UNIVERSITÉ DU QUÉBEC À MONTRÉAL

LISTES DE COURSES INTELLIGENTES BASÉES SUR L'APPLICATION D'ALGORITHMES DE PARTITIONNEMENT ET DE RÉSEAUX DE NEURONES RÉCURRENTS

MÉMOIRE

PRÉSENTÉ

COMME EXIGENCE PARTIELLE

DE LA MAÎTRISE EN INFORMATIQUE

PAR

MOHAMED ACHRAF BOUAOUNE

JUILLET 2023

REMERCIEMENTS

Je tiens avant toute chose à adresser mes sincères remerciements à mon directeur de recherche, le professeur Vladimir Makarenkov, pour son aide précieuse, sa patience et ses conseils avisés tout au long de mon parcours à l'UQAM et à travers les différentes étapes de mes travaux de recherche.

Je veux également remercier les personnes impliquées de près ou de loin dans mon parcours et qui m'ont, chacune à sa façon, apporté de l'aide.

Enfin, je souhaite faire part de ma gratitude et de ma reconnaissance envers ma femme, mes amis et ma famille qui m'ont apporté un soutien indéfectible et sans qui je n'en serai pas là aujourd'hui.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

ACRONYMES

DL **D**eep **L**earning.

DT **D**ecision **T**ree.

GRU **G**ated **R**ecurrent **U**nit.

LSTM **L**ong **S**hort **T**erm **M**emory.

ML **M**achine **L**earning.

MLP **M**ulti **L**ayer **P**erceptron.

RF **R**andom **F**orest.

RNN **R**ecurrent **N**eural **N**etwork.

SVM **S**upport **V**ector **M**achine.

PUBLICATIONS

Article de revue : Chabane N*****, **Bouaoune A***, Tighilt R, Abdar M, Boc A, Lord E, Tahiri N, Mazoure B, Acharya U R, Makarenkov V (2022) Intelligent personalized shopping recommendation using clustering and supervised machine learning algorithms. PLoS ONE 17(12) : e0278364. DOI: 10.1371/journal.pone.0278364

Article de conférence : Chabane N*****, **Bouaoune A***, Tighilt R, Mazoure B, Tahiri N, Makarenkov V (2023), Using Clustering and Machine Learning Methods to Provide Intelligent Grocery Shopping Recommendations. À paraître dans "Proceedings of IFCS 2022" dans le livre "Classification and Data Science in the Digital Age, 1st ed. 2023" de la Springer Series "Studies in Classification, Data Analysis, and Knowledge Organization". Le livre comprend des articles sélectionnés et évalués par des pairs présentés lors de la 17e Conférence de la International Federation of Classification Societies (IFCS 2022), tenue à Porto, Portugal, du 19 au 23 juillet 2022.

***** : ces deux auteurs ont contribué de façon équivalente à cette publication.

Pour chaque article, je me suis chargé entre autres de conceptualiser, de mettre en place et de faire évoluer l'architecture de la solution proposée, d'analyser et de synthétiser les résultats obtenus, de réaliser des figures d'illustration, de rédiger certaines sections des articles et d'étudier en détail et de sélectionner les références citées.

RÉSUMÉ

La recommandation du prochain panier d'épicerie est une tâche essentielle dans le domaine de l'analyse des données du panier de consommation. Avec la popularité du commerce électronique, la pléthore de produits disponibles et la variété de nouvelles promotions offertes chaque semaine, les clients font face à un défi important consistant à sélectionner les bons produits dans leur panier d'achat afin de répondre à leurs besoins réels. Avec ce contexte, les listes de courses représentent un outil central dans les habitudes d'achat de nombreux clients.

Dans ce mémoire, nous présentons un nouveau système de recommandation d'épicerie disponible sur la plateforme CircuitPromo. Notre système en ligne réalise une étape de partitionnement afin de répartir les profils des clients en quatre groupes distincts et utilise différents algorithmes traditionnels d'apprentissage automatique et d'apprentissage profond afin de fournir des recommandations aux utilisateurs en temps réel.

