LISTE DES TABLEAUX

LISTE DES FIGURES

CHAPITRE 1

INTRODUCTION

1.1 Contexte de recherche

De nos jours, l'informatique est de plus en plus présente dans notre monde. La majorité de la population possède maintenant un ordinateur, que ce soit un ordinateur de bureau, un portable, un téléphone intelligent, etc. De plus, avec l'avènement d'Internet depuis les années 2000 et toute la mobilité des systèmes, les gens n'ont jamais été autant connectés et l'information n'a jamais été aussi disponible. Selon Statistique Canada, en 2010, 8 ménages canadiens sur 10 avaient accès à Internet. Plus de la moitié de ces ménages avaient une connexion qui utilisait plus d'un appareil pour accéder à Internet [5]. Ces chiffres sont éloquents et représentent bien la situation dans laquelle nous nous retrouvons. Ces connexions multiples permettent de recueillir de l'information intéressante sur les actions posées par les usagers. Par exemple, une technique de plus en plus utilisée sur Internet est le marketing ciblé [6], qui permet d'envoyer des publicités précises aux usagers selon les sites visités ou les articles achetés précédemment. Pour ce faire, un modèle statistique tente d'établir une concordance avec les données de l'utilisateur et le programme aura pour tâche de prédire avec une certaine probabilité, à quel type de

consommateur appartient un client précis [7]. Ces informations sont devenues un atout incontournable et on commence à peine à se rendre compte de leur importance [8].

Beaucoup d'informations de ce genre sont recueillies, et ce, dans plusieurs domaines pour former des bases de données gigantesques. Cependant, en raison de la quantité de valeurs pouvant être emmagasinée dans un entrepôt de données, il s'avère difficile de manipuler l'information dans son ensemble. Ces données peuvent dissimuler des connaissances de haut niveau très avantageuses pour les compagnies ou à des fins de recherche. Prenons l'exemple du centre de données de climat national. Ce centre regroupe près de 6 pétaoctets de données. (Un pétaoctet correspond à 1 000 000 000 000 000 ou 10^{15} octets). Le centre numéro un de données pense augmenter leur quantité à plus de 15 Pétaoctets d'ici 2020 [9], ce qui constitue une base de données incroyable. Mais comment utiliser une telle quantité d'informations pour faire ressortir de nouvelles connaissances ?

Il existe certaines méthodes statistiques intéressantes pour tenter d'extraire de nouvelles informations à partir de données à l'état brut. Par exemple, deux méthodes sont très utilisées dans le monde statistique, soit l'analyse discriminante et la régression linéaire multiple [10]. En premier lieu, l'analyse discriminante est en majeure partie une projection graphique pour classifier un individu ou un élément. Elle vise deux buts particuliers : le premier étant la description, à savoir à l'intérieur de groupes connus, quelles sont les principales différences que l'on peut déterminer à l'aide des variables mesurées. Le deuxième but, le classement, tend à déterminer le groupe d'appartenance d'une nouvelle

observation uniquement à partir des variables mesurées [3]. Cette méthode fonctionne très bien avec les valeurs numériques, mais présente certaines limitations quant aux résultats avec plusieurs attributs testés simultanément. La deuxième méthode, la régression multiple, est utilisée pour faire de la prédiction de données. En écrivant le problème sous forme de combinaison linéaire multiple en fonction d'un certain nombre de variables prédictives, le tout créé un hyperplan à *m* dimensions [11]. En fait, il faut résoudre un système d'équations en minimisant l'erreur quadratique. Ces deux méthodes statistiques donnent de bons résultats, mais ils ont des limitations dès qu'on dépasse une certaine dimension en termes de nombre de variables et présentent des limites en termes d'efficacité.

Puisque ces méthodes statistiques ne suffisaient pas, il fallait développer une manière plus efficace d'analyser ces nouvelles mines d'information. De nouvelles méthodes algorithmiques ont donc fait leur apparition pour ce qui est de l'analyse des bases de données pour régler cette problématique.

1.2 Le forage de données

Depuis quelques années, cette problématique a donné lieu à un essor sans précédent d'un créneau de recherche en intelligence artificielle, baptisé forage de données ou *data mining* [12], qui propose une approche algorithmique afin d' « analyser le passé pour prédire l'avenir » [13]. En résumé, c'est l'art d'extraire des informations ou même de nouvelles connaissances, grâce aux données. Le *data mining* est soit descriptif ou prédictif.

Les techniques prédictives, ou d'exploration visent à mettre en évidence des informations présentes, mais cachées par le volume des données. Quant aux techniques descriptives, aussi appelées techniques explicatives, elles visent à extrapoler de nouvelles informations à partir des informations déjà disponibles. Ces nouvelles informations peuvent être qualitatives, par exemple un classement, ou quantitatives comme une prédiction [14].

Le data mining se trouve à la frontière entre l'intelligence artificielle, les statistiques, et la théorie liée aux bases de données, comme le présente la figure suivante.

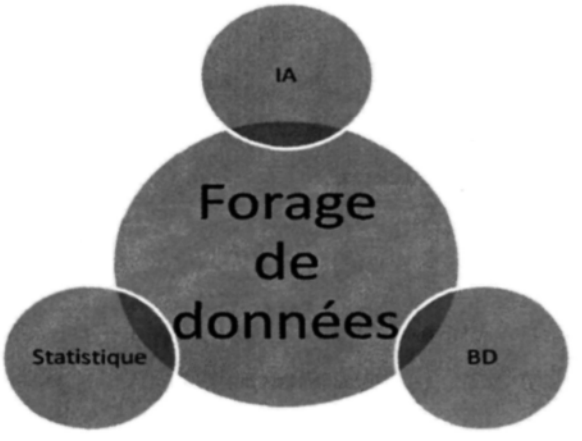


Figure 1 : Origine du data mining

Le problème est qu'à l'état brut, l'immense quantité de données receuillie est, pour la plupart, inutile. C'est pourquoi, à l'aide de l'intelligence artificielle ainsi que de statistique, de nouveaux modèles ou nouvelles connaissances peuvent être extraits. Autant dans le domaine du marketing que dans le domaine médical [15] ou même l'astronomie [2],

de nouvelles découvertes et modèles de décisions ont pu être révélé et utilisé de façon surprenante. Cependant, malgré l'étendue de possibilités que nous offre la fouille de données, le processus dans son ensemble demande une démarche rigoureuse. Il ne s'agit pas du problème majeur : il subsiste un problème dans le choix des algorithmes de data mining [6]. Aussi, il existe plusieurs types d'algorithmes, ce qui rend le choix encore plus important. Par exemple, dans un domaine tel que le marketing, avoir les renseignements sur les types d'acheteurs et les habitudes de ceux-ci donnerait aux décideurs des renseignements d'une valeur non négligeable.

Le contexte de cette recherche est justement en lien avec le marketing. En collaboration avec une compagnie de jeux vidéo, l'objectif du projet est d'arriver à cibler les meilleurs algorithmes permettant de définir un modèle d'acheteur potentiel, de les adapter au contexte particulier du jeu vidéo sur mobile, de les implémenter et d'effectuer une expérimentation complète afin de démontrer l'intérêt et le potentiel de l'exploitation de ce type de système dans le domaine commercial. Ce faisant, des choix sur la stratégie de ventes pourraient être envisageables et le modèle de décision pourrait s'avérer portable sur différents jeux.

Cette recherche a été effectuée en collaboration avec une entreprise de jeux vidéo. La compagnie en question, BlooBuzz, s'est implantée au Saguenay depuis peu et commence à commercialiser plusieurs jeux portables, c'est-à-dire pour les tablettes et les téléphones, et elle veut connaître les éléments des jeux qui pourraient influencer l'achat par

Rapport-Gratuit.com

un joueur. Pour ce faire, les programmeurs récupèrent des données de jeu : le temps d'une session de jeu, les niveaux complétés, le nombre de fois qu'un joueur a fait une action propre au jeu, l'information sur l'achat, etc. Le problème est que dans cet état, les données ne peuvent procurer de nouvelles connaissances. Il s'agit d'une quantité astronomique d'informations (près de 30 millions de données) entassées dans une base de données qui est inutilisable. Le travail consiste donc à adapter plusieurs algorithmes de data mining afin de pouvoir tester leur efficacité et tenter de produire des modèles pouvant être utilisés pour la prise de décision lors du marketing et de la vente d'un produit.

1.3 Résumé de la revue littéraire

Ces dernières années, beaucoup de travaux [16-18] ont été effectués en intelligence artificielle et plus particulièrement, sur les techniques d'apprentissage issues du data mining. Les domaines de recherche sont nombreux et variés. Par exemple, l'université du Québec à Chicoutimi possède un laboratoire de recherche pour les personnes atteintes d'Alzheimer. Leur but est de développer un environnement facilitant le maintien à domicile en aidant les personnes atteintes de la maladie dans leurs tâches quotidiennes. Cependant, le tout s'avère difficile puisque l'évaluation et la prédiction du comportement humain sont des tâches à la fois complexes et ardues [19]. Une recherche a été menée dernièrement pour effectuer des avancées dans la reconnaissance d'activités avec une extension du *flocking*, algorithme normalement utilisé en intelligence artificielle pour le mouvement, mais pouvant servir à la segmentation de données [20].

L'utilisation du *data mining* dans les applications du monde réel est en soi un défi, puisque chaque situation est particulière et que la collecte de données ainsi que l'application des algorithmes est difficile [21]. Pour démontrer que la fouille de données peut s'appliquer malgré cette problématique, des chercheurs ont développé un programme d'aide pour les choix de cours au collégial. En utilisant une version améliorée de l'algorithme Apriori [22], ils ont été en mesure de créer un programme d'aide à la décision pour les choix de cours. En utilisant une liste de choix de cours prédéfinie, le programme créé un cheminement académique particulier pour l'élève. Dans un autre exemple, les scientifiques à l'intérieur d'un vaisseau spatial recueillent des échantillons de données lors de leurs voyages. Certaines méthodes de *data mining* sont déjà utilisées comme les machines à vecteurs de support pour la classification des images de spectres ou pour le déclenchement automatique d'événements lors de certaines observations [5]. Les chercheurs tentent de rendre plus performante leur façon de procéder en utilisant d'autres algorithmes comme K-means pour améliorer l'évaluation de la saisie de données pour des zones radioactives.

Il existe d'autres domaines d'application variés utilisés en *data mining*. Par exemple, en météorologie, on se base sur les connaissances de phénomènes naturels pour prédire le temps qu'il fera [23]. Ou encore, dans le domaine de l'astronomie, on utilise les propriétés des étoiles afin de constituer des groupes qui se ressemblent pour les reconnaître plus facilement [24]. Un logiciel, le Skyline, a été créé pour faciliter la tâche des astronomes. Ou encore, une étude a été faite sur les habitudes de jeu du célèbre World of

Rapport-gratuit.com
LE NUMERO 1 MONDIAL DU MÉMOIRES

Warcraft. Un web service a été créé pour évaluer le rendement personnel d'un joueur comparativement à celui du reste des joueurs [10].

Dans ces recherches, il est question des deux types d'algorithmes en *data mining*, soit la segmentation ainsi que la classification. La segmentation est une technique permettant de créer des regroupements de données qui se ressemblent dans le groupe d'appartenance et qui se dissocient des autres regroupements [7]. Cette technique est souvent utilisée comme méthode d'exploration ou comme prétraitement des données. Il existe plusieurs algorithmes de segmentation, ou *clustering.* Les algorithmes de partitionnement, la segmentation hiérarchique, le partitionnement basé sur la densité ou par grille sont tous des types d'algorithmes pouvant être utilisés pour faire de la segmentation. Les algorithmes de partitionnement divisent les groupes en sous-ensembles et évaluent ensuite ces groupes selon certains critères. Le partitionnement hiérarchique, quant à lui, décompose les objets selon certains critères en une hiérarchie structurée. L'avantage de la méthode hiérarchique comparativement à celle du partitionnement est qu'il ne faut pas indiquer le nombre de regroupements que l'on veut obtenir par l'algorithme, valeur souvent représentée par la lettre K. Cependant, la plupart des méthodes de partitionnement, comme k-means ou EM, sont plus rapides en temps d'exécution [3]. L'autre type, les algorithmes de classification servent à regrouper des individus ayant des propriétés communes. Le but des classes est de trouver des dénominateurs communs dans les individus pour permettre de trouver à quelle classe appartient un nouvel individu arrivant [9]. La différence avec la segmentation est que la classe est déjà connue. Il existe de nombreux algorithmes de

classification. Le plus connu est le C4.5 [25, 26]. Il s'agit d'un algorithme qui donne un arbre de décision comme modèle final. Cet algorithme est facile de compréhension et donne d'excellents résultats, surtout avec des attributs de type nominal. Il n'est pas le seul, les algorithmes CART [27] et CHAID [28] sont aussi deux algorithmes qui produisent des arbres comme modèle résultant. D'autres types de classification existent, comme les réseaux de neurones, l'approche des plus proches voisins ou encore les algorithmes génétiques.

Malgré l'efficacité démontrée des algorithmes de data mining dans plusieurs secteurs [16, 29, 30] très peu d'équipes de recherche ont tenté d'adapter les modèles existants pour les exploiter dans un contexte d'analyse stratégique lié aux jeux vidéo sur mobile. D'où tout l'intérêt que nous portons à ce projet de recherche.

1.4 Contribution du mémoire

La contribution du mémoire se veut riche et possède un volet théorique, un volet pratique et un volet expérimental. Ce mémoire propose de créer une adaptation des algorithmes de fouille de données se basant sur le processus de découverte de nouvelles connaissances. Par la suite, le volet pratique propose l'implémentation d'un prototype et un déploiement à travers une véritable infrastructure logicielle d'entreprise recueillant des données de joueurs à partir de plusieurs titres vendus sur cellulaire ou sur tablette. Pour ce faire, il a fallu évaluer les données fournies par la compagnie avec différents algorithmes

afin de trouver ceux qui fourniront les meilleurs résultats, non seulement en termes de résultats statistiques, mais aussi ceux qui étaient compréhensibles pour un utilisateur néophyte en fouille de données, puisque ces résultats peuvent être utiles à une prise de décision vis-à-vis du marketing des jeux ciblés. En résumé, grâce à cette architecture, il est possible de créer un modèle du type d'acheteur pour les différents jeux ainsi qu'un programme de tests pour l'évaluation d'algorithmes de data mining. L'hypothèse de départ est qu'en utilisant un algorithme de segmentation comme K-means [31] ou EM [32], il est possible d'effectuer un prétraitement des données. Ensuite, ces données transformées seront la base pour la classification. Les deux algorithmes choisis pour faire ressortir les nouvelles connaissances seront le C4.5 [26], pour un arbre de décision, et la méthode des plus proches voisins (Knn) [33, 34].

1.5 Méthodologie de la recherche

Le projet s'est déroulé en plusieurs phases. La première étape consistait en une évaluation des méthodes algorithmiques existantes et des récents travaux liés au *data mining*. Dans les sections précédentes, il a été question des différents domaines et cas d'utilisation de la fouille de données. Cette étape a permis de faire ressortir des pistes de solution pour la suite du travail. Une étude approfondie des grandes familles d'algorithmes a été réalisée, soit la segmentation [35] et la classification [36], ainsi que les algorithmes et la théorie s'y rattachant. Plusieurs algorithmes ont été analysés lors de cette phase.

La deuxième phase était la préparation des données et l'adaptation formelle des algorithmes choisis. Normalement, la première phase du projet aurait dû être la modélisation et la collecte des données. Cependant, la compagnie avait recueilli les données au préalable, sans toutefois penser qu'elle serait amenée à être utilisées pour la fouille de données. La base de données contenait des données bruitées, données qui ne reflètent pas toujours la réalité. Il fallait donc nettoyer ces informations pour que les algorithmes donnent des résultats plus précis. Cette phase est d'une grande importance puisque tous les résultats se basent sur les données.

