REPUBLIQUE ALGERIÈNNE DÈMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mohamed Khider – BISKRA
Faculté des Sciences Exactes et des Sciences de la Nature et de la Vie
Département d'informatique

THÈSE

Présentée pour l'obtention du diplôme de :

DOCTORAT EN SCIENCES

Spécialité : Informatique

Présentée par :

Naili Makhlouf

Thème

Fouille de données dynamique basée sur la stabilité des modèles dynamiques

Soutenue publiquement le : **devant le jury composé de :**

Pr. BABAHENINI Mohamed Chaouki	Professeur	Université de Biskra	Président
Pr. BOURAHLA Mustapha	Professeur	Université de M'Sila	Rapporteur
Dr. MERAH Elkamel	M.C.A	Université de Khenchela	Examinateur
Dr. DJEFFAL Abdelhamid	M.C.A	Université de Biskra	Examinateur
Dr. LAMICHE Chaabane	M.C.A	Université de M'Sila	Examinateur

Remerciements

Je remercie ALLAH tout puissant de m'avoir donné la force et la patience pour terminer ce travail de thèse.

Ma profonde gratitude et mes sincères remerciements vont à mon directeur de recherche Pr Bourahla Mustapha, professeur à l'université de M'Sila, pour m'avoir proposé ce sujet et m'avoir dirigé, conseillé et encouragé tout au long de la réalisation de ce travail ainsi que pour sa patience et sa compréhension.

Je remercie aussi les membres du jury de m'avoir fait l'honneur d'accepter de participer à mon jury de thèse.

Dédicace

Je dédie ce travail :

À mes très chers parents, que Dieu tout puissant les protège

À mes frères et mes sœurs

À ma femme et sa famille

À mes chers enfants

À tous mes enseignants

À mes amis ainsi que tous les gens qui m'ont aidé de près ou de loin à accomplir ce travail

Abstract

Abstract- The present work presents a new approach that is based on the use of stable dynamic models for dynamic data mining. The data mining is an essential technique in the process of extracting knowledge from data. It allows us to model extracted knowledge using a formalism or a modeling technique. However, the data needed for knowledge extraction is collected beforehand and the collection of this data can take a long time. So, our goal is to move towards a solution based on the modeling of systems using dynamic models and to study their stability. Stable dynamic models provide us with a foundation for dynamic data mining. In order to achieve this goal, we propose an approach based on agent-based models, the concept of fixed points and Monte Carlo method. Agent-based models can represent dynamic models that reflect or simulate a dynamic system, where such model can be considered as a data source (data generators). In this work, the concept of fixed points has been used in order to represent the stable states of the agent-based model. Finally, the Monte Carlo method which is a probabilistic method has been used to estimate some specific values, using a very large number of experiments or executions. As a case study, we chose the evacuation system of a supermarket (or a building) in case of danger, for example fire. This complex system consists mainly of the various components that constitute the building such as rows of shelves, entry and exit gates, extinguishers, etc. Also, such buildings are often full of people from different categories (age, health…). Using Monte Carlo method helped us to carry out experiments with several scenarios, which in its turn gave us more data to study this system and extract some knowledge. This knowledge allows us to predict a future situation regarding the evacuation system of the building and to anticipate improvements for its structure, in order to make such buildings more secure and avoid the maximum number of victims.

Keywords- Dynamic data mining, Agent based modelling, Monte Carlo Method, System stability

ملخص

ملخص - يتناول العمل الحالي تقنية جديدة تعتمد على استخدام نماذج ديناميكية مستقرة للتنقيب الديناميكي في البيانات. يعد مجال التنقيب في البيانات تقنية أساسية في عملية استخراج المعرفة من البيانات. ويسمح لنا هذا المجال كذلك، بنمذجة المعرفة المستخرجة باستخدام تقنيات النمذجة. غير أنه، جمع البيانات اللازمة لاستخراج المعرفة يمكن أن يستغرق وقتًا طويلاً. لذلك، هدفنا هو التوجه نحو حل يعتمد على نمذجة النظم باستخدام النماذج الديناميكية ودراسة استقرار هذه النماذج، بحيث توفر النماذج الديناميكية المستقرة أساسًا للتنقيب الديناميكي في البيانات. من أجل تحقيق هذا الهدف، نقترح طريقة جديدة تعتمد على النماذج المعتمدة على الوكلاء، مفهوم النقاط الثابتة وطريقة مونت كارلو. يمكن أن تمثل النماذج المعتمدة على الوكلاء نماذج ديناميكية تعكس أو تحاكي نظامًا ديناميكيًا، حيث يمكن اعتبار هذا النموذج مصدرًا للبيانات (مولدات البيانات). في هذا العمل، تم استخدام مفهوم النقاط الثابتة لتمثيل الحالات المستقرة للنموذج القائم على الوكلاء. كذلك، تم استخدام طريقة مونت كارلو وهي طريقة احتمالية لتقدير بعض القيم المحددة، باستخدام عدد كبير جدًا من التجارب. كدراسة حالة، اخترنا نظام الإخلاء لسوبر ماركت (أو مبنى) في حالة الخطر، على سبيل المثال، الحريق. يتكون هذا النظام المعقد بشكل أساسي من مختلف المكونات التي تشكل المبنى مثل صفوف الأرفف، بوابات الدخول والخروج، طفايات الحريق، إلخ. أيضًا، غالبًا ما تكون هذه المباني مليئة بأشخاص من فئات مختلفة (العمر، الصحة ...). يساعد استخدام طريقة مونت كارلو في إجراء العديد من تجارب لعدة سيناريوهات، والتي بدورها توفر لنا المزيد من البيانات لدراسة هذا النظام واستخراج بعض المعلومات حول نظام الإخلاء. بحيث تسمح لنا هذه المعلومات بالتنبؤ بالأوضاع المستقبلية فيما يتعلق بنظام إخلاء أو إدخال تحسينات على هيكل المبنى، من أجل جعل هذه المباني أكثر أمانًا وتجنب أكبر عدد ممكن من الضحايا.

الكلمات المفتاحية – التنقيب الديناميكي في البيانات، النمذجة المستندة على الوكلاء، طريقة مونت كارلو، استقرار النظام

Résumé

Résumé - Le présent travail présente une nouvelle approche basée sur l'utilisation de modèles dynamiques stables pour la fouille de données dynamique. La fouille de données est une technique essentielle dans le processus d'extraction de connaissances à partir de données. Cela nous permet de modéliser les connaissances extraites à l'aide d'un formalisme ou d'une technique de modélisation. Cependant, les données nécessaires à l'extraction des connaissances sont collectées à l'avance et leur collecte peut prendre beaucoup de temps. Notre objectif est donc de nous orienter vers une solution basée sur la modélisation de systèmes utilisant des modèles dynamiques et d'étudier leur stabilité. Les modèles dynamiques stables nous fournissent une base pour la fouille de données dynamique. Afin d'atteindre cet objectif, nous proposons une approche basée sur des modèles à base d'agents, le concept de points fixes et la méthode de Monte-Carlo. Les modèles basés sur des agents peuvent représenter des modèles dynamiques qui reflètent ou simulent un système dynamique, où un tel modèle peut être considéré comme une source de données (générateurs de données). Dans ce travail, le concept de points fixes a été utilisé afin de représenter les états stables du modèle à base d'agents. Enfin, la méthode de Monte-Carlo, qui est une méthode probabiliste, a été utilisée pour estimer certaines valeurs, en utilisant un très grand nombre d'expériences ou d'exécutions. Comme étude de cas, nous avons choisi le système d'évacuation d'un supermarché (ou d'un bâtiment) en cas de danger, par exemple un incendie. Ce système complexe comprend principalement les divers éléments constitutifs du bâtiment, tels que des rangées d'étagères, des portes d'entrée et de sortie, des extincteurs, etc. De plus, ces bâtiments sont souvent remplis de personnes de différentes catégories (âge, santé, etc.). L'utilisation de la méthode de Monte-Carlo nous a permis d'expérimenter plusieurs scénarios, ce qui nous a permis de disposer de davantage de données pour étudier ce système et en extraire certaines connaissances. Ces connaissances nous permettent de prévoir la situation future concernant le système d'évacuation du bâtiment et d'anticiper des améliorations de sa structure, afin de rendre ces bâtiments plus sûrs et d'éviter le plus grand nombre de victimes.

Mots clés – Fouille de données dynamique, Modélisation à base d'agent, Méthode de Monte-Carlo, Stabilité de système

Table des matières

Liste des figures

Liste des tableaux

Chapitre 1 : Introduction générale

1.1. Contexte de la thèse

Au cours de ces dernières années, l'utilisation des techniques de fouille de données (Data Mining) s'est élargie très rapidement, où elle est devenue omniprésente dans les pratiques des entreprises et celles des personnes. L'explosion d'information qui s'apparait dans le monde à cause de la voluminosité, la sensibilité et la complexité de l'ensemble de données collectées ; joue un rôle principal dans l'apparition du domaine de fouille de données, où l'utilisation de cette dernière améliore grandement les performances des techniques d'analyse de données, afin d'offrir des modèles ou des informations utiles, compréhensibles et à jour. De plus, les techniques et les méthodes de fouille de données permettent aux organisations de tirer plus d'informations à travers des modèles compréhensibles qui sont construits en utilisant des ensembles de données homogènes ou hétérogènes collectés de diverses sources de données (comme les bases de données distribuées, le Web, les entrepôts de données et les images satellitaires).

Généralement, la fouille de données peut être utilisée dans tous les domaines, où il y a un besoin d'analyser une grande collection de données ou de prévoir l'évolution d'un processus ou d'un système donné. Parmi les domaines d'application de la fouille de données, on peut citer le commerce électronique, le marketing, la médecine, la biologie, la sécurité d'information, l'éducation et la télécommunication.

Le processus de fouille de données se base sur des collections de données enregistrées dans des supports de stockages centralisés ou distribués, où ces collections sont statiques et ne subissent pas de changement. Cependant, dans la vie réelle, les données analysées deviennent de plus en plus compliquées et dynamiques (se modifient continuellement), comme les séries temporelles, les données à temps réel, les données satellitaires et les comportements d'individus. De plus, les nécessités d'accélérer le processus de fouille de données et d'avoir des connaissances à jour ou en temps réel pour, par exemple, des besoins de sécurité ou de compétitivité conduisent à rechercher de nouveaux outils pour générer des collections de données, avec la prise en compte des caractéristiques du système étudié. Les défis cités précédemment participent essentiellement à l'apparition de nouvelles tendances de la fouille de données, à savoir la fouille de données dynamique, qui consiste à traiter des données dynamiques, en utilisant des modèles dynamiques. Parmi les techniques utilisées dans la

fouille de données dynamique, on peut citer les réseaux bayésiens dynamiques, les réseaux de neurones dynamiques et le clustering dynamique.

1.2. Problématiques

La fouille de données peut avoir plusieurs tendances selon les types de données traitées, comme la fouille de texte, la fouille de Web et la fouille de données médicales. Cependant, les utilisateurs des systèmes de fouille de données installés sont intéressés également par leurs technologies associées, et le seront d'autant plus que la plupart de ces installations devront être mises à jour à l'avenir au fur et à mesure avec les évolutions qui peuvent s'apparaitre au niveau des systèmes étudiés, ou bien dans des bases ou des entrepôts de données qui se modifient constamment avec le temps.

La fouille de séries temporelles ou la fouille de flux de données peuvent être utilisées par de nombreuses applications telles que le commerce électronique, la détection d'intrusion et la fouille de données ubiquitaires, ce qui nécessite des modèles dynamiques de fouille de données. Par exemple, la fouille de séries temporelles permet d'étudier des tendances cycliques, des tendances saisonnières, des événements ou des processus aléatoires, concernant des phénomènes météorologiques et des changements boursiers. D'autre part, dans la fouille de données spatiales, environnementales ou géographiques (qui peuvent avoir plusieurs aspects, tels que la distance, la topologie et l'aspect de temps), la construction des modèles de fouille de données qui se transforme d'une manière synchrone avec le système étudié, améliore énormément la qualité et la fiabilité des informations obtenues.

En effet, pour chaque technique de fouille de données utilisée, on peut trouver des méthodes dynamiques correspondantes. Par exemple pour les réseaux bayésiens, on trouve les réseaux bayésiens dynamiques, pour le clustering, il existe le clustering dynamique et pour la classification, il y a la classification dynamique.

En outre, les données utilisées pour le processus de fouille de données sont collectées au préalable, et la durée de collection peut être longue. Étant donné que l'objectif principal de la fouille de données est de prendre les bonnes décisions en vue d'avoir un gain sur le fonctionnement d'un système, laisser le système fonctionner pour une longue durée afin de collecter ces données peut entrainer des pertes. Donc, il apparait un besoin pour accélérer le processus de collection de données.

Généralement, les données rassemblées représentent l'historique d'un système donné. Toutefois, pour obtenir ces données plus rapidement, les outils de simulation peuvent fournir des mécanismes et des moyens appropriés. Pour atteindre cet objectif, il est nécessaire de construire en premier lieu, un simulateur (un modèle dynamique) qui représente convenablement le système étudié, avant d'exploiter les données générées par ce modèle alternatif.

1.3. Contribution

Pour résoudre les problèmes ci-dessus qui consistent à obtenir l'ensemble de données nécessaires à une étude de fouille de données à travers un modèle dynamique, nous proposons une solution basée sur la modélisation des systèmes à l'aide des modèles à base d'agents sociaux [1]. La modélisation basée sur les agents est adoptée pour modéliser de nombreux systèmes dynamiques complexes, en particulier ceux qui incluent des individus autonomes comme les sociétés de personnes, les sociétés d'animaux, les robots collaboratifs et les sociétés d'insectes. Parmi les systèmes qu'on peut modéliser par la simulation sociale à base d'agents, il existe les systèmes d'évacuation de constructions ou d'immeubles, tels que les supermarchés, les hôpitaux et les usines.

L'évacuation de supermarchés se déroule principalement dans des bâtiments avec des contraintes différentes comme les emplacements des rangées d'étagères, les emplacements des personnes et celles des portes de sortie. Dans le cas d'une catastrophe comme l'incendie, étudier un tel système en utilisant un modèle dynamique a une grande importance afin d'éviter le maximum de pertes. Donc, le modèle qui représente ce type de système doit prendre en compte plusieurs facteurs tels que le temps, les caractéristiques du supermarché et les caractéristiques des personnes. Dans cette étude, un modèle à base d'agents sociaux a été conçu pour visualiser le comportement dynamique de ce système via ces entités internes et leurs interactions. Pour évaluer la stabilité des comportements de ce modèle, nous utilisons la méthode Monte-Carlo et le concept d'état stable, qui nous fournit une base de données stable pour la fouille de données dynamique.

1.4. Organisation de la thèse

Le reste de la thèse est structurée en quatre chapitres, et elle se termine par une conclusion générale dans laquelle nous avons présenté le bilan et les perspectives de ce travail.

D'abord, le chapitre 2, intitulé « Fouille de données dynamique », introduit les domaines de la fouille de données et la fouille de données dynamique. Dans ce chapitre, nous développons les principaux concepts concernant ces domaines, à savoir leurs définitions, leurs processus de fonctionnement, leurs domaines d'application, leurs tendances et leurs défis. Ce chapitre contient également des définitions d'objets dynamiques, à savoir les données temporelles, les séries temporelles et les flux de données.

Le chapitre 3, intitulé « Modèles de fouille de données dynamique », présente en premier lieu quelques modèles de fouille de données, par exemple les arbres de décision, les règles d'association et les réseaux de neurones. Ensuite, nous mettons l'accent sur les modèles de fouille de données dynamique, par exemple les réseaux bayésiens dynamiques, les réseaux de neurones dynamiques et les chaînes de Markov cachées.

Les chapitres 4 et 5 sont consacrés à la description de l'approche proposée et la manière de l'utiliser. Dans le chapitre 4, nous présentons la méthode de modélisation utilisée qui est la simulation sociale à base d'agents. Cette méthode de modélisation permet de concevoir un modèle dynamique qui représente le système étudié et permet également de générer l'ensemble de données nécessaires pour appliquer une des méthodes de fouille de données dynamique. Ce chapitre est divisé en trois sections. La première section décrit les concepts reliés à la simulation à base d'agents sociaux, à savoir : la simulation informatique, la simulation sociale et la modélisation à base d'agents. La deuxième section comprend le concept d'état stable dans un modèle à base d'agents sociaux. Tandis que la troisième section explique l'application de la méthode de Monte-Carlo pour analyser les résultats générés par la simulation (étudier la stabilité ou la convergence de ces résultats). Ensuite, dans le chapitre 5, nous appliquons la simulation sociale à base d'agents et la méthode de Monte-Carlo pour concevoir, implémenter et analyser un modèle dynamique qui permet de simuler un système qui représente l'évacuation d'un supermarché dans le cas d'un incendie.

Nous terminons cette thèse par une conclusion générale qui présente une synthèse des travaux réalisés et nous citons également quelques perspectives pour des travaux futurs.

Chapitre 2 : La fouille de données dynamique

3.1. Introduction

Il y a plusieurs types de problèmes qui affectent considérablement le fonctionnement de la fouille de données conventionnelle, comme la diversité de types de données, la performance des algorithmes de fouille de données et la protection de données confidentielles. Par conséquent, ces problèmes contribuent décisivement à l'apparition et le développement de plusieurs nouveaux types de fouilles de données, comme la fouille de données multimédia, la fouille de texte, la fouille de Web et la fouille de données parallèles. Donc dans le même contexte, dans ce chapitre, on va introduire le domaine de fouille de données. Ensuite, on va présenter le domaine de fouille de données dynamique.

3.2. La fouille de données

Nous vivons dans un monde plein de données (data), et le plus grand défi consiste non seulement à obtenir ces données, mais également à les exploiter pour trouver de nouvelles et utiles connaissances à partir de cette mine de données. Face à cet intérêt, le domaine de fouille de données est émergé, et prend actuellement, une place très importante dans plusieurs domaines comme la science, l'informatique et l'économie.

Dans ce qui suit, on va montrer les principaux concepts et éléments de base inclus dans le domaine de fouille de données, en présentant aussi ses principaux techniques et domaines d'application.

2.2.1. Définition de la fouille de données

La fouille de données ou le Data Mining (DM) est une technologie qui allie des méthodes d'analyse de données traditionnelles à des algorithmes sophistiqués (par exemple l'apprentissage automatique et la reconnaissance de forme), en explorant de gros volumes de données afin de trouver des modèles ou des connaissances utiles [2]. Comme le montre la figure 2.1, la fouille de données utilise plusieurs concepts et techniques de différents domaines.

La fouille de données offre également des opportunités importantes pour la fouille de nouveaux types de données, ainsi que la fouille d'anciens types de données par de nouvelles manières.

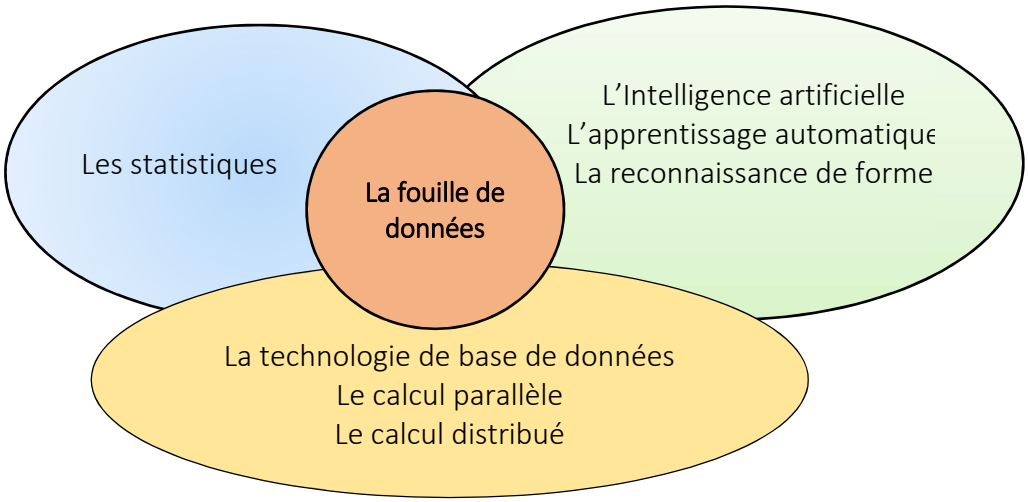


Figure 2.1. La fouille de données comme une confluence de nombreuses disciplines [2]

La fouille de données représente aussi une nouvelle technologie puissante avec le grand potentiel pour aider les entreprises à s'intéresser sur les informations les plus importantes dans leurs entrepôts de données. Elle emploie des techniques de statistique, d'apprentissage automatique et de visualisation, pour découvrir et présenter les connaissances découvertes sous une forme qui est facilement compréhensible.

2.2.2. La liaison entre la fouille de données et l'extraction de connaissances

Bien que la fouille de données et l'extraction de connaissances à partir de données (ECD) soient souvent traitées comme des synonymes. Cependant, la fouille de données fait en réalité partie du processus ECD [3] [4] qui représente le processus global de conversion des données brutes en informations utiles. Comme le montre la figure 2.2, ce processus consiste en une série d'étapes de prétraitement des données au post-traitement des résultats de la fouille de données.

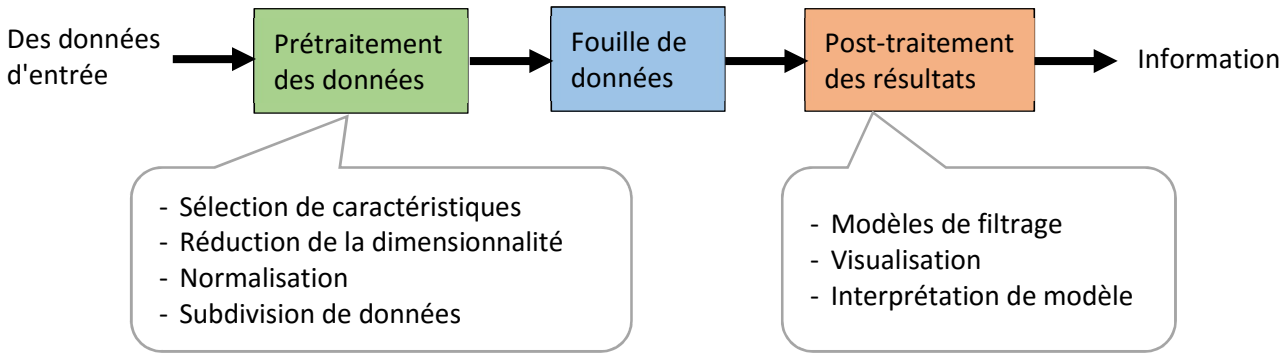


Figure 2.2. Le processus d'extraction des connaissances à partir de données [2].

Un autre plan plus détaillé du processus de découverte de connaissances dans les bases de données est le suivant :

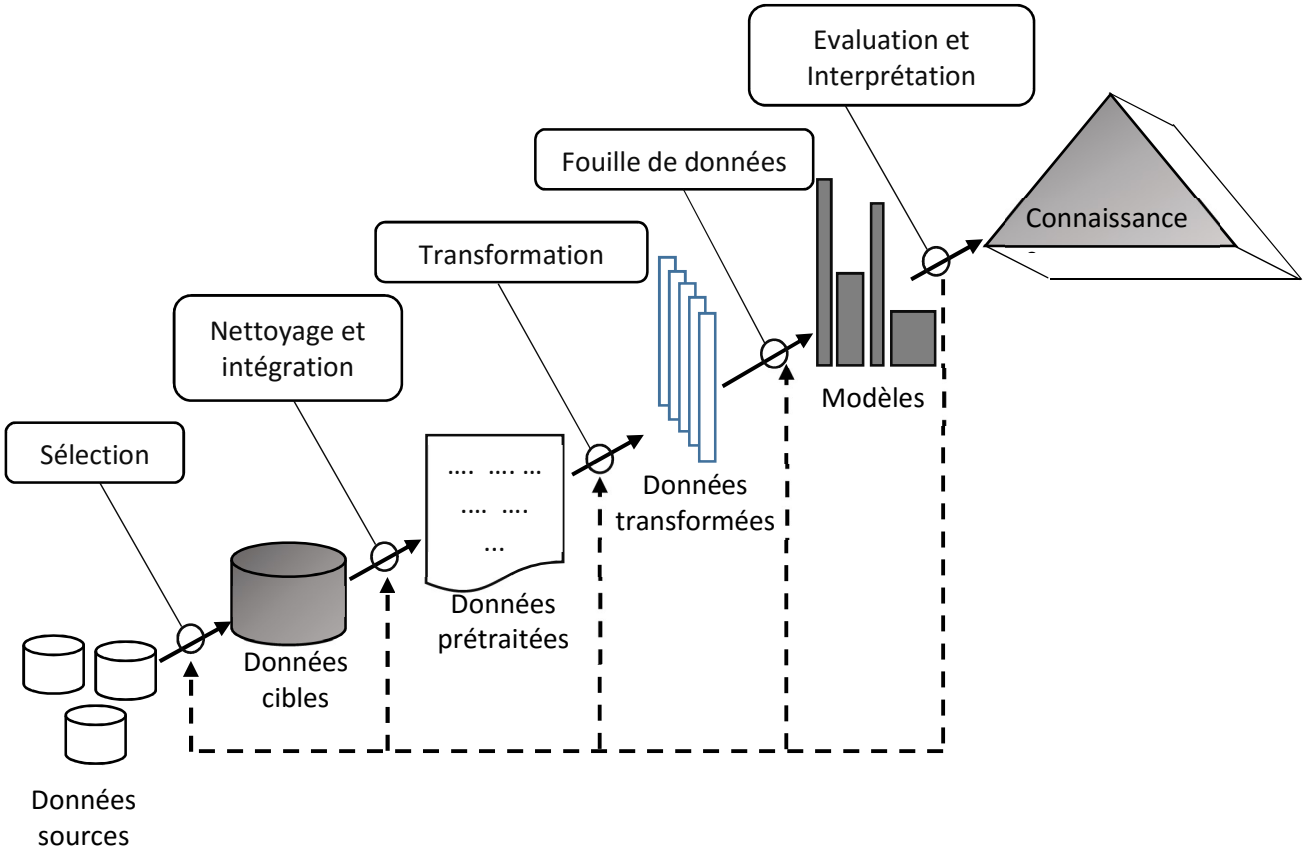


Figure 2.3. Aperçu des étapes qui composent le processus ECD [4].

La figure 2.3 montre bien les différentes étapes du processus ECD, ainsi que les résultats de chaque étape. Comme il est représenté dans cette figure, le processus ECD est exécuté d'une manière séquentielle et itérative. L'étape d'évaluation donne plusieurs possibilités, à savoir finir le processus avec des connaissances nouvelles ou revenir à une étape précédente. La séquence d'étapes de ce processus va être expliquée dans les paragraphes suivants :

D'abord, les données d'entrée peuvent avoir une variété de formats (par exemple, des fichiers textes, des tables Excel ou des tables relationnelles), et elles peuvent être stockées dans un entrepôt de données centralisé ou distribuées sur plusieurs sites.

Ensuite, des opérations de prétraitement de données sont réalisées. Le but du prétraitement est de transformer les données d'entrée brutes dans un format approprié pour une analyse ultérieure. Les étapes de prétraitement des données peuvent inclure la fusion des données provenant de plusieurs sources, le nettoyage des données (supprimer les données bruit et observations répétées), et la sélection des enregistrements et des attributs pertinents pour la tâche de fouille de données en cours. En raison des nombreuses manières dont les données

peuvent être collectées et stockées, le prétraitement des données peut être l'étape la plus laborieuse et la plus longue du processus global de découverte des connaissances [2].

Après l'étape de prétraitement, on obtient un ensemble de données prêtes à être analysées à travers des méthodes de fouille de données comme la classification, le clustering et les règles d'association. Cependant, avant de commencer la fouille de données, il faut identifier d'abord les objectifs, les méthodes et les contraintes de cette opération.

Finalement, après avoir eu des connaissances (des modèles, des tendances et des relations) par des méthodes de fouille de données ; ces résultats doivent être évalués, visualisés et exploités par des techniques appropriées. Par exemple, dans les applications de gestion (business applications), les connaissances offertes par les résultats de la fouille de données peuvent être intégrées à des outils de gestion de l'entreprise, afin que des promotions marketing efficaces puissent être menées et testées. Une telle intégration nécessite une étape de post-traitement qui assure que seulement les résultats valides et utiles sont incorporés dans le système d'aide à la décision. Un autre exemple de post-traitement est la visualisation qui permet de montrer les informations et les résultats de processus de fouille de données en utilisant une variété de graphiques et d'outils de visualisation. Également, des mesures statistiques ou des méthodes de test d'hypothèses peuvent être appliquées pendant le post-traitement pour éliminer les faux résultats de la fouille de données.

2.2.3. Architecture d'un système de fouille de données

La fouille de données est un processus très important où des informations potentiellement utiles et précédemment inconnues sont extraites de grands volumes de données. Un certain nombre de composants sont impliqués dans le processus de fouille de données constituant son architecture. Ces composants sont la source de données, le serveur d'entrepôt de données ou de base de données, le moteur de fouille de données, le module d'évaluation de modèles découverts, l'interface utilisateur graphique et la base de connaissances [3]. La structuration de ces composants est présentée dans la figure suivante.

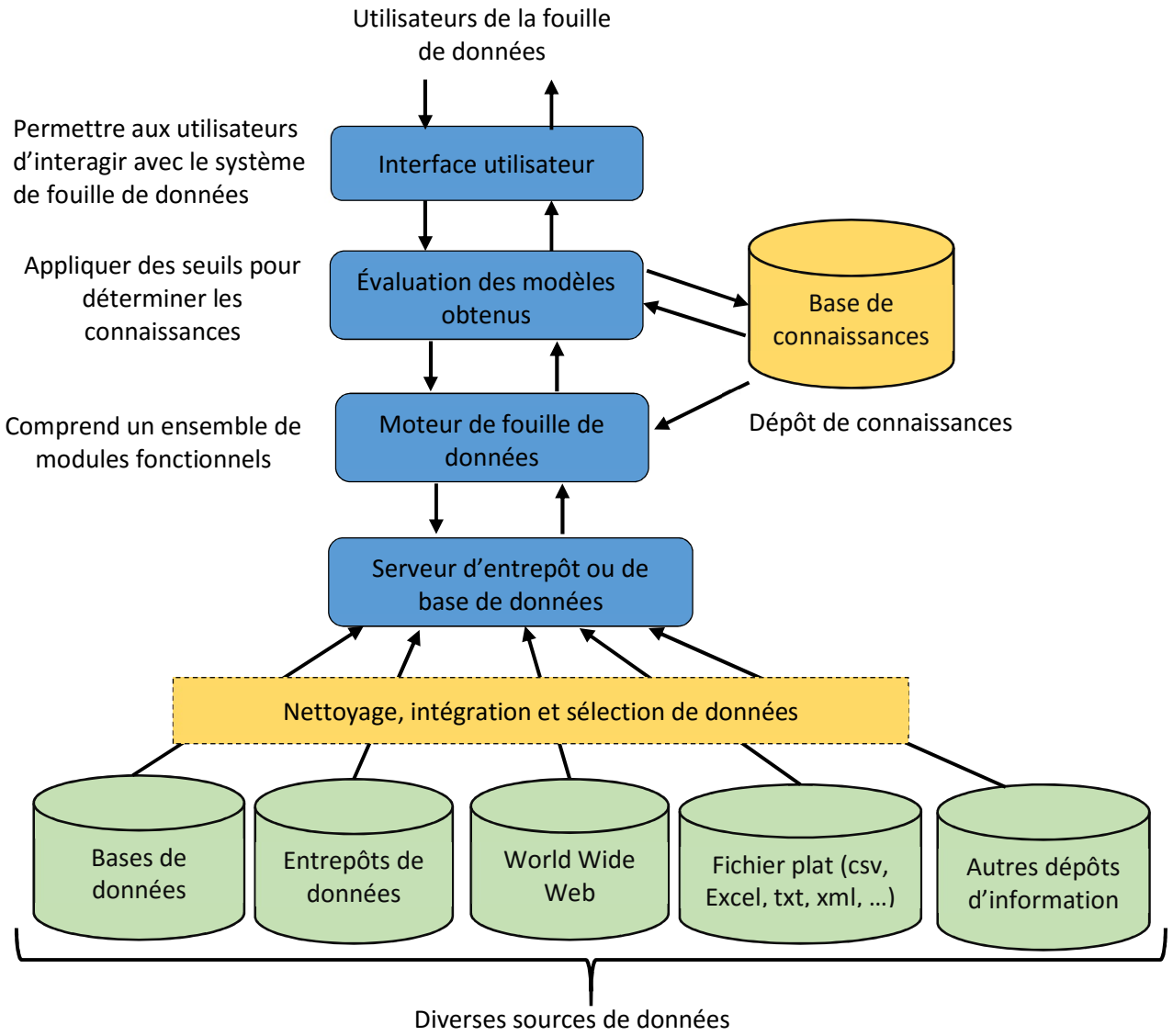


Figure 2.4. Architecture d'un système de fouille de données [3].

Chaque composant du système de fouille de données possède son propre rôle et son importance dans l'exploitation efficace des données. Ces différents modules doivent interagir correctement les uns avec les autres afin de mener à bien le processus complexe de fouille de données. Le séquencement de ces composants ainsi que leurs rôles et leurs modes de fonctionnement sont comme suit :

Les sources de données

Les bases de données, les entrepôts de données, les contenus de Web, les fichiers texte ou d'autres types de documents représentent des exemples réels de sources de données. Généralement, on a besoin de gros volumes de données historiques pour que la fouille de données réussisse.

Le nettoyage, l'intégration et la sélection de données

Les données doivent être nettoyées, intégrées et sélectionnées avant d'être transmises au serveur d'entrepôt ou de base de données. C'est à dire, les données qui proviennent de différentes sources et sous différents formats, ne peuvent pas être utilisées directement pour le processus de fouille de données, parce qu'elles ne sont pas complètes ou fiables. C'est pourquoi ces données doivent être d'abord nettoyées, intégrées et seules les données intéressantes doivent être sélectionnées et transmises au serveur.

Le serveur de base de données ou d'entrepôt de données

Le serveur de base de données ou d'entrepôt de données contient les données réelles qui sont prêtes à être traitées. Par conséquent, le serveur est responsable de la récupération des données pertinentes en fonction de la demande d'exploration de données de l'utilisateur.

Le moteur de fouille de données

C'est le composant central de tout le système de fouille de données. Il consiste en un certain nombre de modules permettant d'effectuer des tâches de fouille de données, notamment la classification, la prédiction, le clustering et les règles d'association.

Le module d'évaluation de modèles

Il est principalement responsable d'évaluer l'intérêt du modèle en utilisant une valeur de seuil. Il interagit avec le moteur de fouille de données pour orienter la recherche vers des motifs (patterns) intéressants.

L'interface utilisateur

Le module d'interface utilisateur communique entre l'utilisateur et le système de fouille de données. Ce module aide l'utilisateur à utiliser le système facilement et efficacement sans connaître la véritable complexité du processus. Lorsque l'utilisateur spécifie une requête ou une tâche, ce module interagit avec le système de fouille de données et affiche le résultat d'une manière facilement compréhensible.

La base de connaissances

La base de connaissances est utile dans tout le processus de fouille de données. Elle est nécessaire pour orienter la recherche et évaluer l'intérêt des modèles de la fouille de données. La base de connaissances peut même contenir les hypothèses des utilisateurs et les données provenant de leurs expériences qui peuvent être utiles dans le processus de fouille de données. Le moteur de fouille de données pourrait obtenir des données de la base de connaissances pour rendre le résultat plus précis et plus fiable. Le module d'évaluation des modèles interagit régulièrement avec la base de connaissances afin d'obtenir des données d'entrée et de les mettre à jour.

2.2.4. Le processus CRISP-DM

De nombreuses organisations dans diverses industries tirent profit de la fouille de données, y compris la fabrication, le marketing, la chimie, l'aérospatiale ; pour augmenter la performance et l'efficacité de leurs activités. Par conséquent, les besoins d'un processus standard de fouille de données ont considérablement augmenté. Un processus de fouille de données doit être fiable et doit pouvoir être exploité par des utilisateurs ayant peu ou pas de connaissances en matière de fouille de données. Par conséquent, en 1990, un processus standard intersectoriel de fouille de données intitulé CRISP-DM (Cross Industry Standard Process for Data Mining) a été publié pour la première fois après être passé par un grand nombre d'ateliers et reçu des contributions de plus de 300 organisations [5, 6].

Le processus CRISP-DM est constitué d'une méthodologie et d'un modèle de processus de fouille de données, qui fournit à quiconque une démarche complète pour la réalisation d'un projet de fouille de données. Selon un sondage publié dans KDnuggets (un site leader dans le business analytics, le big data, la fouille de données, la science de données et l'apprentissage automatique), le processus CRISP-DM est considéré parmi les méthodologies principales de fouille de données [7].

Comme il est montré dans la figure suivante, CRISP-DM divise le cycle de vie d'un projet de fouille de données en six phases [5, 6]. L'ordre des phases n'est pas rigide (le déplacement dans les deux sens entre différentes phases est toujours possible). Il dépend des résultats de chaque phase.

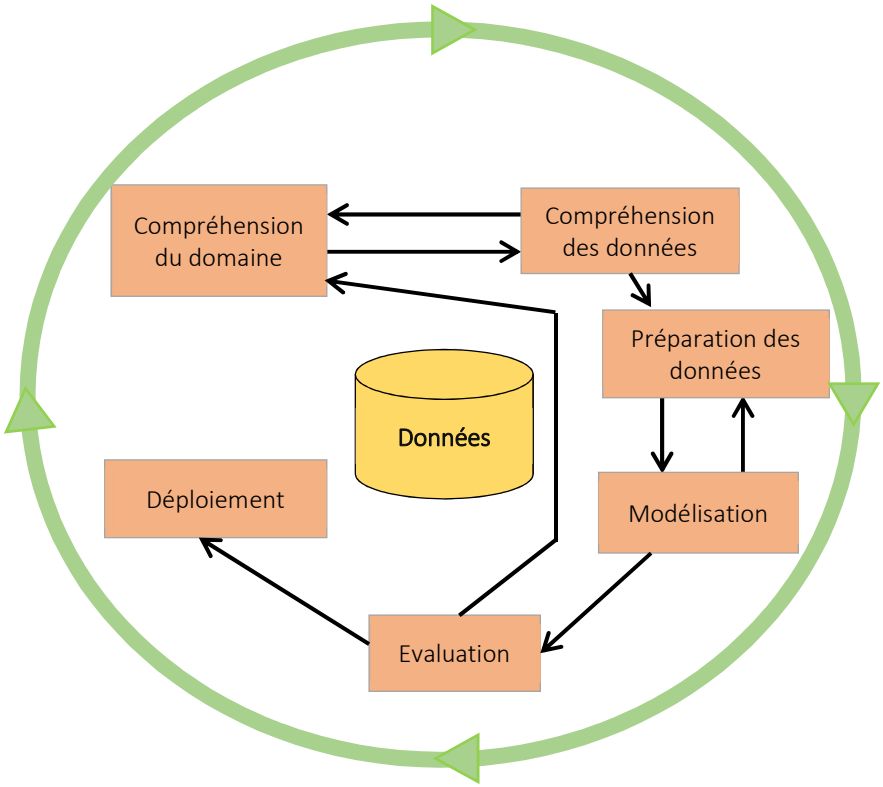


Figure 2.5. Le modèle CRISP-DM [5, 6]

Dans ce qui suit, on va expliquer les phases principales du processus de fouille de données selon le modèle CRISP-DM [8-10] :

a. La compréhension du problème métier

Cette phase initiale se concentre sur la compréhension des objectifs et des exigences du projet du point de vue de l'entreprise, puis sur la conversion de ces informations en une définition du problème de fouille de données et du plan préliminaire conçu pour atteindre les objectifs définis. Parmi ces objectifs, on cite :

- Qu'est-ce que le client veut vraiment accomplir ?
- Quels sont les facteurs importants, les contraintes et les objectifs concurrents, qu'on doit prendre en considération ?

b. La compréhension des données

Cette phase vise à déterminer précisément les données à analyser. Elle commence par une collecte initiale de données et se poursuit par des activités visant à se familiariser avec les données pour identifier les problèmes de qualité des données, relier les données avec leur signification d'un point de vue du métier, découvrir les premiers aperçus des données et

détecter les sous-ensembles intéressants pour former des hypothèses sur les informations cachées. Les actions de cette phase sont ordonnées comme suit :

Recueillir les données : consiste à :

- Énumérer les ensembles de données acquises (lieux, méthodes d'acquisition, problèmes rencontrés et solutions trouvées).
- Recueillir les données à partir des différentes sources de données.