Notre algorithme d'apprentissage profond basé sur l'utilisation d'une architecture de réseau de neurones récurrents à portes (GRU) peut être vu comme une extension du DREAM (*Dynamic REcurrent bAsket Model*) adapté à la classification multi-classes (c'est-à-dire multi-magasins). Parmi les algorithmes d'apprentissage automatique traditionnels, le F-score moyen le plus élevé sur l'ensemble de données considéré (831 clients) a été obtenu à l'aide de Random Forest avec une valeur de 0,516. Notre algorithme d'apprentissage profond a obtenu une performance supérieure avec un F-score moyen de 0,559. Le principal avantage du système présenté ici est de proposer une recommandation indiviualisée et taillée sur-mesure pour chaque client : un modèle d'apprentissage automatique ou d'apprentissage profond distinct est construit pour chaque client considéré. Une telle approche personnalisée nous permet de surpasser les résultats de prédiction fournis par les modèles d'apprentissage profond les plus aboutis.

Mots-clés : Apprentissage automatique supervisé, apprentissage profond, partitionnement, forêt d'arbres décisionnels, réseaux de neurones récurrents, système de recommandation.

CHAPITRE I

INTRODUCTION

L'épicerie est une activité courante qui implique plusieurs facteurs importants tels que le temps, le budget et la pression d'achat (Vincent-Wayne et Aylott, 1998). Dans ce contexte, des listes d'épicerie bien conçues peuvent être un outil de planification et de budgétisation efficace. Plusieurs études ont indiqué qu'une majorité de clients modernes s'appuient sur une liste d'épicerie écrite, mentale ou numérique (Newcomb *et al.*, 2003; Bhattacharya *et al.*, 2012) afin de les aider dans leurs achats. De plus, les mêmes études ont également révélé que les consommateurs avaient généralement un intérêt croissant pour les applications qui les aidaient à gérer de manière interactive leurs listes de courses, tout en les informant sur les prix des produits et les offres spéciales.

1.1 Mise en contexte

En règle générale, les détaillants en épicerie proposent de nouvelles promotions chaque semaine pour attirer de nouveaux clients et améliorer les ventes et leurs bénéfices. Par exemple, Walters et Jamil ont montré que lors d'une course d'épicerie régulière impliquant des produits inter-catégories, 39% des articles dans le panier d'un client étaient des offres spéciales. Ces auteurs ont également conclu qu'environ 30% des clients interrogés étaient fortement influencés par différents

coupons et promotions (Walters et Jamil, 2002).

Alors que les prix spéciaux permettent parfois aux clients de réaliser des économies importantes, des milliers d'entre eux sont généralement publiés chaque semaine, entraînant souvent une énorme surcharge d'informations. Cela rend la tâche de sélectionner les offres les plus avantageuses pour un client donné extrêmement difficile (Park et Chang, 2009).