La troisième phase était l'implémentation de l'architecture logicielle et l'utilisation des modèles théoriques définis à l'étape précédente. Pour ce faire, une application Java a été programmée pour lier le module de *data mining* et la base de données. Il a aussi fallu faire une jonction avec les données de l'entreprise et une simulation de l'environnement pour se rapprocher davantage d'un résultat final.

La quatrième phase a été l'expérimentation. Les objectifs expérimentaux étaient de tester différents algorithmes de segmentation et de classification sur des données réelles provenant d'une entreprise. Pour ce faire, il a fallu recréer l'environnement de l'entreprise dans les murs du LIARA avec un serveur et les ordinateurs connectés à celui-ci. Le programme va chercher l'information ciblée grâce aux requêtes SQL et les transmet aux algorithmes de *data mining.* Les résultats sont ensuite transférés aux modules d'évaluation des données. Tout dépendant du type de tests, le résultat est conservé pour fins d'analyse.

Puisque plusieurs algorithmes sont utilisés, il est possible d'évaluer chaque algorithme et de les comparer entre eux à savoir lequel est le plus performant, selon la situation auquel il se rapporte.

Une analyse comparative des résultats obtenus montre qu'en fin de compte, les résultats expérimentaux sont prometteurs et montrent bien l'intérêt de l'application de ce type de techniques dans le contexte.

1.6 Organisation du mémoire

Le mémoire suit l'approche scientifique et est organisé selon l'ordre chronologique suivi dans la réalisation des différentes étapes du projet. Le premier chapitre porte sur l'introduction au projet. Il relate les buts et objectifs et met en contexte les raisons de cette recherche. De plus, il introduit la problématique du projet, soit la découverte de nouvelles connaissances dans une immense base de données ainsi que la création de l'architecture logicielle supportant le processus. De plus, il met en lumière tous les éléments importants à la compréhension du problème de recherche. Cette section aborde aussi la méthodologie, c'est-à-dire la façon de faire pour chacune des étapes qui ont permis l'écriture de ce mémoire.

Le deuxième chapitre porte sur la revue littéraire; la majeure partie de cette section concerne la fouille de données. Il sera question du processus pour découvrir de nouvelles

connaissances ainsi que les algorithmes choisis pour la fouille de données. Le processus sera expliqué en détail ainsi que les différentes possibilités qui sont disponibles. Les différents algorithmes de segmentation comme K-means et de classification comme C4.5 seront couvert en détail avec la démonstration de leurs forces et faiblesses. Il sera aussi question des différentes techniques pour la préparation des données et l'amélioration des résultats pour certains algorithmes. Une présentation des objectifs et un calendrier du projet seront aussi présentés.

Le chapitre 3 décrira l'implémentation du projet. Puisqu'il s'agit d'un travail en collaboration avec une entreprise et que certaines ententes de confidentialité ont été signées, l'expérimentation et les résultats ne pourront être décrits en détail. Cependant, les étapes de l'élaboration du développement logiciel ainsi que celles qui ont permis de produire le résultat final pourront être fournies. L'implémentation du programme Java, l'écriture des requêtes SQL et le module de transfert de données seront expliqués en détail dans ce chapitre. Les métriques de jeux, c'est-à-dire les données de jeux, seront aussi couvertes. Le choix des bonnes métriques est une étape importante du processus, puisque tout le reste dépend de ce choix. L'explication des tests sera aussi abordée dans cette section.

Le chapitre 4 traitera des résultats et une analyse approfondie de ceux-ci sera effectuée. L'expérimentation a couvert plusieurs algorithmes et les résultats de chacun feront l'objet d'analyse et de comparaison pour les différentes étapes auxquelles ils ont été

utilisés. Les deux algorithmes de segmentation ainsi que ceux de classification pourront être évalués selon différents critères, comme le pourcentage d'erreur, la rapidité d'exécution ou la facilité de compréhension des résultats.

Enfin, le chapitre 5 propose une conclusion qui fera un retour sur l'ensemble du projet. Un bilan des réalisations sera effectué avec les limitations du projet ainsi que les possibilités futures, tel que l'ajustement ou l'ajout de certaines métriques ou encore l'ajout du *data mining* temporel, technique utilisée pour l'application de la fouille de données dans le temps. De plus, un retour sur les différentes approches utilisées ainsi que sur l'étude d'autres algorithmes utilisables sera aussi rédigé. Le mémoire se termine sur un bilan personnel de la recherche.

CHAPITRE 2

LA DÉCOUVERTE DE NOUVELLES CONNAISSANCES

Le présent chapitre a pour but de dresser un portrait des éléments qui permettent de faire de la prédiction de données ou de nouvelles découvertes grâce aux données. À l'aide d'énormes quantités d'informations emmagasinées dans des bases de données, il est possible de découvrir des connaissances de haut niveau. La première partie porte sur le processus de découverte de ces dites connaissances. Par la suite, la deuxième partie concerne le forage de données, une étape importante du processus de découverte ainsi que les algorithmes qui s'y rattachent. Quant à la dernière partie, elle introduit la problématique de recherche, la méthodologie et un échéancier du projet.

2.1 Mise en contexte

Ces dernières années, la quantité de données contenue dans les bases de données n'a cessé d'augmenter. Les informations s'y renfermant sont d'une telle envergure qu'il est très difficile de pouvoir les utiliser à l'état brut. C'est en 1989, lors de la conférence de l'IJCAI[1](*International Joint Conference On Artificial Intelligence*) qui se déroulait à

[1] http://www.informatik.uni-trier.de/~ley/db/conf/ijcai/ijcai89.html

Rapport-Gratuit.com

Détroit, au Michigan, que plusieurs recherches d'envergure ont mis de l'avant une méthode pour être en mesure de faire ressortir des connaissances de systèmes normalement trop complexes. C'est dans cette conférence que la « découverte de connaissances grâce aux bases de données » [2] a été mentionnée pour la première fois. Il s'agit d'un processus pour découvrir des éléments importants pouvant être, au premier abord, dissimulés. La méthode a fait ses preuves dans plusieurs champs d'expertise, soit en intelligence artificielle, en détection de fraude, en marketing, etc. Par exemple, pour ce qui est de la détection de fraude, les banques utilisent ce modèle pour évaluer les transactions et ainsi déterminer, selon les informations recueillies, les transactions qui peuvent s'avérer potentiellement frauduleuses. À l'aide de ce processus et des avancées dans les méthodes de recherche, les bases de données - aussi énormes soient-elles - sont désormais accessibles par les chercheurs et les analystes. Pour y arriver, il est important d'avoir une bonne préparation et de suivre chaque étape du processus avec rigueur pour s'assurer que les résultats soient pertinents.

Dans un autre exemple, une recherche a été faite sur l'analyse des cartes de crédit d'une banque japonaise [4]. En suivant le processus qui sera détaillé plus loin et en utilisant des algorithmes de forage de données, les chercheurs ont réussi à créer des groupes d'acheteurs et, à partir de ces groupes, ils ont trouvé des règles sur le type d'acheteur en lien avec son revenu. Ces règles ont pour but de faciliter la prise de décision sur d'éventuelles demandes de crédit. Pour ce faire, ils ont utilisé le même processus de découverte de connaissances. Par la suite, ils ont utilisé des algorithmes pour la segmentation (création de groupe distinct) et la classification, sujets qui seront

traités dans les prochaines sections. En utilisant l'algorithme K-means [31, 37] pour créer des groupes d'individus similaires, les chercheurs ont ensuite appliqué des algorithmes de réseaux de neurones et des règles de classification pour bâtir leur modèle. Le réseau de neurones est un algorithme de classification pouvant donner d'excellents résultats. Cependant, une des lacunes de cet algorithme est le calibrage initial et continuel du réseau [38]. L'autre algorithme, les règles d'associations, donnent aussi de bons résultats et est facilement interprétable. Cependant, l'algorithme créé un nombre élevé de règles [13]. Ce faisant, un trop grand nombre de règles sur des métriques pouvant être sans importances peut causer des problèmes sur le modèle final. Voici un exemple qui représente une règle très simple d'association soit le rapport entre la profession et le salaire. La première information décrit le support, qui exprime la fréquence d'apparition simultanée des éléments. Le deuxième correspond au niveau de confiance, qui lui décrit le pourcentage des transactions X contenant Y. La dernière ligne comporte la règle d'association extraite.

.925	.914	Profession (Service) → Warrantor (Income over 800YUANS)

Tableau 1.0 : Règle d'associations [4]

2.2 Processus et techniques

La découverte de connaissances et la fouille de données peuvent porter à confusion, puisque ces deux éléments tendent vers le même but. En fait, le premier

Rapport-gratuit.com
LE NUMERO 1 MONDIAL DU MÉMOIRES

pourrait être décrit comme étant le processus global dans lequel la découverte est effectuée, tandis que la fouille de donnée représente une partie précise du processus [2]. La fouille de données ou *data mining*, d'un autre point de vue, correspond à l'utilisation d'un algorithme précis afin d'extraire une similarité, une règle, un regroupement, etc.

La découverte de connaissances

Le processus de découverte de connaissances s'effectue en cinq étapes, tel qu'illustré sur la Fig. 2. En premier lieu, il est important de déterminer les objectifs et de bien choisir son domaine. Celui-ci doit être défini adéquatement puisqu'autrement, la recherche peut s'avérer trop large. En réduisant la portée du projet, il est plus facile d'imposer des contraintes, quoique certains chercheurs analysent que ces contraintes peuvent être un frein à l'exploration [3]. Ayant fixé le domaine, il faut par la suite bien comprendre les demandes du client, s'il y a lieu. La seconde partie est le choix des données ; en effet, ce choix s'avère crucial puisque tout le reste du processus s'appuie sur ces données. La prédiction s'effectue sur les données et il s'agit du point de référence, il faut donc que ces données soient soigneusement choisies. Il est possible d'utiliser plusieurs outils pour la prise de données, soit par sondage ou porte-à-porte, mais une méthode plus rapide est la prise de données automatiques. Tout ce travail se retrouve dans la partie sélection du processus. Par la suite, il faut s'assurer que les données soient nettoyées. Lors de prise d'informations, certaines données peuvent ne pas refléter la réalité, soit due à un mauvais échantillonnage, un problème de précision, une erreur humaine, etc. Il s'agit d'information appelée « bruit » [1]. Ces données doivent être supprimées pour s'assurer que les connaissances à faire ressortir soient valables. Lorsque

le nettoyage est terminé, il faut préciser la sélection en ne prenant que les attributs s'avérant nécessaires pour atteindre le but initial. Toutes les informations contenues dans la base de données ne sont pas nécessairement utiles puisqu'il y a un objectif précis. Cette étape peut s'avérer délicate, puisque les informations qui auront un impact sur le modèle final ne sont pas toujours celles que l'on croit. Lorsque les données sont prêtes, il faut choisir le type d'algorithme à utiliser. On tombe dans la transformation des donnes. Il en existe plusieurs types pour le forage de données : la classification, les arbres de décision, les forêts aléatoires, le clustering, les machines à vecteurs de support (machine à noyaux), les réseaux de neurones, etc. Le choix peut s'arrêter sur un ou plusieurs algorithmes, dépendamment du problème et des résultats attendus. C'est après cette étape que les tests sont effectués. Il faut appliquer les algorithmes sur les données et les résultats sont analysés dans la partie suivante du processus. Le tout complété, il faut représenter les résultats afin qu'ils soient compréhensibles et explicites pour tous et non seulement pour le chercheur. La dernière étape consiste à évaluer et interpréter les résultats pour voir si des connaissances peuvent en être déduites.

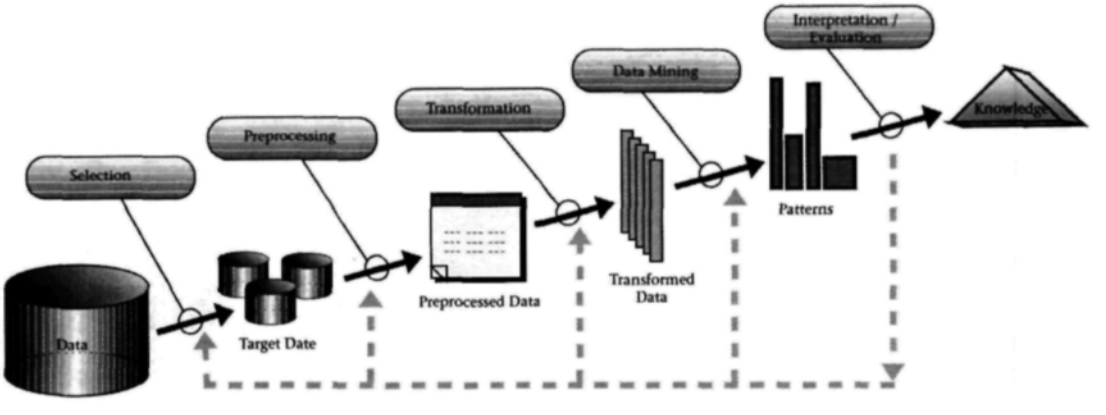


Figure 2 : Processus de découverte de données selon Fayyad [2]

Le forage de données

Comme mentionné, le but du *data mining* est d'analyser le passé pour prédire l'avenir. Cette étape consiste à appliquer un algorithme de découverte sur un ensemble de données afin d'obtenir un modèle. Par exemple, les magasins comme Walmart utilisent cette méthode pour prédire le comportement d'un acheteur. Si un acheteur tend à sélectionner l'article A, il est possible, avec une probabilité Y, qu'il achète le produit B [1]. Un exemple connu sous le nom de « panier de la ménagère » est celui de la bière et des couches pour bébé. En effet, grâce aux tests effectués, la compagnie a évalué que les deux articles étaient souvent vendus ensemble, même s'il s'agit à première vue de prémisses farfelues. Cette règle d'association ou nouvelle connaissance est le résultat d'un algorithme de fouille de données. Il est important de savoir que le choix de l'algorithme influe sur les connaissances ou concepts au final et chaque algorithme n'est pas toujours optimal pour chaque situation.

2.3 Les familles d'algorithmes du data mining

La segmentation

La segmentation, aussi souvent appelée « clustering », consiste en une séparation d'un ensemble ou groupe de données en plusieurs sous-ensembles. Ces sous-ensembles, les clusters, sont des regroupements d'éléments ou individus homogènes entre eux, mais qui diffèrent des autres sous-ensembles [31]. L'objectif est donc de regrouper

les données en « grappes » (clusters) pour former un nouvel attribut, en minimisant l'inertie intra-classe pour former des clusters et en maximisant l'inertie intra-classe pour obtenir des sous-ensembles très différents les uns des autres [39]. La segmentation est non supervisée, c'est-à-dire qu'on ne connaît pas les nouvelles classes qui seront créées. Cette méthode peut être utilisée en termes d'exploration de données ou comme étape de prétraitement pour d'autres algorithmes. Il peut s'avérer plus facile d'extraire des informations en créant préalablement des clusters, puisque plusieurs modèles peuvent être en compétition, ce qui complexifie l'extraction de connaissances [13].

Il existe deux grandes catégories d'algorithmes de segmentation : la méthode hiérarchique et le partitionnement [31, 40]. D'une part, la méthode hiérarchique produit une hiérarchie de clusters et non seulement une partition unique d'objets. De plus, le nombre de clusters(K) n'est pas obligatoirement donné, ce qui constitue un avantage considérable par rapport à certains algorithmes comme K-means. La représentation du résultat est un arbre de clusters, appelé dendrogramme. En ce qui concerne le partitionnement, il faut donner au préalable le nombre de clusters pour l'algorithme. Cependant, cet aspect accélère le traitement puisque les données sont divisées en groupes qui ne se chevauchent pas.