Décrire les données : consiste à :

- Vérifier le volume de données et examiner ses propriétés préliminaires.
- Identifier l'accessibilité, la disponibilité, les types, les plages, les corrélations et la similitude des attributs.
- Identifier la signification de chaque attribut et ses valeurs possibles par rapport au domaine étudié.
- Pour chaque attribut, calculer des statistiques de base (par exemple : la distribution, la moyenne, la valeur max, la valeur min, l'écart-type, la variance, le mode, l'asymétrie).

Explorer les données : consiste à :

- Analyser en détail les propriétés des attributs intéressants.
- Identifier la distribution de chaque attribut, les relations entre paires ou petits nombres d'attributs et les propriétés de sous-populations significatives.

Vérifier la qualité des données : consiste à :

- Identifier les valeurs spéciales et leurs significations. Couvrent-elles tous les cas requis ? Contiennent-elles des erreurs et dans quelle mesure sont-elles courantes ?
- Identifier les attributs manquants, les champs vides et la signification des données manquantes.
- Vérifier la compatibilité entre les significations des attributs et leurs valeurs.
- Vérifier la plausibilité des valeurs, par exemple, certains attributs ont presque les mêmes valeurs.
- Vérifier l'orthographe des valeurs de champs qualitatifs (données de type chaîne de caractères), par exemple, pour des données ayant la même valeur, elles commencent par une lettre minuscule ou une lettre majuscule.

c. La préparation des données

Cette phase de préparation des données regroupe les activités liées à la construction de l'ensemble précis des données à analyser à partir des données brutes initiales. Il est très important de convertir les données brutes en données analytiques, car la qualité des données sélectionnées et nettoyées aura un impact sur la performance du modèle. Les tâches de préparation des données sont susceptibles d'être effectuées plusieurs fois, et non dans un ordre prescrit. Il existe un certain nombre de méthodes utilisées pour le prétraitement, y compris :

- L'échantillonnage qui sélectionne un sous-ensemble représentatif à partir d'une grande population de données.
- La transformation qui modifie ou consolide les données en des formes appropriées à l'exploitation. Les méthodes de transformation des données sont le lissage, la normalisation, l'agrégation, la discrétisation, la construction et la génération hiérarchique de concepts [11].
- Le nettoyage qui supprime le bruit des données.
- La sélection de tables, enregistrements ou attributs, qui sont significatifs dans un contexte particulier.

d. La modélisation

Dans cette phase, diverses techniques de modélisation sont sélectionnées et appliquées (par exemple, des algorithmes d'apprentissage automatique ou des méthodes statistiques), et leurs paramètres sont calibrés à des valeurs optimales. Certains des algorithmes célèbres sont les arbres de décision, les forêts aléatoires, la méthode des k plus proches voisins, la classification naïve bayésienne, la régression linéaire et la régression logistique. Typiquement, il existe plusieurs techniques pour le même type de problème de fouille de données. Certaines techniques ont des exigences spécifiques sur la forme des données, c'est pourquoi il est souvent nécessaire de revenir à la phase de préparation des données.

e. L'évaluation

L'évaluation vise à vérifier les modèles ou les connaissances obtenues afin de s'assurer qu'ils répondent aux objectifs formulés au début du processus. Un des objectifs essentiels est de déterminer s'il existe un problème fonctionnel qui n'a pas été suffisamment pris en compte. À la fin de cette phase, une décision sur l'utilisation des résultats de la fouille de données devrait être prise. Cette phase contribue aussi à la décision de déploiement du modèle ou bien à son amélioration.

f. Le déploiement

Il s'agit de l'étape finale du processus. Son objectif principal est de mettre les connaissances obtenues par la modélisation, sous une forme adaptée, et les intégrer au processus de prise de décision, c'est à dire, les connaissances acquises devront être organisées et présentées d'une manière que le client puisse les utiliser. Cependant, en fonction des besoins, le déploiement peut aller de la simple génération d'un rapport décrivant les connaissances obtenues, jusqu'à la mise en place d'une application permettant l'utilisation du modèle obtenu pour la prédiction de valeurs inconnues d'un élément d'intérêt.

La figure suivante récapitule les différentes phases de procédure de fouille de données, ainsi que les actions élémentaires de chaque phase.

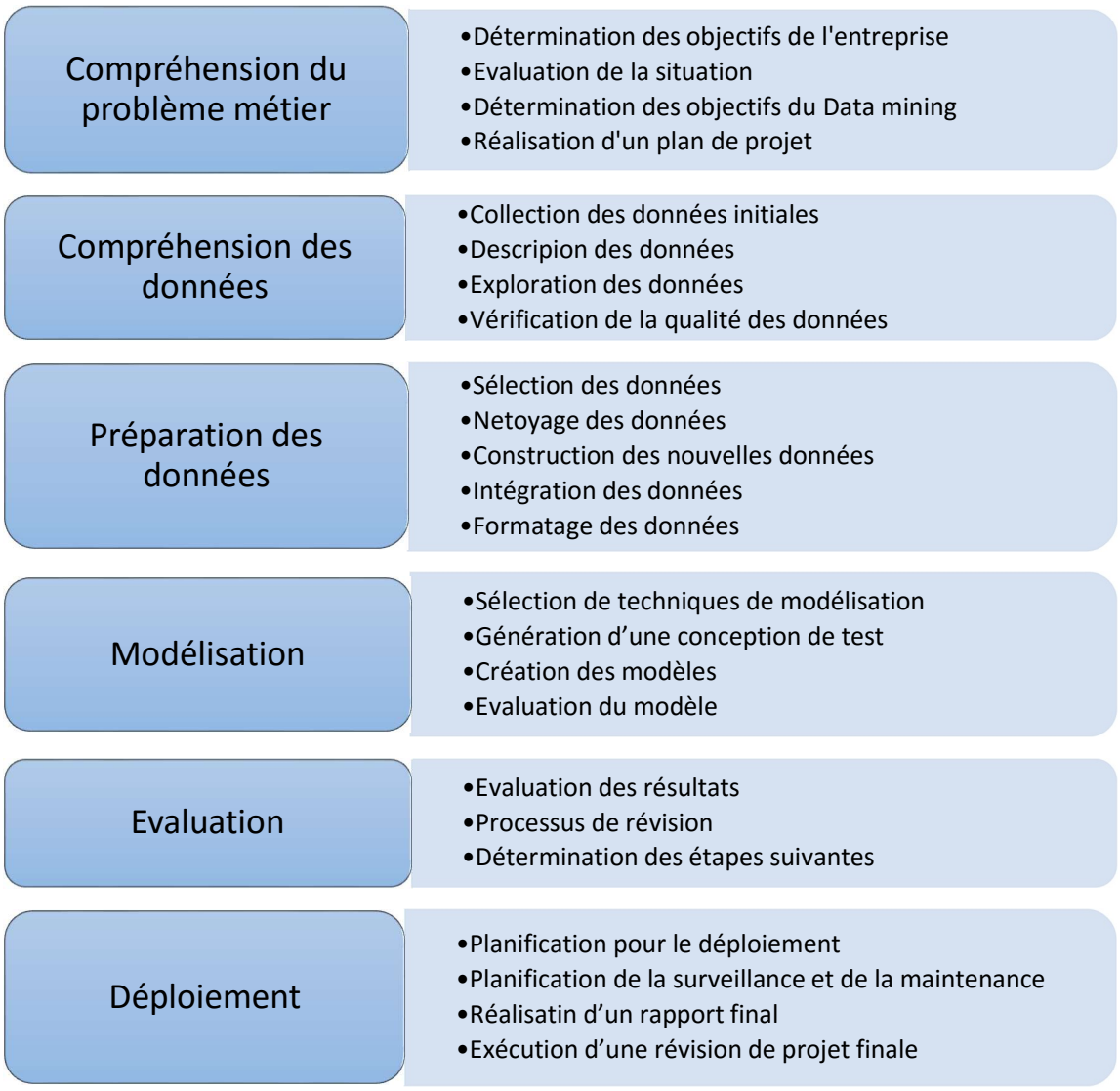


Figure 2.6. Les différentes phases du processus CRISP-DM [6]

2.2.5. Quelques concepts fondamentaux de la fouille de données

Dans ce qui suit, on va présenter quelques concepts fondamentaux utilisés dans le processus de fouille de données. Ces concepts représentent des méthodes comme la réduction et la sélection de données, ainsi que des modèles simples ou étendus comme les forêts aléatoires.

La fouille de données prédictive : elle est généralement utilisée pour identifier un modèle statistique, un réseau de neurones ou un ensemble de modèles pouvant être utilisés pour prédire une réponse intéressante. Par exemple, une société de cartes de crédit peut vouloir s'engager dans la fouille de données prédictive pour dériver un modèle ou un ensemble de modèles (par exemple, réseaux de neurones, arbre de décision) capables d'identifier rapidement les transactions potentiellement frauduleuses [12].

La réduction de données : elle est généralement appliquée aux projets où l'objectif consiste à agréger ou fusionner les données contenues dans des ensembles de données volumineux en des ensembles de données plus petites ou gérables [13]. Les méthodes de réduction des données peuvent inclure le calcul des statistiques descriptives ou des techniques plus sophistiquées comme le clustering et l'analyse des composants principaux.

La sélection de caractéristiques : c'est une des étapes préliminaires du processus de fouille de données applicable lorsque l'ensemble de données comprend plus de variables que ce qui pourrait être inclus (ou serait efficace à inclure) dans la phase de construction du modèle ou même dans les opérations exploratoires initiales.

Le bagging : le concept de bagging désigné « vote » pour la classification et « moyenne » pour la régression. Il s'applique au domaine de la fouille de données prédictive pour combiner les classifications prédites de plusieurs modèles ou du même type de modèle pour différentes données d'apprentissage. Il est également utilisé pour traiter l'instabilité inhérente des résultats lors de l'application de modèles complexes à des ensembles de données relativement petits [14].

Les forêts aléatoires : ce sont des méthodes qui permettent d'obtenir des modèles prédictifs pour la classification et la régression. Elles [15] peuvent être considérées comme une variante de la procédure de bagging, c'est-à-dire une forêt aléatoire est constituée d'un nombre arbitraire d'arbres simples qui permettent de voter pour la classe la plus populaire

(classification), ou dont les réponses sont combinées (moyennées) pour obtenir une estimation de la variable dépendante (régression).

En utilisant des ensembles d'arbres, nous parvenons à améliorer significativement la prévision. L'idée générale de cette méthode est la suivante : au lieu d'essayer d'obtenir une méthode optimisée en une fois, on génère plusieurs prédicteurs avant de mettre en commun leurs différentes prédictions.

Par exemple, les forêts aléatoires peuvent mettre en œuvre des arbres de décision binaire (notamment les arbres CART [16]) avec l'association de la méthode de rééchantillonnage et la randomisation des prédicteurs, c'est-à-dire à chaque itération de la procédure d'apprentissage mise en œuvre pour construire la forêt, un nœud d'un arbre de classification est scindé en deux à partir d'un sous-ensemble des variables du problème sélectionnées de manière aléatoire. Cette étape supplémentaire de randomisation permet d'améliorer les performances du classifieur agrégé. Cependant, elle induit une variabilité supplémentaire, qui vient s'ajouter à l'instabilité de la procédure d'apprentissage.

Le boosting : il consiste à pondérer les observations de l'échantillon d'apprentissage en fonction des performances de la règle de prédiction estimée. Cette méthode procède d'une manière itérative ; c'est-à-dire, un nouveau modèle est estimé à chaque itération et le rééchantillonnage des données d'apprentissage est réalisé en fonction de ses performances ; de telle sorte que la pondération des observations mal classées augmente, tandis que celle des observations bien classées diminue. De la même façon que précédemment, la règle de prédiction agrégée est définie selon un processus de vote à la majorité [17, 18].

Le méta-apprentissage : il s'applique au domaine de la fouille de données prédictive, pour combiner les prédictions de plusieurs modèles, surtout lorsque les types de modèles inclus dans le projet sont très différents [19]. Le plus souvent, les deux approches suivantes sont considérées comme des techniques de méta-apprentissage :

- **La généralisation empilée** [20],où un certain nombre de modèles d'apprentissage de base sont formés à partir du même ensemble de données et dont les résultats sont ensuite utilisés pour un problème d'apprentissage de niveau supérieur ; c'est-à-dire en construisant un modèle reliant les résultats des modèles d'apprentissage de base à la valeur cible.
- **La généralisation en cascade** [21], elle effectue une composition itérative de classificateurs. Dans chaque itération, un classificateur est généré. De plus, l'espace

d'entrée est étendu par l'ajout de nouveaux attributs, où en effectuant à chaque itération une extension des données d'origine par l'insertion de nouveaux attributs.

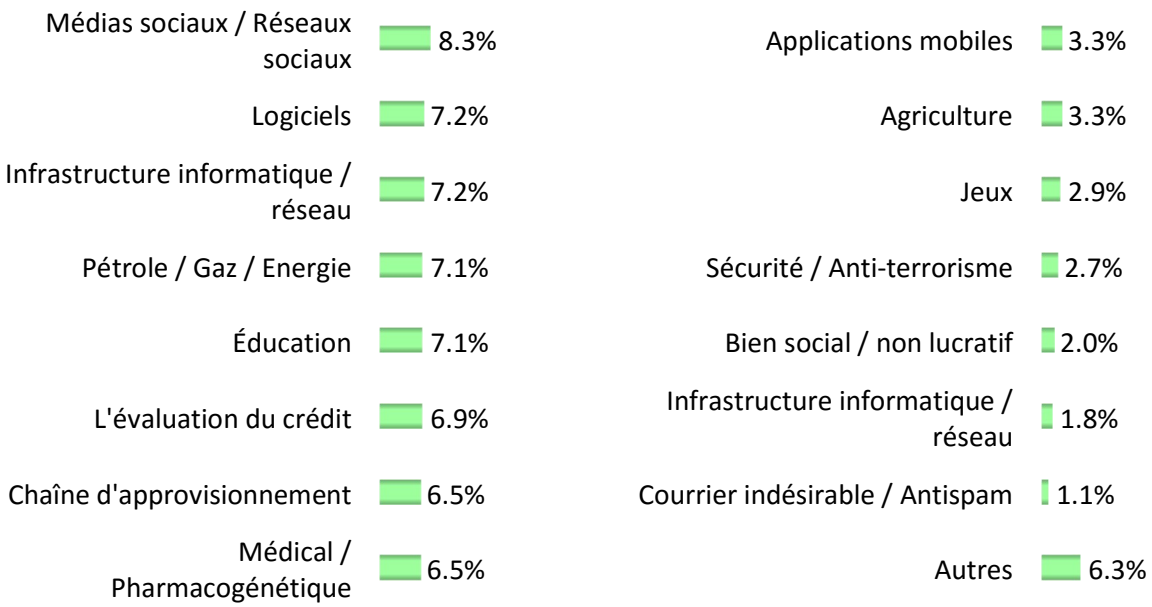
2.2.6. Les principaux domaines d'application de la fouille de données

La fouille de données est un domaine interdisciplinaire avec des applications larges et diverses. Parmi les domaines où la fouille de données est utilisée à grande échelle, il existe la gestion de la relation client CRM (Client Relationship Manager), le domaine bancaire, la publicité et les sciences. De plus, il existe un certain nombre d'organisations qui utilisent déjà régulièrement la fouille de données avec des domaines scientifiques tels que les statistiques, la reconnaissance de formes, et d'autres outils importants [22]. Certaines de ces organisations comprennent les magasins de détail, les hôpitaux, les banques et les compagnies d'assurance.

Un sondage réalisé en 2016 par le site web KDnuggets (spécialisé dans les domaines de la fouille de données, l'apprentissage automatique,le big data, etc.) a demandé aux lecteurs de sélectionner les industries et les domaines dans lesquels ils ont appliqué la fouille de données ou la science de données [22]. Les domaines les plus populaires étaient les suivantes :

Tableau 2.1. Quelques domaines d'application de la fouille de données selon l'année 2016 [22] .

Gestion de la relation client/ Analyse de consommateurs	16.3%	Investissement / Stocks	6.2%
Finance	15.0%	Biotechnologie / Génomique	5.8%
Banque	13.4%	Fabrication	5.6%
Publicité	12.0%	Gouvernement / Militaire	5.6%
Science	12.0%	Recherche / extraction de contenu Web	5.4%
Les soins de santé	12.0%	Automobile / Voitures autonomes	4.5%
Détection de fraude	11.1%	Marketing direct / collecte de fonds	4.3%
Vente au détail	10.3%	Exploitation minière	4.2%
Assurance	9.2%	Tourisme/voyages	4.0%
Commerce électronique	8.9%	Divertissement / Musique / TV / Films	4.0%
Télécommunication	8.3%	Analyse des ressources humaines / effectifs	3.6%



D'après le tableau précédent, parmi les principaux domaines d'application de la fouille de donnée on trouve [22], la gestion de la relation client, l'analyse de comportements consommateurs, la finance et les banques. D'autres domaines ont connu également, une forte croissance en 2016, comme l'antispam, les applications mobiles, le tourisme et voyage. Cependant, l'utilisation de la fouille de données a diminué pendant 2 années consécutives dans les domaines de soins de santé, l'analyse de comportements consommateurs, le marketing direct et la collecte de fonds.

Actuellement, la fouille de données est plus connue auprès de nombreuses entreprises, car elle leur permet d'en savoir plus sur leurs clients et de prendre des décisions marketing intelligentes. [23]. L'industrie a pris conscience de l'importance du patrimoine constitué par ses données et cherche à les explorer en utilisant la fouille de données. Les compagnies les plus avancées dans ce domaine se situent dans le secteur tertiaire. Dans ce qui suit, on va donner un aperçu sur quelques applications de la fouille de données.

La détection d'anomalies

Dans ce contexte, la fouille de données isole et détermine les facteurs qui mènent à la fraude ou l'anomalie. Par exemple, l'entreprise ALERT-KM [24] propose des méthodes pour soutenir l'assurance et le suivi de la surveillance, ainsi que la gestion des connaissances en matière de vérification de la conformité. Ces méthodes impliquent un système de surveillance primaire qui compare certaines conditions d'acceptation prédéterminées avec les données ou les événements réels. Si une anomalie est détectée par le système de surveillance primaire, un rapport d'exception ou une alerte est produit, identifiant l'écart.

La détection d'intrusion et la prévention

Le domaine de fouille de données aide à construire des systèmes pour surveiller et sécuriser les systèmes d'information, ces systèmes de sécurité déclenchent des alarmes lorsque des violations de sécurité sont détectées. Il existe plusieurs types de systèmes de détection d'intrusion (IDS), allant des logiciels antivirus aux systèmes hiérarchiques qui surveillent le trafic d'un réseau entier. Les classifications les plus courantes sont les systèmes de détection d'intrusion réseau (NIDS) et les systèmes de détection d'intrusion hôte (HIDS). Il est également possible de classer les IDS par approche de détection, à savoir la détection basée sur des signatures (la reconnaissance des mauvais modèles tels que les logiciels malveillants) et la détection des anomalies qui repose souvent sur l'apprentissage automatique. Les systèmes qui ont des capacités de répondre aux intrusions détectées sont appelés des systèmes de prévention d'intrusion [13].

Le marketing et la vente au détail

Les techniques de fouille de données peuvent aider par exemple à déterminer quels sont les articles susceptibles d'être achetés ensemble par l'analyse des règles d'association, dans le but d'identifier les opportunités de ventes croisées [25].

De plus, la fouille de données pourrait aider les spécialistes du marketing en leur fournissant des tendances utiles et précises sur les comportements d'achat de leurs clients et les aider à prévoir quels produits leurs clients pourraient être intéressés à acheter. Elle permet aux entreprises d'identifier leurs meilleurs clients, d'attirer des clients, de connaître les clients via le marketing par courrier électronique et de maximiser la rentabilité par l'identification de clients rentables.

D'autre part, la fouille de données peut être utilisée pour regrouper les clients en fonction de leurs comportements (comme l'historique des paiements), ce qui aide à la gestion des relations clients et le marketing ciblé. Habituellement, il devient utile de définir des clients similaires dans un cluster, de conserver de bons clients, d'éliminer les mauvais clients et d'identifier les répondants potentiels pour les promotions commerciales.

La médecine et la santé

Dans les domaines de la médecine et de la santé, l'utilisation des techniques de la fouille de données nous permet d'avoir plus de connaissances sur les relations entre les maladies et

l'efficacité des traitements, les nouveaux médicaments et les services de livraison des médicaments [26, 27].

De plus, les entreprises pharmaceutiques peuvent analyser leurs ventes, les informations sur les systèmes de soins de santé locaux et les données sur l'activité des concurrents sur le marché, pour mieux cibler les médecins et les malades de grande valeur, déterminer quelles activités de marketing auront le plus grand impact dans les prochains mois et appliquer les meilleures pratiques dans une situation de vente spécifique [28].

La télécommunication

Actuellement, l'industrie de télécommunication est en pleine expansion. Elle est très compétitive et a une forte demande pour la fouille de données [29]. Par exemple pour :

- Comprendre l'entreprise concernée et identifier les schémas de télécommunications.
- Utiliser au mieux les ressources et améliorer la qualité de service.
- Analyser les données multidimensionnelles (temps d'appel, durée, localisation de l'appelant, localisation de l'appelé, type d'appel, etc.).
- Identifier les utilisateurs frauduleux.
- Trouver des modèles d'utilisation pour un ensemble de services de communication par groupe de clients, par mois, etc.

Le commerce électronique

C'est aussi le domaine le plus prometteur pour la fouille de données. Il est idéal parce que la plupart des ingrédients nécessaires au succès de la fouille de données sont facilement disponibles, par exemple [13, 30] :

- Les enregistrements de données sont nombreux.
- La collecte électronique fournit des données fiables.
- Les informations atteintes peuvent être facilement transformées en actions et le retour sur investissement peut être mesuré.
- La possibilité d'intégration du commerce électronique et de la fouille de données, ce qui améliore considérablement la génération de connaissances et la prise de décisions commerciales correctes.

L'enseignement et l'éducation

Un défi important auquel l'enseignement supérieur est confronté aujourd'hui est de prédire les trajectoires des étudiants et des anciens étudiants. Quel étudiant va s'inscrire à des programmes de cours particuliers ? Qui aura besoin d'une aide supplémentaire pour obtenir son diplôme ? La gestion des inscriptions et le temps nécessaire pour obtenir un diplôme ? Ces issues et d'autres continuent d'exercer des pressions sur les collèges pour qu'ils recherchent de nouvelles solutions plus rapides et elles peuvent être traitées en utilisant les techniques de fouille de données [31].

De plus, les établissements d'éducation peuvent enseigner et instruire mieux leurs étudiants grâce à l'analyse et à la présentation de données [32]. La fouille de données est rapidement apparue comme un outil hautement souhaitable pour utiliser les capacités actuelles de génération de rapports afin de découvrir et de comprendre les modèles cachés dans de vastes bases de données. Par exemple, en astronomie, où les nouveaux télescopes fournissent des images très détaillées de l'univers, des techniques de fouille de données sont utilisées pour la classification automatique des corps célestes nouvellement découverts.

La sécurité et la détection criminelle

Les agences de renseignement recueillent et analysent des renseignements pour enquêter sur les activités terroristes [33, 34]. L'un des défis auxquels sont confrontés les organismes d'application de la loi et les agences de sécurité est la difficulté d'analyser un grand volume de données impliquées dans des activités criminelles et terroristes. Aujourd'hui, certaines agences de renseignement utilisent des algorithmes de fouille de données sophistiqués, ce qui facilite le traitement de très grandes bases de données. Par exemple, on utilise les techniques de clustering et de règles d'association pour découvrir les liaisons entre les différents objets (comme les personnes, les organisations, les véhicules, etc.) dans les dossiers criminels. La technique de classification est également utilisée pour détecter les spams (les courriels indésirables) et trouver les personnes qui ont données ces courriers. Le comparateur de textes est utilisé pour détecter les informations trompeuses dans le casier judiciaire.

2.2.7. Tendances de la fouille de données

Comme plusieurs types de données (texte, multimédia, biologique …) sont disponibles pour les tâches de fouille de données, la fouille de ces types complexes de données pose des problèmes supplémentaires. Ainsi, nous allons discuter certaines tendances de la fouille de données qui reflètent la poursuite de ces défis :

La fouille de texte

La fouille de données textuelles est le processus de génération d'informations de haute qualité qui fait généralement référence à une combinaison de pertinence, de nouveauté et d'intérêt à partir d'un ensemble de documents textuels. L'information de haute qualité est typiquement dérivée par la conception de modèles et de tendances à travers des moyens tels que l'apprentissage de modèles statistiques [35, 36]. Les tâches typiques de fouille de données textuelles comprennent la catégorisation de texte, l'extraction de concept et l'analyse des opinions des individus. Généralement, le processus de fouille de données textuelles implique :

- La structuration du texte d'entrée (l'analyse de texte, l'addition de certaines caractéristiques, le retrait de certaines caractéristiques et l'insertion de données structurées dans une base de données).
- La dérivation des modèles à partir des données structurées.
- L'évaluation et l'interprétation de la sortie.

La fouille de web

Une extension de la fouille de données textuelles est la fouille de données web qui est l'application de techniques de fouille de données pour découvrir des modèles à partir du web [37]. Comme son nom l'indique, il s'agit d'informations collectées en exploitant le Web. La fouille de web se base sur les données des serveurs web, et surtout les informations organisées et non structurées à partir des activités du navigateur web, des logs du serveur, du contenu de sites web, des structures des sites web et des liens entre leurs éléments.

Habituellement, on trouve trois types de fouilles de web, à savoir la fouille de contenu web, la fouille de structure web et la fouille d'utilisation web [13]. Parmi les applications de fouille de web, on peut citer le regroupement des messages d'actualités de plusieurs fournisseurs d'informations en fonction du sujet, la synthèse automatique de textes et la détection de spams dans les messages électroniques. De plus, les résultats des moteurs de recherche dépendent fortement de la fouille de web et du traitement du langage naturel.

La fouille de données multimédia

Elle consiste à extraire des données de différents types de sources multimédias telles que l'audio, le texte, la vidéo, l'image. Ensuite, ces données sont converties en une représentation numérique sous différents formats analysables, pour être explorées à travers les différentes tâches de fouille de données comme la classification, le clustering ou l'identification de règles d'association.

La fouille de données ubiquitaire

Cette méthode implique la fouille de données à partir d'appareils mobiles, embarqués et ubiquitaires [38]. Bien qu'il y a plusieurs défis de ce type de fouille, tels que la complexité de données, la vie privée et le coût, cette méthode a beaucoup d'opportunités dans divers domaines en particulier dans les réseaux de transport, les soins de santé, les réseaux de capteurs électriques et la gestion de crise [39].

La fouille de données distribuées

Ce type de fouille de données implique une grande quantité de données brutes stockées dans des lieux ou des dispositifs de stockage différents. Les méthodes traditionnelles de fouille de données, conçues pour fonctionner à un endroit centralisé, ne fonctionnent pas bien dans la plupart des environnements informatiques distribués (par exemple : l'internet, les réseaux locaux, les réseaux sans fil et les réseaux de capteurs). Plusieurs algorithmes de fouille de données distribuées sont établis et utilisés pour analyser des données provenant de différents endroits et fournir des aperçus et des rapports appropriés basés sur eux [40, 41].

La fouille de données biologiques

Bien que la fouille de données biologiques puisse être considérée comme une fouille de données de type complexe, la combinaison unique de complexité, de richesse, de taille et d'importance des données biologiques mérite une attention particulière dans la fouille de données. Les applications de la fouille de données à la bio-informatique comprennent l'optimisation du traitement et du diagnostic des maladies à travers la détection de motifs fonctionnels, la détection de domaines de fonctions protéiques, l'inférence de fonctions protéiques, la reconstruction de réseaux d'interactions protéiques et génétiques, et la prédiction de l'emplacement sous-cellulaire des protéines [42].

La fouille de données spatiales et géographiques

Les données géographiques volumineuses et complexes sont collectées à l'aide de techniques modernes d'acquisition de données telles que les systèmes de positionnement global (GPS), les systèmes de télédétection, les services de géolocalisation et l'information géographique volontaire. Ce type de données aboutit également à l'établissement d'un nouveau type de fouille de données qui implique le développement de théories, méthodes et pratiques pour l'extraction d'informations utiles, inconnues et inattendues, à partir de données environnementales, géographiques et extra-atmosphériques. Ce type de fouille de données peut utiliser divers aspects tels que la distance, la topologie et le temps, qui sont principalement utilisés dans les systèmes d'information géographique et les applications de navigation et de localisation. De plus, la fouille de données spatiales et géographiques offre de nouveaux avantages et améliorations à ces systèmes surtout dans l'analyse de ce type de données [43].

Parmi les recherches théoriques et appliquées récentes dans ce domaine, on trouve celles concernant la classification et la prévision spatiales, les règles d'association spatiale, le clustering spatial et la géovisualisation. Ainsi que le développement de nouvelles techniques pour l'analyse de formes ponctuelles, la prédiction dans des données spatio-temporelles, l'analyse de données d'objets en mouvement et l'optimisation dans le contexte de la classification d'images et de l'interpolation spatiale [43, 44] .

La fouille de graphes

L'objectif principal de la fouille de graphes est de fournir de nouveaux principes et des algorithmes efficaces pour explorer les sous-structures topologiques intégrées dans les données de graphes [13]. Il existe un large éventail d'applications de fouille de graphes sur le web, les réseaux d'information, la bio-informatique, l'informatique chimique, la vision par ordinateur et le multimédia. Par conséquent, la fouille de graphes a fait l'objet de nombreuses recherches approfondies comme l'extraction de motifs de graphes, la modélisation statistique des réseaux d'information, le clustering et la classification des graphes et l'analyse de l'évolution des réseaux d'information [13].

2.2.8. Quelques défis de la fouille de données

La recherche en fouille de données est très ponctuelle et les données ne sont pas toujours de type simple ou sont disponibles en un seul endroit. Ces facteurs et d'autres génèrent plusieurs défis dans le processus de fouille de données. Ces défis peuvent être classés comme suit [45, 46] :

a. La méthodologie de la fouille de données et l'interaction avec l'utilisateur

Cette catégorie de défis prend en compte les types de questions suivantes :

L'extraction de différents types de connaissances

Différents utilisateurs peuvent être intéressés par différents types de connaissances (par exemple, la prévision, la classification et la détection d'anomalie). Par conséquent, il est nécessaire que la fouille de données couvre un large éventail de tâches de découverte de connaissances.

La fouille interactive de données à plusieurs niveaux d'abstraction

Le processus de fouille de données doit être interactif, parce qu'il permet aux utilisateurs de cibler la recherche de modèles, en fournissant et en affinant les requêtes de fouille de données en fonction des résultats obtenus.

L'incorporation des connaissances préalables

Les connaissances préalables peuvent être utilisées brièvement et à de multiples niveaux d'abstraction pour orienter le processus de fouille de données et exprimer ses modèles découverts,

Le langage de requête et la fouille de données ad hoc

Des langages de requête de haut niveau doivent être développés pour permettre aux utilisateurs de décrire des tâches ad hoc de fouille de données, en facilitant la spécification : des ensembles de données pertinents pour l'analyse, la connaissance du domaine, les types de connaissances à extraire et les contraintes à appliquer aux motifs découverts. Un tel langage devrait être intégré à une base de données ou à un langage d'interrogation d'entrepôt de données. Également, ce langage de requête doit être optimisé pour une fouille de données efficace et flexible.

La visualisation des résultats de la fouille de données

Une fois que les modèles sont découverts, ils doivent être exprimés dans des langages de haut niveau et des représentations visuelles qui doivent être facilement compréhensibles.

Le traitement de données bruyantes ou incomplètes

Les méthodes de nettoyage de données sont nécessaires pour traiter les données bruyantes ou incomplètes lors de l'extraction de régularités et de connaissances à partir de données. Ces méthodes de nettoyage des données sont indispensables pour que la précision des modèles découverts ne soit pas faible.

L'évaluation de modèles découverts

Beaucoup de modèles découverts peuvent être inintéressants pour un utilisateur donné (par exemple, ils représentent des connaissances communes ou ne contiennent pas de nouveauté). Par conséquent, il faut avoir des techniques et des mesures pour évaluer l'intérêt des modèles découverts, en particulier les mesures subjectives qui évaluent la valeur des modèles par rapport aux attentes d'une classe d'utilisateurs donnée, ainsi que les mesures qui permettent de guider le processus de découverte et réduire l'espace de recherche.

b. Les problèmes de performance

Il peut y avoir des problèmes liés à la performance de processus de fouille de données tels que :

L'efficacité et l'évolutivité des algorithmes de fouille de données

Pour extraire efficacement les informations d'une énorme quantité de données dans des bases de données, l'algorithme de fouille de données doit être efficace et évolutif. De plus, ce facteur doit être pris en compte par bon nombre des questions abordées précédemment dans le cadre de la méthodologie d'exploration de données et de l'interaction avec les utilisateurs.

Les algorithmes de fouille de données parallèles, distribués et incrémentaux

La taille énorme de nombreuses bases de données et la large distribution des données sont des facteurs qui motivent le développement d'algorithmes de fouille de données parallèles et distribués. Ces algorithmes divisent les données en partitions, qui sont traitées en parallèle. Les résultats des partitions sont ensuite fusionnés. De plus, le coût élevé et la complexité de calcul de certains processus de fouille de données, ainsi que l'instabilité de certaines bases de données nécessitent l'utilisation d'algorithmes incrémentiels qui prennent en compte les mises

à jour de base de données sans avoir à explorer à nouveau l'intégralité des données à partir de zéro.

c. La diversité de types de données

Le traitement des types de données relationnelles et complexes

Les bases de données peuvent contenir des objets de données complexes, tels que des données multimédias, des données spatiales, des données temporelles et des données transactionnelles. Étant donné cette diversité des types de données et celle des objectifs de la fouille de données, il n'est pas possible pour un seul système de fouille de données d'exploiter tous ces types de données. Donc, des systèmes de fouille de données spécifiques devraient être construits en fonction de types de données.

L'extraction d'informations à partir de bases de données hétérogènes et de systèmes d'information globaux

Les réseaux informatiques (locaux ou étendus) qui relient de nombreuses sources de données constituent des bases de données énormes, réparties et hétérogènes. L'extraction de connaissances provenant de différentes sources de données (structurées, semi-structurées ou non structurées) avec des sémantiques diverses, pose de grands défis à la fouille de données. Par exemple, la fouille de Web, la fouille de données multisources et la fouille de réseaux d'information sont devenues des domaines de fouille de données prometteurs et en évolution rapide. Par conséquent, ce type de fouille de données doit aider à révéler des informations de haut niveau (relations, tendances ou motifs) dans de multiples bases de données hétérogènes, en améliorant l'échange de données et l'interopérabilité des bases de données hétérogènes.

d. Problèmes de confidentialité

Compte tenu de l'usage croissant de la fouille de données dans notre vie quotidienne, il est important d'étudier l'impact de la fouille de données sur la société. Comment pouvons-nous utiliser la fouille de données au profit de la société et éviter son utilisation abusive ? La divulgation des informations personnelles ou l'utilisation inappropriée des données et la violation potentielle des droits individuels et de la vie privée sont des sujets de préoccupation qui doivent être traités. Par exemple, les modèles découverts peuvent être utilisés pour deviner des propriétés confidentielles, mais ils peuvent aussi conduire à des stéréotypes et des préjugés. Ce problème peut être très sensible et controversé, si les modèles sont basés sur des propriétés telles que la race, le sexe et la nationalité. Des études sur la fouille de données visant à préserver la vie privée sont en cours. La philosophie consiste à observer la sensibilité des

données et à préserver la confidentialité des personnes tout en effectuant une fouille de données réussie.

3.3. La fouille de données dynamique

2.3.1. Motivations de la fouille de données dynamique

Malgré les vastes applications et méthodes de la fouille de données, on peut rencontrer d'autres problèmes concernant les données, comme la multidimensionnalité, la confidentialité et la diversité. Étant donné que les problèmes principaux traités dans cette thèse sont liés principalement au domaine de la fouille de données dynamique, on va présenter dans ce qui suit ses différentes définitions, ses composants de bases ainsi que certaines techniques et applications de ce domaine étendu et complexe.

En considérant, comme exemple, un système de détection de fraude par carte de crédit. Dans ce système, nous pourrions segmenter nos clients en groupes homogènes, explorer les flux de données générés par leurs transactions et classer ces clients en fonction de leur probabilité de fraude.

Étant donné que l'ensemble de données se modifie avec le temps, le modèle de détection de fraude doit être aussi évolué avec le temps. De plus, si le clustering regroupe les clients en utilisant non seulement leurs caractéristiques actuelles, mais aussi leurs séries temporelles, les choses deviennent encore plus difficiles, car nous devons regrouper les vecteurs de fonctions au lieu de vecteurs de valeurs numériques. Un exemple d'une telle série temporelle pourrait être l'évolution du nombre de transactions de nos clients au cours des six derniers mois. Une telle évolution nous en dit plus sur leur comportement qu'un simple attribut unique comme le nombre le plus récent de transactions effectuées par un tel client. C'est dans ces types d'applications que la fouille de données dynamique peut être appliquée.

Nous rappelons à nouveau que l'ensemble du processus ECD (l'extraction de connaissances à partir de données) consiste essentiellement en des activités réalisées avant la fouille de données (telles que la sélection, le prétraitement, la transformation de données [47]), la partie de fouille de données proprement dite et les étapes ultérieures (telles que l'interprétation, l'évaluation et l'utilisation des résultats). Étant donné que la fouille de données n'est qu'une étape de processus ECD[4], la prise en compte de données dynamiques comme les données temporelles et les flux de données doivent également être pris en compte lors des autres étapes.

L'obtention de l'ensemble de données nécessaires à la fouille de données est souvent fastidieuse, coûteuse et prend beaucoup de temps. Également, les approches de fouille de données qui implique la construction de modèle de fouille de données à partir de zéro entraînent la perte de toutes les informations intelligentes recueillies précédemment. En outre, la combinaison de données historiques et de nouvelles données n'est pas toujours possible si les données historiques sont supprimées, perdues ou indisponibles.

Par conséquent, face à ces problèmes, la fouille de données dynamique est apparue comme un nouveau type de fouille de données, dans laquelle de nouveaux concepts, de nouvelles méthodes et de nouveaux modèles sont développés, afin de résoudre les différents types de problèmes cités précédemment.

2.3.2. Le processus de fouille de données dynamique

Habituellement, plusieurs bases de données ont tendance à se modifier dynamiquement, par exemple le contenu de site web, les données transactionnelles des magasins, les données environnementales et les données de surveillances médicales. Dans ce cas, de nouvelles données peuvent être insérées et les données actuelles peuvent être mises à jour ou supprimées. Parmi les définitions de la fouille de données dynamique que l'on puisse rencontrer, il existe :

a. La fouille de données dynamique est un processus qui consiste à mettre à jour dynamiquement les connaissances acquises lors d'un processus de fouille de données préliminaire [48].

b. Une autre définition considère la fouille de données dynamique comme étant un type de fouille de données qui traite des aspects dynamiques dans n'importe quelle étape du processus de fouille de données conventionnelles [49].

c. D'après [50], soit t représente le temps actuel. L'ensemble de données produites avant t est un ensemble de données historiques ou anciennes, l'ensemble de données d'une période de temps avant t à t est appelé l'ensemble de données courantes et l'ensemble de données générées après t est appelé l'ensemble de données nouvelles. Le processus d'utilisation de l'ensemble de données historiques, courantes et nouvelles pour extraire de nouvelles connaissances et règles est appelé fouille de données dynamique.

Un schéma du processus de fouille de données dynamique est représenté par la figure suivante. Cette représentation reflète les définitions mentionnées précédemment, où le processus de fouille de données dynamique est constitué de la fouille de données conventionnelle, des données dynamiques et des procédures nécessaires pour les traiter.

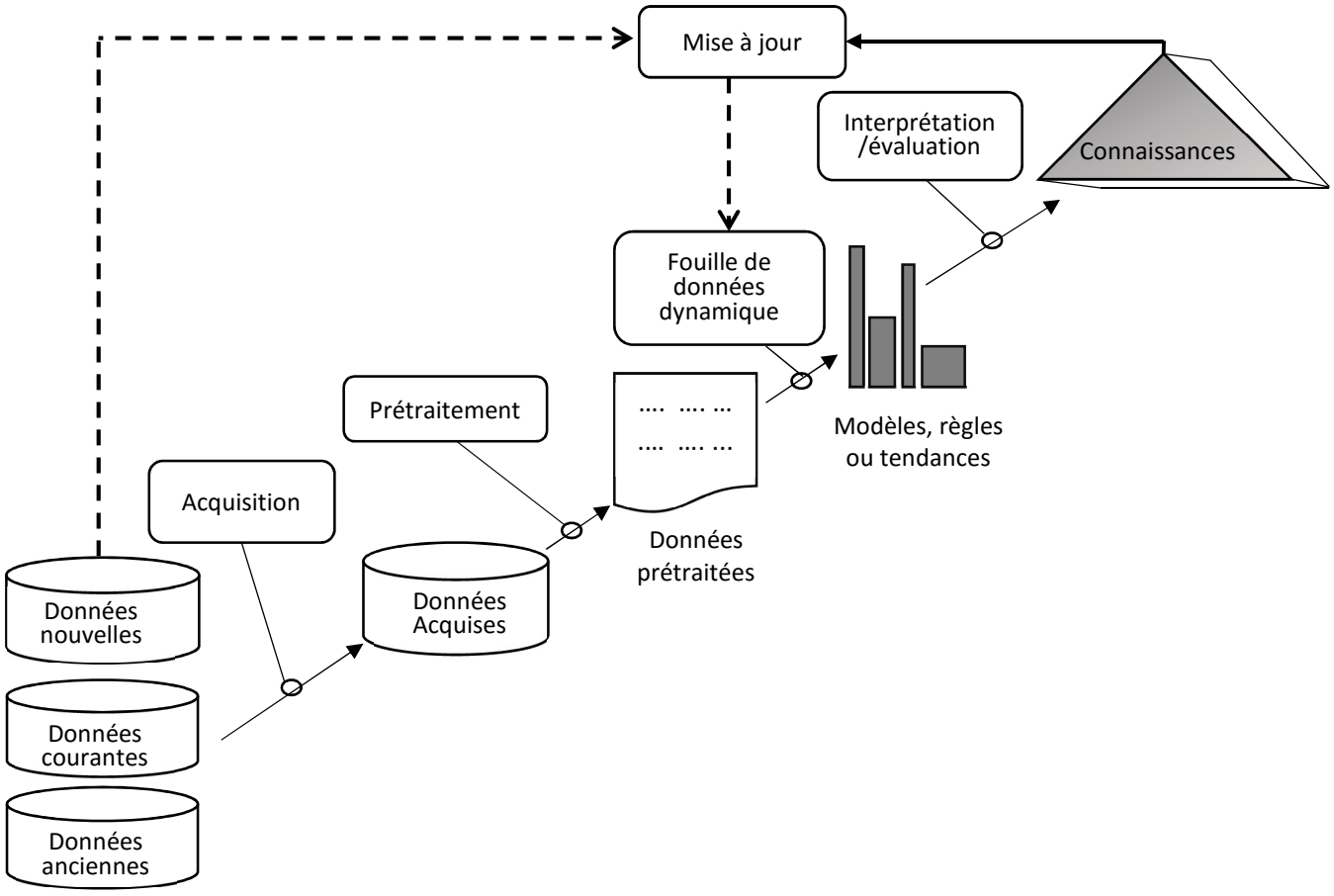


Figure 2.7. Processus de fouille de données dynamique, inspiré de celui de la fouille de données conventionnelles [4]

2.3.3. Les données temporelles et les flux de données

Dans ce contexte, on peut distinguer deux classes principales de données, qui peuvent constituer les données préliminaires à traiter lors d'une fouille de données dynamique, à savoir, les données temporelles (Temporelles Data) et les flux de données (Data Stream). Dans ce qui suit, on va expliquer ces deux classes d'objets avec plus de détail.

2.3.3.1. Les données temporelles

Les données temporelles peuvent être divisées en trois types :

a. Les séquences temporelles

Une séquence temporelle consiste à une séquence ordonnée d'événements ou de transactions [50] [51, 52] horodatées à des intervalles de temps réguliers ou irréguliers. Un exemple de séquence temporelle est la séquence horodatée des achats d'un client sur un site Web.

De plus, dans de nombreuses applications de base de données temporelles ou temps réel, l'état d'un objet de données *o* est modifié à des points de temps discrets, formant une séquence temporelle $S_i = (t_i, v_i)$, où v_i est la valeur de l'état de l'objet *o* au temps t_i [53].

Il y a trois caractéristiques de base de séquences de temps, à savoir [53]:

a. Les séquences temporelles sont ordonnées, c'est-à-dire : $\forall i, j : i > j \rightarrow t_i > t_j$
b. Chaque valeur v_i dépend fonctionnellement du temps t_i, mais l'inverse n'est pas vrai (par exemple, un objet ne peut pas avoir deux états différents en même temps).
c. La valeur v_i peut être unidimensionnelle (par exemple, des températures ou des tensions), bidimensionnelle (par exemple, des positions dans un plan) ou elle a un nombre de dimensions supérieur à 2 (par exemple des positions dans l'espace).

Plusieurs types de requêtes de base sur les séquences temporelles peuvent être identifiés. Cependant, des requêtes complexes peuvent être composées en combinant ces requêtes de base comme le cas dans la fouille de données dynamique. Parmi les requêtes de base, on peut citer :

- Quelle était la valeur au temps t'?
- Quelle était la plage de valeurs dans l'intervalle de temps [t', t''] ?
- À quel moment la valeur était-elle égale à une valeur v' ?
- Dans quel intervalle de temps [t', t''] la valeur était-elle plus grande (plus petite) que v'?

b. Les séries temporelles

Une série temporelle est un objet de données temporel [52] consistant en une séquence de données numériques représentant une liste ordonnée d'événements ou une collection d'observations effectuées chronologiquement et enregistrées à des intervalles de temps égaux (par exemple, par minute, par heure ou par jour). La nature des données des séries temporelles comprend la taille importante des données, leur grande dimensionnalité, la nécessité de les

mettre à jour en permanence et leur nature numérique et continue. Elles sont considérées comme un tout plutôt que comme un champ numérique individuel.

Les données des séries temporelles ont une distance uniforme sur l'échelle de temps. Ils représentent des mesures ordonnées en valeur réelle à des intervalles réguliers dans le temps. Une série temporelle $X = \{x_1, x_2, ..., x_n\}$ pour $t = t_1, t_2, ..., t_n$ est une fonction discrète avec une valeur x_1 pour le temps t_1, x_2 pour le temps t_2, et ainsi de suite. Les données des séries temporelles se composent de fréquences variables et la présence de bruit est également un phénomène courant. Une série temporelle peut être multivariée ou univariée. Une série temporelle multivariée est constituée de plus d'une variable, alors que dans une série temporelle univariée, il existe une seule variable sous-jacente. De plus, les séries temporelles peuvent être stationnaires ou non stationnaires. Une série temporelle stationnaire a une moyenne et une variance qui ne changent pas avec le temps, tandis qu'une série non stationnaire n'a pas de moyenne stable et peut diminuer ou augmenter avec le temps [51].

Une série chronologique est dite continue lorsque les observations sont effectuées de manière continue dans le temps (par exemple, les relevés de température, le débit d'une rivière et la concentration d'un processus chimique). L'adjectif « continu » est utilisé pour les séries de ce type, même lorsque la grandeur mesurée ne peut prendre qu'un ensemble discret de valeurs. Un type spécial de séries temporelles contenues (processus binaires) survient lorsque les observations peuvent prendre une des deux valeurs, habituellement notées 0 et 1 (voir la figure 2.8). Par exemple, en informatique, les positions d'un commutateur « activer » et « désactiver » pourraient être enregistrées comme étant 1 et 0, respectivement.

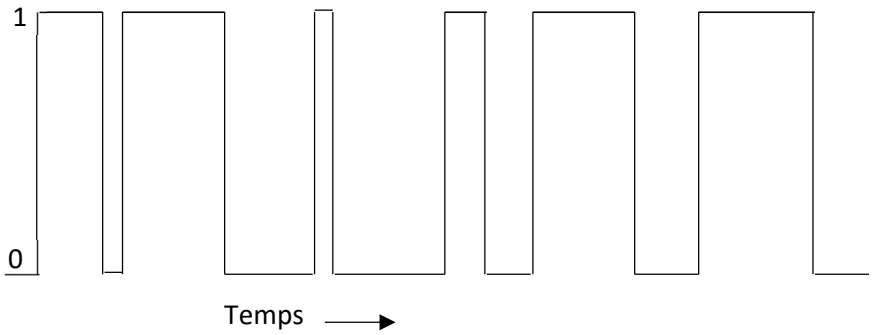


Figure 2.8. Représentation d'un processus binaire.

On dit qu'une série temporelle est discrète lorsque les observations ne sont prises qu'à des moments précis, habituellement à des intervalles égaux, par exemple, le recensement

quinquennal des habitants d'une ville, le taux mensuel de production d'une entreprise et le taux quotidien entre deux devises différentes. Le terme « discret » est utilisé pour les séries de ce type, même lorsque la grandeur mesurée est une variable continue.

Une série temporelle peut être décomposée en quatre composantes [54, 55], chacune exprimant un aspect particulier de la variation des valeurs de la série temporelle. Ces quatre composantes sont :

a. La tendance à long terme : elle indique la direction générale dans laquelle un graphique de séries temporelles évolue dans le temps, par exemple, en utilisant la moyenne mobile pondérée et les méthodes des moindres carrés pour trouver des courbes de tendance.
b. Les variations saisonnières : elles représentent des modèles presque identiques, où la série temporelle semble les suivre au cours des saisons qui correspondent des périodes uniformes successives, comme les saisons de magasinage des fêtes durant l'année.
c. Les variations cycliques : ce sont les oscillations à long terme autour d'une ligne de tendance ou d'une courbe. Ils correspondent à des variations périodiques, mais non saisonnières.
d. Les variations aléatoires : elles caractérisent les changements irréguliers dus à des événements inattendus tels que les conflits de travail et les changements de personnel au sein des entreprises.

L'analyse d'une série temporelle consiste à faire une description mathématique des éléments qui la composent, c'est-à-dire à estimer séparément les quatre composantes. Compte tenu des effets de ces quatre composantes, deux types de modèles différents sont généralement utilisés pour une série temporelle, à savoir les modèles multiplicatifs et les modèles additifs [56].

- Modèle multiplicatif : $Y(t) = T(t) \times S(t) \times C(t) \times I(t)$
- Modèle additif : $Y(t) = T(t) + S(t) + C(t) + I(t)$

Ici $Y(t)$ est l'observation et $T(t)$, $S(t)$, $C(t)$ et $I(t)$ sont respectivement la variation de tendance, la variation saisonnière, la variation cyclique et la variation irrégulière.

Les données temporelles sémantiques

Celles-ci sont définies dans le contexte d'une ontologie. Par exemple, la séquence d'éléments {enfant, adolescent, jeune homme, adulte, âgé} est définie dans le contexte d'une ontologie de la vie humaine [51].

2.3.3.2. Les flux de données

Au cours de ces dernières années, les progrès de la technologie matérielle et de la technologie de l'information et de télécommunication ont facilité la collecte continue de données. De simples transactions de la vie quotidienne (comme l'utilisation de la carte de crédit, l'utilisation de mobile et la navigation sur le Web) conduisent à la production automatisée, continue, rapide et volumineuse de données sous forme de flux. Dans de nombreux cas, ces grands volumes de données peuvent être exploités à la recherche d'informations intéressantes et pertinentes dans une grande variété d'applications.

En général, un flux de données est un ensemble de données produit par incréments dans le temps, plutôt que d'être entièrement disponible avant le début de son traitement. Dans le modèle de flux de données, les données arrivent à grande vitesse et les algorithmes qui doivent les traiter ont des contraintes d'espace et de temps très strictes. De plus, une partie ou la totalité des données d'entrée qui doivent être exploitées ne sont pas disponibles dès le début dans un support de stockage, mais arrivent plutôt comme un ou plusieurs flux de données continus.

Les flux de données diffèrent des autres types de données primitifs de plusieurs façons, à savoir [57] :

- Les éléments de données du flux arrivent en ligne.
- Le système n'a aucun contrôle sur l'ordre dans lequel les éléments de données doivent être traités, soit à l'intérieur d'un flux de données, soit à travers plusieurs flux de données.
- Les flux de données sont potentiellement illimités en taille.
- Une fois qu'un élément d'un flux de données a été traité, il est jeté ou archivé. Il ne peut être récupéré facilement que s'il est explicitement stocké en mémoire, ce qui est généralement rare par rapport à la taille des flux de données.

D'après ces caractéristiques de flux de données, ces derniers ne peuvent pas être stockés dans des systèmes de base de données traditionnels, puisqu'ils ne peuvent pas lire le flux qu'une seule fois dans l'ordre séquentiel [58]. Donc, cela pose de grands défis pour la fouille efficace des flux de données. Bien que les algorithmes utilisés par la fouille de données

statiques supposent que les données soient disponibles à l'avance. Néanmoins, pour les flux de données, elles doivent être traitées immédiatement, sinon elles peuvent être perdues. Puisque ces données arrivent si rapidement qu'il n'est pas possible de les stocker toutes en mémoire conventionnelle, afin d'explorer et d'interagir avec ces données ultérieurement.

Pour faire face à ce problème, les algorithmes de traitement des flux impliquent chacun une synthèse du flux d'une manière ou d'une autre [58]. Parmi les solutions :

a. Concevoir un bon échantillon de flux de données
b. Filtrer le flux de données afin d'éliminer la plupart des éléments indésirables.
c. Estimer le nombre d'éléments différents dans un flux en utilisant beaucoup moins de mémoire que ce qui serait nécessaire si nous listions tous les éléments dans le flux de données.
d. Une autre approche pour résumer un flux consiste à examiner uniquement une fenêtre de longueur fixe (par exemple, cette fenêtre est constituée de *n* derniers éléments). Nous interrogeons ensuite la fenêtre comme s'il s'agissait d'une table dans une base de données. S'il y a beaucoup de flux et/ou que *n* est grand, il est possible que nous ne puissions pas stocker la fenêtre entière pour chaque flux et nous devons donc résumer même les fenêtres.

2.3.4. Le prétraitement des données dynamiques

Le prétraitement de données dans la fouille de données dynamique varie selon le type d'objets dynamiques à traiter.

2.3.4.1. Le prétraitement des données temporelles

Dans certaines applications, l'utilisation de séries temporelles au lieu des attributs individuels habituellement utilisés est plus pertinente pour la fouille de données. Toutefois, il peut y avoir des exigences particulières pour le prétraitement de ce type de données. Dans de tels cas, il pourrait être nécessaire de déterminer de nouvelles mesures comme la distance entre les séries temporelles et/ou appliquer certaines étapes de prétraitement (par exemple, réduire ces objets dynamiques aux vecteurs d'attributs à valeur réelle) [49].

Dans ce qui suit, on va donner un aperçu concernant trois tâches de prétraitements de données, à savoir, la sélection, le nettoyage et la normalisation de données.

a. La sélection des données temporelles

La sélection des caractéristiques est une étape importante dans le prétraitement de données. Elle consiste à déterminer les caractéristiques les plus pertinentes à partir d'un ensemble d'exemples d'apprentissage [47]. Si, toutefois, cet ensemble change de manière dynamique au fil du temps, le jeu de caractéristiques sélectionné peut également le faire. Dans de tels cas, nous aurions besoin d'une méthodologie qui nous aide à mettre à jour dynamiquement l'ensemble des caractéristiques sélectionnées. Par exemple, une approche d'emballage dynamique (Dynamic Wrapper Approach) pour la sélection des caractéristiques [59], où la sélection des caractéristiques et la construction du modèle sont effectuées simultanément.

b. Le nettoyage des données temporelles

Pour que le processus de fouille de données produise des résultats significatifs, les données doivent être prétraitées de manière à les rendre propres. Nous aborderons ici deux aspects de la propreté des données, à savoir, le traitement des données manquantes et la suppression du bruit, qui peuvent être produits à cause du mauvais fonctionnement de l'équipement, de l'erreur humaine et des conditions environnementales[51].

Le traitement des données manquantes

Le traitement spécifique des données manquantes dépend du système étudié, de la quantité et du type de données manquantes. Il existe deux approches pour traiter les données manquantes dans les séries temporelles :

a. Ne pas remplir les valeurs manquantes. Par exemple, dans le calcul des similarités, on peut utiliser une méthode qui calcule les similarités de deux séries temporelles en comparant leurs pentes locales. S'il manque un segment de données dans l'une des deux séries temporelles, nous ignorons simplement cet élément dans notre calcul de similarité.
b. Remplir la valeur manquante avec une estimation. Parmi les méthodes d'estimation, il y a la méthode d'interpolation, par exemple, supposons que nous ayons les points x_1 et x_3, avec les valeurs $f(x_1)$ et $f(x_3)$, respectivement. La valeur du point x_2 est manquante et elle est quelque part entre x_1 et x_3. Nous pouvons utiliser une interpolation linéaire pour estimer $f(x_2)$ comme suit :

$$f(x_2) = f(x_1) + \frac{(x_2 - x_1)\big(f(x_3) - f(x_1)\big)}{x_3 - x_1} \qquad (3.1)$$

La suppression des données bruit

Le bruit est défini comme une erreur aléatoire qui se produit dans le processus de fouille de données. Nous discuterons de deux méthodes de traitement du bruit dans la fouille de données :

a. Binning : ici, les données sont divisées en compartiments ou boites (bins) de taille égale. Ensuite, les données sont lissées en utilisant soit la moyenne, soit la médiane, soit les limites de la boite.
b. Lissage à moyenne mobile (Moving Average Smoothing) : le lissage est fréquemment utilisé en finance pour lisser les fluctuations à court terme dans les cours boursiers. Supposons que nous ayons une série temporelle à 5 points, comme le montre la figure 2.9, et que nous souhaitons calculer la moyenne mobile centrale à 3 points, où nous utilisons les valeurs passées et futures autour du point d'intérêt. Par exemple, le lissage à moyenne mobile au temps 2 (indiquée par une étoile sur le graphique) est la moyenne des valeurs pour les points temporels 1 et 3.

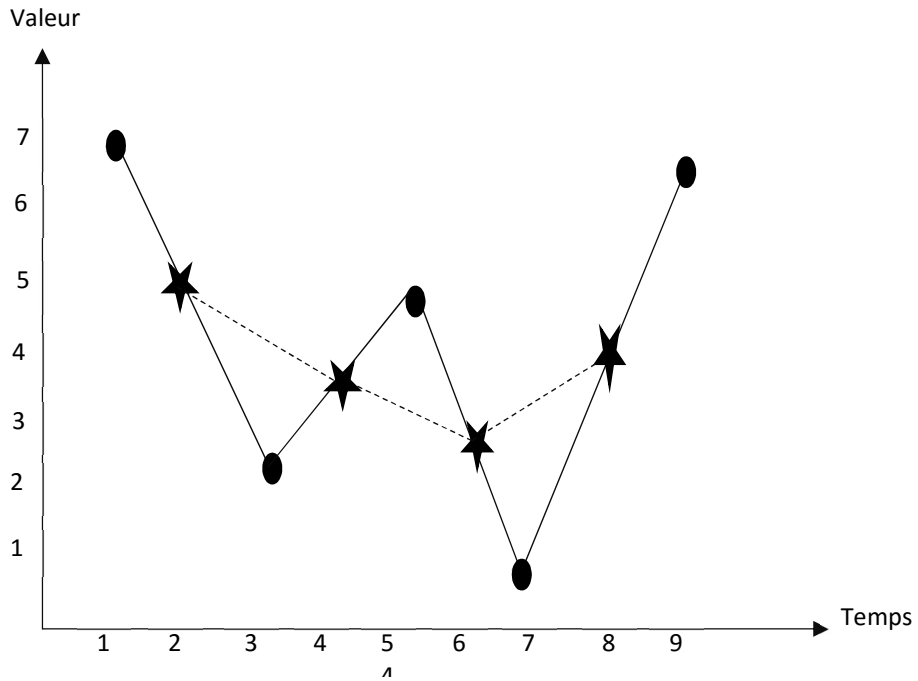


Figure 2.9. Lissage à moyenne mobile d'une série temporelle.

Normalisation des données temporelles

Dans la normalisation des données temporelles, les données sont mises à une échelle donnée de sorte qu'elles se situent dans une plage prédéfinie, telle que l'intervalle [0-1] [51]. La normalisation permet de transformer les données à la même « échelle », et par conséquent,

d'effectuer des comparaisons directes entre leurs valeurs. Supposons, par exemple, que nous voulions comparer les valeurs de températures dans deux villes pendant le mois de février, où dans une ville les températures sont mesurées en Celsius et dans l'autre en Fahrenheit. Nous examinerons deux façons de procéder à la normalisation :

a. Normalisation min-max : pour faire ce type de normalisation, nous avons besoin de connaître le minimum (x_{min}) et le maximum (x_{max}) des données.
b. Normalisation du score Z : ici, la moyenne et l'écart-type des données sont utilisés pour les normaliser.

2.3.4.2. Le prétraitement de flux de données

Dans l'analyse hors ligne des données, le prétraitement des données représente une seule procédure constituée d'une ou plusieurs tâches (sélection, nettoyage …), généralement effectuées par un expert humain avant l'étape de fouille de données. Cependant, dans les flux de données, le traitement manuel n'est pas réalisable, car de nouvelles données arrivent en permanence. Donc, les flux de données nécessitent des méthodes de prétraitement entièrement automatisées, qui permettent d'optimiser les paramètres et de fonctionner de manière autonome. De plus, les modèles de prétraitement doivent pouvoir se mettre à jour automatiquement avec l'évolution des données, de la même manière que les modèles de fouille de flux de données. Sinon, le modèle utilisé précédemment peut devenir inutile.

Pour illustrer les défis liés au prétraitement de flux de données, on considère cet exemple, qui consiste à prédire les embouteillages en fonction des données de détection mobiles [60]. Dans ce système, les personnes utilisant les services de navigation sur des appareils mobiles peuvent choisir d'envoyer des données anonymisées au fournisseur de services, tels que Google, Yandex ou Nokia, qui fournissent des estimations et des prévisions, des embouteillages sur la base de ces données. Pour accomplir cette fonctionnalité, les données de chaque utilisateur sont mappées sur le réseau routier, la vitesse de chaque utilisateur sur chaque segment de route est calculée, les données provenant de plusieurs utilisateurs sont agrégées et finalement la vitesse actuelle du trafic est estimée. Cependant, de nombreux problèmes de prétraitement des données sont associés à cette tâche.

Au premier lieu, le bruit des données GPS peut varier en fonction de l'emplacement et de la densité du réseau de télécommunication. Il peut y avoir des valeurs aberrantes, par exemple, si un véhicule s'est arrêté au milieu d'une route pour attendre un passager ou s'il existe une collision de voitures.

En outre, les réseaux routiers peuvent évoluer au fil du temps, entraînant des variations dans les vitesses de déplacement, ainsi que le nombre et le type de véhicules autorisés (par exemple, les camions lourds pourraient être interdits et de nouveaux itinéraires plus performants pourraient être créés). Donc, toutes ces questions nécessitent des actions de prétraitement automatisées avant d'alimenter les modèles prédictifs avec les données les plus récentes.

Parmi les tâches principales de prétraitement de flux de données, il y a la réduction de flux de données, qui est nécessaire pour traiter les éléments en ligne ou en mode batch le plus rapidement possible et sans hypothèses préalables sur la distribution des données.

Dans ce qui suit, nous décrivons quelques propositions de réduction adaptées à la fouille de flux de données.

a. La réduction de la dimensionnalité

De nombreux algorithmes de sélection de caractéristiques pour les flux de données ont été proposés. La plupart d'entre eux sont des algorithmes incrémentaux venant de ceux qui sont conçus pour le traitement hors ligne [61], alors que d'autres sont spécifiquement conçus pour s'adapter aux flux de données [62]. Les méthodes de sélection des caractéristiques peuvent être divisées en trois groupes selon le moment où la sélection est effectuée (c'est-à-dire avant et indépendamment de l'étape d'apprentissage, ou étroitement liée à celle-ci). Ces groupes sont les filtres, les encapsuleurs et les méthodes hybrides.

b. La réduction du nombre d'instances

Dans le cas de flux de données, les instances enregistrées appartenant à des concepts précédents peuvent dégrader les performances de modèle d'apprentissage si de nouveaux concepts apparaissent. De même, les nouvelles instances représentant un nouveau concept peuvent être classées comme bruit et supprimées par un comportement incorrect du mécanisme de la sélection d'instances, car elles sont incompatibles avec les concepts passés [63]. Donc, une certaine amélioration, maintenance et condensation [61] devraient être effectuées sur les bases d'instances sous la forme d'un processus amélioré de sélection d'instances, qui sélectionnent les cas qui représentent le mieux l'état actuel du flux de données.

La plupart des techniques actuelles de sélection d'instances sont conçues pour des environnements stationnaires et ignorent le phénomène de dérive conceptuelle qui signifie que les propriétés statistiques de la variable à prédire changent au fil du temps de façon imprévue, ce qui cause des problèmes, parce que les prédictions deviennent moins précises au fil du

temps [64]. Parmi les techniques permettant de la sélection d'instances de manière incrémentielle ou par lots, on peut citer l'algorithme d'apprentissage basé sur les instances [65], l'algorithme d'apprentissage fondé sur des instances à partir de flux de données (IBL-DS) [66] et le système de classification et de régression basée sur des instances de flux de données IBLStreams [67].

c. La simplification de l'espace des caractéristiques

La définition et le nombre d'intervalles de discrétisation peuvent changer au fil du temps, en fonction de l'évolution des caractéristiques des données. Par conséquent, les algorithmes de discrétisation des flux de données doivent être capables de gérer l'apparition de dérives conceptuelles. Également, il est souhaitable que les intervalles de discrétisation puissent s'adapter soigneusement à la dérive conceptuelle, sans imposer un coût de calcul supplémentaire lors du recalcul [62].

La discrétisation à fréquence égale (basée sur les histogrammes) peut être considérée comme l'une des premières techniques permettant de traiter la discrétisation incrémentale. En utilisant des quantiles en tant que points de découpage, l'espace de fonctions peut être partitionné en intervalles de fréquence égale. L'estimation des quantiles dans les flux a été étudiée en profondeur sous des formes approximatives [68, 69] et exactes [70, 71]. L'une des alternatives de discrétisation les plus agiles et les plus efficaces est l'algorithme de discrétisation incrémentielle (IDA) [68] qui se rapproche des quantiles grâce au maintien d'un échantillon de réservoir du flux d'entrée. Dans ce cas, les intervalles sont structurés à l'aide de tas d'intervalles qui est une structure de données efficace, permettant d'insérer et de supprimer des éléments d'une manière plus simple, et de récupérer le maximum et le minimum (les limites d'intervalle) en temps constant.

2.3.5. Tendances de la fouille de données dynamique

En effet, la fouille de données dynamique devient un domaine de recherche bien établi. Elle aborde les problèmes concernant l'extraction dynamique de connaissances à partir de données; à savoir la sélection dynamique de caractéristiques, le regroupement de trajectoires de caractéristiques[72], la fouille incrémentale de données [73-76] et la fouille de flux de données[62, 77, 78]. Parmi les tendances de la fouille de données dynamique, on peut citer la fouille de flux de données, la fouille de séries temporelles et la fouille de données incrémentale.

Dans ce qui suit, on va mettre l'accent sur quelques types de la fouille de données dynamique.

2.3.5.1. La fouille de séries temporelles

Les données de séries temporelles peuvent être générées par de nombreux processus naturels et économiques, tels que les marchés boursiers, et les observations scientifiques, médicales et naturelles. Parmi les tâches qu'on peut appliquer dans la fouille de séries temporelles, on trouve les suivantes [55] :

La recherche de similitude

Une recherche de similarité a comme but de trouver des séquences de données qui ne diffèrent que légèrement de la séquence de requêtes.

De nombreuses requêtes de similarité de séries temporelles nécessitent la correspondance de sous-séquences, c'est-à-dire la recherche de l'ensemble de séquences qui contiennent des sous-séquences similaires à une séquence de requête donnée.

Pour la recherche de similarité, il est souvent nécessaire d'effectuer d'abord une réduction des données ou de la dimensionnalité et/ou une transformation des données de séries temporelles comme la transformation de normalisation. Les techniques habituelles de réduction de la dimensionnalité comprennent la transformation de Fourier discrète (DFT), la transformation en ondelettes discrètes (DWT), la décomposition en valeurs singulières (SVD) et l'analyse des composantes principales (ACP). Avec de telles techniques, les données ou les signaux sources sont mappés sur d'autres signaux dans un espace transformé. Un petit sous-ensemble des coefficients transformés (les plus forts) est sauvegardé sous forme de caractéristiques qui forment un espace de caractéristiques qui est une projection de l'espace transformé. De plus, des indices peuvent être construits sur les séries temporelles originales ou transformées pour accélérer la recherche de similarité.

Une autre technique de transformation est basée sur la transformation des séries temporelles en approximations agrégées par morceaux, afin que les données puissent être considérées comme une séquence de représentations symboliques. Dans ce cas, le problème de la recherche de similarité se transforme alors en un problème de correspondance de sous-séquences dans les données de séquences symboliques. Nous pouvons ensuite, identifier des motifs séquentiels fréquents et construire des mécanismes d'index ou de hachage pour une recherche efficace basée sur ces motifs.

Parmi les techniques utilisées dans la recherche de similarité, on cite les suivantes :

- La correspondance atomique : trouver des paires de fenêtres de petite taille, sans espace, qui sont similaires.
- La couture de fenêtres : assembler des fenêtres similaires pour former des paires de grandes sous-séquences similaires, ce qui permet de créer des espaces entre les correspondances atomiques.
- L'ordonnancement des sous-séquences : ordonner linéairement les correspondances de sous-séquences pour déterminer s'il existe suffisamment de pièces similaires.

L'analyse de régression et de tendance

En général, dans les tâches de régression ou d'analyse de tendances, un modèle intégré est construit en utilisant les quatre composantes principales qui caractérisent les données de séries temporelles, à savoir les tendances, les variations saisonnières, les fluctuations cycliques et les variations aléatoires.

L'analyse des tendances peut également être utilisée pour la prévision des séries temporelles, c'est-à-dire pour trouver une fonction mathématique qui produira approximativement les tendances historiques d'une série temporelle et l'utiliser pour faire des prévisions à long ou à court terme des valeurs futures. La modélisation ARIMA (moyenne mobile intégrée autorégressive), la modélisation des séries temporelles à mémoire longue et l'autorégression sont des méthodes populaires pour une telle analyse.

La classification de séries temporelles

En général, les méthodes de classification des séquences de données peuvent être organisées en trois catégories [55]:

a. La classification basée sur les caractéristiques, qui transforme une séquence en un vecteur de caractéristiques et applique ensuite les méthodes de classification conventionnelles.
b. La classification basée sur la distance des séquences, où la fonction de distance qui mesure la similarité entre les séquences détermine la qualité de la classification de manière significative.
c. La classification basée sur un modèle, comme l'utilisation d'un modèle de Markov caché (HMM) ou d'autres modèles statistiques pour classer les séquences.

Les techniques de sélection des caractéristiques ne peuvent pas être facilement appliquées aux données de séries temporelles sans discrétisation. Cependant, la discrétisation peut entraîner une perte d'informations. Une méthode nommée « Time Series Shapelets » [79] utilise les sous-séquences de séries temporelles qui peuvent représenter au maximum une classe en tant que caractéristiques, ce qui permet d'obtenir des résultats de classification de qualité.

2.3.5.2. La fouille de données en temps réel

Certains problèmes de fouille de données sont souvent associés à la variation du taux de production de données, où la quantité de données et le nombre d'attributs se modifient en temps réel. Dans ce cas, la situation des données dépasse de plus en plus les capacités des méthodes statiques de fouille de données. De nombreuses applications nécessitent des modèles dynamiques de fouille de données en temps réel, comme l'astronomie [80] et les systèmes de fabrication [81].

Le terme « temps réel » est utilisé pour décrire comment un algorithme de fouille de données peut s'adapter instantanément ou simultanément à chaque changement de données. Cependant, le problème venant de la production de données en temps réel est généralement étroitement lié au fait que les algorithmes de fouille de données conventionnels fonctionnent en mode batch, où il est nécessaire d'avoir toutes les données pertinentes dès le début, avant la tâche principale de la fouille de données. De plus, la fouille de données en temps réel doit avoir certaines caractéristiques indépendamment de la quantité de données impliquées, à savoir l'apprentissage incrémental, l'apprentissage décrémental, l'ajout d'attribut, la suppression d'attribut, le traitement distribué et le traitement parallèle. Par conséquent, la mise à niveau de la fouille de données statique vers la fouille de données en temps réel passe par l'utilisation d'une méthode appelée apprentissage automatique en temps réel, où l'utilisation de cette méthode avec les méthodes habituelles de fouille de données permet d'explorer les données en temps réel [82].

2.3.5.3. La fouille de flux de données

Des recherches substantielles sur la fouille de flux de données ont permis la mise au point de méthodes efficaces pour les domaines de fouille de motifs séquentiels [83, 84] , l'analyse multidimensionnelle et l'entrepôt de flux de données [85, 86], ainsi que la classification, le clustering, l'analyse d'anomalies et la détection en ligne d'événements rares [87-89].

Les méthodes de la fouille de flux de données mentionnées précédemment ainsi que d'autres, s'appuient généralement sur le développement des algorithmes à balayage unique ou à quelques balayages en utilisant des capacités de calcul et de stockage limitées. Cela inclut la collecte de données de flux dans des fenêtres coulissantes ou des fenêtres temporelles inclinées (où les données les plus récentes sont enregistrées à la granularité la plus fine et les données les plus éloignées sont enregistrées à une granularité plus grossière), ainsi que des techniques d'exploration telles que le microclustering, l'agrégation limitée et l'approximation [55, 90].

En effet, la fouille de flux de données peut être utilisée dans de nombreuses applications, comme la détection de motifs ou d'anomalies en temps réel dans les flux de données qui se produisent par les réseaux informatiques, les réseaux électriques, les réseaux de capteurs, les conversations téléphoniques et les recherches sur le Web [89, 91-93].

3.4. Conclusion

Dans ce chapitre, on a présenté quelques définitions et concepts de bases concernant les domaines de la fouille de données statiques et de la fouille de données dynamique. Dans la section de la fouille de données dynamique, on a présenté également certains concepts de bases concernant les objets dynamiques à traiter (par exemple, les flux de données et les données temporelles), ainsi que des tendances de ce domaine, comme la fouille de séries temporelles et la fouille de flux de données. Dans le chapitre suivant, on va mettre l'accent sur les modèles de la fouille de données, puis les modèles de la fouille de données dynamique comme les réseaux bayésiens dynamiques, les réseaux de neurones récurrents et le clustering dynamique.

Chapitre 3. Modèles de fouilles de données dynamiques

3.1. Introduction

Vu l'importance de choisir le type de modèles ou de connaissances le plus pertinent vis-à-vis les objectifs définis au début d'un processus de fouille de données, l'analyste de données doit avoir une large information à propos de ces types de modèles et de leurs domaines d'application appropriés. En effet, les modèles de la fouille de données peuvent être classés selon plusieurs critères, à savoir, le type de données traitées (par exemple, les données multimédias, les données texte, les données web et les séries temporelles), le type de tâche effectuée (par exemple, le clustering, la classification et les règles d'association) ou le domaine d'application (par exemple, le commerce électronique, la télécommunication et la biologie). Cette diversité de critères engendre plusieurs modèles divisés en plusieurs classes.

De la même manière, les modèles de la fouille de données dynamique doivent être aussi pertinents pour s'adapter aux différents types de données dynamiques, ainsi que les différents domaines d'application et les tâches de la fouille de données ; ce qui engendre également plusieurs types de modèles dynamiques, à savoir les réseaux bayésiens dynamiques, les arbres de décision dynamique et les réseaux de neurones dynamiques.

Après cette introduction, on va montrer respectivement dans ce chapitre quelques types de modèles incorporés dans les domaines de la fouille de données et la fouille de données dynamique.

3.2. Les modèles et les tâches de fouille de données

Les tâches de fouille de données comme la classification, le clustering et les règles d'association peuvent être appliqués dans plusieurs différents domaines. Ces tâches utilisent d'une manière générale deux stratégies, à savoir l'apprentissage supervisé et l'apprentissage non supervisé. Dans l'apprentissage supervisé, un ensemble de données d'apprentissage est utilisé pour déterminer les paramètres du modèle, alors que dans l'apprentissage non supervisé, on n'utilise pas un ensemble de données d'apprentissage (les modèles sont uniquement descriptifs), et le but du processus est de construire un modèle qui décrit des régularités intéressantes dans les données.

En plus, les tâches de fouille de données peuvent être classées selon l'objectif global du processus de fouille de données en tâches descriptives et prédictives [94]. La figure suivante

représente les différentes techniques utilisées dans la fouille de données, ainsi que la catégorie de chaque technique.

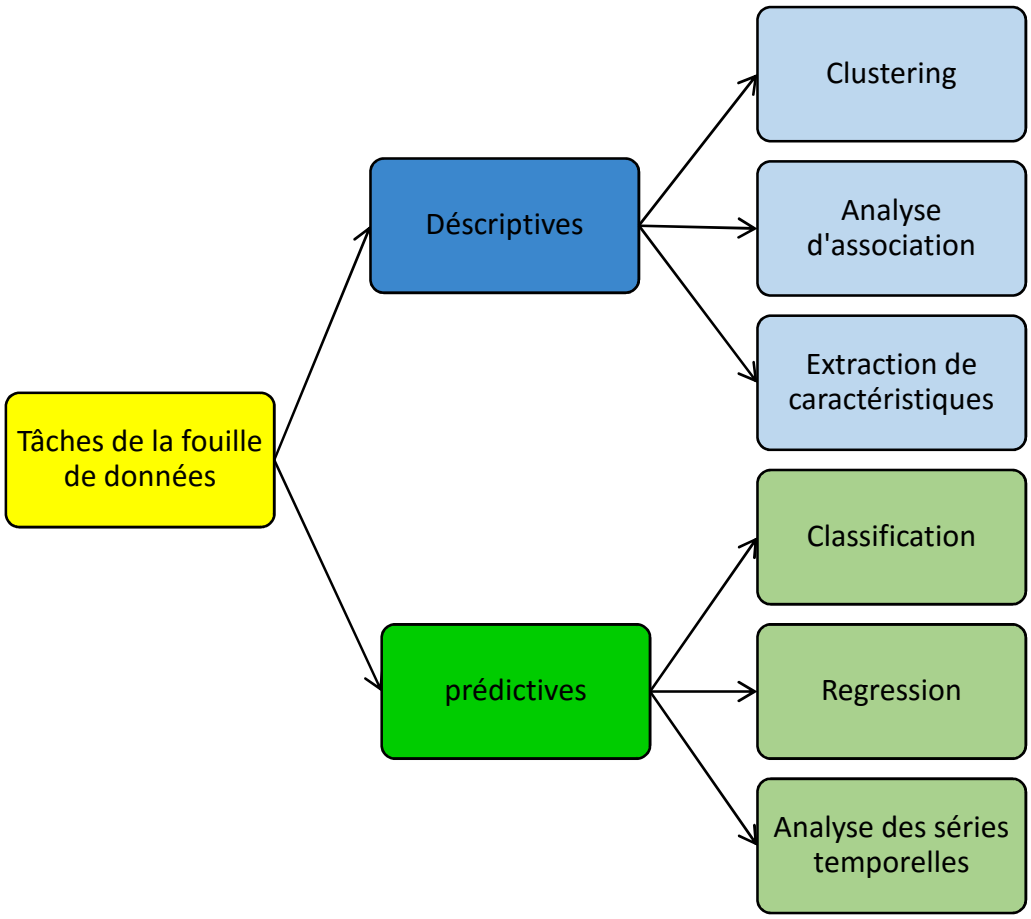


Figure 3.1. Classification des tâches de la fouille de données.

Les tâches descriptives ont pour objectif de trouver des formes, des tendances ou des associations compréhensibles ou interprétables par l'homme, c'est-à-dire qu'elles résument les données d'une manière donnée. Du point de vue de l'apprentissage automatique, nous pourrions comparer ces algorithmes à l'apprentissage non supervisé qui permet d'identifier des modèles dans les données qui élargissent notre connaissance et notre compréhension du monde que les données reflètent. Les tâches de clustering, d'analyse d'association, d'analyse de corrélation et de découverte de motifs fréquents peuvent entrer dans la catégorie d'analyse descriptive.

Dans la modélisation prédictive, après avoir examiné les données et la construction complète du modèle de prédiction, ce modèle permet de prédire un résultat d'intérêt [95]. La prédiction est l'une des techniques de fouille de données permettant de découvrir la relation entre des variables dépendantes et indépendantes. Ainsi, les tâches prédictives permettent de construire un modèle permettant de prédire l'occurrence d'un événement ou la valeur d'une

variable, en se basant sur d'autres informations existantes. Du point de vue de l'apprentissage automatique, on parle également d'apprentissage supervisé. Les données historiques à partir desquelles nous construisons nos modèles prédictifs seront déjà associées à des résultats spécifiques. Les tâches de classification et de régression sont au cœur de ce que nous considérons souvent comme de la fouille de données prédictive.

Nous décrivons brièvement ci-dessous les principales techniques de fouille de données prédictive et descriptive, ainsi que quelques modèles utilisés dans ces techniques, comme les réseaux de neurones et les arbres de décision.

3.2.1. La classification

La classification utilise un ensemble d'exemples préclassifiés pour élaborer un modèle permettant de classer la population d'enregistrements [96].

Le processus de classification des données implique les tâches d'apprentissage, de test et de calibration. Au premier lieu, les données d'apprentissage sont analysées par un algorithme de classification, afin de construire des règles ou des modèles de classification. Ensuite, d'autres données de test sont utilisées pour estimer l'exactitude de ces règles ou ces modèles. Si la précision est acceptable, ces règles ou ces modèles de prévision peuvent être appliqués aux nouveaux enregistrements de données, sinon le modèle va être réévalué ou calibré, afin d'atteindre la précision prédéfinie à l'avance.

Par exemple, la surveillance de la fraude par cartes de crédit, en surveillant des millions de comptes qui sont représentés par des enregistrements d'activités frauduleuses ou légales. Dans ce cas, l'algorithme d'apprentissage du classificateur utilise ces exemples préclassifiés pour déterminer l'ensemble des paramètres requis pour une discrimination appropriée.

Parmi les types de modèles de classification, on peut citer :

- Les réseaux de neurones artificiels.
- Les arbres de décision.
- La classification bayésienne.
- La machine à vecteurs de support.

3.2.2. Le clustering

Le regroupement ou le partitionnement de données (clustering) est une technique utilisée dans la fouille de données qui permet de créer des grappes d'objets (clusters) ayant des caractéristiques similaires [97].

Les techniques de clustering définissent les classes et placent les objets dans chaque classe, tandis que dans les techniques de classification, les objets sont assignés dans des classes prédéfinies.

Nous pouvons prendre comme exemple la gestion des livres dans la bibliothèque. Dans une bibliothèque, il existe un large éventail de livres dans divers sujets. Le défi est de savoir comment conserver ces livres de manière qui permet aux lecteurs de prendre plusieurs livres sans se déplacer plusieurs fois entre les différents rangés d'étagères. À l'aide de la technique de clustering, on peut conserver des livres présentant certaines similitudes dans une grappe ou une étagère et les étiqueter avec un nom significatif. Si les lecteurs veulent des livres sur le même sujet qui sont situés dans une étagère spécifique, ils n'auront qu'à aller sur cette étagère au lieu de chercher toute la bibliothèque.

Parmi les types de méthodes de clustering, on cite :

- Les méthodes basées sur des centroïdes comme l'algorithme k-means.
- Les méthodes de clustering hiérarchique.
- Les algorithmes basés sur la densité tels que DBSCAN ou OPTICS.

3.2.3. La régression

L'analyse de régression est une technique de fouille de données utilisée pour découvrir des relations entre des variables dépendantes et des variables indépendantes [98].

Par exemple, cette technique peut être utilisée dans la vente pour prévoir le bénéfice futur. Si nous considérons par exemple que la vente est une variable indépendante, le bénéfice pourrait être une variable dépendante. Ensuite, en se basant sur les données historiques sur les ventes et les bénéfices, on peut établir une courbe de régression ajustée qui est utilisée pour la prévision des bénéfices.

Malheureusement, beaucoup de problèmes du monde réel ne sont pas de simples prédictions. Par exemple, les volumes de ventes, les cours boursiers et les taux de défaillance des produits sont tous très difficiles à prévoir, car ils peuvent dépendre d'interactions

complexes de plusieurs variables prédictives. Par conséquent, des techniques plus complexes (par exemple, la régression logistique, les arbres de décision ou les réseaux de neurones) peuvent être nécessaires pour prévoir les valeurs futures.

Les mêmes types de modèles peuvent être utilisés à la fois pour la régression et la classification. Par exemple, les réseaux de neurones et les arbres de décision CART peuvent créer des modèles de classification et de régression. Parmi les types de méthodes de régression les plus reconnus [99, 100], on cite :

- La régression linéaire
- La régression non linéaire
- La régression linéaire multivariée
- La régression non linéaire multivariée
- La régression logistique
- La régression polynomiale

3.2.4. Les règles d'association

L'analyse des règles d'association est une technique de fouille de données permettant de déterminer la manière dont les éléments (items) sont associés les uns aux autres dans une base de transactions [25, 101].

Soit I un ensemble fini d'éléments (un ensemble d'items) et D un ensemble de sous-ensembles de I (une base de transactions). Chaque $T \in D$ est appelé une transaction. Nous disons que la transaction $T \in D$ supporte l'itemset $X \subseteq I$ (un sous ensemble d'items de I), si $X \subseteq T$ est vérifiée.

Une règle d'association est une expression $X \Rightarrow Y$, où X, Y sont des itemsets et $X \cap Y = \emptyset$. Il existe trois paramètres pour mesurer une règle d'association. Ces paramètres sont :

- Le support d'une transaction X est le pourcentage du nombre de transactions T qui contient l'itemset X par rapport au nombre total de transactions D, c'est-à-dire, $supp(X) = |\{T \in D \mid X \subseteq T\}| / |D|$.
- Le support d'une règle $X \Rightarrow Y$ est défini comme $supp\ (X \Rightarrow Y) = supp\ (XUY)$.
- L'indice de confiance de la règle $X \Rightarrow Y$ est défini comme le pourcentage du nombre de transactions contenant Y et X par rapport au nombre total de transactions contenant X, c'est-à-dire, $conf\ (X \Rightarrow Y) = supp\ (XUY) / supp\ (X)$. Également, la confiance de la

règle *(X ⇒ Y)* peut être comprise comme la probabilité conditionnelle $P(Y \subseteq T \mid X \subseteq T$.

- Le lift d'une règle $X \Rightarrow Y$ représente le rapport entre le support observé et celui attendu si X et Y étaient indépendants. Il est défini comme suit : *supp (XUY)/ (supp (X) supp (Y))*.

Par exemple, sur une base des données de vente historique, les détaillants pourraient découvrir que les clients achètent toujours du café lorsqu'ils achètent du lait, et qu'ils peuvent donc mettre du café et du lait les uns à côté des autres pour gagner du temps et augmenter les ventes.

Voici quelques types de règles d'association :

- Les règles d'association à plusieurs niveaux.
- Les règles d'association multidimensionnelle.
- Les règles d'association quantitative.

3.2.5. Les arbres de décision

L'arbre de décision est l'une des techniques de fouille de données les plus utilisées, car son modèle est facile à comprendre pour les utilisateurs. Elle est initialement utilisée dans la théorie de la décision et les statistiques, et devenue un outil très efficace dans d'autres domaines tels que la fouille de données, l'apprentissage automatique et la reconnaissance de formes [102]. Il existe plusieurs algorithmes d'induction d'arbre de décision, parmi ces algorithmes, on peut citer l'algorithme ID3[103] qui a été développé par John Ross Quinlan. Plus tard, il a présenté C4.5 [104], qui était le successeur de ID3. Les algorithmes ID3 et C4.5 adoptent une approche gourmande, dans lesquels il n'y a pas de retour en arrière et les arbres sont construits d'une manière récursive en utilisant une approche diviser pour régner de haut en bas.

Un arbre de décision est une structure qui inclut un nœud racine, des branches, des nœuds internes et des nœuds de feuille. Chaque nœud interne représente une condition sur un attribut, chaque branche indique le résultat d'une condition et chaque nœud de feuille porte une étiquette de classe. Le nœud le plus haut dans l'arborescence est le nœud racine. Par exemple, l'arbre de décision suivant (figure 3.2) indique si un client d'une entreprise est susceptible d'acheter un ordinateur ou non. Chaque nœud interne représente une condition sur un attribut et chaque nœud de feuilles représente une valeur de variable cible, à savoir acheter un ordinateur ou non.

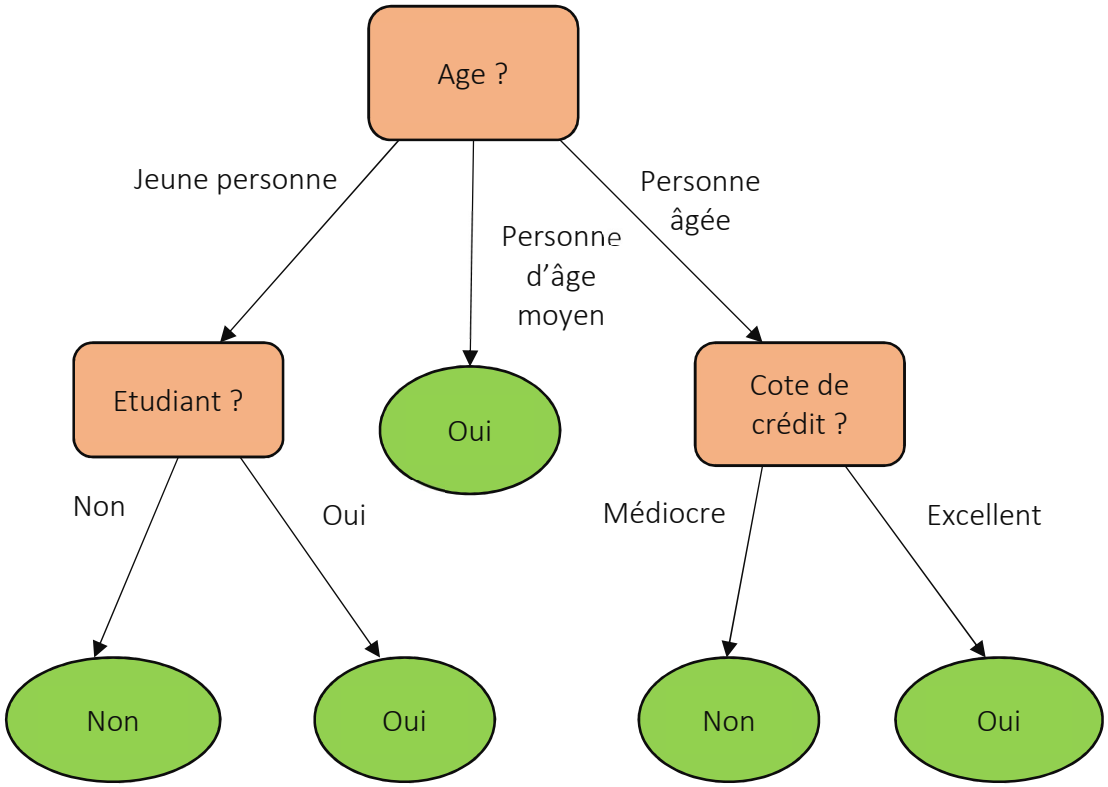


Figure 3.2. Exemple d'un arbre de décision

3.2.6. Les réseaux bayésiens

Le réseau bayésien est un modèle graphique probabiliste, utilisé pour représenter les indépendances conditionnelles entre un ensemble de variables aléatoires. Un réseau bayésien est représenté essentiellement par un graphe orienté acyclique DAG (Directed Acyclic Graph), les variables sont représentées par des nœuds et les relations entre ces nœuds sont représentées par des arcs. Dans ce graphe, nous associons à chaque nœud une table de probabilités conditionnelles qui représente la distribution de probabilité de la variable correspondante démontrant les dépendances entre cette variable et ses parents (les nœuds commençant des arcs vers cette variable). Donc, le processus d'apprentissage du réseau bayésien comprend l'apprentissage de la structure et des paramètres du réseau.

Les réseaux bayésiens ont été largement utilisés pour extraire les dépendances sous-jacentes entre un ensemble de variables aléatoires. Puis, utiliser ces modèles pour répondre à des questions de prévision ou pour trouver l'explication la plus probable d'une séquence d'événements.

La séparation directionnelle [105, 106] (D-Séparation) est un concept important de réseaux bayésiens qui nous aide à découvrir les relations entre un ensemble de variables aléatoires.

Considérons *X*, *Y* et *Z* comme étant trois sous-ensembles de nœuds dans un DAG donné. Nous disons que *Z* sépare *X* de *Y* (noté par $X \perp Y \mid Z$), si pour chaque nœud *x* appartenant à *X*, et un autre nœud *y* appartenant à *Y*, il existe un nœud *z* appartenant à *Z* qui convient à une des situations représentées à la figure 3.3 (un nœud ombré signifie que le nœud est observé) :

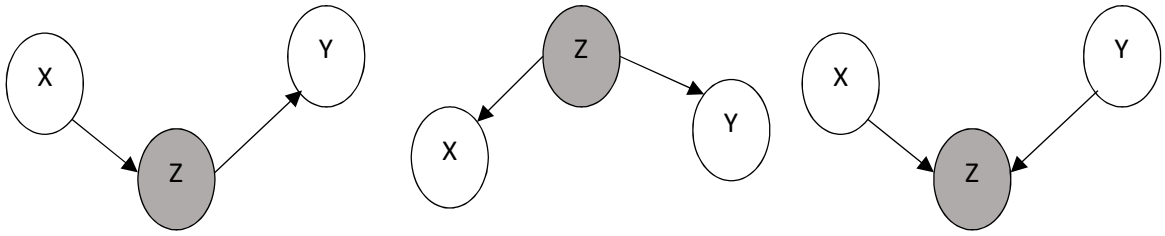


Figure 3.3. Les règles de D-Séparation

La probabilité conditionnelle de deux ensembles indépendants de variables aléatoires *A* et *B*, étant donné un autre ensemble *C,* peut être formulée comme suit :

$$P(A, B|C) = P(A|C).\ P(B|C) \qquad (3.1)$$

La probabilité d'un ensemble de variables dans un réseau bayésien est le produit des probabilités conditionnelles de chaque variable de l'ensemble, compte tenu des valeurs de ses parents. Cette propriété peut être formulée comme suit :

$$P(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} P(x_i \mid Parents(x_i)) \qquad (3.2)$$

Les réseaux bayésiens sont caractérisés par les propriétés suivantes :

- La propriété de suffisance causale indique que les variables qui sont les parents des variables observées du domaine ne doivent pas être cachées.
- La propriété causale de Markov détermine l'indépendance d'une variable donnée par rapport à l'ensemble de ses non-descendants compte tenu de ses parents.

La propriété de fidélité indique que la structure du réseau bayésien et la distribution de probabilité s'accordent sur les relations de dépendance/indépendance découvertes entre les variables. Dans ce cas, on dit que la distribution de probabilité et la structure sont fidèles l'une à l'autre [107].

La Frontière de Markov *B(x)* est la plus petite famille d'une variable *x* donnée, qui se compose de ses parents, de ses partenaires (les autres parents de ses enfants) et de ses enfants. Cette famille sépare (protège) le nœud *x* de tous les autres nœuds variables. En d'autres termes,

disons que *x* est une variable de l'ensemble de variables *U*, donc la frontière de Markov est la couverture de Markov minimale de *x*, qui est l'ensemble minimal de variables $BL(x) \subseteq U$ (il ne pouvait pas être unique), ce qui rend *la variable x* indépendante de toute autre variable $y \notin \{BL(x) \cup \{x\}\}$, et on note : $B(x) = x \perp y \mid BL(x)$.

3.2.7. Les réseaux de neurones artificiels

Le réseau de neurones artificiels (RNA) est modelé sur le réseau neuronal biologique. Il représente une interconnexion de nœuds analogues aux neurones. Chaque réseau neuronal comporte trois éléments essentiels, à savoir le caractère du nœud, la topologie du réseau et les règles d'apprentissage. Le caractère de nœud détermine la façon dont les signaux sont traités par le nœud. Il est constitué du nombre d'entrées et de sorties associées au nœud, du poids associé à chaque entrée et sortie, et de la fonction d'activation. La topologie du réseau détermine la façon dont les nœuds sont organisés et connectés. Les règles d'apprentissage déterminent comment les poids sont initialisés et ajustés. Les sections suivantes expliquent chacune de ces composantes [108].

La structure d'un nœud de RNA

La structure de base d'un nœud RNA est illustrée à la figure 3.4. Chaque nœud reçoit ses entrées des nœuds « en amont » et fournit des sorties aux nœuds « en aval ». Chaque connexion entre 2 nœuds possède un poids. Lorsque la somme pondérée des entrées dépasse la valeur seuil du nœud, ce nœud active et transmet un signal via une fonction de transfert aux nœuds voisins. Ce processus peut être exprimé sous la forme d'un modèle mathématique [109]:

$$y = f(\sum_{i=0}^{n} w_i x_i < T)\,, \qquad (3.3)$$

où y est la sortie du nœud, f la fonction de transfert, w_i le poids de l'entrée x_i, et T la valeur de seuil. La fonction de transfert a plusieurs formes. Par exemple, la fonction de transfert non linéaire est plus utile que la fonction linéaire [110], car seuls quelques problèmes sont séparables linéairement.

Parmi les fonctions de transfert les plus simples et les plus utilisées, il existe la fonction step :

$$y = \begin{cases} 0 \quad if \sum_{i=0}^{n} w_i x_i < T \\ 1 \quad if \sum_{i=0}^{n} w_i x_i \geq T \end{cases} \qquad (3.4)$$

La fonction sigmoïde est aussi souvent utilisée comme fonction d'activation, puisque cette fonction et sa dérivée sont des fonctions continues. Cette fonction est formulée comme suit :

$$y = \frac{1}{1 + \exp(-\beta x)} \qquad (3.5)$$

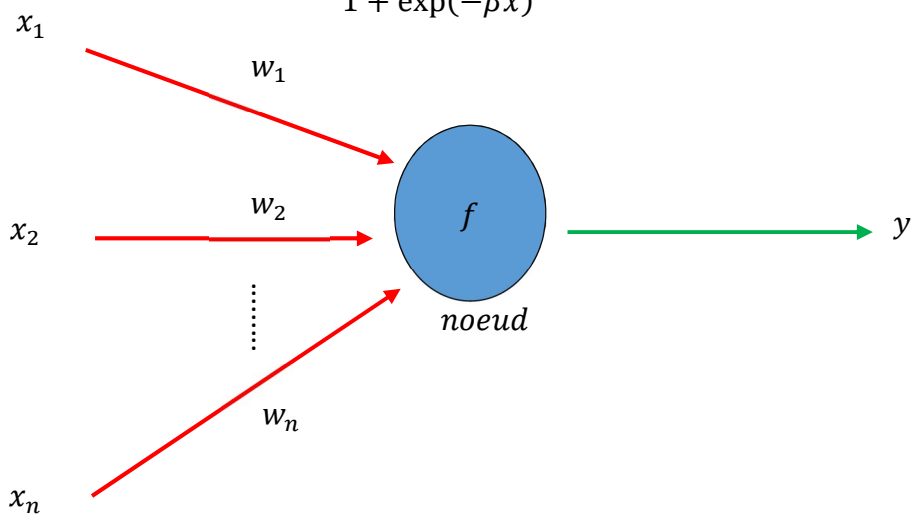


Figure 3.4. Structure d'un nœud de réseau de neurones. x_i : les entrées, w_i : les poids, f : la fonction de transfert et y : la sortie.

Les topologies de réseau de neurones

Selon la figure 3.5(a) représentant l'architecture générale d'un réseau de neurones artificiels, les nœuds sont organisés en tableaux linéaires, appelés couches. Généralement, il y a une couche d'entrée, une couche de sortie, ainsi qu'aucune, ou plusieurs couches cachées. La conception de la topologie du réseau implique de déterminer le nombre de nœuds à chaque couche, le nombre de couches dans le réseau et les différentes connexions entre les nœuds. Ces facteurs sont d'abord déterminés par intuition et optimisés par de multiples cycles d'expériences. Aussi, quelques méthodes rationnelles peuvent être utilisées pour concevoir un réseau neuronal. Par exemple, le réseau neuronal génétique utilise un algorithme générique pour résoudre certains problèmes concernant la topologie du réseau, par exemple la sélection de caractéristiques d'entrée [111, 112].

Il existe deux types de connexions entre les nœuds. La première est une connexion unidirectionnelle sans boucle de retour. L'autre est une connexion en boucle dans laquelle la sortie des nœuds peut être l'entrée des nœuds de niveau précédent ou de même niveau [113]. Sur la base du type de connexions susmentionné, les réseaux de neurones peuvent être classés

en deux types, à savoir les réseaux proactifs (feedforward) et les réseaux récurrents (feedback), comme il est montré dans les figures 3.5 (b) et (c).

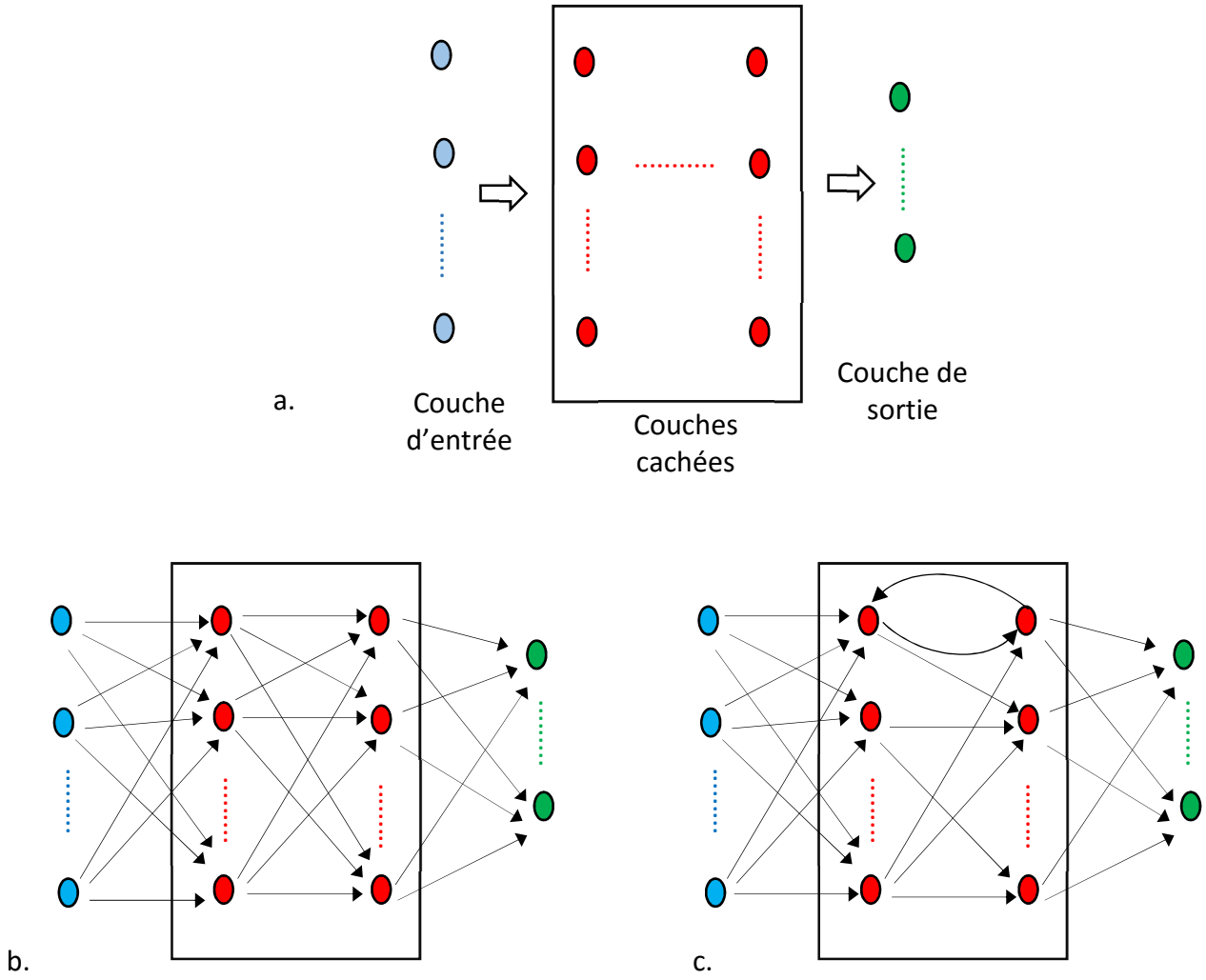


Figure 3.5. (a) Structure générale des réseaux de neurones artificiels. (b) Réseau de neurones perception. (c) Réseau de neurones récurrent [108].

Dans les réseaux proactifs (réseau statique), le signal ne circule que dans un seul sens, c'est-à-dire qu'une entrée est associée à une sortie particulière. Tandis que, dans les réseaux récurrents (réseaux dynamiques), pour une entrée, l'état du réseau récurrent se modifie durant chaque cycle, jusqu'à ce qu'il atteigne un point d'équilibre, ainsi une entrée produit une série de sorties. Le réseau de neurones de types perception est un réseau proactif largement utilisé [114]. Parmi les réseaux récurrents les plus connus, il existe le réseau de Hopfiel [115] et les cartes autoorganisées de Kohonen [116].

Les modes d'apprentissage de réseaux de neurones

La construction des réseaux de neurones artificiels implique l'utilisation des processus d'apprentissage automatique pour former ces réseaux. Pendant l'apprentissage, les poids sont ajustés aux valeurs souhaitées. L'apprentissage peut être classé dans trois formes de base [117] :

- **L'apprentissage supervisé** : dans ce type d'apprentissage, un ensemble d'exemples est fourni, constitué d'entrées et de sorties (cibles correspondantes). Les poids sont ajustés pour minimiser l'erreur entre la sortie de réseau et la sortie correcte. Lorsque le réseau produit les sorties souhaitées pour une série d'entrées, les poids sont fixes et le réseau peut être mis en service.
- **L'apprentissage non supervisé** : ce type d'apprentissage utilise un ensemble d'exemples sans les valeurs cibles. Le réseau tente de découvrir le modèle ou la tendance sous-jacente dans les données d'entrée seulement.
- **L'apprentissage par renforcement** : ce type d'apprentissage peut être considéré comme une forme intermédiaire des deux types d'apprentissages ci-dessus. Dans ce cas, la machine d'apprentissage agit sur l'environnement et reçoit une réponse de l'environnement. Le système d'apprentissage classe son action comme bonne (gratifiante) ou mauvaise (punissable) en fonction de la réponse environnementale et ajuste en conséquence ses paramètres.

Les réseaux perception

Un réseau perception est un réseau de neurones proactif (Feedforward), développé par Rosenblatt [118]. Un réseau perception à couche cachée unique a ses limites, il ne peut résoudre que des problèmes séparables linéairement. Par exemple, la fonction XOR qui ne peut pas être simulée avec le réseau perception monocouche [119]. Le réseau perception multicouche, comme le montre la figure 3.5(b), peut être utilisé pour approximer toutes les fonctions continues. Un algorithme de rétropropagation [120] est généralement utilisé dans l'apprentissage du réseau perception multicouche. Dans cet algorithme, l'entrée est d'abord propagée sur le réseau et la sortie est calculée. Ensuite, l'erreur entre la sortie calculée et la sortie correcte, appelée fonction de coût, est propagée en arrière de la sortie à l'entrée pour ajuster les poids. Mathématiquement, l'algorithme minimise la fonction de coût avec une méthode de descente de gradient, de sorte qu'il ne peut être appliqué qu'aux réseaux dotés de fonctions de transfert différentiables [108].

3.3. Les modèles et les tâches de fouille de données dynamique

L'utilisation de la fouille de données pour les bases de données statiques ne peut pas répondre aux exigences en temps réel ou aux données telles que les séries temporelles ou les flux de données. Par conséquent, les méthodes de la fouille de données dynamique sont rapidement apparues comme solutions pour ces catégories de problèmes. Comparé à la fouille de données statique, la fouille de données dynamique à une grande capacité à traiter des objets dynamiques qui sont réguliers ou irréguliers.

Parmi les tâches de la fouille de données dynamique, on trouve le clustering dynamique, la classification dynamique, la régression dynamique et la détection dynamique d'anomalies. Par conséquent, on note qu'il y a une forte correspondance entre les techniques de la fouille de données statiques et leurs versions dynamiques correspondantes, où les objectifs ou les types de connaissances à identifier sont les mêmes. Généralement, dans la fouille de données dynamique, les méthodes de la fouille de données (clustering, classification, etc.) sont modifiées pour s'adapter avec des ensembles d'objets dynamiques comme les séries temporelles et les flux de données. De plus, la fouille de données dynamique concerne également la mise à jour des modèles de fouille de données due aux changements instantanés apparus au niveau des bases de données (l'insertion, la modification et la suppression de données).

3.3.1. Le clustering dynamique

Le clustering de jeux de données volumineux concernant des systèmes dynamiques et complexes tels que les réseaux sociaux, le commerce électronique, les données de flux ou les trajectoires de vidéosurveillance représente un enjeu important dans le domaine de la fouille de données dynamique. Le clustering dans la fouille de données représente un domaine de recherche intéressant dans lequel on cherche à diviser un ensemble de données en un ensemble de groupes d'individus homogènes (clusters). Malgré l'existence de nombreux algorithmes qui abordent le problème du clustering, la complexité, l'instabilité et l'immensité de certains ensembles de données nécessitent la conception et le développement d'algorithmes de clustering dynamiques qui traite instantanément chaque modification au niveau des données analysées.

Dans le problème de clustering dynamique, l'ensemble de données est collecté pendant un intervalle de temps et peut être utilisé pour produire la première configuration de clusters.

Ensuite, l'état de cet ensemble de données est modifié en raison de l'ajout, de la suppression ou de la modification de données de l'ensemble de données, ce qui nécessite également une mise à jour dynamique de la configuration de clusters. Actuellement, on peut noter de nombreux travaux qui traitent le clustering de jeux de données dynamiques comme les flux de données [121]. Dans un autre exemple [122], un algorithme de clustering de flux de données est basé sur un système neuronal. Cet algorithme décrit un nouveau modèle de réseau de neurones artificiels qui possède des caractéristiques utiles, telles que la capacité d'apprentissage incrémental et la vitesse de fonctionnement élevée, ainsi que certaines fonctionnalités telles qu'une contre-mesure adaptative à l'instabilité des clusters, un mécanisme de rejet des données erronées et une mesure de similarité par paire.

D'autres types de clustering dynamique sont basés sur la modélisation à base d'individus ou à base d'agents. Ces méthodes utilisent la modélisation des enregistrements de données et des clusters en tant qu'ensemble d'individus indépendants pour résoudre des problèmes de clustering dynamique des données. À titre d'exemple, dans la méthode introduite [123], les enregistrements de données et les clusters sont représentés par des agents. Les agents enregistrements recherchent de façon proactive les clusters appropriés, tandis que les agents clusters changent leurs membres (enregistrements de données) en temps réel chaque fois qu'un nouvel élément de données arrive. De plus, Delope et Maravall [124] introduisent un algorithme de clustering de données utilisant des automates cellulaires linéaires. Ils considèrent les éléments de données individuels comme des cellules appartenant à un automate cellulaire unidimensionnel et utilisent les modèles de ségrégation sociale et la capacité des objets eux-mêmes à se déplacer de manière autonome dans le treillis. Ils utilisent aussi l'algorithme de colonie de fourmis pour distribuer les données aléatoirement dans le treillis.

3.3.2. Les règles d'association dynamiques

Parmi les méthodes qui traitent la fouille de règles d'association dans des bases de données qui se modifient durant le temps est la méthode[48]. L'approche proposée met à jour dynamiquement les connaissances obtenues lors du processus de fouille de données précédent. Les transactions sur une longue durée sont divisées en une série d'épisodes consécutifs et les connaissances obtenues au cours de l'épisode courant dépendent de l'ensemble des transactions courantes et de connaissance découverte au cours du dernier épisode. Cette approche découvre les règles d'association de données actuelles en utilisant les mises à jour qui se sont produites pendant l'épisode en cours, ainsi que les règles d'association de données qui ont été découvertes dans l'épisode précédent.

3.3.3. La classification et la régression dynamique

Les modèles de régression ou de classification doivent être mis à jour au fil du temps lorsque de nouveaux exemples d'apprentissage qui représentent des modèles différents sont apparus.

Une méthodologie proposée pour la mise à jour des modèles de prévision [125] basée sur la régression à vecteurs de support change constamment la composition des ensembles de données utilisés pour l'apprentissage, la validation et le test. L'idée de base est d'ajouter toujours les exemples les plus récents à l'ensemble de données d'apprentissage, afin d'extraire les informations les plus récentes sur les changements de motifs, en prenant en compte que la proportion de données de l'apprentissage par rapport à celles de la validation doit rester stable dans le temps. Dans ce qui suit, on va présenter les étapes de cette méthode de prévision.

Dans la méthode citée précédemment, nous devons tout d'abord identifier la durée des cycles de la série. À cette fin, une inspection graphique, une analyse par autocorrélation ou une analyse spectrale peuvent être effectuées. Ensuite, nous définissons un ensemble de données de test contenant au moins $k \geq 2$ cycles de longueur c donnée.

Ensuite, nous devons définir la configuration des ensembles de données d'apprentissage et de validation pour prédire le premier cycle de l'ensemble de données de test. Les données d'apprentissage contiennent deux parties, la première partie étant un ensemble de données passées (ou historiques) et la deuxième partie contenant les informations les plus récentes.

Pour prédire le deuxième cycle de l'ensemble de données de test, nous ajoutons le premier cycle de l'ensemble de données de test (ces données font maintenant partie du passé) à l'ensemble de données d'apprentissage et nous construisons un modèle prédictif différent, ce qui nous permet de nous assurer à nouveau que les informations les plus récentes ont une influence sur la construction du modèle.

Afin que la proportion de données appartenant aux ensembles d'apprentissage et de test reste stable dans le temps, nous déplaçons les données de la partie historique de l'ensemble de données d'apprentissage vers l'ensemble de données de test. La quantité de données à déplacer est déterminée en fonction de la proportion initiale de données dans chaque sous-ensemble. Cette même stratégie sera appliquée pour construire des modèles prédictifs pour le reste des cycles de l'ensemble de tests, ainsi que pour toute observation future.

D’autres méthodes de mise à jour des modèles de classification et de clustering qui permettent de détecter les changements dans les données de flux ont été présentées par d’autres travaux de recherche, par exemple [126] [127].

3.3.4. Les arbres de décision incrémentales

L’un des principaux inconvénients des algorithmes statiques de classification (par exemple, les arbres de décision) est qu'ils ne tiennent pas compte du temps auquel les données sont arrivées. Cela a incité les chercheurs à développer de nouvelles méthodes de classification incrémentale (par exemple, les arbres de décision incrémentales) qui sont mises à jour lorsque de nouvelles données arrivent, au lieu de réexécuter l'algorithme à partir de zéro [128].

Les algorithmes incrémentales semblables à l’algorithme illustré à la figure 3.7 mettent à jour leurs modèles prédictifs lorsque de nouvelles données apparaissent à différents moments dans le temps, contrairement aux algorithmes de classification habituelles (statiques) qui construisent des modèles par lots (figure 3.6). De plus, la classification incrémentale permet d’extraire de nouvelles informations (modèles) à partir d'un ensemble de données futures qui seront disponibles ultérieurement sans perdre au préalable les connaissances du domaine étudié[129].

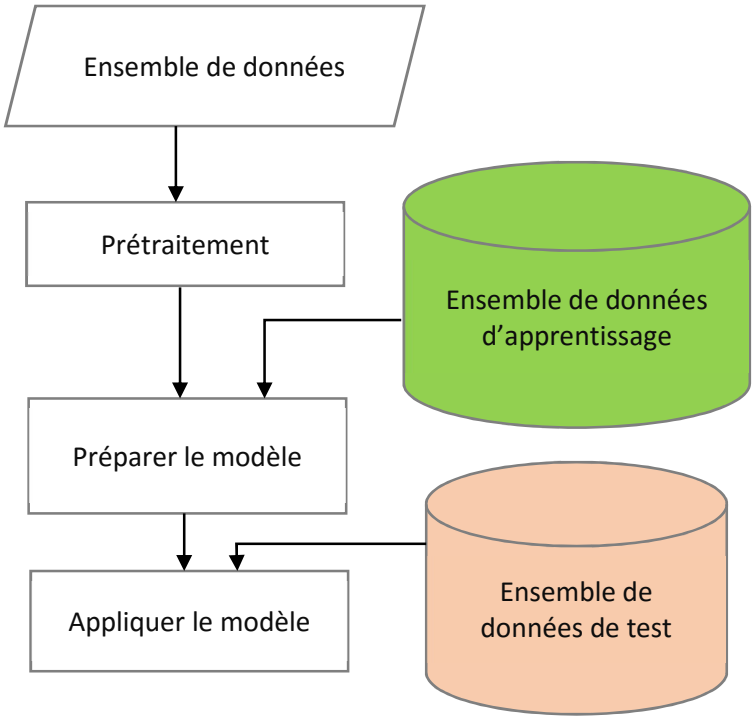


Figure 3.6. Modèle de classification traditionnel [128]

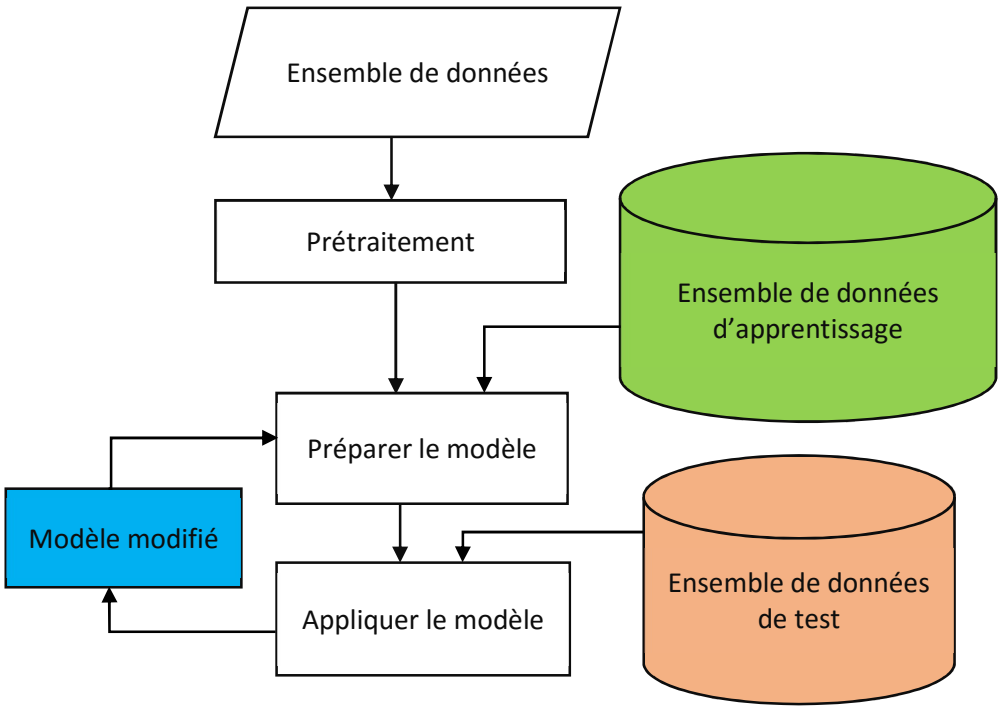


Figure 3.7. Modèle de classification incrémentale [128]

Les méthodes d'arbre de décision incrémentales permettent à un arbre existant d'être mis à jour en utilisant seulement de nouvelles instances de données individuelles, sans avoir à retraiter les instances passées. Cela peut être utile dans les situations suivantes [130, 131] :

- L'ensemble de données n'est pas disponible lorsque l'arbre est mis à jour (c'est-à-dire que les données n'ont pas été stockées).
- L'ensemble de données d'origine est trop grand pour être traité d'un seul coup.
- Les caractéristiques des données changent au fil du temps.

Dans les paragraphes suivants, on présente quelques méthodes d'induction d'arbres de décision incrémentales et non incrémentales. Ces méthodes sont :

- ID5R [130] présente un algorithme incrémental pour induire des arbres de décision équivalents à ceux formés par l'algorithme ID3[103].
- ITI [132] est une autre méthode efficace pour induire progressivement des arbres de décision. Le même arbre est produit pour un ensemble de données, quel que soit l'ordre de présentation des données (incrémentale ou en mode batch). Cette méthode peut aussi accueillir des variables numériques, des tâches multi-classes et des valeurs manquantes.
- CART [133] est un algorithme d'arbre de décision non incrémental pour les problèmes de classification et de régression. CART incrémental [134] représente un CART modifié pour incorporer les données de façon incrémentale.
- L'apprentissage très rapide des arbres de décision VFDT [135] (Very Fast Decision Trees Learner) réduit le temps d'apprentissage pour les grands ensembles de données incrémentales en sous-échantillonnant le flux de données entrant. Tandis que, l'algorithme VFDTc [136] étend l'algorithme VFDT pour les données continues, la dérive conceptuelle et l'application des classificateurs naïve bayes dans les feuilles.

3.3.5. Les modèles de Markov cachés

Dans les modèles de Markov simples comme les chaînes de Markov, les états de modèle sont directement visibles pour l'observateur, et donc les probabilités de transition d'états sont les seuls paramètres. Alors que dans les modèles de Markov cachés HMM (Hidden Markov Model), les états de modèle ne sont pas directement visibles, mais les sorties (les données ou les symboles) associées aux états sont visibles. Chaque état a une distribution de probabilité sur les jetons de sortie possibles. Par conséquent, la séquence de jetons générée par un modèle de Markov caché donne des informations sur la séquence d'états. L'adjectif « caché » se réfère à la séquence d'états à travers laquelle le modèle passe, et non aux paramètres du modèle ; le

modèle est encore appelé modèle de Markov caché, même si ces paramètres sont connus exactement [137].

Récemment, les modèles de Markov cachés ont été généralisés aux modèles de Markov en paire PMM (Pairwise Markov Models) et aux modèles de Markov triplet TMM (Triplet Markov Models). Ces modèles de Markov cachés permettent la prise en compte de structures de données plus complexes [138, 139] et la modélisation de données non stationnaires [140, 141].

Nous définissons un HMM comme un 5-tuple (S, V, Π, A, B) [142], où $S = \{s_1, s_2 \dots, s_N\}$ est un ensemble de N états, $V = \{v_1, v_2 \dots, v_M\}$ est un ensemble de M symboles possibles dans un vocabulaire donné, $\Pi = \{\pi_i\}$ sont les probabilités d'état initiales, $A = \{a_{ij}\}$ sont les probabilités de transition d'état et B = $\{b_i(v_k)\}$ sont les probabilités de sortie ou d'émission.

Nous utilisons $\lambda = (\Pi, A, B)$ pour désigner tous les paramètres. La signification de chaque paramètre est la suivante :

- π_i :la probabilité que le système commence à l'état i au début.
- a_{ij} : la probabilité de passer de l'état i à l'état j.
- $b_i(v_k)$: la probabilité de générer le symbole v_k à l'état i.

De plus, nous avons les contraintes suivantes :

$$\begin{cases} \sum_{i=1}^{N} \pi_i = 1 \\ \sum_{j=1}^{N} a_{ij} = 1 \quad pour\ i = 1,2, \dots, N \\ \sum_{k=1}^{M} b_i(v_k) = 1 \quad pour\ i = 1,2, \dots, N \end{cases} \tag{3.6}$$

Les modèles de Markov cachés sont particulièrement connus pour leurs applications dans l'apprentissage par renforcement et la reconnaissance des formes temporelles, telles que l'écriture manuscrite, la reconnaissance des gestes, le marquage partiel de la parole, le suivi de la partition musicale et la bio-informatique [143] [144].

3.3.6. Les réseaux bayésiens dynamiques

L'étude du comportement temporel de tout système représente un enjeu fondamental dans tous les domaines. Par exemple, l'analyse de séries temporelles qui vise à analyser les sorties d'un processus stochastique représente un champ approprié pour appliquer une version dynamique des algorithmes de réseau bayésien.

La probabilité de jointure dans le réseau bayésien dynamique DBN (Dynamic Bayesian Network) pourrait être utilisée afin de modéliser les dépendances entre un ensemble de processus stochastiques. Cette probabilité est formulée comme suit [145] :

$$P(x[1], x[2], \dots x[T]) = P(x[1]) \prod_{t=2}^{T} \prod_{i=1}^{N} P(x_i[t] | Parents(x_i[t])) \quad , \qquad (3.7)$$

où $x_i[t]$ représente la variable identifiée par i à un instant t et $Parents(x_i[t])$ représente les parents de cette variable à cet instant t.

Nous pouvons distinguer deux groupes principaux de réseaux de neurones dynamiques, à savoir les réseaux de neurones dynamiques homogènes et les réseaux de neurones dynamiques non homogènes. Dans le premier groupe, ni les arcs ni leurs directions dans le réseau ne seront modifiés (invariants dans le temps). Tandis que les réseaux de neurones dynamiques non homogènes sont à l'opposé de ceux qui sont homogènes [146-152] , où les paramètres et la structure peuvent changer avec le temps.

Afin de générer un réseau de neurones dynamique, il est recommandé de sélectionner uniquement les variables les plus importantes. Parmi les méthodes utilisées à cette fin, il y a la méthode LASSO [153] (Least Absolute Shrinkage and Selection Operator) qui est utilisée dans divers réseaux de neurones dynamiques [154, 155]. Cette méthode d'analyse de régression aide à la sélection de variables afin de construire un modèle plus précis.

Le problème LASSO peut être représenté par le modèle suivant [156]:

$$\begin{cases} \hat{\beta}^{LASSO} = argmin_{\beta} \left\{ \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{P} x_{ij} \beta_j \right)^2 \right\} \\ \text{sous la contrainte } \sum_{j=1}^{P} |\beta_j| \leq \vartheta \end{cases} \qquad (3.8)$$

L'idée de base de cette méthode est de minimiser l'erreur de prédiction et en même temps ne pas violer la contrainte imposée à la somme des coefficients β_i . En choisissant de petites valeurs pour le seuil ϑ, certains des coefficients auront tendance à $zéro$ (rétrécissement), ce qui conduit à exclure les prédicteurs correspondants.

3.3.7. Les réseaux de neurones dynamiques

Les réseaux de neurones récurrents RNN (Recurrent neural network) représentent une classe de modèles informatiques conçus par analogie avec le cerveau. Ce qui les distingue des réseaux de neurones plus connus, c'est l'existence de cycles fermés dans la topologie de connexion. À cause de ces cycles, les réseaux de neurones récurrents peuvent présenter une dynamique autonome en l'absence de tout apport [157]. Parmi les applications de réseaux de neurones récurrents, il y a la reconnaissance de la parole [158-160] et les séries temporelles [161-163]. Dans ce qui suit, on va présenter quelques réseaux de neurones récurrents les plus connus.

Réseaux de Hopfield

Le réseau de Hopfield est un réseau récurrent comportant une seule couche. Un réseau de Hopfield à trois nœuds est présenté à la figure 3.8. Dans le réseau de Hopfield, chaque nœud est connecté à tous les autres nœuds (entièrement connecté), mais pas à lui-même. À partir d'une entrée, l'état du réseau de Hopfield est mis à jour jusqu'à ce qu'il converge vers un état d'équilibre, appelé état attracteur. Étant donné que toute entrée mène éventuellement le réseau vers l'un des états attracteurs, le réseau de Hopfield peut être utilisé comme mémoire associative pour récupérer des motifs partiellement corrompus. Le problème du voyageur de commerce peut être résolu avec le réseau de Hopfield [115].

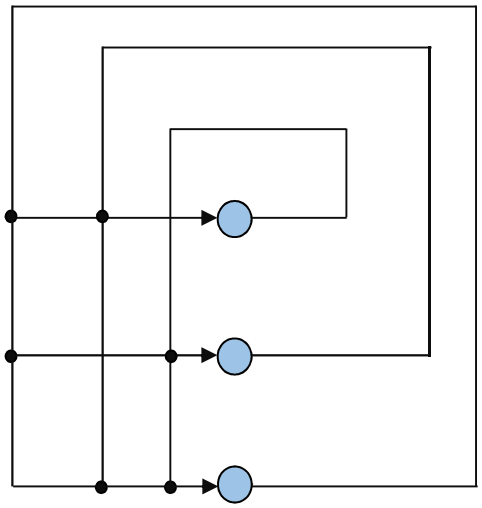


Figure 3.8. Réseau de Hopfield

Cartes de Kohonen

La carte de Kohonen [116] est un autre type de réseau récurrent. Ce réseau a une seule couche cachée. Les nœuds de la couche cachée sont généralement organisés en grille rectangulaire bidimensionnelle, comme le montre la figure 3.9. Chaque nœud de la couche cachée est connecté à tous les nœuds d'entrée. La carte de Kohonen utilise l'apprentissage non supervisé avec la règle d'apprentissage du gagnant qui prend tout. Seuls le nœud ayant la réponse la plus élevée et ses nœuds voisins obtiennent la mise à jour de leur poids, de sorte qu'ils sont plus susceptibles de répondre à des modèles d'entrée similaires. Les cartes de Kohonen peuvent être utilisées pour la projection de données de haute dimension dans un espace bidimensionnel et le clustering de données.

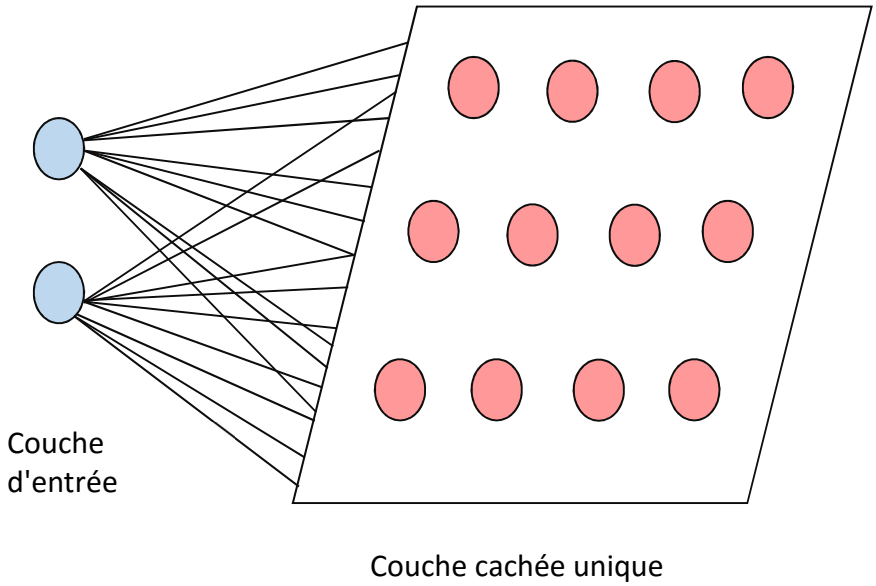


Figure 3.9. Carte de Kohonen

Réseaux Elman et réseaux Jordan

Un réseau d'Elman est un réseau à trois couches avec une couche de contexte (comme illustré dans la figure 3.10). La couche cachée est reliée à ces unités de contexte [164]. À chaque itération, l'entrée est retransmise et une règle d'apprentissage est appliquée. Les connexions finales sauvegardent une copie des valeurs précédentes des unités cachées dans les unités de contexte (puisqu'elles se propagent sur les connexions avant l'application de la règle d'apprentissage). Ainsi, le réseau peut maintenir une sorte d'état, lui permet d'effectuer des tâches telles que la prédiction séquentielle qui dépassent la puissance d'une perception multicouche standard.

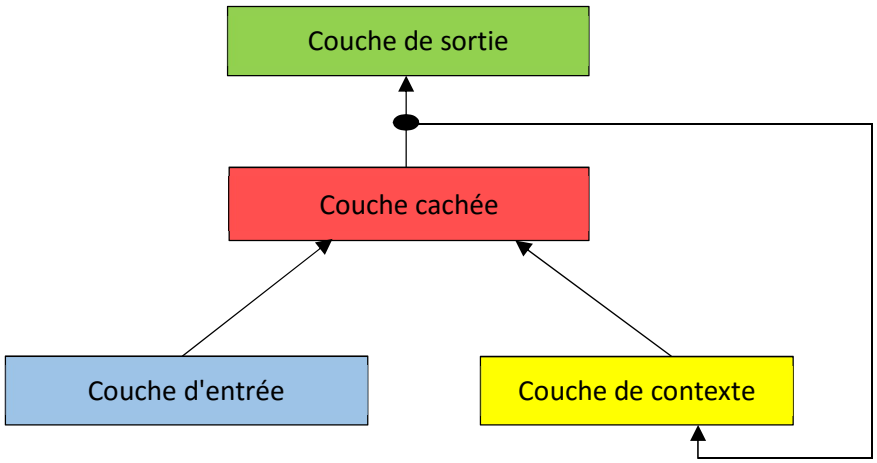


Figure 3.10. Réseau Elman

Les réseaux Jordan sont similaires aux réseaux Elman. Néanmoins, les unités de contexte sont alimentées à partir de la couche de sortie au lieu de la couche cachée intermédiaire (comme illustré dans la figure 3.11). Les unités de contexte d'un réseau Jordan constituent également la couche d'état et elles peuvent avoir des connexions récurrentes à elles-mêmes [164].

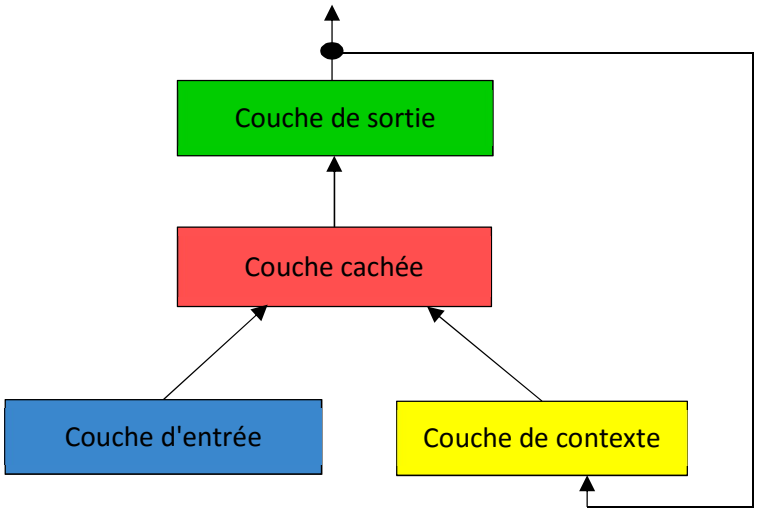


Figure 3.11. Réseau Jordan

Réseaux de neurones récurrents pour la modélisation des systèmes dynamiques

Mathématiquement, les réseaux de neurones récurrents sont un type des systèmes dynamiques. Non seulement le cerveau est caractérisé par une connectivité massivement récurrente, mais cette connectivité également est dynamique et continue.

En effet , les systèmes dynamiques eux-mêmes représentent actuellement un pilier de la neuroscience computationnelle [165]. L'activité persistante des réseaux de neurones biologiques est supposée résulter des modifications dynamiques dans l'espace d'états des neurones [166], le calcul dynamique sous-tend une variété de modèles pour le traitement de l'information et le fonctionnement de la mémoire dans le cerveau [167-169].

Motivés par ces faits et d'autres intérêts, divers chercheurs ont développé des techniques permettant de former des réseaux récurrents à l'approximation de systèmes dynamiques. Cet ensemble de réseaux récurrents comprend la rétropropagation à travers le temps [170], l'apprentissage récurrent en temps réel [171], le filtre de Kalman étendu [172], le calcul des réservoirs [157] et l'apprentissage phase-space [173].

Dans ce qui suit, nous introduisons un réseau récurrent construit pour la modélisation d'une classe générale de systèmes dynamiques[174]. Le réseau est destiné à la modélisation de processus réels dans lesquels des mesures expérimentales des variables externes et des variables d'état sont obtenues à des instants discrets. Pour cet exemple, on va présenter principalement les différentes couches de ce réseau, les connexions entre ses nœuds et la façon de modéliser des systèmes dynamiques en utilisant ce type de réseau. Plus de détails peuvent être trouvés dans [174], par exemple, les poids des connexions, les nœuds cachés et le mode d'apprentissage.

De nombreux processus du monde réel peuvent être représentés comme des systèmes dynamiques. Un système dynamique peut être décrit en termes d'évolution d'une ou plusieurs variables d'état en réponse à une ou plusieurs variables externes.

Dans ce qui suit, nous examinerons les systèmes dynamiques qui peuvent être modélisés à l'aide de l'équation suivante :

$$\frac{\partial v(\tau)}{\partial \tau} = F(v(\tau), x(\tau)) \quad , \qquad (3,9)$$

où x est un vecteur de variables externes, v est un vecteur de variables d'état et F une fonction non linéaire. D'après cette définition, les variables externes sont indépendantes de manière causale des variables d'état. Étant donné $v(\tau = 0)$ un ensemble de conditions initiales et $x(\tau)$ la séquence temporelle des variables externes, l'équation (3,9) détermine l'évolution

des variables d'état ; c'est-à-dire $v(\tau)$, où $\tau > 0$. Donc, le problème abordé est l'apprentissage de la séquence temporelle des variables d'état.

Ce réseau récurrent est basé sur la solution de l'équation différentielle de premier ordre (3,9) en utilisant l'extension de Taylor autour du point $v(\tau - \delta\tau)$. Cette solution est illustrée dans l'équation suivante :

$$v(\tau) = v(\tau - \delta\tau) + \frac{\partial v(\tau - \delta\tau)}{\partial \tau} \delta\tau \qquad (3,10)$$

Les séparations entre les périodes de mesure $\delta\tau$ sont supposées suffisamment petites pour supprimer les termes supplémentaires de l'ordre $(\delta\tau)^2$.

Cette solution est modélisée à l'aide du réseau récurrent illustré à la figure 3.12. Ce réseau récurrent peut être considéré comme un réseau divisé en deux parties. La première partie est un réseau proactif (feedforward) standard qui implémente directement la fonction *F* décrite par l'équation (3.9). Les entrées de cette étape $x(\tau)$ sont les entrées externes, les variables $v(\tau)$ sont les variables d'état à une certaine période τ et les variables $y(\tau)$ sont les sorties qui représentent les dérivées temporelles des variables d'état qui correspondent à la période τ. La relation entre ces variables est décrite comme suit :

$$y(\tau) = F(v(\tau), x(\tau)) \quad , \qquad (3,11)$$

La deuxième partie du réseau est la partie récurrente et elle met en œuvre l'équation 3.10 via les connexions un-à-un depuis les sorties aux variables d'état. Le cycle se répète ensuite pour chaque période dans laquelle des entrées externes sont définies.

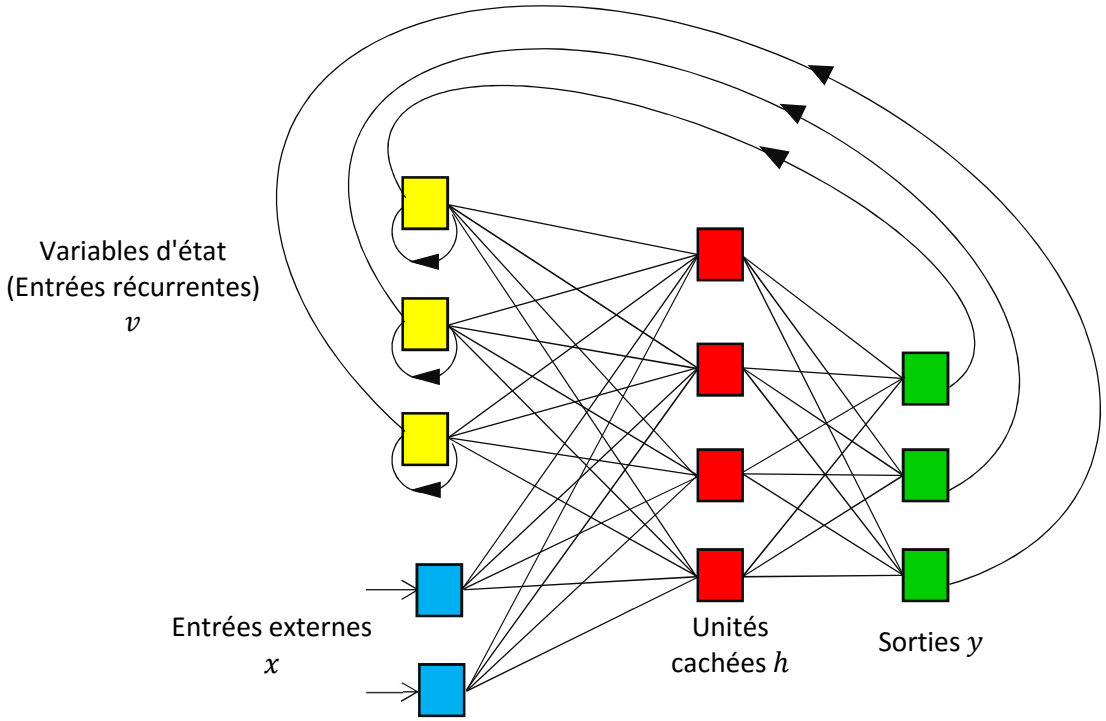


Figure 3.12. Architecture d'un réseau de neurones récurrent pour la modélisation d'un système dynamique [174]

Tout ce qui se passe dans la première phase du réseau le fait à la phase $t-1$. La partie récurrente du réseau donne les valeurs des variables d'état qui correspondent à la phase suivante t . Ce principe est illustré dans l'équation suivante :

$$v(t) = v(t-1) + y(t-1)\,\Delta(t)\text{ , où }\Delta(t) = \tau_t - \tau_{t-1} \qquad (3.11)$$

Donc, les entrées externes et les variables d'état qui correspondent à la période $t-1$ donnent naissance aux variables d'état qui correspondent à la période t.

Ce réseau produira une séquence complète des variables d'état, $v(1), v(2), \ldots, v(T)$, en fonction des conditions initiales des variables d'état, $v(0)$, la séquence d'entrées externes, $x(0), x(1), \ldots, x(T-1)$ et les pas de temps $\Delta(1), \Delta(2), \ldots, \Delta(T)$.

3.4. Conclusion

Dans ce chapitre, on a vu un aperçu sur un ensemble de modèles et de tâches qui peuvent être utilisés dans le processus de fouille de données, comme les réseaux de neurones artificiels,

les réseaux bayésiens et les arbres de décision. Ainsi qu'un aperçu sur un ensemble de modèles et de tâches dynamiques qu'on peut utiliser dans le processus de fouille de données dynamique. Dans le chapitre suivant, on va présenter le modèle de simulation sociale à base d'agents qui représente un modèle dynamique. Ce modèle nous permet après l'atteinte de son état stable d'avoir l'ensemble de données nécessaires pour une étude de fouille de données dynamique.

Chapitre 4 : Fouille de données dynamique basée sur la simulation sociale à base d'agents

4.1. Introduction

Étant donné que l'obtention de l'ensemble de données représente un défi important pour une étude de fouille de données, surtout dans le cas où les données ne sont pas complètes ou ne sont pas encore collectées, il est nécessaire d'avoir des méthodes appropriées pour résoudre tels problèmes. La solution que l'on propose est l'utilisation d'un modèle dynamique qui peut générer cet ensemble de données. Ce type de modèle dynamique est un modèle de simulation sociale à base d'agents.

L'un des principaux défis de la simulation sociale est l'étude de l'émergence de macro-propriétés à partir de micro-interactions ou interactions à un niveau inférieur [175, 176]; c'est-à-dire clarifier le lien entre les phénomènes sociaux considérés au niveau d'une société donnée et les phénomènes locaux considérés au niveau des individus de cette société. Un autre défi majeur est de présenter l'aspect dynamique de la simulation, c'est-à-dire étudier l'importance de l'évolution du système en tant que système dynamique et le rôle du temps dans les phénomènes sociaux étudiés.

Pour surmonter ces défis, nous pouvons utiliser la simulation à base d'agents sociaux pour construire un modèle dynamique de simulation sociale qui peut refléter très bien les comportements d'une société donnée en fonction de ceux de ses individus. En outre, l'application de la méthode de Monte-Carlo sur le modèle de simulation sociale à base d'agents à des fins expérimentales et analytiques peut aider à mieux comprendre les systèmes étudiés.

4.2. La simulation informatique

Selon Shannon [177], la simulation informatique est le processus de conception d'un modèle informatique qui simule un système, puis effectue des expériences en utilisant ce modèle pour comprendre le comportement du système réel et évaluer de différentes stratégies d'exploitation.

La simulation informatique comporte généralement les tâches suivantes [178] :

a. Identifier le système à étudier.
b. Construire un modèle informatique (modèle opérationnel) qui simule ce système.
c. Mener des expériences en utilisant le modèle opérationnel.

d. Analyser les résultats d'expériences réalisés afin de parvenir à une meilleure compréhension du système modélisé.

4.3. La simulation sociale

Les systèmes sociaux désignent généralement des organisations d'individus divisées en groupes ou structures ayant des fonctions, des caractéristiques, des origines ou des statuts différents. Les sciences sociales sont une catégorie de disciplines académiques qui s'intéressent à la société et aux relations entre les individus au sein d'une société. Les sciences sociales dans leur ensemble ont de nombreuses branches, dont chacune est considérée comme une science sociale, et elles comprennent, sans toutefois s'y limiter, l'anthropologie, l'archéologie, l'économie, l'histoire, la géographie humaine, la linguistique, les sciences politiques, la psychologie et la santé publique [179].

Avec la popularité croissante des logiciels sociaux et l'intérêt académique accru pour l'analyse des réseaux sociaux, l'informatique sociale a attiré une attention croissante ces dernières années. L'informatique sociale adopte une approche informatique pour l'étude et la modélisation des interactions sociales [180, 181]. En tant que domaine de recherche de l'informatique sociale, la simulation informatique de phénomène social est un domaine de recherche prometteur situé à l'intersection des sciences sociales, des sciences mathématiques et de l'informatique [182]. Donc, on peut considérer que la simulation sociale est la simulation de phénomènes ou d'objets sociaux (société, organisations, marchés, êtres humains) qui sont réalisés par ordinateur.

En effet, les sciences sociales et naturelles font grand usage de la simulation informatique pour expliquer, prévoir et étudier certains phénomènes [176], ainsi que la validation de théories scientifiques ou l'étude d'un système réel [178]. Pour ce faire, il faut établir des modèles qui fonctionnent sur ordinateur, mais en respectant les descriptions et surtout les contraintes des systèmes étudiés.

En outre, la simulation est un excellent moyen pour modéliser et comprendre les processus sociaux, ainsi que pour prédire l'avenir, c'est-à-dire développer un modèle qui reproduise fidèlement le côté dynamique d'un comportement donné, puis simuler le passage du temps, afin d'obtenir des informations qui ne sont pas disponibles précédemment. La simulation sociale en général apporte plusieurs avantages aux recherches dans les domaines sociaux, car elle permet de :

- Simuler et étudier les phénomènes qui pourraient résulter de l'analyse sociale.
- Concevoir des modèles basés sur des phénomènes observés dans la nature, où une société donnée peut être modélisée explicitement avec des entités indépendantes.
- Étudier des propriétés difficiles à observer dans la nature.
- Prédire certaines situations qui ne peuvent pas être calculées directement.
- Comparer deux systèmes ou plus.
- Évaluer des systèmes réels ou virtuels.

4.4. La modélisation à base d'agents

La modélisation à base d'agents ABM (Agent Based Modeling) est utilisée pour modéliser de nombreux systèmes dynamiques et complexes, notamment ceux qui incluent des individus autonomes, tels que des sociétés humaines, des sociétés animales, des robots, des sociétés d'insectes. Elle nous permet de représenter l'évolution d'un ensemble d'agents dans un environnement où l'organisation et l'interaction entre agents sont essentielles.

4.4.1. La structure d'un modèle à base d'agents

Principalement, un modèle à base d'agents consiste en un ensemble organisé d'agents interagissant dans un environnement commun. Ce système a également une frontière clairement définie avec des entrées et des sorties bien définies. Pour développer un modèle à base d'agents, il faut identifier, modéliser et programmer ses éléments. Un modèle à base d'agents comporte généralement trois éléments [183] :

a. Un ensemble d'agents, leurs attributs et leurs comportements.
b. L'ensemble des relations et des méthodes d'interaction entre les agents, qui permettent de définir la topologie sous-jacente de la connectivité, c'est-à-dire définir comment et avec qui les agents interagissent.
c. L'environnement des agents où ils se situent. Également, les agents interagissent avec leur environnement en plus avec d'autres agents.

Après l'identification du modèle conceptuel, il est nécessaire de construire et d'exécuter le modèle opérationnel qui permet de simuler les comportements et les interactions des agents, en utilisant une plateforme de modélisation à base d'agents ou un langage de programmation.

Pour exécuter un modèle à base d'agents, les agents doivent exécuter de manière répétée leurs comportements et leurs interactions. Ce processus fonctionne souvent sur la base d'un compteur de temps comme dans des structures de simulation à événements discrets.

La structure d'un modèle basé sur les agents est illustrée à la figure 4.1 dont chacune des composantes est discutée dans la présente section.

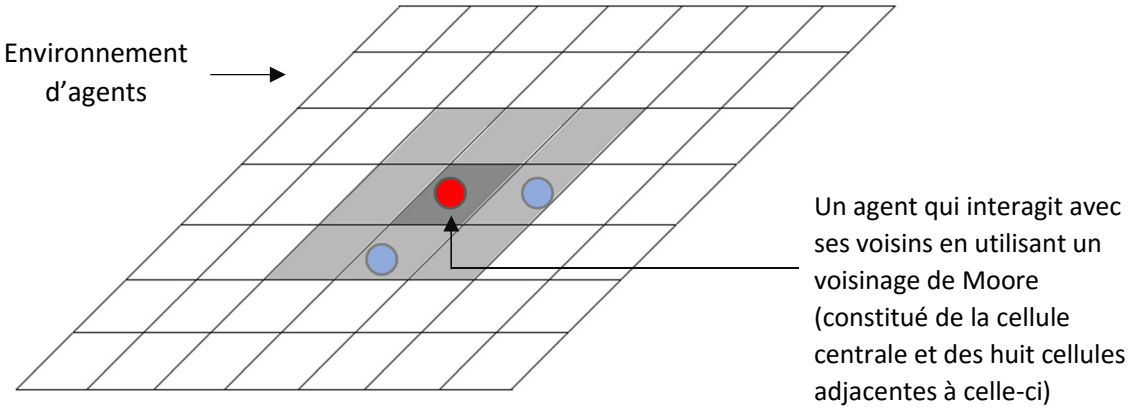


Figure 4.1. Structure d'un modèle à base d'agents [183]

a. Un ensemble d'agents autonomes

Un agent peut être considéré comme un système informatique situé dans un environnement donné, et qui est capable d'agir de manière autonome dans cet environnement, afin d'atteindre les objectifs qui lui ont été délégués [184]. Selon une autre définition, on considère un agent comme étant une entité logicielle ou matérielle située dans un environnement virtuel ou réel, et qui possède certaines caractéristiques, par exemple, l'agent peut agir dans son environnement et se gérer par des objectifs individuels propres, des buts, des motivations ou des fonctions de satisfaction [185].

Chaque agent possède un ensemble d'attributs qui décrivent son état et un ensemble de comportements spécifiques (règles régissant son comportement) qui définissent comment un agent se comporte en réaction à des changements dans son environnement et, éventuellement, à un ensemble de buts ou d'objectifs.

Par exemple, dans un scénario d'évacuation d'un bâtiment, nous voudrions peut-être comprendre comment les gens pourraient évacuer ce bâtiment. Donc, on peut modéliser ces individus en tant qu'agents ayant plusieurs attributs, comme la vitesse de déplacement, la position dans le bâtiment et le degré de perturbation. Les comportements des individus comportent les différentes règles d'action ou réaction, comme les règles de déplacement ou les

règles d'interaction avec l'environnement ou les autres individus. Les objectifs comportent par exemple les différentes stratégies utilisées par ces individus pour éviter le danger et atteindre la porte de sortie.

b. Un ensemble de relations entre les agents

Chaque relation définit comment un agent interagit avec d'autres agents ou avec son environnement. Cela implique également la manière dont chaque agent est connecté à d'autres agents, c'est-à-dire une « topologie de connexion sous-jacente », par exemple, la manière dont les personnes interagissent entre eux lorsqu'elles tentent de quitter le bâtiment.

Les relations et les limites entre agents doivent être clairement spécifiées. La raison en est que les agents doivent être autonomes. C'est-à-dire que l'agent doit être capable de prendre ses décisions en fonction de son propre état et de celui de son environnement. Les relations ou les interactions entre agents peuvent être simples ou extrêmement complexes.

Quelques règles et facteurs généraux de modularité tels que le couplage et la cohésion qui existent en génie logiciel peuvent être utilisés. Par exemple, si deux types d'agents ont une relation extrêmement complexe et étroitement couplée dans laquelle leurs limites fonctionnelles sont difficiles à définir, les deux types d'agents pourraient alors être mieux conceptualisés comme un agent unique.

Tous les agents ne doivent pas interagir avec tous les autres agents. Si un agent doit prendre une décision basée sur l'état d'un autre agent, il doit interagir avec cet agent pour le découvrir.

c. L'environnement des agents

C'est le « monde » dans lequel les agents existent, c'est-à-dire l'ensemble minimum de variables ou de structures « globales » nécessaires pour définir comment les agents réagissent à leur environnement (par exemple, l'alarme incendie, le bâtiment où se trouvent les personnes et la capacité de chaque couloir). L'environnement est l'élément du système avec lequel les agents interagissent et il n'est pas généralement considéré comme un agent à part entière, c'est-à-dire qu'il est passif et global (il n'affirme pas activement un comportement et il affecte potentiellement tous les agents). Il peut avoir une limite simple ou complexe, selon le système modélisé.

4.4.2. L'analyse et la conception des modèles à base d'agents

Comme la simulation à base d'agents est un processus complexe qui nécessite des études approfondies avant de commencer à rassembler et analyser les résultats, un modèle bien conçu pour un tel processus serait plus que nécessaire. Parmi les méthodes d'analyse et de conception qui nous permettent de décrire les modèles à base d'individus IBM (Individuals Based Model) ou les modèles à base d'agents ABM (Agents Based Model) est la méthode [186]. Cette méthode se base sur un protocole standard à trois blocs, dénoté par ODD (Overview Design Details).

Dans la méthode citée précédemment, en se basant premièrement sur la vue d'ensemble du modèle, puis sur les concepts de conception (Design) et finalement sur les détails. Dans un tel protocole, le premier bloc « Vue » vise à donner l'objectif et la structure globale du modèle à base d'agents. Les concepts du Design de modèle dans ce protocole, comme son nom l'indique, servent à clarifier les concepts généraux du modèle. Après avoir spécifié la vue et les concepts de Design du modèle, les données d'initialisation, les données d'entrée et les sous-modèles du modèle global sont rassemblés dans la partie Details de ce protocole.

Dans l'article [187], l'auteur discute les principes relatifs à la modélisation à base d'agents. Selon l'auteur, il est essentiel d'avoir une image claire du processus de modélisation, c'est-à-dire en décrivant, expliquant et justifiant le but, les mécanismes, les agents et leurs comportements dans le système. Afin de valider le modèle conçu, les résultats de ce dernier doivent être comparés de manière rigoureuse à des preuves empiriques.

La conception de modèles à base d'agents a toujours été un domaine de recherche actif, où il est très important de suivre les meilleures normes, afin d'aider à la conception de bons modèles à base d'agents pour les projets socio-environnementaux. Étant donné que la modélisation consiste à créer une abstraction d'un système du monde réel; Doran J.E [188] considère que définir le meilleur niveau d'abstraction et d'agrégation est très important pour pouvoir se concentrer uniquement sur l'objectif du modèle. En outre, l'auteur a exprimé le même point de vue que [187] sur la définition des agents du modèle qui doit être parmi les premières choses à faire, où l'auteur souligne que la cognition et l'architecture de l'agent doivent être bien prises en compte.

Dans leur travail, García-Magariño et al. [189] ont proposé un processus itératif (spécification des exigences, développement, évaluation et déploiement) appelé PEABS (Process for developing Efficient Agent-Based Simulators) pour aider au développement de simulateurs à base d'agents dans lesquels ils mettent l'accent sur l'efficacité.

Nikolic et Ghorbani [190] ont travaillé sur des systèmes sociotechniques, où ils ont proposé une méthode pour développer de tels systèmes. Leur méthode est décomposée en un ensemble d'étapes, à savoir l'analyse (identification et conceptualisation du but et du système), la conception de modèles qui comprend l'identification des agents et de leur comportement, la mise en œuvre de logiciels et l'évaluation des modèles.

Dans leur article [191], García-Magariño et al. proposent une nouvelle approche pour le développement des simulateurs à base d'agents ABS (Agent-Based Simulators). La technique proposée pourrait guider les modélisateurs dans la conception et la mise en œuvre des processus décisionnels des agents dans des scénarios non déterministes. Ces ABS sont déployés à la fois comme des applications mobiles et des outils en ligne. L'approche proposée est expliquée avec deux études de cas dans les domaines de la santé et du tourisme.

En fait, la conception de logiciels de simulation à base d'agents a été au cœur de plusieurs travaux au cours des dernières années [192-196]. Afin d'aider la communauté de simulation à base d'agents à réaliser des expériences efficaces, un nombre important de simulateurs à base d'agents ont déjà été proposés et mis en service. Nous en verrons quelques-unes dans ce qui suit.

Le simulateur à base d'agents piloté par des cartes auto-organisatrices, dénoté par SOM (Self-Organizing Maps), proposé par Resta et al. [197]. Selon les auteurs, dans ce simulateur, les agents peuvent apprendre et interagir, où l'impact de leurs interactions sur le comportement global et l'évolution du système économique, ainsi que l'impact des connexions spatiales modélisées entre les agents sur la décision individuelle, pourraient également être remarqués. Ces caractéristiques aideraient à détecter et étudier quelques modèles intéressants.

En outre, le simulateur à base d'agents pour les itinéraires touristiques urbains ABSTUR (Agent-based Simulator for Tourist Urban Routes) de Garcia-Maganrino [198] a été conçu pour aider à choisir les meilleurs itinéraires touristiques, en se basant sur des informations des itinéraires et de différents types de touristes. Ce système simule le nombre de personnes qui peuvent choisir certains itinéraires en fonction de leurs caractéristiques, ce qui pourrait aider à éviter les itinéraires surchargés et à économiser du temps et de l'argent.

4.5. La simulation sociale à base d'agents

La simulation en sociologie permet de modéliser certains phénomènes observés dans une société donnée. L'application de la modélisation à base d'agents pour la simulation de

phénomènes sociaux est généralement associée à la tendance sociologique de l'individualisme, qui considère l'individu comme l'unité de base [199]. De plus, l'expressivité du modèle basé sur les agents facilite le dialogue avec les non-spécialistes des sciences sociales, et permet d'inclure explicitement des modèles d'individus au sens sociologique dans le système modélisé.

L'application de la modélisation à base d'agents dans la simulation sociale est appelée la simulation sociale à base d'agents ABSS (Agent-Based Social Simulation) [175, 200, 201]. Par conséquent, comme le montre la figure 4.2, ce type de simulation représente une zone d'intersection de la simulation informatique, la modélisation à base d'agents et les sciences sociales.

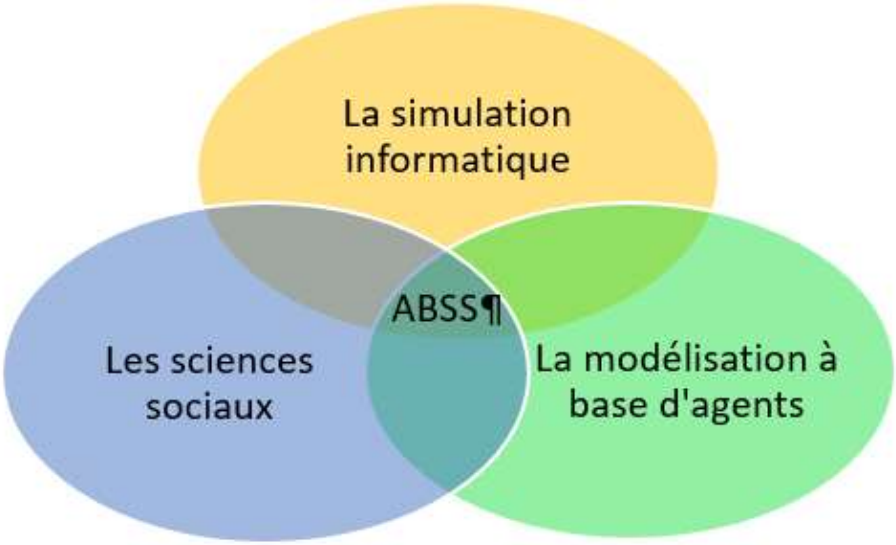


Figure 4.2. Les trois domaines constituant l'ABSS [175]

À titre d'exemple, les systèmes d'évacuation résultant de catastrophes comme les incendies, les tremblements de terre et les inondations, sont des types de systèmes sociaux qui comprennent des personnes ayant des comportements différents, et qui sont au même endroit, en même temps et au début d'une catastrophe. Dans les opérations d'évacuation, les personnes ont tendance à s'orienter le plus vite possible vers une zone de sécurité et évitent les sources de danger et les collisions. Les endroits où se déroule une opération d'évacuation sont divers, par exemple, les gares, les stades, les bateaux, les avions, les tunnels, etc.

En effet, l'analyse des systèmes d'évacuation présente d'énormes avantages tels que :

- Aider à la préparation de plans de sauvetage efficaces et efficients.

- Construire des bâtiments sécurisés pour éviter le maximum de victimes en cas de catastrophe.
- Prévoir des situations qui peuvent se produire, mais qui sont inconnues (elles n'ont pas fait partie des connaissances préalables ou ne sont pas produites auparavant).

Pour effectuer une telle analyse, il est important d'utiliser un modèle pertinent pour représenter ce type de systèmes. Ce modèle pourrait être un modèle physique, un modèle analytique ou un modèle de simulation [202]. Cependant, le modèle physique et le modèle analytique pourraient être impossibles à construire en raison du coût, du temps ou d'effort nécessaires pour le faire, donc on peut préférer le modèle de simulation. De plus, il pourrait être très difficile et parfois impossible de se fier uniquement à des modèles analytiques pour analyser et évaluer des systèmes constitués de composantes différentes ayant des états variables dans le temps. Donc, pour résoudre ce problème, la modélisation à base d'agents dénotée par ABM (Agent Based Modeling) pourrait être une des meilleures solutions.

En effet, les sciences sociales font partie d'un large éventail de domaines qui profite de la modélisation à base d'agents, où les chercheurs utilisent largement la simulation informatique pour expliquer, analyser et prédire des phénomènes sociaux [203-212].

Après avoir introduit la simulation sociale à base d'agents, nous abordons dans les sections suivantes l'intégration du concept d'état stable et d'analyse de Monte-Carlo dans les modèles ABSS. Une telle intégration est très importante pour analyser efficacement les résultats d'expériences réalisés en utilisant la simulation par un modèle ABSS.

4.6. Le concept d'état stable dans la simulation sociale à base d'agents

Selon la théorie des systèmes dynamiques, un système est à l'état stable si les variables qui définissent son comportement ne se modifient pas durant le temps.

Lorsque nous parlons de stabilité, on dit qu'un état stable d'un système donné est approché asymptotiquement, si le système est dans un état stable et son comportement récemment observé se poursuivra. Cependant, dans les systèmes stochastiques, les probabilités que divers états soient répétés resteront constantes [213].

Soit x le vecteur des variables d'état d'un système dynamique, et f la fonction qui indique comment ce système se modifie. Dans des circonstances particulières, ce système ne se

modifie pas et il peut être bloqué dans un état spécial. On appelle ces états des points fixes du système dynamique [214].

En temps continu, cela signifie que pour chaque propriété p du système, la dérivée partielle par rapport au temps est nulle, comme le montre l'équation suivante. :

$$\frac{\partial p}{\partial t} = 0 \qquad (4.1)$$

En temps discret, cela signifie que la variation de chaque propriété $\dot{p}$ est nulle, comme le montre l'équation suivante. :

$$\dot{p}_t - \dot{p}_{t-1} = 0 \text{ ; pour tout } t \qquad (4.2)$$

Cependant, les points fixes qu'un système peut posséder peuvent être divisés en trois types, à savoir :

a. Un point fixe $\tilde{x}$ est appelé stable, si pour toutes les valeurs de départ x_0 près de $\tilde{x}$, le système ne reste pas seulement à proximité de $\tilde{x}$ mais aussi $x(t) \rightarrow \tilde{x}$ lorsque $t \rightarrow \infty$.
b. Un point fixe $\tilde{x}$ est appelé marginalement stable ou neutre, si pour toutes les valeurs de départ x_0 près de $\tilde{x}$, le système reste près de $\tilde{x}$ mais ne converge pas vers $\tilde{x}$.
c. Un point fixe $\tilde{x}$ est appelé instable s'il n'est ni stable ni marginalement stable, c'est à dire il existe des valeurs de départ x_0 très proches de $\tilde{x}$ de sorte que le système s'éloigne de $\tilde{x}$.

La figure 4.3 illustre chacune de ces possibilités :

- Le point fixe à gauche de la figure est stable. Toutes les trajectoires qui sont près de $\tilde{x}$ restent proches et convergent vers $\tilde{x}$.
- Le point fixe au centre de la figure est marginalement stable (neutre). Les trajectoires qui commencent près de $\tilde{x}$ restent à proximité, mais ne convergent jamais vers $\tilde{x}$.
- Le point fixe à droite de la figure est instable. Il y a des trajectoires qui commencent près de $\tilde{x}$ et s'éloignent de $\tilde{x}$.

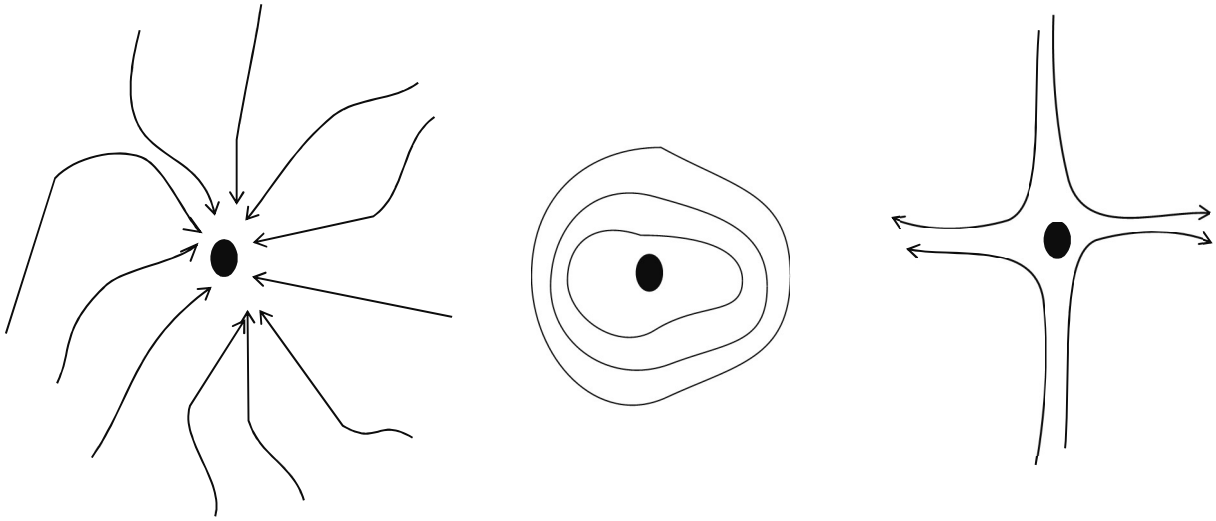


Figure 4.3. Exemples de points fixes avec trois différents types de stabilité. Le point fixe à gauche est stable, le point fixe au centre est marginalement stable et le point fixe à droite est instable [214].

En ce qui concerne les modèles ABSS, ils peuvent être considérés comme des variantes des modèles de simulation à événement discret [215]. Il convient également de mentionner que pour chaque expérience sur le modèle de simulation, il faut identifier les valeurs de paramètres d'entrée. Ensuite, après la réalisation de cette expérience, on obtient des valeurs de paramètres de sorties qui représentent aussi les valeurs de paramètres d'états du modèle dynamique (résultat de la simulation). De plus, ces valeurs des paramètres de sortie peuvent se modifier durant l'écoulement du temps, donc on peut les considérer comme des variables aléatoires qui atteignent des valeurs stables après un certain temps. Ces valeurs stables des paramètres de sortie sont considérées également comme le résultat final de cette expérience.

Soit Y_t un paramètre de sortie du modèle ABSS. Lorsque t est suffisamment grand, Y_t tend souvent vers une valeur stable Y_s (s est le moment où Y_t tient son état stable Y_s), par conséquent :

$$Y_{t+1} - Y_t = 0 \; ; \; \forall \, t \geq s \qquad (4.3)$$

Souvent, la valeur de Y_s est unique et liée à une seule expérience sur le modèle ABSS.

4.7. L'application de la méthode de Monte-Carlo dans l'ABSS

La méthode de Monte-Carlo est une technique heuristique dans laquelle un grand nombre des valeurs générées aléatoirement sont analysées en utilisant un modèle probabiliste pour trouver une solution approximative à un problème numérique qui serait difficile à résoudre en utilisant d'autres méthodes [216, 217].

Afin d'avoir une bonne connaissance du comportement des systèmes sociaux qui sont simulés par des modèles ABSS, il est nécessaire d'obtenir des résultats de simulation stables. C'est pourquoi notre proposition profite de la méthode de Monte-Carlo dans le processus ABSS, afin de tirer des conclusions plus stables et plus générales. Cette approche se base sur la réalisation d'un grand nombre d'expériences en utilisant des échantillons aléatoires des paramètres d'entrée. Cet ensemble d'expériences permet d'avoir des informations concernant les résultats finals de la simulation, par exemple, l'estimation et l'écart type d'un paramètre de sorties. Dans ce qui suit, on va donner plus de détails sur cette approche.

Soit $X = (X_1, X_2, \dots, X_r)$ un ensemble de variables aléatoires qui représentent r paramètres d'entrée d'un modèle ABSS donné, $Y = (Y_1, Y_2, \dots, Y_s)$ l'ensemble des variables aléatoires qui représentent s paramètres de sortie et n est le nombre d'expériences à effectuer.

Les étapes suivantes décrivent généralement ce qu'il faut faire lorsqu'on utilise la méthode de Monte-Carlo :

a. Identifier la distribution de probabilité de chaque paramètre d'entrée.
b. Effectuer n opérations d'échantillonnage des variables aléatoires X en fonction de leurs distributions.
c. Réaliser n expérimentations en utilisant les valeurs de X, afin d'atteindre les valeurs de Y.
d. Analyser les valeurs de Y, afin d'avoir certaines informations comme l'estimation et l'écart type.

À titre d'exemple, considérant $z_1, z_2, \dots, z_n$ les valeurs d'une variable de sortie Z pour n expériences. Dans ce cas, Z peut être considéré comme étant une variable aléatoire, où son estimation μ, son écart-type σ et sa distribution de probabilité sont inconnus.

La moyenne et l'écart-type de Z pour n expériences sont définis respectivement par les équations suivantes :

$$\hat{\mu}_n = \frac{1}{n} \sum_{i=1}^{n} z_i \qquad (4.4)$$

$$\hat{\sigma}_n = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (z_i - \hat{\mu}_n)^2} \qquad (4.5)$$

De plus, si Δ un opérateur de comparaison $(=, =, \geq, \leq, <, >)$ et z une valeur donnée de Z, alors pour estimer la valeur de la moyenne μ, l'écart-type σ et la probabilité $P\ (Z\ \Delta\ z)$, on peut utiliser les formules suivantes [218] :

$$\mu\ \approx\ \hat{\mu}_n \tag{4.6}$$

$$\sigma\ \approx\ \sqrt{\frac{1}{n}\sum_{i=1}^{n}(z_i - \hat{\mu}_n)^2} \tag{4.7}$$

$$P(Z\ \Delta\ z) \approx \frac{1}{n}\sum_{i=1}^{n} I_i\ \ , \tag{4.8}$$

tel que I une variable aléatoire définie comme suit :

$$I_k = \begin{cases} 1\ \ , si\ Z\ \Delta\ z\ dans\ la\ k^{th}\ expérimentation \\ 0\ \ ,\ sinon \end{cases}\ , k = 1,2, \dots, n \tag{4.9}$$

Les valeurs calculées au-dessus ne sont que des estimations, mais par la loi des grands nombres leur convergence vers les valeurs exactes μ et σ est assurée lorsque la taille de l'échantillon n tend vers l'infini, de sorte que :

$$P\left(\lim_{n\to\infty} \hat{\mu}_n = \mu\right) = 1 \tag{4.10}$$

$$P\left(\lim_{n\to\infty} \hat{\sigma}_n = \sigma\right) = 1 \tag{4.11}$$

Dans le même contexte, lorsque $la\ valeur\ de\ n$ est suffisamment grande et sur la base du théorème de la limite centrale [219], on peut estimer une distribution normale approximative $N\ (\hat{\mu}_n, \hat{\sigma}_n/\sqrt{n})$ de la valeur moyenne μ.

De plus, on peut contrôler la différence entre la valeur estimée $\hat{\mu}_n$ et la valeur requise μ au moyen d'un intervalle de confiance [220]. Par exemple, pour la probabilité $\alpha = 0{,}95$, on peut être sûr avec une confiance α (95 %), que la valeur réelle de μ se situe entre $\mu_{inf} = \hat{\mu}_n - 1.96\ \hat{\sigma}_n/\sqrt{n}$, et $\mu_{sup} = \hat{\mu}_n + 1.96\hat{\sigma}_n/\sqrt{n}$, c'est-à-dire $P\ (\mu_{inf} \leq \mu \leq \mu_{sup}) = 0.95$. On note également que la taille de l'intervalle de confiance $|\mu_{sup}, \mu_{inf}|$, qui représente l'incertitude de l'estimation μ, est proportionnelle à $\sqrt{n}$.

4.8. Conclusion

La méthode de Monte-Carlo peut être utilisée pour donner plus de fiabilité à la simulation sociale à base d'agents. L'application de cette méthode sur les résultats d'un grand nombre d'expérimentations réalisées en utilisant des échantillons aléatoires de paramètres d'entrés nous permet d'estimer quelques caractéristiques des paramètres d'état de ce modèle, tels que la valeur moyenne et l'écart type. Ces valeurs stationnaires permettent de déduire des informations plus fiables sur le système étudié. Par conséquent, l'application de cette approche à l'étude d'un système d'évacuation d'un bâtiment fait l'objet du chapitre suivant. Cette approche se base sur le développement d'un modèle à base d'agents représentant le plus fidèlement possible le comportement des individus dans un supermarché en cas de catastrophe d'incendie.

Chapitre 5 : Le cas d'étude

5.1. Introduction

L'évacuation d'urgence est l'évasion urgente et immédiate de personnes loin d'une zone qui représente une menace imminente. Parmi les exemples d'évacuation, on peut citer l'évacuation à petite échelle d'un bâtiment, en raison d'une attaque armée ou d'un incendie, ou bien l'évacuation à grande échelle des villes en raison d'inondations, de tornades ou de bombardements. Dans certaines situations comme la propagation d'une épidémie, des matières dangereuses ou une contamination possible, les personnes évacuées peuvent être décontaminées avant d'être transportées hors de la zone contaminée.

Dans ce chapitre, nous abordons la simulation d'un système d'évacuation de supermarché en cas d'incendie (le présent travail a été publié sous la forme d'un article dans le journal international IJSPM [1]). Le supermarché est un établissement de vente au détail en libre-service, où le bâtiment contient différents produits, plusieurs étages, des portes, des caisses, etc. Sur la base du processus de simulation informatique et des concepts discutés précédemment, les étapes de la simulation d'un système d'évacuation d'un supermarché sont discutées en plus de détail dans les sections suivantes.

5.2. La spécification du système étudié

Les systèmes d'évacuation des supermarchés sont considérés comme des systèmes dynamiques et complexes. Dans ce type de systèmes, nous devons faire face au problème d'évacuation d'un grand nombre de personnes de différentes caractéristiques, dans des immeubles ayant des caractéristiques différentes comme les positions des rangées d'étagères et les positions des portes de sortie. De plus, pour faire face à des catastrophes telles que la propagation du feu, l'étude de ce type de système à l'aide d'un modèle dynamique porte une grande importance pour éviter le maximum de pertes, où ce modèle doit prendre en compte plusieurs facteurs tels que le temps, les caractéristiques du bâtiment et les caractéristiques des personnes.

Le modèle dynamique proposé dans cette étude est un modèle de simulation sociale à base d'agents. Ce type de modèles permet de modéliser et visualiser le comportement dynamique du système global via les comportements de ses entités internes qui interagissent entre eux.

De plus, nous utilisons deux concepts qui traitent la notion de stabilité, à savoir le concept d'état stable afin d'obtenir des résultats plus stables des expériences effectuées, où ces résultats avec les données d'entrées représentent l'ensemble de données initiales pour effectuer une étude de fouille de données dynamique, ainsi que la méthode de Monté Carlo pour calculer les caractéristiques globales du modèle via une approche probabiliste.

La modélisation de système d'évacuation d'un supermarché en raison de l'incendie implique la modélisation de ses composants internes, la modélisation de la propagation du feu, ainsi que la modélisation des personnes qui se déplacent dans cet immeuble et de leurs comportements avant et après l'apparition du feu.

Concernant la structure interne de la plupart des supermarchés, on trouve, par exemple, les composants suivants : les rangées d'étagères contenant plusieurs produits de différents types, les extincteurs du feu, les portes de sortie et les caisses.

Les personnes à l'intérieur du supermarché peuvent être classées selon plusieurs critères, et chaque classe de personnes possède des caractéristiques différentes telles que les caractéristiques physiques (par exemple : homme / femme, enfant / adulte, sain / malade), les caractéristiques psychologiques (par exemple : agressif/calme) et les caractéristiques culturelles et intellectuelles (par exemple : éduqué / inéduqué, conscient/inconscient).

De plus, chaque personne présente des comportements dans le supermarché qui peuvent différer selon la propagation du feu. Avant l'apparition du feu, on peut citer :

- Rechercher des marchandises ou aller à la porte de sortie.
- Éviter les collisions avec des obstacles et avec d'autres personnes.
- Modifier la vitesse de déplacement.
- Accompagner une autre personne (par exemple, des adultes accompagnent des enfants).

Après l'apparition de l'incendie, on peut citer :

- S'éloigner du feu,
- Aller à la porte la moins encombrée.
- Aller à la porte la plus proche.

L'état initial de la simulation est l'apparence de la catastrophe (l'apparence du feu) et l'état final est quand le système arrive à son état stable, où il n'y a pas de modifications significatives

au niveau des paramètres de sortie du modèle. Dans ce cas, les gens seront à l'extérieur du bâtiment ou bien ils seront morts, donc aucune variable décrivant le système, y compris les paramètres de sortie, n'aura de nouvelles valeurs (ils arrivent à leurs états stables).

L'état stable considéré dans une telle simulation est celui lié aux paramètres de sortie du modèle, à savoir le nombre de décès, le nombre de survivants et le nombre de blessés qui représentent également les résultats d'expériences de la simulation. Graphiquement, comme le montre la figure 5.1, cet état stable correspond aux coordonnées y des points de fin de la courbe en noir représentant le nombre de personnes décédées, la courbe en rouge qui représente le nombre de personnes blessées et la courbe en vert qui représente le nombre de personnes survivantes. Le temps de fin de la simulation est la coordonnée x commune de ces points de terminaison.

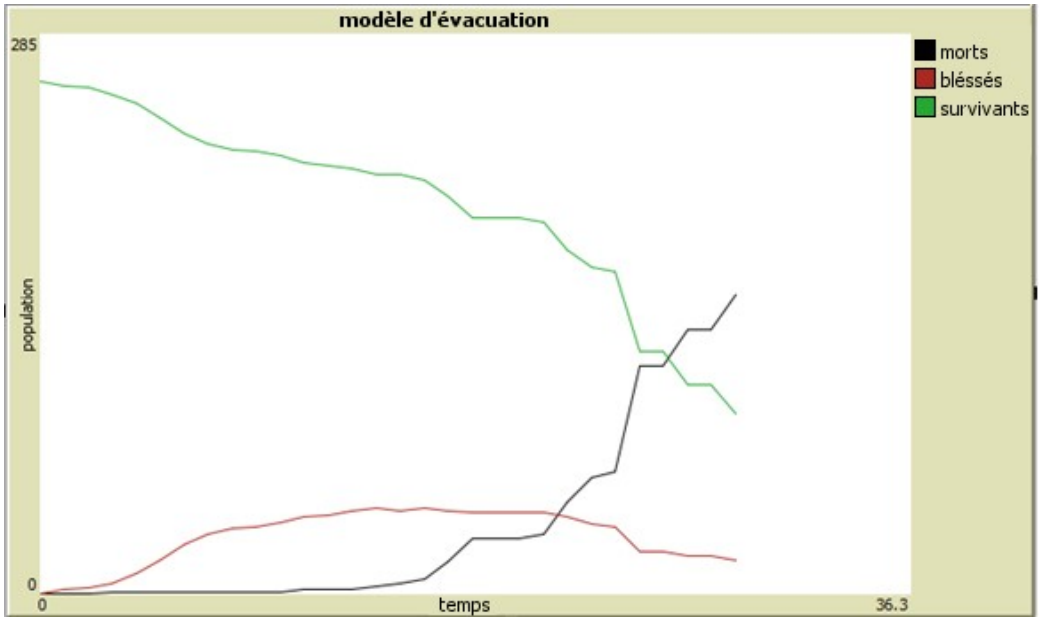


Figure 5.1. Courbes des résultats de simulation

Après avoir discuté la spécification du système étudié qui permet d'identifier les contraintes et les hypothèses de la simulation, dans la section suivante, nous identifions le modèle conceptuel de la simulation à travers un ensemble de diagrammes UML qui permettent de décrire la structure et le comportement du simulateur de ce système.

5.3. La conception du modèle opérationnel

En raison de sa simplicité et de sa richesse en termes de diagrammes, le langage UML (Unified Modelling Language) [221] est considéré comme un bon choix pour concevoir le modèle de simulation.

Dans ce contexte, nous utilisons le diagramme de classes pour décrire la structure du modèle opérationnel, par lequel nous représentons les éléments essentiels du système, tels que les personnes qui se déplacent dans le supermarché et la structure interne du supermarché.

De plus, le diagramme d'états-transitions est utilisé pour décrire les comportements dynamiques des personnes dans le supermarché. Les comportements des personnes sont divisés en deux types, l'un lié aux individus indépendants et l'autre lié aux enfants qui accompagnent des adultes.

Le diagramme de classe illustré à la figure 5.2 décrit le modèle opérationnel selon l'environnement NetLogo [222]. Dans ce diagramme, l'élément principal est l'agent, qui peut être un « turtle » (agent dynamique), un « patch » (une position) ou un « link » (lien entre deux turtles). Chaque type d'agent a ses propres attributs et méthodes.

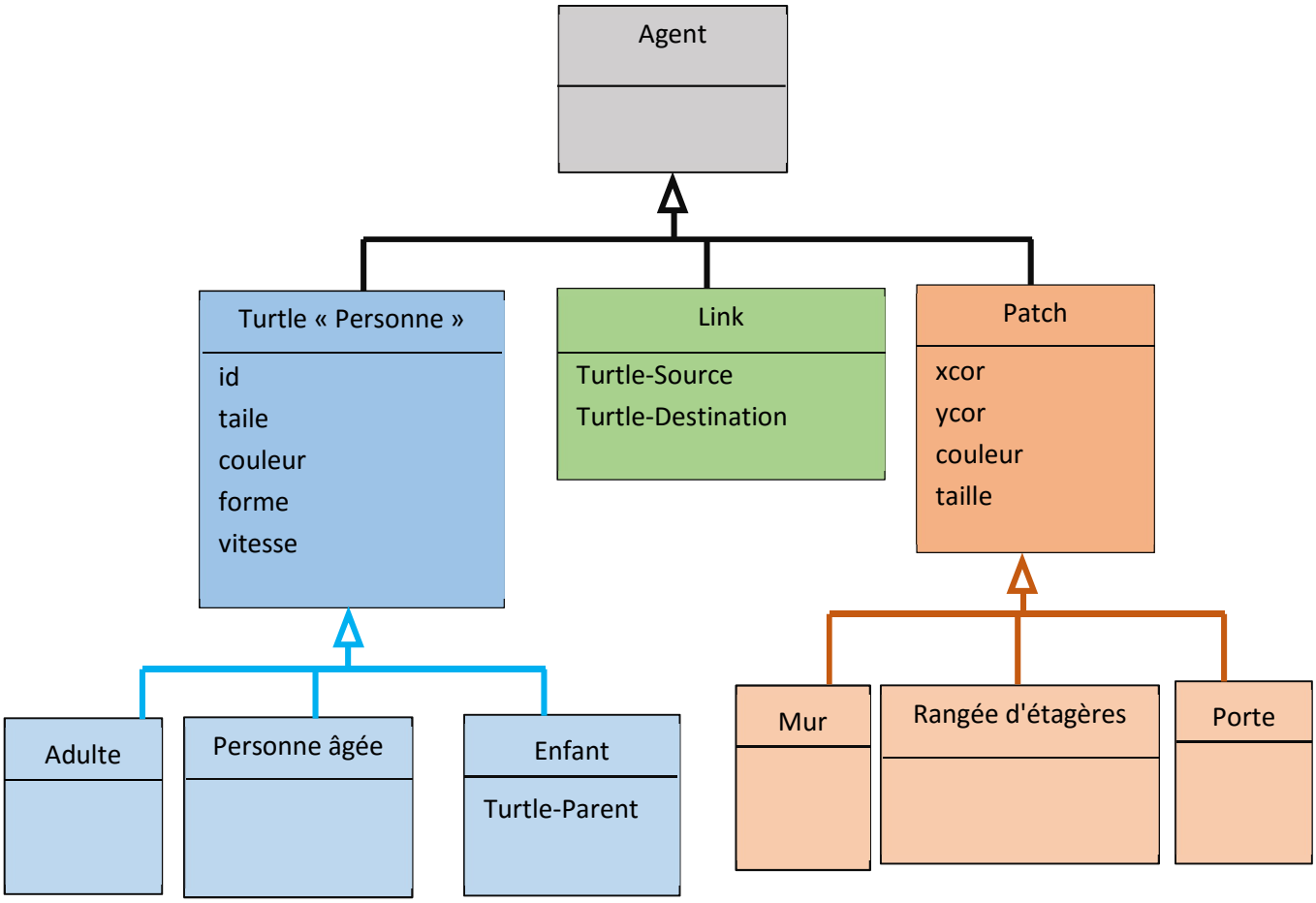


Figure 5.2. Diagramme de classes représentant la structure du modèle de simulation.

Les diagrammes d'états-transitions suivants spécifient respectivement les comportements des individus indépendants et les individus enfants qui accompagnent des adultes. Dans ces diagrammes, les rectangles aux coins arrondis représentent des états, et chaque flèche représente une transition d'un état à un autre. Chaque flèche de transition est associée à une contrainte, à l'exception de la flèche liée à l'état initial qui est représenté par un point de couleur noire et à l'état final qui est représenté par un cercle avec un point de couleur noire.

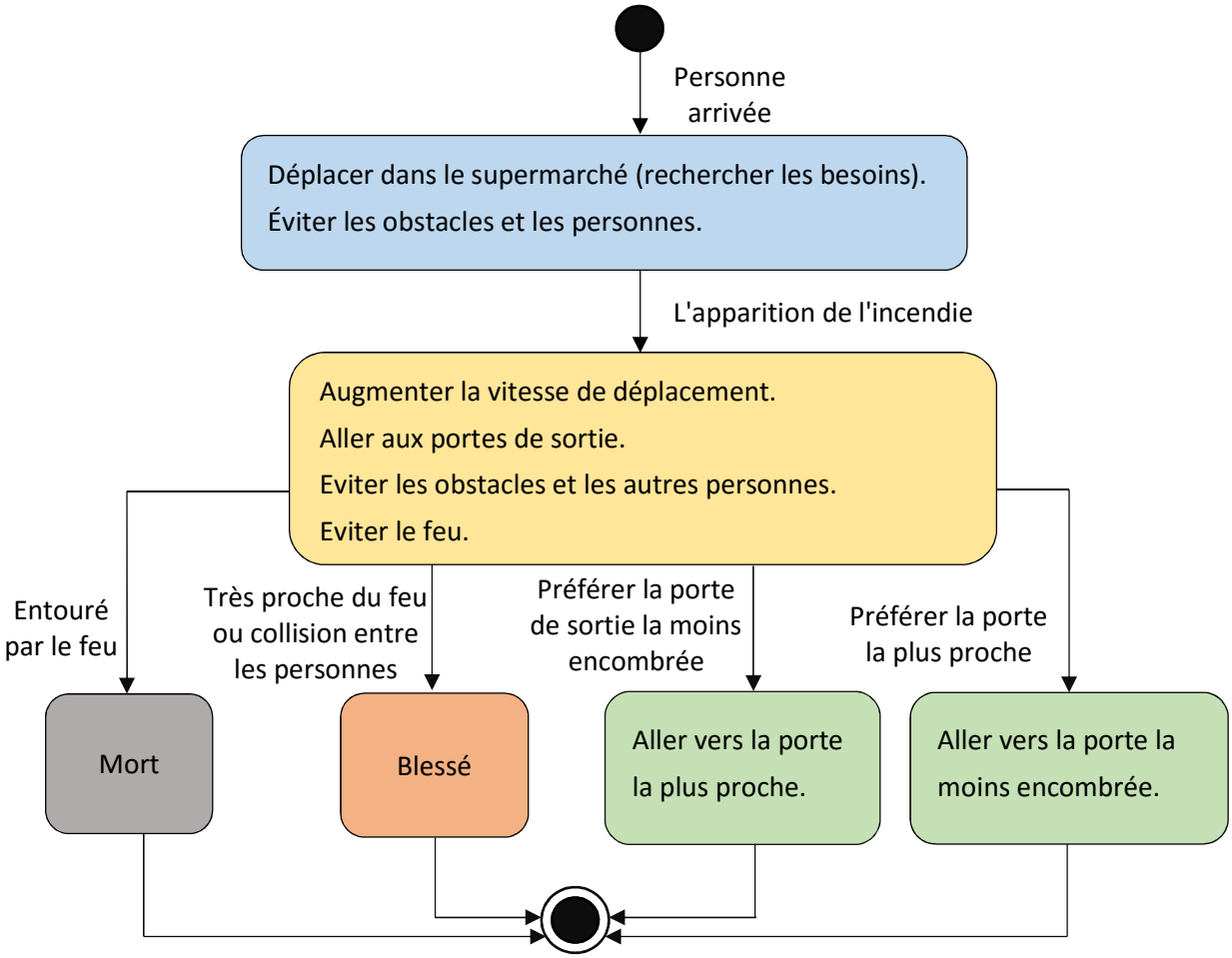


Figure 5.3. Diagramme d'état transition représentant les comportements des individus indépendants.

Le diagramme de la figure 5.3 représente les comportements d'un adulte, d'une personne âgée ou d'un enfant, qui agissent de façon indépendante. Ces comportements peuvent être changés en fonction de l'existence ou pas du feu. Avant l'observation du feu, ils peuvent continuer leurs achats et éviter les obstacles. Après avoir observé le feu, ils se précipitent vers la porte la moins encombrée ou se dirigent vers la porte la plus proche. L'état de chaque personne se modifie en fonction de l'évaluation de l'environnement proche.

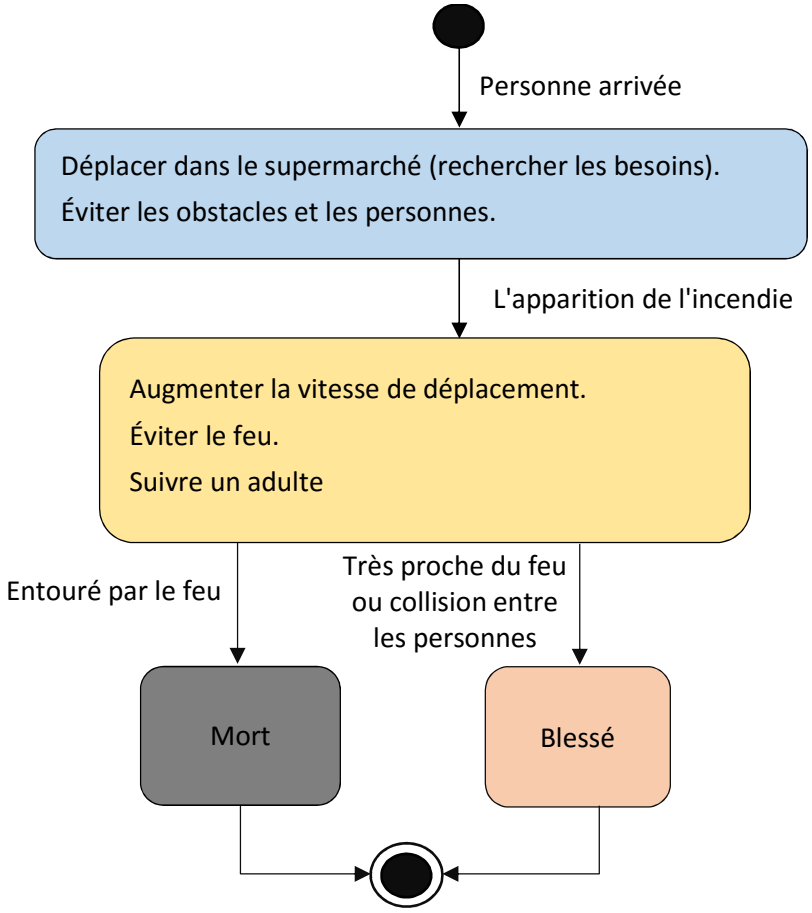


Figure 5.4. Diagramme d'état transition représentant les comportements des enfants accompagnés d'adultes.

Dans la figure 5.4, le comportement d'une personne dépendante (les enfants qui accompagnent des adultes) dépend de celui de la personne indépendante accompagnée. Dans ce cas, l'enfant suit l'adulte dans ses décisions et il est plus vulnérable aux blessures.

Après avoir expliqué le modèle conceptuel qui décrit la structure et le comportement de l'ensemble des agents considérés dans le modèle opérationnel, nous expliquons dans la section suivante la mise en œuvre de ce modèle opérationnel sur l'environnement NetLogo.

5.4. L'implémentation du modèle opérationnel

Pour implémenter le modèle opérationnel nous utilisons des outils divers, à savoir, l'environnement NetLogo, l'environnement R et le package RNetLogo. Nous présentons brièvement ces environnements de développement dans les paragraphes suivants.

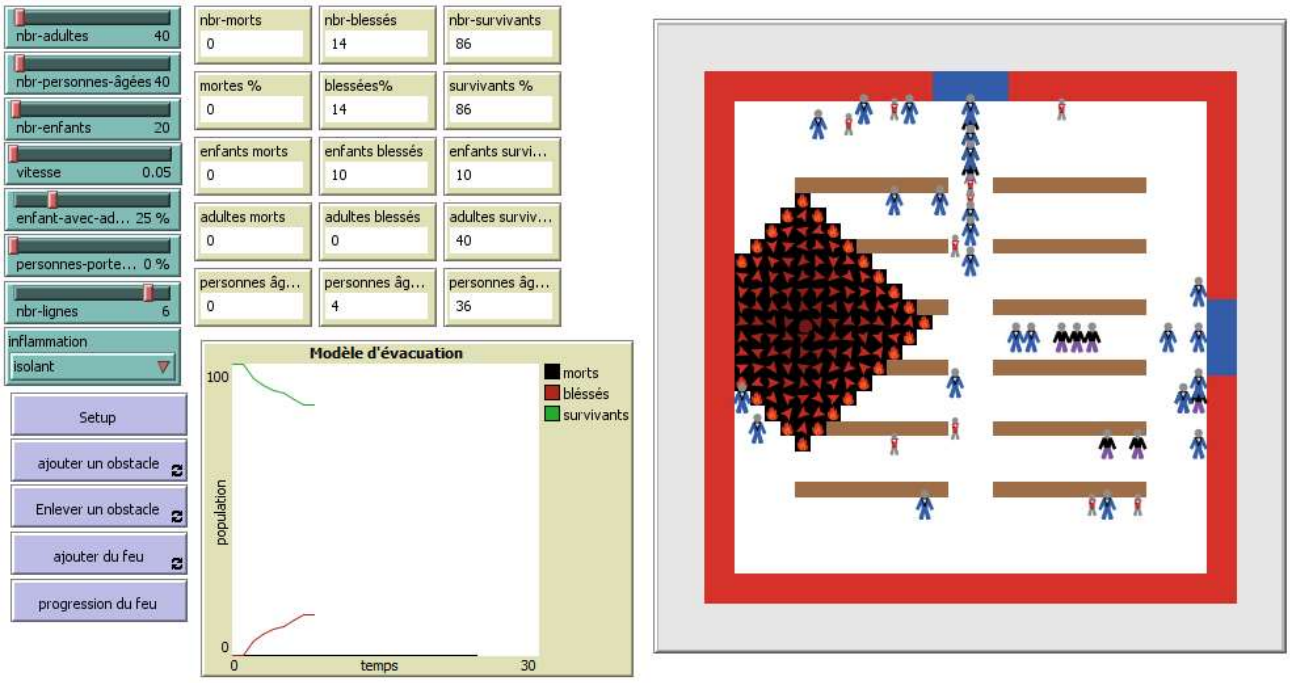


Figure 5.5. Interface d'utilisateur graphique permettant de créer et manipuler le modèle NetLogo.