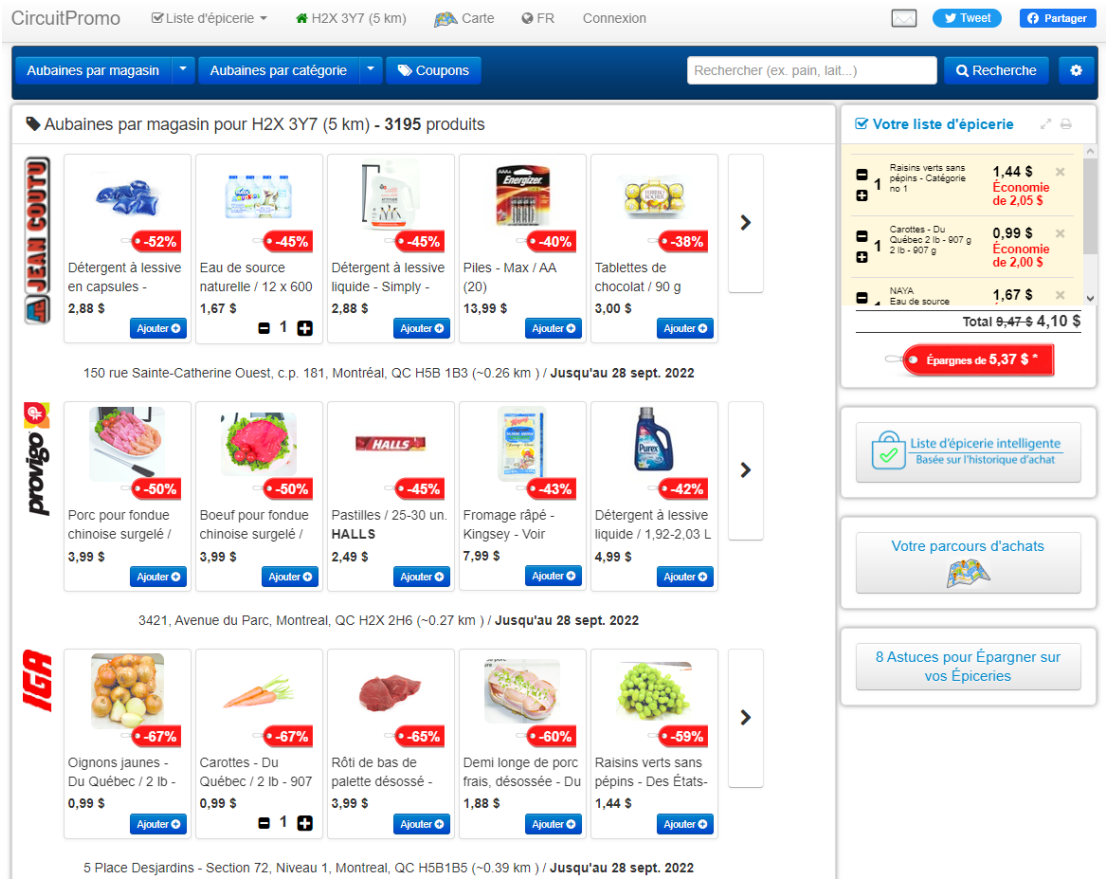

FIGURE 1.1 – Page d'accueil du site CircuitPromo. Des produits avec un rabais de différents magasins situés à proximité d'un code postal ou d'une adresse spécifiée (au Canada) peuvent être ajoutés au panier de l'utilisateur.

Avec le développement des achats en ligne, les progrès récents des techniques

d'apprentissage automatique et les réactions favorables de nombreux clients aux applications conviviales visant à améliorer leur expérience d'achat, le développement d'un système de recommandation d'épicerie en ligne capable de fournir de précieuses recommandations individuelles semble être une tâche très pertinente. CircuitPromo [1] est un bon exemple d'un tel système de recommandation. CircuitPromo est une base de données et un site Web d'achats canadiens qui permettent aux utilisateurs de gérer leurs listes d'épicerie en fonction des promotions hebdomadaires disponibles dans la plupart des grandes épiceries situées dans leur région (Tahiri *et al.*, 2019).

1.2 Problématique et contributions

L'un des principaux objectifs de ce mémoire est de présenter un nouveau système de recommandation pour l'épicerie basé sur l'apprentissage profond et prenant en compte les historiques d'achat, les profils, les préférences et les promotions hebdomadaires disponibles pour les utilisateurs de CircuitPromo afin de les aider à créer une liste d'épicerie hebdomadaire personnalisée et qui leur est profitable (voir Fig. 1.1).

Nos principales contributions sont les suivantes :

1. Nous présentons notre nouveau système de recommandation personnalisé disponible sur la plateforme CircuitPromo ;
2. Nous effectuons une analyse de l'étape de partitionnement qui permet de répartir les clients canadiens en groupes distincts, sans chevauchement, en fonction de leurs habitudes d'achat ;
3. Nous définissons les caractéristiques du ratio de fidélité basé sur la quantité (RFQ) et du ratio de fidélité basé sur le prix (RFP) pour caractériser le

1. http ://CircuitPromo.ca

comportement d'achat du client ;

4. Nous décrivons et appliquons un nouveau modèle RNN-GRU d'apprentissage profond (c'est-à-dire un modèle DREAM multi-classes étendu) pour prédire si un produit donné doit être inclus dans le prochain panier multi-magasins de l'utilisateur, et pour recommander le magasin où l'achat doit être effectué (le cas échéant) ;
5. Nos résultats et nos analyses du partitionnement suggèrent que différents modèles de prédiction (c'est-à-dire des modèles d'apprentissage automatique traditionnels ou notre modèle d'apprentissage profond) devraient être utilisés pour différents groupes de clients.

1.3 Structure du mémoire

Le mémoire est organisé de la façon suivante. Nous résumons d'abord dans le second chapitre les travaux connexes dans le domaine des systèmes de recommandation. Au sein du chapitre 3, nous commençons par présenter les principales fonctionnalités de la plateforme CircuitPromo ainsi que le système de recommandation qui y est associé avant de décrire les données utilisées lors de nos expérimentations. Le chapitre 4 aborde dans les détails l'application des algorithmes de partitionnement, d'apprentissage automatique traditionnels et d'apprentissage profond. Nos résultats sont ensuite décrits et discutés dans le chapitre 5, avant d'aboutir à nos principales conclusions.

CHAPITRE II

ÉTAT DE L'ART

2.1 Vue d'ensemble sur les systèmes de recommandation

Les systèmes de recommandation (*SR*) (Ricci *et al.*, 2015) constituent un domaine d'étude de plus en plus important depuis les premiers articles de recherche sur le filtrage collaboratif au milieu des années 90 (Resnick *et al.*, 1994; Shardanand et Maes, 1995; Park *et al.*, 2012), et l'expansion du commerce électronique et les achats en ligne (Zhou *et al.*, 2007). Les systèmes de recommandation comprennent des algorithmes et des logiciels visant à fournir aux utilisateurs des recommandations d'articles personnalisées pour leur apporter de l'aide face à des problèmes tels que la surcharge d'informations et pour les aider dans les processus de prise de décision. Les éléments recommandés représentent la sortie du système de recommandation et leur nature peut varier selon le contexte. Entre autres, les éléments peuvent inclure des films, des chansons, des produits vendus au détail ou des documents en ligne (Ricci *et al.*, 2015; Lu *et al.*, 2015).

De nos jours, plusieurs stratégies pour construire des systèmes de recommandation ont été décrites dans la littérature. Nous vous présentons ici les plus populaires d'entre elles, et celles liées à notre cas d'étude.

Dans leurs travaux, Melville et Sindhwani ont classé les techniques de SR en trois

grandes catégories (Melville et Sindhwani, 2017) :

— Approches de filtrage collaboratif (*CF*, pour *collaborative filtering*) ;
— Approches de filtrage basé sur le contenu (*CB*, pour *content-based*) ;
— Approches hybrides.

Le filtrage collaboratif est l'une des techniques les plus populaires et les plus efficaces (Deshpande et Karypis, 2004; Koren et Bell, 2015). Elle est basée sur le concept de *bouche-à-oreille* et admet qu'un utilisateur fait confiance à un autre utilisateur avec un raisonnement et des goûts similaires. Il suppose également que deux utilisateurs similaires ont des intérêts similaires et que deux éléments similaires ont des notes similaires (Verma et Aggarwal, 2020). Les limitations les plus courantes rencontrées par les méthodes de filtrage collaboratif sont les problèmes de démarrage à foid (*cold start*) et de matrices creuses (*sparse matrix*) (Lika *et al.*, 2014). Le problème de démarrage à froid se caractérise par le manque d'informations initiales concernant un utilisateur ou un élément nouvellement introduit, alors que le problème de matrice creuse se produit généralement lorsqu'un utilisateur donné a tendance à interagir avec quelques éléments uniquement sur la quantité massive de produits disponibles (Shi *et al.*, 2014; Lu *et al.*, 2015; