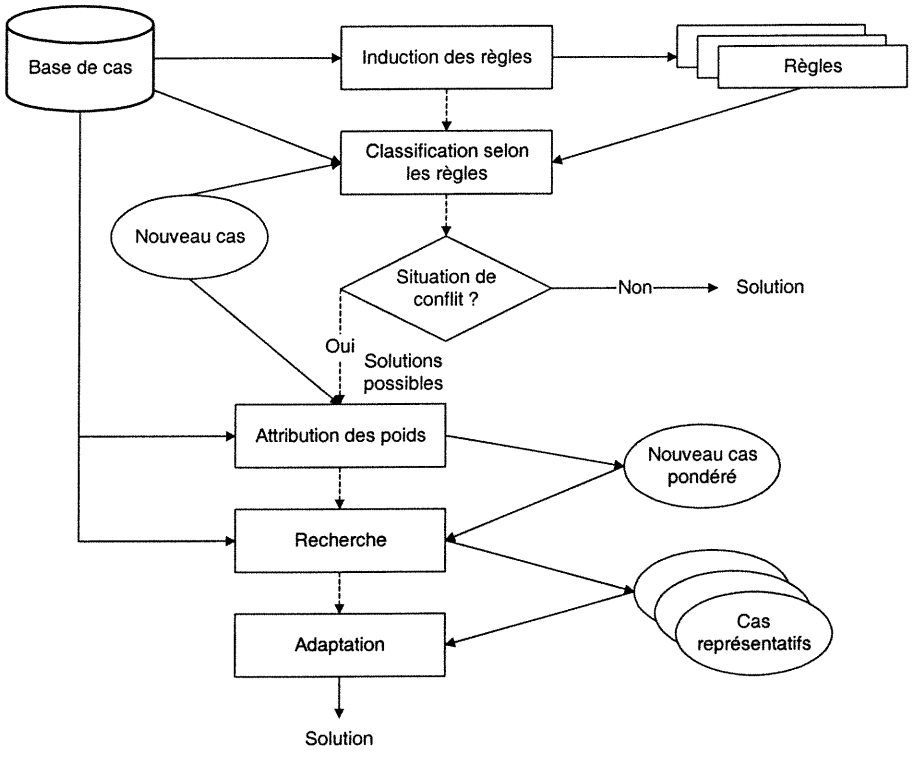


Figure 1.5 : Classification à l'aide du système ELEM2-CBR

Ce système hybride a été évalué selon six bases de données différentes et a été comparé avec des approches pures de raisonnement à base de cas et d'induction de règles. L'approche des auteurs est intéressante car le système ELEM2-CBR donne dans certains cas de meilleurs taux de réussite de classification que les approches pures!

1.3 Conclusion

Les approches combinant des techniques de data mining au processus de raisonnement à base de cas semblent prometteuses. Les deux systèmes dont nous venons de discuter ouvrent des portes aux chercheurs dans ce domaine afin d'améliorer la capacité des systèmes de classification.

Notre projet est appliqué, comme le système CALIBRE, au domaine du marketing. Tout comme ses créateurs, notre but est la simplification du processus de

génération des listes de clients potentiels afin d'améliorer l'efficacité des campagnes publicitaires. Par contre, CALIBRE est constitué de modules préfabriqués (par SAS et ART*Entreprise) ce qui n'est pas notre cas. C'est pourquoi notre approche pour atteindre cet objectif est différente de celle du système CALIBRE.

Notre prototype est d'avantage basé sur le fonctionnement du système ELEM2-CBR. Comme nous le verrons dans le chapitre quatre présentant notre prototype, une approche semblable à celle des auteurs de ELEM2-CBR a été développée.

Chapitre 2 : Data Mining

Where is the knowledge we have lost in information ?

- T.S. ELIOT, Choruses from the Rock

2.1 Introduction au data mining

Le *Data Mining* (DM), une branche du domaine de l'*Intelligence Artificielle* (IA) [LUG et STU 97], a été développé vers la fin des années 1980. Les premiers ateliers sur ce sujet ont eu lieu en 1989; mais c'est en 1995 lors de la première conférence internationale nommée KDD-95 (First International Conference on Knowledge Discovery and Data Mining) que le data mining a vraiment pris son envol [FAY 96]. L'apparition du DM découle en fait de l'évolution des besoins dans le domaine des bases de données. Le concept de *Base de Données* (BD) est apparu autour des années 60 où les premiers balbutiements du stockage de données dans de simples fichiers ont débuté. Par la suite, la recherche et le développement dans ce domaine ont rapidement évolué vers des techniques plus complexes : bases de données relationnelles, hachage, langage SQL et autres, interfaces et rapports évolués, entrepôts de données, etc. C'est l'apparition dans les années 80 de très larges bases de données qui donnera naissance à ce qu'on appellera le data mining. En effet, ce sont entre autres l'avènement de l'Internet et l'évolution des

bases de données dont l'utilisation est de plus en plus simple et peu coûteuse qui ont entraîné l'accumulation d'une très grande quantité de données dans les compagnies et les laboratoires de recherche. Ainsi, ces entreprises se retrouvent avec une abondance de données très riches en informations mais tout à fait indéchiffrables et donc souvent inutilisées. Le data mining attaque directement ce problème et est devenu un outil indispensable permettant d'analyser ces montagnes de données pour en retirer des structures intelligibles.

Comme nous le mentionnions ci-dessus, le DM est un domaine plutôt récent. C'est pour cette raison que dans la littérature nous retrouvons différents noms donnés au data mining comme *knowledge mining from databases*, *knowledge extraction*, *data/pattern analysis* ou encore *Knowledge Discovery in Databases*[3] (KDD) [HAN et KAM 01]. Les deux termes les plus utilisés restent Data Mining et KDD. Tout au long de ce travail nous utiliserons le terme data mining qui semble le plus largement répandu.

Depuis plusieurs années, le domaine du data mining gagne en popularité et évolue très rapidement vu les besoins pressants des entreprises et l'engouement des chercheurs envers ces techniques. Aujourd'hui, le data mining est utilisé dans des domaines aussi contrastants que ceux de la finance, de l'astronomie et du sport professionnel [HED 96]. Le DM est aussi en train de faire une percée intéressante dans deux domaines qui sont actuellement sous les feux de la rampe : la biologie avec le déchiffrage de l'ADN [KAS 99, SAL 99] et la recherche d'informations sur l'Internet [BOL *et al.* 00]. Plusieurs compagnies de logiciels informatiques ont d'ailleurs lancé des outils de data mining sous forme de *suites logicielles*[4] en voyant la complexité et l'engouement de plusieurs milieux pour le DM. Des suites telles que Clementine de SPSS, Darwin de Oracle, Intelligent Miner d'IBM et Entreprise Miner de SAS Institute mettent à la disposition de leurs utilisateurs plusieurs techniques de data mining ainsi que des engins de transformation et de visualisation des données. Un excellent aperçu de

[3] Ou *découverte de connaissances dans les bases de données.* (OLF)
[4] Une suite logicielle est un « Ensemble de logiciels de même marque et aux fonctions théoriquement complémentaires vendus dans un même emballage. » (OLF)

chacune de ces suites est donné dans [BER *et al.* 99].

Dans ce chapitre, nous mettons en place toutes les bases nécessaires à la compréhension du processus de data mining. La section 2.2 expose les principes de base et les différentes définitions reliées au data mining. Par la suite, dans la section 2.3, nous discuterons du processus de DM. La technique de DM des arbres de décision est présentée dans la section 2.4 qui se subdivise selon trois volets : la construction d'un arbre de décision, sa simplification et une présentation des techniques les plus connues concernant ce type d'algorithme.

2.2 Principes de base et définitions

Le data mining étant un domaine encore jeune, la communauté des chercheurs est encore divisée en ce qui a trait à la définition exacte de ce qu'est le DM. Voici un aperçu de ce que l'on peut retrouver dans la littérature.

Le *data mining* ...

- ... est un processus consistant à extraire des informations inconnues, valides et utiles de grandes bases de données et ensuite d'utiliser ces informations pour prendre d'importantes décisions au sein d'une entreprise [CAB *et al.* 97].

- ... c'est le processus de découverte de nouvelles corrélations, de nouveaux « patterns » et de nouvelles tendances significatives et ce en fouillant dans de grandes quantités de données stockées dans des BD [BER *et al.* 99].

- ... réfère à l'extraction ou la « fouille » de connaissances provenant de grandes quantités de données [HAN et KAM 01].

- … c'est le processus non trivial qui consiste à identifier des « patterns » valides, nouveaux et potentiellement utiles dans les données [FAY 96].

- … c'est le processus d'identification de « patterns » cachés et de relations dans les données dans le but de prendre des décisions plus profitables pour l'entreprise [GRO 99].

Nous pouvons observer que les définitions ci-dessus ne mettent pas en valeur les mêmes caractéristiques du DM. Par exemple, la première définition insiste sur le fait que le DM est effectué dans le but d'aider une entreprise à prendre des décisions et d'améliorer la rentabilité de celle-ci. D'autres, en revanche, tiennent à spécifier que les informations ressortant du processus doivent être nouvelles et non triviales. Voici la définition sur laquelle notre travail est basé :

> *Le DM est un processus d'analyse de larges BD dont le but est de faire ressortir de nouvelles informations concrètes sur les données afin de mieux comprendre certaines situations et de prendre de meilleures décisions.*

Avant d'entrer de plain-pied dans la description du processus de DM, intéressons-nous à la problématique du choix d'une technique de DM. Le choix d'une technique dépend de l'objectif visé par la fouille de larges bases de données et peut être différent d'un projet à un autre. Par exemple, une entreprise peut vouloir classer ses nouveaux clients dans des groupes déjà identifiés dans le but de prendre une décision d'après leurs groupes respectifs (But : classification). Cette même entreprise peut par la suite décider d'identifier quels produits se vendent bien ensemble dans le but d'augmenter le total de ses ventes (But : vente croisée).

Le choix d'une technique de data mining n'est pas simple et ne doit pas être pris à la légère, car ce choix influencera directement la qualité des résultats obtenus. Il dépend entre autres du type de données à explorer, du type d'apprentissage désiré, du domaine d'application et surtout, comme nous venons de l'expliquer, de l'objectif à

atteindre. Plusieurs questions se posent donc avant de choisir une technique de DM, les deux suivantes sont les plus importantes :

- *Quel est le but visé par le processus de DM ?*

Le choix de la technique de DM à utiliser dans un système est principalement déterminé par l'objectif visé. À ce stade, il faut identifier le type d'informations recherchées dans les données et identifier les techniques qui peuvent effectuer cette tâche. Le Tableau 2.1 présente un aperçu des différentes tâches pouvant être accomplies à l'aide de techniques de DM. Un exemple basé sur des données d'une librairie fictive est proposé pour chaque tâche.

- *Quelle technique est la plus appropriée à mes données et à mon domaine d'application ?*

Le type de données disponibles nous permet d'éliminer, dès le départ, certaines techniques qui ne les traitent tout simplement pas. Par exemple, il serait peut-être préférable d'utiliser une autre technique que celle des réseaux de neurones si nos données sont majoritairement qualitatives, car les réseaux de neurones ne traitent pas directement ce type de données. Le domaine d'application de son côté peut nous donner des indices concernant les algorithmes les plus appropriés à notre situation en observant ce qui a été utilisé dans ce domaine auparavant. Par exemple, si une compagnie de téléphonie désire fouiller ses données, elle pourrait s'orienter vers la technique des arbres de décision, car elle a déjà été utilisée efficacement dans ce domaine.

Tableau 2.1 : Tâches accomplies par différentes techniques de DM

Tâche	Description	Principales techniques
Description de catégories	Donner une description d'un groupe prédéfini de données. **Ex. :** *Les caractéristiques des clients achetant plus de 500$ de livres par année sont définies comme suit : âges compris entre 20 et 30 ans, célibataires, étudiants...*	Statistiques et Arbres de décision
Classification	Trouver les caractéristiques différenciant des groupes prédéfinis de données pour mieux classer de nouvelles données. **Ex. :** *Les catégories bon_acheteur, moyen_acheteur et mauvais_acheteur sont déjà déterminées. Sachant que les mauvais_acheteurs préfèrent les livres de poche contrairement aux deux autres groupes qui eux préfèrent un autre format de livres, la catégorie d'un nouveau client pourra être déterminée d'après cette caractéristique et ainsi un service plus personnalisé pourra être offert.*	Arbres de décision et réseaux de neurones
Segmentation	Générer des groupes dans une base de données. Ces groupes peuvent par la suite servir à la classification. **Ex. :** *On divise les clients par rapport à leur situation géographique, le budget qu'ils allouent à l'achat de livres, etc. Cette segmentation pourrait par exemple être utile pour déterminer l'endroit le plus profitable pour ouvrir une nouvelle succursale.*	Statistiques et réseaux de neurones
Prédiction	Prédis la valeur de certaines variables ou de la catégorie des données. **Ex. :** *Un client est classé comme bon_acheteur mais on ne connaît pas son type de livre préféré. On peut tenter de le prévoir d'après son classement. Ou l'inverse : on connaît son type de livre préféré et on prévoit sa catégorie (semblable à la classification).*	Statistiques (régression)

En se basant sur ces deux questions, certains choix se sont imposés quant aux techniques de DM que nous pouvions utiliser. Le but du module de DM de notre travail est de classer les profils des clients d'une banque, c'est-à-dire d'identifier la catégorie qui leur convient le mieux. Ainsi, nous effectuons une tâche de classification (deuxième tâche du tableau précédent). Nos données, quant à elles, ne nous limitent pas vraiment. Examinons les techniques de DM les plus connues permettant d'atteindre notre objectif de classification :

Les statistiques : Le domaine des statistiques offre un vaste éventail de techniques très robustes dont l'efficacité est prouvée mathématiquement. Les systèmes de DM utilisent souvent les techniques de régression linéaire, de régression non linéaire et de régression logistique pour classifier des données. Le principe de la régression est de trouver une équation représentant le plus fidèlement possible le comportement des données à analyser. Par exemple, pour la régression linéaire, on cherche une équation de la forme $y = \alpha+\beta x$ pour représenter la droite de la Figure 2.1. La droite de cette figure est celle qui représente le mieux les données, c'est-à-dire que l'équation qu'elle représente minimise la somme des distances entre les points et la droite. Le principe est le même pour la régression non linéaire et pour la régression logistique sauf que les équations à trouver sont plus complexes[5].

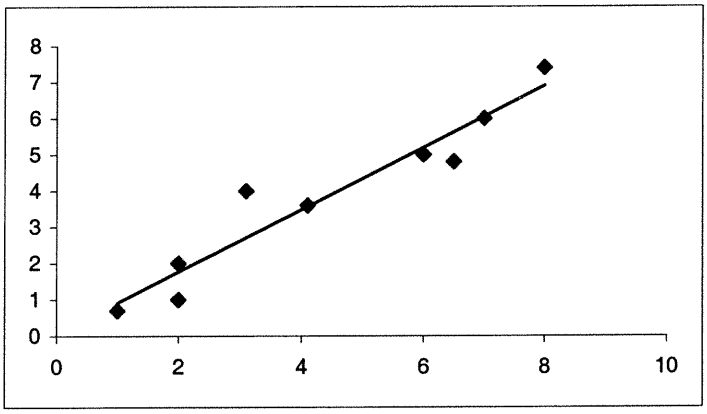


Figure 2.1 : Régression linéaire

[5] Pour plus de détails sur la régression : [DEV95] et [HOG et CRA 95].

L'efficacité de l'approche statistique a été amplement prouvée, elle offre une solide base théorique et elle convient à une foule de domaines. Par contre, son utilisation n'est pas toujours simple et sa terminologie très pointue. Plusieurs notions demandent un certain niveau de connaissance des mathématiques et des théories statistiques.

Les arbres de décision : Les arbres de décision offrent une technique de classification très intuitive et simple. Il existe différentes méthodes pour construire de telles structures, mais le principe de base reste toujours le même. Un arbre possède une tête, des branches et des feuilles comme l'illustre la figure suivante (Figure 2.2). Chaque embranchement ou *nœud* représente un attribut sur lequel des *conditions* doivent être appliquées. Par exemple, le « nœud tête » de la figure dira sur quel attribut les conditions 1 et 2 doivent être vérifiées. Les nœuds terminaux sont appelés feuilles. Une branche est donc composée d'une suite de nœuds et de conditions, c'est ce qu'on appelle une *règle*. Ainsi, les données peuvent être classées avec l'ensemble de règles obtenues dans un arbre. De la même façon, une règle donne les caractéristiques des différentes catégories qui modélisent les données.

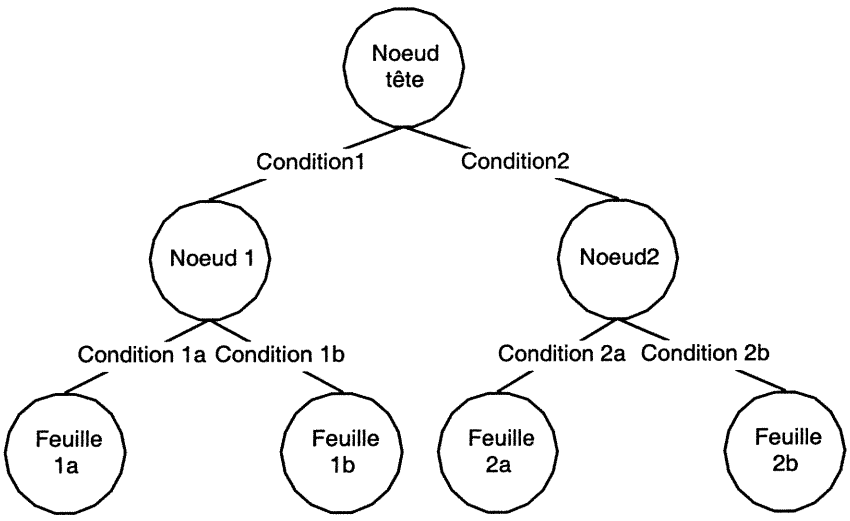


Figure 2.2 : Exemple d'un arbre de décision

Cette technique d'apprentissage est dite supervisée, c'est-à-dire que l'on doit connaître le classement des données *a priori*. Nous verrons plus en détail le développement d'arbres de décision à la section 2.2.

Le plus grand avantage des arbres de décision est sans contredit leur simplicité. Par contre, les arbres de décision peuvent devenir complexes si la quantité de nœuds devient trop grande. En effet, plus on a de données, plus le nombre de branches dans l'arbre risque de devenir très grand et plus il sera ardu de le déchiffrer. C'est pour cette raison que des algorithmes de simplification d'arbres ont été développés. Les arbres de décision donnent généralement de bons résultats et sont utilisés dans pratiquement tous les domaines. Les algorithmes d'arbre de décision les plus connus sont ID3 [QUI 86], C4.5 [QUI 93], CART [BRE *et al.* 84] et CHAID [MAG94].

La classification bayésienne : Cette technique comme son nom l'indique est basée sur le *théorème de Bayes*. C'est donc une technique venant du domaine des mathématiques, utilisant la théorie probabiliste. La classification bayésienne prédit la probabilité que certaines données appartiennent à une certaine catégorie. Une façon simple de classer de nouvelles données avec cette technique est la *classification bayésienne naïve*, c'est-à-dire que les nouvelles données sont classées dans la catégorie ayant la plus grande probabilité d'être correcte.

Le théorème de Bayes calcule la probabilité *a posteriori* d'une certaine hypothèse sachant qu'un ensemble de données précis ait été observé. En classification bayésienne, l'hypothèse H suppose que les données X appartiennent à la catégorie C. La probabilité que H soit vraie est donc calculée en sachant que X a été observé. Cette probabilité a posteriori est représentée par $P(H|X)$. La formule ci-dessous cite le théorème de Bayes :

$$P(H|X) = \frac{P(X|H) * P(H)}{P(X)}$$

où P($X|H$) est la probabilité que X soit observé sachant que l'hypothèse H est vraie;

P(H) est la probabilité que H soit vraie peut importe la valeur de X;

P(X) est la probabilité que X soit observé peut importe l'hypothèse H.

Ainsi, pour trouver la catégorie qui possède le P($H|X$) maximal, le théorème de Bayes doit être appliqué sur chaque catégorie C. Ce qui revient à calculer P($H_{Ci}|X$) pour chaque i = 1, ..., *nombre_de_catégories* où l'hypothèse H_{Ci} suppose que les données X appartiennent à la catégorie C_i.

Donnons un exemple plus concret du calcul de la formule de Bayes. Supposons que notre échantillon de données porte sur le domaine des animaux et que les données sont décrites par le nombre de pattes de l'animal et par son aspect extérieur (poil, plumes...). Soit X qui a quatre pattes et qui est poilu. Disons que H est l'hypothèse supposant que X est dans la catégorie des chats. Alors:

- P($H|X$) est la probabilité que X soit un chat sachant qu'il est poilu et a quatre pattes.
- P($X|H$) est la probabilité que X soit poilu et ait quatre pattes sachant que X est un chat.
- P(H) est la probabilité d'avoir des chats dans notre échantillon.
- P(X) est la probabilité d'avoir des animaux à quatre pattes et poilus dans notre échantillon.

Cet exemple est effectué avec une seule catégorie, celle des chats. Il faut donc effectuer ce processus en changeant la catégorie chaque fois pour celle des chiens, des hirondelles, etc. L'observation X sera ensuite classée dans la catégorie ayant obtenu le plus grand P($H|X$).

La méthode naïve a par contre un gros défaut. Pour l'utiliser, les données doivent absolument être indépendantes, ce qui est rarement le cas dans la réalité. Si nos données ne sont pas indépendantes, une alternative à la méthode naïve appelée « *Bayesian belief networks* » peut être utilisée. Cette méthode étant plus complexe, nous ne la décrirons pas dans le cadre de ce travail. Par contre, Heckerman [HEC 96] en donne une très bonne introduction. Cette technique de classification est très efficace, mais aussi très coûteuse en temps de calcul. Ce défaut en fait donc une méthode moins intéressante à implémenter.

Les réseaux de neurones : La théorie des réseaux de neurones est basée sur le fonctionnement du cerveau humain. C'est par le biais de simples unités appelées *neurones* que ces réseaux arrivent à comprendre et à classer les données qui lui sont fournies. Les réseaux de neurones sont basés sur des notions mathématiques et statistiques ce qui leur donnent une base très robuste. Le principe de base de ces réseaux est de combiner plusieurs neurones afin d'obtenir un réseau qui pourra résoudre des problèmes complexes.

L'unité de base est le *neurone formel* (Figure 2.3). Il a été développé en 1949 par McCulloch et Pitts, deux neurologues américains. Son fonctionnement est relativement simple : le neurone formel reçoit X_i valeurs en entrée, il pondère chacune selon un poids w_i , en fait la sommation et y additionne un certain biais b. Par la suite, une fonction f est appliquée sur ce résultat et si la sortie de la fonction est inférieure ou égale à zéro, la valeur de sortie sera 0 sinon elle sera 1. En fait, le neurone formel est un classificateur linéaire; c'est-à-dire qu'il peut diviser les données en deux catégories. Le problème ici est d'arriver à déterminer les bonnes valeurs à attribuer à chacun des poids w_i. Deux algorithmes d'apprentissage ont été développés pour que les poids s'ajustent avec l'expérience du réseau vers leur valeur optimale : l'algorithme de l'*adaline* et celui du *perceptron*. C'est de cette façon qu'un neurone apprend de ses erreurs et s'améliore avec le temps.

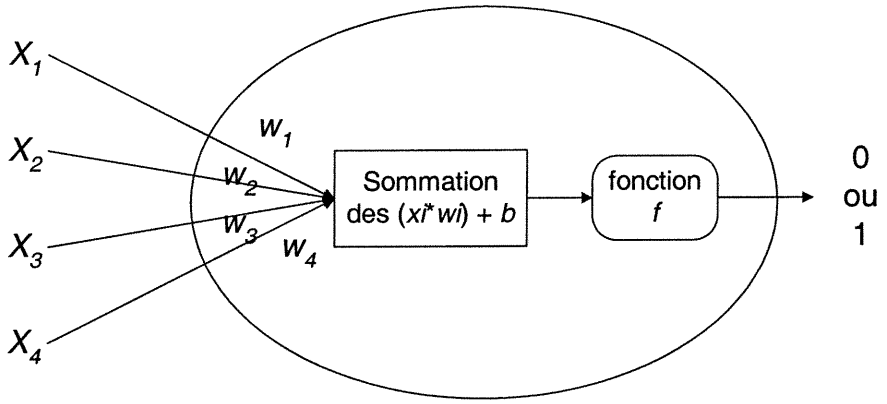


Figure 2.3 : Neurone formel

Or, il n'est pas toujours possible de séparer linéairement les données. Pour contourner ce problème, on combine en couches plusieurs neurones pour obtenir un réseau (Figure 2.4). Ce genre de réseau de neurones permet de diviser les données en plus de deux catégories. On appelle aussi cette structure *perceptron multicouches*. Son fonctionnement est semblable à celui du neurone formel sauf en ce qui a trait à la technique d'apprentissage. Pour régler les poids dans un réseau de neurones, l'algorithme de *rétropropagation* est utilisé. Cet algorithme est basé sur la propriété de différentiabilité de la fonction utilisée pour fixer les poids. Nous ne pousserons pas l'explication de la rétropropagation plus loin dans le cadre de ce travail.

Il est à noter que les réseaux de neurones les plus populaires sont le *multilayer perceptron* et les *self-organizing map de Kohonen*. L'ouvrage de Bishop [BIS 95] offre des explications plus complètes sur les réseaux de neurones et leur utilisation.

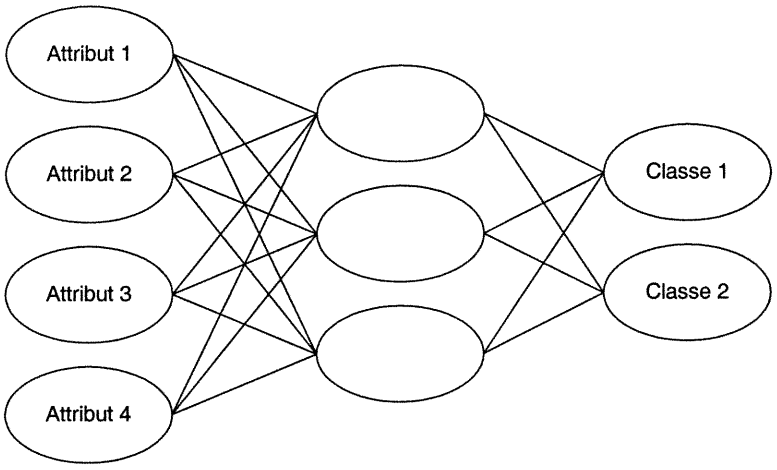


Figure 2.4 : Réseau de neurones complètement connecté avec une couche cachée

Les réseaux de neurones peuvent par contre devenir très complexes et être particulièrement difficiles à calibrer. Ils sont aussi beaucoup moins intuitifs que certaines autres techniques et leurs résultats sont plus difficiles à interpréter. De plus, ils sont préférablement utilisés avec des données numériques [HER *et al.* 91, BIS 95]. Le grand avantage de la technique des réseaux de neurones est qu'elle a fait ses preuves quant à l'efficacité de ses prédictions sur des problèmes complexes. Les réseaux de neurones sont principalement utilisés dans le domaine financier pour la détection de fraude, la prévision des marchés boursiers et la gestion de portefeuille [BEN *et al.* 01, REF 95].

Les algorithmes génétiques : Les algorithmes génétiques, tout comme les réseaux de neurones, tiennent leurs racines du domaine de la biologie. Les algorithmes génétiques sont basés sur la théorie de l'évolution développée par Darwin à la fin des années 1850. Le principe est de faire « évoluer » différentes solutions à un problème afin de dénicher la meilleure solution possible à celui-ci.