La plateforme NetLogo [222, 223] est un environnement de modélisation programmable pour la simulation de phénomènes naturels et sociaux.

NetLogo connait quatre types d'agents : les turtles, les patches, les links et l'observateur. Les turtles sont les agents qui peuvent se déplacer dans le monde. Le monde (world) est un espace à deux dimensions qui est divisé en une grille de patches (plaques, carreaux, cases). Chaque patch est un morceau carré de « sol » sur lequel les turtles peuvent se déplacer. Un link est un agent qui relie deux turtles. L'observateur n'a pas de position déterminée. Il observe d'en haut le monde des agents.

Les turtles, les patches et les links peuvent suivre des instructions provenant de l'utilisateur ou d'autres agents dans le modèle. Il est également possible de définir le comportement des agents en utilisant le panneau de code dans NetLogo. La figure 5.5 montre l'interface utilisateur graphique (GUI) de l'environnement NetLogo, qui comprend les composants suivants :

- Les éléments de contrôle : ils comprennent des boutons, des curseurs, des sélecteurs et des interrupteurs.
- Les éléments d'affichage : ils comprennent des moniteurs, des courbes et le monde (espace contenant les agents).

Dans le monde (figure 5.5), les personnes sont représentées par des turtles de formes différentes (des adultes, des personnes âgées et des enfants). Les patches colorés en marron

représentent les rangées d'étagères. Les patches colorés en rouge représentent les murs et les patches colorés en bleu représentent les portes de sortie.

Les courbes dans la figure 5.1 montrent les fréquences des individus dans le modèle, de sorte que le noir indique le nombre de morts, le rouge représente le nombre de blessés et le vert représente le nombre de survivants.

Le langage R [224]est un environnement de calcul statistique et graphique qui fournit un langage de programmation, de nombreux graphiques, des fonctions de débogage et des interfaces avec d'autres langages.

Le package RNetLogo [225] fournit une interface permettant d'utiliser NetLogo à partir de R, soit avec ou sans une interface graphique interactive (GUI). De plus, ce package fournit des fonctions pour charger des modèles NetLogo, exécuter des commandes et obtenir des rapports d'exécution.

Après avoir présenté les différents outils et les plateformes utilisées dans ce travail, dans la section suivante, on va expliquer les expériences réalisées pour étudier le système d'évacuation, où nous simulons ce système, en considérant différents scénarios initiaux, puis nous analysons les résultats de la simulation afin d'obtenir de nouvelles informations sur ce système.

5.5. L'expérimentation et l'analyse de résultats

Dans cette section, nous discuterons la réalisation de plusieurs expériences en utilisant le modèle opérationnel décrit précédemment. Ces expériences sont faites en modifiant les valeurs des paramètres d'entrée afin d'obtenir des résultats pour des scénarios différents, puis analyser ces résultats pour tirer des connaissances et des conclusions sur le système étudié. Parmi les paramètres d'entrée que l'on peut utiliser :

- l'emplacement de départ de l'incendie,
- le nombre initial de personnes,
- le nombre de rangées d'étagères,
- le type de rangées d'étagères,
- le nombre et l'emplacement des portes de sortie,
- les emplacements des extincteurs d'incendie,
- l'existence et l'emplacement de matières inflammables.

Dans cette simulation, il est nécessaire d'établir des hypothèses relatives au système étudié ou à son modèle, dont certaines sont mentionnées dans les paragraphes suivants.

D'abord, étant donné la multitude de paramètres des systèmes d'évacuation qui pourraient être étudiés dans ce type de simulation et leur similitude en termes de méthode de traitement, nous n'en aborderons que quelques-uns, à savoir :

- le point de départ du feu,
- le nombre de rangées d'étagères,
- la position des produits très inflammables,
- le nombre initial de personnes selon des catégories d'âge différentes (enfants, adultes et personnes âgées),
- le pourcentage d'enfants qui sont accompagnés par des adultes par rapport au nombre total d'enfants.

Ces paramètres d'entrée sont considérés comme des variables indépendantes. Tandis que, les paramètres de sortie dans ce modèle de simulation sont les variables dépendantes, et ils sont représentées par le nombre de morts, le nombre de survivants et le nombre de blessées.

En ce qui concerne les expériences faites sur le modèle de simulation, on peut distinguer deux types d'expériences. D'une part, les expériences directes utilisant des paramètres d'entrée dont les valeurs sont déterminées manuellement (ajoutées par l'interface d'utilisateur graphique dans l'environnement NetLogo qui est dédié au modèle à base d'agents). D'autre part, les expériences faites en utilisant des paramètres d'entrée, dont les valeurs, sont définies aléatoirement et automatiquement. Dans le dernier cas, on considère les paramètres d'entrées comme étant des variables aléatoires ayant des distributions de probabilité différentes. De plus, nous appliquons la méthode de Monte-Carlo pour définir la distribution du nombre de victimes, où cette méthode nécessite un nombre d'expériences très élevé, et pour cela, nous générons automatiquement 1000 expériences en utilisant le langage R, l'environnement NetLogo et le package RNetLogo.

5.5.1. Expériences utilisant des valeurs d'entrées déterminées manuellement

Dans cette section, on va présenter quelques expériences réalisées en utilisant des valeurs d'entrées déterminées manuellement (par l'utilisateur du simulateur). Les expériences effectuées donnent à plusieurs reprises des résultats attendus, mais elles représentent également un moyen de prouver que le modèle construit reflète bien le système étudié. Cependant, comme il sera montré dans la section 5.5.2, pour obtenir des informations plus précises sur le système étudié, nous devons effectuer un grand nombre d'expériences pour couvrir le plus grand nombre de cas pris au hasard, et dans ce cas nous devrons utiliser des méthodes automatiques soit pour la construction de l'ensemble de données d'entrées ou bien pour l'analyse de résultats tirées des expériences réalisées (l'utilisation de la méthode Monte-Carlo).

a. Le point de départ du feu

Dans ces expériences, nous considérons les valeurs fixes des paramètres présentées dans le tableau 5.1. Ensuite, nous changeons l'emplacement de la première apparition du feu à chaque fois (tableau 5.2). Dans la première expérience, nous considérons que le point de départ du feu est situé aux bords du supermarché, mais loin des portes de sortie. Dans la deuxième expérience, nous considérons que le point de départ du feu est situé au milieu du supermarché. Dans la troisième expérience, nous considérons que le feu est apparu près des portes de sortie.

Tableau 5.1. Valeurs des paramètres inchangés, concernant les points d'apparition du feu.

Paramètre	Valeur
Nombre d'adultes (mais pas vieux)	40
Nombre de personnes âgées	40
Nombre d'enfants	20
Pourcentage d'enfants d'adultes	25 %
Nombre de rangées d'étagères	8

Tableau 5.2. Résultats des expériences concernant les points d'apparition du feu.

Expérience	1	2	3
Lieu de départ du feu	Aux bords, loin des portes de sortie	Au milieu de l'immeuble	Près des portes de sortie
Nombre d'adultes morts	3	8	29
Nombre d'enfants morts	8	6	18
Nombre de personnes âgées mortes	10	9	36
Nombre d'adultes blessés	10	6	6
Nombre d'enfants blessés	3	7	2
Nombre de personnes âgées blessées	1	9	3
Nombre total de morts	21	23	83
Nombre total de blessés	14	22	11

Après ces expériences et selon les résultats montrés dans le tableau 5.2, nous constatons que :

- Le nombre de morts et de blessés augmente lorsque la distance entre le point du départ de l'incendie et les portes de sortie diminue.
- L'augmentation du nombre de portes de sortie permet de réduire le nombre de morts et de blessés.

b. Nombre de rangées d'étagères

Dans ce cas, il s'agit d'étudier les conséquences de la modification du nombre de rangées d'étagères. Les valeurs des autres paramètres sont donc fixées (tableau 5.3), tandis que le nombre de rangées est modifié dans chaque expérience et il prend successivement les valeurs introduites dans le tableau 5.4.

Tableau 5.3. Valeurs des paramètres inchangés, concernant le nombre de rangées d'étagères.

Paramètre	Valeur
Nombre d'adultes	40
Nombre de personnes âgées	40
Nombre d'enfants	20
Pourcentage d'enfants accompagnés d'adultes	25 %
Lieu de départ de l'incendie	Milieu

Tableau 5.4. Résultats des expériences concernant le nombre de rangées d'étagères.

Expérience	1	2	3	4	5	6	7
Nombre de rangées d'étagères	2	4	6	8	10	12	14
Nombre d'adultes morts	3	2	3	3	3	3	3
Nombre d'enfants morts	3	5	5	7	7	7	8
Nombre de personnes âgées mortes	10	10	11	13	15	15	15
Nombre d'adultes blessés	3	5	5	5	4	4	4
Nombre d'enfants blessés	9	6	6	5	5	6	6
Nombre de personnes âgées blessées	0	1	1	0	1	1	1
Nombre total de morts	16	17	19	23	25	25	26
Nombre total de blessés	12	12	12	10	10	11	11

D'après les résultats montrés ci-dessus (tableau 5.4), nous constatons que le nombre total de victimes (nombre de morts et de blessés) augmente lorsque le nombre de rangées d'étagères augmente.

c. Le nombre de personnes selon les différentes catégories d'âge

Pour ces expériences, nous étudions l'impact du nombre de personnes de différentes catégories d'âge sur la valeur du nombre de victimes. Les valeurs des paramètres restants sont déterminées dans le tableau 5.5. Ensuite, plusieurs expériences sont menées avec des nombres différents de personnes de différents âges (tableau 5.6).

Tableau 5.5. Valeurs des paramètres inchangés, concernant les nombres de personnes de différents âges.

Paramètre	Valeur
Lieu de départ de l'incendie	Milieu
Nombre de rangées d'étagères	8

Tableau 5.6. Résultats des expériences concernant les nombres de personnes de différents âges

Expérience	1	2	3	4	5	6	7
Nombre d'adultes	15	25	10	10	15	30	40
Nombre de personnes âgées	15	10	25	10	10	30	40
Nombre d'enfants	10	5	5	20	15	20	25
Pourcentage d'enfants accompagnés d'adultes	25 %	25 %	25 %	25 %	80 %	25 %	25 %
Nombre d'adultes morts	0	2	1	1	0	2	4
Nombre d'enfants morts	3	2	2	9	10	5	10
Nombre de personnes âgées mortes	3	4	8	4	5	12	16
Nombre d'adultes blessés	1	2	1	0	2	4	5
Nombre d'enfants blessés	1	0	0	1	1	6	5
Nombre de personnes âgées blessées	0	0	0	1	0	0	0

Nombre total de morts	6	8	11	14	15	19	30
Nombre total de blessés	2	2	1	2	3	10	10

En ce qui concerne les expériences qui sont réalisées (tableau 5.6), nous concluons que :

- Le nombre de morts et de blessés augmente lorsque le nombre de personnes augmente (expériences 6 et 7).
- Le nombre de morts et de blessés augmente lorsque le pourcentage d'enfants accompagnés par des adultes augmente (expérience 5).
- Le nombre de décès chez les personnes âgées et les enfants est supérieur au nombre de décès chez les adultes non âgés.

d. Types et organisation des rangées d'étagères

Dans ces expériences, on va étudier l'impact des types de rangées d'étagères et leur organisation sur les individus et la propagation du feu. Les types des rangées d'étagères sont divisés comme suit :

- Les étagères ininflammables : par exemple l'eau minérale, les boissons gazeuses et les jus de fruits.
- Les étagères moyennement inflammables : comme la nourriture, les vêtements et le bois.
- Les étagères très inflammables : comme les produits pétroliers et chimiques, les matières plastiques et les pesticides.

Le tableau 5.7 représente les paramètres dont les valeurs sont inchangées. Tandis que le tableau 5.8 représente les résultats des expériences selon les différentes positions des rangées d'étagères.

Tableau 5.7. Valeurs des paramètres inchangés concernant les positions des rangées d'étagères

Paramètre	Valeur
Nombre d'adultes non âgées	40
Nombre de personnes âgées	40

Nombre d'enfants	20
Pourcentage d'enfants accompagnés par des adultes	25 %
Nombre de rangées d'étagères	12
Nombre d'étagères ininflammables	4
Nombre d'étagères moyennement inflammables	4
Nombre d'étagères très inflammables	4

Tableau 5.8. Résultats des expériences concernant les positions des rangées d'étagères.

Expérience	1	2	3	4
Positions de rangées d'étagères	Rangées d'étagères composées d'une manière mixte	Étagères très inflammables près de portes de sortie	Étagères très inflammables au milieu du supermarché	Des étagères très inflammables loin des portes de sortie
Nombre d'adultes morts	3	8	5	1
Nombre d'enfants morts	5	8	9	6
Nombre de personnes âgées mortes	8	18	11	10
Nombre d'adultes blessés	4	3	6	2
Nombre d'enfants blessés	6	4	4	5
Nombre de personnes âgées blessées	1	0	2	4
Nombre total de morts	16	34	25	17
Nombre total de blessés	11	7	12	11

D'après les résultats des expériences présentées ci-dessus, nous constatons que :

- Des rangées d'étagères différentes avec des positions mélangées diminuent le nombre total de morts et de blessés.
- Garder les étagères des produits très inflammables loin des portes de sortie et du milieu du supermarché diminue aussi le nombre de morts et de blessés.

5.5.2. Expériences avec l'application de la méthode Monte-Carlo

Dans ces expériences, nous identifions d'abord la distribution de probabilité de chaque paramètre d'entrée, en fonction de nombreux facteurs, par exemple, la structure du supermarché et la densité de personnes par rapport à la superficie du supermarché. Ces distributions de probabilité nous permettent de générer automatiquement et aléatoirement les valeurs des paramètres d'entrées. Ensuite, nous effectuons des expériences qui correspondent à ces valeurs d'entrée. Finalement, nous estimons les valeurs des résultats (paramètres de sortie) tirés des expériences effectuées à travers la méthode de Monte-Carlo.

5.5.2.1. Les distributions de probabilité des paramètres d'entrée

Pour définir le nombre de personnes dans le supermarché avant l'apparition de l'incendie, nous prenons en considération la densité de personnes par rapport à la superficie du supermarché sur laquelle les gens peuvent se déplacer. Par exemple, pour le modèle proposé nous avons 30 x 30 unités d'espace moins l'espace de rangées d'étagères. Dans le cas moyen, nous avons une personne dans chaque 3 unités d'espace. Donc la densité de visiteurs est d'environ 300 personnes dans le supermarché à un moment donné avant l'apparition du feu. Selon les considérations précédentes, une distribution de Poisson avec un paramètre égal à 300 pourrait être un choix raisonnable pour le processus d'arrivée de visiteurs. De plus, cette variable aléatoire peut comprendre trois variables aléatoires (nombre d'adultes, nombre de personnes âgées et nombre d'enfants). Par conséquent, ces variables ont également une distribution de Poisson avec 100 comme paramètre.

Le point de départ du feu est représenté par deux variables aléatoires, à savoir, la coordonnée x-incendie et la coordonnée y-incendie, ayant chacune une distribution uniforme continue sur l'intervalle [-16, 16].

Le nombre de rangées d'étagères est représenté par une variable aléatoire ayant une distribution uniforme discrète sur l'ensemble {2, 4…, 14}.

Dans le même contexte, la variable aléatoire enfant-adulte indique le pourcentage des enfants qui accompagnent des adultes. Cette variable aléatoire a une distribution uniforme continue sur l'intervalle [0, 1].

À la lumière de ce qui précède, nous décrivons dans le tableau suivant, les distributions de probabilité des variables aléatoires d'entrée et leurs paramètres correspondants.

Tableau 5.9. Variables d'entrée du modèle de simulation

Paramètre d'entrée	**Distribution de probabilité de variable aléatoire**	**Fonction de probabilité**	**Paramètres de la fonction de probabilité**
Nombre d'adultes	Poisson	$P(X)= \frac{e^{-\mu}\,\mu^{x}}{x!}$	$\mu = 100$
Nombre de personnes âgées	Poisson	$P(X)= \frac{e^{-\mu}\,\mu^{x}}{x!}$	$\mu = 100$
Nombre d'enfants	Poisson	$P(X)= \frac{e^{-\mu}\,\mu^{x}}{x!}$	$\mu = 100$
Nombre de rangées d'étagères	Uniforme discret	$P(X = x_k) = \frac{1}{n}, k = 1, \dots, n$	$x_1 = 2, x_2 = 4,$ $x_3 = 6, x_4 = 8,$ $x_5 = 10,$ $x_6 = 12, x_7 = 14$
Pourcentage d'enfants accompagnés d'adultes	Uniforme continu	$f(x) = \begin{cases} \frac{1}{b-a}, & for\ a \le x \le b, \\ 0 & for\ x < a\ or\ x > b \end{cases}$	$a = 0,$ $b = 1$
Coordonnée x de premier lieu du feu	Uniforme continu	$f(x) = \begin{cases} \frac{1}{b-a}, & for\ a \le x \le b, \\ 0 & for\ x < a\ or\ x > b \end{cases}$	$a = -16,$ $b = 16$
Coordonnée y de premier lieu du feu	Uniforme continu	$f(x) = \begin{cases} \frac{1}{b-a}, & for\ a \le x \le b, \\ 0 & for\ x < a\ or\ x > b \end{cases}$	$a = -16,$ $b = 16$

5.5.2.2. L'échantillonnage des paramètres d'entrée et l'expérimentation

Le code source dans le tableau 5.10 inclut les instructions nécessaires pour :

- Générer les valeurs de chaque paramètre d'entrée.
- Réaliser les expériences sur le modèle de simulation (implémenté sur l'environnement NetLogo).

- Transmettre les résultats de ces expériences depuis le modèle opérationnel de simulation vers l'environnement R, pour analyser ces résultats.

Les opérations précédentes sont répétées automatiquement 1000 fois, ce qui représente le nombre d'expériences effectuées. Pour chaque expérience, les valeurs de paramètres d'entrée et de sortie sont ajoutées à une table de données définie dans l'environnement R (le tableau 5.11 représente un aperçu de cette table de données).

Tableau 5.10. Code source représentant l'implémentation des expériences de la simulation.

N°	Instruction
1.	i<-1
2.	**while**(i<=1000){ *# répétez l'expérimentation 1000 fois*
3.	**NLCommand**("set nbr-adultes", M[i,"adults"]) *# définir le nombre d'adultes*
4.	**NLCommand**("set nbr-old-people", M[i,"old.people"]) *# définir le nombre de personnes âgées*
5.	**NLCommand**("set nbr-children", M[i,"children"]) *# définir le nombre d'enfants*
6.	**NLCommand**("set child-with-adulte", M[i,"child.with.adult"]) *# définir le pourcentage d'enfants*
7.	**NLCommand**("set rows-number", M[i,"rows"]) *# définir le nombre de rangées d'étagères*
8.	**NLCommand**("setup") *# initialise the operational model*
9.	**NLCommand**("ask patch", M[i,"x.fire"], M[i,"y.fire"],
10.	*# définir l'emplacement de départ de l'incendie,*
11.	"[sprout-fires 1[set color red set pcolor black]]")
12.	**NLDoCommand**(100, " fire- propagation ") *# effectuer la simulation 100 fois pour obtenir des*
13.	*# résultats de sortie plus stable*
14	M[i,"deaths"] <- **NLReport**("nbr-deaths") *# obtenir le nombre total de morts*
15.	M[i,"adult.deaths"] <- **NLReport**("nbr-adult-deaths") *# obtenir le nombre d'adultes morts*
16.	M[i,"children.deaths"] <- **NLReport**("nbr-children-deaths") *# obtenir le nombre d'enfants morts*
17.	M[i,"old-people.deaths"] <- **NLReport**("nbr-old-deaths") *# nombre de personnes âgées mortes*
18.	M[i,"injuries"] <- **NLReport**("nbr-injuries") *# obtenir le nombre total de blessés*
19.	M[i,"adult.injuries"] <- **NLReport**("nbr-adult-injuries") *# obtenir le nombre d'adultes blessés*
20.	M[i,"children.injuries"] <- **NLReport**("nbr-children-injuries") *# obtenir le nombre d'enfants blessés*
21.	M[i,"old.people.injuries"] <- **NLReport**("nbr-old-injuries") *# nombre de personnes âgées blessées*
22	i<-i+1
23	}

Dans le code source présenté dans le tableau 5.10, les expressions colorées en bleu représentent les fonctions de R fournies par le package RNetLogo, et les expressions qui commencent par « # » représentent des commentaires.

Pour calculer les valeurs estimées des nombres de morts et de blessés, nous avons effectué 1000 expériences afin d'obtenir un grand nombre des données des paramètres d'entrée et de sortie, ce qui permet d'identifier le comportement du système étudié de manière plus stable et plus précise.

Le tableau 5.11 montre les valeurs des paramètres d'entrée et de sortie correspondent aux expériences réalisées. Dans ce tableau, chaque colonne représente une expérience effectuée et les points représentent la séquence d'expériences de 4 à 999.

Tableau 5.11. Aperçu des valeurs des paramètres d'entrée générées aléatoirement.

Expérience	1	2	3	. . .	998	998	1000
Nombre d'adultes	110	114	98	. . .	86	107	87
Nombre de personnes âgées	103	83	104	. . .	108	94	108
Nombre d'enfants	100	123	94	. . .	114	128	109
Pourcentage d'enfants accompagnés d'adultes	0.05	0.35	0.05	. . .	0.54	0.53	0.23
Nombre de rangées d'étagères	12	6	4	. . .	10	2	12
Coordonnée x de position d'apparition du feu (x-incendie)	1.02	-1.84	11.08	. . .	-2.57	-13.02	-11.39
Coordonnée y de position d'apparition du feu (y-incendie)	11.71	15	-1.13	. . .	-5.7	13.19	-8.74
Nombre d'adultes morts	13	11	22	. . .	5	0	0
Nombre d'enfants morts	36	21	36	. . .	24	1	1
Nombre de personnes âgées mortes	27	16	41	. . .	18	1	1
Nombre d'adultes blessés	16	21	17	. . .	7	9	2
Nombre d'enfants blessés	57	88	47	. . .	69	104	86
Nombre de personnes âgées blessées	10	26	21	. . .	0	18	1

5.5.2.3. L'analyse des valeurs des paramètres de sortie

Après avoir effectué ces expériences en utilisant les environnements NetLogo et R, nous analysons leurs résultats par l'application de la méthode Monte-Carlo pour, par exemple, estimer les valeurs du nombre de mortes et du nombre de blessés, ainsi que la distribution de probabilité de chacun de ces paramètres de sortie.

Le tableau 5.12 montre quelques paramètres statistiques des variables de sortie. Ces paramètres sont calculés à base des expériences réalisées (tableau 5.11) et de l'application de la méthode de Monte-Carlo montrée dans la section 4.7.

Tableau 5.12. Paramètres statistiques des variables de sortie obtenus à l'aide de la méthode Monte-Carlo.

Paramètre de sortie	Estimation	Écart-type
Nombre total de morts	46.42	37.60
Nombre total de blessés	78.03	19.91
Nombre total de survivants	174.94	32.11
Nombre d'adultes morts	6.225	8.34
Nombre d'enfants morts	20.24	16.14
Nombre de personnes âgées mortes	19.97	15.94
Nombre d'adultes blessés	8.38	7.31
Nombre d'enfants blessés	63.58	15.08
Nombre de personnes âgées blessées	6.06	7.47

Dans ce qui suit, les figures 5.6, 5.7, 5.8 et 5.9 montrent l'évolution des paramètres (l'estimation et l'écart-type) du nombre de morts et du nombre de blessés, par rapport au nombre d'expériences réalisées. Dans ces figures, l'intersection entre la ligne discontinue horizontale colorée en rouge et l'axe des coordonnées Y donne la valeur stable du paramètre statistique. Tandis que, l'intersection entre la ligne discontinue verticale colorée en bleu et l'axe des coordonnées X, donne le nombre minimum d'expériences pour atteindre cette valeur stable.

La figure 5.6 présente l'évolution du nombre moyen de morts par rapport au nombre d'expériences réalisées. Selon cette figure, on constate qu'après environ 520 expériences, le nombre moyen de morts tend vers la valeur stable 46,42.

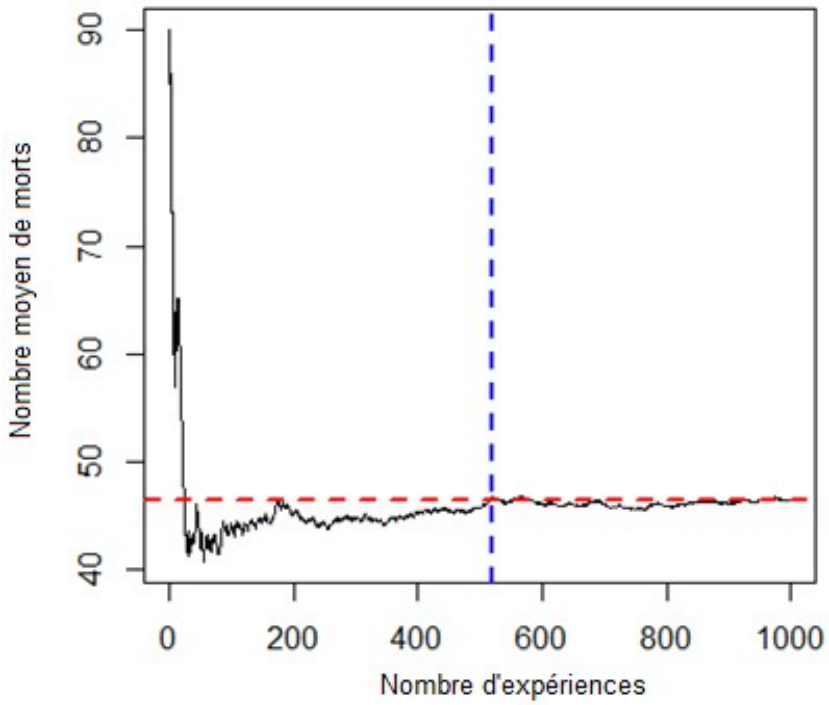


Figure 5.6. Évolution du nombre moyen de morts par rapport au nombre d'expériences réalisées.

La figure 5.7 présente l'écart-type du nombre de décès par rapport au nombre d'expériences effectuées. D'après cette figure, on constate que cet écart-type tend vers une valeur stable 37,60 après environ plus de 350 expériences.

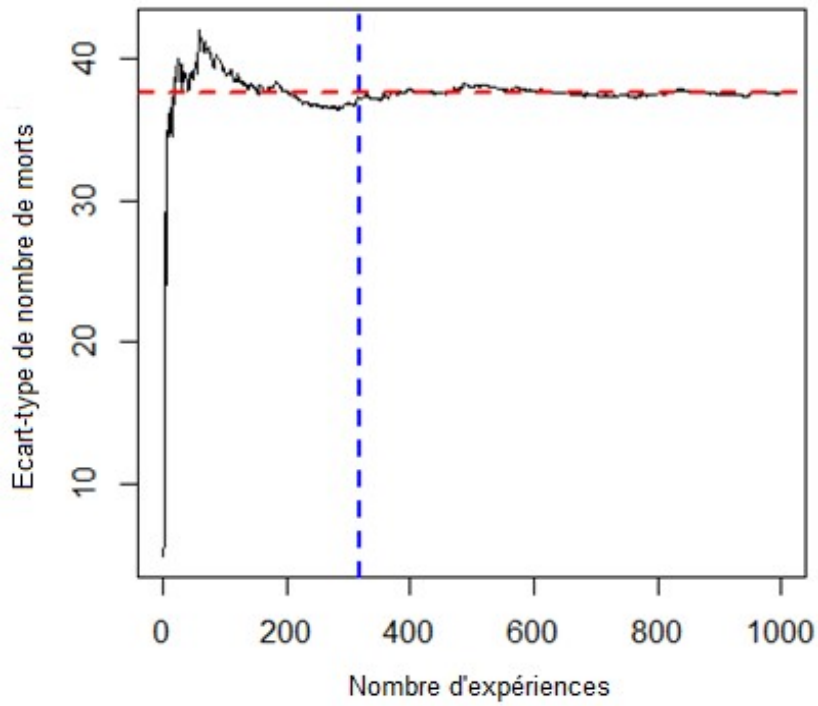


Figure 5.7. Évolution de l'écart-type du nombre de morts par rapport au nombre d'expériences effectuées.

Dans la figure 5.8, nous présentons l'évolution du nombre moyen de blessés par rapport au nombre d'expériences réalisées. D'après cette figure, on constate que cette variable tend vers une valeur stable 78,03 après environ plus de 550 expériences.

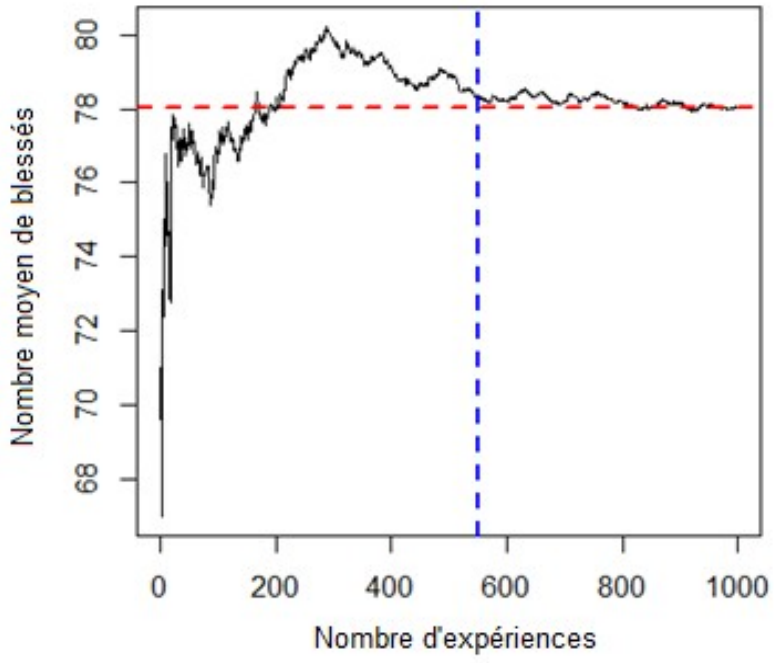


Figure 5.8. Évolution du nombre moyen de blessés par rapport au nombre d'expériences réalisées.

La figure 5.9 présente la variation de l'écart-type du nombre de blessés par rapport au nombre d'expériences effectuées. Selon cette figure, on remarque que cette variable tend également à une valeur stable 19,91 après environ 700 expériences.

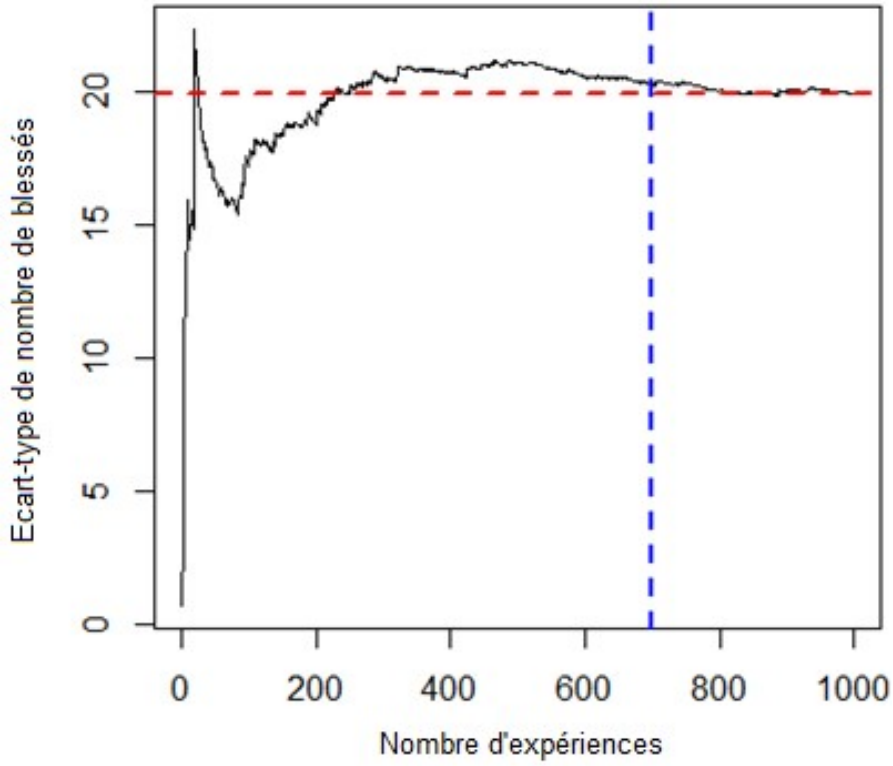


Figure 5.9. Évolution de l'écart-type du nombre de blessés par rapport au nombre d'expériences réalisées.

5.5.2.4. Les résultats généraux tirés des expériences effectuées

Sur la base du tableau 5.12 et les figures 5.6 à 5.9, nous pouvons déduire plusieurs résultats utiles, parmi ceux-ci, on cite :

- Les estimations et les écarts-types des paramètres de sortie tendent à des valeurs stables lorsque le nombre d'expériences est suffisamment grand.
- Dans la plupart des cas, le nombre moyen de blessés est supérieur à celui de morts.
- Le nombre commun d'expériences nécessaire pour atteindre les valeurs stables des moyennes et des écarts-types des paramètres de sortie est d'au moins 700 expériences.
- Le nombre moyen de victimes, à savoir, 46 morts et 78 blessés, représente moins de 50% du nombre moyen de personnes avant l'apparition de l'incendie (300 personnes).
- Comme il a été montré dans la section 4.7, la distribution normale approximative de la valeur moyenne du nombre de morts est $N\ (46.42, 1.19)$, alors que celle du nombre de blessés est $N\ (78.03, 0.63)$.

5.6. Conclusion

La simulation sociale à base d'agents offre un grand avantage pour étudier, analyser et modéliser des phénomènes sociaux complexes comme les systèmes d'évacuation dans les supermarchés, ce qui permet à concevoir des supermarchés avec une structure plus sécurisée, par exemple, en choisissant des emplacements pertinents pour les portes de sortie, les rangées d'étagères et les extincteurs d'incendie afin d'éviter un grand nombre de victimes. On peut appliquer également ce modèle à d'autres types d'immeubles tels que les stades, les théâtres et les hôpitaux. Étant donné la multiplicité et la variété des paramètres qui peuvent être inclus dans le modèle de simulation, il est très utile de tirer profit de la simulation de Monte-Carlo et du concept de l'état stable, ce qui permet d'obtenir des résultats plus fiables et plus stables.

Chapitre 6 : Conclusions générales et travaux futurs

L'objectif principal de ce travail de thèse est de mettre l'accent sur la nouvelle tendance de la fouille de données, à savoir la fouille de données dynamique, et également sur la stabilité des modèles dynamiques utilisés lors de la production ou de la génération de données.

Notre étude est motivée par la grande importance du domaine de la fouille de données dynamique. Car dans la plupart des cas, dans les études de fouille de données statique, on remarque que les données sont toujours supposées des données statiques. Cette supposition donne comme résultats des modèles qui ne sont pas tellement à jour (par exemple, dans les meilleurs cas, en traitent les données des dernières années). Cependant, les données utilisées sont des collections de données qui sont mises à jour continuellement (ajout, suppression ou modification), comme exemple, les séries temporelles, les comportements individuels, les images audiovisuelles. Par conséquent, la prise en considération de ces changements au niveau des bases de données, donne des modèles de fouille de données, à jour et plus utile.

Un autre problème qui a été traité avec la fouille de données dynamique est la disponibilité de données nécessaires pour le processus de fouille de données dynamique. Généralement, dans le processus de fouille de données, on utilise des données historiques qui décrivent les caractéristiques d'un tel système. Dans un tel cas, on est obligé d'attendre une longue durée (plusieurs mois ou plusieurs années) pour collecter les données nécessaires pour réaliser l'étude désirée. De plus, l'opération de collection de données peut rencontrer plusieurs autres obstacles, à savoir les ensembles de données collectées sont de petite taille, ainsi que l'existence de données manquantes ou relativement erronées. Ou bien, dans le cas d'étudier des scénarios potentiels sur des systèmes ou des phénomènes réels ou irréels, comme les incendies, les tremblements de terre ou les attaques nucléaires.

La méthode qu'on a utilisée pour faire face à ces problèmes est basée sur la simulation sociale à base d'agents, ainsi que la méthode de Monte-Carlo et du concept d'état stable. Dans cette méthode, on utilise un modèle dynamique (simulateur) qui représente le plus fidèlement possible le système étudié. Ensuite, on identifie les périodes dans lesquelles le modèle est dans ses états stables, pour avoir l'ensemble de données nécessaires à l'accomplissement du processus désiré de fouille de données dynamique.

Afin de réaliser les objectifs cités précédemment, nous avons présenté d'abord les domaines de la fouille de données et de la fouille de données dynamique, où on a cité par

exemple leurs principaux concepts et techniques. Dans le deuxième chapitre, nous avons présenté quelques modèles dynamiques qu'on peut utiliser dans le processus de fouille de données dynamique, à savoir les réseaux bayésiens dynamiques, les réseaux de neurones dynamiques et les chaines de Markov cachées. Dans le troisième chapitre, nous présentons la simulation sociale à base d'agents qui représente un outil de modélisation puissant qui nous permit d'acquérir l'ensemble de données nécessaires pour un processus de fouille de données dynamique. Ensuite, on a montré comment utiliser la simulation à base d'agents avec le concept d'état stable qui permet de décrire les états stables du modèle dynamique utilisé, ainsi que la méthode de Monte-Carlo qui permet d'avoir des connaissances intéressantes sur la base des données générées par le modèle dynamique.

Pour montrer l'efficacité et l'importance de l'approche proposée, on a traité le problème de l'évacuation dans un supermarché lors d'un incendie, où la simulation à base d'agents sociaux offre un grand avantage pour modéliser ce phénomène social. Ce type de simulateur peut aider les experts tels que les architectes à concevoir des supermarchés avec une structure plus sécurisée, par exemple en sélectionnant des emplacements pertinents pour les portes de sortie, les rangées d'étagères et les extincteurs d'incendie, afin d'éviter un grand nombre de victimes.

Comme travaux futurs, nous visons à considérer plusieurs axes de recherches, dans ce qui suit, on va citer quelques exemples.

Utiliser des algorithmes avancés de fouille de données ou d'apprentissage automatique, tels que les règles d'association floue ou l'apprentissage par renforcement, afin d'obtenir de meilleures connaissances et d'améliorer la précision des prédictions lors de l'étude de systèmes et de phénomènes dynamiques complexes.

Appliquer d'autres méthodes de fouille de données dynamique, par exemple le clustering dynamique et les réseaux bayésiens dynamiques avec la simulation à base d'agents pour étudier d'autres systèmes réels ou virtuels.

Utiliser des algorithmes métaheuristiques (par exemple, l'algorithme de la colonie de fourmis ou l'algorithme Firefly) avec la simulation à base d'agent, afin d'optimiser le processus de fouille de données dynamique.

Bibliographie

[1] M. Naili, M. Bourahla, and M. Naili, "Stability-based model for evacuation system using agent-based social simulation and Monte Carlo method," *International Journal of Simulation and Process Modelling,* 2018.

[2] P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*: Pearson, 2005.

[3] M. H. Dunham, *Data Mining: Introductory and Advanced Topics*: Pearson, 2002.

[4] U. M. Feyyad, "Data mining and knowledge discovery: making sense out of data," *IEEE Expert,* vol. 11, pp. 20-25, 1996.

