Université de Montpellier
Faculté des Sciences

Master 1 Informatique - parcours AIGLE

HMIN226I Stage effectué dans le cadre du Cursus Master en Ingénierie

Développement d'algorithmes de recommandation tenant compte de l'expérience utilisateur

---

Rapport de stage

Université du Québec à Montréal
du 3 juin 2019 au 30 août 2019

Étudiante :
Amandine Paillard

Date de rédaction :
30 août 2019

Tuteur de stage :
Hafedh Mili

Soutenu le :
5 septembre 2019



Résumé

Ce rapport concerne mon stage au laboratoire de recherche sur les technologies du commerce électronique (LATECE) de la Faculté des Sciences de l'Université du Québec à Montréal (UQAM) au Canada. Ce dernier a été réalisé à l'occasion de ma première année de master informatique parcours Architectures et Ingénierie du logiciel et du Web en Cursus Master en Ingénierie à la Faculté des Sciences de l'Université de Montpellier. Pendant ces trois mois, l'occasion m'a été donnée de participer à l'élaboration d'algorithmes de recommandation. Ces algorithmes prennent place dans un projet de plus grande envergure : le développement d'un cadre conceptuel et logiciel pour le développement d'applications de gestion de l'expérience consommateur.

Tout au long de mon séjour différentes missions m'ont été assignées. Parmi ces dernières, sont à soulever l'utilisation d'apprentissage machine pour le développement d'algorithmes de recommandation, la réalisation d'expérimentation de ces algorithmes ou la collaboration à la rédaction d'articles scientifiques.

En plus d'étudier de nouvelles connaissances, ce stage m'a également permis de découvrir une nouvelle culture et d'en ressortir grandie. J'ai aussi pu aborder le monde de la recherche sous un nouvel angle. Je ressors de ces trois mois à Montréal enrichie de nouvelles compétences, qu'elles soient humaines ou techniques.

Remerciements

Avant d'entamer ce rapport, il y a quelques personnes qu'il me faut remercier. Toutes ont participé d'une manière ou d'une autre à ce que ce stage soit une bonne expérience.

Concernant la préparation en France, je voudrais remercier Mme Marianne Huchard pour m'avoir aussi bien aiguillée et accompagnée dans mes recherches. Merci également à Mme Anne-Élisabeth Baert qui m'a donné l'occasion de réaliser un stage à l'étranger et qui a dû signer ma convention à de multiples reprises. Merci à Mattéo, Nicolas et Théo pour avoir répondu à mes nombreuses questions sur le Québec.

Pour leur accueil à l'UQAM, leur bienveillance ainsi que leur encadrement, je suis reconnaissante envers M. Hafedh Mili ainsi qu'à Mme Imen Benzarti. Je pense également à l'administration de la Faculté des Sciences pour leur amabilité.

Enfin merci à Yoann et Thomas pour les bons moments qu'on a pu passer à Montréal, ainsi qu'à Félix et Thibault pour avoir à nouveau relu un de mes rapports.

Table des matières

Chapitre 1

Introduction

Ce stage s'inscrit dans le cadre de ma première année de master informatique parcours Architectures et Ingénierie du logiciel et du Web (AIGLE) en Cursus Master en Ingénierie à la Faculté des Sciences de l'Université de Montpellier. Le Cursus Master en Ingénierie préconise que les étudiants réalisent une cinquantaine de semaines de stage avant la fin de leur master et ceci non seulement dans le but de leur faire découvrir le monde du travail et de la recherche mais aussi de développer leur culture professionnelle. De plus, les étudiants sont vivement encouragés à réaliser un stage à l'étranger pour leur ouvrir de nouveaux horizons.

Devant alors réaliser un stage à l'étranger, la question de la destination était capitale. Les pays du nord m'ont toujours attirée et parmi eux, le Canada était plutôt bien placé. Ce pays, de part son respect pour la nature, l'importance accordée à l'accueil et son ouverture m'ont toujours fascinée. Mon stage a eu lieu au Québec, dans la métropole de Montréal. La Faculté des Sciences de l'Université du Québec à Montréal (UQAM) [1] m'a accueillie au sein d'un de ses laboratoires : le Laboratoire de recherche sur les technologies du commerce électronique (LATECE). Pendant ces trois mois de stage, mon travail s'est inscrit dans un projet de plus grande envergure : le développement d'un cadre conceptuel et logiciel pour le développement d'applications de gestion de l'expérience consommateur. En particulier, ce stage m'a donné l'occasion d'en apprendre plus sur les algorithmes de recommandation.

Parmi les diverses missions qui m'ont été confiées, sont à citer la familiarisation avec la littérature propre au domaine (gestion de l'expérience client du point de vue logiciel, apprentissage machine, algorithmes de recommandation), la programmation de certains algorithmes conçus par Mme Imen Benzarti, doctorante finissante sur ce projet. La participation à l'expérimentation (recensement et préparation de bases de données expérimentales) ainsi que l'analyse des résultats sont également à noter.

Ce rapport commence par présenter plus en détails l'organisme qui m'a accueillie ainsi que l'enjeu de mon travail en partie 2. Il décrit ensuite les outils et méthodologies qui ont pu être utilisés dans la réalisation des missions en partie 3. Le travail effectué est présenté en partie 4, juste avant une conclusion sur cette expérience en partie 5.

1. L'acronyme UQAM avec un accent grave ne s'utilise que pour la promotion de l'université. Il est d'usage de l'utiliser sans accent grave. Voir `http://servicecom.uqam.ca/normes-et-directives/logo-uqam-et-normes-graphiques.html` pour plus d'informations.

Chapitre 2

Contexte du stage

2.1 Organisme d'accueil : l'Université du Québec à Montréal

2.1.1 Présentation

L'UQAM a été créée le 9 avril 1969 et est située à Montréal. C'est une université de langue française qui bénéficie d'un rayonnement à l'international assez important. Elle fait partie du réseau Université du Québec, créé le 18 décembre 1968. Ce réseau vise à « accroître le niveau de formation de la population », « assurer le développement scientifique du Québec » et à « contribuer au développement de ses régions ». [1] De nombreuses réformes du système éducatif québécois sont à l'origine de ce réseau avec la volonté de permettre au plus grand nombre d'accéder aux études supérieures. Cette notion d'ouverture reste, cinquante ans après, une des valeurs piliers de l'UQAM.

2.1.2 Activité

Depuis sa création, l'UQAM a vu 269 107 étudiants sortir de ses bancs avec un diplôme. [2] Pour indication, la population d'étudiants au sein de l'université en 2018 était de 38 883 étudiants. Les trois quarts sont des étudiants de premier cycle.

L'UQAM propose pas moins de trois cent dix formations différentes au sein de quarante départements et écoles. Ces derniers sont regroupés en six facultés et une école : les facultés des arts, de communication, des sciences politiques et de droit, des sciences, des sciences de l'éducation et des sciences humaines ainsi que l'école des sciences de la gestion. L'école de gestion regroupe pas moins du tiers de la population étudiante diplômée de l'université entre 1969 et 2019.

Enfin, en avril 2019 la Fondation de l'UQAM [3] a atteint le palier de 200 millions de dollars réunis depuis 1969 pour ses activités d'enseignement, de recherche et de création.

2.1.3 Organisation interne

L'organigramme suivant reprend l'organisation de l'Université du Québec à Montréal. La suite de ce rapport s'intéresse à la Faculté des Sciences.

1. Voir `http://www.uquebec.ca/reseau/fr/reseau-de-luq/presentation-generale`.
2. Ce chiffre correspond au nombre de diplômés entre les années 1969 et 2018
3. Fondation créée en 1979 recueillant des dons pour assurer la qualité des activités de l'UQAM et assurer son accessibilité.