Au départ, un ensemble de solutions est généré et forme la *population* initiale. Les solutions de cet ensemble sont appelées *individus* et ont été sélectionnées de façon aléatoire. Chaque solution (ou individu) possède ses

propres caractéristiques nommées *chromosomes*. De nouvelles solutions seront générées à partir de cet ensemble initial et créeront de nouvelles *générations* de solutions qui devraient être plus performantes que les précédentes. Pour créer une nouvelle génération, plusieurs étapes doivent être exécutées :

1. Calculer le *coefficient d'adaptation* de chaque individu X à l'aide d'une fonction d'adaptation $f(X)$. On obtiendra ainsi la probabilité avec laquelle chaque individu peut se reproduire.

2. « Accoupler » les parents selon leur probabilité de reproduction calculée en 1. Plus un individu est « bon » d'un point de vue reproducteur, plus il a de chance de se reproduire. Ainsi, la population devrait s'améliorer à travers la suite des générations. La figure suivante (Figure 2.5) représente la probabilité de reproduction d'une population de quatre individus. On voit que l'individu 4 devrait s'accoupler 1 fois sur 2 tandis que cette probabilité tombe à 1 fois sur 4 pour l'individu 1. Lorsque les deux parents sont sélectionnés, on croise leurs chromosomes pour obtenir un nouvel individu. Finalement, les individus trop faibles ou trop vieux sont éliminés de la population.

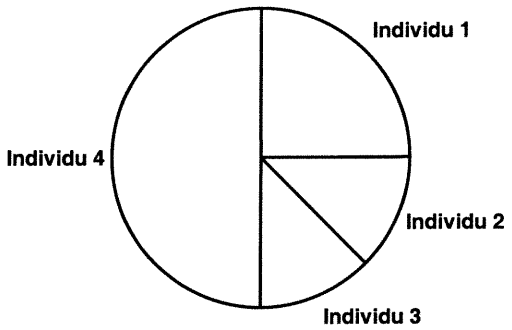


Figure 2.5 : Probabilité de reproduction d'une population de 4 individus

3. Effectuer des *mutations* sur la nouvelle génération d'individus. Une mutation consiste à inverser un chromosome d'un individu. Les mutations sont extrêmement rares, mais néanmoins nécessaires à l'évolution de la population.

4. Si la condition d'arrêt définie par l'usager n'est pas satisfaite, retourner à l'étape 1 avec la nouvelle génération.

Les algorithmes génétiques sont très efficaces pour résoudre des problèmes complexes et sont très simples à implémenter. Par contre, leur fonctionnement est encore plus ou moins bien compris et les résultats peuvent être décevants si l'algorithme est démarré avec une mauvaise population initiale. Le livre de Goldberg, un des pionniers dans le domaine des algorithmes génétiques, présente plus en profondeur ces derniers [GOL 89].

Les différentes techniques de classification que nous venons d'explorer ci-dessus ne forment en fait que la partie centrale du *processus* de data mining. La section suivante présente les différentes étapes qui précèdent et qui suivent l'utilisation d'une technique de DM.

2.3 Processus

Le processus de data mining est constitué de plusieurs étapes toutes très importantes. Ces étapes sont présentées dans la Figure 2.6 et seront expliquées plus en détail par la suite. Évidemment, le processus de data mining se doit d'être précédé par une analyse des besoins et par la détermination des objectifs de l'entreprise, car ce sont ces derniers qui dirigeront les choix effectués tout au long du processus. Cette définition de tâche se fait en collaboration avec les experts du domaine dans lequel l'application du data mining se fera (Ex. : experts financiers, biologistes, médecins, etc.). Comme dans

tout projet, il faut s'assurer de bien comprendre les problèmes, les questions et les objectifs de l'entreprise avant que le développement d'un projet important ne débute.

Une fois encore, certains auteurs ne voient pas tout à fait le processus de DM de la même façon. Le processus présenté ci-dessous est celui le plus fréquemment rencontré. Par contre, dans la littérature, il peut arriver de voir quelques ajouts à ce dernier. Il faut aussi spécifier que le processus de DM n'est pas linéaire, c'est-à-dire que l'on peut retourner à n'importe quelle étape lorsque le besoin s'en fait sentir. En revanche, l'ordre des différentes étapes se doit d'être respecté. Le processus de data mining est constitué de quatre étapes :

1. Sélection et intégration des différentes BD.
2. Nettoyage et transformation des données utiles au data mining.
3. Technique de data mining.
4. Analyse, évaluation et présentation des nouvelles informations découvertes.

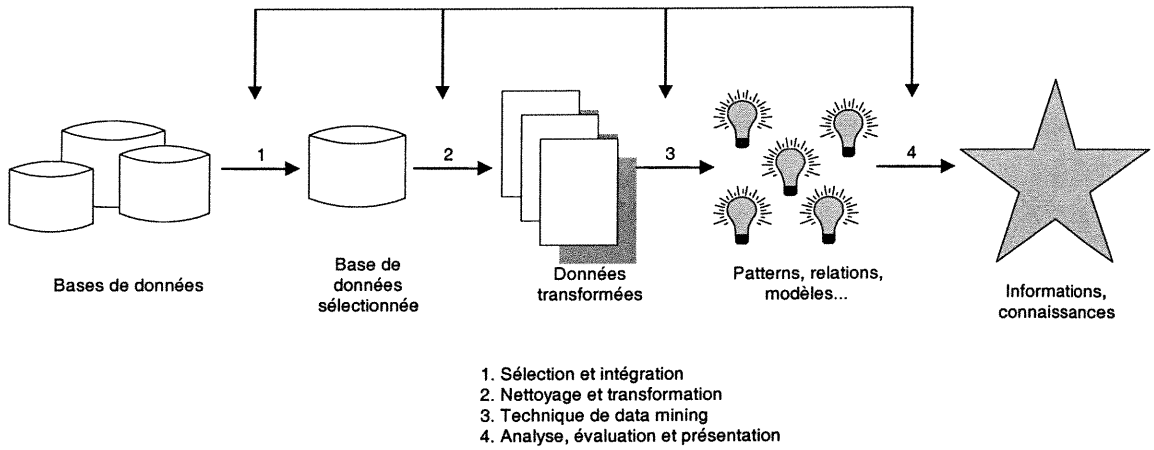


Figure 2.6 : Le processus de data mining

1. Sélection et intégration

Il arrive très fréquemment d'avoir besoin de données se retrouvant dans des bases de données différentes et contenant d'autre part des données inutiles pour arriver à notre but. L'étape de sélection et d'intégration règle les différents problèmes causés par cette situation. À cette étape, les différentes bases de données utiles au data mining doivent être tout d'abord identifiées et intégrées en une seule. Ceci implique une bonne connaissance des données présentes dans les différentes bases et des techniques permettant d'arriver à transformer des bases de types incompatibles afin de les réunir toutes sous le format voulu.

Pour ce qui est de l'étape de sélection, l'objectif est d'éliminer les variables qui sont inutiles à l'atteinte du but du data mining. Par exemple, une banque dont l'objectif est de déterminer si l'attribution d'un prêt personnel à un client est risquée ou pas, éliminera la variable donnant le nombre de téléviseurs que le client possède chez lui, car elle lui est inutile afin de déterminer le comportement financier de ce dernier. On éliminera donc de la BD ce genre de variables. En fait, cette étape sert à simplifier la base de données que l'on veut explorer et permet de bien cibler l'information à explorer.

2. Nettoyage et transformation

L'étape de nettoyage des données permet d'uniformiser les valeurs de certaines variables. Cette étape est surtout nécessaire lorsque la base de données contient des champs qui doivent être remplis à la main par plusieurs personnes. En effet, il peut arriver que deux personnes différentes n'entrent pas la même valeur dans un champ bien qu'elles veuillent dire la même chose. Par exemple, nous savons que les valeurs « Pepsi » et « Pepsi Cola » réfèrent à la même boisson. Il faudrait par exemple uniformiser toutes ces valeurs à « Pepsi Cola » pour éliminer l'ambiguïté due aux différents termes. L'élimination des données inconsistantes ainsi que celle des données redondantes sont deux tâches qui font aussi partie de l'étape de nettoyage. Ces tâches doivent être effectuées car ces données pourraient apporter des variations importantes et

indésirables dans nos calculs. Par exemple, il peut arriver que l'on retrouve la valeur 170 dans la variable âge d'un certain client. Cette valeur est évidemment fausse, il faut donc l'éliminer car l'effet de cette dernière pourrait créer un important biais et être néfaste par la suite. Il faut aussi faire attention aux attributs comportant plusieurs valeurs manquantes (champs vides). Si un attribut a plus de 10% de valeurs manquantes, il faudra penser à la retirer ou à combler les valeurs manquantes selon certaines techniques statistiques : moyennes, fréquences, etc. L'étape de nettoyage consiste donc à essayer autant que possible d'enlever le bruit dans les données et d'améliorer leur qualité.

L'étape de transformation nous permet de son côté de jouer avec nos données. Nous pouvons par exemple créer de nouvelles variables en combinant des variables existantes (ratio entre deux variables, somme de trois autres, etc.). Par exemple, il pourrait être avantageux de créer une variable qui nous donne le nombre total de prêts d'un client plutôt que d'utiliser une série de variables indiquant le nombre de prêts personnels du client, le nombre de prêts hypothécaires du client, etc. L'important est de bien cibler notre but et de toujours faire les transformations en fonction de ce but. D'ailleurs, ce genre de décisions s'effectue généralement à l'aide d'analyses statistiques des données et avec l'avis des experts du domaine d'application. C'est aussi à l'étape de transformation que les données sont transformées afin d'être utilisables par la technique de data mining choisie. Par exemple, nous pourrions transformer des données de type textuel en nombres ou encore effectuer une normalisation de nos données entre [0,1] pour les fournir en entrée à un réseau de neurones. Ainsi, à la fin de cette étape toutes les données nécessaires afin d'atteindre un objectif de DM précis seront bien définies.

3. Technique de data mining

Cette étape représente le cœur du processus de DM. Contrairement aux étapes précédentes, une fois la technique de DM choisie et implantée, cette partie est en principe totalement automatisée. Le résultat obtenu à la sortie de cette étape dépend de la technique de DM choisie. Différents types de résultats peuvent être obtenus : ensembles de règles, relations, segments... En fonction du but visé, il se peut que plusieurs

techniques de DM différentes soient utilisées conjointement, mais de façon séquentielle.

Parmi les techniques les plus populaires, nous retrouvons les arbres de décision et les réseaux de neurones. Ces deux techniques ainsi que plusieurs autres ont d'ailleurs déjà été présentées à la fin de la section 2.2.

4. Analyse, évaluation et présentation

Ici, les résultats obtenus en sortie de l'étape 3 sont interprétés, et ce, en gardant toujours le but visé par le processus bien en tête. L'analyse des résultats sert à identifier les informations les plus utiles et les plus significatives retrouvées dans les données. Les résultats sont évalués en mesurant l'utilité et la robustesse de chacune des informations retrouvées et ce ne sont que les informations pertinentes qui sont conservées. La présentation des résultats sous une forme compréhensible et conviviale représente la dernière étape du processus de DM. La majorité du temps cette étape implique des techniques de visualisation des données.

Il est important de noter que les étapes demandant le plus d'efforts sont les deux premières, c'est-à-dire celles qui préparent les données pour la technique de data mining. En effet, ces étapes de prétraitement peuvent demander jusqu'à 50% de l'effort total, car elles ne sont pas complètement automatisées. De plus, il est à noter que la qualité des résultats est due en partie à la technique de data mining utilisée mais également à la qualité des premières étapes du processus. Ainsi, les étapes de préparation des données sont cruciales à la réussite du processus et une attention particulière devrait leur être accordée.

La technique de DM des arbres de décision est la technique de DM que nous avons décidé de privilégier pour l'implémentation de notre prototype. Notre choix s'est arrêté sur cette technique pour plusieurs raisons. La principale étant que nous voulons utiliser la technique la plus simple et la plus intuitive à comprendre. Ce choix est déterminé par notre objectif de départ qui est de trouver une alternative *simple* aux

méthodes statistiques. Ainsi, nous avons automatiquement éliminé les techniques statistiques, la technique de classification bayésienne basée sur la théorie probabiliste et les réseaux de neurones qui sont aussi complexes pour des gens n'ayant pas ou peu de formation mathématique. Nous avons opté pour les arbres de décision plutôt que pour les algorithmes génétiques pour une raison bien simple : contrairement aux algorithmes génétiques, l'efficacité des arbres de décision a été amplement prouvée et leur fonctionnement est bien compris.

La section suivante aborde le principe général de construction et de simplification des arbres de décision. Les algorithmes les plus connus seront présentés dans la section 2.2.3. L'algorithme que nous utilisons dans notre prototype, appelé algorithme C4.5, est présenté en détail à la toute fin de cette section.

2.4 Arbres de décision

Les structures d'arbres existent depuis longtemps en informatique, mais leur utilisation dans le domaine du data mining est plutôt récente. L'unité de base nécessaire à la construction d'arbres de décision est l'exemple (dans ce travail, le profil d'un client constitue un exemple). C'est pour cette raison que l'on dit que les arbres de décision construisent des modèles de classification de façon inductive, c'est-à-dire que le modèle permettant d'effectuer une classification est construit en généralisant à partir d'exemples spécifiques déjà classés. Comme nous allons le voir, plusieurs algorithmes sont basés sur cette technique dont CART [BRE et al. 84], CHAID [HAR 75] et C4.5 [QUI 93].

Un arbre de décision est une façon de structurer des données, comme son nom l'indique, sous forme d'arbre avec des branches et des feuilles. Cette technique est basée sur le principe de « *diviser pour régner* » (*divide-and-conquer*), c'est-à-dire que nos données sont divisées dans des groupes de plus en plus petits et plus faciles à gérer. Mais, avant d'aller plus loin, présentons la terminologie dont nous nous servirons tout au long de cette section. Une base de données contient des données disposées en lignes et

en colonnes. Le Tableau 2.2 représente une base de données concernant des clients de la librairie fictive que nous avons introduite dans la section 2.1.1. Cette BD utilisée pour construire un arbre contient cinq lignes et quatre colonnes.

Tableau 2.2 : BD d'une librairie fictive

Âge	Budget_par_année	Format_préféré	Type
30	610	Éd. Spéciale	Bon_acheteur
18	250	Format de poche	Mauvais_acheteur
28	535	Éd. Spéciale	Bon_acheteur
40	276	Couverture rigide	Moyen_acheteur
37	578	Couverture rigide	Bon_acheteur

Chaque ligne représente un exemple de client ou une *observation* et chaque colonne un *attribut* caractérisant les observations. Le *domaine* d'un attribut est formé par toutes ses *valeurs* différentes et est donc borné supérieurement. Par exemple, le *domaine* de l'attribut *Format_préféré* contient trois valeurs : {Éd. Spéciale, Format de poche, Couverture rigide} et la *valeur* de l'attribut *Âge* pour la deuxième observation est 18. Parmi tous les attributs, un seul est dit *dépendant*. Cet attribut est appelé *catégorie* (*C*). Dans la base de données de la librairie, la catégorie est représentée par l'attribut *Type,* car la valeur que prendra l'attribut *Type* pour chaque observation dépend directement de la valeur de tous les autres attributs. La variable C_i représente les différentes valeurs contenues dans le domaine de la catégorie *C*.

En résumé, nous avons un ensemble *S* contenant *n* observations et *m* attributs. Un attribut parmi les *m* représente la catégorie *C*. On dit que le domaine d'un attribut contient *i* valeurs distinctes. C'est à partir de ce genre de bases de données que les arbres de décision peuvent être construits.

Comme nous l'avons mentionné, un arbre est formé de *branches* et de *feuilles*. Un arbre de décision est construit du haut vers le bas, c'est-à-dire de la *tête* vers les *feuilles* et est composé de séries de *nœuds* et d'*arcs* qui forment les branches. Un nœud contient un attribut selon lequel les observations seront divisées. Chaque nœud possède

deux arcs ou plus. Ces arcs représentent différentes conditions à satisfaire pour pouvoir les traverser et atteindre le nœud suivant. On appelle *nœuds internes* les nœuds qui ont des arcs menant à leurs enfants et *nœuds terminaux* les nœuds qui n'ont pas d'enfants (feuilles). Chaque branche est constituée d'une série de conditions sur différents attributs qu'il faut satisfaire pour atteindre une feuille. De leur côté, les feuilles donnent une classification, c'est-à-dire qu'elles contiennent une valeur du domaine de la catégorie. La Figure 2.7 présente un exemple d'arbre de décision. Dans cette figure, les rectangles sont des nœuds internes et contiennent l'attribut sur lequel la BD sera divisée selon les différentes conditions des arcs. Les cercles sont les feuilles et donnent la catégorie prédite pour la classification d'une observation.

Par exemple, l'attribut *Budget_par_année* divise les observations selon qu'ils allouent moins ou plus de 535 $ à l'achat de livres dans une année. Si nous suivons la branche de gauche, nous arrivons directement dans une feuille qui nous dicte de fixer la catégorie de notre observation à la valeur *Bon_acheteur*. Par contre, si nous continuons vers la droite, il faudra vérifier une autre condition avant d'arriver à un nœud terminal.

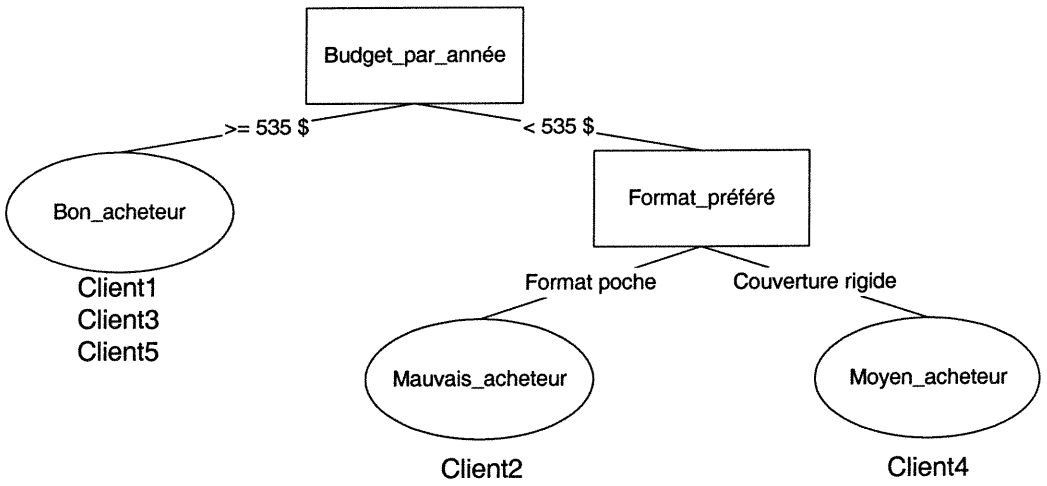


Figure 2.7 : Exemple d'un arbre de décision

De plus, les arbres de décision permettent de transformer chacune de leurs branches sous la forme de *règles de production*. Un arbre peut donc s'écrire comme un

ensemble de règles de production. Si un arbre de décision est le moindrement grand, les règles de production permettent de mieux visualiser les résultats. Une règle est formée de deux parties. La partie de gauche représente une série de conditions (partie *si*) et la partie de droite donne le résultat de la classification si toutes les conditions sont satisfaites (partie *alors*). La règle de production suivante représente la branche complètement à droite dans l'arbre de décision construit avec les données de la librairie.

Si *Budget_par_année* < *535*
et *Format_préféré* = *Couverture rigide* **alors** *Type* = *Moyen_acheteur*

Mais comment décider quel attribut choisir pour chaque nœud ? Comment déterminer les conditions des différents arcs qui nous font avancer d'un nœud à l'autre ? Quand faut-il cesser d'ajouter des nœuds à notre arbre ? C'est ce que nous allons maintenant examiner.

2.4.1 Construction

Le processus à suivre pour construire un arbre de décision est le même peu importe l'algorithme choisi. Il faut tout d'abord déterminer parmi nos attributs lequel représentera la catégorie (attribut dépendant), car les observations seront classées par rapport à cet attribut. Dans le cas de la BD du Tableau 2.2, la catégorie est définie par l'attribut *Type*.

L'algorithme servant à construire un arbre de décision est récursif. Il commence par déterminer l'attribut qui sera à la tête de l'arbre et pour chaque enfant de la tête le même processus est repris. Pour déterminer l'attribut à utiliser dans un certain nœud interne d'un arbre, l'algorithme utilise une *fonction de ramification* (*splitting rule*) qui détermine selon quel attribut on obtiendra la meilleure la division. Les algorithmes de construction d'arbres sont *voraces* (*greedy algorithm*), c'est-à-dire qu'ils essaieront toutes les divisions possibles pour un nœud et choisiront ensuite la meilleure. Plus un attribut permet d'isoler rapidement et efficacement les différentes valeurs du domaine de

la catégorie, plus la fonction de ramification la favorisera. Par exemple, c'est l'attribut *Budget_par_année* qui offre la meilleure division, car elle permet d'isoler complètement la catégorie *Bon_acheteur* d'un seul coup (vers la gauche). Lorsque l'algorithme récursif va vers la droite, il ne tient maintenant compte que des observations ayant une valeur plus petite que 535 pour l'attribut *Budget_par_année.*

Le nombre d'enfants d'un nœud dont l'attribut vient d'être fixé dépend de l'algorithme utilisé. Certains arbres sont de type binaire, c'est-à-dire que chaque nœud a deux enfants sauf les nœuds terminaux qui eux n'en ont aucun. D'autres permettent d'avoir plus de deux enfants rattachés à un certain parent. Les arcs entre deux nœuds sont étiquetés par une condition sur la valeur de l'attribut dans le nœud parent. Lorsque les valeurs dans le domaine d'un attribut sont qualitatives, on crée habituellement un arc pour chacune des valeurs. Si les valeurs sont quantitatives, la majorité des algorithmes vont déterminer la valeur selon laquelle la division est la meilleure et cette dernière sera utilisée pour construire les conditions sur les arcs. Par exemple, c'est la valeur 535 qui offrait la meilleure division pour l'attribut *Budget_par_année* donc un arc contient la condition >= *535* et l'autre < *535*.

Tous les algorithmes offrent différentes *conditions d'arrêt* qui permettent de savoir quand arrêter d'agrandir un arbre. La plupart des algorithmes fixent un nombre minimal de cas devant représenter une feuille. Dans l'arbre de la Figure 2.7, il n'y avait pas de minimum. Dans ce cas, la condition d'arrêt est la suivante : si toutes les observations dans un nœud ont la même valeur pour la catégorie, ce nœud est terminal. Habituellement, une feuille prend la valeur de catégorie des observations qu'elle contient. Si les observations n'ont pas toutes la même valeur de catégorie alors la plus fréquente sera choisie.

Le pseudo-code ci-dessous présente l'algorithme de construction d'arbre de décision.

```
Construction(base)
   Si (condition d'arrêt est satisfaite)
      - Créer une feuille
      - Attribuer à celle-ci la valeur de catégorie majoritaire
   Sinon
      - Calculer la fonction de ramification (FR) pour chaque attribut
      - Créer un nœud avec l'attribut maximisant la FR
      - Déterminer les conditions des arcs
      - Créer les enfants
      - Reprendre le processus pour les n enfants en divisant la base
        de données : Construction (baseDivisée_n)
```

2.4.2 Simplification

Dans notre exemple de librairie fictive, la base de données étant de petite taille nous avons obtenu un arbre plutôt compact, facilement compréhensible et utilisable tel quel. Par contre, lorsqu'un arbre de décision est construit avec des bases de données plus consistantes, l'arbre peut rapidement devenir beaucoup trop imposant, complexe et incompréhensible.

Pour simplifier les arbres de décision obtenus avec l'algorithme présenté à la section 2.4.1, il existe différentes techniques de *simplification* (*pruning*) qui sont habituellement appliquées directement sur l'arbre. Ces techniques sont basées sur le principe du *rasoir d'Occam.* Ce principe dicte qu'il faut toujours tenter de choisir la solution la plus simple parmi celles qui satisfont les données. Ainsi, nous cherchons à trouver l'arbre le plus petit et le plus simple, mais qui classe tout de même les données efficacement.

Il existe plusieurs méthodes différentes pour simplifier un arbre de décision.

Certains algorithmes tentent de simplifier l'arbre pendant qu'ils le construisent. Chaque fois qu'un nœud est créé, une fonction calcule s'il doit être terminal ou non. Cette méthode n'est pas utilisée dans la majorité des algorithmes d'arbres de décision parce qu'elle n'est pas totalement fiable [QUI 93]. La méthode la plus populaire construit l'arbre au complet et tente de le simplifier par la suite. Dans ce cas, plusieurs façons de simplifier un arbre sont disponibles, nous en verrons quelques-unes dans la section suivante. Finalement, d'autres méthodes proposent de simplifier l'ensemble des règles de production généré par un arbre plutôt que l'arbre lui-même.

2.4.3 Arbres les plus populaires

Il existe une foule d'arbres de décision différents basés sur les principes que nous avons mentionnés dans la section précédente. En fait, ce qui les différencie c'est la fonction de ramification qu'ils utilisent, les conditions d'arrêt ainsi que la technique de simplification employée. Les trois algorithmes d'arbres de décision les plus connus sont CART, CHAID et C4.5. Regardons de plus près le fonctionnement de chacun.

2.4.3.1 CART

L'algorithme CART (Classification And Regression Trees) a été développé en 1984 par quatre statisticiens très reconnus dans leur domaine : L. Breiman, J. H. Friedman, R. Olsen et C. Stone [BRE *et al.* 84]. CART construit des arbres de décision binaires, ainsi tous les nœuds internes possèdent exactement deux enfants chacun.

Pour construire un arbre, CART analyse toutes les possibilités de division pour chaque attribut. La fonction de ramification utilisée par défaut s'appelle *fonction GINI* (d'autres fonctions sont aussi disponibles). Cette fonction trouve tout d'abord la valeur la plus fréquente du domaine de la catégorie et tentera d'isoler cette catégorie. Dans l'exemple de librairie que nous avons utilisé plus haut, la fonction GINI tentera tout d'abord d'isoler les observations dont le *Type* est *Bon_acheteur*. La valeur

Bon_acheteur sera donc isolée en identifiant quel attribut permet cette division et la fonction GINI formera les conditions des arcs selon cet attribut. Ce processus est effectué récursivement pour chaque nœud en ciblant toujours la catégorie la plus fréquente.

CART utilise parallèlement plusieurs conditions d'arrêt. Il arrête de construire une branche si toutes les observations dans un nœud sont identiques ou encore s'il y a trop peu d'observations dans un nœud. Lorsque CART décide d'arrêter la croissance d'une branche, il détermine la catégorie du nœud terminal simplement en choisissant la valeur de catégorie qui apparaît le plus fréquemment dans les observations du nœud.

Une fois l'arbre complètement construit, c'est-à-dire lorsque toutes les branches satisfont les conditions d'arrêt, CART simplifie l'arbre. Pour ce faire, il génère des sous-arbres possibles de l'arbre complet en simplifiant ou retirant complètement des branches. Une fois tous les sous-arbres obtenus, l'algorithme détermine quel est le meilleur sous-arbre à l'aide de tests déterminant le taux d'erreur de chacun. Le taux d'erreur est le résultat d'une équation calculée en fonction du nombre de mauvaises classifications dans les nœuds terminaux. Le sous-arbre ayant le plus petit taux d'erreur sera sélectionné. Cette méthode de simplification est par contre très gourmande en temps de calcul.

L'algorithme CART et son code source sont maintenant sous la propriété de l'entreprise Salford Systems [URL 7].

2.4.3.2 CHAID

Les bases de l'algorithme CHAID viennent aussi du domaine des statistiques. CHAID est l'acronyme de « Chi Squared Automatic Interaction Detection » et a été développé par J.A. Hartigan [HAR75]. Cet algorithme peut être utilisé pour effectuer des tâches de prédiction et de classification. L'objectif visé par CHAID est de découvrir des relations entre une variable dépendante et plusieurs variables prédictives. CHAID

permet de développer des arbres avec des nœuds possédant deux branches ou plus contrairement à CART qui est exclusivement binaire.

Comme son nom l'indique, CHAID utilise le *test chi-carré* (bien connu des statisticiens) pour construire un arbre de décision. Ce test est utilisé pour déterminer quel attribut utiliser dans un nœud interne, combien de branches le nœud possède et il fixe aussi les conditions des arcs. Premièrement, l'attribut formant le nœud courant doit être déterminé. Pour ce faire, le test chi-carré est effectué pour chaque attribut indépendant de la base de données. Ce test mesure la probabilité qu'il n'y ait pas d'associations entre l'attribut indépendant et l'attribut de catégorie. C'est-à-dire que l'on obtient la probabilité que l'attribut indépendant n'influence pas le choix de la valeur de l'attribut de catégorie. Dans le jargon statistique, on teste la probabilité que *l'hypothèse nulle* (H_0) soit correcte et la probabilité obtenue par le test est appelée *valeur-p*. Par exemple, pour la base de données de la librairie, il faut calculer la *valeur-p* de chaque attribut (*Âge*, *Budget_par_année* et *Format_préféré*) afin de déterminer la probabilité selon laquelle ces attributs n'influencent pas le *Type*. Les fréquences des valeurs pour chaque attribut indépendant sont calculées en fonction de l'attribut de catégorie. Ces fréquences peuvent se présenter, pour plus de simplicité, sous la forme d'un tableau. Le Tableau 2.3 présente celui pour l'attribut *Format_préféré*.

Tableau 2.3 : Tableau des fréquences des valeurs d'un attribut indépendant en fonction de l'attribut de catégorie

Format vs Type	*Bon_acheteur*	*Moyen_acheteur*	*Mauvais_acheteur*	Total
Éd. Spéciale	2	0	0	2
Format de poche	0	0	1	1
Couverture rigide	1	1	0	2
Total	3	1	1	5

Le test chi-carré est le suivant :

$$\text{valeur} - \text{p} = \text{P}(\chi_2 \geq X^2)$$

où

- χ_2 représente la fonction de distribution chi-carré avec $(r\text{-}1)(c\text{-}1)$ degrés de liberté. La variable r représente le nombre de valeurs dans le domaine de l'attribut indépendant considéré et c le nombre valeurs dans le domaine de l'attribut de catégorie.

- X^2 représente la statistique chi-carré :

$$X^2 = \sum \frac{(nb\_observé - nb\_attendu)^2}{nb\_attendu}$$

La sommation est faite pour chacune des r x c cellules du tableau. La variable *nb_observé* représente le nombre d'observations comptées dans l'échantillon; ce nombre est celui illustré dans chaque cellule du tableau. Le calcul du *nb_attendu* est un peu plus long à effectuer. Calculons, par exemple, le nombre de clients que nous devrions nous attendre à avoir pour le *Type Bon_acheteur* et qui préfèrent les *Éd. Spéciales*. Le nombre total de clients qui préfèrent les *Éd. Spéciales* est de 2 sur 5 donc de 40 %. Ainsi en supposant l'hypothèse nulle, il devrait y avoir 40 % de *Bon_acheteur* qui préfèrent les *Éd. Spéciales*, ce qui fait que *nb_attendu* est de 1.2 client.

Ainsi après avoir calculé la valeur-p pour chaque attribut indépendant, l'attribut ayant la plus petite valeur-p est considéré. Si sa valeur-p est suffisamment petite alors un nouveau nœud est créé avec cet attribut. En revanche, si la valeur-p est trop grande, une feuille est créée et la croissance de cette branche est terminée. L'arbre cesse aussi de grandir si tous les attributs indépendants ont déjà été utilisés. CHAID possède donc deux conditions d'arrêt. De plus, CHAID permet de traiter autant les attributs qualitatifs que

quantitatifs. Il faudra par contre catégoriser les valeurs des attributs quantitatifs en faisant un test chi-carré pour chaque division possible afin de calculer leur valeur-p et de transformer les attributs numériques en attributs textuels.

Par contre, CHAID ne possède pas de méthode de simplification. En fait, la simplification de l'arbre de décision est implicite à l'algorithme. Elle est effectuée en vérifiant si la valeur-p est assez petite, c'est-à-dire qu'elle est bien sous le niveau acceptable pour que l'hypothèse soit vraie. Ce niveau est souvent fixé à 0.0005 (0.05%).

L'algorithme CHAID est utilisé par la compagnie de logiciel statistique SPSS dans leur module nommé *Answer Tree* [URL 6].

2.4.3.3 C4.5

L'algorithme C4.5 est un descendant d'un autre algorithme de construction d'arbres de décision nommé ID3 [QUI 83, QUI 86]. Tous deux ont été créés par le chercheur J.R. Quinlan. Les principes et formules du C4.5 et du ID3 sont basés sur la *théorie de l'information* de Shannon crée en 1948 [SHA 48]. Nous verrons plus en détail cette théorie tout au long de cette section. La suite de ce texte portant sur l'algorithme C4.5 repose sur les références suivantes : [QUI 93], [QUI *et al.* 87], [QUI 87a], [QUI 87b], [QUI 86] et [QUI 83].

ID3 fut développé au début des années 1980 et il est essentiel de noter qu'il possède une limitation importante: ID3 ne fonctionne qu'avec des attributs prenant des valeurs qualitatives. Le C4.5 a principalement été développé par Quinlan afin d'éliminer cette limitation de l'algorithme ID3. Présentons donc tout d'abord le fonctionnement de l'algorithme ID3.

Pour construire un arbre de décision, l'algorithme ID3 se base sur la théorie de l'information de Shannon. Cette théorie affirme que plus un attribut nous apporte de l'information, plus il est intéressant de le considérer pour effectuer une division dans nos

données. En d'autres mots, plus un attribut nous donne des renseignements sur la classification des observations, plus il apporte de l'information selon Shannon. Ainsi, l'attribut qui nous apporte le plus d'information sur l'ensemble des données sera choisi pour former le nœud courant. Pour calculer la quantité d'information apportée par un attribut, il faut observer le gain d'information obtenu en connaissant les valeurs de l'attribut *X* par rapport à l'information que l'on avait avant de connaître ces dernières.

Soit *S* l'ensemble des données utilisées pour construire l'arbre, si l'on divise *S* selon les *n* valeurs du domaine de notre attribut de catégorie, nous obtenons un ensemble de catégories disjointes $C_1, C_2, \ldots, C_n$. On dit que la quantité d'information nécessaire pour identifier la valeur de la catégorie d'une observation de *S* est *info(S)*. C'est aussi ce qui est appelé l'*entropie* de *S* :

$$info(S) = -\sum_{i=1}^{n} p_i * \log_2(p_i)$$

Dans cette formule, p_i représente la probabilité de chacune des *n* valeurs du domaine de l'attribut de catégorie. Cette formule calcule donc l'information générale fournie par l'attribut de catégorie seul.

Maintenant, il faut calculer l'information fournie par la connaissance d'un attribut *X* pour trouver le gain d'information que celui-ci apporte. Nous voulons donc diviser l'ensemble *S* selon un attribut *X* autre que celui de catégorie. Avec cette division, nous obtenons un nouvel ensemble de catégories disjointes $T_1, T_2, \ldots, T_n$ où *n* est le nombre de valeurs dans le domaine de l'attribut *X*. La quantité d'information nécessaire pour identifier la catégorie d'un élément de *S* sachant *X* est:

$$info_X(S) = \sum_{i=1}^{n} \frac{|S_i|}{|S|} * info(S_i)$$

C'est l'entropie de l'ensemble S sachant que X a été observé. C'est-à-dire, la moyenne pondérée de toutes les divisions possibles de S selon les n valeurs du domaine de l'attribut X.

Une fois que nous avons en notre possession la quantité d'information générale de l'ensemble S ainsi que la quantité d'information apportée par chaque attribut X, il ne reste plus qu'à trouver l'attribut qui apporte le plus d'information. Ce calcul est simple, c'est la formule du gain :

$$gain(X) = info(S) - info_X(S)$$

Cette formule représente le gain d'information apporté par l'attribut X sur l'ensemble S, c'est-à-dire, la différence entre info(S) représentant l'information nécessaire pour identifier un élément de S et $info_X(S)$, l'information nécessaire pour identifier un élément de S lorsque X est observé. Ainsi, nous obtenons le gain d'information apporté par la connaissance de X. L'attribut donnant le plus grand gain sera utilisé pour former le nœud de l'arbre.

Ces calculs sont repris récursivement pour chacune des branches jusqu'à ce que l'arbre soit complètement construit. L'algorithme s'arrête lorsque toutes les observations tombant dans une feuille sont de la même catégorie ou qu'il ne reste que deux observations dans une feuille.

Illustrons l'algorithme que nous venons de présenter à l'aide d'un exemple simple. Soit la base de données présentée dans le Tableau 2.4 qui donne les conditions météorologiques pour jouer au golf :

Tableau 2.4 : Base de données pour le golf

PRÉVISION	TEMPÉRATURE	HUMIDITÉ	VENTEUX	JOUER
ensoleillé	85	85	non	ne_pas_jouer
ensoleillé	80	90	oui	ne_pas_jouer
nuageux	83	78	non	jouer
pluvieux	70	96	non	jouer
pluvieux	68	80	non	jouer
pluvieux	65	70	oui	ne_pas_jouer
nuageux	64	65	oui	jouer
ensoleillé	72	95	non	ne_pas_jouer
ensoleillé	69	70	non	jouer
pluvieux	75	80	non	jouer
ensoleillé	75	70	oui	jouer
nuageux	72	90	oui	jouer
nuageux	81	75	non	jouer
pluvieux	71	80	oui	ne_pas_jouer

Dans cette base de données, nous avons 4 attributs indépendants, une catégorie (attribut dépendant *JOUER*) et 14 observations. Seuls les attributs *PRÉVISION* et *VENTEUX* sont utilisables par ID3, car ils sont qualitatifs, les attributs continus (quantitatifs) ne pourront être utilisés que dans l'algorithme C4.5.

Premièrement, le calcul de l'entropie de la base de données sur le golf est effectué. Ce calcul s'exécute en fonction des valeurs de l'attribut de catégorie. *JOUER* peut prendre deux valeurs différentes, *jouer* et *ne_pas_jouer*. Neuf observations sur quatorze dans la base prennent la valeur *jouer* et les cinq autres prennent la valeur *ne_pas_jouer*. Nous obtenons donc:

$$info(S) = -\ (\ 9/14 * \log(9/14) + 5/14 * \log(5/14)\ ) = 0.94$$

Deuxièmement, il faut calculer l'information apportée par la connaissance de chaque attribut. Pour l'attribut *PRÉVISION*, nous obtenons

$$info_{PRÉVISION}(S) = 5/14 * info(S_1) + 4/14 * info(S_2) + 5/14 * info(S_3) = 0.694$$

L'attribut *PRÉVISION* contenant trois valeurs dans son domaine, cette formule se divise en trois parties. La première partie représente l'apport de la valeur *PRÉVISION* = *ensoleillé*; 5 observations sur 14 sont de ce type et forment le sous-ensemble S_1. La seconde partie est celle de *PRÉVISION* = *nuageux* (4 sur 14, sous-ensemble S_2) et la dernière celle de *PRÉVISION* = *pluvieux* (5 sur 14, sous-ensemble S_3). Le résultat $info(S_1)$ représente l'entropie du sous-ensemble S_1 qui n'est formé que des observations donc la valeur de l'attribut *PRÉVISION* est égale à *ensoleillé*. Cette entropie est calculée avec les deux probabilités de catégorie suivante p_i = (2/5, 3/5). Le principe est le même pour $info(S_2)$ avec les probabilités (4/4, 0) et $info(S_3)$ avec les probabilités (3/5, 2/5). Le même calcul sera effectué pour l'attribut *VENTEUX*.

Puis finalement, le gain d'information apporté par chaque attribut est calculé. Le résultat pour l'attribut *PRÉVISION* est le suivant :

$$gain(PRÉVISION) = info(S) - info_{PRÉVISION}(S) = 0.94 - 0.694 = 0.246$$

et pour l'attribut *VENTEUX*:

$$gain(VENTEUX) = info(S) - info_{VENTEUX}(S) = 0.94 - 0.892 = 0.048$$

Ainsi, c'est l'attribut *PRÉVISION* qui sera sélectionné pour former le nœud de tête de l'AD car 0.246 est supérieur à 0.048. Le nœud formé par *PRÉVISION* aura 3 branches, une pour chacune de ses valeurs : *ensoleillé*, *nuageux* et *pluvieux*.

L'algorithme C4.5 dont nous allons maintenant discuter, est en fait un élargissement des fonctionnalités de ID3. Il fonctionne exactement de la même façon qu'ID3 pour ce qui est des attributs qualitatifs. Par contre, l'algorithme C4.5 possède la capacité de traiter les attributs quantitatifs. Pour ce faire, les attributs numériques sont transformés en un genre d'attribut qualitatif de façon à pouvoir par la suite utiliser l'algorithme ID3.

La discrétisation d'un attribut numérique se fait de la façon suivante. Soit l'attribut X qui est continu, examinons les valeurs de son domaine. Tout d'abord, il faut classer ses valeurs en ordre croissant, nous avons donc V_1, V_2, V_3, ... , V_m. Pour chaque valeur V_j, $j = 1, \ldots, m$, il faut diviser les données en deux groupes à la seule condition que chaque groupe contienne au moins deux données. Puis, pour chaque division, il faut calculer le gain et finalement choisir la division qui maximise le gain.

Revenons à l'exemple de la base de données sur le golf. L'attribut *HUMIDITÉ* possède les valeurs suivantes en ordre croissant : [65, 70, 70, 70, 75, 78, 80, 80, 80, 85, 90, 90, 95, 96]. Des divisions seront tentées entre 70 et 75, 75 et 78, 78 et 80, 80 et 85, 85 et 90 puis entre 90 et 95. Pour chacune de ces six divisions, le calcul suivant est effectué :

$$gain(\text{HUMIDITÉ}_i) = info(S) - info_{HUMIDITE\ pour\ i}(S) \qquad \text{où } i = 1, \ldots, 6$$

Ainsi, la meilleure division est celle entre 75 et 78. Le nœud formé par l'attribut *HUMIDITÉ* aura donc deux branches : une posant la condition *HUMIDITÉ* <= 75 et l'autre *HUMIDITÉ* > 75. De cette façon, les attributs continus pourront aussi être inclus dans la construction des arbres de décision.

Une modification peut être apportée à l'algorithme C4.5 pour améliorer son efficacité. La notion de gain présentée ci-dessus est biaisée, elle a en effet tendance à favoriser les attributs qui prennent un grand nombre de valeurs. Par exemple, si $info_X(S)$ est calculée sur un attribut X où le nombre de valeurs du domaine est égal au nombre d'observations (c.-à-d. toutes les observations prennent une valeur différente), nous aurons $info_X(S) = 0$. Ainsi, le gain sera toujours maximal, car nous ne soustrairons rien de l'entropie de départ $info(S)$. Pour remédier à ce problème, Quinlan suggère d'utiliser l'équation du $gainRatio(X)$ plutôt que celle du $gain(X)$ présentée plus haut :

$$gainRatio(X) = \frac{gain(X)}{splitInfo(X)}$$

où *splitInfo(X)* :

$$SplitInfo(X) = \sum_{i=1}^{n} \frac{|S_i|}{|S|} * \log_2\left(\frac{|S_i|}{|S|}\right)$$

Ainsi en suivant l'algorithme décrit ci-dessus avec l'équation du *gainRatio* nous obtenons l'arbre de la Figure 2.8 et l'ensemble des cinq règles de productions du Tableau 2.5 pour notre BD sur le golf.

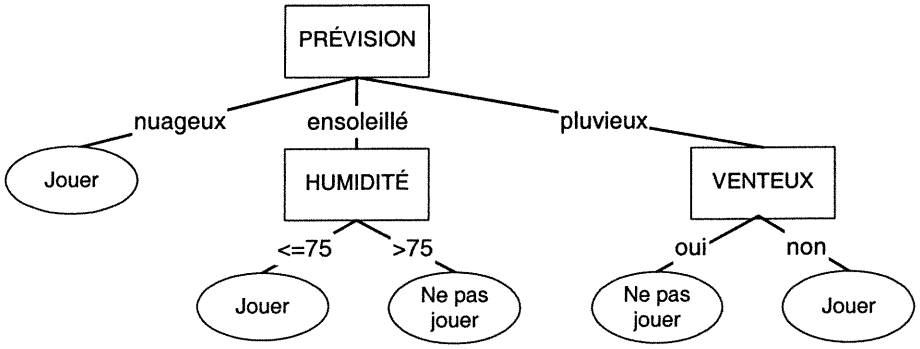


Figure 2.8 : Arbre de décision pour l'exemple sur le golf.

Tableau 2.5 : Règles de production pour la BD sur le golf

Si PRÉVISION = nuageux **alors** *Jouer*
Si PRÉVISION = ensoleillé **et** HUMIDITÉ <= 75 **alors** *Jouer*
Si PRÉVISION = ensoleillé **et** HUMIDITÉ >75 **alors** *Ne pas jouer*
Si PRÉVISION = pluvieux **et** VENTEUX = oui **alors** *Ne pas jouer*
Si PRÉVISION = pluvieux **et** VENTEUX = non **alors** *Jouer*

Quinlan propose deux techniques pour simplifier les arbres de décision obtenus avec l'algorithme C4.5. Une de ces technique fonctionne selon le principe du *MDL* (*Minimum Description Length*) [QUI 93] et travaille directement sur l'arbre tandis que l'autre s'applique sur l'ensemble des règles de production. Cette dernière est très intéressante, elle minimise l'ensemble des règles de production plutôt que d'affecter l'arbre directement. De plus, selon Quinlan [QUI 87b], c'est la technique la plus efficace d'entre toutes malgré qu'elle ait un plus long temps de calcul que celle du MDL. Nous

avons opté pour la simplification des règles de production, car le temps de calcul de cette technique sur notre base de données reste acceptable. Ceci est attribuable à la taille relativement restreinte de notre BD.

La simplification de l'ensemble des règles de production est constituée de deux étapes : la diminution du nombre de conditions dans chaque règle et la diminution du nombre de règles. Dans la première étape, les règles de l'ensemble sont considérées de manière individuelle. L'algorithme tente d'enlever une par une les conditions dans une règle et observe les performances de la règle ainsi réduite. Si la règle offre une aussi bonne performance de classification en ayant une condition en moins, nous éliminerons cette condition et la règle sera simplifiée. En fait, c'est simplement l'application du rasoir d'Occam : si la règle ne fait pas plus d'erreurs de classification en ayant une condition en moins cela signifie que cette condition est inutile au classement et qu'elle peut donc être éliminée de la règle. Cette étape simplifie le résultat donné par l'ensemble de règles de production. L'algorithme C4.5 applique ce principe pour toutes les conditions de toutes les règles.

La deuxième étape est en fait une reprise du principe que nous venons de présenter, mais à l'échelle de l'ensemble des règles. Les règles de production sont retirées une à une de l'ensemble et les observations qui satisfaisaient cette règle sont redistribuées. Si l'ensemble des règles ne fait pas plus d'erreurs qu'avant, la règle peut être retirée. Ainsi, nous nous retrouvons à la toute fin avec un ensemble de règles de production beaucoup plus petit, plus simple et plus facile à interpréter! De plus, dans la majorité des cas, le pourcentage d'erreurs est plus bas grâce à la généralisation des règles. Nous observerons cet effet dans le chapitre quatre qui présente nos résultats.

2.5 Conclusion

Ce chapitre présentait le data mining, un processus d'analyse de bases de données dont l'objectif est de faire ressortir d'une masse d'informations disponibles des

connaissances utiles, valables et non triviales. Nous avons aussi élaboré sur la technique des arbres de décision (AD) qui permet de représenter les données de façon générale sous la forme de règles de production. La technique des AD est celle employée dans notre travail, car son fonctionnement et sa structure sont simples et intuitifs.

Nous allons utiliser la technique des arbres de décision appelée C4.5 afin de classer les profils des clients d'une banque (tâche de classification). Nous avons privilégié cette technique, car son efficacité a été amplement prouvée, mais également parce que cet algorithme n'est pas propriétaire, nous pouvons donc le reproduire sans avoir à nous soucier des droits de propriété.

Chapitre 3 : Raisonnement à base de cas

Reasoning is an art not a science...

- WOS et al., Automated reasoning 1984

3.1 Introduction au raisonnement à base de cas

C'est aux États-Unis au début des années 1980 qu'ont vu le jour les tous premiers systèmes de *Raisonnement à Base de Cas* (RBC) [KOL 93, WAT 97, AAM et PLA 94] . En fait, c'est aux travaux effectués en 1977 par Schank et Abelson, deux chercheurs en sciences cognitives de l'Université de Yale, que l'on doit la naissance des bases du RBC [SCH et ABE 77]. L'idée développée par ces deux chercheurs est la suivante : l'être humain a la capacité de mémoriser les caractéristiques de situations typiques dans lesquelles il se retrouve sous la forme de MOPs (*Memory Organisation Packets*) [SCH 82]. Les MOPs sont des structures de la mémoire conceptuelle décrivant la forme générale de certaines situations comme aller au cinéma ou conduire une voiture. Ces structures nous servent à faire de l'inférence sur le déroulement d'autres situations semblables ou encore à développer des attentes envers certains choix que l'on effectue. Schank introduira au début des années 1980 le premier modèle cognitif pour le raisonnement à base de cas. Par la suite, la communauté des chercheurs en IA adopte

immédiatement le RBC. Le premier système de RBC créé par Janet Kolodner nommé CYRUS verra le jour en 1983 [KOL 83].

Le fonctionnement du RBC est calqué sur le comportement humain. Les raisonnements ou les choix effectués par l'être humain lors de nouvelles expériences sont basés sur ceux qu'il a expérimentés dans des situations similaires vécues précédemment. En d'autres mots, le RBC est basé sur la réutilisation et l'adaptation d'anciennes solutions utilisées dans différentes situations pour en solutionner de nouvelles sans répéter les mêmes erreurs. Par exemple, pour régler la cause sur laquelle il travaille, un avocat tentera tout d'abord de se souvenir d'une cause qu'il a défendue précédemment et qui semble similaire à celle qu'il défend actuellement. Si une telle cause existe, il ajustera son ancienne plaidoirie à la nouvelle situation et réutilisera une expérience vécue pour résoudre un nouveau problème. L'ajustement de l'ancienne expérience dépendra évidemment du succès ou de l'échec de celle-ci. Cet exemple est bien précis, c'est ce qu'on appelle la jurisprudence. Ce processus est effectué dans bien des situations de la vie courante comme préparer un repas ou régler un conflit entre deux enfants!

Du point de vue du domaine de l'informatique, le RBC est une technique d'intelligence artificielle très intuitive basée sur le fonctionnement de la mémoire et du raisonnement humain. Il est intéressant de noter que le RBC est fondamentalement différent de la majorité des techniques d'IA habituelles. En effet, le fonctionnement du RBC repose sur l'apprentissage de cas bien précis tandis que normalement en IA on tente de généraliser le plus possible les comportements ou les situations rencontrés. De plus, le RBC étant une technique très récente (moins de 20 ans), il reste encore plusieurs avenues à explorer et c'est ce que nous tenterons de faire à travers ce travail. Malgré le jeune âge de cette technique, il existe aujourd'hui des systèmes de RBC dans plusieurs domaines. Les applications les plus populaires sont celles touchant aux domaines du design, de la résolution de conflit, de la planification et du support aux usagers. Donnons un exemple d'utilisation du RBC pour chacun de ces domaines :

- *Design* : L'utilisation d'anciens plans de salles à manger peuvent être adaptés et combinés pour créer l'aménagement d'une nouvelle salle à manger dans une nouvelle maison.

- *Résolution de conflit* : La résolution des conflits entre un syndicat et son employeur peut être effectuée par une base de conflits semblable déjà utilisée dans le milieu militaire.

- *Planification* : La réutilisation et l'adaptation d'anciens horaires de travail efficaces dans le but de créer un nouvel horaire pour des infirmières dans un hôpital.

- *Support aux usagers* : L'utilisation d'un système de RBC dans une compagnie de téléphonie pour le service de support après vente. Ce système contient une foule de cas problèmes typiques déjà résolus et les téléphonistes s'y réfèrent pour apporter des réponses aux questions des clients.

Plusieurs logiciels contenant des outils de RBC ont été développés par différentes compagnies dans le but de simplifier la construction d'un système de RBC. Parmi ceux disponibles, nous pouvons retrouver ART*Entreprise de la compagnie californienne Brightware [URL 8], CBR3 produit par Inference Corp. [URL 9] et ReCall de Isoft [URL 10]. Une brève présentation de chacun de ces outils de RBC et de plusieurs autres est exposée dans [WAT 97].

Une partie de notre projet consiste à développer un système de raisonnement à base de cas. Le but du processus de RBC utilisé dans notre système est de classifier les profils de clients de la banque avec laquelle nous travaillons (banqueX) en utilisant d'anciens profils de clients de celle-ci. En fait, le cycle de RBC est seulement utilisé si les règles de production générées par l'algorithme C4.5 n'ont pas réussi à classer le nouveau profil du client. L'architecture du cycle de raisonnement à base de cas se prête

très bien à la classification. Un nouveau cas sera simplement classé dans la même catégorie que le ou les cas les plus similaires à ce dernier.

La suite de ce chapitre est divisée en trois sections. La définition de base de cas et de plusieurs autres notions utiles sont données dans la section 3.2. Cette section contient également une présentation des principes de base reliés au RBC. Par la suite, nous allons examiner le processus ou cycle du RBC (section 3.3) et finalement dans la section 3.4, nous présentons différents algorithmes utilisés en RBC ayant comme objectif la classification de cas.

3.2 Principes de base et définitions

Débutons en donnant une définition générale de ce qu'est le raisonnement à base de cas. Le fonctionnement du RBC est basé sur la résolution de nouveaux problèmes en adaptant d'anciennes solutions ayant résolu dans le passé des problèmes similaires. Le RBC est très intéressant car son approche pour résoudre des problèmes est intuitive et très simple à comprendre.

En réalité, nous nous servons tous constamment du principe de fonctionnement du RBC. Nos expériences passées nous servent de référence pour prendre des décisions dans de nouvelles situations semblables à celles que nous avons déjà vécues. Un exemple que l'on peut facilement retrouver dans la littérature est celui du mode de raisonnement d'un médecin. En effet, les médecins se servent constamment des expériences qu'ils ont vécues avec leurs différents patients dans le cadre de leur travail. Lorsqu'un nouveau patient se présente et présente certains symptômes, le médecin tente de se remémorer un ancien patient qui présentait des symptômes similaires. Il suggérera alors au nouveau patient un traitement semblable.

Par le biais de cet exemple, nous remarquons rapidement un avantage considérable des systèmes de raisonnement à base de cas. Ils évitent de reprendre à

partir de zéro le processus de résolution d'un problème qui a déjà été résolu et ainsi ils sauvent beaucoup de temps. En effet, le médecin n'a pas eu besoin d'explorer toutes les maladies et tous les traitements reliés aux symptômes de son patient en partant des bases de la médecine. Il a plutôt débuté sa recherche avec une solution déjà existante ayant donné de bons résultats sur un patient rencontré dans les semaines précédentes. Le médecin a simplement adapté le traitement au nouveau patient et a ainsi sauvé énormément de temps de recherche de diagnostic et d'évaluation de traitements. Les systèmes de RBC impliquent donc les notions de mémoire, de recherche, d'adaptation et d'apprentissage.

3.2.1 La mémoire

Le cœur du fonctionnement du RBC est basé sur une structure semblable à celle de la mémoire humaine. C'est grâce à sa mémoire que le médecin de notre exemple arrive à se remémorer le patient qu'il a soigné la semaine précédente ainsi que tous les patients qu'il a pu observer au cours de sa carrière. Par contre, la mémoire humaine à ses limites et elle peut quelques fois avoir certaines failles! Heureusement dans le domaine de l'informatique, la mémoire d'un ordinateur n'oublie pas et offre aujourd'hui des capacités impressionnantes. Ainsi, la « mémoire » d'un système de RBC est un espace appelé *base de cas*. C'est dans la base de cas que sont entreposés toutes les expériences, situations, problèmes et solutions qu'un système de RBC a pu rencontrer dans le passé.

Les expériences contenues dans la base de cas sont appelées *cas*. Un cas est habituellement constitué de deux parties : le *problème* et la *solution*. La partie problème est en fait une situation ou un état défini de façon précise par une série d'attributs illustrant le contexte dans lequel un cas est survenu. De son coté, la partie solution décrit les actions ou les choix à effectuer pour solutionner correctement le problème. Le Tableau 3.1 présente un échantillon de cas que l'on pourrait retrouver dans différents systèmes de raisonnement à base de cas.

Tableau 3.1 : Exemples de cas dans différents systèmes de RBC

Problème/Situation	Solution
Médecine : Patiente de 42 ans, fumeuse, la prise de sang est normale et elle souffre de migraines chroniques.	Prescrire le médicament *X*. Prendre une demie dose par jour pendant 7 jours. Deuxième rencontre dans 10 jours.
Planification : Congrès de 150 personnes durant 4 jours. Nécessite 4 salles de kiosques et un auditorium. Besoin d'un service de traiteur.	Le centre des congrès *ABC* offre 6 salles de kiosques et un auditorium de 250 places. Engager le service de traiteur *MNO* juste en face du centre des congrès.
Finances : Client de 30 ans, marié et sans enfant, possédant un prêt hypothécaire de 60 000$, une carte de crédit et un salaire de 70 000$ par année.	Accorder un prêt personnel de moins de 20 000$ avec un taux d'intérêt de 7% sur 10 ans.

Une façon simple de représenter le fonctionnement du raisonnement à base de cas du point de vue des problèmes et de leurs solutions est illustrée à la Figure 3.1. Cette représentation tirée de [LEA 96] est basée sur deux ensembles, celui des problèmes et celui des solutions. Tout d'abord, un nouveau problème est entré dans le système de RBC. La première étape consiste à rechercher le cas avec les attributs les plus similaires au nouveau problème (flèche R dans la figure). De cette façon, une solution présente dans l'ensemble des solutions est proposée. Cette dernière peut être modifiée si nécessaire (flèche A), donnant ainsi une nouvelle solution adaptée au contexte du problème entré au départ dans le système.

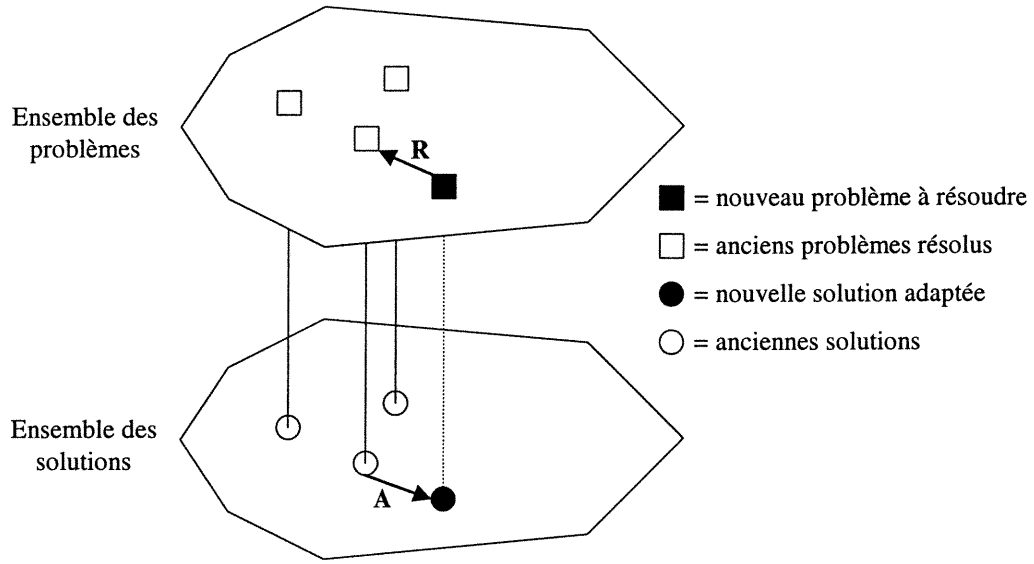


Figure 3.1 : Construction d'une solution avec le RBC

3.2.2 La recherche

Par contre, la mémoire pouvant contenir une très grande quantité de cas, un mécanisme de recherche est nécessaire afin de retrouver le cas le plus similaire au problème rencontré. Le choix du cas le plus similaire est une opération délicate qui peut dépendre d'une foule de facteurs. Pour simplifier cette tâche, les cas sont définis selon un ou plusieurs *index*. Les index permettent de retrouver le bon cas au bon moment. En fait, ils mettent en valeur le contexte dans lequel un cas a bien fonctionné et le moment où il serait avantageux de le faire ressortir. Les index permettent aussi d'accélérer la recherche du cas le plus similaire dans la base de cas. Un index est formé par une combinaison de certains attributs d'un cas et doit satisfaire certaines propriétés. Un bon index doit :

- être prédictif;
- représenter les aspects importants du contexte d'un cas;
- être assez général pour englober différents problèmes similaires;
- être assez précis pour bien représenter le problème pour lequel il a été défini et le faire ressortir au bon moment.

Prenons l'exemple d'un évaluateur financier dont la tâche est de déterminer si un client est un bon candidat pour un prêt personnel. Un bon index pour ce client pourrait tenir compte de son âge, de son salaire ainsi que de son niveau d'endettement. Ces trois attributs semblent bien prédictifs et représentent convenablement un client faisant une demande de prêt. D'autres attributs tels que l'adresse ou la langue parlée par le client ne feront pas partie de l'index, car ils ne sont probablement pas prédictifs du comportement financier de ce dernier. En résumé, un bon index permettra de choisir un cas de façon appropriée et exactement au moment où il est utile à la prise de décision. Il est par contre complexe de trouver un bon index possédant toutes les qualités énumérées ci-dessus, c'est ce qu'on appelle le *problème d'indexation*.

3.2.3 L'adaptation

Une fois le cas le plus similaire retrouvé à l'aide des index, le système doit décider du niveau d'adaptation de la solution retrouvée. Présentons ce processus avec l'exemple du médecin développé un peu plus haut. Supposons qu'un médecin rencontre une patiente *Y* présentant certaines caractéristiques (sexe, âge...) et certains symptômes (maux de tête, allergies...). Le premier réflexe du médecin sera de tenter de se rappeler un des ses patients qui éprouvait les mêmes symptômes. Une fois ce cas retrouvé, il tentera d'adapter sa solution au cas de la patiente *Y*. Par exemple, le médecin se souvient d'un patient *Z* qui l'a consulté le mois précédent et qui éprouvait exactement les mêmes symptômes que la patiente *Y*. Les seules différences entre ces deux cas sont le sexe et le poids des patients. Le traitement donné à la patiente *Y* sera donc adapté de façon à ce que le temps de traitement soit légèrement moins long ou que la concentration du médicament soit moins grande car le poids de la patiente est moindre. Il peut arriver, dans certains cas, que l'adaptation soit inutile. Lorsque cette situation survient, la solution est directement appliquée au nouveau cas.

3.2.4 L'apprentissage

Dans certaines situations, il est possible que l'adaptation effectuée sur un nouveau cas soit complexe ou encore que l'on ne trouve pas de cas suffisamment similaires à ce dernier. Lorsque ces situations se présentent, nous sommes probablement en présence d'un nouveau cas qu'il serait intéressant de mémoriser pour une utilisation ultérieure. L'apprentissage de nouveaux cas se fait principalement par l'intégration de ces cas à la base du système. Par contre, il est moins évident de décider quand un cas est assez intéressant pour l'ajouter à la base de cas. Une deuxième forme d'apprentissage en RBC est représentée par l'ajustement des poids accordés aux différents attributs indexés d'un cas. Les attributs menant à de bons résultats seront récompensés en augmentant leur poids, ce qui leur donne une plus grande influence dans la base de cas. L'apprentissage en RBC est une notion importante afin d'améliorer la performance de ce genre de systèmes.

La majorité des systèmes de raisonnement à base de cas est composée des quatre notions que nous venons de présenter. Ce genre de système peut également effectuer plusieurs tâches différentes. Ces tâches sont généralement divisées en deux catégories : les tâches de *résolution de problèmes* et les tâches d'*interprétation* [LEA 96, KOL et LEA 96].

Résolution de problèmes : L'objectif des systèmes de RBC effectuant de la résolution de problèmes est de trouver une solution à un nouveau problème en se basant sur d'anciennes solutions que le système a en mémoire. Ici, la notion d'adaptation est importante, car une solution correcte et efficace est recherchée. De plus, ces systèmes sont habituellement aptes à fournir des explications justifiant leurs choix. Les domaines de la planification, du design et du diagnostic fonctionnent tous sous ce principe. L'exemple du médecin et de sa patiente, donné précédemment, est représentatif de ce type de tâche.

Interprétation : L'objectif des systèmes d'interprétation est de classer ou de caractériser certaines situations à l'aide de la base d'expériences que le système a en mémoire. Cette classification permet d'interpréter une situation afin de mieux décider ou de justifier les actions à entreprendre. Ce type de tâche est principalement utilisé dans le domaine du droit ainsi que pour effectuer de la classification. Un très bon exemple de cette tâche est celui de la jurisprudence en droit. Un avocat évoquera une cause passée semblable à celle qu'il défend afin de justifier le droit d'obtenir la même sentence que cette dernière pour son client. Dans les systèmes dont la tâche est l'interprétation, la notion de recherche est la plus importante.

Puisque notre travail consiste à classer les clients d'une banque, nous nous situons dans la catégorie des tâches interprétatives.

Nous terminons ainsi le survol des quatre notions formant les bases du processus du raisonnement à base de cas et des différentes tâches qu'elles peuvent accomplir. Chacune de ces notions donnait des exemples concrets des étapes du cycle de RBC. Regardons maintenant d'une façon plus détaillée et plus technique l'utilisation de chacune de ces notions à l'intérieur du processus de RBC.

3.3 Processus

La forme générale du processus de raisonnement à base de cas peut être visualisée sous la forme d'un cycle se répétant chaque fois qu'un nouveau cas se présente. Ce cycle, appelé cycle des quatre « R », est représenté à la Figure 3.2.

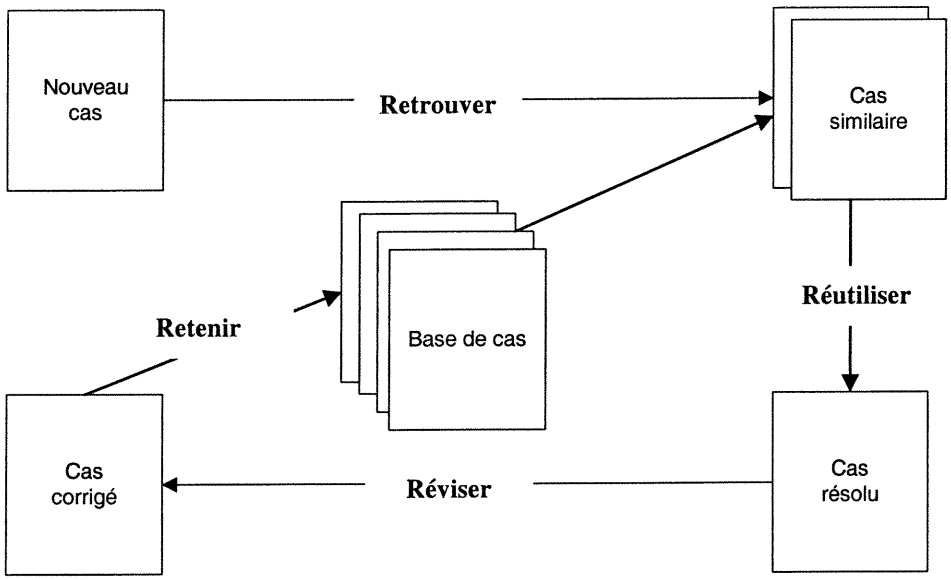


Figure 3.2 : Cycle du raisonnement à base de cas [AAM et PLA 94]

Le cycle de RBC est déclenché par l'arrivée d'un *nouveau cas*. La première étape est de *retrouver* dans la *base de cas* un *cas similaire* au nouveau cas. La solution du cas similaire sera *réutilisée* pour formuler une hypothèse quant à la solution du nouveau cas. Après avoir vérifié et adapté l'hypothèse de départ, une solution est proposée, ce qui donne le *cas résolu*. Le cas résolu est *révisé*, ce qui permet au système de s'assurer de la qualité des solutions proposées et de faire les derniers ajustements à la solution. Cette étape génère un *cas corrigé*. Finalement, le cas corrigé est *retenu* par le système, c'est-à-dire qu'il est intégré à la base si nécessaire. C'est à cette dernière étape que les variables du système peuvent être mises à jour afin d'améliorer l'efficacité et l'autonomie du système de RBC.

Le reste de cette section est consacré à la description de chacune des étapes du cycle des quatre « R » du processus de raisonnement à base de cas.

3.3.1 Retrouver

Comme son nom l'indique, cette étape consiste à effectuer une recherche dans la mémoire (ou base de cas) pour en ressortir un ensemble des cas les plus similaires au nouveau cas que l'on tente de résoudre. Généralement, cette étape est divisée en deux tâches : retrouver les cas plausibles et sélectionner le ou les meilleurs cas.

La première tâche construit un ensemble de « bons » cas. Ces bons cas sont retrouvés à l'aide des index définis sur le nouveau cas. De cette façon, l'ensemble de cas similaires sur lesquels la recherche du meilleur cas est effectuée se retrouve considérablement réduit ce qui accélère le processus. Pour déterminer cet ensemble de « bons » cas, la recherche est souvent basée sur la correspondance entre les attributs du nouveau cas et ceux des cas en mémoire. Les cas peuvent être retrouvés simplement selon les attributs ou encore selon des caractéristiques inférées par les attributs. De plus, certains systèmes de RBC n'attribuent pas la même importance à tous les attributs de l'index, d'où l'utilisation de *poids*.

La technique la plus populaire pour évaluer la similarité entre deux cas est celle des *Plus Proches Voisins* (PPV, en anglais *nearest neighbors*). Cette technique ne s'applique par contre que sur les attributs dont les valeurs sont numériques. Pour les attributs avec des valeurs qualitatives, différentes heuristiques sont développées selon chaque système. La Figure 3.3 illustre la technique des PPV.

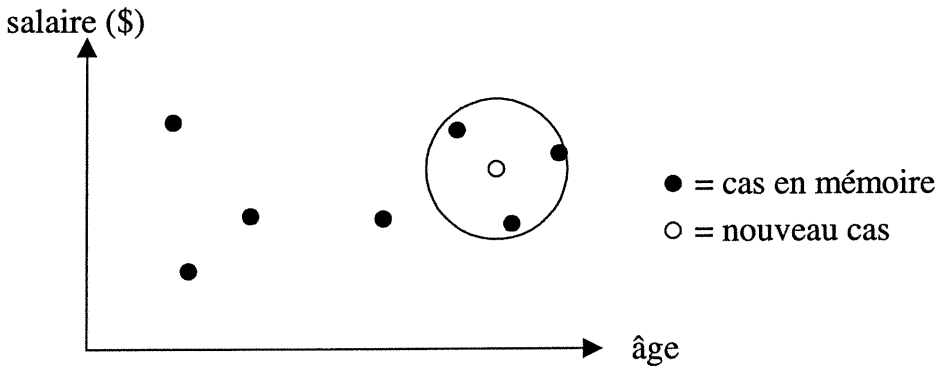


Figure 3.3 : Exemple du fonctionnement de la technique des plus proches voisins

Dans le but de simplifier la figure, les cas utilisés n'ont que deux attributs dans leur index. L'objectif est de trouver quels sont les cas « les plus près » du nouveau cas. Le calcul des PPV n'est en fait que la sommation de distances euclidiennes pondérées. Les *m* points les plus rapprochés du nouveau cas formeront l'ensemble des « bons » cas. La valeur de *m* est habituellement déterminée par l'usager. Le calcul de similarité selon la technique des plus proches voisins doit être effectué pour chaque cas en mémoire. Plus la similarité est petite plus le cas en mémoire est similaire au nouveau cas. Voici la formule classique des plus proches voisins :

$$similarité_i = \sum_{j=1}^{n} w_j * \left| x_{i,j} - x_{c,j} \right|$$

où

$i = 1, \ldots, nb\_cas$ est le nombre de cas dans la base de cas.

$j = 1, \ldots, nb\_attributs$ est le nombre d'attributs dans l'index des cas.

w_j est le poids de l'attribut j.

$x_{i,j}$ représente la valeur de l'attribut j du cas i.

$x_{c,j}$ représente la valeur de l'attribut j du nouveau cas.

Une fois ce calcul exécuté pour chaque cas, l'ensemble des « bons » cas est construit avec les *m* meilleurs cas ou avec les cas dont la similarité est plus petite qu'un certain seuil. Par exemple, supposons que $m = 1$ et que le nouveau cas est un client de 30 ans ayant un salaire de 30 000$. De plus, supposons que les poids sont tous égaux à 0.5 et que la base de cas contient deux cas : un client de 25 ans avec un salaire de 32 000 $ et un autre de 35 ans avec un salaire de 35 000 $. Ainsi, les calculs de similarité sont les suivants :

$$similarité_1 = 0.5 * | 25 - 30 | + 0.5 * | 32\,000 - 30\,000 | = 1002.5$$
$$similarité_2 = 0.5 * | 35 - 30 | + 0.5 * | 35\,000 - 30\,000 | = 2502.5$$

Ainsi, le client de 25 ans est plus similaire au nouveau client que celui de 35 ans.

Il existe plusieurs variantes de l'algorithme des plus proches voisins comme les plus proches voisins symétriques [NOC *et al.* 00] ou encore l'adaptation de cet algorithme selon la logique floue (*fuzzy PPV*) [KEL *et al.* 85].

De plus, d'autres fonctions de similarité peuvent être utilisées pour la recherche des cas les plus similaires dans la base. Une autre technique fréquemment employée est appelée méthode *par induction*. Elle consiste en fait à remplacer les PPV par un arbre de décision comme ceux vus dans le chapitre précédent. D'autres techniques de data mining présentées dans la section 2.1.1 pourraient aussi très bien remplacer la technique des PPV. Par exemple, des chercheurs de l'Université du Massachusetts utilisent les algorithmes génétiques dans leur système appelé OFF-BROADWAY [SKA 93].

L'étape de recherche dont nous venons de discuter est d'une importance capitale lorsque le but d'un système de RBC est de classer de nouveaux cas. Vu l'impact de cette étape, la section 3.4 présente les algorithmes de recherche les plus connus (excepté celui de PPV que nous venons de décrire).

3.3.2 Réutiliser

Cette étape permet de réutiliser le plus efficacement possible la solution du meilleur cas retrouvé à la première étape. Tout d'abord, la solution du meilleur cas en mémoire est copiée dans la solution du nouveau cas. Par contre, comme il est très rare que deux cas soient exactement identiques, la solution du nouveau cas doit être modifiée selon les différences qu'elle présente par rapport au meilleur cas.

L'adaptation d'une solution n'est pas obligatoire. Si les cas sont pratiquement identiques, aucune adaptation n'est nécessaire. En revanche, dès qu'une différence est présente entre deux cas, l'adaptation de la solution peut être nécessaire. Il est à noter que si la tâche du système de RBC en est une de classification, les différences sont peu importantes ainsi l'adaptation devient superflue. Par contre, pour toutes les autres tâches, si la modification de la solution est nécessaire, il faut d'abord identifier les parties à

adapter. L'identification de celles-ci peut s'effectuer de plusieurs façons, mais généralement le choix est régi par les différences présentes entre les cas. Selon Kolodner [KOL 93], un cas peut être adapté de deux façons :

- Par *substitution* : Ce genre d'adaptation consiste à choisir et à remplacer des parties d'anciennes solutions pour en former de nouvelles. Kolodner donne comme exemple un système de planification de menus. Un plat de lasagne à la viande peut être adapté en remplaçant la viande par des épinards comme le suggère une recette de pâtes végétariennes présente dans la base. L'adaptation de la recette peut aussi consister à augmenter les proportions pour satisfaire un groupe de 20 personnes. Cette méthode implique donc que certaines alternatives existent dans la base de cas.

- Par *transformation* : Cette méthode d'adaptation est utilisée si une partie d'une nouvelle solution doit être ajoutée ou éliminée ou encore lorsqu'aucune alternative n'existe pour la partie à adapter. Ce type d'adaptation travaille avec des contraintes et des heuristiques intuitives. Par exemple, bien qu'il n'existe pas de recette de lasagne contenant des carottes dans la base de cas, cet ingrédient peut être ajouté à la lasagne si la contrainte « doit contenir un légume orange » est exigée pour quelque raison que ce soit. Un autre exemple serait de remplacer le bœuf de la lasagne par du porc haché même si aucune recette de pâtes contenant du porc n'est présente dans la base de cas.

Les contraintes ou les démarches à suivre pour résoudre un cas selon ces deux techniques, sont basées sur des heuristiques et des stratégies prédéfinies par les experts du domaine d'application ainsi que par les développeurs du système. Il existe d'autres façons d'adapter les solutions d'un cas [KOL 93], mais celles-ci étant moins utilisées nous ne les expliquerons pas dans le cadre de ce travail.

À la sortie de l'étape de réutilisation, une solution est proposée au nouveau cas.

3.3.3 Réviser

Cette étape donne une dernière opportunité au système d'apprendre de ses erreurs. Si la solution proposée est correcte, aucune révision n'est nécessaire. Par contre, si la solution semble erronée, le système tentera de la corriger.

Pour déterminer si une solution est correcte ou non, plusieurs systèmes interagissent directement avec un expert en lui demandant son avis par rapport à la solution proposée. Le système s'ajustera alors selon les commentaires apportés par l'expert. Une autre possibilité serait de chercher le cas le plus similaire au cas résolu pour voir s'ils sont suffisamment similaires et ainsi confirmer si la nouvelle solution est réaliste. Finalement, la solution erronée est corrigée selon les résultats de ces révisions.

Cette étape est utile au bon fonctionnement des systèmes de RBC. En fait, elle permet au système d'apprendre de ses erreurs afin de ne pas les répéter et de s'améliorer tout en prenant de l'expérience. Ainsi, la sortie de cette avant-dernière étape donne un cas corrigé dont la solution est correcte.

3.3.4 Retenir

À cette étape, le système mémorise le cas corrigé pour en faire une utilisation ultérieure. Par contre, le système de RBC ne mémorise pas systématiquement tous les nouveaux cas rencontrés. En effet, il évite autant que possible de créer des doublons dans sa base de cas.

Le plus difficile à cette étape est de choisir un bon index pour le nouveau cas. En réalité, nous revenons au problème d'indexation présenté précédemment. Il faut déterminer le bon index pour que le nouveau cas en mémoire soit retrouvé au bon moment et dans le bon contexte. Certains systèmes utilisent automatiquement tous les attributs pour former le nouvel index, d'autres utilisent une combinaison des index des cas les plus similaires au cas à retenir.

Finalement, le cas est intégré et tous les index et les poids sont mis à jour selon leurs performances. Un cas ayant mal influencé la solution du nouveau cas se verra puni et ses attributs indexés auront des poids plus faibles et vice-versa pour les cas utiles ou les attributs ayant eu une influence positive.

3.4 Algorithmes de recherche les plus populaires

Notre travail effectuant une tâche de classification, l'étape de recherche est la plus importante à la réussite de notre système. L'algorithme des plus proches voisins est la technique la plus fréquemment utilisée pour retrouver les cas les plus similaires dans une base de cas. Cette technique a été décrite dans la section précédente. Dans la présente section, nous aborderons trois algorithmes de recherche bien connus dans le domaine du raisonnement à base de cas. Tous ces algorithmes servent à la classification de nouveaux cas.

Premièrement, la technique utilisée dans le système PROTOS est présentée. Ce système de classification utilisant un réseau de liens est très bien connu dans la communauté des chercheurs en RBC. Deuxièmement, nous présentons la recherche de cas selon l'induction de règles. Troisièmement, une technique basée sur le fonctionnement de la logique et des ensembles flous est introduite. Cet algorithme comme nous allons le constater offre de nouvelles options intéressantes à la recherche plus classique utilisée jusqu'à maintenant.

3.4.1 Réseau de liens

Le système de raisonnement à base de cas PROTOS a été développé par les chercheurs Ray Bareiss et Bruce Porter à la fin des années 1980 [BAR 89, BAR *et al.* 88, POR *et al.* 90]. Ce système effectue une tâche de classification de cas et gère l'acquisition de nouvelles connaissances. PROTOS est présenté comme un système à base d'exemples, mais il peut être considéré comme un système à base de cas, car la

définition des « exemples » de Bareiss et Porter est la même que celle des cas en RBC. Ajoutons que PROTOS est un système largement reconnu dans la communauté des chercheurs en raisonnement à base de cas. Les créateurs de PROTOS ont évalué les performances du système à l'aide de données venant du domaine de l'audiologie. Expliquons plus en détail l'algorithme de classification de PROTOS avec un exemple tiré de ce domaine.

Pour arriver à effectuer correctement sa classification, la mémoire de PROTOS contient un *réseau de liens*. Le système se guide à l'aide de ces liens pour arriver à trouver le cas le plus similaire au nouveau cas (étape de recherche). Les cas dans PROTOS sont des patients atteints de maladies de l'oreille. Il existe quatre types de liens dans ce système :

- Liens associatifs : Ces liens sont les premiers à être utilisés. Ils permettent d'identifier la *catégorie* la plus similaire au nouveau cas. Pour PROTOS, chaque maladie de l'oreille forme une catégorie différente. Ces liens permettent de lier les attributs d'un nouveau cas aux différentes catégories. De plus, ces liens sont pondérés selon l'importance de l'attribut dans la description de la maladie la plus plausible. La catégorie (ou maladie) choisie comme étant la plus similaire forme ce que PROTOS appelle l'*hypothèse* de base.

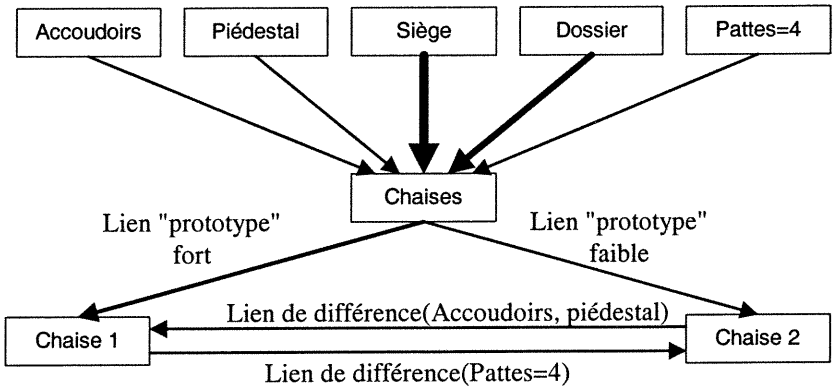
- Liens de censure : Ce sont des liens dits négatifs. Ils permettent à PROTOS d'éviter de suivre de mauvais liens associatifs dans certaines situations bien précises. Les liens de censure évitent à PROTOS de répéter ses erreurs.

- Liens « prototypes » : Ces liens associent les cas de la base de cas aux catégories. Ils sont utilisés pour rechercher le cas le plus similaire au nouveau cas dans la catégorie suggérée par les liens

associatifs. Il existe deux types de liens « prototypes » : les liens forts et les liens faibles. Les cas les plus représentatifs d'une catégorie sont marqués comme étant forts. PROTOS choisira les liens forts en premier lieu et ce n'est seulement que si ces liens ne donnent pas de bons résultats que le système ira consulter les liens faibles.

- Liens de différence : Ces liens relient les cas de la base en tenant compte des différences entre ceux-ci. PROTOS les utilise pour se diriger dans sa recherche afin de dénicher la meilleure classification. Chacun de ces liens est identifié par les attributs dont les valeurs diffèrent entre les deux cas qu'il relie. Ces liens de différence sont essentiels à la réussite de la classification. En fait, avant de suivre un de ces liens, PROTOS évalue la similarité entre le cas présent et le nouveau cas. Si la similarité est suffisamment bonne, l'hypothèse du lien associatif de départ est confirmée. Sinon, les attributs les plus divergents entre les deux cas sont identifiés et selon les différences identifiées PROTOS suivra le lien de différence approprié afin de trouver un autre cas plus similaire. Les liens de différence sont suivis jusqu'à ce que le niveau de similarité soit satisfaisant.

La Figure 3.4 tirée de [BAR *et al.* 88], présente un réseau de liens bien simple. Cette figure représente un exemple d'une catégorie nommée *Chaise* qui contient deux cas. Les liens associatifs entre la catégorie et cinq attributs sont visibles dans la partie supérieure de la figure. Les liens de censure bien qu'absents de cette figure sont aussi à ce niveau, c'est-à-dire entre les attributs et les catégories. Nous retrouvons aussi deux liens « prototypes », un fort et un faible ainsi que deux liens de différence qui permettent de se déplacer d'un cas à l'autre.



Les flèches sans étiquette sont des liens associatifs.
L'épaisseur des flèches représente le poids de l'attribut.

Figure 3.4 : Exemple des différents liens dans le système PROTOS

Le réseau de liens utilisé par PROTOS s'enrichit à l'aide des interactions entre le système et un expert. Dès qu'un expert en audiologie rejette la classification offerte par PROTOS, ce dernier demandera des explications à l'expert. Ces explications sont formulées selon le vocabulaire compris par le système et elles lui permettent de créer de nouveaux liens offrant de nouvelles directions à explorer. Les nouveaux liens ajoutés seront à leur tour pondérés et les poids des liens utilisés seront ajustés selon leur influence sur la qualité de la classification. C'est ainsi que PROTOS arrive à se perfectionner et à donner des classifications correctes.

3.4.2 Induction de règles

L'induction de règles est un principe provenant du domaine de l'apprentissage machine [MIC *et al.* 83, MIC *et al.* 86, MIC *et al.* 98] qui permet de gérer de larges bases de cas. En fait, cet algorithme construit une structure d'arbre avec les cas de la base du système et crée un ensemble de règles utiles à la recherche des cas similaires. Les algorithmes d'arbres de décision dont nous avons discuté à la section 2.4.3 sont des exemples de méthodes de recherche par induction de règles.

Lorsqu'un nouveau cas se présente, ce genre d'algorithme confronte les attributs du nouveau cas avec chacune des conditions de chacune des règles induites par l'arbre. Finalement, l'algorithme classe le cas dans la catégorie dictée par la règle la plus appropriée. Comme nous avons déjà amplement présenté le fonctionnement des arbres de décision à la section 2.4, nous n'élaborerons pas plus sur ce sujet.

Le système INERCA fonctionne sous ce principe [KLA *et al.* 95]. Il utilise des *arbres k-d* pour simplifier son processus de recherche des cas les plus similaires. Le système ELEM2-CBR [CER *et al.* 99] présenté dans le premier chapitre est un autre bon exemple de ce genre de systèmes.

3.4.3 Logique et ensembles flous

Depuis moins d'une dizaine d'années, plusieurs systèmes de RBC ont intégré à leurs techniques de recherche de cas la notion de *logique floue* ou d'*ensembles flous*. Les ensembles flous ont été introduits en 1965 par Lotfi Zadeh [ZAD 65]. Les notions de logique et d'ensembles flous jonglent avec les concepts d'incertitude et d'imprécision. L'ouvrage de Klir et Yuan traite plus en profondeur de ces concepts et de leur utilisation dans différents domaines d'application [KLI et YUA 95]. Discutons plutôt de l'application de la logique et des ensembles flous dans les systèmes de RBC.

Dans le domaine du RBC, les principes de la logique et des ensembles flous sont principalement appliqués à la représentation des cas et au processus de recherche des cas les plus similaires.

La première application consiste à représenter les cas de manière floue. En d'autres mots, les attributs présents dans les cas d'une base de cas peuvent être transformés en attributs flous. Par exemple, la valeur de l'attribut représentant l'âge d'un client de 45 ans sera transformée en un intervalle entre 40 et 50 ans et des probabilités sont attribuées en partant de un pour la valeur 45 jusqu'à zéro pour les valeur 40 et 50.

Le papier de Liao *et al.* présente ce principe ainsi que les avantages et inconvénients qu'il apporte au processus de RBC [LIA *et al.* 98].

La deuxième application utilise le fait qu'il apparaît naturel d'appliquer la logique et les ensembles flous au processus de recherche des cas les plus similaires. En effet, la notion de similarité comme nous l'utilisons couramment est RBC est un concept qui peut être qualifié de flou. Par exemple, dans quelles limites peut-on affirmer que deux patients peuvent recevoir le même traitement lorsque leurs symptômes sont semblables ? En fait, la logique floue permet de travailler avec l'ambiguïté et l'incertitude entraînées par la définition de similarité du point de vue humain.

Le principe de la logique et des ensembles flous appliqués à la notion de similarité du raisonnement à base de cas est énoncé comme suit par Dubois *et al.* [DUB *et al.* 97] :

> *« The more similar are the problem description attributes, the more similar are the outcome attributes. »*

Ce principe peut être énoncé de façon plus formelle. Soit un cas défini selon deux parties : le problème et la solution. La partie problème, notée P, est formée d'un ensemble de n attributs caractérisant le problème tandis que la partie solution, notée S, décrit la solution au problème. Un cas est donc constitué d'une paire (p, s) formée des valeurs des attributs du problème et de la solution. Ce qui revient à dire que :

Plus p_1 et p_2 sont P-similaires, plus s_1 et s_2 sont S-similaires.

Cet énoncé s'applique aux problèmes dits déterministes. La définition des problèmes déterministes et plusieurs autres détails techniques sur cette théorie sont explicitement présentés dans [DUB *et al.* 97].

Keller *et al.* présentent une méthode de recherche des cas similaires basée sur le principe des plus proches voisins à laquelle les principes de la logique et des ensembles flous ont été intégrés [KEL *et al.* 85]. Cette méthode se nomme *Fuzzy Nearest Neighbors*. La différence entre l'algorithme des plus proches voisins classiques et l'algorithme flou est principalement au niveau du classement. Au lieu de classer simplement le nouveau cas directement dans une seule classe, l'algorithme des plus proches voisins flous donne la probabilité d'appartenance du cas pour chaque catégorie. La méthode développée par Keller *et al.* a été appliquée avec succès à l'intérieur d'un système de raisonnement à base de cas dans le domaine de la météorologie [BJA 01]. Il est intéressant de noter que la logique et les ensembles flous ont déjà été largement utilisés dans ce domaine [BJA 01].

3.5 Conclusion

Ce chapitre a introduit le raisonnement à base de cas, une technique d'intelligence artificielle calquée sur le mode de raisonnement humain. Un système de raisonnement à base de cas utilise des solutions liées à certains problèmes qu'il a préalablement expérimentées afin de générer de nouvelles solutions à de nouveaux problèmes. Ce processus de raisonnement est divisé en quatre étapes (Retrouver, Réutiliser, Réviser et Retenir) et effectue deux types de tâches (résolution de problèmes et interprétation).

Notre prototype, présenté dans le chapitre quatre, implémente une tâche d'interprétation : il « interprète » les caractéristiques d'un nouveau profil par rapport aux profils de la base afin de le classer dans la catégorie qui le représente le mieux. C'est à partir de cette catégorie que les experts en marketing décideront quels produits et services offrir à ce client.

La technique des plus proches voisins est utilisée dans l'étape de recherche du processus de RBC de notre prototype. Nous avons privilégié cette technique plutôt que celles présentées dans la section 3.4 pour plusieurs raisons. Tout d'abord, nous avons

éliminé la possibilité d'utiliser un réseau de liens, car il exigeait le support continuel d'un expert de la banqueX. Nous utilisons déjà un algorithme d'induction de règles (algorithme C4.5) dans le système hybride, donc en utiliser un autre pour le processus de RBC serait illogique. De plus, nous préférons nous appuyer sur les bases stables des PPV plutôt que sur les principes de la logique et des ensembles flous qui ont plus ou moins été utilisés en RBC.

Ce chapitre conclut la présentation de la théorie sur laquelle notre prototype est basé. Le chapitre quatre présente, en premier lieu, le processus de développement de notre projet. En deuxième lieu, il donne des détails quant à l'utilisation combinée que nous avons faite des techniques de data mining et du raisonnement à base de cas. Puis, suivent les résultats et les discussions sur le prototype.

Chapitre 4 : Prototype

Ce que nous devons apprendre à faire, nous l'apprenons en le faisant...
- ARISTOTLE, Ethics

Rappelons tout d'abord l'objectif de notre travail :

> *Proposer une alternative simple, efficace et intuitive aux méthodes statistiques pour la classification de profils de clients dans le but de déterminer leurs besoins en produits et services bancaires.*

En d'autres mots, ce projet vise à développer un prototype capable de classer les nouveaux profils des clients d'une banque afin de déterminer selon leurs caractéristiques quels produits et services bancaires seraient susceptibles de les intéresser. De plus, les algorithmes utilisés dans notre prototype se doivent d'être intuitifs car les usagers du système n'auront pas nécessairement de base en mathématiques ni en statistique.

Pour atteindre cet objectif, notre prototype utilise les deux techniques présentées dans les chapitres précédents, soient le data mining et le raisonnement à base de cas.

Notre choix s'est arrêté sur ces deux techniques, car elles reposent sur des bases beaucoup plus intuitives que celles des techniques statistiques actuellement utilisées à la banqueX. Le prototype présenté dans ce chapitre combine l'utilisation de ces techniques afin d'obtenir le classement le plus probable du nouveau profil d'un client et ainsi répondre de façon efficace à ses besoins bancaires.

Ce chapitre présente tout d'abord, dans la section 4.1, une vue générale de notre prototype selon les phases de son développement. La section 4.2 présente l'architecture de notre prototype ainsi que les algorithmes choisis. L'évaluation du système développé ainsi que les résultats des tests sont donnés dans la section 4.3. Finalement, les améliorations et extensions possibles à notre prototype se retrouvent dans la section 4.4.

4.1 Phases du développement

Cette section présente de façon générale les différentes phases du développement de ce projet. La Figure 4.1 ci-dessous illustre ces cinq phases.

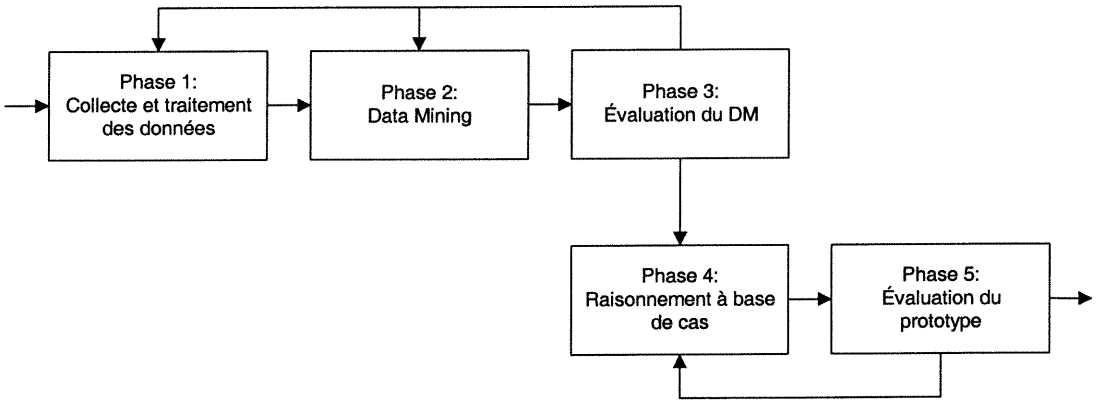


Figure 4.1 : Déroulement global du projet

Expliquons brièvement chacune de ces phases. Il est à noter que la section 4.2 décrit plus en détails ce que nous présentons ici.

Phase 1 : Collecte et traitement des données

Cette première phase, consiste à rassembler les données nécessaires à notre étude et à choisir les attributs utilisés par la suite.

Les données (formant les profils de clients) utilisées tout au long de ce travail proviennent de différentes bases de données du département d'informations stratégiques de la banqueX. Tout d'abord, l'ensemble des données nécessaires à ce travail sont intégrées en une seule base. Cette étape est celle de *Sélection et d'intégration* du processus de data mining.

Par la suite, nous révisons la base de données ainsi obtenue pour éliminer les profils de clients incohérents. Des exemples de profils incohérents sont présentés en 4.2.2.2. C'est également dans cette phase que certains attributs de la base sont transformés. Dans le processus de data mining, cette étape consiste à effectuer le *Nettoyage et la transformation* des données contenues dans la base.

Précisons que les données fournies par la banque sont de différentes natures. Certaines sont numériques multivaluées, d'autres sont symboliques. Voici des exemples concrets des valeurs que peuvent prendre ces différents types de données :

Numériques (quantitatives) multivaluées :

Âge : {0, 1, ..., 98, 99, 100, 101}

Nombre de REER : {0, 1, 2, 3, 4, 5}

Solde des prêts personnels : {0, 1000, 2500, 3000, ..., 30 000, 35 500}

Symboliques (qualitatives) :

Sexe : {F, M}

Code postal : {A3K 1U3, A5S 2J3, ..., Z2E 2L8}

Phase 2 : Data Mining

La deuxième phase de ce projet renferme la *technique de data mining* proprement dite. Cette phase permet d'obtenir un ensemble de règles définissant les caractéristiques de chaque catégorie[6] de clients définie par la banqueX. Ces règles représentent des informations sur l'utilisation que font les clients des différents produits et services offerts par la banque. Le nombre de prêts hypothécaires qu'un individu possède ou le montant original de ses prêts sont des exemples d'attributs bancaires. Voici un exemple de règle utilisant ces attributs : « les clients dont le profil est dans la catégorie 5 possèdent un ou plusieurs prêts hypothécaires dont le montant original est supérieur à 80 000$ ». Les caractéristiques des catégories de clients, qui sont mises en évidence à cette étape via les règles produites, nous sont utiles pour classer les profils des clients ainsi que pour déterminer quels sont leurs besoins en services et produits financiers.

La technique de DM sélectionnée pour accomplir ce travail est l'algorithme C4.5 [QUI 93] basé sur les arbres de décision. Un bref rappel de cet algorithme ainsi qu'un court exemple de son fonctionnement sur les données de la banqueX est donné plus loin (section 4.2.3). Cet algorithme a été sélectionné pour plusieurs raisons. Premièrement, le data mining est une méthode bien connue d'extraction d'informations dans de larges bases de données. Deuxièmement, il nous permet de générer de façon simple et efficace des règles définissant les caractéristiques générales des catégories de clients. Troisièmement, parmi les algorithmes de data mining ce sont les arbres de décision qui sont les plus simples et les plus intuitifs. Ceci est un grand avantage puisque notre but est d'offrir une alternative intuitive aux gens n'ayant pas ou peu de connaissances avancées en statistiques.

[6] Ici, contrairement à la section 2.4, nous référons aux différentes valeurs de l'attribut dépendant (ou catégorie) directement sous le nom de *catégories*. Cette nuance est apportée afin d'alléger le texte.

Le data mining permet donc de bien déterminer la structure générale de chacune des catégories de clients. Ces structures pourront ensuite servir de guide au classement d'un nouveau profil de client.

Phase 3 : Évaluation du DM

La troisième phase évalue l'efficacité de l'algorithme de data mining employé dans la phase deux. Cette phase représente l'étape *d'analyse, d'évaluation et de présentation* du processus de data mining. Nous y évaluons les résultats obtenus par l'algorithme de construction d'arbre de décision C4.5 ainsi que la qualité de la simplification de l'ensemble des règles qu'il a effectuée.

Lorsque les résultats obtenus ne sont pas satisfaisants, les deux premières phases doivent être revues. Un taux de réussite de classification des nouveaux cas trop bas ou un arbre trop complexe sont deux exemples de résultats insatisfaisants. Dans ces situations, une analyse approfondie des données de départ pourrait être nécessaire.

Phase 4 : Raisonnement à Base de Cas

Le processus de raisonnement à base de cas est développé dans la quatrième phase du développement. Cette phase implémente et ajuste le système de raisonnement à base de cas. Rappelons que les quatre étapes (quatre « R ») constituant le cycle du RBC sont : Retrouver, Réutiliser, Réviser et Retenir. Le prototype utilise les étapes Retrouver et Réutiliser, les deux autres n'étant pas nécessaires pour effectuer une tâche de classification. Nous verrons comment ces deux étapes sont intégrées dans le prototype un peu plus loin dans ce chapitre.

Un échantillon aléatoire de profils de clients fournis par la banqueX est utilisé pour constituer la base de cas initiale. Une fois cette base initialisée, nous utilisons le processus de RBC pour classer des profils seulement si un conflit est survenu lors du classement selon les règles générées par les arbres de décision.

Ainsi, des connaissances spécifiques sur les clients de la banque sont utilisées sous la forme de cas. Le RBC est donc employé seulement si les connaissances générales ne sont pas suffisantes pour arriver à classer un nouveau profil de client.

Phase 5 : Évaluation du prototype

La cinquième phase évalue la qualité du système dans son ensemble. L'évaluation du prototype consiste à classer des nouveaux profils de clients et à comparer le résultat de cette classification avec celle obtenue selon des méthodes statistiques de la banqueX.

Nous avons émis comme postulat de base que le processus de classification statistique de la banqueX obtient un taux de réussite de classement de 100% (ce qui n'est fort probablement pas le cas). Ce postulat sert principalement à l'évaluation des résultats obtenus avec le prototype développé dans ce travail. Si les résultats ne sont pas satisfaisants, le processus employé dans le prototype doit être revu.

D'un autre point de vue, le prototype fournit également un outil de comparaison aux experts de la banqueX. En effet, les résultats obtenus avec le prototype permettent de valider la classification statistique de la banque. Si le prototype n'obtient pas un taux de réussite de 100%, alors les experts de la banque pourraient être appelés, à leur tour, à étudier ses résultats. Cette analyse permettrait éventuellement à la banqueX de déterminer la nature de certaines améliorations pouvant être intéressantes à apporter dans leur processus de classification.

L'objectif et les phases du déroulement de ce travail étant précisés, présentons maintenant l'architecture du prototype et les algorithmes utilisés pour effectuer la classification.

4.2 Architecture du prototype

L'architecture ainsi que le fonctionnement du prototype sont inspirés en grande partie du système ELEM2-CBR présenté dans le premier chapitre. Le prototype est, comme ELEM2-CBR, un système hybride d'Induction de Règles (IR) et de Raisonnement à Base de Cas (RBC). Rappelons que l'induction de règles est un principe qui englobe tous les algorithmes produisant des ensembles ou systèmes de règles à partir de groupes de données. La technique d'IR que nous utilisons dans ce prototype en est une de Data Mining (DM) : l'algorithme d'Arbres de Décision C4.5 (AD).

Deux points différencient le prototype proposé et le système ELEM2-CBR. La technique d'IR utilisée à l'intérieur du prototype présenté dans ce travail n'est pas la même que celle employée par ELEM2-CBR. L'algorithme C4.5 étant plus reconnu et plus éprouvé que ELEM2 (utilisée par ELEM2-CBR), nous avons donc opté pour cet algorithme. Une autre différence réside dans l'étape de RBC où les données quantitatives doivent être discrétisées. Pour effectuer cette transformation, nous utilisons un principe tiré du domaine de la logique et des ensembles flous contrairement à ELEM2-CBR qui utilise une heuristique basée sur l'entropie. Ce choix nous permet de sauver du temps de calcul tout en obtenant des résultats satisfaisants.

La section suivante présente le processus de classification utilisé dans le prototype en donnant un aperçu général de son fonctionnement. Les sections 4.2.2, 4.2.3 et 4.2.4, quant à elles, décrivent en détails les différentes parties reliées au classement de profils de clients.

4.2.1 Processus de classification

Comme nous venons de le mentionner, l'architecture du prototype implémente une combinaison des techniques de raisonnement à base de cas et d'induction de règles. L'approche hybride suggérée présente l'avantage de combiner des propriétés

complémentaires du RBC et des AD. Cette approche permet de solutionner des problèmes que ni le RBC ni les AD utilisés seuls n'auraient réussi à solutionner de façon satisfaisante. Ceci est dû au fait que les arbres de décision utilisent des connaissances plus générales sous forme de règles, tandis que le raisonnement à base de cas utilise des connaissances spécifiques du domaine d'application sous la forme de cas. Ainsi, nous combinons deux types de connaissances complémentaires (générales et spécifiques) ce qui permet au système d'obtenir de meilleurs résultats. Il est à noter que cette approche hybride basée sur ELEM2-CBR n'utilise pas les AD comme support au RBC ni le RBC comme support aux AD comme certains systèmes le font avec d'autres techniques d'induction de règles [KOT 89, SKA et RIS 90]. Le prototype hybride que nous avons développé utilise plutôt une approche qui combine les techniques de RBC et des AD afin qu'elles se supportent mutuellement. En d'autres mots, aucune n'occupe seulement un rôle de support pour l'autre.

La Figure 4.2 illustre le processus selon lequel le prototype classifie les nouveaux profils des clients. Tout d'abord, la base de cas est initialisée et un ensemble de règles est généré selon l'algorithme d'arbres de décision C4.5.

Par la suite, le nouveau profil du client est classé selon les règles induites à partir de la base de cas. Un ensemble de règles couvrant le nouveau profil du client est ainsi obtenu et analysé afin de déterminer si une situation de conflit survient au niveau du classement suggéré par ces dernières. Par exemple, une situation conflictuelle survient lorsque deux règles couvrent le nouveau profil du client et que la première suggère de le classer dans la catégorie 1 tandis que l'autre suggère la catégorie 5. S'il n'y a pas de conflit alors le nouveau profil du client est classé dans la catégorie suggérée par la ou les règles.

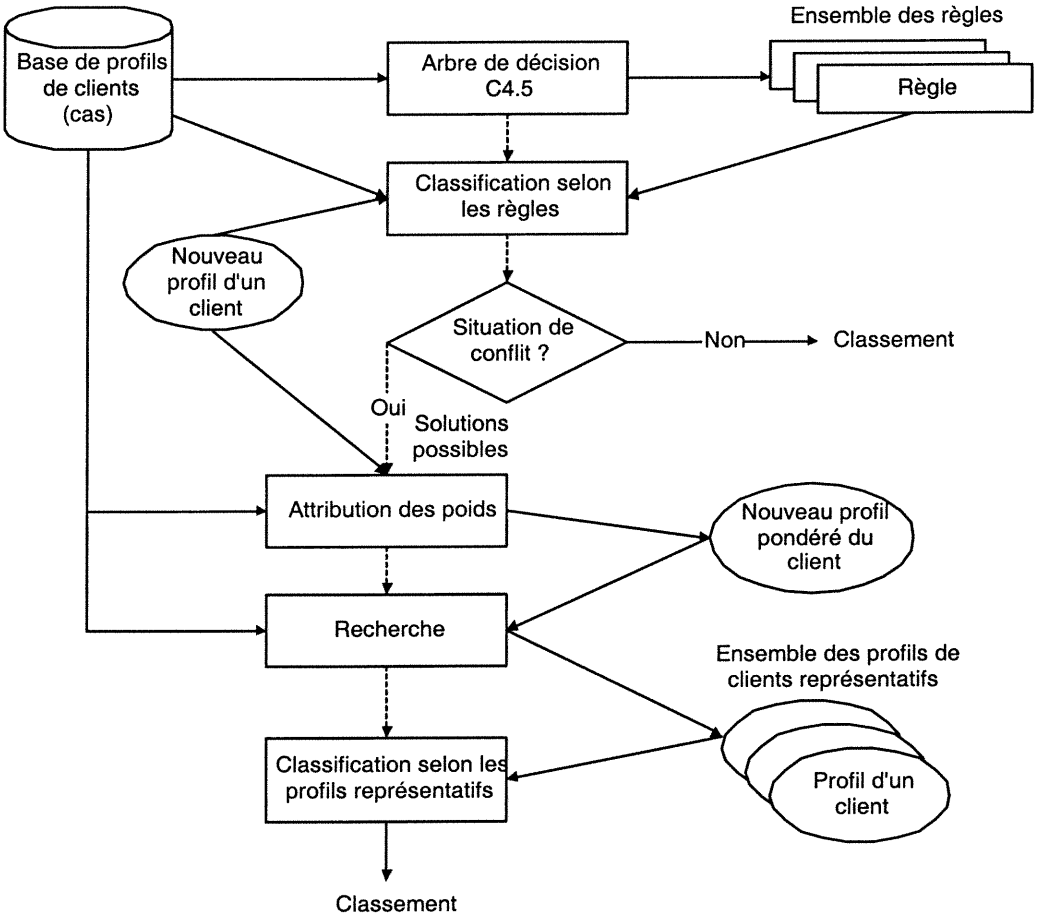


Figure 4.2 : Processus de classification du prototype

En revanche, si certaines règles entrent en conflit, c'est plutôt le processus de raisonnement à base de cas qui détermine quelle catégorie, parmi celles suggérées, doit être attribuée au profil. Pour se faire, une recherche ayant pour but d'identifier efficacement les cas les plus similaires au nouveau profil du client est effectuée dans la base de cas. Afin d'arriver à obtenir une bonne classification, les attributs du nouveau profil de client sont tout d'abord pondérés pour refléter leur niveau d'influence sur les caractéristiques du client. Par exemple, l'attribut *nombre de REER* du profil dont la valeur est de deux (*nombre de REER* = 2) peut obtenir un poids de 0.9 car il est très représentatif de ce client. Pour ce même client, l'attribut *nombre de comptes de transactions*, fixé à zéro, se voit concéder un poids de 0.2, car il représente beaucoup

moins bien le profil du client. L'attribution des poids du nouveau cas est effectuée par l'algorithme selon une équation mathématique présentée dans la section 4.2.4.

Par la suite, une mesure de similarité entre chaque cas de la base et le nouveau profil du client est calculée. Cette recherche des cas les plus similaires est basée sur la proximité de chaque attribut. Plus les valeurs des deux attributs sont proches, plus le cas et le nouveau profil du client sont similaires. Par exemple, un cas possédant un REER et un compte de transactions est plus similaire au profil du client donné en exemple ci-dessus qu'un autre cas ayant sept REER et trois comptes de transactions. Seuls les cas les plus similaires sont conservés (étape de Recherche) et le classement du nouveau profil du client est déterminé à partir du classement suggéré par ce petit groupe de cas représentatifs (étape de Réutilisation).

Le survol de la méthode hybride employée dans ce travail étant maintenant terminé, présentons les sections suivantes détaillant le processus de classification (Figure 4.2). Les trois prochaines sections décrivent l'initialisation de la base de cas, la construction de l'ensemble de règles, l'équation de calcul de poids et l'algorithme de recherche du processus de RBC ainsi que les algorithmes de classification. Ces sections décrivent donc les phases 1, 2 et 4 de la Figure 4.1. Les phases 3 et 5 seront présentées dans la partie évaluation de ce chapitre (section 4.3)

4.2.2 Base de cas

Pour la réalisation du prototype nous avons eu accès à un échantillon de profils bancaires de clients réels d'une grande banque canadienne. La banqueX conserve dans différentes BD des millions de données sur leurs clients : transactions effectuées, produits détenus, services utilisés et autres. Ces données, tant quantitatives que qualitatives, se retrouvent dans différents types de bases de données et sont utilisées par les différents départements de la banque.

La banqueX nous a fourni un échantillon d'environ 11 000 profils de clients possédant chacun 210 attributs[7] distribués dans deux bases de données de formats différents. Cet échantillon de profils et d'attributs, sélectionné par les experts en statistique de la banqueX de façon aléatoire, conserve cependant les proportions de la base de données originale. La première partie de ce travail consiste à sélectionner, intégrer, nettoyer et transformer les données suffisantes et nécessaires à l'accomplissement de notre objectif. Le résultat de ces nombreuses étapes constitue la base de cas initiale.

4.2.2.1 Sélection et intégration

L'intégration des bases de données fournies par la banqueX a été simple. Deux bases de données nous ont été fournies : l'une sous format Excel'97 et l'autre sous format SPSS. Une base de données sous SPSS peut facilement être convertie vers le format Excel'97. Ainsi, nous avons obtenu une base de données au format Excel'97 contenant 210 attributs numériques multivalués et symboliques.