Pour évaluer les meilleurs groupements de données, la stratégie le plus souvent utilisée est une fonction de distance, appliquée à chaque donnée. Tout d'abord, il faut préciser cette notion de distance. La distance doit être supérieure ou égale à 0 (1). La distance vaut 0 si l'élément x est égal à l'élément y (2). La distance est la même si

l'élément x et y sont interchangé (3). La distance entre x et y est inférieure ou égale à la somme de la distance entre x et y et la distance entre y et z (4).

(1) $d(x, y) \geq 0$

(2) $d(x, y) = 0$ Si $x = y$

(3) $d(x, y) = d(y, x)$

(4) $d(x, y) \leq d(x, y) + d(y, z)$

Avec cette distance, il est possible de comparer les individus entre eux. La fonction de distance la plus souvent utilisée est la distance euclidienne. Prenons deux individus x et y, possédant n attributs : la distance euclidienne est la somme de la différence au carré de tous les attributs de chaque individu. Voici l'expression écrite à l'aide d'une formule mathématique :

$$d(x,y) = \sum_{i=i}^{n} (x_i - y_i)^2$$

Où x_i et y_i sont dans le domaine i. Il n'est pas nécessaire de résoudre la racine carrée, puisqu'on peut directement comparer les distances au carré. Cette étape améliore la vitesse de résolution de l'équation.

De plus, il existe d'autres méthodes de calcul. La distance de Manhattan, par exemple, se traduit en faisant la somme des différences absolues de tous les attributs. Voici l'expression écrite à l'aide d'une formule mathématique :

$$d(x,y) = \sum_{i=i}^{n} |x_i - y_i|$$

Aussi, la distance de Minkowski est une autre forme d'évaluation de distance. Celle-ci utilise le même procédé que la distance euclidienne, mais inclut un paramètre variable comme exposant. Ce faisant, plus l'exposant est élevé, plus l'écart entre les éléments comparés est important. L'équation s'écrit comme suit :

$$d(x,y) = \sqrt[q]{\sum_{i=i}^{n} |x_i - y_i|^q}$$

Il est important de noter que pour chaque méthode, il faut normaliser tous les éléments pour éviter que les écarts entre les attributs ne faussent les résultats. En effet, chaque domaine d'un attribut étant différent, si cette étape est omise, la différence pouvant exister pourrait grandement altérer les résultats. Pour ce faire, il faut soustraire le minimum à l'élément à normaliser, soit l'élément X_i . Par la suite, il faut diviser ce résultat par la différence entre la valeur maximale et la valeur minimale, ce qui nous donne la valeur normalisée, X* [41]. L'équation s'écrit comme suit :

$$X^* = \frac{X_i - \min(X)}{\max(X) - \min(X)}$$

Cette technique s'applique bien aux attributs numériques. Si les attributs sont nominaux, deux valeurs identiques donneront une distance de 0 et deux différents, une valeur de 1. De cette façon, il est à prévoir que les nouveaux clusters créés seront plus

efficaces. Tous ces éléments sont préalables à la segmentation. Il s'agit maintenant de choisir un algorithme de *clustering*.

Les algorithmes de segmentation

Le choix de l'algorithme est un facteur déterminant pour la suite du processus. Plusieurs algorithmes existent pour la segmentation, mais il serait trop long de tous énumérer. C'est pourquoi il sera question de deux algorithmes en particulier, K-means et EM.

L'algorithme K-means

K-means, l'un des algorithmes les plus utilisés en data mining [42], consiste à diviser l'ensemble des données en un nombre prédéfini de clusters, appelé k [41]. Tous les k points sont ensuite positionnés semi-aléatoirement (1) (se référer à la figure 3 de la page suivante). Ces points, les centroides [31], deviennent les centres des nouveaux clusters crées. Chaque élément est par la suite rattaché à l'un de ces nouveaux groupes, selon la méthode de distance utilisée (2). On replace ensuite le centre de gravité pour qu'il se retrouve au centre des individus de son groupe (3). On répète l'étape 2 et 3 jusqu'à l'algorithme converge et que les centres ne se déplacent plus ou presque, ou jusqu'à ce qu'une fonction avec critère d'arrêt soit activée. Pour un algorithme itératif, il n'est pas certain que l'algorithme atteigne un minimum global pour enclencher l'arrêt [23]. Cependant, en pratique, l'algorithme converge rapidement vers un résultat. La figure 3 démontre le processus dans son ensemble.

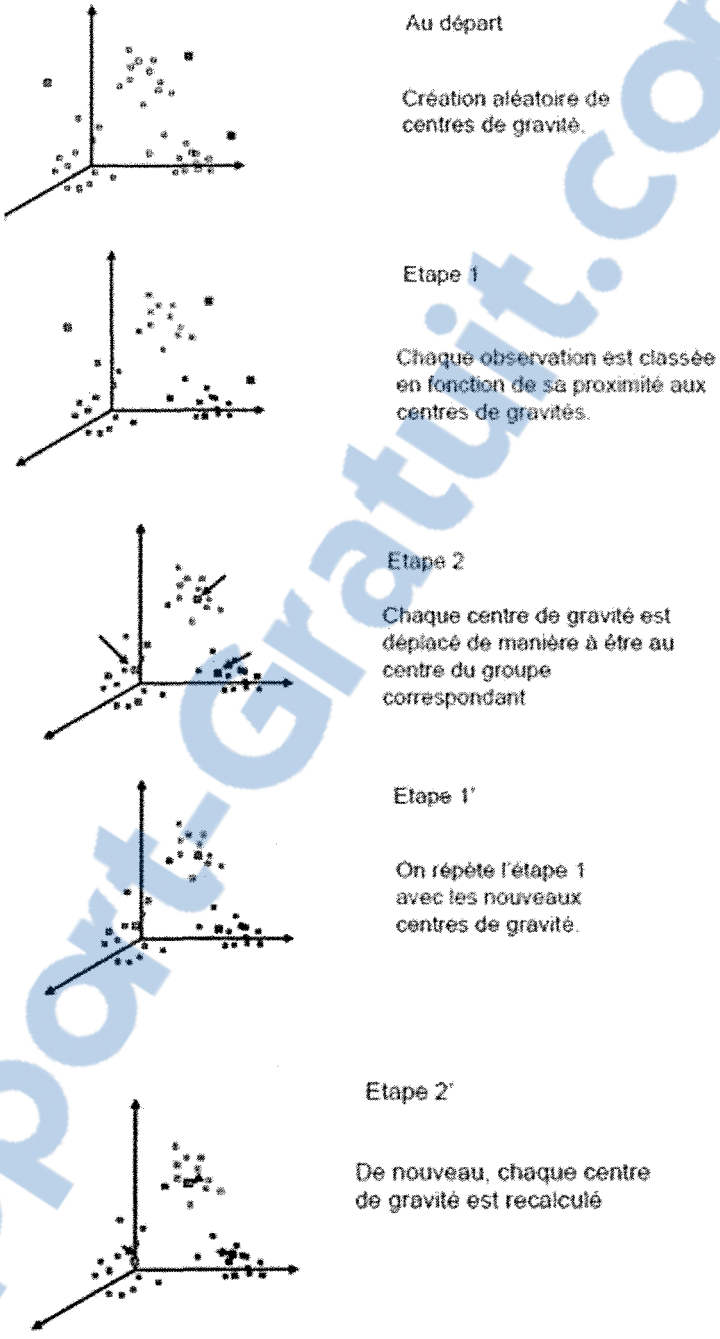


Figure 3 : Étapes de l'algorithme K-means

Rapport-Gratuit.com

Il s'agit d'un résumé de l'algorithme. De façon formelle, on peut définir X comme un ensemble de données dont chaque donnée est constituée de p attributs. On nomme « centre de gravité » g de X une donnée synthétique dont chaque attribut est égal à la moyenne de cet attribut dans X, soit $c=(m_1, m_2, \ldots, m_p)$. Par la suite, l'inertie d'un ensemble de X de N données s'exprime selon la formule suivante, où g est le centre de gravité de X.

$$I = \sum_{i=i}^{i=N} d^2(x_i, g)$$

Le pseudo-code de l'algorithme s'écrit comme suit :

Algorithme K-means

Nécessite 2 paramètres : le jeu de données X, le nombre de groupe à constituer $k \in N$
$I_w \leftarrow \infty$
Prendre k centre arbitraries $c_j \in D$
Répéter
  Pour $j \in \{1, \ldots k\}$ faire
    $G_j \leftarrow \emptyset$
  Fin pour
  Pour $i \in \{1, \ldots N\}$ faire
    $j^* \leftarrow \arg\min_{j \in \{1,\ldots k\}} d(x_i, c_j)$
    $G_{j^*} \leftarrow G_{j^*} \cup \{x_i\}$
  Fin pour
  Pour $j \in \{1, \ldots k\}$ faire
    $c_j \leftarrow$ *centre de gravité de* G_j
  Fin pour
  ReCalculer I_w
Jusque $I_w <$ *seuil*

Algorithme 1 : K-means

Cette méthode de segmentation présente beaucoup d'avantages, comme la vitesse d'exécution, de l'ordre linéaire O(tkn), où n correspond à la taille de X , k au nombre de clusters et t au nombre d'itérations. Cependant, elle présente tout de même certains inconvénients. Tout d'abord, il faut spécifier le k, soit le nombre de clusters. Aussi, l'algorithme gère mal les points isolés puisque chaque élément doit appartenir à un cluster. Une autre méthode, l'algorithme Expectation Maximisation (EM), représente une autre solution.

L'Algorithme EM

L'algorithme EM est une approche générale qui effectue un calcul itératif pour trouver des estimateurs du maximum de vraisemblance lorsque les données sont incomplètes ou manquantes. Il tient cette appellation du fait que l'algorithme consiste en une étape de calcul d'espérance et une de maximisation [8].

Avant d'entrer dans les détails d'EM, il faut d'abord savoir que l'algorithme utilise la méthode du maximum de vraisemblance pour faire la recherche. Il s'agit d'une technique qui permet de calculer, pour un échantillonnage donné, la meilleure valeur d'un paramètre d'une loi de probabilité. Il faut donc construire une fonction de vraisemblance et maximiser son logarithme par rapport aux paramètres inconnus. Pour donner un exemple, prenons Y, un vecteur aléatoire lié à certaines données y et une fonction s'exprimant $f(y|\theta)$, où θ correspond à $(\theta_i, \ldots, \theta_n)^T$. y est le vecteur comprenant des valeurs inconnues dans un espace donné. Pour le reconstruire, il faut se baser sur un autre

Rapport-gratuit.com
LE NUMERO 1 MONDIAL DU MÉMOIRES

vecteur avec un échantillonnage complet, disons le vecteur z. Si toutes les données comprises dans le vecteur z étaient ajoutées au vecteur y, celui pourrait être reconstruit. Par exemple, si un dé est lancé cinq fois (1,3,4,1,2) et par la suite, une deuxième observation avec l'élément z manquant ressemble à (1,3,z,1,2), l'algorithme remplacera z par 4 puisqu'il se fie sur l'observation précédente.

Pour ce qui est de l'algorithme, la première étape est l'évaluation de l'espérance. Cette évaluation se fait en fonction des données observées et les paramètres à notre disposition. Par la suite, il faut procéder avec la maximisation, qui consiste à évaluer le maximum de vraisemblance. Enfin, l'algorithme itère sur ces deux étapes en reprenant le paramètre trouvé en M comme point initial. Voici le pseudo-code de l'algorithme.

Algorithme EM

Nécessite 2 paramètres : le jeu de données X, une mixture de paramètres $\theta_{k,k\in\{1,\dots K\}}$
Initialiser, éventuellement aléatoirement, les paramètres de la mixture : les θ_k et les w_k
$l_{new} \leftarrow -\infty$
Répéter
  $l \leftarrow l_{new}$
  Pour $k \in \{1, \dots K\}$ faire
    Pour $i \in \{1, \dots N\}$ faire
      estimer $w_{i,k}$: la probabilité pour x_i d'appartenir à la distribution k : $w_{i,k} = \Pr[G_k|x_i]$ à l'aide de la formule décrivant la distribution du groupe G_k
    Fin pour
    $w_k \leftarrow \sum_{i=1}^{i=N} w_{i,k}$
  Fin pour
  Pour $k \in \{1, \dots K\}$ faire
    calculer θ_k
  Fin pour
  $l_{new} \leftarrow \log - vraisemblance$
Jusque $|l - l_{new}| < seuil$

Algorithme 2 : Expectation Maximisation (EM)

Ces deux méthodes de segmentation sont largement utilisées. Cependant, comme mentionnée plus tôt, la segmentation est souvent utilisée comme prétraitement, avant d'envoyer les nouvelles données dans un autre algorithme. Voici un résumé de quatres différents algorithmes selon la taille de l'échantillon de données, le nombre de clusters à utiliser, le type de données ainsi que le type de logiciel avec lequel ces algorithmes peuvent être utilisés.

	Taille de l'ensemble de données	Nombre de clusters	Type d'ensemble de données	Type de logiciel
K-means	Très grand et petit ensemble	Grand et petit nombre de clusters	Ensemble de données idéaux ou aléatoires	Visualisation d'arbre et de segmentation
HC alg.	Très grand et petit ensemble	Grand et petit nombre de clusters	Ensemble de données idéales ou aléatoires	Visualisation d'arbre et de segmentation
SOM alg.	Très grand et petit ensemble	Grand et petit nombre de clusters	Ensemble de données idéales ou aléatoires	Visualisation d'arbre et de segmentation
EM alg.	Très grand et petit ensemble	Grand et petit nombre de clusters	Ensemble de données idéales ou aléatoires	Visualisation d'arbre et de segmentation

Tableau 2.0 : Résumé de 4 algorithmes de segmentation [3]

L'autre famille utilisée, la classification, nous permet de classer les nouvelles instances découvertes.

La classification

Le but de la classification est de prendre plusieurs attributs pour représenter une classe. En fait, il faut prédire si une donnée ou un individu appartient à un groupe ou à une catégorie donnée. Pour ce faire, le processus se réalise en deux étapes. La première partie est de créer un modèle sur un ensemble d'apprentissages (ou training set en anglais). Chaque élément doit donc appartenir à une classe prédéfinie. Cet ensemble ne comprend pas toutes les données, puisqu'une partie est nécessaire pour tester l'efficacité du modèle. Le modèle résultant peut être représenté comme un arbre de décision, des règles de classifications, des fonctions mathématiques, etc. Par la suite, il faut tester avec un deuxième ensemble, l'ensemble de tests. Cette étape permet d'évaluer le taux d'erreur. Le taux d'erreur est obtenu avec le pourcentage de données de tests incorrectement évaluées par le modèle. Une notion reste néanmoins importante pour l'élaboration des ensembles d'entraînement et de test : la notion d'induction. L'induction qui consiste à déterminer les attributs pertinents qui permettront de former le modèle de prédiction. La taille des deux ensembles est un réglage important. Si le modèle d'apprentissage est trop complexe, alors il n'y aura aucun apprentissage ou un risque de surapprentissage puisqu'il s'agit d'un calque sur les données. D'un autre côté, si le modèle est trop simple et qu'on ne peut faire des liens entre les attributs, alors on peut comparer la tâche à une sélection de classe aléatoire. Il faut donc un compromis entre ces deux cas de figure. Il

faut que le modèle soit de taille moyenne et qu'il prenne un recul face aux données [13]. Si la taille de l'échantillon est trop petite, il existe des méthodes pour tenter de pallier au problème. L'une d'entre elles s'appelle le « bootstrap ». Cet algorithme utilise un principe de pige avec remise pour construire un grand ensemble de données d'entrainement. De ce fait, certaines données apparaîtront de multiples fois dans cet ensemble. Par la suite, les données n'ayant jamais été pigées se retrouvent dans l'ensemble de tests [37]. Grâce à ces ensembles, des évaluations du modèle peuvent être effectuées pour tester le taux d'erreur. Ce processus est ensuite exécuté en boucle pour obtenir plusieurs modèles. Finalement, on calcule l'erreur moyenne de chacun des modèles obtenus et celui se rapprochant le plus de ce taux d'erreur est sélectionné comme "modèle moyen". Cette solution nous permet donc de croiser les données de différentes manières afin de simuler un plus grand échantillon de données. Par contre, ceci reste une simulation et ne fournit pas toujours un résultat réaliste. Le premier algorithme qui sera traité sera l'algorithme C4.5 [24].

Algorithme C4.5

Une des premières représentations du modèle est l'arbre de décision. Pour représenter cette méthode, une revue de l'algorithme C4.5 sera utilisée. L'arbre de décision est composé de nœuds internes, qui correspondent aux attributs. Les branches correspondent au test sur la valeur d'un attribut et les feuilles à la classe donnée. Un résultat d'arbre de décision ressemble à la figure 4. Il s'agit d'un arbre qui décrit le profil

d'emprunt d'une institution bancaire. Par exemple, si un client donné dont le solde moyen est inférieur à 5000$ et que son âge est supérieur à 25, alors l'emprunt est risqué.

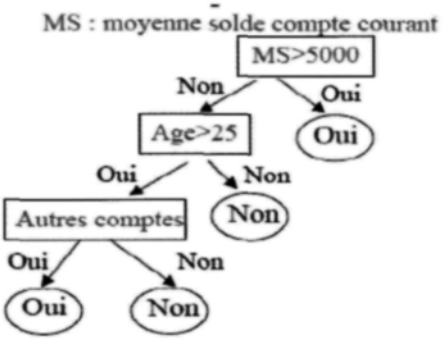


Figure 4 : Exemple d'arbre de décision

Voyons un exemple complet. Il est possible de prévoir si un joueur ira jouer au golf ou non en se basant sur des données météorologiques.

Un exemple est présenté dans le tableau 3.0 .

Tableau 3.0 : données joueur de golf

Température	Pluie	Vent	Jouer au golf
Chaud	Non	Non	Oui
Chaud	Oui	Non	Oui
Froid	Oui	Oui	Non
Froid	Non	Oui	Non
Froid	Non	Non	Oui

Avec ce type de données, il est possible de faire un modèle se basant sur l'attribut « jouer au golf ». Voici l'arbre de décision résultant (avec un résultat de 100% de classement) :

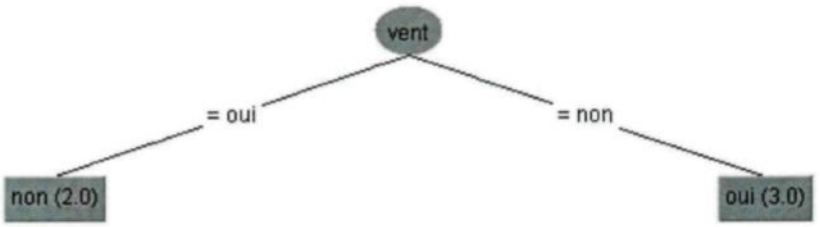


Figure 5 : Exemple d'arbre de décision avec C4.5

Cependant, il est à noter que la construction du modèle s'effectue sur les données existantes. Si une nouvelle donnée s'additionne, il se peut que le modèle préalablement créé se modifie ou donne un mauvais résultat puisque l'échantillon est trop petit. En ajoutant une nouvelle information comme dans le tableau 3, l'arbre peut être modifié comme dans la figure 3.

Tableau 4.0 : Ajout de données au modèle d'évaluation

Température	Pluie	Vent	Jouer au Golf
Chaud	Non	Oui	Oui

L'arbre ressemblera maintenant à ceci (83 % des éléments bien classés) :

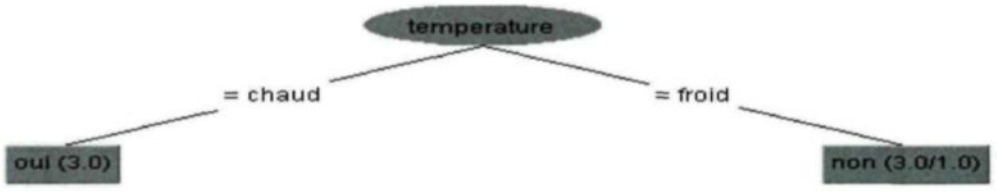


Figure 6 : Exemple d'arbre de décision avec C4.5 avec nouvelle donnée

Il est donc nécessaire d'avoir un échantillon de données assez important pour que le modèle soit stable. L'algorithme C4.5 se base sur la notion d'entropie, tirée de la théorie de l'information, pour créer l'arbre. Pour chaque attribut, il faut déterminer le gain d'information et prendre le plus élevé. Cette évaluation se fait grâce à l'entropie. Au départ, toutes les instances d'apprentissage sont à la racine de l'arbre. Il faut ensuite tester chaque attribut et trouver l'élément avec le gain d'entropie le plus grand. Soient *P* et *N*, deux classes et *S* un ensemble d'instances avec *p* éléments de *P* et *n* éléments de *N*. La formule suivante nous permet d'évaluer si l'information fait partie de *P* ou de *N* :

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

L'information nécessaire pour classifier les instances dans les sous-arbres se calcule comme suit :

$$E(A) = \sum_{i=1}^{v}\frac{p_i + n_i}{p+n}I(p_i, n_i)$$

On peut ensuite calculer le gain d'information du branchement.

$$Gain(A) = I(p,n) - E(A)$$

Lorsque l'attribut avec le gain d'information le plus important a été trouvé, il devient le premier nœud de l'arbre. On recommence l'opération pour les attributs restants afin de créer l'arbre. L'algorithme C4.5 se résume comme suit :

Algorithme 1 : Création de l'arbre de décision

Nécessite 2 paramètres : l'ensemble X et l'ensemble d'attributs { aj { 1,…,p} où p est le nombre d'attributs restants à considérer

Créer un nœud racine
Si tous les éléments de X sont positifs alors
    Racine.etiquette ← -
    Return racine
Fin si
Si A= ø alors
    Racine.etiquette value la plus présente de la classe parmi les X
Fin si
A* = arg maxa = A gain(X,a)
Racine.etiquette←a*
Pour toutes les valeurs v, de a* faire
    Ajouter une branche à racine correspondant à la valeur v_i
    Former X_{a^*vi} ⊂ X dont l'attribut a* vaut v_i
    Si X_{a^*vi} = ø alors
        À l'extrémité de cette branche, mettre une feuille étiquetée
    Avec la valeur
        La plus présente de la classe parmi les X
    Sinon
        À l'extrémité de cette branche, mettre ID3($X_{a^*= vi}$, A-{a*})

Par la suite, lorsque l'arbre est créé, il faut passer les nouvelles données pour évaluer la classe à laquelle elle appartient.

Algorithme 2 : Classification d'une donnée dans un arbre de décision

Nécessite 2 paramètres : arbre de décision AD, exemple x
// on utilise une variable nc(le nœud courant) pour parcourir l'arbre
nc ← racine (AD)
Tant-que nc != feuille **faire**
    En fonction de l'attribut testé dans nc et de sa valeur dans x, suivre l'une des branches de nc. Le nœud atteint devient nc.
Fin tant-que
Retourner l'étiquette (nc)

Rapport-Gratuit.com

L'algorithme offre plusieurs avantages. L'avantage le plus flagrant est la facilité de compréhension des résultats. En effet, l'arbre produit par l'algorithme est interprétable facilement, même pour une personne n'ayant pas de connaissances en fouille de données. De plus, l'algorithme C4.5 est robuste même avec des valeurs manquantes et des données bruitées. La classification des éléments est facile puisqu'il faut simplement parcourir l'arbre pour trouver la classe à laquelle il appartient. Aussi, cet algorithme est intéressant lorsque les classes sont nominales, mais donne de moins bons résultats lorsqu'on introduit des données numériques, puisque le nombre de feuilles créées peut rapidement devenir élevé [13]. C'est pourquoi le principe d'élagage a été créé. Il existe deux approches : éviter la trop grande croissance de l'arbre en arrêtant sa construction au bon moment ou construire l'arbre au complet et couper les branches peu significatives par la suite. Pour ce faire, un sous-arbre entier est remplacé par un nœud [20].

Algorithme Knn, plus proche voisin

L'algorithme des K-plus proches voisins est un algorithme simple, mais qui peut donner des résultats intéressants si l'étendue des données est assez grande. Il s'agit d'une méthode de classification largement utilisée dans plusieurs domaines et qui se retrouve aussi parmi les 10 meilleurs algorithmes de fouille de données [42]. Pour faire une analogie, il est possible de comparer cet algorithme à un quartier. Normalement, les maisons se retrouvant proches les unes des autres ont des caractéristiques semblables. On peut donc les regrouper et leur donner une classification. L'algorithme utilise cette même logique pour tenter de regrouper les éléments se trouvant près les uns des autres.

Tout d'abord, trois paramètres sont à prendre en considération, soit l'échantillon de données, le nombre de plus proches voisins à sélectionner (K) et le point que nous voulons évaluer (X). Par la suite, pour chaque élément de l'échantillon, nous évaluons la distance entre le point de référence X et le point X_i de l'ensemble d'apprentissages et nous vérifions si la distance les séparant est inférieure à une des distances contenues dans la liste des plus proches voisins. Si c'est le cas, le point est rajouté à la liste. Si le nombre d'éléments dans la liste est supérieur à k, la dernière valeur est tout simplement supprimée de la liste. L'algorithme en soi n'est pas très compliqué et peut donner de bon résultat avec la force brute si l'échantillonnage n'est pas trop grand. Cependant, puisque nous parlons de fouille de données, le nombre d'individus à évaluer est souvent très grand, voilà pourquoi un algorithme d'optimisation est nécessaire. Il existe plusieurs types d'arbres pour accélérer la recherche comme le KD-tree ou le balltree. L'algorithme du balltree sera couvert plus loin dans ce rapport. Tout d'abord, il faut comprendre l'algorithme du plus proche voisin. Voici le pseudo-code représentant l'algorithme:

Algorithme : prédiction de la classe d'une donnée par la méthode des k plus proches voisins

---

Nécessite 3 paramètres : un ensemble d'exemples X, une donnée x et $k \in \{1,...,k\}$

Pour chaque exemple $x_i \in X$ faire

Calculer la distance entre x_i et x : $\delta(x_i, x)$

Fin pour

Pour $j \in \{1,...,k\}$ faire

$KNN[j] \leftarrow \arg\min_{i \in \{1,...,n\}} \delta(x_i, x)$

$\delta(x_i, x) \leftarrow +\infty$

Fin pour

Déterminer la classe de x à partir de la classe des exemples dont le numéro est stocké dans le tableau KNN

Rapport-gratuit.com
LE NUMERO 1 MONDIAL DU MÉMOIRES

Le balltree et le kd-tree [43, 44] offrent de très bons résultats grâce à une méthode d'organisation spatiale des données d'apprentissage en régions sous forme d'hyper-rectangles ou hyper-sphères pour une recherche plus rapide des voisins. Un des problèmes du KD-Tree, qui utilise la séparation en hyper-rectangles, réside dans l'application de l'algorithme sur un échantillon de données possédant un nombre d'attributs élevés (supérieur à 20)[2]. Lors de la division, l'hyper-espace créé peut se déformer et donc perdre de son efficacité. Le balltree est un bon compromis puisqu'il ne se déforme pas malgré le grand nombre d'attributs. Le kd-tree réagit mieux avec un petit nombre d'éléments à évaluer, mais le balltree, qui utilise une sphère (ou un cercle lorsqu'on parle de deux dimensions) peut, en théorie, s'adapter à nombre élevé dû à sa forme. De plus, l'algorithme balltree permet qu'il y ait des intersections entre les régions et qu'elles n'occupent pas nécessairement pas tout l'espace. Ces deux points sont des éléments critiques et importants qui font la force de la représentation avec le balltree. Un autre critère à considérer est le type d'arbre pour emmagasiner les données. En effet, il existe plusieurs façons de construire un arbre et les différentes méthodes peuvent faire varier énormément le temps de calcul. Il existe plusieurs méthodes pour la construction de l'arbre :

- Construction du bas vers le haut(BottomUp)
- Construction du haut vers le bas (TopDown)
- Construction du milieu vers l'extérieur (MiddleOut)

[2] http://scikit-learn.sourceforge.net/dev/modules/neighbors.html

Pour chaque type de construction d'arbre, des tests ont été effectués pour évaluer le temps de création afin de voir lequel pourrait être utilisé dans un contexte de jeu vidéo.

Création de l'arbre – TopDown

Puisque dans tous les cas, la recherche est sensiblement la même, une seule méthode était suffisante pour comprendre le concept de l'algorithme. La création de l'arbre TopDown a donc été choisi pour à des fins de démonstration. Voici, en images, comment il est possible de créer l'arbre de recherche ainsi que de trouver les plus proches voisins.

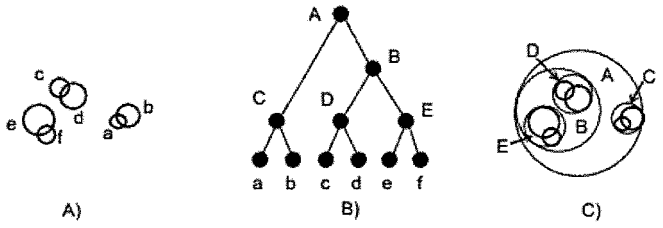


Figure 7 : A) Un ensemble de balles sur le plan B) Un arbre binaire représentant les balles C) Les balles dans l'arbre résultant

Les nœuds contiennent l'information des coordonnées du point central ainsi que celle du rayon. Les feuilles contiennent les points avec leur position pour être en mesure de faire le calcul de la distance. Dans l'exemple ci-dessus, il est possible de voir comment plusieurs ensembles de cercles peuvent être regroupés en arbre et à quoi ressemble le résultat final. Cependant, il manque plusieurs étapes de conception. Les

étapes de création de l'arbre ainsi que les étapes du parcours dans le graphe seront traitées dans la page suivante.

Recherche des plus proches voisins dans le balltree

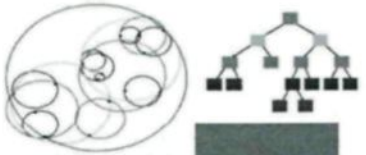
La recherche dans le balltree se fait de façon récursive. Une fois que l'arbre est construit, comme détaillé dans la première partie de la page précédente, la recherche débute dans le haut de l'arbre. La première étape est d'allez chercher les points se retrouvant dans le même cercle que l'élément recherché. Dans l'exemple précédant, la figure 6 montrent le chemin à parcourir pour trouver les deux premiers plus proches voisins. Avec cette information, il est possible de créer un cercle minimal pour l'évaluation des prochains points. À partir de la phase 5, il est facile de constater que certains points se retrouvent encore dans la proximité de Q, le point de référence. Il faut donc faire un retour arrière dans l'arbre et aller évaluer l'autre partie de celui-ci. Les figures 8 et 9 démontrent de façon évidente la façon de procédé. Puisqu'une feuille a été atteinte, on compare la distance des deux éléments avec le point de référence Q et on détermine s'il peut être introduit dans la liste des plus proches voisins. Dans l'exemple ci-dessus, on trouve un point plus proche et on remarque dans la figure 10 que le cercle de proximité a diminué et qu'il n'y a plus d'éléments plus proches. Cependant, il est à noter que l'algorithme va considérer les deux derniers cercles bleus puisqu'ils n'ont pas été encore visités.

La création de l'arbre en détail :

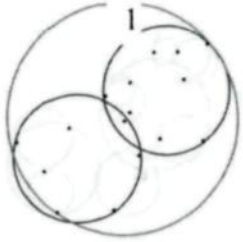
- Phase 1 : regroupement de tous les éléments et création du premier noeud de l'arbre

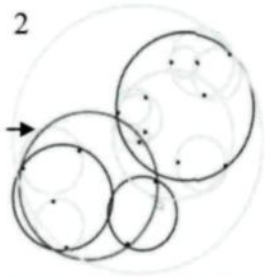
- Phase 2 : Subdivision en deux pour avoir tous les éléments dans 2 noeuds et ajout dans l'arbre

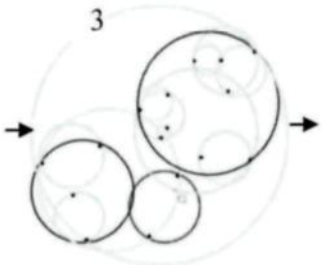
- Phase 3: Subdivision en 4 groupes pour prendre tous les éléments et ajuster l'arbre en conséquent

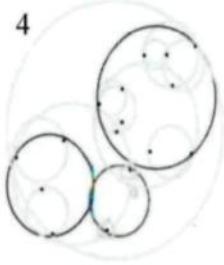


- Phase X : Lorsqu'il ne reste plus d'éléments à regrouper en pair de deux, la construction de l'arbre est terminé









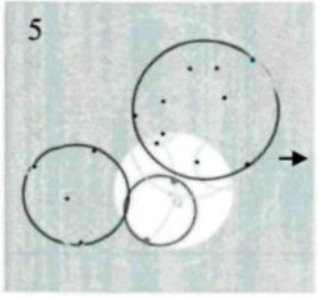
Légende

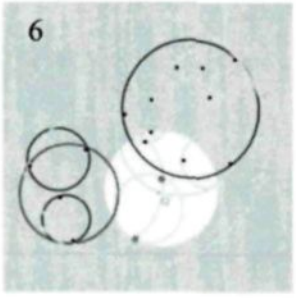
En rouge :
l'endroit actuel

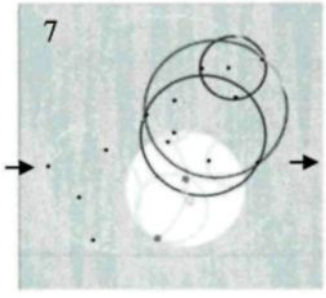
En bleu :
les endroits qui reste à visiter

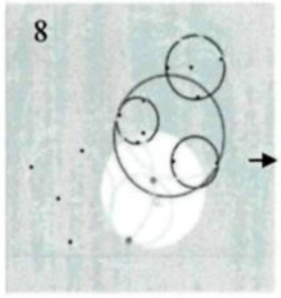
Q
point à évaluer

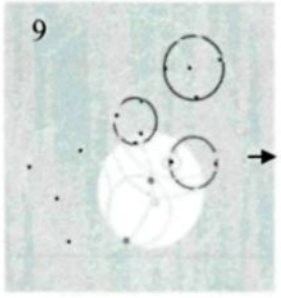
Rose
points plus proches

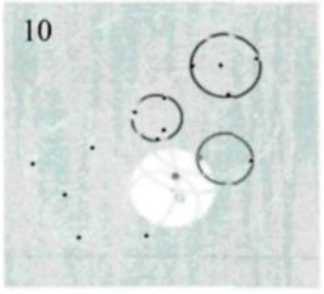












2.4 Exemple d'application en jeu vidéo

Dans le domaine du jeu vidéo, les exemples utilisant le data mining sont très récents. Il existe plusieurs façons de recueillir l'information provenant d'un jeu. Wood et al. [45] ont identifié quatre méthodes pour le faire : une méthode basée sur les sondages auprès de joueurs, une pendant le test en ligne, une avec l'observation de participants et une dernière est une entrevue en ligne. Ces méthodes nécessitent le recrutement de personnes ce qui, selon l'auteur, représente l'un des plus gros problèmes pour la recherche. D'autres techniques marginales ont vu le jour, par exemple dans l'un des jeux les plus populaires de l'histoire, *World of Warcraft,* qui consistait à créer des avatars et de les amener dans le jeu pour observer et recueillir de l'information. Cette méthode s'avère cependant couteûse en temps et ne permet que de faire des recherches ciblées. La méthode devenait totalement inefficace au fur et à mesure que l'échantillonnage était important en termes de volume de données. Les chercheurs se sont donc intéressés à un moyen d'aller recueillir de l'information différemment. Il existe des services web qui permettent aux joueurs de pouvoir analyser leur performance et la comparer aux autres joueurs. Cependant, il est impossible de prendre toute l'information en une seule étape, puisque les données sont séparées par page sur un serveur distant. Certains chercheurs ont donc créé une application qui permettait de prendre toutes les données disponibles et de les emmagasiner dans une base de données locale. Les données comportent des informations très intéressantes : plus

de 330 statistiques telles que le type de héros, l'équipement qu'il possède, le nombre de quêtes complétées, etc.

Il s'agit d'une mine d'information extrêmement intéressante en termes de recherche. Les chercheurs ont commencé par créer une structure permettant de stocker toutes les données de façon structurée et utilisable. Par la suite, ils ont pu tester des algorithmes de data mining pour faire ressortir des informations de haut niveau. Par exemple, grâce à l'inventaire des joueurs et à un algorithme de réseaux bayésiens [45], ils ont pu classifier 94% des joueurs selon la bonne classe utilisée dans le jeu, seulement en se basant sur les objets que renfermait leur inventaire. Ils ont aussi été en mesure d'identifier les classes atteignant le niveau maximal le plus rapidement, les objets le plus populaires, etc. Des informations qui s'avèrent très utiles pour le design du jeu et pour l'équilibre d'un tel titre.

Dans un autre exemple, le *data mining* a été utilisé pour créer un modèle afin de conceptualiser l'expérience du joueur [46]. En se basant sur la rétention, c'est-à-dire l'expérience qui amène le joueur à continuer de jouer, qui le garde actif et en lien avec le jeu, leur objectif était d'identifier les aspects de jeu qui permettent de maximiser cette rétention. La figure 8 représente l'interactivité qui existe entre les différents éléments à considérer, soit le joueur, le design et le jeu ainsi que les différentes connexions qui les relient. L'approche permet de créer un modèle prédictif construit à partir de la corrélation entre le joueur et les différents éléments de jeu. Le résultat final produit des

recommandations pour améliorer la rétention. Pour ce faire, les auteurs identifient deux aspects à mesurer : le premier étant sur la base de données complète, comme le nombre de joueurs actifs durant le mois, et l'autre sur les individus spécifiques, comme le temps total en jeu.

Figure 8 : L'interaction entre le joueur, le jeu et le design [1]

Ils utilisent les données de jeu et les envoient à leur système qui évalue les éléments de jeux et les classent selon l'effet prédit sur le joueur. Leur programme utilise un algorithme de régression linéaire et utilise l'outil Weka pour produire le modèle prédictif et pour lister en ordre croissant les aspects de jeu les plus influents sur la rétention.

Chaque jeu possède ses propres données et leur analyse peut se faire sous différentes formes ou différents angles. La problématique de l'évaluation du comportement

d'un joueur s'avère compliquée, surtout lorsqu'il existe une limitation sur les caractéristiques de celui-ci. En effet, plus la liste d'informations pertinentes est longue, plus facile et solide seront les conclusions pouvant en être extraites. La problématique spécifique porte sur la découverte de nouvelles connaissances avec les algorithmes de *data mining* et l'identification du meilleur procédé pour faire ressortir les connaissances. Chaque algorithme possédant leur particularité, plusieurs caractéristiques peuvent être avantageuses dans un sens et problématiques dans l'autre.

Est-il possible de créer un modèle à partir de ces données de jeu? Comment pourrait-on l'utiliser comme connaissance ou comme outil de prise de décision pour le marketing? Comment extraire ces modèles de façon efficace? Les articles nous donnent des pistes de solution, mais ne règlent pas complètement nos questionnements. La base de données actuelle comporte des milliers d'appareils et chaque appareil contient plusieurs informations de bas niveau sur le jeu. L'hypothèse de départ est qu'en utilisant un algorithme de clustering, soit K-means ou EM, il sera possible de regrouper les joueurs selon leur performance et le type de joueur et ensuite utiliser ces informations dans un algorithme de classification tel que C4.5 ou Knn pour classifier l'attribut d'achat. Ce faisant, un arbre de décision pourrait être créé comme ceux illustrés précédemment avec les informations sur le joueur. Par la suite, il pourrait être possible de créer un modèle reflétant les joueurs susceptibles d'acheter. Aussi, en ayant un patron du type de joueurs, il serait possible de tenter d'étendre ce modèle pour prendre une plus grande part de marché.

Rapport-Gratuit.com

2.5 Conclusion

Le projet se place donc entre l'informatique et le marketing. Grâce aux algorithmes du *data mining*, il faut produire un modèle fiable permettant de prédire le comportement d'un acheteur en se basant sur des données de jeu. Pour ce faire, avec l'évaluation des algorithmes qui vient d'être présentée, il faut produire un programme qui permettra l'utilisation des données de l'entreprise via les différents algorithmes. En premier lieu, le programme devra s'assurer d'aller chercher les bonnes données pour ensuite les nettoyer et les envoyer dans le module de *data mining*. D'une part, une segmentation permet de créer des types de joueurs ainsi que des regroupements sur les données. Ces regroupements permettent de passer à la seconde phase, qui consiste à classifier les personnes selon l'attribut d'achat. Avec les modèles résultant de la classification, la compagnie pourra analyser si un nouveau joueur est susceptible d'acheter ou non avec une certaine marge d'erreur. En utilisant différents algorithmes au niveau du *clustering* et de la classification, il serait intéressant d'obtenir plusieurs modèles pour identifier la meilleure méthode de segmentation. Le premier chapitre aborde plusieurs thèmes. Une vue d'ensemble concernant la problématique de la découverte de connaissances a été réalisée. Les concepts de base ont été d'abord introduits et appuyés par des exemples. Par la suite, une étude ciblée sur les différentes familles d'approches de data mining a été présentée. Dans cette étude, les principaux modèles algorithmiques existant dans la littérature (C4.5, K-Means, l'algorithme des plus proches voisins et EM) ont été

passés en revue afin de faire ressortir les forces et faiblesses de chacun de ces modèles. Ils ont été soulignés de façon à positionner les choix d'algorithmes dans la suite du mémoire.

Rapport-gratuit.com
LE NUMERO 1 MONDIAL DU MÉMOIRES

CHAPITRE 3

MODÈLE PROPOSÉ ET VALIDATION

Dans le chapitre précédent, nous avons fait un survol de plusieurs méthodes et algorithmes qui existent pour effectuer du data mining. Les résultats obtenus dans plusieurs domaines de recherche indiquent que les mêmes approches, avec les ajustements nécessaires, pourraient être incorporées dans le processus de réalisation de jeux vidéo pour ainsi fournir des indicateurs puissants pour la prise de décision et la vision stratégique d'une compagnie. Cependant, les modèles proposés ne figurent que très rarement dans le domaine du jeu et demandent une adaptation propre, et ce, pour chaque situation. Par exemple, on peut évaluer les comportements des acheteurs de chaussures, et ce, indépendamment de la marque ou du style. Ces valeurs seront des identifiants pour la prise de décision. D'une chaussure à une autre, le modèle de données reste sensiblement le même. Ce n'est pas du tout le cas avec le jeu vidéo. Chaque titre est une entité à elle seule et demande des connaissances et une évaluation propre au jeu à évaluer. Dans ce chapitre, il sera question de l'adaptation nécessaire pour le contexte de la recherche ainsi que l'architecture de travail. Il est important de bien établir les différences entre chaque algorithme et de comprendre l'impact de chacun dans le processus de découverte de

connaissances. De plus, une validation des décisions ainsi que l'impact de celles-ci seront couverts dans les sections subséquentes.

3.1 Les données

En data mining, le plus important est l'échantillon de données utilisées. Sans cette information, ou avec un échantillon inadéquat, on peut se retrouver avec des résultats qui ne reflètent pas la réalité, ou qui reflètent une réalité dissimulée. Or, le but de l'exercice est de comprendre les données pour en tirer avantage. Le modèle présenté se base sur deux phases distinctes, soit la segmentation des données, première étape primordiale pour permettre une première transformation des résultats. Ensuite, avec les données transformées par la segmentation, il est possible de classifier nos échantillons plus facilement. Il est important de comprendre et détailler la préparation des données pour chaque algorithme ainsi que l'adaptation effectuée pour chacun d'eux. Cependant, il faut d'abord bien comprendre toute l'information mise à notre disposition avant d'adapter ou changer les données.

Le jeu évalué était composé de plusieurs métriques de jeu. Le choix des métriques à incorporer dans tel ou tel algorithme dépend de la relation qu'elle peut avoir sur notre constat final. Par exemple, est-ce que le fait qu'une personne ait regardé la cinématique initiale au complet pourrait avoir un impact sur l'intention d'acheter un jeu ? Est-ce que le fait d'appuyer plusieurs fois sur le bouton « *recommencé* » est un facteur important ? Il s'agissait d'une étape délicate, car celle-ci influence directement tout le processus pour

l'obtention de notre modèle. En effet, pour chaque jeu, il existe un nombre X de métriques pouvant être discriminantes. Toutes ces métriques sont donc à prendre en considération et il est aussi important de comprendre comment elles sont constituées.

Les métriques du projet

Dès le commencement du projet, l'entreprise possédait une immense base de données contenant plus 30 millions d'événements, provenant de plus de 200 000 joueurs différents. Ces données sont principalement des informations sur des événements de bas niveau à l'intérieur des jeux et non des données personnelles sur les joueurs. Le format des données est tel que pour chaque événement recueilli, il est possible de connaître le type de cet événement ainsi que sa valeur (si nécessaire), un identificateur unique correspondant à l'appareil du joueur (e.g. l'ID de son téléphone), un identificateur correspondant à la session dans laquelle l'événement se produit, un identificateur correspondant au niveau dans le jeu (e.g. niveau 12) et le temps auquel l'événement se produit (i.e. la date et l'heure, sans connaissance du fuseau horaire).

Tout d'abord, le type d'un événement est déterminé par son nom. Un exemple inventé, "Achat" représente le fait que l'utilisateur a appuyé sur le bouton d'achat, mais cela n'indique pas qu'une transaction a été effectuée. Un tel événement ne possède pas de valeur additionnelle, puisqu'aucune information supplémentaire n'est nécessaire. En contre-exemple, un autre événement comme « Temps de niveau » indique qu'un niveau est

terminé et nécessite donc une information supplémentaire, soit le temps que le joueur a mis afin de compléter ce niveau. Concernant l'appareil du joueur, l'identificateur nous permet d'isoler une série d'événements pour un utilisateur donné. Cependant, cela ne nous permet pas de différentier le type de l'appareil (téléphone ou tablette). De plus, certains jeux offrent la possibilité de posséder plusieurs profils, chacun possédant sa propre progression. Deux personnes peuvent donc jouer une partie différente sur le même téléphone, et ce, sans distinction possible. Ensuite, l'identificateur de session nous permet de vérifier les événements apparaissant dans une même session de jeu. Une nouvelle session est créée lorsqu'un joueur démarre l'application sur son appareil, puis se termine après un certain temps d'inactivité. Aussi, chaque session est caractérisée par une durée. Celle-ci forme une information précieuse puisque le temps nécessaire à la réalisation des niveaux doit concorder avec le temps de jeu des joueurs. D'autre part, le niveau du jeu dans lequel se produit l'événement nous permet d'obtenir de l'information sur la progression d'un joueur, son désir de reprendre des niveaux afin d'améliorer ses résultats, etc. Finalement, le temps auquel se produit un événement nous informe sur les habitudes de vie d'un joueur (e.g. joue-t-il beaucoup l'après-midi, plutôt le soir, etc.).

Le choix des métriques selon l'algorithme

En partant de ces métriques, il fallait faire une sélection pour alimenter les algorithmes. Puisque le nombre de métriques est élevé et qu'il s'agit de métriques de bas niveau, la décision a été de créer des regroupements pour représenter des concepts de haut niveau grâce à la segmentation. Comme mentionnée auparavant, la segmentation peut être utilisée afin de créer une nouvelle classe à partir d'autres éléments. Par exemple, pour créer une nouvelle métrique «performance des joueurs», un regroupement entre le temps des niveaux et le nombre d'essais pour chaque niveau a été effectué. Ainsi, en utilisant un algorithme de segmentation, on obtient un classement de haut niveau selon les résultats des joueurs qu'on peut par la suite utiliser pour la classification de notre but, à savoir l'achat du produit.

La segmentation

Pour les algorithmes de segmentation, il fallait procéder en deux étapes. La première était l'évaluation des groupements possibles. Pour un jeu, nous avions le choix entre une quarantaine de métriques qui ont chacune leur spécificité. Plusieurs regroupements ont été tentés : le style de joueur (explorateur ou testeur en surface), le rendement du joueur face aux autres joueurs, les habitudes du joueur (occasionnel, moyen, accroc), etc. Ces regroupements étaient possibles en jumelant plusieurs métriques de bas niveau. Pour les habitudes d'un joueur par exemple, on pouvait compter le nombre de fois

où il se connectait à chaque mois et le nombre de temps passé par session, ce qui nous donnait un résultat par rapport aux autres joueurs. Puisque l'échantillonnage était assez grand, l'efficacité de nos algorithmes était accrue.

Une fois ces regroupements réalisés de façon théorique, il fallait nettoyer les données avant de les transférer à l'algorithme de data mining. Un des problèmes d'analyse de l'information de façon automatisée comme l'a fait l'entreprise avant de nous remettre la base de données est que certaines erreurs, pouvant paraître flagrantes pour un humain, sont tout de même des données de même format à l'intérieur de la base de données. De ce fait, ces données peuvent causer un mauvais fonctionnement des algorithmes. Le problème est qu'un très faible pourcentage des résultats enregistrés peut se trouver loin des autres et en fausser l'analyse. Concrètement, imaginons un ensemble de valeurs de temps nécessaire à la complétion d'un niveau particulier. Si certains joueurs réalisent les objectifs du niveau en environ une minute, d'autres en deux et certains en trois, on peut alors créer trois sous-ensembles plutôt homogènes. Par contre, si quelques joueurs ont laissé leur téléphone allumé et ont complété un niveau en 25 minutes, ceci n'est pas représentatif et risque d'altérer les ensembles. De ce fait, il fallait trouver une façon d'écarter ces données bruits avant de lancer un algorithme de segmentation. Pour ce faire, les données sont triées dans chaque cluster selon la distance entre ces données et le centre de gravité de celui-ci. Par la suite, un algorithme de type vorace est exécuté pour analyser le changement de variance de l'ensemble après avoir enlevé la donnée la plus éloignée. Si le changement dans la variance est très petit, alors l'ensemble était relativement homogène. La suppression est alors

annulée et nous mettons fin à l'algorithme. Dans le cas contraire, la suppression est validée et l'algorithme recommence. Cette méthode permet d'ignorer efficacement les données bruitées d'un ensemble et d'obtenir de meilleurs résultats lors de l'analyse. Lorsque le choix des données de regroupement ainsi que le nettoyage sont effectués, il nous reste à transmettre l'information aux algorithmes.

K-Means

L'algorithme K-Means a été le premier algorithme considéré pour la segmentation. La première phase n'est pas le but en soit du projet, mais bien une phase intermédiaire à l'obtention d'un modèle final. Il fallait donc un algorithme simple et efficace donnant un taux de réussite intéressant. Pour permettre l'utilisation de K-Means, quelques adaptations étaient nécessaires.

La première étape était la fixation du K, soit le nombre de clusters qui seront créés pour le modèle. Comme décrit dans la revue littéraire, ce choix est important et a une influence directe sur le pourcentage de réussite de l'algorithme. Pour pallier au problème, deux méthodes ont été utilisées. La première méthode fut par essai et erreur. Une batterie de tests a été faite pour un K variant avec plusieurs métriques différentes. Cependant, ce genre de test s'avère fastidieux. La deuxième méthode est une méthode basée sur l'algorithme EM. Pour trouver la valeur de K, on commence avec un nombre égal à 2. Par la suite, on calcule le pourcentage d'erreur et on le compare avec le pourcentage d'erreur

pour la valeur de K équivalent à 3. Si le pourcentage d'erreur est inférieur à une certaine valeur limite qu'on détermine, alors on garde la plus petite valeur de K. Autrement, on continue en boucle en incrémentant sa valeur jusqu'à ce que le pourcentage d'erreur varie très peu. Cette façon de faire permet de trouver un K de façon automatique.

Par la suite, lorsque la valeur de K est choisie, il faut sélectionner les données. Lors des requêtes à la base de données, il faut plus d'information que celles nécessaires dans l'algorithme. Il est donc nécessaire d'ignorer certains éléments de la requête. Pour ce faire, un paramètre a été ajouté pour indiquer les éléments que l'on doit ignorer. Il faut préciser que cette étape est utilisée dans tous les algorithmes et non seulement pour K-Means.

EM

Pour l'algorithme EM, un des avantages et que la valeur de K peut se déterminer automatiquement. L'adaptation de cet algorithme a été beaucoup plus simple au point de vue de l'expérimentation, puisqu'une fois l'algorithme en place avec le module de transfert des métriques, EM fonctionne a peut près de la même manière dans le processus. Cependant, une des différences est dans les options de l'algorithme. Pour l'algorithme EM, on doit paramétrer le nombre d'itérations pour la validation croisée.

Rapport-Gratuit.com

Lorsque les résultats sont obtenus pour les deux algorithmes, des tables temporaires sont créées pour emmagasiner les résultats de la segmentation. Ces résultats seront ensuite utilisés pour la phase finale, soit la classification. Les deux algorithmes utilisés sont le C4.5 et l'algorithme des plus proches voisins.

La classification

En ce qui a trait à l'étape de la classification, la première étape est de définir deux ensembles de données distincts pour l'entrainement du modèle, ainsi que pour l'évaluation de celui-ci. Normalement, l'ensemble de tests est plus petit que l'ensemble d'entraînement (environ 10% à 25% des données sont utilisés pour tester le modèle). Au début du projet, l'intérêt était davantage à analyser les données d'un nouveau jeu. Cependant, ce jeu était récemment apparu sur le marché et présentait encore un faible échantillon de données de joueurs. Le problème avec cette situation est que les ensembles d'entrainement et de tests sont souvent trop petits pour représenter adéquatement les comportements moyens des joueurs et il est donc difficile d'obtenir des résultats reflétant bien la réalité. Une solution consiste à utiliser un algorithme appelé le « bootstrap », expliqué précédemment. L'utilisation du « bootstrap » n'est pas nécessaire lorsqu'on travaille avec un très grand ensemble de données, comme ce fut le cas avec le second jeu. Dans ce cas, on construit aléatoirement les ensembles d'entraînement et de tests en respectant un ratio défini. Plus le nombre de données est important, plus stable sera le modèle final. Chaque algorithme de

classification utilisé nécessite la définition de certains paramètres pour son bon fonctionnement.

C4.5

L'algorithme C4.5 est un bon algorithme qui donne des résultats étonnants quant à son efficacité et l'expressivité du modèle de sortie, un point qui s'avérait important pour le projet. Puisque l'algorithme gère plus difficilement les valeurs numériques, les données segmentées en un nombre restreint de groupes venaient résoudre ce problème. De plus, puisqu'une première phase de segmentation était effectuée, il n'était pas nécessaire d'effectuer à nouveau un nettoyage des données. En effet, les valeurs fournies à l'algorithme étant des groupes préalablement formés, l'étape de nettoyage n'était pas nécessaire. De plus, il est possible d'avoir différentes options au niveau de l'affichage des résultats. On peut garder l'arbre complet, ce qui permet de voir où se situe chaque métrique dans l'arbre et où les séparations ont été effectuées. Il est aussi possible d'utiliser l'élagage, concept expliqué dans la revue littéraire, qui permet d'avoir la visualisation d'un arbre condensé.

Algorithme KNN

L'algorithme des plus proches voisins est un algorithme qui se base sur son voisinage pour classer la métrique sélectionnée. Il est important de déterminer certaines

Rapport-gratuit.com
LE NUMERO 1 MONDIAL DU MÉMOIRES

options pour s'assurer de son bon fonctionnement. Premièrement, il faut déterminer le type d'arbre pour optimiser l'application de l'algorithme. Pour des raisons d'étude comparative et de choix de la meilleure alternative, deux types d'arbres ont été utilisés soient le KD-tree et le balltree. Il fallait aussi déterminer la valeur de K, soit le nombre de voisins à comparer pour classer la nouvelle information. Pour ce faire, un éventail de tests avec un K variable a été mis au point pour identifier le meilleur résultat.

3.2 Développement et mise en place

La première étape du processus fut de mettre en place l'environnement de travail. Il fallait reconstruire la base de données avec les mêmes caractéristiques que celle de la compagnie. L'environnement de l'entreprise a été recréé à l'intérieur des murs du LIARA afin de simuler l'accès à la base de données en temps réel. Tout d'abord, il fallait mettre en place un serveur PostgreSQL identique à celui de l'entreprise, accessible par notre application en réseau local. D'ailleurs, des modifications ont été effectuées dans le code source de Weka [36, 47] afin d'optimiser l'exécution de requêtes SQL dans le contexte du projet.

Réalisation

Au début du projet, il fallait choisir un langage de programmation, une plateforme pour la gestion de la base de données ainsi qu'une plateforme pour le *data mining*.

Programmer des algorithmes de *data mining* est une tâche ardue et complexe. Étant donné le court laps de temps réservé au projet, il n'était pas viable de reprogrammer un grand nombre d'algorithmes dans leur intégralité, sans même savoir lesquels répondraient le mieux à nos besoins. Par conséquent, nous avons opté pour l'utilisation de l'outil Weka, un logiciel open source sous licence GPL comprenant une quantité impressionnante d'algorithmes avancés et qui proposaient des fonctionnalités qui pouvaient accélérer le processus de développement. Le fait que Weka soit distribué sous licence GPL amène certaines restrictions pour la vente de l'outil développé. Par contre, cela ne limite pas l'entreprise pour ses plans futurs. Il suffirait de reprogrammer les algorithmes utilisés pour se départir de ces restrictions. L'outil Weka est programmé en Java, ce qui rendait le langage un choix approprié pour le projet. En plus, Java est un langage de haut niveau, multiplateforme et permet d'accéder facilement à un grand nombre de systèmes de gestion de bases de données à l'aide de pilotes natifs JDBC ou de passerelles JDBC-ODBC. D'ailleurs, l'entreprise utilisait PostgreSQL, un SGBD relationnel-objet disponible sur un grand nombre de plateformes et fournissant un pilote JDBC pour un accès facile en Java.

Par la suite, l'écriture des requêtes SQL était une partie importante du projet, car elles fournissent les données aux algorithmes de *data mining*, qui pouvaient eux-mêmes être paramétrés différemment selon la nature des données reçues. De plus, les résultats devaient parfois être enregistrés sous une autre forme dans de nouvelles tables afin d'être réutilisés dans le futur. De ce fait, il fallait une solution générique pour l'exécution de requêtes SQL tout en ayant la liberté de paramétrer les algorithmes en fonction de celles-ci.

La décision la plus simple était d'utiliser des fichiers externes au code. Ceci permettait d'une part de créer un outil générique pouvant recevoir en entrée un ensemble de requêtes à exécuter, ainsi qu'un ensemble d'instructions sur le travail à effectuer avec les données recueillies sans nécessiter de compilation après chaque modification des requêtes.

Pour simplifier le contrôle d'exécution du programme, l'utilisateur doit écrire, dans le fichier principal, chaque fichier SQL à être exécuté dans l'ordre dans lequel il souhaite que ceux-ci soient exécutés. Chaque requête SQL est écrite dans un fichier ayant l'extension ".sql". Les fichiers SQL comprennent deux sections. La première est une liste de pairs paramètre-valeur qui nous permet de configurer le comportement de l'application (en-tête). Par exemple, certains paramètres auront comme effet de modifier le comportement d'un algorithme particulier, alors que d'autres indiqueront à l'application de générer une nouvelle requête afin d'écrire les résultats d'un traitement à l'intérieur de différentes tables SQL. Les requêtes d'insertion ou de modification sont toujours générés "just-in-time" et n'apparaissent dans les fichiers SQL que sous forme de paramètres. Chaque paramètre commence par le caractère "@", suivi du nom du paramètre en question. L'affectation d'une valeur se fait à l'aide du caractère "=" suivi de la valeur. Le format des différentes valeurs possible dépend des paramètres utilisés. Une explication détaillée de chaque paramètre apparaît en annexe. La deuxième section comprend une requête SQL standard faisant référence à différentes tables de la base de données de l'entreprise. Le caractère "&" délimite la section paramètre de la section SQL. Il est également possible

d'écrire des commentaires à l'intérieur des fichiers SQL en utilisant les délimiteurs "/* ... */". Voici un exemple d'en-tête de fichier utilisant l'algorithme K-means.

```
1   @algorithm          =   kmeans
2   @ignore             =   1
3   @normalize          =   true
4   @orderClusters      =   true
5   @nClusters          =   4
6   @commitTable        =   "AI_ClusterSessionsPerMonth"
7   -
```

Figure 9 : Exemple d'en-tête SQL

3.3 Implémentation

Le projet Java est séparé en trois grands modules, soit le "DataAnalyser", le "DataBridge" et le système Weka. Tout d'abord, le "DataAnalyser", qui est en fait le point d'entrée de l'application, s'occupe de lire les différents fichiers de requêtes. Ensuite, il utilise le module "DataBridge" pour interagir avec le module de base de données de Weka (exécution de requêtes, récupération de métadonnées sur les tables, etc.). Finalement, il utilise Weka pour représenter les différentes données et exécuter des algorithmes sur ceux-ci. Le rôle du "DataAnalyser" est en fait d'effectuer un ensemble d'opérations de haut niveau entre chaque étape du traitement des données, dans le but de contrôler adéquatement le comportement des différents algorithmes utilisés. La prochaine section élabore les détails d'implémentation et de fonctionnement concernant les principales fonctions du "DataAnalyser", mais voici d'abord un résumé du concept global.

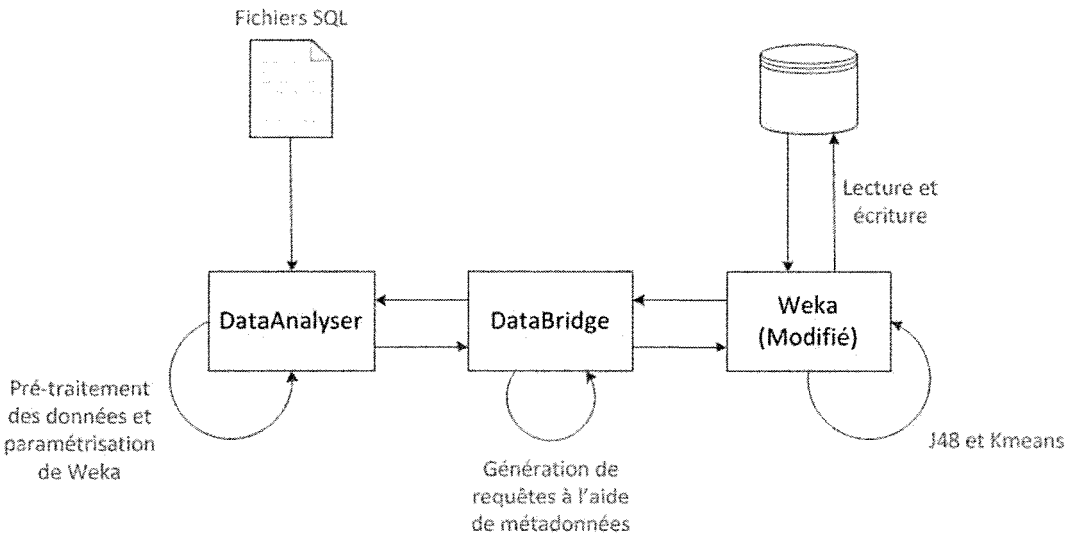


Figure 10 : Résumé du processus global du projet

Une fois un algorithme exécuté, il est important de conserver les résultats pour une utilisation future. De ce fait, il fallait gérer l'insertion de ces résultats dans des tables spécifiques de la base de données. Pour ce faire, une fonction générant automatiquement la structure d'une requête d'insertion a été programmée grâce aux métas-données de la table visée. Puisque cette partie ne s'occupe pas des données, mais de la création, elle remplit la partie "Values" avec des points d'interrogation. Le résultat pouvait donc ressembler à ceci: "Insert into nomTable(colonne1, colonne2, ..., colonneN) Values (?, ?, ..., ?). Ce format nous permettait d'utiliser le type Java "PreparedStatement" et de paramétrer la requête en remplaçant les caractères d'interrogation avec les valeurs souhaités.

Finalement, une fois l'arbre ou le modèle construit, celui-ci représente un profil d'acheteur défini par un ensemble de règles précises. Ceci a permis de comparer de nouvelles données pour les classifier. Le problème est que l'arbre expliquant le profil

d'acheteur est généré dynamiquement et peut changer avec le temps. Comme il fallait une méthode dynamique et générique pour représenter les règles et pouvoir vérifier de nouvelles données à l'exécution, la solution adoptée est celle de permettre à l'application de créer une nouvelle classe Java depuis cet arbre. Pour ce faire, l'application génère un code source sous forme de chaîne de caractères pour représenter les différentes règles. Par la suite, on utilise le "JavaCompiler", ainsi que le "ClassLoader" de java afin de compiler la chaîne dans un nouveau fichier source et appeler dynamiquement les méthodes créées à l'aide de mécanismes de réflexion. Cette solution est facilement lisible par un programmeur et très efficace puisqu'il suffit de passer les nouvelles données comme paramètre à la fonction pour obtenir le résultat souhaité. Les nouvelles données peuvent provenir de n'importe quel jeu, tant qu'elles sont toujours récupérées dans le même format.

3.4 Conclusion

Ce chapitre nous permet de comprendre les étapes nécessaires à l'adaptation des algorithmes dans le contexte du jeu vidéo. Pour chaque algorithme, il a fallu paramétrer certaines entrées et ajuster celles-ci pour obtenir des résultats significatifs. De plus, il était primordial de créer une architecture de programme qui se rapprochait de celle de l'entreprise pour permettre une réutilisation future. Chaque étape a été analysée et testée dans le détail et nous a permis de recueillir des résultats intéressants, résultats qui seront abordés dans le prochain chapitre.

CHAPITRE 4

EXPÉRIMENTATION ET ANALYSE DES RÉSULTATS

Dans le chapitre précédent, nous avons expliqué l'architecture de notre programme et les adaptations pour permettre de les appliquer dans notre contexte. Le présent chapitre portera sur l'expérimentation proprement dite et l'analyse des résultats obtenus. Il sera tout d'abord question des contraintes du projet, contraintes qui ont eu un impact significatif sur le projet en général. Par la suite, une revue des objectifs théoriques sera proposée pour ensuite expliquer les protocoles de test. Suivront les résultats expérimentaux et une analyse de ces différents résultats selon leur pourcentage de réussite et leur facilité d'utilisation.

4.1 Contraintes

Avant de discuter de l'expérimentation et des résultats obtenus, il est important de réviser les différentes contraintes qui ont été des facteurs pour le projet. Cette section décrit deux des contraintes majeures importantes rencontrées. La première contrainte vient du fait que le projet a débuté après que l'entreprise ait récolté un grand ensemble de données et qu'il était peu viable d'ajouter des métriques à celles existantes. La deuxième grande

contrainte est reliée à la difficulté d'accéder à une version à jour de la base de données, puisque le travail devait se faire avec une copie locale.

Tout d'abord, le fait que l'entreprise ait récolté les données sans l'intervention d'un expert en *data mining* s'avérait contraignant puisque certaines informations nécessaires étaient inaccessibles. En effet, normalement, le choix des données est trop important pour être pris à la légère; il s'agit d'une tâche qui incombe au spécialiste du *data mining*. Il était donc très difficile par la suite de demander l'ajout de métriques. Ce faisant, certaines métriques, comme le type d'appareil, n'étaient pas disponibles. Or, il est envisageable de dire que cette métrique aurait pu avoir une influence sur le modèle final pour la prise de décision. De plus, d'autres exemples comme l'âge et le sexe auraient pu aussi s'avérer intéressants, quoique difficiles à recueillir. Il est à noter que l'ajout de nouvelles métriques aurait pu devenir un désavantage. En effet, les données sont l'élément clé du *data mining*. En ajoutant certaines métriques, il aurait fallu délaisser un important bassin des anciennes données et ce faisant, une mine d'informations inexploitée.

Ensuite, l'autre contrainte rencontrée était celle de travailler avec une base de données inaccessible de l'extérieur des murs de l'entreprise. Par conséquent, il fallait recevoir une mise à jour hebdomadaire de l'ensemble des données. Cependant, ceci fut difficile étant donné la taille de la base de données. Il a donc été impossible de faire un suivi des données en temps réel. Une solution aurait été de simuler des mises à jour quotidiennes en utilisant seulement les événements s'étant produits avant une certaine date

Rapport-Gratuit.com

et d'ajouter les données ignorées peu à peu. Cependant, cette solution n'était pas enviable puisqu'elle aurait eu comme effet de limiter les données d'apprentissage et de tests.

4.2 Objectifs expérimentaux

L'objectif de cette recherche était de faire ressortir un modèle qui pourrait aider à l'identification d'acheteurs potentiels pour un jeu particulier. Grâce à la segmentation, des regroupements pouvaient être réalisés pour ensuite classifier ces groupements en rapport avec l'achat ou non du jeu. Nos objectifs étaient clairs, réaliser une architecture de test permettant de pouvoir envoyer des données nettoyées à nos algorithmes pour ensuite produire des clusters intéressants grâce à la segmentation. Par la suite, il fallait produire un modèle représentant un bon taux de réussite et qui faciliterait la prise de décision ou l'ajustement du jeu et des futurs jeux du même genre grâce aux nouvelles découvertes identifiées par la fouille de données.

4.3 Protocole de test

Nos phases de test se révélaient être en plusieurs étapes. La première phase de test s'est déroulée lors du nettoyage des données. En effet, lors de cette section du processus, certaines données ont été retirées de l'échantillonnage afin d'obtenir des résultats plus significatifs. Or, il fallait s'assurer que les données qui étaient retirées s'avéraient bel et bien erronées. Pour ce faire, nous avons recueilli ces données dans un fichier et les avons

rapidement aux résultats de la première étape, la segmentation. Il faut aussi déterminer le nombre de clusters. Il est soit fourni par le fichier SQL, soit l'algorithme trouve lui-même le K idéal avec la méthode détaillée dans le chapitre 2.

Voici un exemple de la recherche du meilleur K. La ligne verte indique le résultat conservé par l'algorithme :

Tableau 5.0 : Recherche du meilleur *K* avec K-means

Nombres de clusters	Écart-type	Différence
2	18.41	-
3	14.52	3.89
4	12.31	2.21
5	9.81	2.5
6	8.29	1.52

L'algorithme produit des résultats et retourne un tableau comprenant les placements de la segmentation. De plus, il est possible d'avoir l'information précise sur un cluster comme le nombre de joueurs ainsi que les bornes inférieures et supérieures de ce cluster.

La prochaine étape est de fournir les segments aux algorithmes de classification. Tout dépendamment de l'algorithme, on peut décider de mettre certains groupements ou non dans le résultat final, en fonction de la contribution au modèle. Pour ce qui est de

Rapport-gratuit.com
LE NUMERO 1 MONDIAL DU MÉMOIRES

l'algorithme des plus proches voisins, il faut préciser le nombre de voisins pour la classification. Lorsque ces étapes furent terminées, il fallait analyser les résultats de nos algorithmes.

4.4 Résultats expérimentaux

Phase de segmentation

Pour ce qui est de la phase de segmentation, voici un exemple de cluster pour un niveau donné. L'information de la première colonne indique le nom du cluster. La seconde colonne indique le numéro du cluster. La troisième et quatrième indiquent les limites inférieures et supérieures des clusters, les valeurs représentant le temps en seconde. La cinquième correspond au nom de l'attribut. La sixième est le niveau de jeu correspondant et la dernière, le nombre d'individus se retrouvant dans le cluster. Pour la première ligne, on peut donc conclure que le meilleur temps est de 49 secondes et le pire temps 170 secondes

Nom	No Cluster	Limite inférieure	Limite supérieure	Attribut testé	Niveau	Nombre de personnes
AI_BestLevelTimes	0	49	170	Meilleur Temps	3	111513
AI_BestLevelTimes	1	171	322	Meilleur Temps	3	21603
AI_BestLevelTimes	2	323	662	Meilleur Temps	3	3541
AI_BestLevelTimes	3	663	1659	Meilleur Temps	3	650

Tableau 6.0 : Cluster d'un niveau

pour le niveau 3 et 111513 individus ont été classés dans ce cluster.

Ce cluster est un résultat d'une segmentation avec l'algorithme K-means. L'algorithme EM a été testé pour le processus, mais les résultats observés ne correspondaient pas à la réalité recherchée. En fait, il faut séparer les joueurs en groupe totalement différent. L'algorithme EM a fourni des groupes qui se chevauchaient dans certains cas. En fait, les limites d'un groupe se retrouvaient dans un autre groupe. Cependant, il est intéressant de comparer les deux algorithmes sur un modèle qui fonctionne. Les deux prochains tableaux représentent les différentes informations recueillies après avoir effectué la phase de segmentation sur un même ensemble.

Tableau 7.0 : Résultat de segmentation avec **EM**

Numéro cluster	Borne inférieure	Borne supérieure	Écart-type	Nombre d'individus
0	1.53	2.21	1.54	248184
1	7.35	12.57	7.35	26233
2	21.67	54.43	21.67	3817
3	77.74	147.5	77.74	559

Tableau 8.0 : Résultat de segmentation avec **K-means**

Numéro cluster	Borne inférieure	Borne supérieure	Déviation (+/-)	Somme erreur quadratique	Nombre d'individus
0	1	11	2.1	14.2	263842
1	12	44	8.63	14.2	12010
2	45	127	20.16	14.2	2597
3	128	585	71	14.2	345

Phase de classification

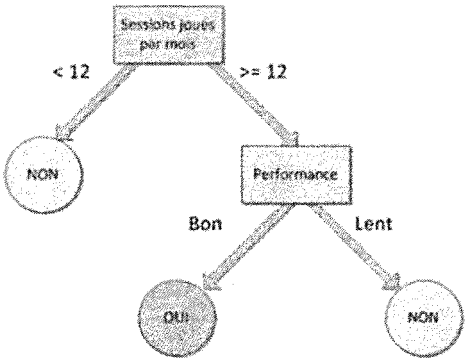
Suite au manque de données d'achats pour le premier jeu, il était difficile de construire un profil d'acheteur stable, qui se différencie d'un profil de non-acheteurs. En effet, le jeu en question ne compte que peu d'acheteurs par rapport aux gens ayant seulement essayé le jeu. La minceur de l'ensemble d'acheteurs, ainsi que le faible ratio acheteurs / non-acheteurs (~0.5%) a rendu le processus de création d'un arbre de décision difficile et a produit des résultats instables, malgré l'implémentation de techniques de *minbootstrap*. À chaque fois qu'un nouvel arbre était créé, celui-ci différait trop de l'ancien pour être considéré comme valable. Malgré plusieurs tests, les résultats demeuraient trop instables.

Suite à ces résultats, il y eut un transfert sur un deuxième jeu plus ancien, pour lequel l'ensemble des données était beaucoup plus volumineux. Selon la version de la base de données, ce jeu détenait un ratio et un nombre d'acheteurs beaucoup plus élevés. Il a fallu transposer nos différentes requêtes SQL sur ce jeu. Après cette opération, les résultats obtenus étaient beaucoup plus stables et intéressants. En utilisant seulement les données sur le nombre de sessions jouées par mois, ainsi que la performance relative moyenne (basée sur les meilleurs temps par niveau des joueurs), le modèle trouve 58% des joueurs

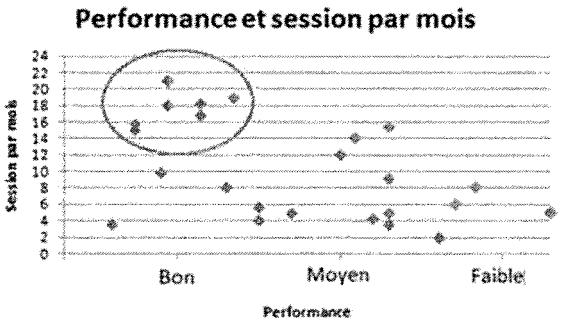
acheteurs et 98% des joueurs non acheteurs, avec un taux d'erreur global de 4% (2% taux erreur non-acheteurs, 42% taux erreur acheteurs).

Description de l'arbre et du graphique

Les graphiques ci-dessous représentent les informations qu'il est permis de présenter pour le deuxième jeu. La figure (a) montre un arbre de décision qui explique que les acheteurs sont des joueurs ayant 12 sessions ou plus par mois, et ayant une bonne performance moyenne dans le jeu. Cette performance pourrait expliquer que la clientèle de ce jeu est formée de joueurs « casual », dont la zone de *flow* est peu abrupte. La zone de flow est une zone où le joueur ressent une immersion totale au jeu avec un fort sentiment de plaisir et d'amusement [11]. Ce type de joueur a du plaisir lorsqu'un jeu présente des défis réalistes, qui ne demandent pas de posséder d'excellentes habiletés. D'ailleurs, il peut être intéressant de noter que 50% des sessions de jeux sont de 15 minutes ou moins et que le temps moyen pour effectuer un niveau est de 4 minutes. Cela montre que souvent, un joueur n'a pas le temps de refaire un niveau plusieurs fois afin d'améliorer sa performance. Finalement, la deuxième figure montre où se trouvent les données considérées comme formant un profil d'acheteur, comparativement aux autres données (toujours en utilisant les variables de performance et de nombre de sessions par mois). Le modèle à la page suivante présente un taux d'erreur de 4,4%.



(a) Arbre de décision



(b) Exemple de distribution de données. Les données entourées représentent les acheteurs.

Ces résultats représentent l'algorithme C4.5. L'algorithme des plus proches voisins a aussi été utilisé. Les résultats obtenus sont intéressants, quoique beaucoup plus difficile d'utilisation. Il est possible de faire une classification à près de 85% d'efficacité. Cependant, il s'avère d'un chiffre qui peut laisser place à interprétation. Ce volet sera traité dans la section analyse comparative.

4.5 Analyse comparative et discussion

Le premier constat et celui le plus frappant est la délimitation que l'algorithme EM effectue(voir tableau 6 et 7). En effet, le premier cluster se situe de 1.53 à 2.21 pour EM tandis que K-means résout le problème avec la limite inférieure à 1 et la limite supérieure 11. L'algorithme K-means couvre toutes les valeurs tandis que EM laisse des trous en ce qui concerne la segmentation. De plus, dans certains cas, l'algorithme EM englobe des

sous-groupes dans un autre groupe, ce qui n'est vraiment pas un résultat recherché. En effet, si un groupe se situe à l'intérieur d'un autre groupe, on pourrait donc se retrouver dans la classification avec un groupe d'acheteur à l'intérieur d'un groupe non acheteur, ce qui est incohérent. De plus, le temps d'exécution est nettement plus rapide pour l'algorithme K-means. Sur l'ordinateur utilisé (les caractéristiques seront couvertes plus loin), le temps d'exécution pour l'algorithme K-means est de 8,7 secondes tandis que l'algorithme EM prend 51,7 secondes. Il est serait donc plus viable, dans un processus que l'on voudrait implanter dans une compagnie, de choisir l'algorithme K-Means.

Pour la section classification, les deux algorithmes ont produit de bons résultats. La différence majeure est dans la lecture de ces résultats. Avec l'algorithme C4.5, on peut produire un arbre de décision qui est facilement interprétable. Une personne qui ne connaît pas l'informatique peut facilement comprendre et analyser l'arbre résultant, ce qui est totalement différent avec la méthode des plus proches voisins. En effet, l'algorithme Knn nous permet bien de classifier une donnée, mais ne nous explique pas comment elle s'y prend pour le faire. Elle se base sur ses voisins et le résultat final consiste en une classification. On ne voit pas le processus interne pour en arriver à ce résultat. De plus, le temps de classification est énorme lorsqu'on le compare avec C4.5. L'algorithme des plus proches voisins prend près d'une heure à s'exécuter avec un nombre de voisins réduits, soit 2. Il ne serait donc pas viable d'utiliser cet algorithme dans un contexte d'entreprise afin d'être une aide à la décision pour les responsables du marketing. De plus, le taux d'efficacité est bon, mais 90% des joueurs achètent et 10% des joueurs n'achètent pas. Il y

a donc 90% de chance qu'on pige un joueur qui n'achète pas et 90% de chance d'avoir un voisin qui n'achète pas non plus. Le taux d'efficacité ne départit pas les deux informations, contrairement à celui du C4.5. Aussi, lorsqu'on utilise seulement les regroupements, le pourcentage de réussite est élevé. Cependant, lorsqu'on ajoute toutes les métriques, ce pourcentage réduit rapidement. L'avantage avec le C4.5 est l'utilisation de l'élagage, qui nous permet d'avoir un meilleur aperçu des regroupements qui ont vraiment un impact. Il est donc clair que l'algorithme C4.5 est plus viable et beaucoup plus intéressant dans le contexte qui nous intéresse.

Les tests ont été effectués sur un ordinateur connecté directement à la base de données locale. Il s'agit d'un portable muni d'un processeur Intel i7 – 2640M avec une fréquence de 2.8Ghz dans chaque cœur. L'ordinateur est muni de 8 Go de mémoire vive et un système d'exploitation Windows 7 de 64 bits y est installé. Le temps total pour faire tourner l'application dure une cinquante de minutes avec les algorithmes de K-means ou EM et C4.5. Pour l'algorithme des plus proches voisins, le temps peut doubler et même tripler. De plus, lorsque la base de données se retrouve à l'externe, le temps peut tripler ou même quadrupler dans les deux cas, en fonction de la distance entre les données et l'ordinateur qui effectue l'analyse.

CHAPITRE 5

CONCLUSION

La fouille de données est un domaine qui est utilisé depuis quelques années et on commence à comprendre toute la force que ce procédé peut avoir. Même s'il s'avère difficile de l'appliquer dans le monde réel, l'idée générale et les découvertes qui ont pu ressortir avec les techniques sont surprenantes. Pour un domaine comme celui du marketing par exemple, être en mesure de reconnaître les traits d'un acheteur procure un avantage non négligeable. Il existe de nombreuses techniques de fouille de données, autant pour la segmentation que pour la classification. Les recherches ne cessent de progresser et les techniques se peaufinent avec les nouvelles découvertes et les améliorations des algorithmes. Les algorithmes comme K-means [37], EM [48], C4.5 [9] ou l'approche des plus proches voisins [49] ont été utilisés à de nombreuses reprises et on fait leur preuve quant à leur efficacité. Dans ce chapitre, un retour sera fait sur les objectifs réalisés, sur les limitations et les projets futurs qui pourraient être réalisés.

5.1 Objectifs réalisés

Tout au long du travail, certains objectifs étaient importants à réaliser. Tout d'abord, il fallait faire une revue des méthodes existantes pour permettre un choix éclairé

Rapport-Gratuit.com

quant à l'utilisation des bons algorithmes lors de nos tests sur les données. Le choix était difficile puisque plusieurs méthodes existantes s'offraient à nous. De plus, nous avions l'étape de la classification ainsi que celle de la segmentation. Pour chacune des étapes, nous avons choisi de comparer deux algorithmes. Ces algorithmes ont été choisis en fonction de leur efficacité et de leur facilité de compréhension quant aux résultats finaux. La deuxième phase a été la phase d'implémentation du logiciel. Ce programme permettant de recueillir les données, de les nettoyer et de les envoyer dans le module algorithmique selon les besoins. Le processus se séparait en deux étapes. La première phase étant de faire des regroupements d'individus qu'on appelle « cluster ». Le logiciel devait aussi conserver ces résultats après la phase de segmentation, pour ensuite diriger les résultats vers les algorithmes de classification. Les algorithmes de classification nous permettant de produire notre modèle final grâce aux données préalablement segmentées. Pour la phase de segmentation, l'algorithme K-means a démontré une meilleure constance au niveau de l'efficacité et du rendement. L'algorithme EM créant des groupes à l'intérieur des limites d'autres groupes. Pour la partie de classification, avec l'algorithme C4.5, l'arbre de décision peut être reformaté en code (arbre de si) pour faire des tests singuliers sur de nouveaux individus, sans avoir besoin de recréer l'arbre. Pour l'algorithme des plus proches voisins, le problème est qu'on ne peut conserver d'arbre. Il faut donc tester une nouvelle donnée avec l'ensemble des anciennes recueillies. De plus, il est plus difficile de comprendre l'efficacité de cet algorithme comparativement au C4.5, qui offre un arbre plus compréhensible et visuel. Le tout a permis d'atteindre le but initial, soit de comparer des algorithmes de data mining dans le contexte du jeu vidéo en plus de permettre de déceler

des connaissances pertinentes sur le jeu sélectionné. Cette réussite démontre l'applicabilité ainsi que l'intérêt pour ce genre d'approche dans notre contexte.

5.2 Limitation et travaux futurs

Ce mémoire a présenté une façon de créer une architecture pour l'obtention d'un modèle d'acheteur dans un contexte de jeu vidéo. Le modèle présenté comporte des limitations et effleure à peine toute la possibilité qu'offre le *data mining*. En effet, lors du modèle final, seulement deux métriques regroupées se retrouvent dans l'arbre résultant. Ces deux métriques ne s'appliquent pas directement au jeu, mais aux habitudes des joueurs. La précision du modèle pourrait être encore plus grande en sélectionnant des métriques ciblées qui influent sur l'expérience de jeu. Malgré tous les tests effectués avec les nombreuses métriques présentes, l'arbre le plus stable et le plus intéressant s'est avéré être un arbre simplifié. Cependant, en ajoutant certaines métriques mentionnées plus tôt, telles que le type d'appareil ou des données de jeu beaucoup plus spécifiques, il pourrait être possible de créer un arbre plus complexe et plus précis. De plus, il existe ce qu'on appelle le *data mining* temporel [39, 40], méthode qui consiste à appliquer le processus de fouille de données dans le temps. Avec cette méthode, il serait possible d'évaluer les éléments qui influent sur l'achat et faire des modifications dans le temps. Il serait possible de voir au fur et à mesure l'impact des changements sur les habitudes des joueurs. En ayant un module qui est directement connecté à la base de données de la compagnie, l'évaluation pourrait se faire de façon systématique et permettrait un suivi de l'évolution des modèles. Aussi, en

Rapport-gratuit.com
LE NUMERO 1 MONDIAL DU MÉMOIRES

ayant des métriques plus précises, il serait possible d'évaluer la suite d'action d'un joueur pour vérifier la difficulté et la compréhension d'un joueur pour un niveau donné. Les méthodes pourraient donc servir à l'amélioration du design général du jeu.

5.3 Bilan personnel

Pour conclure, il est primordial de dire que ce projet m'a énormément appris. Le fait de travailler en équipe sur un projet aussi élaboré m'a beaucoup aidé, autant au niveau du *data mining* que dans l'approfondissement de mes connaissances des bases des données et de la programmation en java. Étant professeur d'informatique au Cégep, il me sera possible de réutiliser ces connaissances dans mon enseignement. Tout le projet m'a aussi grandement appris sur le monde de la recherche scientifique.

BIBLIOGRAPHIE

1. Lennart E. Nacke , J.N., Karolien Poels, Anders Drachen, Hannu J. Korhonen et al., *Playability and Player Experience Research* 2009.
2. Usama Fayyad, G.P.-S., and Padhraic Smyth, *From Data Mining to Knowledge Discovery in Databases.* AI Magazine, 1996. **17**(3): p. 37-54.
3. Abbas, O.A., *Comparisons between Data clustering algorithms.* The international Arab Journal of Information technology, 2008. **5**(3).
4. Zheng, Y., *Analysis of credit card data based on data mining technique*, 2009, Universite du Quebec a Chicoutimi (Canada): Canada. p. 84.
5. Kiri L. Wagstaff, B.B., *K-means in space: a radiation sensitivity evaluation.* Proceedings of the 26th Annual International Conference on Machine Learning, 2009: p. 1097-1104
6. Ke Wang, S.Z., *Mining Customer Value: From Association Rules to Direct Marketing* ¤.
7. Anil K. Jain, R.C.D., *Algorithms for clustering data.* 1988.
8. Michael J. Shaw, C.S., Gek Woo Tan, Michael E. Welge, *Knowledge management and data mining for marketing.* Elsevier Science, 2001.
9. Baltié, J., *Data mining : ID3 et C4.5.* 2002.
10. Chris Lewis, N.W.-F., *Mining Game Statistics from Web Services: A World of Warcraft Armory case study.* ACM Transactions on Knowledge Discovery from Data, 2010.
11. Chen, J., *Flow in Games (and Everything Else).Communications of the ACM*, 2007. **50**(4).
12. Ian H.Witten, E.F., *Data mining : Pratical machine learning tools and techniques.* 2005.
13. Bouzouane, A., *Notes du cours en Data Mining, donné à la session d'hiver 2012 à l'Université du Québec à Chicoutimi.*, 2012.
14. Tufféry, S., *Data mining et statistique décisionnelle : L'intelligence dans les bases de données.* 2005.
15. Shomona Gracia Jacob, R.G.R., *Mining of Classification Patterns in Clinical Data through Data Mining Algorithms.* 2012.
16. David M. Fram, J.S.A., MD, PhD,William DuMouchel PhD, *Empirical Bayesian Data Mining forDiscovering Patterns in Post-Marketing Drug Safety.*
17. Martin Ester, R.G., Wen Jin, Zengjian Hu, *A Microeconomic Data Mining Problem: Customer-Oriented Catalog Segmentation.* **KDD'04**.
18. Ruggieri, S., D. Pedreschi, and F. Turini, *Data mining for discrimination discovery.* ACM Transactions on Knowledge Discovery from Data, 2010. **4**(2): p. 1-40.
19. Konstantinos Avgerinakis, A.B., Ioannis Kompatsiaris, *Activity Detection and Recognition of Daily Living Events.* 2013.

20. Lapalu, J., *FORAGE NON SUPERVISÉ DE DONNÉES POUR LA PRÉDICTION D'ACTIVITÉS DANS LES HABITATS INTELLIGENTS.* 2013: p. 96.
21. Shouning Qu, C.D., Peihua Liu, *Research and Implementation on Data mining Applied to Learning Guidance System* 2005.
22. R. Agrawal, R.S., *Mining sequential patterns.* ICDE'95, 1995: p. 3-14.
23. Jayanta Basak, A.S.a.M.S.S., *Weather Data Mining Using Independent Component Analysis.* Journal of Machine Learning Research 2004.
24. Christian Bohm, A.O., Claudia Plan, *SkyDist: Data Mining on Skyline Objects.* Springer, 2010.
25. Korting, T.S., *C4.5 algorithm and Multivariate Decision Trees.* Image Processing Division, National Institute for Space Research – INPE.
26. Quinlan, J.R., *C4.5: Programs for Machine Learning.* Morgan Kaufmann Publishers, 1993.
27. Breiman L., F.J.H., Olshen A., *classification and regression trees.* 1984.
28. Kass, G.V., *An Exploratory Technique for Investigating Large Quantities of Categorical Data.* Applied Statistics, 1980. **29**(2): p. 119-127.
30. Jiuyong Li, A.W. and H.H. Fu , Jie Chen, *Mining Risk Patterns in Medical Data.* 2005. **KDD**.
31. Khaled Alsabti, S.R., Vineet Singh, *An efficient k-means clustering algorithm.* Electrical Engineering and Computer Science, 1997.
32. Borman, S., *The Expectation Maximization Algorithm A short tutorial.* 2004.
33. Sunil Arya, D.M.M., Nathan S. Netanyahu , Ruth Silverman and Angela Y. Wu, *An Optimal Algorithm for Approximate Nearest Neighbor Searching in Fixed Dimensions.* 1994.
34. Angiulli, F. and F. Fassetti, *Nearest Neighbor-Based Classification of Uncertain Data.* ACM Trans. Knowl. Discov. Data, 2013. **7**(1): p. 1-35.
35. JON KLEINBERG, C.P., PRABHAKAR RAGHAVAN, *Segmentation Problems.* ACM Trans. Knowl. Discov. Data, 2004. **51**(2).
36. Luís C. Borges, V.M.M., Jorge Bernardino, *Comparison of Data Mining techniques and tools for data classification.* 2013.
37. Tapas Kanungo, D.M.M., C.D.P. Nathan S. Netanyahu, Ruth Silverman,, and Angela Y. Wu, *An Efficient k-Means Clustering Algorithm: Analysis and Implementation.* TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, 2002. **24**(7): p. 12.
38. Kovalerchuk, R.a.A.a.B., *Neural Networks for Data Mining: Constrains and Open Problems*, Central Washington University, Ellensburg, USA.
39. Theodoulidis, M.H.S.a.B., *Knowledge Discovery in Temporal Databases.* p. 9.
40. Claudia M. Antunes, A.L.O., *Temporal Data Mining: an overview.* computer science.
41. Junjie Wu, H.X., Jian Chen, *Adapting the right measures for K-means clustering.* Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, 2009: p. 877-886.

42. Wu, X., et al., *Top 10 algorithms in data mining.* Knowledge and Information Systems, 2007. **14**(1): p. 1-37.
43. Stern, H., *NEAREST NEIGHBOUR MATCHING USING KD-TREES.* DALHOUSIE UNIVERSITY DEPARTMENT OF COMPUTER SCIENCE, 2002.
44. Nitin Bhatia, V., *Survey of Nearest Neighbor Techniques.* International Journal of Computer Science and Information Security, 2010. **8**(2).
45. Wood, R.T.A., *Online data collection from video game players: Methodological issues.* 2004.
46. Ben G. Weber, M.M., Arnav Jhala, *Using Data Mining to Model Player Experience.* ACM 2011.
47. Narendra Sharma , A.B., Mr. Ratnesh Litoriya *Comparison the various clustering algorithms of weka tools.* ISSN, 2012. **2**(5).
48. Sugato Basu, M.B., Raymond J. Mooney, *A Probabilistic Framework for Semi-Supervised Clustering.* 2004.
49. Benny Y. M. Fung, V.T.Y.N., *Classification of Heterogeneous Gene Expression Data.* SIGKDD Explorations. **5**(2).