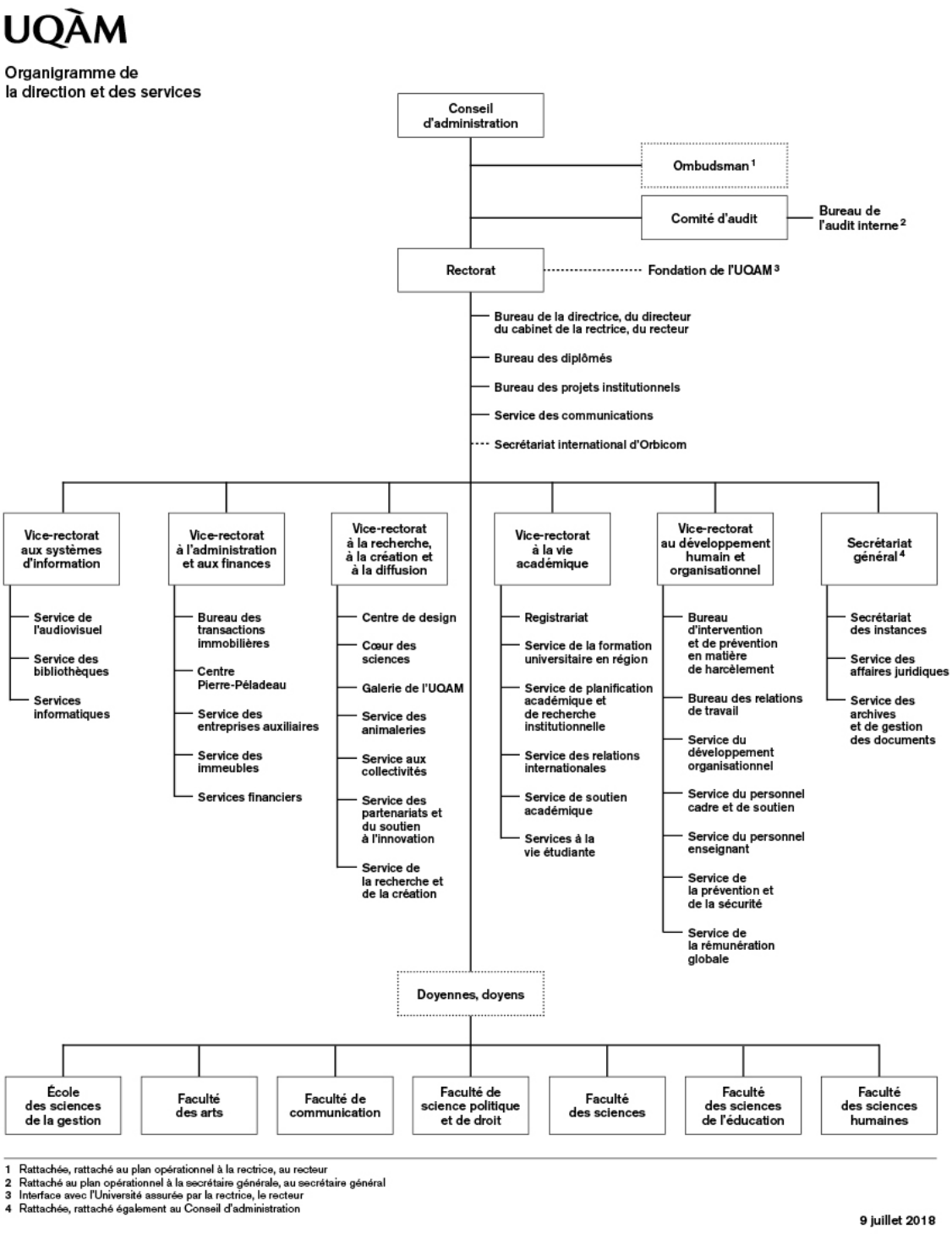

Figure 2.1 – Organigramme de l'UQAM. Source : `https://uqam.ca/services/organigramme/`

2.2 LATECE : Laboratoire de recherche sur les technologies du commerce électronique

2.2.1 Présentation

Le LATECE est un des treize Centres Institutionnels de Recherche (CIR) de la Faculté des Sciences de l'UQAM. Il a pour thématique principale de recherche les applications internet. Créé à l'origine en 2002 par des professeurs spécialisés en génie logiciel, intelligence artificielle, systèmes distribués et télécommunications, le laboratoire reste libre dans ses choix de problèmes à traiter, dans l'approche de solutions et dans la manière de présenter les résultats associés.

Les valeurs du LATECE sont :

— **Science et conscience** : l'éthique et autres questions sociales ne sont pas mises de côté.
— **Logiciel libre** : le laboratoire essaie autant que possible de développer un code ouvert à tous. [4]
— **Ancré dans son milieu, ouvert sur le monde** : ce CIR travaille avec des acteurs au profil divers et variés : qu'ils soit créateurs ou utilisateurs d'application, implantés au Québec ou à l'international.

2.2.2 Activité

La mission principale du laboratoire est de proposer des techniques, outils et langages pour chacune des étapes de la production d'une application internet (conception, développement, déploiement, exploitation). À cet objectif sont associés trois buts :

— *L'avancement des sciences et des technologies de l'Internet par la recherche subventionnée.*
— *La formation de personnel hautement qualifié dans les technologies de l'Internet.*
— *La diffusion des connaissances.*

Depuis 2010, plus de quatre cents articles ont été publiés. Le LATECE a accueilli plus de deux cents stagiaires (étudiants et stagiaires postdoctoraux) et a pu obtenir plus de six millions de dollars en fonds de recherche.

2.2.3 Organisation interne

À ce jour le LATECE regroupe vingt-cinq chercheurs (réguliers et associés confondus) venant de neuf institutions de recherches différentes (elles-mêmes situées dans quatre pays différents). Le laboratoire s'ouvre également sur d'autres groupes de recherches, qu'ils soient « UQAMiens », montréalais ou étrangers.

Parmi les recherches menées par le LATECE, plusieurs thématiques se distinguent :

— Modélisation, validation et mise en œuvre de processus d'affaires.
— Processus d'affaires intelligents – modèles, infrastructures et applications.
— Fouille de données – algorithmes, outils, applications et enjeux.
— Gestion de patrimoine et qualité logiciels.
— L'informatique nuagique – modèles, outils, langages et patrons de conception.
— Informatique et société.

4. Quand les contrats de recherche le permettent.

2.3 Enjeu du stage

Ce stage s'inscrit dans la thématique **Processus d'affaires intelligents – modèles, infrastructures et applications** précédemment citée. Cette thématique regroupe plusieurs profils d'applications comme la gestion d'inventaire en milieu hospitalier ou la gestion de l'expérience consommateur. Concernant le second point, une équipe de chercheurs travaille sur une nouvelle architecture logicielle permettant la gestion de l'expérience client tenant compte du contexte, on parle de *CA-CEM* pour « *context aware - customer experience management* » en anglais.

La gestion de l'expérience client est une branche du marketing visant, pour une entreprise, à proposer aux clients les produits et services les plus opportuns et de la façon la plus adaptée tout en tenant compte de leurs goûts ou préférences personnelles. L'expérience client intervient à tout moment de la relation entre le client et l'entreprise : que ce soit la prise de contact ou la phase de finalisation (achat, souscription à un service, etc.).
Parler d'*expérience client tenant compte du contexte* rajoute une notion supplémentaire qui peut prendre plusieurs formes différentes. Le contexte peut tout aussi bien être l'étape du processus d'achat dans laquelle le client se trouve, s'il a déjà fixé son choix sur un produit particulier ou s'il est à l'écoute de suggestions par exemple. Le contexte peut aussi avoir une notion temporelle ; le système est capable de proposer des articles différents à un utilisateur selon son historique. Par exemple si un client achète des couches culottes de taille 2 depuis déjà deux mois, des couches de taille supérieure vont lui être proposées car son bébé grandit.

L'équipe du LATECE a pu proposer une méthodologie ainsi qu'une architecture applicative permettant de répondre aux attentes dans ce domaine, en particulier la nécessité d'ouverture. En effet, l'architecture doit pouvoir être appliquée à n'importe quel commerce, indépendamment de son activité, de la diversité des produits et services qu'elle propose et du profil de ses clients.

Une telle application doit mettre à disposition diverses fonctionnalités intervenant à des étapes distinctes du processus d'achat de l'utilisateur [5]. Ce stage se concentre sur l'élaboration et l'analyse d'algorithmes de recommandation répondant à ce contexte. En plus du développement de ces algorithmes, il est nécessaire de préparer diverses bases de données expérimentales afin d'évaluer leur efficacité. L'analyse des résultats obtenus permet d'améliorer les algorithmes.

5. Voir annexe A.

Chapitre 3

Outils et méthodologie

3.1 État de l'art

Une analyse de l'existant a été nécessaire avant d'envisager l'expérimentation des algorithmes de recommandation. Cette étude fut réalisée par la lecture d'articles scientifiques, de livres ou de ressources en ligne.

3.1.1 Qu'est-ce qu'un algorithme de recommandation ?

Un algorithme de recommandation sert à filtrer une grande quantité de données et ne proposer à un utilisateur uniquement du contenu susceptible de l'intéresser. Pour fonctionner, un système de recommandation va se baser sur diverses similarités : similarité entre les utilisateurs du système, similarité de produits (pour un utilisateur donné) et similarité de contexte (certains achats peuvent être les mêmes selon la période de l'année par exemple).

D'après Katsov, 2018, un système de recommandation a plusieurs objectifs :

— **Pertinence** : la qualité des suggestions faites, si l'utilisateur apprécie les produits suggérés.
— **Nouveauté** : il ne s'agit pas de montrer à l'utilisateur tous les produits populaires qu'il peut déjà connaître.
— **Sérendipité** : proposer des produits peu connus et fortuits.
— **Diversité** : ne pas proposer des produits identiques.

Toujours d'après Katsov, 2018, il existe de multiples types d'algorithmes de recommandation mais les principaux sont le **filtrage collaboratif** et le **filtrage à base de contenu**.

Le **filtrage collaboratif** va analyser les évaluations des utilisateurs ayant évalués le même produit pour proposer les meilleurs articles en lien avec ce dernier. De fait, l'algorithme a besoin de l'historique des utilisateurs et nécessite que beaucoup de produits aient été évalués pour obtenir de bons résultats. Par ailleurs, cela mène à ce que seule une partie du catalogue soit mise en lumière.

Le **filtrage à base de contenu** tient compte des préférences des utilisateurs. Il va regarder quels produits le consommateur a évalué positivement et lui proposer des produits similaires ; il s'agit du type d'algorithme de recommandation le mieux adapté dans le cadre de ce projet. Ce type d'algorithme permet de proposer des produits personnalisés mais nécessite un travail de traitement des produits pour les classer en catégories. De plus, une question se pose quant aux suggestions pour un nouvel utilisateur.

Parmi les autres catégories d'algorithmes de recommandation Katsov, 2018 compte :

— **Recommandation hybride** : combine le filtrage collaboratif et celui à base de contenu.
— **Recommandation contextuelle** : se concentre sur d'autres aspects tels que le lieu ou la période de l'année.
— **Recommandation non personnalisée** : *les dernières nouveautés* par exemple.
— **Recommandation multi-objectif** : peut regrouper plusieurs catégories d'algorithme.

3.1.2 Apprentissage automatique supervisé et régression logistique

L'apprentissage automatique permet de répondre à une question en étudiant des données de manière « autonome ». Ce domaine est très vaste, autant dans le type des données utilisées que dans le but de l'apprentissage. Dans le contexte de ce stage, en analysant des avis (notes) de films d'un utilisateur, le but de l'apprentissage est de prédire quels sont les films appréciés (et donc recommandables) à l'utilisateur.
On distingue deux étapes dans l'apprentissage automatique. La première permet de définir un *modèle* ; on parle d'entraînement. La seconde correspond à l'utilisation du modèle précédemment déduit.
Le modèle établi doit être évalué, en analysant si les estimations réalisées sont correctes ; il s'agit de la validation.

Il existe deux types d'apprentissage : supervisé et non supervisé. C'est le premier type qui est utilisé dans le cadre de ce stage : les notes des utilisateurs concernant les films sont connues.
On distingue la classification de la régression. Lorsque les étiquettes des données est une classe, une valeur discrète, il s'agit de classification. En revanche pour la régression, les étiquettes sont des valeurs continues. Dans ce contexte, on utilise la régression logistique qui est un mélange de classification et de régression : on cherche à prédire la note qu'un utilisateur va associer à un film, c'est cette note qui détermine si l'utilisateur aime ou non ce film.

Dans le cadre de la recommandation, diverses approches existent et sont détaillées dans la suite de ce rapport. Celle étudiée par Mme Benzarti se base sur les variables véristiques introduites par Yager.

3.1.3 Logique floue et variables véristiques

La logique floue apporte un degré de confiance (une valeur comprise entre 0 et 1) à une assertion. Elle diffère de l'aspect binaire de la logique classique où un fait est soit vrai, soit faux (on dit qu'elle est multivaluée). La logique floue représente plus aisément les imprécisions et les incertitudes. Par exemple, affirmer en logique classique le fait *le chat est dans la cuisine* signifie que le chat est bel est bien dans la cuisine. En revanche, avec la logique floue une telle affirmation s'associe à un degré de confiance : si ce fait est associé à un degré de 0.8 par exemple, cela signifie que le chat est *probablement* dans la cuisine, mais on ne peut pas l'affirmer avec certitude. [1] On distingue ainsi différents niveaux de confiance.
La logique floue a été introduite par Lotfi Zadeh en 1965, cette logique se base sur sa théorie des sous-ensembles flous. En 1978, Zadeh introduit l'idée de variables véristiques, que Yager va étendre [2].

1. On dit que *le chat est dans la cuisine avec un degré de confiance de 0.8*
2. Voir Yager, 1987a, Yager, 1987b, Yager, 1988, Yager, 2000 et Yager, 2003.

Par exemple, `étudieEn` est une variable floue telle que celles initialement proposées par Zadeh. Elle représente le master informatique suivie par un étudiant de la Faculté des Sciences de Montpellier. Si l'on considère l'ensemble des masters $\{AIGLE, DECOL, IPS, IMAGINA, MIT\}$, `étudieEn` prend pour valeur un des éléments de cet ensemble. Un étudiant suit un seul de ces masters et un certain degré de confiance est associé à chacune des valeurs de l'ensemble, représentant la **probabilité** qu'à l'étudiant de suivre ce master. On appelle ce type de variable floue une **variable possibiliste**. Elle peut être vue comme une disjonction car un seul des éléments de l'ensemble est vrai. Les variables possibilistes sont également appelées variables disjonctives.

Les variables introduites par Yager sont légèrement différentes. La variable `préfèreCouleurs` représente les couleurs préférées d'une personne donnée et peut avoir pour valeur l'ensemble `C`, tel que `C = {Bleu, Rouge, Vert}`. Ici, en associant un degré de confiance aux valeurs de l'ensemble, on ne représente pas la probabilité qu'elles ont de faire partie des couleurs préférées d'une personne mais on évalue leur degré de préférence. Ainsi, toutes ces couleurs sont appréciées mais certaines plus que d'autres. Ces variables sont appelées **variables véristiques** (ou variables conjonctives) et s'écrivent de la forme `préfèreCouleurs isv C`. À l'inverse des **variables possibilistes**, tous les éléments de l'ensemble sont vrais en même temps. Ces variables peuvent être vues comme une conjonction et sont aussi appelées *variables conjonctives*.

Dans le but de pouvoir appliquer les systèmes inférences de Zadeh, Yager a adapté sa théorie des variables véristiques (variables conjonctives) pour revenir à des variables possibilistes (variables disjonctives). Pour ce faire, il transforme une variable conjonctive `V` en une variable disjonctive `V*` de telle sorte que `V*` est la disjonction de l'ensemble des parties de `D`, avec `D` le domaine sur lequel est défini `V`. [3]
En revenant à l'exemple précédent, la variable véristique `C` devient possibiliste (`C*`) avec `C*` l'ensemble des parties de l'ensemble des couleurs existantes [4]. Pour rappel, seul un des sous-ensembles composant `C*` est certain.

Yager a trouvé quatre différentes sémantiques pour les variables véristiques. Selon leur sémantique, ces variables se traduisent en variables possibilistes de façons différentes. Pour une variable véristique `V` et une valeur `A`, on a :

- **Fait positif ouvert** : `V isv A` peut être compris familièrement par « *J'aime les éléments de `A` mais peut être d'autres choses également* ». Dans ce cas la variable possibiliste peut être traduite par la disjonction des sous-ensembles de `A`.
- **Fait positif fermé** : `V isv(c) A` [5] peut être compris familièrement par « *Je n'aime que les éléments de `A`* ». La variable possibiliste ne peut qu'être traduite par `A`.
- **Fait négatif ouvert** : `V isv(n) A` peut être compris familièrement par « *Je n'aime pas les éléments de `A` et peut être d'autres choses...* ». La variable possibiliste peut être traduite par la disjonction des sous-ensembles comprenant le complément de `A`.
- **Fait négatif fermé** : `V isv(n,c) A` [6] peut être compris familièrement par « *Je n'aime pas les éléments de `A` et strictement eux* » ou « *J'aime tout sauf les éléments de `A`* ». La variable possibiliste peut être traduite par le complément de `A`.

3. Dans l'exemple précédent, `D` serait « *toutes les couleurs existantes* ».
4. Ainsi, `{Bleu, Rouge, Vert}` est inclus dans l'ensemble.
5. Il est important de souligner que le `c` de `isv(c)` signifie *closed*, il n'y a pas de lien avec la variable représentant un des sous ensembles de `C*`
6. Même remarque que précédemment pour `isv(n,c)`

Par ailleurs, Yager a défini quelques mesures supplémentaires

— `ver(x)` représente la confiance dans le fait que `x` (un élément du domaine) fasse partie d'une variable véristique `X` ; c'est-à-dire que `x` fasse partie soit solution de `X`. [7]
— `rebuff(x)` représente la confiance dans le fait que `x` ne fasse pas partie d'une variable véristique `X`. [8]

De `rebuff(x)`, on déduit `poss(x)` $= 1 -$ `rebuff(x)` qui représente l'absence de preuve que `x` n'est pas une solution.

Pour finir, Yager précise que le calcul des précédentes mesures change selon la sémantique de la variable véristiques. En suivant les variables précédemment étudiées, on obtient les mesures ci-contre.

Type de variable véristique	**ver(x)**	**rebuff(x)**	**poss(x)**
`V isv A`	`A(x)`	0	1
`V isv(c) A`	`A(x)`	1−`A(x)`	`A(x)`
`V isv(n) A`	0	`A(x)`	1−`A(x)`
`V isv(n,c) A`	1−`A(x)`	`A(x)`	1−`A(x)`

3.2 Outils et ressources utilisés

Les algorithmes développés tout au long de ce stage sont réalisés en Scala et ont été implémentés avec l'aide du logiciel Eclipse. Scala est un langage de programmation dit « multiparadigme » : il est à la fois fonctionnel et orienté objet (voir Scala center, 2019). Créé par l'École polytechnique fédérale de Lausanne il se compile en *bytecode* et est exécutable sous une JVM.

Lors de ces trois mois de stage, les outils suivants ont été utilisés :

Apache Spark : *cadriciel en code source ouvert* de calcul distribué (voir Apache Software Foundation, 2019b). Spark SQL a été utilisé ; il s'agit de la branche de Spark permettant de traiter les données avec des requêtes SQL, semblables à celles issues d'une base de données relationnelle.

Eclipse : environnement de développement utilisé principalement pour Java, mais supportant d'autres langages avec les bons *plugins*. Dans le cadre de ce stage, Eclipse a été utilisé pour développer et exécuter des projets Scala.

Git : logiciel de gestion de version libre et décentralisé créé en 2005 par Linus Torvald.

GitHub : hébergeur en ligne comprenant un gestionnaire de version basé sur Git : `https://github.com/`.

Maven : outil utilisé ici pour gérer le cycle de vie d'un projet (voir Apache Software Foundation, 2019a).

Parmi les technologies précédemment citées, Scala et Spark sont des découvertes. L'occasion de travailler avec Maven s'était déjà présentée par le passé mais sans qu'il s'agisse d'une utilisation poussée.

7. La formule est `ver(x)` $= min_{X \in \mathtt{X}*} \mu_X(x)$ avec $\mu_X(x)$ la fonction d'appartenance de X.
8. Une des trois formules possibles est `rebuff(x)` $= 1 - max_{X \in \mathtt{X}*} \mu_X(c)$ avec $\mu_X(x)$.

3.3 Méthodologie

Le début du travail concernant la mission du stage a été précédé d'une phase de mise en place nécessaire. Cette dernière ayant duré deux semaines, elle a permis la découverte de l'environnement de travail au LATECE, la littérature inhérente au domaine ainsi que les algorithmes relatifs à la mission du stage. Pour rappel, les différentes missions constituant ce stage sont les suivantes :

- **Programmation d'algorithmes de recommandation**.
- **Participation à l'expérimentation par la recherche et la préparation de bases de données expérimentales** : Il s'agit de rechercher et de mettre en forme des *datasets* disponibles sur Internet afin de pouvoir étudier le comportement des algorithmes avec des données différentes.
- **Analyse de résultats** : pour chacun des algorithmes développés, plusieurs tests sont à effectuer pour pouvoir comparer leur pertinence. Ces tests sont réalisés sur le même jeu de données, avec ces données réparties aléatoirement en deux groupes : entraînement et validation. Ainsi en faisant plusieurs tests similaires, on s'assure que les résultats obtenus par les algorithmes soient bien représentatifs de leur qualité et non pas aléatoires. Par ailleurs certains algorithmes possèdent des paramètres inhérents nécessitant des tests. De fait, ces derniers ont été réalisés pour un algorithme et un jeu de données, en modifiant un seul paramètre à la fois.

Chacune de ses missions ont été réalisées sous le regard de la doctorante finissante de M. Hafedh Mili : Mme Imen Benzarti. Lors des trois mois constituant le stage, Mme Benzarti a travaillé sur la publication d'un article dans deux conférences ainsi que dans une revue papier. De ce fait, la nature conjointe du travail réalisé explique qu'une planification poussée n'a pas été choisie pour ce stage, l'organisation du travail dépendant fortement des résultats obtenus par les algorithmes.

En plus de ces tâches, une partie du stage a été consacrée à leur automatisation. Cette automatisation a pris plusieurs formes : création d'un *pipeline* lançant tous les algorithmes à tester, génération des données d'entrée aux algorithmes, enregistrement automatique des résultats, nettoyage des fichiers générés.

Chapitre 4

Travail réalisé

4.1 Étude et programmation d'algorithmes de recommandation

4.1.1 Objectif

Le but du travail réalisé au LATECE est de fournir un algorithme de recommandation performant et suffisamment général pour pouvoir être appliqué à une multitude de produits et utilisateurs différents. Il s'inscrit dans le cadre des travaux de Mili et al., 2016. La différence sémantique dans la description des produits est apparue comme une complication. En effet, l'interprétation de la manière dont est représentée un produit peut différer : *le produit possède toutes ces caractéristiques* ou *l'utilisateur aime une caractéristique parmi celles-ci.*
Pour répondre à ce problème, Mme Imen Benzarti s'est intéressée aux variables véristiques introduites par Yager et a imaginé un algorithme de recommandation tirant profit de ces variables : le `VeristicContentBasedRecommendation`. Dans le but de vérifier sa théorie, il a été nécessaire de comparer cet algorithme avec d'autres, inspirés de la littérature.

Ces algorithmes ont été développés dans un contexte inspiré d'une plateforme de visionnage de films. Leur but est d'analyser les avis d'utilisateurs sur des films pour leur proposer des films pouvant les intéresser. Les avis ont la forme de note. Un film possède des catégories telles qu'*action*, *comédie*, ou *horreur* et un utilisateur a des préférences. Ces préférences correspondent aux mêmes catégories que celles décrivant les films. Cependant, si un film possède unilatéralement un ensemble de catégories, un utilisateur peut en préférer une aux dépends des autres. Enfin, l'année de production d'un film est connue, et selon la note qu'a attribuée un utilisateur à un film, il est possible de déduire si ce dernier a des périodes qu'il préfère. C'est dans cet objectif que les années de production de films ont aussi été floutées.

4.1.2 `VeristicContentBasedRecommendation` (ou `VeristicCB`)

Le `VeristicContentBasedRecommendation` (`VeristicCB`) ou **Approche à base de variables véristiques** est constitué de plusieurs étapes. D'abord il construit le profil de l'utilisateur en apprenant des notes que ce dernier a attribué et en attribuant un score à chacun des genres disponibles ; ainsi le profil de l'utilisateur est l'ensemble des genres qu'il a évalué avec le score associé. Ensuite il choisit les films à recommander selon s'ils correspondent au profil établi. Ici les variables véristiques sont les genres d'un film.

Construction du profil utilisateur : transformation des notes en variables véristiques

Il est important de rappeler que les recommandations sont faites pour conseiller au mieux **un** utilisateur. C'est pour ça que `VeristicCB` cherche à construire un profil pour chaque utilisateur. Pour construire ce profil, les `K` meilleures et pires notes sont prises en compte.

La première étape de traitement consiste à standardiser [1] et normaliser les notes données par les utilisateurs sous la forme d'échelle de Likert [2]. Cela permet deux choses. La première est d'avoir des notes comprises entre $[-1, 1]$ avec une valeur positive représentant un film apprécié tandis qu'une valeur négative représente un film non apprécié. La seconde est d'atténuer les différences de notations des utilisateurs. Comme les notes données sont suggestives, la même note ne représentera pas la même chose pour deux utilisateurs différents : une note moyenne peut signifier un film *apprécié* pour un utilisateur sévère, alors qu'elle représentera un film *pas bon* pour un utilisateur généreux.

Une fois que chacune des notes est comprise entre $[-1, 1]$, cette note est partagée de manière égale entre tous les genres (action, romance,...) le constituant. À ce stade, il est possible d'identifier quels films ont été appréciés. On observe alors une disjonction de cas :

— La note est positive : le film et ses genres seront considérés comme une variable `véristique positive ouverte`.
— La note est négative : le film et ses genres seront considérés comme une variable `véristique négative ouverte`.

On peut maintenant regrouper les films aimés ou non entre eux dans deux variables comme présenté en 3. Il reste à construire le profil utilisateur en les étudiant.

Construction du profil utilisateur : calcul des scores des genres

À cette étape, l'algorithme connaît deux variables représentant les films (mais surtout leurs genres) que l'utilisateur a aimé ou non. Il va essayer de déduire quels sont les genres préférés et moins appréciés de l'utilisateur afin de pouvoir lui conseiller au mieux des films.

Pour ce faire `VeristicCB` utilise les mesures `ver(genre)` et `poss(genre)` introduites par Yager et présentées précédemment. Pour calculer le score d'un genre, les règles suivantes sont utilisées :

— $(ver(genre) = fort \wedge poss(genre) = fort) \longrightarrow score(genre) = fort$
Ce cas peut être compris naïvement par « *Ce genre apparaît systématiquement quand la note du film est élevée* » ou « *Ce genre n'apparaît pas dans les films moins appréciés* ».
— $(ver(genre) = fort \wedge poss(genre) = faible) \longrightarrow score(genre) = moyen$
Ce cas peut être compris naïvement par « *Ce genre a toujours été noté de manière extrême, que ce soit négativement ou positivement* ».
— $(ver(genre) = faible \wedge poss(genre) = fort) \longrightarrow score(genre) = faible$
Ce cas peut être compris naïvement par « *Ce genre n'apparaît dans aucun des films bien noté* » ou « *Ce genre n'a jamais été noté* ».
— $(ver(genre) = faible \wedge poss(genre) = faible) \longrightarrow score(genre) = tres\_faible$
Ce cas peut être compris naïvement par « *Ce genre n'a jamais été noté positivement mais a été noté négativement ; il n'est pas aimé* ».

1. En statistiques, standardiser revient à calculer la moyenne d'un ensemble de données et de soustraire le résultat à chacune des entrées.
2. Développée par Francis Likert, elle va de 1 à 5.

Ces règles sont combinées dans un système d'inférence flou [3] et le score final est obtenu en utilisant des fonctions d'appartenance (semi-)triangulaires ainsi qu'une méthode de défuzzification [4] de centre de gravité.

Recommandation de films correspondant au profil

Pour cette dernière étape, les préférences de l'utilisateur sont connues. Il y a deux façons de satisfaire ses exigences et les deux versions ont été testées dans les expérimentations.

— Groupement mono-valué : un film est recommandé à partir du moment où il possède un genre apprécié par l'utilisateur.
— Groupement multi-valué : un film est recommandé s'il est similaire au profil de l'utilisateur, c'est à dire si les genres le décrivant sont appréciés par l'utilisateur. Pour calculer la similarité, le calcul se base sur la différence (distance) entre le profil et les films ; avec une distance faible, ce film aura une bonne similarité et donc de bonnes chances d'être recommandé.

4.1.3 Algorithmes inspirés de la littérature

Les algorithmes choisis pour comparer le `VeristicContentBasedRecommendation` proviennent de la littérature ou sont des standards en apprentissage automatique.

— **Approche naïve bayésienne** : cette approche utilise un classifieur naïf bayésien pour prédire le profil de l'utilisateur en étudiant ses notes. Par profil, il est entendu les genre(s) et époque(s) de films qu'il préfère.
— **Recouvrement à logique floue** : ici le modèle appris (c'est-à-dire le profil utilisateur) est représentée par un arbre de décisions flou. Cette approche provient de Mao, Lu, G. Zhang et J. Zhang, 2015.
— **Approche à base de similarité floue** : ce modèle calcule le score d'un film en se basant sur les précédentes notes de l'utilisateur. La note obtenue est donc la moyenne des précédentes notes mais en les pondérant selon le degré de ressemblance du film à noter avec ceux de l'historique. Diverses méthodes de calcul de la similarité et d'agrégation ont été testées et celles gardées sont la similarité cosinus [5] et la somme pondérée.
Deux versions de cet algorithme ont été testées. Cette approche vient de Zenebe et Norcio, 2009.
— **Méthode des K plus proches voisins** : cette approche n'est pas basée sur la construction du profil utilisateur mais sur la prédiction de la note qu'il donnerait aux films à recommander. Comme pour l'approche précédente, la méthode de calcul de similarité choisie est la similarité cosinus, la méthode d'agrégation est la somme pondérée.

4.2 Expérimentation et analyse de résultats

Le but de cette phase d'expérimentation est d'étudier le comportement de l'algorithme `VeristicCB` avec les autres rapportés dans la littérature. Les algorithmes sont alors comparés sur les mêmes mesures, sur un `dataset` donné.

3. Pour notamment déterminer $fort$, $moyen$, $faible$ et $tres\_faible$
4. La défuzzification est le processus final d'un système flou, permettant de passer d'une variable flou à un résultat numérique
5. La similarité cosinus calcule la distance de deux vecteurs en prenant le cosinus de leur angle

4.2.1 *Datasets*

Les *datasets* suivants ont été utilisés pour l'expérimentation :

— `MovieLens 100k` : données extraites du site `https://movielens.org/`. On retrouve des notes d'utilisateurs (représentés par un identifiant) sur des films. Les informations disponibles sur ces derniers sont leur titre, année de sortie, genres et *tags* (des mots-clefs décrivant le film et générés individuellement pour chaque utilisateur). Ce *dataset* comprend 100 000 notes de films, de 1 000 utilisateurs sur 1 700 films. Le *dataset* date d'avril 1998.
— `MovieLens 1M` : extrait du site `https://movielens.org/`, il comprend 1 000 000 avis de 6 000 utilisateurs sur 4 000 films et date de février 2003.
— `MovieLens 10M` : extrait du site `https://movielens.org/`, il comprend 10 000 000 avis de 72 000 utilisateurs sur 10 000 films et date de janvier 2009.

Les *dataset*s de MovieLens sont disponibles ici : `https://grouplens.org/datasets/movielens/`.

Pour chacun des *datasets*, on crée deux sous-*datasets*. Le premier comprend 80 % du fichier de notes originel et est utilisé pour la phase d'entraînement. Les 20 % restants sont préservés pour la validation. La répartition de ces deux sous-*datasets* est faite aléatoirement, suivant ces mesures et pour chaque utilisateur ; en effet, pour chacun des utilisateurs on doit s'assurer d'avoir assez de données pour apprendre ses préférence des notes sur les produits. Ainsi, on va prévoir les notes qu'un utilisateur attribuerait à un produit qu'il ne connaît pas, c'est à dire un produit figurant dans le *dataset* de validation.

4.2.2 Mesures

Dans ce contexte, il faut considérer le vocabulaire suivant, pour un utilisateur `X` et un film `Y` :

— **Vrai positif** : le modèle prédit que « `X` **aime** `Y` » (note positive), et la note qu'a attribué `X` à `Y` dans le *dataset* de validation est **positive**.
— **Vrai négatif** : le modèle prédit que « `X` **n'aime pas** `Y` » (note négative), et la note qu'a attribué `X` à `Y` dans le *dataset* de validation est **négative**.
— **Faux positif** : le modèle prédit que « `X` **aime** `Y` » (note positive), et la note qu'a attribué `X` à `Y` dans le *dataset* de validation est **négative**. Cette mesure correspond à l'erreur.
— **Faux négatif** : le modèle prédit que « `X` **n'aime pas** `Y` » (note négative), et la note qu'a attribué `X` à `Y` dans le *dataset* de validation est **positive**.

On rappelle que les algorithmes vont produire une liste de `N` produits recommandés. Ils sont comparés sur les mesures suivantes :

— `Précision@N` : correspond à la pertinence du résultat : parmi les `N` produits proposés, combien sont corrects ? La formule est $\frac{vrai\_positif}{vrai\_positif+vrai\_negatif}$.
— `Rappel@N` : correspond à la capacité du modèle à trouver toutes les réponses correctes : combien de prédictions correctes sur celles existantes ? Cela se calcule $\frac{vrai\_positif}{vrai\_positif+faux\_negatif}$.
— `F1-score@N` : moyenne harmonique de la précision et du rappel. La formule est $2\times\frac{precision\times rappel}{precision+rappel}$.
— `nDCG@N` (*Normalized Discounted Cumulative Gain*) : représente la pertinence du classement de la recommandation. Vérifie que le premier (resp. le second, ...) produit conseillé, même s'il est aimé par l'utilisateur, est celui que ce dernier va préférer en premier (resp. en second,

...). Cette mesure est très importante pour les algorithmes de recommandation et se calcule $\frac{DCG}{IDCG}$. Avec `DCG` le *Discounted Cumulative Gain* et `IDCG` le *Ideal Discounted Cumulative Gain*[6].

— `MAE` (*Mean Absolute Error*) : l'erreur absolue moyenne est la moyenne des valeurs absolues des écarts entre les prévisions du modèle et les données de validation.

— `RMSE` (*Rooted Mean Score Error*) : l'erreur quadratique moyenne est la racine carrée de la moyenne des carrés des écarts entre les données prédites et observées.

Dans les expérimentations, la `Précision`, le `Rappel` et le `F1-score` ont été calculés avec `N` valant 5 et 10. Pour le `nDCG`, `N` a pris les valeurs 5, 10 et 15.

4.2.3 Résultats

Les expérimentations ont permis de démontrer que globalements l'algorithme développé par Mme Imen BENZARTI obtient de meilleur résultat que ceux de la littérature (seul le `nDCG` est légèrement plus faible qu'un des algorithmes). Concernant les améliorations de l'algorithme `VeristicCB` étudiées, diverses expérimentations ont été réalisées.
Concernant le paramètre `K` qui représente le nombre d'éléments recommandés, la performance de `VeristicCB` se stabilise à partir de la valeur 10.
De plus, il s'avère que le regroupement multi-valué de la propriété `genre` améliore les résultats : un utilisateur se verra alors proposer en premier des films dont les genres correspondent à tous ceux qu'il apprécie, au lieu de lui proposer un film ayant au moins un genre apprécié.
Enfin, prendre en compte les différentes sémantiques de variables véristiques, à savoir positive ouverte (`V isv() A`) et négatives ouvertes (`V isv(n) A`, améliore aussi les résultats.

4.3 Automatisation

Chacun des algorithmes a été développé indépendamment, dans un projet Java différent. De plus, chacun utilise des fichiers stockant les *datasets* d'entraînement et de validation et génère des fichiers intermédiaires et finaux. Il est à noter qu'à chaque expérimentation, les fichiers de *datasets* changent. Enfin, les mesures étudiées s'affichent sur la console. Ainsi, l'expérimentation de tous les algorithmes (ou même d'une partie d'entre eux) nécessite une période de gestion des fichiers : de ce fait est née l'idée d'automatiser une partie de ce travail.

Les fichiers intermédiaires générés ont pu être supprimés automatiquement par l'inclusion dans le programme d'une commande de script `bash`. Pour le changement manuel des fichiers d'entrée des programmes, en début de stage, un programme `bash` personnel a été créé. Cette méthode n'a plus été nécessaire quand tous les algorithmes ont pu être regroupés dans un projet général qui s'occupe à la fois de générer les fichiers d'entrée et d'exécuter les algorithmes voulus. Ce projet permet à la personne voulant obtenir les résultats des algorithmes désirés sur un *dataset* choisi en ne réalisant qu'une seule exécution et offre une architecture commune à tous les projets. Enfin, l'affichage console des résultats a été changé pour enregistrer toutes les mesures calculées ainsi que les divers paramètres de l'exécution (type de *dataset*, algorithmes choisis et valeur de `K` pour le `VeristicCB`) dans un seul fichier commun.

Une partie du stage a également été consacrée à la recherche de l'optimisation des différents algorithmes. Pour certains, sur les plus grands *datasets* une erreur de mémoire était levée pendant

6. Pour les formules de `DCG` et `IDCG`, le lecteur est invité à se reporter à l'adresse suivante `https://en.wikipedia.org/wiki/Discounted_cumulative_gain`.

l'exécution. Cette dernière a pu être corrigée en ajoutant et modifiant des paramètres propres à `Spark`[7]. En revanche, les recherches entamées pour optimiser le temps d'utilisation du ramasse-miettes pendant l'exécution, qui peut représenter jusqu'à 20% du temps d'exécution selon les algorithmes, n'ont pas abouties par manque de temps.

7. Notamment le niveau de parallélisme des exécuteurs.

Chapitre 5

Bilans et perspectives

5.1 Présentation des résultats

Les missions de ce stage ont été menées à bien. Le travail réalisé a permis l'obtention de résultats comparant la pertinence des algorithmes développés. Ces résultats vont être publiés dans un article scientifique.

Au total, six algorithmes de classification différents ont pu être testés, dont l'algorithme développé par Mme Imen Benzarti. Concernant ce dernier, différentes versions et améliorations ont pu être expérimentées, totalisant sept algorithmes. De ce fait, l'automatisation réalisée pour tester les algorithmes rend l'expérimentation plus simple et demeure un outil réutilisable.
Les algorithmes ont été testés sur trois *datasets* différents issus de *MovieLens* avec 100 000, 1 000 000 et 10 000 000 d'instances. Par ailleurs, deux autres *datasets* ont été étudiés sans être utilisés à ce jour ; il s'agit de *Yelp* et de *KKBox*.

En plus de permettre la découverte d'un autre point de vue de la recherche, cette fois plus proche du processus de publication, ce stage a été l'occasion de découvrir la littérature propre aux domaines des algorithmes de recommandations, de l'apprentissage machine et de la gestion de l'expérience client du point de vue logiciel.

5.2 Difficultés rencontrées

Ce stage n'a pas été sans difficulté. La principale étant le temps de traitement qu'ont pu prendre certains algorithmes. Les données à traiter étaient parfois conséquentes (10 000 000 entrées au maximum) et un des algorithmes n'a pas pu être testé en dépit de tous les efforts fournis. En effet, une étape de l'exécution des K plus proches voisins a pris plus de 70 heures et a généré plus 250 Go de fichiers temporaires. Exécutant les algorithmes sur une machine personnelle, cette dernière a vite montré ses limites. Une solution envisagée pour résoudre cet aspect aurait été le calcul distribué, cependant cette expérience est arrivée relativement tardivement dans le stage et cette piste n'a pas pu être explorée.

Une autre difficulté rencontrée, et ce malgré le fait de travailler avec Java depuis quatre années désormais, a été le paramétrage d'Eclipse et de ses différents modules. En effet plusieurs réinstallations d'Eclipse, de Java, des erreurs de *Java Native Interface* et une phase de paramétrage ont été nécessaires pour pouvoir faire tourner les algorithmes. Sans parler des quelques hésitations pour configurer convenablement le *CLASSPATH* et les configurations d'exécutions. Néanmoins, ces complications ont pu être surmontées.

5.3 Conclusion et perspectives

Ce stage m'a permis d'apprendre de nouveaux langages, outils et connaissances. J'ai pu suivre le processus de publication d'articles scientifiques de l'intérieur. J'ai également eu l'occasion d'assister à des conférences organisées par le LATECE sur l'apprentissage automatique et le monde de la recherche.

Lors de ces trois mois à l'étranger, j'ai aussi pu constater des différences culturelles. J'ai pu découvrir une partie du Québec en visitant Montréal et Québec, à travers des découvertes culturelles, géographiques, musicales ou gustatives. Cependant, bien que Montréal soit une ville plaisante et très animée de part ses nombreux festivals en été (auxquels j'ai eu la chance d'assister), mon stage m'a permis de réaliser que je ne souhaite pas m'expatrier. Avant cette expérience, l'idée de travailler en Amérique du Nord ne m'était pas totalement fermée, mais au cours de ces trois mois j'ai pu me rendre compte de certaines différences avec la France auxquelles je suis profondément attachée.

Je ressors de ces trois mois à Montréal enrichie de nouvelles compétences, qu'elles soient humaines ou techniques.

Glossaire

Cadriciel (ou *Framework*) : groupement d'outils suivant une architecture spécifique et visant à fournir des fonctionnalités facilitant le développement d'applications.

CLASSPATH : paramètre permettant à la machine virtuelle Java de savoir où se trouvent les différents fichiers (classes, packages, ...) dont elle a besoin pour son exécution.

***Dataset* (ou *data set*)** : groupement de données.

JVM (*Java virtual machine*) : machine virtuelle permettant d'exécuter du *bytecode* (code Java compilé).

Pipeline : enchaînement de programmes de manière à ce qui est produit par un programme soit utilisé en entrée d'un autre.

Ramasse-miettes ou *Garbage Collector* : système permettant de gérer l'utilisation de la mémoire par des programmes.

SQL : de l'anglais « *Structured Query Language* » langage de programmation utilisé pour communiquer avec une base de données relationnelle.

Bibliographie

Ajzen, Icek. *From intentions to actions : A theory of planned behavior*. Springer Berlin Heidelberg, 1985.

Apache Software Foundation. *Apache Maven, https://maven.apache.org*. 2019.

— *Apache Spark™- Unified Analytics Engine for Big Data, http://spark.apache.org/*. 2019.

Bagozzi, Richard, Zeynep Gurhan-Canliu et Joseph Priester. *The Social Psychology of Consumer Behavior*. Open University Press, 2007.

Bagozzi, Richard, Geraldine Henderson, Pratibha A. Dabholkar et Dawn Iacobucci. *Network analyses of hierarchical cognitive connections between concrete and abstract goals : An application to consumer recycling attitudes and behaviors (Networks in marketing)*. 1996.

Fazio, Russell H. *How do attitudes guide behavior (Handbook of motivation and cognition : Foundations and social behavior)*. Guilford Publications, 1986.

Fishbein, Martin et Icek Ajzen. *Belief, attitude, intention and behavior : An introduction to theory and research*. Addison-Wesley Pub, 1975.

Katsov, Ilya. *Introduction to Algorithmic Marketing, Artificial Intelligence for Marketing Operations*. Grid Dynamics, 2018. isbn : 978-0-692-98904-3.

Mao, Mingsong, Jie Lu, Guangquan Zhang et Jinlong Zhang. « A Fuzzy Content Matching-based e-Commerce Recommendation Approach ». In : IEEE, 2015.

Mili, Hafedh, Imen Benzarti, Marie-Jean Meurs, Abdellatif Obaid, Javier Gonzales-Huerta, Narjes Haj-Salem et Anis Boubaker. « A context-aware customer experience management development framework based on ontologies and computational intelligence ». In : *Sentiment Analysis and Ontology Engineering - An Environnement of Computational Intelligence* (2016), p. 273–311.

Scala center. *The Scala programming language, https://scala-lang.org/*. 2019.

Yager, Ronald R. « Fuzzy logic methods in recommender systems ». In : *Fuzzy Sets and Systems* 136.2 (2003), p. 133–149. issn : 01650114. doi : 10.1016/S0165-0114(02)00223-3.

— « Reasoning with conjunctive knowledge ». In : *Fuzzy Sets and Systems* 28.1 (1988), p. 69–83.

— « Set-based representations of conjunctive and disjunctive knowledge ». In : *Information Sciences* 41.1 (1987), p. 1–22.

— « Toward a theory of conjunctive variables ». In : *International Journal Of General System* 13.3 (1987), p. 203–227.

Yager, Ronald R. « Veristic variables ». In : *IEEE Transactions on Systems, Man, and Cybernetics, Part B : Cybernetics* 30.1 (2000), p. 71–84. issn : 10834419. doi : 10.1109/3477.826948.

Zenebe, Azene et Anthony F. Norcio. « Representation, similarity measures and aggregation methods using fuzzy sets for content-based recommender systems ». In : *Fuzzy sets and systems* (2009), p. 76–94.

Annexe A

Processus d'achat

A.1 Bref état de l'art

Le but de l'architecture logicielle développée par les chercheurs du LATECE est de fournir une solution permettant la gestion de l'expérience client tenant compte du contexte à tous types de détaillant. Dans la réalisation d'une telle architecture, les membres de l'équipe du laboratoire ont également défini une méthodologie permettant au *framework* de répondre aux attentes des entreprises à chacune des étapes du processus d'achat de leur client. Par processus d'achat, il est question du temps entre le moment où le client a un besoin ou une envie et le moment où il satisfait ce besoin ou cette envie. Le but de la méthodologie étudiée par les chercheurs du LATECE est d'identifier les moments où une société peut interagir avec le consommateur pour le conseiller [1].

Le comportement des acheteurs est depuis longtemps scruté et de nombreux modèles psychologiques ont été proposés. La théorie de l'action raisonnée (voir Fishbein et Ajzen, 1975) considère que chacune des actions d'un consommateur sont pensées dans le but de satisfaire un besoin ou une envie (on parle d'intention). Cette intention a deux origines : le positionnement du consommateur face à cette intention (sa probabilité à atteindre son objectif) et la « norme subjective » (le regard de la société).

Cette théorie est cependant considérée comme incomplète et de nombreuses autres hypothèses s'en inspirent en ajoutant différents facteurs. Ainsi avec la théorie du comportement planifié, Ajzen, 1985 prend en compte la confiance d'un consommateur en sa capacité à réaliser son objectif. Fazio, 1986 lui détaille le comportement du consommateur dans son modèle *MODE (motivation and opportunity as determinants)*. Le comportement du consommateur est alors régulé par deux aspects : conscient et inconscient. Enfin avec la théorie de l'essai, Bagozzi, Henderson, Dabholkar et Iacobucci, 1996 considèrent que l'on doit prendre en compte la difficulté d'atteindre son but. En effet, cette dernière diffère selon si l'objectif du consommateur est quelque chose de simple réalisable par une seule action, ou s'il s'agit d'un but plus complexe, qui va se composer en différents petits objectifs. Par exemple on distingue les buts « acheter du saumon » et « avoir une vie saine ».

Le modèle étudié dans ce cadre est celui de Bagozzi, Gurhan-Canliu et Priester, 2007, qui s'inspire de Bagozzi, Henderson, Dabholkar et Iacobucci, 1996, il est détaillé dans la partie suivante.

1. et satisfaire son besoin avec son produit.

A.2 La théorie de l'essai de Bagozzi, Gurhan-Canliu et Priester, 2007

Le modèle présenté dans cette section se base sur celui de Bagozzi, Gurhan-Canliu et Priester, 2007 avec de légères nuances présentées dans le travail des chercheurs du LATECE. La théorie de l'essai regroupe les différentes aspects pouvant influencer un consommateur lorsqu'il essaie d'accomplir un objectif. Il est à noter qu'il peut y avoir différents types d'objectifs (simple et complexe (ou comportementaux)) mais pour faciliter la présentation du modèle, les chercheurs Mili et al., 2016 les considèrent de façon égale.

Plusieurs étapes se distinguent dans le processus d'achat d'un consommateur. Brièvement, il s'agit de :

- L'**identification des objectifs du consommateur** : un consommateur peut avoir plusieurs envies et besoins. Lors de cette étape, il prend conscience d'un certain nombre de ces objectifs.
- L'**établissement du but** : une fois les intentions et divers objectifs d'un consommateurs identifiées, il faut choisir lequel va se transformer en but, lequel il va chercher à atteindre.
- La **planification de l'objectif** : une fois un objectif défini le consommateur va réaliser un « plan d'attaque » pour l'atteindre. Cet objectif peut trouver son origine dans des intentions personnelles ou sociétales (ou les deux).
- L'**essai** : le déroulement de chacune des étapes planifiée précédemment, jusqu'à l'étape finale qui termine le processus.
- L'**étape finale** : dans notre cas, souvent un achat.
- L'**analyse du résultat** : la satisfaction du besoin ou l'échec.
- La **réaction** à ce résultat.

A.2.1 Identification des objectifs du consommateur

Il y a de nombreux facteurs qui influent dans la naissance d'une envie ou d'un besoin chez un individu.

Faisabilité de l'intention : une envie devient plus facilement un objectif si elle semble plausible. Par exemple, *acheter un appartement* semble plus réalisable qu'*acheter un château*. Si la seconde intention peut rester au stade de fantasme, la première a des chances de devenir un objectif.

Émotions positives anticipées : représente à la fois la sensation d'accomplissement perçue à l'idée que le consommateur atteigne son objectif mais aussi sa perception des chances de parvenir à ses fins.

Émotions négatives anticipées : représente à la fois la « peur » perçue à l'idée que le consommateur n'atteigne pas son objectif mais aussi sa perception des chances qu'il a d'échouer.

Prévision du résultat : représente l'attente qu'un individu a d'un but.

Identité sociale : l'appartenance d'un individu à un groupe social peut avoir une influence positive ou négative dans une envie.

Fréquence du comportement : représente en quelque sorte la facilité avec laquelle l'individu va se faire à une idée. Par exemple, si une personne pense pour la première fois à investir dans une moto, elle va être plus réticente que si elle a déjà pensé à cela par le passé. [2] Par *comportement*, il est sous-entendu expérience de consommation dans notre cas.

2. Ce facteur n'apparaît pas dans Bagozzi, Gurhan-Canliu et Priester, 2007 mais est un apport de Mili et al., 2016.

A.2.2 Établissement du but

Connaissant ses intentions personnelles suite à l'étape A.2.1, un individu choisit laquelle de ses envies va se transformer en objectif. Il est important de noter qu'à ce stade, la pression sociale n'est pas encore considérée.

A.2.3 Identification des désirs comportementaux du consommateur

Un objectif du consommateur peut être lié à une envie plus globale de changer de comportement. S'il choisit d'adopter un nouveau comportement, l'utilisateur peut avoir plusieurs objectifs ; c'est pourquoi les chercheurs du LATECE ne différencient pas cette étape avec celle présentée en A.2.1. Les facteurs entrant en jeu à cette étape sont les suivants.

Identité sociale : voir A.2.1.

Prévision du résultat : voir A.2.1.

Fréquence du comportement : voir A.2.1.

Attitude : influence un individu à entreprendre ou non une action. D'après Fishbein et Ajzen, 1975 il s'agit d'une combinaison de ce que le consommateur pense obtenir suite à une action et de son avis sur cette conséquence.

Normes subjectives : perception qu'à une personne de la norme sociale. C'est à dire qu'elle pense que l'on attend d'elle de faire ou de ne pas faire une action. D'après Fishbein et Ajzen, 1975, ce facteur est composé de deux aspects : la confiance en ce que l'on croit que la société attend de nous et la conséquence de cette attente (elle peut être positive comme négative).

Contrôle comportemental perçu : vision personnelle sur la difficulté d'adopter un comportement ou de réaliser une tâche. Dans la théorie du comportement planifié, Ajzen, 1985 considère qu'il s'agit d'une combinaison entre la confiance d'un individu en ses capacités (qualité, aisance...) utiles pour la réalisation d'un objectif couplée à l'importance perçue de cette capacité pour la dite tâche.

Efficacité personnelle : d'après Albert Bandura [3], il s'agit de la capacité d'un individu à persévérer.

Dans leur travaux, les chercheurs du LATECE ont choisi de considérer que les deux derniers facteurs étaient identiques.

A.2.4 Planification de l'objectif

L'objectif du consommateur étant défini, cette étape lui permet de planifier la réalisation de cet objectif. Cette étape fait directement suite à l'identification des objectifs comportementaux du consommateur (A.2.3) et à l'établissement du but (A.2.2). Elle est influencée par :

Fréquence du comportement : si l'individu a déjà réalisé cet objectif par le passé, il va pouvoir s'inspirer de ces précédentes expériences (voir A.2.1).

Valeurs morales secondaires : c'est par ce biais qu'une personne juge une action ou d'autres individus. Il permet également de justifier un comportement.

Standards auto-évalués : ils désignent l'image que le client a de lui même, celle qu'il veut avoir.

Les deux derniers facteurs sont eux mêmes influencés par l'identité sociale du consommateur.

3. dans « Self-efficacy mechanism in human agency » (1982) paru chez American Psychologist. 122–147. doi :10.1037/0003-066X.37.2.122.

A.2.5 Essai

La définition de cette étape a évolué entre Bagozzi, Henderson, Dabholkar et Iacobucci, 1996 et Bagozzi, Gurhan-Canliu et Priester, 2007 mais dans ce *framework*, elle représente les actions du consommateur entre le moment de la planification et l'obtention de son objectif. Il est à noter que l'individu peut devoir légèrement dévier de son plan suite à des imprévus.

A.2.6 Étape finale

Il s'agit en fait d'une partie de l'étape précédente. Comme son prédécesseur, elle dépend à la fois de la fréquence du comportement (voir A.2.1) mais aussi de la récence de ce dernier.

Par étape finale, il est sous entendu *achat*.

A.2.7 Analyse du résultat

Une fois l'étape finale accomplie, le consommateur peut attester la réussite ou non de son objectif. Les décisions qui font suite à ce constat sont prises dans l'étape suivante.

A.2.8 Réaction

Le consommateur va prendre en compte cette expérience dans ses prochaines tentatives qu'il ait atteint son but ou non. Chacune des décisions prises lors du prochain processus peuvent donc être impactées par ce retour d'expérience.

A.3 Adaptation de la théorie au framework

A.3.1 Framework

Le but de l'architecture logicielle développée est de permettre aux entreprises d'interagir avec leurs clients à chacune des étapes du processus identifiées précédemment tout en gardant une trace de l'historique du client afin de lui apporter des conseils plus adaptés. Les étapes précédemment citées concernent deux types d'activités : une activité de réflexion de la part du client et une d'interaction. De même, concernant les facteurs influençant ces étapes, certains sont propres à un client, d'autres peuvent être déduits.

Ainsi, l'enjeu est de provoquer une interaction avec le client pour créer ou renforcer un besoin selon son profil. La manière dont l'interaction à lieu a aussi son importance et va changer selon le profil du client. Selon l'individu le moyen de contact, en plus du contenu, diffère. La difficulté d'un tel processus est le peu d'information quant à l'étape où se trouve le client dans son processus d'achat.

En reprenant les étapes de Bagozzi, Gurhan-Canliu et Priester, 2007, voici comment une société peut influer sur le processus d'achat de ses clients :

— **Identification des intentions** et **Établissement du but** : ces deux étapes sont peu distingables car il s'agit d'activité de réflexion. Cependant en connaissant le profil du client, son identité sociale, ses valeurs, les précédents achats qu'il a fait, il devient possible de cibler ses besoins ou d'en créer.

— **Planification de l'objectif** : le but du consommateur est déjà fixé, il s'agit de faciliter ses démarches. En rendant l'accès à son objectif plus simple, on crée une interaction avec lui. Ces interactions peuvent prendre plusieurs formes selon le profil ses préférences : lui proposer un magasin près de chez lui, de passer commande en ligne, une réduction.
— **Essai** et **Étape finale** : ces étapes ne peuvent se faire sans interaction. Les scénarios habituels de gestion de l'expérience client tenant compte du contexte entre en jeu.
— **Analyse du résultat** et **Réaction** : il s'agit d'une étape délicate car les retours utilisateurs sont souvent mal perçus par ces derniers. Cependant de nombreux moyens existent allant des enquêtes de satisfaction aux messages postés sur les réseaux sociaux.

A.3.2 Outils

Pour rendre un tel *framework* fonctionnel, divers outils peuvent être utilisés, parmi lesquels :

Customer scripting : spécifications des interactions entre une société fournissant un service et ses potentiels clients. Le *scripting* peut rester général ou bien personnalisé. Il définit également la manière dont la société va interagir avec un client : comment, par quel biais, à quelle fréquence, ...

Ontologies : ensemble de règles et de relations qui permettent d'exprimer un domaine donné. Ici, les ontologies concernent autant les clients, que les produits et services de l'entreprise.

Data-mining : pour de l'analyse de requêtes, de sentiments ou d'opinion dans le cas de processus informatisés.

Recherche d'informations : pour mieux cibler les consommateurs, avoir des informations plus précises et limiter l'absence d'informations.