Parmi ces 210 attributs, seuls ceux qui sont constamment utilisés en marketing ont été conservés. Ainsi, certains attributs ont pu être retirés de la base de données car ils n'étaient pas considérés comme nécessaires d'un point de vue marketing. Par exemple, le nombre de transactions effectuées au guichet automatique par un client n'est pas un attribut pertinent au développement de campagnes publicitaires. Le choix des attributs les plus importants pour différentes campagnes publicitaires a été réalisé par les experts de la banqueX. Après cette sélection, seulement 24 attributs ont été retenus.

4.2.2.2 Nettoyage et transformation

Le nettoyage des données de la base consistait principalement à s'assurer que les données de chaque profil étaient cohérentes. À cette étape, nous procédons donc à un

[7] Il est à noter que l'attribut dépendant dans ces BD se nomme *Catégorie* et possède six valeurs : {*Catégorie1*, *Catégorie2*, ..., *Catégorie5*, *Catégorie6*}.

traitement de cohérence statique, c'est-à-dire sur les données. Aucun traitement dynamique n'est effectué. Ce traitement statique doit premièrement vérifier qu'il n'y a pas de redondance dans les données. Cette vérification consiste donc à s'assurer qu'aucun profil de client ne se retrouve plus d'une fois dans la base. La base de données contient le numéro de client de chaque profil et ce numéro est unique à chaque client. Après vérification, chaque numéro de client n'est présent qu'une seule fois dans la base. Deuxièmement, certaines contraintes du type « si *X* alors *Y* doit être zéro » doivent être respectées sur les données. Plusieurs attributs peuvent être mis deux par deux de cette façon : « si le *nombre de comptes* est de zéro alors le *solde de ces comptes* doit être à zéro ». Par exemple, si le nombre de comptes REER d'un client est à zéro alors le solde de ces comptes doit obligatoirement être lui aussi à zéro. Un exemple d'un autre type de contraintes sur les données est le suivant : certains profils de clients ont été éliminés, car ils énonçaient que certains des clients de la banqueX étaient inscrits à la banque depuis plus longtemps que leur propre âge! Par exemple, dans la BD nous avons trouvé un client qui fait affaire avec la banqueX depuis 50 ans et pourtant son profil indique qu'il n'a que 34 ans. La très grande majorité des incohérences dans les données ont été de ce type. Une partie du bruit est ainsi éliminé des données.

De plus, certains autres attributs ont été transformés afin d'optimiser l'information contenue dans la base. La transformation d'attributs est délicate car le développement des règles représentant les catégories de profils de clients est basé sur les attributs disponibles dans la base. Si un attribut est remplacé ou transformé, il peut y avoir une perte importante d'informations. Habituellement, les transformations sur les attributs sont laissées à la discrétion des experts du domaine d'application (dans notre cas ce sont les statisticiens de la banqueX). Une seule transformation a été exécutée dans la base et elle se rapporte aux REER. Huit attributs concernant les REER ont été additionnés quatre par quatre pour obtenir deux attributs représentant toutes les informations liées aux REER des clients. Ceci nous a permis de réduire la redondance de l'information de la base sans pour autant en diminuer la qualité.

Ainsi, suite à ces deux étapes, la base de cas contient une dizaine de profils en moins et chaque profil est représenté par les 18 attributs les plus importants selon la banqueX. La liste de ces 18 attributs numériques et leur signification est présentée dans l'Annexe A. De plus, chaque profil de client a été préalablement classé dans six catégories par la banqueX selon certaines techniques statistiques. Ces catégories ne peuvent être définies plus précisément pour des raisons de confidentialité.

La figure suivante (Figure 4.3) illustre de façon hiérarchique la structure des attributs d'un cas. Chaque feuille représente un attribut, il y en a 23 au total. Les attributs en caractères gras sont ceux utilisés dans le prototype, il y en a 18. Les cinq autres attributs présentés, comme le code postal, le sexe et le numéro de client, sont importants du point de vue de la banque, car ils leur permettent de retracer les dossiers des clients et de personnaliser l'interaction avec ces derniers. Ces attributs sont seulement utilisés dans un but d'identification et sont inutiles à la classification des profils. Ces cinq attributs d'identification ne sont donc pas considérés comme faisant partie des cas dans les bases que nous utilisons.

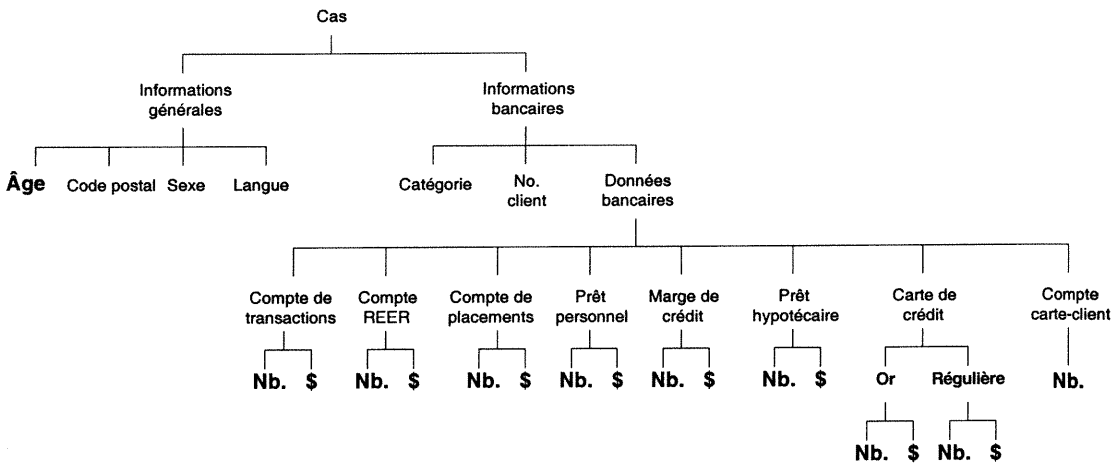


Figure 4.3 : Structure des cas utilisés par le prototype

Dans la base de cas, chaque profil de client est représenté par une ligne contenant ses 18 attributs ainsi que sa catégorie. Voici à quoi ressemblent les lignes qui constituent la base de cas :

Âge	Nb. de comptes de transactions	Solde moyen des comptes de transactions	Nb. de comptes REER	Solde des comptes REER	...	Nb. de cartes de crédit régulière	Limite de crédit des cartes de crédit régulières	Nb. de comptes carte-client	CATÉGORIE
44	3	4135	5	30199	...	0	0	1	2
38	1	150	1	8072	...	0	0	1	6
29	2	9517	1	5086	...	0	0	1	1
31	0	0	1	154	...	1	2000	0	2
59	1	6005	0	0	...	0	0	1	6
35	3	250	1	239	...	0	0	1	3

Ces étapes de sélection, d'intégration, de nettoyage et de transformation des BD et des données elles-mêmes ont nécessité environ deux mois de travail. Plusieurs analyses statistiques telles que des analyses de fréquences et de dépendances ont été menées. De plus, des discussions avec les experts de la banqueX nous ont permis de bien connaître les différents attributs et les liens qu'ils possèdent entre eux. De cette façon, les profils des clients formant la base de cas sont tous cohérents et la qualité de la base n'en est qu'améliorée.

4.2.3 Construction de l'arbre et des règles

À partir de la base de cas, nous allons construire un ensemble de règles décrivant les caractéristiques des catégories de clients. Ces règles servent à déterminer la catégorie du nouveau profil d'un client lors d'un « premier » classement. Pour créer cet ensemble de règles, nous avons implémenté l'algorithme C4.5. Rappelons que cet algorithme construit un arbre de décision et simplifie l'ensemble des règles de production obtenu par cet arbre. L'arbre de décision est construit selon le gain d'entropie apporté par un

certain attribut X sur l'ensemble de la base S. Rappelons les équations utilisées pour calculer ce gain :

$$gainRatio(X) = \frac{gain(X)}{splitInfo(X)}$$

où

$$SplitInfo(X) = \sum_{i=1}^{n} \frac{|S_i|}{|S|} * \log_2\left(\frac{|S_i|}{|S|}\right)$$

et

$$gain(X) = -\sum_{i=1}^{n} p_i * \log_2(p_i) - \sum_{i=1}^{n} \frac{|S_i|}{|S|} * \text{info}(S_i)$$

Les détails et les explications concernant ces formules ont été donnés à la section 2.4.3.3. L'attribut offrant le plus grand gain forme donc le nœud courant. Le processus est repris de façon récursive pour chaque branche de l'arbre jusqu'à ce qu'une condition d'arrêt soit rencontrée pour former une feuille. Chaque branche forme une règle de production constituée d'une suite de conditions formées par les nœuds des branches. La première partie de l'Annexe B présente des règles obtenues avec l'algorithme C4.5 sur la base de données de 18 attributs utilisée dans ce travail. Nous pouvons observer que l'attribut M52000 (Montants originaux des prêts hypothécaires) forme le nœud de tête. C'est donc ce dernier qui a obtenu le plus grand gain d'entropie au départ de l'algorithme.

Ces règles de production sont par la suite simplifiées. Le processus de simplification utilisé dans le prototype consiste à réduire le nombre de conditions dans chaque règle ainsi que le nombre de ces règles de production dans l'ensemble. Ces simplifications s'effectuent tout en s'assurant que le taux de réussite de classement des règles de départ ne diminue pas. Il est à noter que la simplification des règles de production permet aussi de réduire le bruit contenu dans les données. En effet, les règles simplifiées généralisent les caractéristiques des attributs des profils de clients. Ainsi, les profils les plus bruités ont une moins grande influence sur la structure des règles de

production utilisées pour la classification. Nous n'expliquons pas davantage le processus suivi par l'algorithme C4.5, car nous l'utilisons exactement tel qu'il a été expliqué dans la section 2.4.3.3.

À l'aide de l'algorithme C4.5, nous obtenons donc un ensemble de règles représentant chacune des catégories de client. Par la suite, chaque cas de la base est classé dans une ou plusieurs règles qu'il satisfait. On dit qu'une règle *couvre* un cas si les contraintes dictées par la règle sont toutes satisfaites par les attributs du cas. Par exemple, un cas dont les quatre attributs v_1, v_2, v_3 et v_4 prennent les valeurs $v_1 = 4$, $v_2 = 10$, $v_3 = 0$ et $v_4 = 1$ serait couvert par les règles numéro 1 et 3 de l'ensemble suivant :

Règle 1 : $v_1 < 7$, $v_2 > 3$ et $v_4 = 1$
Règle 2 : $v_1 < 1$, $v_2 > 2$, $v_3 = 0$ et $v_4 > 0$
Règle 3 : $v_3 < 2$, $v_4 > 0$

La base de données utilisée étant importante, les arbres générés étaient grands : de l'ordre de plusieurs milliers de nœuds et d'une profondeur moyenne de 110 nœuds. Dans cette situation, la simplification de l'arbre via les règles de production était absolument nécessaire. Le Tableau 4.1 donne un exemple de quelques-unes des règles que nous avons obtenues à la fin du processus de construction et de simplification de l'arbre. Le règles simplifiées ont une profondeur moyenne de six nœuds.

Tableau 4.1 : Exemple de règles de production générées par l'algorithme C4.5 avec un ensemble de 6000 profils

Si S300 >= 10651 **et** CAGE < 25 **et** M100 >= 556 **alors** *Catégorie1* (22, 4) CF= 79.55
Si D140 < 1 **et** S300 < 71469 **et** D800 < 1 **et** CAGE >= 55 **et** D300 >= 2 **alors** *Catégorie5* (86, 15) CF= 81.98
Si D800 < 1 **et** D520 < 1 **et** M100 < 349 **et** M100 >= 24 **et** CAGE >= 66 **alors** *Catégorie5* (69, 10) CF=84.78
Si M520 < 35000 **et** D800 < 1 **et** M100 < 25 **et** CAGE < 45 **et** M500 >= 19997 **alors** *Catégorie4* (63, 1) CF=97.62
Si D140 >= 2 **et** M500 < 16083 **et** S140 < 20251 **et** M520 < 33000 **et** D100 < 2 **alors** *Catégorie2* (143, 9) CF=93.36
...
CATÉGORIE PAR DEFAUT : *Catégorie6* (3451, 2033) CF-->41.1

Regardons de plus près la première règle. Elle stipule que si un client a un solde d'au moins 10 651$ dans ses comptes de placements (S300 >= 10651), est âgé de moins de 25 ans (CAGE < 25) et si ses comptes de transactions accusent un solde moyen de 556$ ou plus (M100 >= 556) alors, il a une probabilité de 79.55% d'être placé dans la catégorie 1. Ce pourcentage est appelé facteur de certitude (*Certainty Factor, CF*) [QUI 87b].

Ce facteur de certitude représente un estimé du taux de réussite d'une règle; un facteur de certitude élevé indique que la règle classe correctement un grand nombre des cas qu'elle couvre. Le CF n'est en fait que le ratio du nombre de profils de clients bien classés d'une règle, dont on soustrait 0.5, puis que l'on divise par le nombre total de profils couverts par cette même règle multipliée par 100. La soustraction de 0.5 du nombre de clients bien classés par une règle est nécessaire afin d'obtenir un meilleur estimé du taux de réussite d'une règle. Cette marge d'erreur est en fait une correction de continuité, standard en statistique, qui donne de bien meilleures approximations pour les distributions binomiales [MOO et MCC 93, pp. 384-386]. En effet, l'utilisation de la

formule sans le 0.5 donne un estimé trop optimiste du taux de réussite (surtout quand le nombre de clients est petit), car les règles ont été développées pour représenter un ensemble de données bien précis. Pour avoir un CF réaliste sur le classement de profils inconnus, une marge d'erreur doit donc être ajoutée sous la forme du 0.5. La formule du CF est la suivante :

$$CF_i = \frac{profils\_bien\_classés - 0.5}{total\_profils} * 100$$

Dans le Tableau 4.1, les nombres entre parenthèses qui suivent le nom de la catégorie représentent respectivement, le nombre de profils couverts par cette règle et le nombre de ceux-ci mal classés par la règle. Par exemple, pour la première règle du tableau, des 22 clients sont couverts par la règle, 4 ne sont pas de la catégorie 1 alors qu'ils le devraient selon le classement fourni par la banque. Ainsi, le CF de la première règle est (22 - 4 - 0.5) / 22 * 100 = 79.55.

Pour classer un nouveau profil de client, l'ensemble des règles générées par l'algorithme C4.5 est utilisé. Si une seule règle couvre le profil, celui-ci est classé dans la catégorie suggérée par cette dernière. Si plus d'une règle couvre le profil, deux alternatives sont possibles :

- Toutes les règles dictent une même catégorie alors, le nouveau profil est classé dans celle-ci;

- Nous sommes face à un conflit entre les catégories dictées par les règles. Afin de résoudre ce conflit, nous avons alors recours au processus de raisonnement à base de cas.

4.2.4 Raisonnement à base de cas

Le raisonnement à base de cas permet au système de choisir, parmi les catégories conflictuelles, celle qui doit être attribuée à un nouveau cas. Pour ce faire, la base de cas est fouillée afin de trouver l'ensemble des cas les plus similaires au nouveau cas. Le principe sur lequel la recherche est basée est le suivant : plus les attributs définissant un cas sont similaires aux attributs d'un autre cas, plus la probabilité que ces deux cas soient dans la même catégorie est grande.

En raisonnement à base de cas, l'étape de recherche est d'une importance capitale. En effet, si la recherche des cas similaires est mauvaise, la classification effectuée par le système en est directement affectée et les résultats peuvent être faussés. Pour réaliser une recherche efficace et qui fait ressortir les bons cas au bon moment, des poids doivent être assignés aux attributs du nouveau profil de client à classer. Plus un attribut est représentatif, plus le poids qui lui est donné est grand. La méthode utilisée pour fixer les poids des attributs est une variante d'une méthode appelée *Probability Ranking Principle* (PRP) [COO 73], utilisée au départ dans la recherche de documents. Cette méthode est utilisée dans le système ELEM2-CBR et les auteurs la présente comme suit [CER *et al.* 99] :

> « *If a case retrieval system's response to a new case (query) is a ranking of cases in the case base in order of decreasing probability of query relevance, where the probabilities are estimated as accurately as possible on the basis of whatever data has been made available to the system for the purpose of using the retrieval result to solve the problem represented in the query, then the overall effectiveness of the system in terms of the probability of relevant cases being retrieved will be the best that is obtainable on the basis of that data.*»

Présentons cette méthode de façon plus pratique. Soit un nouveau profil pour un client de la banqueX représenté par un ensemble de 18 paires *attribut-valeur* $\{av_1, av_2, ..., av_{18}\}$. Les poids pour chaque paire attribut-valeur du nouveau cas sont calculés selon la formule suivante [CER *et al.* 99, ROB et SPA 76] :

$$w(av_i) = \log \frac{(r+0.5)(N-n-R+r+0.5)}{(n-r+0.5)(R-r+0.5)}$$

où il y a N cas dans la base dont R cas possèdent des affinités avec le nouveau profil. La paire attribut-valeur av_i apparaît n fois dans l'ensemble N et r fois dans l'ensemble R. Les valeurs de 0.5, ajoutées aux quatre arguments de cette formule, sont nécessaires pour des raisons statistiques qui ne seront pas explicitées ici [COX 70]. De plus, ces valeurs évitent d'obtenir des divisions par zéro ou un log(0) qui donneraient des erreurs de calcul.

Les variables N et n sont facilement déterminées en examinant la base de cas du prototype. Les variables R et r, de leur côté, sont déterminées à partir de l'ensemble de règles conflictuelles décrites à la section précédente. En effet, lorsqu'il y a conflit entre plusieurs catégories, le profil est probablement situé à la frontière des régions définies par ces catégories. Ainsi, les cas couverts par toutes ces règles sont considérés comme ayant des affinités avec le nouveau profil du client. Le nombre de cas dans l'ensemble des règles conflictuelles détermine la variable R et ainsi r devient facilement calculable. Il est à noter que cette formule a été développée pour être utilisée avec des attributs ne prenant que des valeurs qualitatives, ce qui n'est pas notre cas. Nous avons donc établi des intervalles pour chacun des attributs afin de « discrétiser » les données. Ces intervalles sont calculés en additionnant et soustrayant 10% de la valeur de l'attribut. Ce principe provient du domaine de la logique et des ensembles flous [KLI et YUA 95, LIA *et al.* 98]. La Figure 4.4 présente cette transformation selon les principes des ensembles flous. Par exemple, un attribut a_1 ayant une valeur de 50 (paire attribut-valeur av_1) est transformé en un intervalle couvrant les valeurs entre 45 et 55 inclusivement. Ainsi, tous les cas dont la valeur pour l'attribut a_1 est entre 45 et 55 sont considérés

comme étant égaux à la paire attribut-valeur av_1. En d'autres mots, peu importe la probabilité d'appartenance (*membership grade*) de la valeur de l'attribut, dès que cette valeur est entre 45 et 55, ce cas est considéré comme possédant des affinitées avec le cas ayant une caleur de 50.

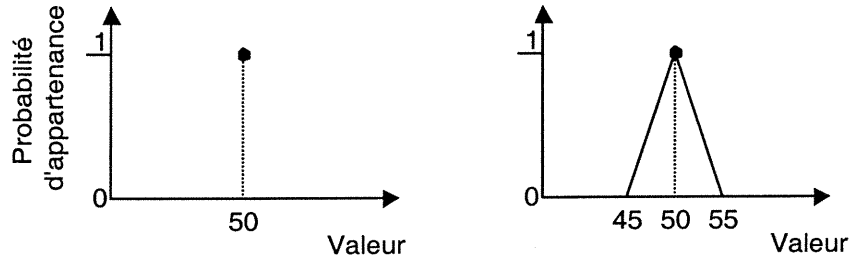


Figure 4.4 : Transformation d'un nombre selon les principes de la logique et des ensembles flous

Décrivons le calcul des poids pour le profil suivant :

Nouveau client #183 :

Âge :45 (av_1)

Nbre. de comptes de transactions : 2 (av_2) *Solde de ces comptes : 2356 (av_3)*

Nbre. de comptes REER : 0 (av_4) *Solde de ces comptes : 0 (av_5)*

Nbre. de comptes placements : 0 (av_6) *Solde de ces comptes : 0 (av_7)*

Nbre. de prêts personnels 1 (av_8) *Montant original de ces prêts : 25000 (av_9)*

Nbre. de marges de crédit : 0 (av_{10}) *Limite totale de ces marges : 0 (av_{11})*

Nbre. de prêts hypothécaires :1 (av_{12}) *Montant original de ces prêts : 98533 (av_{13})*

Nbre. de cartes de crédit or : 0 (av_{14}) *Limite totale des cartes : 0 (av_{15})*

Nbre. de cartes de crédit régulières :1 (av_{16}) *Limite totale des cartes : 4100 (av_{17})*

Nbre. de comptes carte-client : 1 (av_{18})

Vu la grande quantité de calculs à effectuer, nous ne les démontrons que pour la paire av_1.

La variable N, représentant le nombre de cas dans la base de cas, est la même pour toutes les paires attribut-valeur du profil du nouveau client. Fixons cette valeur à $N = 1000$. Parmi ces 1000 cas, nous devons trouver combien ont leur attribut d'âge (av_1) dans l'intervalle [40.5, 49.5]. Disons, par exemple, que nous obtenons pour cette paire $n = 243$. Les variables R et r sont déterminées comme nous le disions par l'ensemble des cas des règles conflictuelles. Supposons que nous avons un conflit, car le profil du nouveau client #183 est couvert par deux règles : la première suggère la catégorie 2 alors que la seconde suggère la catégorie 1. R représente le nombre de cas de la base qui satisfont ces deux règles et r est déterminé selon le même principe que n. Disons que 500 cas de la base de 1000 cas satisfont les deux règles et que parmi celles-ci 123 cas ($r = 123$) avaient une valeur d'âge entre 40.5 et 49.5. Ainsi la formule du calcul pour le poids du premier attribut du profil se présente comme suit:

$$w(av_1) = \log \frac{(123-0.5)*(1000-243-500+123-0.5)}{(243-123-0.5)*(500-123-0.5)} = 0.014$$

La formule générale calculant le poids est ainsi appliquée à chaque paire attribut-valeur du profil.

Les poids que nous venons d'attribuer sont maintenant utilisés pour mesurer la similarité entre les cas de la base et le profil du nouveau client. La formule mesurant cette similarité nous permet d'ordonner de façon décroissante les cas de la base selon leur similitude avec le nouveau profil. C'est en fait l'application du Probability Ranking Principle (PRP). La formule de similarité est la suivante :

$$Similarité(x,q) = \sum_{i=1}^{n} w_i \times Simil(x_i, q_i)$$

où

$$Simil(x_i, q_i) = \begin{cases} 0 & \text{si } x_i \neq q_i \text{ et que la valeur de l'attribut est qualitative;} \\ 1 & \text{si } x_i = q_i \text{ et que la valeur de l'attribut est qualitative;} \\ 1 - |norm(x_i) - norm(q_i)| & \text{si la valeur de l'attribut est quantitative.} \end{cases}$$

et où x représente un cas de la base de cas, q le nouveau profil du client et w les poids. L'indice i représente la position de l'attribut considéré.

Il est à noter que les valeurs des attributs sont normalisées (*norm*) afin d'éliminer le biais causé par les différents ordres de grandeur entre celles-ci. La fonction de normalisation que nous avons utilisée est la suivante :

$$norm(x_i) = x_i / x_i\_max$$

La variable x_i_max représente la valeur maximale prise par l'attribut i du cas x. Le même principe s'applique évidemment pour normaliser q_i. Cette fonction renvoie toutes les valeurs des attributs dans l'intervalle [0, 1].

Une fois le coefficient de similarité calculé pour tous les cas de la base, ceux-ci sont ordonnés de façon décroissante selon ce coefficient. Un sous-ensemble S des k cas les plus similaires est sélectionné et analysé. La valeur de k est déterminée par l'usager. Si les k cas du sous-ensemble S suggèrent tous la même catégorie au nouveau profil de client, il est classé dans celle-ci. En revanche, si plusieurs catégories sont suggérées par l'ensemble S, nous devons calculer le *coefficient de décision* (ou *Decision Score*). Ce coefficient est calculé à partir de la sommation des coefficients de similarité pour chaque catégorie C_i. La formule suivante est utilisée pour effectuer ce calcul :

$$DS(C_i) = \sum_{j=1}^{m} Similarité(c_j, q)$$

où c_j est le j ième cas de la catégorie C_i et q le nouveau cas. Ainsi, le nouveau profil est finalement classé dans la catégorie ayant le plus grand coefficient de décision.

Grâce au classement effectué par le système hybride décrit ci-dessus, le département de marketing de la banqueX pourrait parvenir à définir le type de produits et de services bancaires qui seraient susceptibles d'intéresser ce client. Par exemple, sachant que le profil bancaire d'un client appartient à la catégorie 1, la banqueX pourrait lui offrir des prêts personnels car ce sont ces derniers que ce type de clients utilise ou recherche.

4.3 Évaluation du prototype

Le prototype que nous avons développé est évalué selon deux points de vue. Ces évaluations sont de type numériques. Elles reflètent la qualité de nos résultats sous la forme de pourcentages de réussite et de données statistiques. Le prototype pourrait également être évalué par des experts en marketing de la banqueX. Par contre, ce type d'évaluation étant très exigeant en terme de temps de travail et d'accessibilité des experts, nous n'avons pas été en mesure d'évaluer le prototype sous cet angle.

Comme nous l'avons illustré dans la Figure 4.1, représentant les phases de développement du prototype, ce dernier est évalué en deux phases : évaluation du data mining (Phase 3) et évaluation du prototype (Phase 5).

Voici les tests effectués :

- Évaluer la qualité des règles générées par la technique de data mining.

- Analyser les résultats de la classification en faisant varier k, le nombre de cas conservés dans le sous-ensemble S (sous-ensemble contenant les k cas les plus similaires).

- Évaluer la qualité de la classification du système dans sa globalité.

De plus, précisons que le prototype a été implémenté avec le langage de programmation Java de Sun, utilisant l'ensemble d'outils de développement jdk1.3.0. Le langage Java a été sélectionné pour plusieurs raisons. Tout d'abord, ce langage est reconnu pour sa portabilité d'un système d'exploitation à l'autre et permet donc au programme d'être facilement utilisé dans différents milieux. De plus, le Java est un langage orienté-objet, ce qui est avantageux du point de vue de la conception, de l'implémentation et de la maintenance du système. Tous nos programmes (algorithmes et tests) ont été exécutés sur un ordinateur Pentium III, 750Mhz avec 512Mb de RAM utilisant le système d'exploitation Windows NT 4.0 sp6 de Microsoft.

4.3.1 Procédure

Pour chacun des tests effectués sur le prototype, une validation croisée (*cross-validation*) est effectuée sur l'ensemble des données de la banqueX. La validation croisée consiste à répéter 10 fois chaque test avec 10 échantillons de données différents. Cette validation nous permet d'affirmer que les résultats obtenus ne sont pas simplement le fruit du hasard quant au choix des données de départ.

4.3.1.1 Data mining

Pour évaluer la qualité des règles générées par la technique de data mining, nous avons suivi la procédure habituellement utilisée par Quinlan pour évaluer les arbres de décision générés par l'algorithme C4.5.

Dans [QUI 87b], Quinlan utilise deux ensembles de données : un ensemble d'apprentissage et un ensemble de test. De plus, il fixe respectivement un ratio de 2/3-1/3 sur la quantité de données contenues dans ces deux ensembles. L'ensemble utilisé pour construire un arbre et simplifier les règles contient les deux tiers du total des données. Le tiers restant est divisé en deux sous-ensembles de cardinalité égale pour

constituer deux sous-ensembles de tests. Cette division est effectuée afin de s'assurer que les proportions réelles des catégories de profils de clients soient conservées. Nous avons donc sélectionné 10 ensembles d'échantillons aléatoires tirés parmi les 11 073 profils de clients fournis par la banqueX. Chacun des 10 ensembles d'échantillons contient 8000 profils et est constitué comme suit : 6000 profils formant l'ensemble de *training* (*apprentissage*) et 2000 profils, divisés en deux groupes égaux, constituant les deux groupes de tests des règles générées par l'arbre.

Pour chaque échantillon, nous avons exécuté l'algorithme C4.5[8] en lui fournissant l'ensemble des 6000 profils d'apprentissage. La section 2.4.3.3 du chapitre sur le data mining décrit en détail la construction et la simplification exécutée par l'algorithme C4.5. Par la suite, c'est avec les deux groupes de tests que nous avons réellement mesuré l'efficacité de l'ensemble de règles simplifiées (appelé ensemble de règles final) sur de nouveaux profils.

Les résultats obtenus sont présentés dans la section 4.3.2.

4.3.1.2 Analyse de la variable k

Ce test a pour but de déterminer la valeur optimale de la variable *k*. Cette variable est utilisée lorsqu'il y a un conflit entre les différents cas les plus similaires au nouveau profil d'un client. Rappelons que ce type de conflit survient si le classement suggéré au nouveau profil n'est pas unanime entre les *k* cas les plus similaires retrouvés dans la base (sous-ensemble *S*). Ce conflit est réglé en calculant la somme des coefficients de similarité de chaque catégorie présente dans *S*.

Le résultat de la sommation des coefficients de similarité dépend directement du nombre *k* de cas conservés dans l'ensemble *S* et influence donc directement la classification attribuée au nouveau profil de client. Illustrons cette influence par un

[8] L'algorithme C4.5 avec la formule du *gainRatio* et le *test de Fisher* à 0.1%.

exemple où la variable $k = 6$. Les six cas ayant obtenus les plus grands coefficients de similarité et qui donc se retrouvent dans S sont les suivants :

Cas1 : Suggère la *Catégorie 1* avec une mesure de similarité = 9.6527
Cas2 : Suggère la *Catégorie 5* avec une mesure de similarité = 9.1584
Cas3 : Suggère la *Catégorie 5* avec une mesure de similarité = 8.6465
Cas4 : Suggère la *Catégorie 3* avec une mesure de similarité = 8.2458
Cas5 : Suggère la *Catégorie 3* avec une mesure de similarité = 8.1140
Cas6 : Suggère la *Catégorie 3* avec une mesure de similarité = 7.2674

Les six cas n'étant pas tous dans la même catégorie, le coefficient de décision doit être calculé pour chaque catégorie (dans cet exemple, les Catégories 1, 3 et 5). Rappelons que le coefficient de décision (Decision Score, DS) représente la sommation des coefficients de similarité par catégorie et permet de déterminer le classement final. Les coefficients de décision obtenus sont les suivants :

DS(*Catégorie 1*) = 9.6527
DS(*Catégorie 5*) = 9.1584 + 8.6465 = 17.8049
DS(*Catégorie 3*) = 8.2458 + 8.1140 + 7.2674 = 23.6272

Le nouveau cas est classé dans la catégorie dont le DS est le plus élevé. Dans cette situation, avec $k = 6$, le nouveau profil serait donc classé dans la Catégorie 3. En revanche, si nous fixons $k = 4$:

Cas1 : Suggère la *Catégorie 1* avec une mesure de similarité = 9.6527
Cas2 : Suggère la *Catégorie 5* avec une mesure de similarité = 9.1584
Cas3 : Suggère la *Catégorie 5* avec une mesure de similarité = 8.6465
Cas4 : Suggère la *Catégorie 3* avec une mesure de similarité = 8.2458

donc,

DS(*Catégorie 1*) = 9.6527

DS(*Catégorie 5*) = 9.1584 + 8.6465 = 17.8049

DS(*Catégorie 3*) = 8.2458

Cette fois, c'est la Catégorie 5 qui aurait été choisie. Puis finalement, si $k = 1$, c'est bien évidemment la Catégorie 1 qui aurait été suggérée. Ainsi, l'influence de la valeur de la variable k n'est pas à négliger.

C'est pour cette raison qu'une analyse des résultats de la classification effectuée par le prototype a été menée en fixant la variable k à différentes valeurs. De cette façon, nous avons trouvé la valeur optimale de k pour les profils de la banqueX.

4.3.1.3 Data mining et raisonnement à base de cas

Nous évaluons maintenant la qualité des résultats du prototype dans son entier, data mining et raisonnement à base de cas utilisés conjointement. Pour ce faire, nous avons besoin de fournir au prototype un fichier contenant des nouveaux cas pour tester l'efficacité de la classification ainsi qu'une base de cas et un ensemble de règles.

La base de cas ainsi que l'ensemble de règles utilisés sont différents pour chacune des 10 exécutions du prototype. Ces changements nous permettent de nous assurer de l'efficacité du processus de classification. En ce qui concerne l'échantillon des nouveaux profils à classer, ce dernier demeure le même pour chacun des 10 tests. La base de cas est formée d'un ensemble de 6000 profils de la banqueX encore une fois tirés de façon aléatoire. Chacune des 10 exécutions du prototype est testée selon les 10 bases de cas et ensembles de règles utilisés lors de la phase d'évaluation du processus de data mining.

Voici maintenant les résultats obtenus pour chacun de ces trois tests.

4.3.2 Résultats

Les premiers résultats présentés sont ceux portant sur la partie concernant le data mining. Le Tableau 4.2 illustre le nombre de règles de production générées par l'arbre de décision de l'algorithme C4.5 [QUI 93] avant et après la simplification de l'ensemble des règles. Rappelons que cette simplification est nécessaire afin de rendre les résultats plus simples et plus compréhensibles.

Tableau 4.2 : Nombre de règles avant et après la simplification

	Ensemble de règles original	Ensemble de règles final
Échantillon 1	1682	51
Échantillon 2	1641	43
Échantillon 3	1652	62
Échantillon 4	1633	58
Échantillon 5	1659	48
Échantillon 6	1627	39
Échantillon 7	1631	49
Échantillon 8	1659	48
Échantillon 9	1618	49
Échantillon 10	1639	49

Il est évident, d'après les résultats présentés dans le tableau précédent que la simplification des règles était nécessaire. Des règles contenant plus de 1000 conditions sont tout à fait incompréhensibles. Ainsi, l'algorithme de simplification du C4.5 réduit significativement l'ensemble de règles généré par l'arbre de décision. Ce taux de réduction est normal puisque Quinlan en présente des semblables dans [QUI 87b]. Ainsi, les résultats ressortant de cette étape sont satisfaisants.

Comme il est bien plus simple et bien plus efficace d'analyser les règles de l'ensemble final, c'est ce dernier qui est utilisé tout au long de ce travail. Un exemple

d'une partie d'un ensemble de règles de production que nous avons obtenu et simplifié avec l'algorithme C4.5 est présenté dans l'Annexe B.

Par contre, il faut s'assurer que la qualité des règles de l'ensemble final est au moins aussi bonne que celle de l'ensemble original. En d'autres mots, il faut s'assurer que le Facteur de Certitude (CF) moyen de l'ensemble réduit de règles, c'est-à-dire que la mesure de la qualité des règles, est plus grand ou égal à celui de l'ensemble de règles original. Si ce n'est pas le cas, cela signifie que certaines informations importantes représentant les profils ont été perdues.

Nous retrouvons dans le Tableau 4.3 les pourcentages de réussite de classement des profils de clients (le CF) pour l'ensemble original et pour l'ensemble final lors de l'étape de test des règles (deux tests sur des sous-ensembles de 1000 profils de clients).

Tableau 4.3 : Pourcentages de réussite de l'algorithme C4.5 pour le classement de profils de clients

	Ensemble original (% réussite)	Sous-ensemble final Test1 (% réussite)	Sous-ensemble final Test2 (% réussite)
Échantillon 1	55.87	70.65	73.41
Échantillon 2	56.17	70.44	75.27
Échantillon 3	55.58	70.82	67.31
Échantillon 4	56.03	72.14	70.10
Échantillon 5	54.37	66.94	72.10
Échantillon 6	55.80	74.53	75.97
Échantillon 7	55.42	68.33	69.65
Échantillon 8	55.46	73.90	71.23
Échantillon 9	55.86	73.71	75.08
Échantillon 10	54.63	69.62	67.81
Moyenne	55.52	71.11	71.80

Comme ce tableau le démontre, nous obtenons une moyenne de réussite de classement de 55.52% (100% – 55.52% = 44.5% d'erreurs de classification) lorsque l'ensemble de règles générées par l'arbre C4.5 n'est pas simplifié. Le taux de réussite de l'ensemble de règles final augmente à 71.45% (100% – 71.45% = 28.5% d'erreurs). Sur certaines base de données, Quinlan a déjà obtenu des pourcentages semblables [QUI 87b]. Ce taux de 71.45% est donc satisfaisant.

L'augmentation d'environ 15% du nombre de bons classements illustrée ci-dessus est attribuable à la généralisation des règles. L'explication de ce phénomène de « généralisation » provient du domaine de l'apprentissage machine. Dans ce domaine, la variation du taux de réussite en classification est attribuable à « l'overfitting » des données sur les règles de production de l'arbre de décision. En d'autres mots, les arbres très complexes ont tendance à représenter les données d'apprentissage de façon trop précise. Quinlan donne une excellente explication de ce phénomène dans [QUI 93] . Il explique que l'effet de « l'overfitting » :

> « ... *is readily seen in the extreme example of random data in which the class of each case is quite unrelated to its attribute values. I [Quinlan] constructed an artificial dataset of this kind with ten attributes, each of which took the value 0 or 1 with equal probability. The class was also binary,* yes *with probability 0.25 and* no *with probability 0.75. One thousand randomly generated cases were split into a training set of 500 and a test set of 500. From this data, C4.5's initial tree-building routine produces a nonsensical tree of 119 nodes that has an error rate of more than 35% on the test cases.*
>
> *This small example illustrates the twin perils that can come from too gullible acceptance of the initial tree: It is often extremely complex, and can actually have a higher error rate than a simpler tree. For the random data above, a tree consisting of just the leaf* no *would have an expected error rate of 25% on unseen cases, yet the elaborate tree is noticeably less accurate.* »

Cet exemple illustre très bien le désavantage de développer un arbre qui tente de représenter trop précisément les données. Ainsi, ce sont les ensembles de règles finaux que nous utiliserons pour effectuer la suite de l'évaluation du prototype.

Lorsqu'il y a conflit entre plusieurs règles, l'algorithme d'arbres de décision classe simplement le nouveau profil du client dans la règle possédant le plus grand CF. L'intégration du processus de RBC à l'arbre de décision, permet de gérer les situations conflictuelles à l'aide de cas plutôt que de règles. Le conflit est réglé en retrouvant les k cas les plus similaires au nouveau profil. Comme nous le mentionnions précédemment, la variable k joue un grand rôle afin de déterminer le classement du nouveau profil dans le processus de RBC. Le Tableau 4.4 contient le nombre d'erreurs de classement en fonction de différentes valeurs de k. Cette série de tests de classement a été effectuée sur un échantillon de 1095 nouveaux profils.

Tableau 4.4 : Nombre d'erreurs de classement du prototype en fonction de k (nombre de cas les plus similaires au nouveau profil)

	k = 1	k = 3	k = 5	k = 10	k = 20	k = 50
Ens. de règles 1	256	301	319	327	334	350
Ens. de règles 2	248	300	315	325	335	352
Ens. de règles 3	245	297	315	321	327	338
Ens. de règles 4	249	302	320	329	338	350
Ens. de règles 5	253	301	321	330	340	357
Ens. de règles 6	261	310	326	330	344	353
Ens. de règles 7	253	295	318	324	330	343
Ens. de règles 8	258	304	323	326	332	348
Ens. de règles 9	346	301	312	319	324	347
Ens. de règles 10	264	309	325	335	343	359
Moyenne	253.3	302.0	319.4	326.6	334.7	349.7

La Figure 4.5 représente graphiquement la courbe du nombre d'erreurs de classement en fonction de la valeur de *k*. Elle démontre que plus *k* est petit, plus le nombre d'erreurs chute. Ainsi, dans le prototype, la variable *k* est optimale lorsqu'elle est égale à 1.

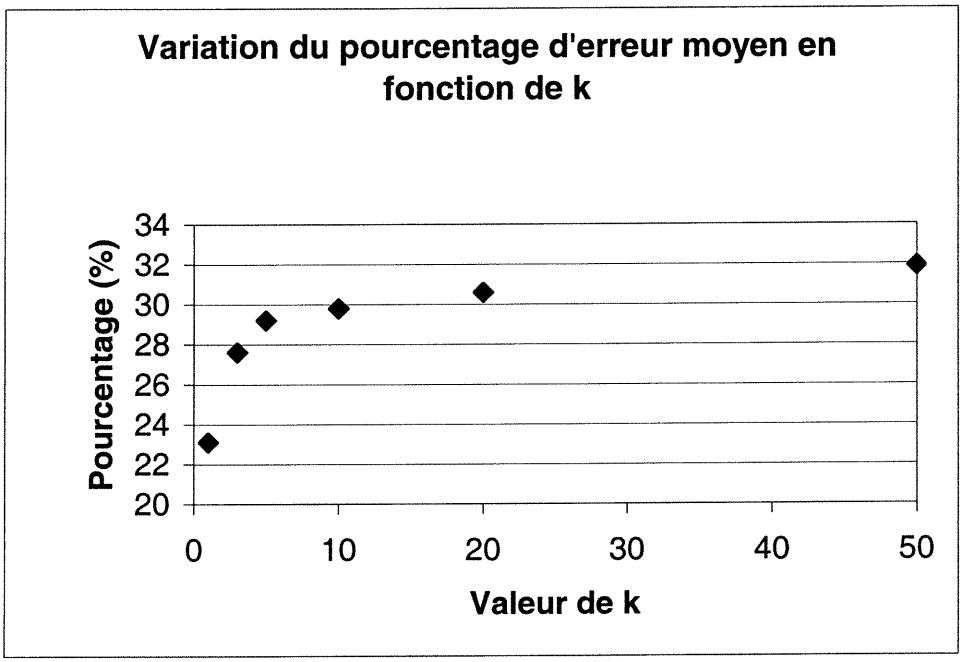


Figure 4.5 : Variation du pourcentage d'erreur moyen en fonction de la variable *k*

Le Tableau 4.5 présente le pourcentage de réussite de classification du prototype lorsque la variable *k* est optimale ($k = 1$). C'est-à-dire que le processus de RBC attribue au nouveau profil de client le même classement que celui dicté par le cas lui étant le plus similaire. Comme nous l'avons précisé auparavant, pour chacun de ces tests l'ensemble de règles ainsi que la base de cas utilisés varient. Par contre, l'ensemble des nouveaux profils de clients à classer reste toujours fixe.

Tableau 4.5 : Pourcentage de réussite du prototype ($k = 1$)

	Pourcentage de réussite (%)
Test 1	79.91
Test 2	76.53
Test 3	77.08
Test 4	77.08
Test 5	78.08
Test 6	78.26
Test 7	79.27
Test 8	77.08
Test 9	77.99
Test 10	78.99
Moyenne	78.03

Le système arrive donc à classer correctement 78% des nouveaux profils de clients. Il est normal qu'un système de ce type n'arrive pas à classer parfaitement tous les profils, mais une partie du pourcentage d'erreur que nous effectuons peut être justifiée par les données elles-mêmes.

En effet, l'échantillon d'environ 11 000 profils fourni par la banqueX était préclassé selon les méthodes statistiques employées par les experts de la banque. Chaque profil de client est donc relié à une des 6 catégories prédéfinies par la banque ce qui nous permet d'évaluer l'efficacité du prototype en le comparant au classement de la banque. Par contre, les experts de la banqueX ne classent pas les profils de clients de façon individuelle, mais bien par *ménages*. Un ménage est formé par l'ensemble des personnes résidant à la même adresse civique. Prenons par exemple, les trois clients suivants faisant partie de la même famille et résidant à la même adresse :

	Client1	**Client2**	**Client3**
Lien de parenté	Père	Mère	Adolescent
Âge	46	44	16
Attributs bancaires :			
• Nb. de comptes de transactions et *solde ($)*	1, *5 000$*	1, *4 000$*	1, *200$*
• Nb. d'hypothèques et *montant original ($)*	1, *80 000$*	0, *0$*	0, *0$*
• Nb. de comptes REER et *solde ($)*	2, *35 000$*	3, *50 000$*	0, *0$*
Classement de la banqueX	**Catégorie 3**		

La banqueX a classé ce ménage dans la catégorie 3 car le père de famille possède un prêt hypothécaire et que le ménage dispose de plusieurs comptes REER dont le solde s'élève à plus de 60 000$.

En revanche, ces trois individus sont considérés comme étant indépendants dans le prototype car la banqueX ne nous a pas fourni l'information nécessaire pour reconstituer les ménages. Observons donc ces clients sous cet angle. Le client1 (le père) devrait logiquement être classé dans la catégorie 3 comme la banqueX nous le suggère car ses attributs bancaires sont caractéristiques de cette catégorie. Le client2 (la mère) aurait aussi de grandes chances d'être correctement classé vu le solde de ses REER. Le client3 (l'adolescent) nous cause par contre beaucoup plus de problèmes. Il n'a en sa possession seulement qu'un compte de transactions dont le solde est inférieur à 500$. Les caractéristiques des clients de la catégorie 3, comme nous venons de le mentionner, démontrent que cette catégorie de clients possède des REER et/ou des hypothèques, ce qui n'est pas le cas pour le client3. Ainsi, le prototype a tendance à le classer dans la catégorie 5 qui est composée des clients possédant très peu de produits bancaires et dont les soldes sont peu élevés. Le prototype présente donc des erreurs de classement en raison de ce biais induit par le regroupement des clients en ménages. Ce genre d'erreur pourrait être corrigé soit en révisant les profils avec les experts de la banqueX soit en obtenant directement le classement par ménages de la banqueX. Malheureusement, pour des raisons de disponibilité des experts et de confidentialité, la classification par ménages nous est inaccessible.

Finalement, le Tableau 4.6 démontre que l'approche hybride, combinant les arbres de décision et le RBC, adopté dans le prototype surpasse en efficacité ces deux techniques employées de façon individuelle. Les données du tableau représentent les pourcentages de réussite de classement du même ensemble de test selon ces trois méthodes.

Tableau 4.6 : Comparaison des résultats obtenus selon la méthode hybride, les arbres de décision et le RBC

	Méthode hybride	Arbres de décision	Raisonnement à base de cas
Test 1	79.91	72.03	77.90
Test 2	76.53	72.86	73.52
Test 3	77.08	69.07	71.96
Test 4	77.08	71.12	73.61
Test 5	78.08	69.52	76.07
Test 6	78.26	75.25	73.06
Test 7	79.27	68.99	76.62
Test 8	77.08	72.57	72.15
Test 9	77.99	74.40	71.42
Test 10	78.99	68.72	75.89
Moyenne (M)	78.03	71.45	74.22

La Figure 4.6 fait ressortir plus clairement les résultats présentés dans le tableau précédent. Dans cet histogramme, il apparaît évident que la méthode hybride surpasse en efficacité de classement les deux autres méthodes.

Nous constatons que l'approche hybride obtient un pourcentage de réussite plus élevé d'environ 4% en comparaison avec le raisonnement à base de cas et de 6% avec les arbres de décision.

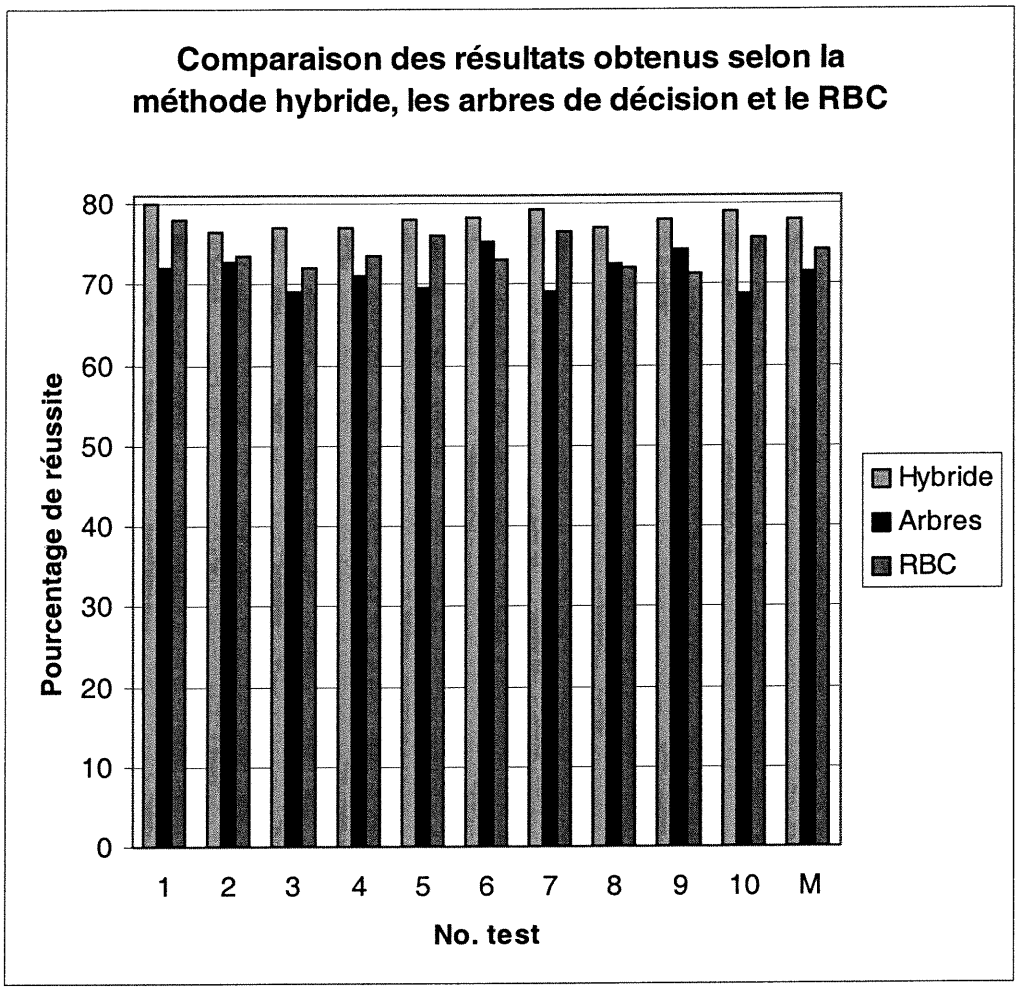


Figure 4.6 : Histogramme du pourcentage de réussite de la méthode hybride, des arbres de décision et du RBC

Outre l'amélioration des résultats obtenus pour la classification de nouveaux profils de clients avec la méthode hybride, il est à noter que la performance du prototype au point de vue du temps de calcul est aussi très intéressante. En effet, l'approche hybride est près de 2.3 fois plus rapide que le RBC utilisé seul. Ce gain de rapidité est attribuable au court temps d'exécution des arbres de décision (environ 7 fois plus rapides que le RBC) comme l'illustre la Figure 4.7.

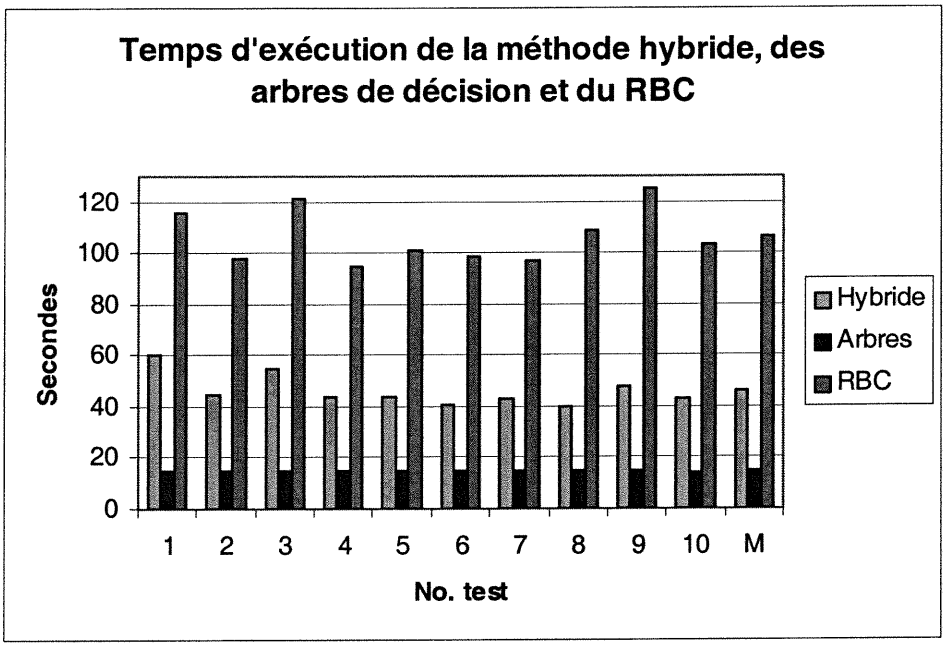


Figure 4.7 : Histogramme du temps d'exécution de la méthode hybride, des arbres de décision et du RBC

Ainsi, le prototype hybride combine les qualités des méthodes de raisonnement à base de cas et des arbres de décision. En effet, le processus de raisonnement à base de cas complète les arbres de décision en augmentant le taux de réussite de classement tandis que les arbres de décision réduisent considérablement le temps de calcul nécessaire à la classification de nouveaux profils de clients. Le prototype hybride développé dans ce travail se présente donc non seulement comme une alternative intéressante aux systèmes statistiques employés à la banqueX, mais aussi comme une combinaison avantageuse et efficace des techniques de RBC et des arbres de décision.

4.4 Améliorations possibles

Les améliorations dont pourrait bénéficier le prototype développé dans ce travail peuvent être divisées en deux catégories soient : au niveau de l'implémentation de celui-ci ainsi qu'au niveau des données à traiter.

Un objectif à long terme du prototype serait d'être inclus comme partie intégrante du système de gestion des profils des clients de la banqueX. En effet, dès l'inscription d'un nouveau client ou dès la modification de son profil, le système pourrait mettre presque instantanément à jour la catégorie qui lui est associée. De plus, due à la très grande masse de données que la banque possède sur ses clients, l'utilisation de fichiers séquentiels, comme nous le faisons actuellement, serait impossible pour des raisons de performances. Une connectivité directe aux bases de données devrait alors être établie, par exemple avec JDBC. Autre avantage indéniable pour les employés de la banqueX, un tel système intégré leur donnerait une vue globale de la situation de leurs clients et ce en temps réel, ce qui permettrait au service de marketing de préparer plus rapidement des campagnes de publicité ciblée.

Une grande masse de données à traiter entraîne inévitablement la question de la performance du prototype. Une optimisation des algorithmes utilisés devrait être effectuée, notamment au niveau de la construction de l'arbre de décision et de la simplification de l'ensemble de règles qu'il génère, soient les étapes les plus exigeantes en temps de calcul. Aussi, pour un déploiement à grande échelle du prototype, le choix du langage de programmation pourrait être révisé en faveur d'un langage plus performant que Java quant à la rapidité d'exécution, par exemple le C++.

Au niveau des données traitées par le prototype, plusieurs autres améliorations pourraient être apportées. Premièrement, il faudrait obtenir de la banque les profils des clients regroupés en ménages. Ce regroupement des profils de clients ne nous a pas été fourni par la banqueX pour des raisons de respect de la vie privée des clients, ce regroupement étant effectué selon leurs adresses. Ceci nous permettrait d'effectuer un meilleur classement des nouveaux profils des clients et ainsi d'augmenter le taux de réussite du prototype comparativement au classement obtenu par les techniques statistiques de la banque. De plus, les besoins de la banque pourraient être mieux servis en tenant compte de certains attributs déjà contenus dans le profil des clients, mais qui ont été laissés pour contre dans le prototype. En effet, le prototype a été implémenté

seulement avec les attributs les plus représentatifs et les plus généraux pour des raisons de temps de calcul.

Certains de ces attributs concernent des données de type transactionnel, par exemple le nombre de dépôts qu'un client a effectués à un guichet automatique. Présentement, la banqueX n'utilise pas ces données mais étudie la possibilité de les analyser à l'aide de techniques statistiques. Il est à noter que le prototype serait en mesure d'intégrer ce type d'attributs sans grandes modifications.

Un autre type de données qui pourrait être utilisé en conjonction avec le prototype, est celui regroupant les attributs de type démographique (code postal, sexe, état civil, salaire, etc.). L'ensemble de ces attributs pourrait également permettre à la banque de prédire le moment propice où ses clients auront le plus besoin de produits et de services financiers précis. Par exemple, l'observation mensuelle ou hebdomadaire des soldes de comptes REER de clients pourrait donner de bons indices sur le moment propice pour lancer une campagne sur ce produit. Le même genre d'observation dans le temps, mais effectué sur l'état civil des clients, pourrait révéler à quel moment proposer des prêts personnels (parfois nécessaires afin de célébrer un mariage par exemple!). Pour arriver à obtenir ce genre d'information, ces attributs combinés à un nouvel algorithme de data mining permettraient d'analyser les tendances de données changeantes dans le temps (Trend analysis of time-series [HAN et KAN 01]). Un nouvel algorithme serait en effet nécessaire, car celui utilisé présentement dans le prototype n'effectue pas ce genre de tâche. Ce dernier ajout comblerait la troisième partie de la question que nous avions éludée dans l'introduction :

Qui a besoin de quoi et quand ?

4.5 Conclusion

Ce dernier chapitre a présenté une évaluation en plusieurs points du prototype de classification de profils de clients développé dans ce travail. Nous avons évalué tour à tour la qualité de l'ensemble de règles utilisé pour effectuer une première classification, celle de l'impact du choix de la valeur de la variable k contenue dans le processus de RBC ainsi que la qualité globale du prototype hybride.

Nous avons également démontré que dans notre situation, l'utilisation d'un système hybride combinant le raisonnement à base de cas et l'induction de règles à l'aide d'arbres de décision (algorithme C4.5), était plus performant qu'un système n'implémentant qu'une seule de ces techniques.

Les résultats obtenus avec le prototype pourraient cependant être améliorés en intégrant la notion de ménages permettant de regrouper les clients venant d'une même famille. Nous avons finalement mentionné quelques extensions qui permettraient au prototype de s'intégrer au système de la banqueX.

Conclusion

What we call the beginning is often the end
And to make an end is to make a beginning.
The end is where we start from...
- T.S. ELIOT, « Four Quartets »

De nos jours, la majorité des entreprises utilise des plans de marketing. Les entreprises emmagasinent des données sur leurs clients, les analysent et peuvent ainsi déterminer quel type de clientèle cibler dans leurs campagnes publicitaires pour chacun de leurs produits et services. Habituellement, ces analyses sont menées par des statisticiens chevronnés qui développent des modèles mathématiques pour chacun des profils de leur clientèle. Ainsi, les équipes d'experts en marketing doivent exposer leurs idées et leurs besoins à des équipes de statisticiens, ce qui peut entraîner certains désagréments.

L'objectif de notre travail était donc de développer et implémenter un système permettant aux employés d'une banque n'ayant pas nécessairement de base en statistique de générer eux-mêmes des listes de clients à cibler dans le but d'effectuer des campagnes publicitaires efficaces. Dans ce travail, nous avons suggéré un prototype

combinant une technique de data mining et le processus de raisonnement à base de cas. Ces deux méthodes combinées offrent une approche simple et intuitive qui peut être une alternative ou un complément aux techniques statistiques actuellement employées dans le domaine du marketing bancaire.

La technique de data mining utilisée était celle des arbres de décision. Elle a permis de classer dans différentes catégories des profils bancaires de clients selon certaines connaissances générales. Chaque catégorie de client a été définie par un ensemble de règles qui décrivent les caractéristiques propres à chacune. Pour déterminer la catégorie du profil d'un client, une première classification était tentée avec ces règles. Si la classification était impossible de cette façon, le processus de raisonnement à base de cas prenait en charge le profil et le classait selon son degré de similarité avec différents profils de clients que le système avait en mémoire. Cette approche hybride est semblable à celle du système ELEM2-CBR. En fait, nous avons effectué deux changements sur l'approche de ELEM2-CBR. Premièrement, nous avons utilisé une méthode de génération de règles plus connue et éprouvée que celle utilisée dans ELEM2-CBR. Deuxièmement, la technique de discrétisation des attributs numériques est basée sur les principes de la logique et des ensembles flous plutôt que sur des calculs d'entropie. Cette modification simplifie ce processus tout en conservant sont efficacité.

L'originalité et la contribution apportées par notre travail reposent sur deux points. Premièrement, le RBC n'a pas encore été largement appliqué dans le domaine du marketing, ainsi nous ouvrirons peut-être cette avenue avec un petit nombre d'autres chercheurs. Cette avenue est intéressante pour les entreprises qui utilisent des techniques statistiques afin de produire leurs campagnes publicitaires et qui recherchent des méthodes plus intuitives pour arriver à leurs fins. En effet, l'utilisation d'un système combinant le data mining et le raisonnement à base de cas éviterait les problèmes de compréhension entre les statisticiens et les experts en marketing. Il en résulterait une amélioration de l'efficacité du marketing des produits d'une entreprise.

Deuxièmement, ce travail a aussi attaqué le problème de l'utilisation de grandes bases de cas dans des systèmes de RBC. Les chercheurs dans ce domaine sont conscients que la recherche des cas les plus similaires dans de larges bases est plutôt périlleuse et qu'elle peut devenir très complexe. Dans notre situation, ce problème se devait d'être résolu car la base de cas utilisée est d'une taille relativement grande : échantillon de 6 000 observations (parmi la base de profils en contenant environ 11 000) possédant chacune 18 attributs numériques multivalués. Le processus de data mining a été utilisé pour effectuer la tâche de recherche de façon plus efficace. Notre prototype hybride arrive à classer des profils de clients jusqu'à quatre fois plus rapidement que le raisonnement à base de cas seul.

Le chapitre portant sur le prototype, démontre que l'approche hybride utilisant les arbres de décision et le raisonnement à base de cas est une alternative intéressante pour effectuer le classement des profils de clients de la banqueX. Le taux de réussite moyen du classement des profils de clients du prototype est de 78%. Ce taux de réussite est acceptable vu le biais induit par le regroupement des clients en ménages qui n'a pas été pris en compte dans les données. Rappelons que nous désirons déterminer la catégorie du profil d'un client car c'est cette dernière qui nous permet de déterminer quels produits et services pourraient lui être recommandés. Ainsi, selon notre postulat de base, les experts de la banque devraient étudier les résultats obtenus par le prototype. Cette analyse pourrait leur permettre de dénicher de nouvelles informations concernant le comportement financier de leurs clients et éventuellement d'améliorer leur processus de classification.

Certaines améliorations devraient être effectuées afin d'augmenter le rendement du prototype. L'utilisation des profils de clients regroupés en ménages permettrait d'améliorer le taux de réussite du classement des profils du prototype. Nous avons également vu que certaines améliorations permettraient d'intégrer une dimension temporelle aux campagnes de marketing qui utiliseraient les classements des profils de clients obtenus par le système développé dans ce travail.

De plus, afin de permettre au prototype d'être utilisé en milieu bancaire, quelques extensions devraient y être apportées. Par exemple, l'optimisation du temps de calcul et de la gestion de la mémoire devrait être envisagée pour permettre au système d'être utilisé à l'échelle des bases de données de la banqueX. La connectivité avec ces bases via un système de gestion de données devrait aussi être développée. Finalement, le prototype devrait idéalement être évalué par les experts en marketing et en statistique de la banque afin d'en certifier la validité et l'efficacité.

Somme toute, le processus de classification suggéré par l'approche hybride de ce travail présente des résultats intéressants. Les avantages de la combinaison des connaissances spécifiques du RBC et des connaissances générales du DM devraient être étudiés de façon plus détaillée. Nous espérons que ce travail entraînera différentes réflexions sur ce sujet.

Annexe A

Le tableau suivant contient la liste des 18 attributs utilisés pour décrire le profil d'un client et leurs significations respectives. Les étiquettes des attributs sont celles utilisées par la banqueX. Ces attributs sont de type quantitatif continu et multivarié. Les données sont réelles et nous ont été fournies par une grande banque canadienne. De plus, chaque profil de client est préclassé dans des catégories de 1 à 6.

CLVAGE : Âge du client
D10002 : Nombre de comptes de transactions M10002 : Solde moyen des comptes de transactions
D14000 : Nombre total de comptes REER M14000 : Solde des comptes REER
D30002 : Nombre total de comptes de placements S30002 : Solde des comptes de placements
D50000 : Nombre de prêts personnels M50000 : Montants originaux des prêts personnels
D51000 : Nombre. de marges de crédit M51000 : Limite de crédit des marges de crédit
D52000 : Nombre de prêts hypothécaires M52000 : Montants originaux des prêts hypothécaires
D54000 : Nombre de cartes de crédit or M54000 : Limite de crédit des cartes de crédit or
D55000 : Nombre de cartes de crédit régulière M55000 : Limite de crédit des cartes de crédit régulière
D80000 : Nombre de comptes carte-client

Annexe B

Ci-dessous sont présentés deux sous-ensembles de cinq règles. Ils ont été générés et simplifiés par l'algorithme d'arbres de décision nommé C4.5. Le premier sous-ensemble présente des règles générées directement par l'arbre. Afin de faciliter l'observation de ces règles, les parties en gras sont celles retrouvées dans les règles simplifiées. Le deuxième sous-ensemble présente ces mêmes règles mais simplifiées.

Règles de départ :

1. [M52000 (<68500.0), S30002 (<75065.0), M10002 (<99986.0), D14000 (<2.0), D52000 (<2.0), S30002 (<74489.0), M50000 (<90000.0), M54000 (<15000.0), CLVAGE (<65.0), M51000 (<60000.0), S14000 (<62319.0), S14000 (<53037.0), D14000 (<1.0), S30002 (<71469.0), M52000 (<35000.0), M10002 (<80760.0), S30002 (>=8071.0), M50000 (<10002.0), D10002 (<4.0), M51000 (<25000.0), M51000 (<18000.0), M54000 (<6000.0), D80000 (<1.0), M10002 (<11744.0), D51000 (<1.0), S30002 (>=10117.0), S30002 (>=10545.0), **S30002 (>=10651.0)**, CLVAGE (>=10.0), CLVAGE (>=13.0), **CLVAGE (<25.0)**, **M10002 (>=556.0)**] : Catégorie1

2. [M52000 (<68500.0), S30002 (<75065.0), M10002 (<99986.0), D14000 (<2.0), D52000 (<2.0), S30002 (<74489.0), M50000 (<90000.0), M54000 (<15000.0), CLVAGE (<65.0), M51000 (<60000.0), S14000 (<62319.0), S14000 (<53037.0), **D14000 (<1.0)**, S30002 (<71469.0), M52000 (<35000.0), M10002 (<80760.0), S30002 (<8071.0), D50000 (<1.0), M10002 (<52596.0), S30002 (<7224.0), D10002 (<1.0), M52000 (<20000.0), D54000 (<1.0), S30002 (>=1040.0), **D80000 (<1.0)**, **S30002 (<6612.0)**, **S30002 (>=3376.0)**, CLVAGE (>=62.0)] : Catégorie5

3. [M52000 (<68500.0), S30002 (<75065.0), M10002 (<99986.0), D14000 (<2.0), D52000 (<2.0), S30002 (<74489.0), M50000 (<90000.0), M54000 (<15000.0), CLVAGE (<65.0), M51000 (<60000.0), S14000 (<62319.0), S14000 (<53037.0), D14000 (<1.0), S30002 (<71469.0), M52000 (<35000.0), M10002 (<80760.0), S30002 (<8071.0), D50000 (<1.0), M10002 (<52596.0), S30002 (<7224.0), D10002 (>=1.0), M54000 (<8000.0), D80000 (<1.0), M54000 (<6000.0), **M52000 (<30002.0)**, M10002 (>=-20.0), D54000 (<1.0), S30002 (<5062.0), S30002 (<4882.0), M10002 (>=-3.0), CLVAGE (>=13.0), **D30002 (<1.0)**, M10002 (<27957.0), M55000

(<4600.0), M55000 (<3800.0), CLVAGE (<36.0), **M55000 (>=2000.0)**, **CLVAGE (<34.0)**] : Catégorie4

4. [M52000 (<68500.0), S30002 (<75065.0), M10002 (<99986.0), D14000 (>=2.0), M50000 (<31318.0), D52000 (<2.0), **M54000 (>=6000.0)**, S14000 (<47213.0), D10002 (<4.0), S14000 (<32856.0), **M54000 (<10002.0)**, **D51000 (>=1.0)**] : Catégorie3

5. [M52000 (<68500.0), S30002 (<75065.0), M10002 (<99986.0), D14000 (<2.0), D52000 (<2.0), S30002 (<74489.0), M50000 (<90000.0), M54000 (<15000.0), CLVAGE (<65.0), M51000 (<60000.0), S14000 (<62319.0), S14000 (<53037.0), D14000 (<1.0), S30002 (<71469.0), M52000 (<35000.0), M10002 (<80760.0), S30002 (>=8071.0), M50000 (<10002.0), D10002 (<4.0), M51000 (<25000.0), M51000 (<18000.0), M54000 (<6000.0), D80000 (<1.0), M10002 (<11744.0), D51000 (<1.0), S30002 (>=10117.0), S30002 (>=10545.0), S30002 (>=10651.0), CLVAGE (>=10.0), CLVAGE (>=13.0), CLVAGE (>=25.0), CLVAGE (>=38.0), **CLVAGE (<55.0)**, CLVAGE (>=49.0), **S30002 (>=21417.0)**, **S30002 (<24092.0)**] : Catégorie2

Règles simplifiées :

1. [S30002 (>=10651.0), CLVAGE (<25.0), M10002 (>=556.0)] : Catégorie1

2. [D14000 (<1.0), D80000 (<1.0), S30002 (<6612.0), S30002 (>=3376.0)] : Catégorie5

3. [M52000 (<30002.0), D30002 (<1.0), M55000 (>=2000.0), CLVAGE (<34.0)] : Catégorie4

4. [M54000 (>=6000.0), M54000 (<10002.0), D51000 (>=1.0)] : Catégorie3

5. [CLVAGE (<55.0), S30002 (>=21417.0), S30002 (<24092.0)] : Catégorie2

Bibliographie

1. [AAM et PLA 94] Aamodt A. et Plaza E., *Case-based Reasoning : Foundational Issues, Methodological Variations, and System Approaches.* Artificial Intelligence Communications, IOS Press, Vol. 7, no. 1, 1994. pp. 39-59.

2. [AN *et al.* 95] An A., Cercone N., Chan C., Shan N. et Huang X., *ELEM : A Method for Inducing Rules from Examples.* Proc. 15th Ann. Conf. British Computer Soc. Specialist Group on Expert Systems, UK, 1995. pp. 85-99.

3. [AN 97] An A., *Analysis Methodologies for Integrated and Enhanced Problem Solving.* Thèse de Doctorat, Department of Computer Science, Université de Regina, Canada, 1997.

4. [AN et CER 98] An A. et Cercone N., *ELEM2 : A Learning System for More Accurate Classifications.* Proc. 12th Biennial Conf. Canadian Soc. Computational Studies of Intelligence, AI 1998, Springer Verlag, Vancouver, Canada, 1998. pp. 426-441.

5. [BAR *et al.* 88] Bareiss E.R., Porter B.W. et Weir C.C., *Protos : An Exemplar-Based Learning Apprentice.* International Journal of Man-Machine Studies, Vol. 29, no. 5, 1988. pp. 549-561.

6. [BAR 89] Bareiss E.R., *Exemplar-Based Knowledge Acquisition : A Unified Approach to Concept Representation, Classification, and Learning.* Academic Press Inc., 1989.

7. [BEN *et al.* 01] Bengio Y., Lauzon V.-P. et Ducharme R., *Experiments on the Application of IOHMMs to Model Financial Returns Series.* IEEE Trans. on Neural Networks, Vol. 12, no. 1, 2001. pp. 113-123.

8. [BER *et al.* 99] Berson A., Smith S. et Thearling K., *Building Data Mining Applications for CRM*. McGraw-Hill, 1999.

9. [BIS 95] Bishop C.M., *Neural Networks for Pattern Recognition*. Oxford University Press, Oxford, UK, 1995.

10. [BOL *et al.* 00] Bollacker K. D., Lawrence S. et Glies C. L., *Discovering Relevant Scientific Literature on the Web*. IEEE Intelligent Systems, Vol.15, no. 2, Mars/Avril, 2000. pp. 42-47.

11. [BRE *et al.* 84] Breiman L., Friedman J. H., Olshen R. et Stone C., *Classification and Regression Trees*. Wadsworth International Group, Monterey, CA, 1984.

12. [BRI 96] Brightware Inc., *ARTScript Programming Guide 3, Rules & CBR*. 1996.

13. [CAB *et al.* 97] Cabena P., Hadjinian P., Stadler R., Verhees J. et Zanasi A., *Discovering Data Mining, From Concept to Implementation*. Prentice-Hall Inc., Upper Saddle River, NJ, 1997.

14. [CER *et al.* 99] Cercone N., An A. et Chan C., *Rule-Induction and Case-Based Reasoning : Hybrid Architectures Appear Advantageous*. IEEE Trans. On Knowledge And Data Engineering, Vol. 11, no. 1, 1999. pp. 166-173.

15. [COO 73] Cooper W.S., *On Selecting a Measure of Retrieval Effectiveness*. Journal of Am. Soc. Information Science, Vol. 24, 1973. pp. 413-424.

16. [COX 70] Cox D.R., *Analysis of Binary Data*. Methuen, London, 1970.

17. [DUB *et al.* 97] Dubois D., Esteva F., Garcia P, Godo L., Mantaras R.L. et Prade H., *Fuzzy Modeling of Case-Based Reasoning and Decision*. Proc. ICCBR 1997, Leake D.B. et Plaza E. éditeurs, Sprigner-Verlag, Berlin, 1997. pp. 599-610.

18. [FAG et BLO 99] Fagan M. et Bloor K., *Case-Based Reasoning for Candidate List Extraction in a Marketing Domain.* Proc. ICCBR 1999, Althoff K.-D., Bergmann R. et Branting L.K. éditeurs, Sprigner-Verlag, Berlin, 1999. pp. 426-437.

19. [FAY 96] Fayyad U. M., *Data Mining and Knowledge Discovery : Making Sense Out of Data.* IEEE Intelligent Systems, Vol. 11, no. 5, Octobre, 1996. pp. 20-25.

20. [FIN *et al.* 63] Finney D.J., Latscha R., Bennett B. M. et Hsu P., *Tables for Testing Significance in a 2x2 Contingency Table.* Cambridge University Press, Cambridge, 1963.

21. [FRI 97] Friedman J.H., *Data Mining and Statistics : What's the Connection?.* Proc. of the 29th Symp. Interface, Scott D.W. éditeur, Houston, 1997. pp. 3-9.

22. [GOL 89] Goldberg D.E., *Genetic Algorithms in Search, Optimisation, and Machine Learning.* Addison-Wesley, MA, 1989.

23. [GRO 99] Groth R., *Data Mining : Building Competitive Advantage.* Prentice-Hall PTR, Upper Saddle River, NJ, 1999.

24. [HAN 01] Hansen B.K., *Weather Prediction Using Case-Based Reasoning and Fuzzy Set Theory.* Mémoire de maîtrise, Technical University of Nova Scotia, Halifax, Canada, 2000.

25. [HAN et KAM 01] Han J. et Kamber M., *Data Mining : Concepts and Techniques.* Morgan Kaufmann Publishers, 2001.

26. [HAND 98] Hand D.J., *Data Mining : Statistics and More?.* The American Statistician, Vol. 52, no. 2, Mai, 1998. pp. 112-118.

27. [HAR 75] Hartigan J.A., *Clustering Algorithms.* John Wiley & Sons, NY, 1975.

28. [HEC 96] Heckerman, D., *Bayesian networks for knowledge discovery*. Advances in Knowledge discovery and Data mining, Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P. et Uthurusamy, R. éditeurs, MIT Press, MA, 1996, pp. 273-305.

29. [HED 96] Hedberg S. R., *Searching for the mother lode : tales of the first data miners*. IEEE Intelligent Systems,Vol. 11, no. 5, Octobre, 1996. pp. 4-7.

30. [HER *et al.* 91] Hertz J., Koperski K. et Palmer R.G., *Introduction to the Theory of Neural Computation*. Addison-Wesley, MA, 1991.

31. [KAS 99] Kasif S., *Datascope : Mining Biological Sequences*. IEEE Intelligent Systems, Vol. 14, no. 6, Nov./Dec., 1999. pp. 38-43.

32. [KEL *et al.* 85] Keller J.M., Gray M.R. et Givens J.A., *A Fuzzy K-Nearest Neighbor Algorithm*. IEEE Trans. on systems, man, and cybernetics, Vol. 15, no. 4, Juillet/Août, 1985. pp. 580-585.

33. [KLA *et al.* 95] Klaus-Dieter A., Auriol E., Bergmann R., Breen S., Dittrich S., Johnston R., Manago M., Traphoener R. et Wess S., *Case-Based Reasoning for Decision Support and Diagnostic Problem Solving : The INERCA Approach*. Proc. 3rd workshop German special interest group on CBR, Bartsch-Sporl D., Janetzko D. et Wess S. éditeurs, 1995. pp. 63-72.

34. [KLI et YUA 95] Klir J.G. et Yuan B., *Fuzzy sets and fuzzy logic : Theory and Applications*. Prentice-Hall PTR, Upper Saddle River, NJ, 1995.

35. [KOL 93] Kolodner J., *Case-Based Reasoning*. Morgan Kaufmann Publishers, 1993.

36. [KOL et LEA 96] Kolodner J. et Leake D.B., *A Tutorial Introduction to Case-Based Reasoning*. Case-Based Reasoning: Experiences, Lessons and Future Directions, Leake D.B. éditeur, MIT Press, 1996. pp. 31-65.

37. [KOT 89] Koton P., *Using Experience in Learning and Problem Solving*. Thèse de Doctorat, Rapport technique no. MIT/LCS/TR-441, Laboratory of Computer Science, Massachusetts Inst. Of Technology, 1989.

38. [LEA 96] Leake D.B., *CBR in Context : The Present and Future*. Case-Based Reasoning: Experiences, Lessons and Future Directions, Leake D.B. éditeur, MIT Press, 1996. pp. 3-30.

39. [LIA *et al.* 98] Liao T.W., Mount C.R. et Zhang Z., *Similarity measures for retrieval in case-based reasoning systems*. Applied Artificial Intelligence, Vol. 12, 1998. pp. 267-288.

40. [LUG et STU 97] Luger G. F. et Stubblefield W. A., *Artificial Intelligence : Structures and Strategies for Complex Problem Solving*. 3e édition, Addison-Wesley, 1997.

41. [MAG94] Magidson J., *The CHAID approach to segmentation modeling : CHI-squared automatic interaction detection*. Advanced Methods of Marketing Research, Begozzi R.P. éditeur, Blackwell Business, MA, 1994. pp. 118-159.

42. [MIC *et al.* 83] Michalski R.S, Carbonell J.G. et Mitchell T.M., *Machine Learning : An Artificial Intelligence Approach*. Tioga, Vol. 1, Palo Alto, CA, 1983.

43. [MIC *et al.* 86]. Michalski R.S, Carbonell J.G. et Mitchell T.M., *Machine Learning : An Artificial Intelligence Approach*. Morgan Kaufmann, Vol. 2, Los Altos, CA, 1986.

44. [MIC *et al.* 98] Michalski R.S., Brakto I et Kubat M., *Machine Learning and Data Mining : Methods and Applications*. John Wiley & Sons, NY, 1998.

45. [MOO et MCC 93] Moore D.S. et McCabe G.P., *Introduction to the Practice of Statistics*. W.H. Freeman and company, NY, 1993.

46. [NOC *et al.* 00] Nock R., Sebban M. et Jappy P., *A Symmetric Nearest Neighbour Learning Rule*. Proc. EWCBR 2000, Blanzieri R. et Portinale L. éditeurs, Sprigner-Verlag, Berlin, 2000. pp. 222-233.

47. [POR *et al.* 90] Porter B.W., Bareiss E.R. et Holte R., *Concept Learning and Heuristic Classification in Weak-Theory Domains*. Artificial Intelligence, vol. 45, 1990. pp. 229-263.

48. [QUI 83] Quinlan J.R., *Learning Efficient Classification Procedures and their Application to Chess End Games*. Dans [MIC *et al.* 83], 1983. pp. 463-482.

49. [QUI 86] Quinlan J.R., *Induction of Decision Trees*. Machine Learning, no. 1, 1986. pp. 81-106.

50. [QUI *et al.* 87] Quinlan J.R., Compton P.J., Horn K.A. et Lazarus L., *Inductive Knowledge Acquisition : A Case Study*. Applications of Expert Systems, J.R. Quinlan éditeur, Wokingham, Addison-Wesley, UK, 1987. pp. 157-173.

51. [QUI 87a] Quinlan J.R., *Generating Production Rules From Decision Trees*. Proceedings of the Tenth International Joint Conference on Artificial Intelligence, McDermott J. éditeur, Morgan Kaufmann, San Mateo, CA, 1987. pp. 304-307.

52. [QUI 87b] Quinlan J.R., *Simplifying decision trees*, International Journal of Man-Machine Studies, Vol. 27, no.3, 1987. pp. 221-234.

53. [QUI 93] Quinlan J.R., *C4.5 : Programs for Machine Learning*. Morgan Kaufmann Publishers, 1993.

54. [REF 95] Refenes A., *Neural Networks in the Capital Markets*. John Wiley & Sons Inc., NY, 1995.

55. [ROB et SPA 76] Robertson S.E. et Sparck Jones K., *Relevance Weighting of Search Terms*. Journal of the American Society for Information Science, Vol. 27, Mai/Juin, 1976. pp. 129-146.

56. [SAL 99] Salzberg S. L., *Gene Discovery in DNA Sequences*. IEEE Intelligent Systems, Vol. 14, no. 6, Novembre/Décembre, 1999. pp. 44-48.

57. [SCH et ABE 77] Schank R. et Abelson R., *Scripts, plans, goals and understanding*. Erlbaum, Northvale, NJ, 1977.

58. [SCH 82] Schank R., *Dynamic memory : A theory of learning in computers and people*. Cambridge University Press, NY, 1982.

59. [SHA 48] Shannon C.E., *A mathematical theory of communication*. Bell System Technical Journal, vol. 27, 1948. pp. 379-423 et pp. 623-656.

60. [SKA 93] Skalak, D.B., *Using a Genetic Algorithm to Learn Prototypes for Case Retrieval and Classification*. Proc. AAAI 1993, Case-Based Reasoning Workshop, Technical Report WS-93-01, AAAI Press, Washington, DC, 1993. pp. 211-215.

61. [SKA et RIS 90] Skalak D.B. et Rissland E.L., *Inductive Learning in a Mixed Paradigm Setting*. Proc. AAAI 1990, AAAI Press, Boston, MA, 1990. pp. 840-847.

62. [STE 90] Stevens R., *UNIX Network Programming*. Prentice-Hall, 1990.

63. [WAT 97] Watson I. D., *Applying Case-Based Reasoning : Techniques for Enterprise Systems*. Morgan Kaufmann Publishers, 1997.

64. [ZAD 65] Zadeh L.A., *Fuzzy Sets*. Information and Control, Vol. 8, 1965. pp. 338-353.

URLs :

65. [URL 1] Online Machine Learning Resources : *http://www.ai.univie.ac.at/oefai/ml/ml-resources.html*

66. [URL 2] Machine Learning & CBR Resources : *http://www.aic.nrl.navy.mil/~aha*

67. [URL 3] The Data Mine : *http://www.cs.bham.ac.uk/~anp/TheDataMine.html*

68. [URL 4] Knowledge Discovery Mine : *http://www.kdnuggets.com*

69. [URL 5] Office de la Langue Française : *http://www.olf.gouv.qc.ca*

70. [URL 6] SPSS (CHAID) : *http://www.spss.com*

71. [URL 7] Salford Systems (CART) : *http://www.salford-systems.com*

72. [URL 8] Brightware : *http://www.brightware.com*

73. [URL 9] Inference Corp. (CBR3) : *http://www.inference.com/*

74. [URL 10] Isoft (ReCall) : *http://www.alice-soft.com/html/prod_recall.htm*

75. [URL 11] SAS Institute : *http://www.sas.com*