[5] D. T. Larose, *Discovering Knowledge in Data: An Introduction to Data Mining*, 2004.

[6] P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer, and R. Wirth, *CRISP-DM 1.0 : Step-by-step data mining guide*, 2000.

[7] G. Piatetsky-Shapiro. (2014). *Data Mining Methodology*. Available: https://www.kdnuggets.com/2014/10/crisp-dm-top-methodology-analytics-data-mining-data-science-projects.html

[8] C. Shearer, *The CRISP-DM model: the new blueprint for data mining* vol. 5, 2000.

[9] G. Harper and S. D. Pickett, "Methods for mining HTS data," *Drug Discov Today,* vol. 11, pp. 694-9, Aug 2006.

[10] O. Niaksu, "CRISP Data Mining Methodology Extension for Medical Domain," *Baltic Journal of Modern Computing,* pp. 92-109, 2015.

[11] J. Han, M. Kamber, and J. Pei, "Data Transformation and Data Discretization," in *Data Mining- Concepts and Techniques*, ed: Morgan Kaufmann, 2011, pp. 111-112.

[12] S. M. Weiss and N. Indurkhya, *Predictive Data Mining*: Morgan Kaufmann, 1997.

[13] J. Han, M. Kamber, and J. Pei, "Major Tasks in Data Preprocessing," in *Data Mining Concepts and Techniques*, 3 ed: Morgan Kaufmann, 2011.

[14] L. Breiman, "Bagging Predictors," *Machine Learning,* vol. 24, pp. 123-140, 1996.

[15] L. Breiman, "Random Forests," *Machine Learning,* vol. 45, pp. 5-32, 2001.

[16] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and Regression Trees*, 1 ed.: Taylor & Francis, 1984.

[17] R. Meir and G. Rätsch, "An Introduction to Boosting and Leveraging," vol. 2600, pp. 118-183, 2003.

[18] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer, "An efficient boosting algorithm for combining preferences," *The Journal of Machine Learning Research,* vol. 4, pp. 933-969 2003

[19] C. Lemke, M. Budka, and B. Gabrys, "Metalearning: a survey of trends and technologies," *Artif Intell Rev,* vol. 44, pp. 117-130, 2015.

[20] D. H. Wolpert, "Stacked generalization," *Neural Networks,* vol. 5, pp. 241-259, 1992.

[21] J. Gama and P. Brazdil, "Cascade Generalization," *Machine Learning,* vol. 41, pp. 315-343, 2000.

[22] G. Piatetsky. (2016). *Where Analytics, Data Mining, Data Science were applied in 2016*. Available: https://www.kdnuggets.com/2016/12/poll-analytics-data-mining-data-science-applied-2016.html

[23] S. P. Deshpande and V. M. Thakare, "Data Mining System and Applications: A Review," *International Journal of Distributed and Parallel systems,* vol. 1, pp. 32-44, 2010.

[24] P. Goldschmidt, "Managing the false alarms: A framework for assurance and verification of surveillance monitoring," *Information Systems Frontiers,* vol. 9, pp. 541-556, 2007.

[25] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," pp. 207-216, 1993.

[26] N. Jothi, N. A. A. Rashid, and W. Husain, "Data Mining in Healthcare – A Review," *Procedia Computer Science,* vol. 72, pp. 306-313, 2015.

[27] E. Rojas, J. Munoz-Gama, M. Sepulveda, and D. Capurro, "Process mining in healthcare: A literature review," *J Biomed Inform,* vol. 61, pp. 224-36, Jun 2016.
[28] Y. Hamuro, N. Katoh, Y. Matsuda, and K. Yada, "Mining Pharmacy Data Helps to Make Profits," *Data Mining and Knowledge Discovery,* vol. 2, pp. 391-398, 1998.
[29] G. M. Weiss, "Data Mining in Telecommunications," in *Data Mining and Knowledge Discovery Handbook*, R. L. Maimon O., Ed., ed: Springer, Boston, MA, 2005.
[30] G. Hongjiu, "Data Mining in the Application of E-Commerce Website," presented at the Intelligence Computation and Evolutionary Computation, Berlin, Heidelberg, 2013.
[31] B. Bakhshinategh, O. R. Zaiane, S. ElAtia, and D. Ipperciel, "Educational data mining applications and tasks: A survey of the last 10 years," *Education and Information Technologies,* vol. 23, pp. 537-553, 2017.
[32] C. Romero and S. Ventura, "Data mining in education," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery,* vol. 3, pp. 12-27, 2013.
[33] J. Mena, *Investigative Data Mining for Security and Criminal Detection*: Butterworth-Heinemann, 2003.
[34] M. R. Keyvanpour, M. Javideh, and M. R. Ebrahimi, "Detecting and investigating crime by means of data mining: a general crime matching framework," *Procedia Computer Science,* vol. 3, pp. 872-880, 2011.
[35] C. C. Aggarwal and C. Zhai, "An Introduction to Text Mining," in *Mining Text Data*, ed: Springer, Boston, MA, 2012, pp. 1-10.
[36] K. B. Cohen and L. Hunter, "Getting started in text mining," *PLoS Comput Biol,* vol. 4, p. e20, Jan 2008.
[37] T. Srivastava, P. Desikan, and V. Kumar, "Web Mining – Concepts, Applications and Research Directions," in *Foundations and Advances in Data Mining*, T. Y. L. Wesley Chu, Ed., ed: Springer Berlin Heidelberg New York, 2005.
[38] M. M. Gaber, A. Zaslavsky, and S. Krishnaswamy, "A Survey of Classification Methods in Data Streams," vol. 31, pp. 39-59, 2007.
[39] K. Vanhoof and I. Lauth, "Application Challenges for Ubiquitous Knowledge Discovery," vol. 6202, pp. 108-125, 2010.
[40] L. Zeng, L. Li, L. Duan, K. Lu, Z. Shi, M. Wang, W. Wu, and P. Luo, "Distributed data mining: a survey," *Information Technology and Management,* vol. 13, pp. 403-409, 2012.
[41] W. Gan, J. C.-W. Lin, H.-C. Chao, and J. Zhan, "Data mining in distributed environment: a survey," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery,* vol. 7, p. e1216, 2017.
[42] X. Li, S.-K. Ng, and J. T. L. Wang, *Biological Data Mining and Its Applications in Healthcare* 1ed.: World Scientific Pub Co Inc, 2013.
[43] J. Mennis and D. Guo, "Spatial data mining and geographic knowledge discovery—An introduction," *Computers, Environment and Urban Systems,* pp. 403–408, 2009.
[44] T. S. Körting, L. M. Garcia Fonseca, and G. Câmara, "GeoDMA—Geographic Data Mining Analyst," *Computers & Geosciences,* vol. 57, pp. 133-145, 2013.
[45] S. Sumathi and S. N. Sivanandam, "Major and Privacy Issues in Data Mining and Knowledge Discovery," in *Introduction to Data Mining and its Applications*. vol. 29, ed, 2006, pp. 271-291.
[46] J. Han, M. Kamber, and J. Pei, "Major Issues in Data Mining," in *Data Mining. Concepts and Techniques*, 3 ed: Morgan Kaufmann, 2011.
[47] A. Famili, W. Shen, R. Weber, and E. Simoudis, "Data preprocessing and intelligent data analysis," *Intelligent Data Analysis,* vol. 1, pp. 3-23, 1997.
[48] V. Raghavan and A. Hafez, "Dynamic Data Mining," presented at the Intelligent Problem Solving. Methodologies and Approaches: 13th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, 2000.
[49] R. Weber, "Dynamic Data Mining," in *Encyclopedia of Data Warehousing and Mining*, J. Wang, Ed., 2 ed: Information Science Reference, 2009.

[50] J. Du, J. Zhou, C. Li, and L. Yang, "An overview of dynamic data mining," presented at the 3rd International Conference on Informative and Cybernetics for Computational Social Systems (ICCSS), Jinzhou, China, 2016.
[51] T. Mitsa, "Temporal data types and preprocessing," in *Temporal data mining*, 1 ed: Chapman & Hall/CRC 2010, pp. 22-26.
[52] T.-c. Fu, "A review on time series data mining," *Engineering Applications of Artificial Intelligence,* vol. 24, pp. 164-181, 2011.
[53] L. Lin, T. Risch, M. Sköld, and D. Badal, "Indexing values of time sequences," in *CIKM '96 Proceedings of the fifth international conference on Information and knowledge management*, Rockville, Maryland, USA, 1996, pp. 223-232
[54] Y. Dodge, "Série chronologique," in *Statistique: Dictionnaire encyclopédique* ed: Springer, 2007
[55] J. Han, M. Kamber, and J. Pei, "Mining Sequence Data: Time-Series, Symbolic Sequences, and Biological Sequences," in *Data Mining: Concepts and Techniques*, 3 ed: Morgan Kaufmann; 3 edition 2011, pp. 587-590.
[56] E. Dagum, "Time Series Modelling and Decomposition," *Statistica,* vol. 70, 2013.
[57] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom, "Models and issues in data stream systems," presented at the Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, Madison, Wisconsin, 2002.
[58] J. Leskovec, A. Rajaraman, and J. Ullman, *Mining Of Massive Datasets*, 2 ed.: Cambridge University Press, 2016.
[59] J. Guajardo, R. Weber, and J. Miranda, "A Forecasting Methodology Using Support Vector Regression and Dynamic Feature Selection," *Journal of Information & Knowledge Management,* vol. 05, pp. 329-335, 2006.
[60] G. Krempl, M. Spiliopoulou, J. Stefanowski, I. Žliobaite, D. Brzeziński, E. Hüllermeier, M. Last, V. Lemaire, T. Noack, A. Shaker, and S. Sievi, "Open challenges for data stream mining research," *ACM SIGKDD Explorations Newsletter,* vol. 16, pp. 1-10, 2014.
[61] S. García, J. Luengo, and F. Herrera, *Data preprocessing in data mining*: springer, 2015.
[62] S. Ramírez-Gallego, B. Krawczyk, S. García, M. Woźniak, and F. Herrera, "A survey on data preprocessing for data stream mining: Current status and future directions," *Neurocomputing,* vol. 239, pp. 39-57, 2017.
[63] N. Lu, J. Lu, G. Zhang, and R. Lopez de Mantaras, "A concept drift-tolerant case-base editing technique," *Artificial Intelligence,* vol. 230, pp. 108-133, 2016.
[64] A. Tsymbal, *The Problem of Concept Drift: Definitions and Related Work*, 2004.
[65] D. W. Aha, D. Kibler, and M. K. Albert, "Instance-Based Learning Algorithms," *Machine Learning,* vol. 6, pp. 37-66, 1991.
[66] J. Beringer and E. Hüllermeier, "Efficient instance-based learning on data streams," *Intelligent Data Analysis,* vol. 11, pp. 627-650, 2007.
[67] A. Shaker and E. Hüllermeier, "IBLStreams: a system for instance-based classification and regression on data streams," *Evolving Systems,* vol. 3, pp. 235-249, 2012.
[68] G. I. Webb, "Contrary to Popular Belief Incremental Discretization can be Sound, Computationally Efficient and Extremely Useful for Streaming Data," in *2014 IEEE International Conference on Data Mining*, 2014, pp. 1031-1036.
[69] Y. Ben-Haim and E. Tom-Tov, "A Streaming Parallel Decision Tree Algorithm," *J. Mach. Learn. Res.,* vol. 11, pp. 849-872, 2010.
[70] A. Gupta and F. X. Zane, "Counting inversions in lists," presented at the Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms, Baltimore, Maryland, 2003.
[71] S. Guha and A. McGregor, "Stream Order and Order Statistics: Quantile Estimation in Random-Order Streams," *SIAM J. Comput.,* vol. 38, pp. 2044-2059, 2009.
[72] A. Joentgen, L. Mikenina, R. Weber, and H. J. Zimmermann, "Dynamic fuzzy data analysis based on similarity between functions," *Fuzzy Sets and Systems,* vol. 105, pp. 81-90, 1999.

[73] Y. Wang, "An Incremental Classification Algorithm for Mining Data with Feature Space Heterogeneity," *Mathematical Problems in Engineering,* vol. 2014, pp. 1-9, 2014.
[74] C.-W. Lin, G.-C. Lan, and T.-P. Hong, "An incremental mining algorithm for high utility itemsets," *Expert Systems with Applications,* vol. 39, pp. 7173-7180, 2012.
[75] H.-Y. Chang, J.-C. Lin, M.-L. Cheng, and S.-C. Huang, "A Novel Incremental Data Mining Algorithm Based on FP-growth for Big Data," presented at the International Conference on Networking and Network Applications (NaNA), Hakodate, Japan, 2016.
[76] Y. Jing, T. Li, H. Fujita, B. Wang, and N. Cheng, "An incremental attribute reduction method for dynamic data mining," *Information Sciences,* vol. 465, pp. 202-218, 2018.
[77] M.-Y. Chen and E. Lughofer, "Data stream mining and soft computing applications," *Applied Soft Computing,* vol. 68, pp. 667-668, 2018.
[78] R. Mythily, A. Banu, and S. Raghunathan, "Clustering Models for Data Stream Mining," *Procedia Computer Science,* vol. 46, pp. 619-626, 2015.
[79] L. Ye and E. Keogh, "Time series shapelets: A New Primitive for Data Mining," in *KDD '09* 2009, pp. 947-956.
[80] S. G. Djorgovski, M. J. Graham, C. Donalek, A. A. Mahabal, A. J. Drake, M. Turmon, and T. Fuchs, "Real-time data mining of massive data streams from synoptic sky surveys," *Future Generation Computer Systems,* vol. 59, pp. 95-104, 2016.
[81] X. Zhao and D. W. Rosen, "A data mining approach in real-time measurement for polymer additive manufacturing process with exposure controlled projection lithography," *Journal of Manufacturing Systems,* vol. 43, pp. 271-286, 2017.
[82] S. Sayad, *Real Time Data Mining*: Self-Help Publishers, 2011.
[83] S. Nasreen, M. A. Azam, K. Shehzad, U. Naeem, and M. A. Ghazanfar, "Frequent Pattern Mining Algorithms for Finding Associated Frequent Patterns for Data Streams: A Survey," *Procedia Computer Science,* vol. 37, pp. 109-116, 2014.
[84] C. C. Aggarwal and J. Han, *Frequent Pattern Mining*: Springer Publishing Company, Incorporated, 2014.
[85] Y. Pitarch, A. Laurent, M. Plantevit, and P. Poncelet, "Multidimensional Data Stream Summarization Using Extended Tilted-Time Windows," in *2009 International Conference on Advanced Information Networking and Applications Workshops*, 2009, pp. 250-254.
[86] A. Cuzzocrea, "CAMS: OLAPing Multidimensional Data Streams Efficiently," Berlin, Heidelberg, 2009, pp. 48-62.
[87] J. Leskovec, A. Rajaraman, and J. D. Ullman, "Mining Data Streams," in *Mining of Massive Datasets*, 2 ed New York, NY, USA: Cambridge University Press, 2011, pp. 131-162.
[88] L. Khan and W. Fan, "Tutorial: Data Stream Mining and Its Applications," in *Database Systems for Advanced Applications* Berlin, Heidelberg, 2012, pp. 328-329.
[89] J. Gama and M. M. Gaber, *Learning from Data Streams.*
Processing Techniques in Sensor Networks: Springer-Verlag Berlin Heidelberg, 2007.
[90] E. Ikonomovska, S. Loskovska, and D. Gjorgjevikj, *A survey of stream data mining*, 2007.
[91] A. R. Ganguly, J. Gama, O. A. Omitaomu, M. M. Gaber, and R. R. Vatsavai, *Knowledge Discovery from Sensor Data*: CRC Press, Inc., 2008.
[92] J. Gama, *Knowledge Discovery from Data Streams*: Chapman \& Hall/CRC, 2010.
[93] M. Sayed-Mouchaweh and E. Lughofer, *Learning in Non-Stationary Environments: Methods and Applications*: Springer Publishing Company, Incorporated, 2012.
[94] W. Graham, *Descriptive and Predictive Analytics*: Springer, New York, NY, 2011.
[95] H. Gulati, "Predictive analytics using data mining technique," in *2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom)*, 2015, pp. 713-716.
[96] E. Alpaydin, *Introduction to Machine Learning*, 2014.
[97] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM Computing Surveys,* vol. 31, pp. 264-323, 1999.

[98] A. Giloni and M. Padberg, "Alternative methods of linear regression," *Mathematical and Computer Modelling,* vol. 35, pp. 361-374, 2002.
[99] D. A. Freedman, *Statistical Models: Theory and Practice*: Cambridge University Press, 2005.
[100] D. Birkes and Y. Dodge, *Alternative Methods of Regression*: Wiley 1993.
[101] J. Hipp, U. Güntzer, and G. Nakhaeizadeh, "Algorithms for association rule mining --- a general survey and comparison," *ACM SIGKDD Explorations Newsletter,* vol. 2, pp. 58-64, 2000.
[102] L. C. Reddy and V. M, "A Review on Data mining from Past to the Future," *International Journal of Computer Applications,* vol. 15, pp. 19-22, 2011.
[103] J. R. Quinlan, "Induction of decision trees," *Machine Learning,* vol. 1, pp. 81-106, 1986.
[104] J. R. Quinlan, *C4.5: Programs for Machine Learning*: Morgan Kaufmann Publishers, 1993.
[105] J. Pearl, *Probabilistic reasoning in intelligent systems: networks of plausible inference*: Morgan Kaufmann Publishers Inc., 1988.
[106] J. Pearl, *Causality: Models, Reasoning and Inference*, 2 ed.: Cambridge University Press, 2009.
[107] D. Margaritis, "Learning Bayesian Network Model Structure from Data," PhD thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA, 2003.
[108] D. J. Livingstone, "Overview of Artificial Neural Networks," in *Artificial Neural Networks. Methods and Applications*, 1 ed: Humana Press, 2009, pp. 14-22.
[109] H. Nielsen, J. Engelbrecht, S. Brunak, and G. V. Heijne, "A Neural Network Method for Identification of Prokaryotic and Eukaryotic Signal Peptides and Prediction of their Cleavage Sites," *International Journal of Neural Systems,* vol. 08, pp. 581-599, 1997.
[110] B. Story, *Neural Networks for Beginners: An Easy-to-Use Manual for Understanding Artificial Neural Network Programming*: CreateSpace Independent Publishing Platform, 2017.
[111] S. U. Amin, K. Agarwal, and R. Beg, "Genetic neural network based data mining in prediction of heart disease using risk factors," pp. 1227-1231, 2013.
[112] S.-S. So and M. Karplus, "Evolutionary Optimization in Quantitative Structure−Activity Relationship: An Application of Genetic Neural Networks," *Journal of Medicinal Chemistry,* vol. 39, pp. 1521-1530, 1996/01/01 1996.
[113] S. I. Gallant, "Connectionist expert systems," *Commun. ACM,* vol. 31, pp. 152-169, 1988.
[114] M. W. Gardner and S. R. Dorling, "Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences," *Atmospheric Environment,* vol. 32, pp. 2627-2636, 1998.
[115] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proceedings of the National Academy of Sciences,* vol. 79, pp. 2554-2558, 1982.
[116] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE,* vol. 78, pp. 1464-1480, 1990.
[117] G. P. Zhang, "Neural Networks For Data Mining," in *The Data Mining and Knowledge Discovery Handbook* ed: Springer, 2009, pp. 419-444.
[118] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review,* vol. 65, pp. 386-408, 1958.
[119] C. von der Malsburg, *Frank Rosenblatt: Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*, 1986.
[120] D. E. Rumelhart, Ed., *Parallel distributed processing: explorations in the microstructure of cognition, vol. 1: foundations*. MIT Press, 1986, p.^pp. Pages.
[121] A. Amini, T. Y. Wah, and H. Saboohi, "On Density-Based Data Streams Clustering Algorithms: A Survey," *Journal of Computer Science and Technology,* vol. 29, pp. 116-141, 2014.
[122] D. O. Cardoso, F. M. G. França, and J. Gama, "WCDS: A Two-Phase Weightless Neural System for Data Stream Clustering," *New Generation Computing,* vol. 35, pp. 391-416, 2017.

[123] G. Rzevski, P. Skobelev, I. Minakov, and S. Volman, "Dynamic pattern discovery using multi-agent technology," in *TELE-INFO'07 Proceedings of the 6th WSEAS Int. Conference on Telecommunications and Informatics* Dallas, Texas, 2007, pp. 75-81

[124] J. de Lope and D. Maravall, "Data clustering using a linear cellular automata-based algorithm," *Neurocomputing,* vol. 114, pp. 86-91, 2013.

[125] R. Weber and J. Guajardo, "Dynamic Data Mining for Improved Forecasting in Logistics and Supply Chain Management," in *Dynamics in Logistics*, Berlin, Heidelberg, 2008, pp. 57-63.

[126] C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu, "On High Dimensional Projected Clustering of Data Streams," *Data Mining and Knowledge Discovery,* vol. 10, pp. 251-273, 2005/05/01 2005.

[127] C. C. Aggarwal, *Data streams. Models and algorithms*, 1 ed.: Springer US, 2007.

[128] P. Parita and P. Singh, "A review on tree based incremental classification," *International Journal of Computer Science and Information Technologies,* vol. 5, 2014.

[129] A. S. Al-Hegami, "Classical and incremental classification in data mining process," *International Journal of Computer Science and Network Security,* vol. 7, 2007.

[130] P. E. Utgoff, "Incremental Induction of Decision Trees," *Machine Learning,* vol. 4, pp. 161-186, November 01 1989.

[131] C. Sam and W. Fai, "An incremental decision tree learning methodology regarding attributes in medical data mining," in *2009 International Conference on Machine Learning and Cybernetics*, 2009, pp. 1694-1699.

[132] P. E. Utgoff, N. C. Berkman, and J. A. Clouse, "Decision Tree Induction Based on Efficient Tree Restructuring," *Machine Learning,* vol. 29, pp. 5-44, 1997.

[133] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and Regression Trees*: Chapman and Hall/CRC, 1984.

[134] S. L. Crawford, "Extensions to the CART algorithm," *International Journal of Man-Machine Studies,* vol. 31, pp. 197-217, 1989.

[135] P. Domingos and G. Hulten, "Mining high-speed data streams," pp. 71-80, 2000.

[136] J. Gama, R. Fernandes, and R. Rocha, "Decision trees for mining data streams," *Intelligent Data Analysis,* vol. 10, pp. 23-45, 2006.

[137] D. Uderino, S. Floriani, F. Caferini, and K. Rodrigerz, "A Combined Method of Hidden Markov Model and K-Means Clustering to Temporary Datasets Grouping," *Journal of Mechatronics,* vol. 3, pp. 237-246, 2015.

[138] W. Pieczynski, "Multisensor triplet Markov chains and theory of evidence," *International Journal of Approximate Reasoning,* vol. 45, pp. 1-16, 2007.

[139] M. E. Y. Boudaren, E. Monfrini, W. Pieczynski, and A. Aïssani, "Dempster–Shafer fusion of multisensor signals in nonstationary Markovian context," *EURASIP Journal on Advances in Signal Processing,* vol. 2012, 2012.

[140] P. Lanchantin and W. Pieczynski, "Unsupervised restoration of hidden nonstationary Markov chains using evidential priors," *IEEE Transactions on Signal Processing,* vol. 53, pp. 3091-3098, 2005.

[141] M. E. Y. Boudaren, E. Monfrini, and W. Pieczynski, "Unsupervised Segmentation of Random Discrete Data Hidden With Switching Noise Distributions," *IEEE Signal Processing Letters,* vol. 19, pp. 619-622, 2012.

[142] C. Zhai, "A Brief Note on the Hidden Markov Models (HMMs)," 2003.

[143] K. H. Choo, J. C. Tong, and L. Zhang, "Recent Applications of Hidden Markov Models in Computational Biology," *Genomics, Proteomics & Bioinformatics,* vol. 2, pp. 84-96, 2004.

[144] P. Dymarski, *Hidden Markov Models. Theory and Applications*: InTech, 2011.

[145] H. Lähdesmäki and I. Shmulevich, "Learning the structure of dynamic Bayesian networks from time series and steady state measurements," *Machine Learning,* vol. 71, pp. 185-217, 2008.

[146] M. Grzegorczyk, "A non-homogeneous dynamic Bayesian network with a hidden Markov model dependency structure among the temporal data points," *Machine Learning,* vol. 102, pp. 155-207, 2015.
[147] F. Dondelinger, S. Lèbre, and D. Husmeier, "Non-homogeneous dynamic Bayesian networks with Bayesian regularization for inferring gene regulatory networks with gradually time-varying structure," *Machine Learning,* vol. 90, pp. 191-230, 2012.
[148] M. Grzegorczyk and D. Husmeier, "Non-homogeneous dynamic Bayesian networks for continuous data," *Machine Learning,* vol. 83, pp. 355-419, 2011.
[149] Y. Jia and J. Huan, "Constructing non-stationary Dynamic Bayesian Networks with a flexible lag choosing mechanism," *BMC Bioinformatics,* vol. 11 Suppl 6, p. S27, Oct 07 2010.
[150] S. H. Nielsen and T. D. Nielsen, "Adapting Bayes network structures to non-stationary domains," *International Journal of Approximate Reasoning,* vol. 49, pp. 379-397, 2008.
[151] C. Gonzales, S. Dubuisson, and C. Manfredotti, "A New Algorithm for Learning Non-Stationary Dynamic Bayesian Networks With Application to Event Detection," in *28th International Florida Artificial Intelligence Research Society Conference Intelligence (FLAIRS-28)*, Hollywood, Florida, 2015, pp. 564-569.
[152] Shijiazhu and Y. Wang, "Modelling non-stationary gene regulatory process with hidden Markov Dynamic Bayesian Network," presented at the 2012 IEEE International Conference on Bioinformatics and Biomedicine Philadelphia, PA, USA 2012.
[153] R. Tibshirani, "Regression Shrinkage and Selection Via the Lasso," *Journal of the Royal Statistical Society, Series B,* vol. 58, pp. 267-288, 1994.
[154] J. Chiquet, A. Smith, G. Grasseau, C. Matias, and C. Ambroise, "SIMoNe: Statistical Inference for MOdular NEtworks," *Bioinformatics,* vol. 25, pp. 417-8, Feb 1 2009.
[155] S. Lebre, "Inferring dynamic genetic networks with low order independencies," *Stat Appl Genet Mol Biol,* vol. 8, p. Article 9, 2009.
[156] R. Tibshirani, "Regression Shrinkage and Selection via the Lasso," *Journal of the Royal Statistical Society. Series B (Methodological),* vol. 58, pp. 267-288, 1996.
[157] M. Lukoševičius and H. Jaeger, "Reservoir computing approaches to recurrent neural network training," *Computer Science Review,* vol. 3, pp. 127-149, 2009.
[158] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," presented at the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, 2013.
[159] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, "End-to-end attention-based large vocabulary speech recognition," presented at the 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Shanghai, China 2016.
[160] Y. Miao, M. Gowayyed, and F. Metze, "EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding," pp. 167-174, 2015.
[161] W. Bao, J. Yue, and Y. Rao, "A deep learning framework for financial time series using stacked autoencoders and long-short term memory," *PLoS One,* vol. 12, p. e0180944, 2017.
[162] F. A. Gers, D. Eck, and J. Schmidhuber, "Applying LSTM to Time Series Predictable Through Time-Window Approaches," in *12th Italian Workshop on Neural Nets*, Vietri sul Mare, Salerno, Italy, 2001, pp. 193-200.
[163] C. L. Giles, S. Lawrence, and A. C. Tsoi, "Noisy Time Series Prediction using Recurrent Neural Networks and Grammatical Inference," *Machine Learning,* vol. 44, pp. 161-183, 2001.
[164] H. Cruse, *Neural Networks as Cybernetic Systems*: Thieme Medical Publishers, Incorporated, 1996.
[165] A. P. Trischler and G. M. D'Eleuterio, "Synthesis of recurrent neural networks for dynamical system simulation," *Neural Netw,* vol. 80, pp. 67-78, Aug 2016.
[166] D. J. Amit, *Modeling Brain Function: The World of Attractor Neural Networks*: Cambridge University Press 1992.

[167] V. S. Afraimovich, M. I. Rabinovich, and P. Varona, "Heteroclinic Contours in Neural Ensembles and the Winnerless Competition Principle," *International Journal of Bifurcation and Chaos,* vol. 14, pp. 1195-1208, 2004.
[168] C. Eliasmith and C. H. Anderson, *Neural Engineering.Computation, Representation, and Dynamics in Neurobiological Systems*: MIT Press, 2004.
[169] K. Kaneko and I. Tsuda, "Chaotic itinerancy," *Chaos,* vol. 13, pp. 926-36, Sep 2003.
[170] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE,* vol. 78, pp. 1550-1560, 1990.
[171] R. J. Williams and D. Zipser, "A Learning Algorithm for Continually Running Fully Recurrent Neural Networks," *Neural Computation,* vol. 1, pp. 270-280, 1989.
[172] L. A. Feldkamp, D. V. Prokhorov, C. F. Eagen, and F. Yuan, "Enhanced Multi-Stream Kalman Filter Training for Recurrent Networks," *Nonlinear Modeling,* pp. 29-53, 1998.
[173] F.-S. Tsung and G. W. Cottrell, "Phase-space learning," presented at the Proceedings of the 7th International Conference on Neural Information Processing Systems, Denver, Colorado, 1994.
[174] C. A. L. Bailer-Jones, D. J. C. MacKay, and P. J. Withers, "A recurrent neural network for modelling dynamical systems," *Network: Computation in Neural Systems,* vol. 9, pp. 531-547, 2009.
[175] P. Davidsson, "Agent Based Social Simulation: A Computer Science View," *Journal of Artificial Societies and Social Simulation,* vol. 5, 2002.
[176] T. Shingo, S. David, and R. Juliette, *Advancing Social Simulation: The First World Congress*: Springer Japan, 2007.
[177] R. E. Shannon, "Simulation modeling and methodology," in *Winter Simulation Conference*, 1976, pp. 9-15.
[178] P. A. Fishwick, "Computer simulation: Growth through extension," *Transactions of the Society for Computer Simulation,* vol. 14, pp. 13-23, 1997.
[179] A. Kuper, *The Social Science Encyclopedia* vol. 31, 2005.
[180] F. Wang, K. M. Carley, D. Zeng, and W. Mao, "Social Computing: From Social Informatics to Social Intelligence," *IEEE Intelligent Systems,* vol. 22, pp. 79-83, 2007.
[181] D. Zeng, F. Wang, and K. M. Carley, "Guest Editors' Introduction: Social Computing," *IEEE Intelligent Systems,* vol. 22, pp. 20-22, 2007.
[182] R. Conte, N. Gilbert, and J. S. Sichman, "MAS and Social Simulation: A Suitable Commitment," Berlin, Heidelberg, 1998, pp. 1-9.
[183] S. J. Taylor, *Agent-based Modeling and Simulation*: PALGRAVE MACMILLAN, 2014.
[184] N. R. Jennings and M. J. Wooldridge, *Agent Technology Foundations, Applications, and Markets*, 1 ed.: Springer-Verlag Berlin Heidelberg, 1998.
[185] J. Ferber, *Multi-Agent System: An Introduction to Distributed Artificial Intelligence* vol. 4, 2001.
[186] V. Grimm, U. Berger, F. Bastiansen, S. Eliassen, V. Ginot, J. Giske, J. Goss-Custard, T. Grand, S. K. Heinz, G. Huse, A. Huth, J. U. Jepsen, C. Jørgensen, W. M. Mooij, B. Müller, G. Pe'er, C. Piou, S. F. Railsback, A. M. Robbins, M. M. Robbins, E. Rossmanith, N. Rüger, E. Strand, S. Souissi, R. A. Stillman, R. Vabø, U. Visser, and D. L. DeAngelis, "A standard protocol for describing individual-based and agent-based models," *Ecological Modelling,* vol. 198, pp. 115-126, 2006.
[187] D. Helbing, " Agent-Based Modeling," in *Social Self-Organization*, D. Helbing, Ed., 1 ed Heidelberg: Springer-Verlag Berlin 2012, pp. 25-70.
[188] J. Doran, "Agent Design for Agent-Based Modelling," in *Agent-Based Computational Modelling Applications in Demography, Social, Economic and Environmental Sciences*, T. F. Francesco C. Billari, Alexia Prskawetz, Jürgen Scheffran, Ed., ed, 2006, pp. 215-223.
[189] I. García-Magariño, A. Gómez-Rodríguez, J. C. González-Moreno, and G. Palacios-Navarro, "PEABS: A Process for developing Efficient Agent-Based Simulators," *Engineering Applications of Artificial Intelligence,* vol. 46, pp. 104-112, 2015.

[190] I. Nikolic and A. Ghorbani, "A method for developing agent-based models of socio-technical systems," presented at the International Conference on Networking, Sensing and Control, Delft, Netherlands, 2011.
[191] I. García-Magariño, G. Palacios-Navarro, and R. Lacuesta, "TABSAOND: A technique for developing agent-based simulation apps and online tools with nondeterministic decisions," *Simulation Modelling Practice and Theory,* vol. 77, pp. 84-107, 2017.
[192] S. Abar, G. K. Theodoropoulos, P. Lemarinier, and G. M. P. O'Hare, "Agent Based Modelling and Simulation tools: A review of the state-of-art software," *Computer Science Review,* vol. 24, pp. 13-33, 2017.
[193] I. García-Magariño and R. Lacuesta, "ABS-SmartPriority: An Agent-Based Simulator of Strategies for Managing Self-Reported Priorities in Smart Cities," *Wireless Communications and Mobile Computing,* vol. 2017, pp. 1-9, 2017.
[194] N. Hooshangi and A. Alesheikh, "Developing an Agent-Based Simulation System for Post-Earthquake Operations in Uncertainty Conditions: A Proposed Method for Collaboration among Agents," *ISPRS International Journal of Geo-Information,* vol. 7, p. 27, 2018.
[195] F. Lamy, T. Bossomaier, and P. Perez, "An ontologic agent-based model of recreational polydrug use: SimUse," *International Journal of Simulation and Process Modelling,* vol. 10, p. 207, 2015.
[196] V. Nascimento, M. J. Viamonte, A. Canito, and N. Silva, "An agent-based electronic market simulator enhanced with ontology matching services and emergent social networks," *International Journal of Simulation and Process Modelling,* vol. 10, p. 265, 2015.
[197] M. Resta, "An agent-based simulator driven by variants of Self-Organizing Maps," *Neurocomputing,* vol. 147, pp. 207-224, 2015.
[198] I. García-Magariño, "ABSTUR: An Agent-based Simulator for Tourist Urban Routes," *Expert Systems with Applications,* vol. 42, pp. 5287-5302, 2015.
[199] J. W. Bae, E. Paik, K. Kim, K. Singh, and M. Sajjad, "Combining Microsimulation and Agent-based Model for Micro-level Population Dynamics," *Procedia Computer Science,* vol. 80, pp. 507-517, 2016.
[200] D. Phan and F. Amblard, *Agent-based Modelling and Simulation in the Social and Human Sciences*: The Bardwell Press, 2007.
[201] X. Li, W. Mao, D. Zeng, and F.-Y. Wang, "Agent-Based Social Simulation and Modeling in Social Computing," presented at the International Conference on Intelligence and Security Informatics, Taipei, Taiwan, 2008.
[202] A. M. Law, "Systems, Models, and Simulation," in *Simulation Modeling and Analysis*, 5 ed New York.USA: McGraw-Hill Education, 2015.
[203] P. Caplat, M. Anand, and C. Bauch, "Symmetric competition causes population oscillations in an individual-based model of forest dynamics," *Ecological Modelling,* vol. 211, pp. 491-500, 2008.
[204] E. Ch'ng, "An Artificial Life-Based Vegetation Modelling Approach for Biodiversity Research," in *Nature-Inspired Informatics for Intelligent Applications and Knowledge Discovery: Implications in Business, Science, and Engineering*, R. Chiong, Ed., ed: IGI Global: Hershey, 2010, pp. 68-118.
[205] E. Wirth, G. Szabó, and A. Czinkóczky, "Measure of Landscape Heterogeneity by Agent-Based Methodology," *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences,* vol. III-8, pp. 145-151, 2016.
[206] H. P. N. Hughes, C. W. Clegg, M. A. Robinson, and R. M. Crowder, "Agent-based modelling and simulation: The potential contribution to organizational psychology," *Journal of Occupational and Organizational Psychology,* vol. 85, pp. 487-502, 2012.
[207] M. Niazi and A. Hussain, "Agent-based tools for modeling and simulation of self-organization in peer-to-peer, ad hoc, and other complex networks," *IEEE Communications Magazine,* vol. 47, pp. 166-173, 2009.

[208] F. Lorig and I. J. Timm, "How to model the "human factor" for agent-based simulation in social media analysis?: work in progress paper," in *Agent Directed Simulation* Tampa, Florida 2014
[209] M. Kvasnička, "Viral Video Diffusion in a Fixed Social Network: An Agent-based Model," *Procedia Economics and Finance,* vol. 12, pp. 334-342, 2014.
[210] L. Vanhaverbeke and C. Macharis, "An agent-based model of consumer mobility in a retail environment," *Procedia - Social and Behavioral Sciences,* vol. 20, pp. 186-196, 2011.
[211] F. M. Stefan and A. P. F. Atman, "Is there any connection between the network morphology and the fluctuations of the stock market index?," *Physica A: Statistical Mechanics and its Applications,* vol. 419, pp. 630-641, 2015.
[212] G. D. P. A. Aschwanden, T. Wullschleger, H. Müller, and G. Schmitt, "Agent based evaluation of dynamic city models," *Automation in Construction,* vol. 22, pp. 81-89, 2012.
[213] P. A. Gagniuc, "The Steady State of a Markov Chain," in *Markov Chains: From Theory to Implementation and Experimentation*, 1 ed: Wiley, 2017, pp. 46-59.
[214] E. R. Scheinerman, *Invitation to Dynamical Systems*, 1 ed.: Prentice Hall, 1995.
[215] A. M. Law, "Agent-Based Simulation and System Dynamics," in *Simulation Modeling and Analysis*, 5 ed: McGraw-Hill, 2013.
[216] N. Metropolis and S. Ulam, "The Monte Carlo Method," *Journal of the American Statistical Association,* vol. 44, pp. 335-341, 1949/09/01 1949.
[217] N. Metropolis, "The Beginning of the Monte Carlo Method," *Los Alamos Science,* 1987.
[218] D. B. Thomas and W. Luk, "Estimation of Sample Mean and Variance for Monte-Carlo Simulations," in *Int. Conf. on Field-Programmable Technology*, 2008.
[219] A. Agresti, C. A. Franklin, and B. Klingenberg, "How Sample Means Vary Around the Population Mean," in *Statistics: The Art and Science of Learning from Data, Global Edition*, 4 ed: Pearson, 2016, pp. 337-349.
[220] A. Agresti, C. A. Franklin, and B. Klingenberg, " Constructing a Confidence Interval to Estimate a Population Mean," in *Statistics: The Art and Science of Learning From Data, Books a la Carte Edition* 4ed: Pearson, 2016.
[221] M. Fowler, *UML Distilled: A Brief Guide to the Standard Object Modeling Language*, 3 ed.: Addison-Wesley Professional, 2003.
[222] U. Wilensky and W. Rand, *An introduction to agent-based modeling: Modeling natural, social and engineered complex systems with NetLogo*. Cambridge: MIT Press, 2015.
[223] D. Stigberg, "An Introduction to the NetLogo Modeling Environment," in *Ecologist-Developed Spatially-Explicit Dynamic Landscape Models*, ed: Springer, Boston, MA, 2012, pp. 27-41.
[224] R. D. C. Team, "R: A Language and Environment for Statistical Computing," in *Ecological Modelling*, ed. Vienna, Austria: R Foundation for Statistical Computing, 2011.
[225] J. C. Thiele, "RMarriesNetLogo: Introduction to theRNetLogoPackage," *Journal of Statistical Software,* vol. 58, 2014.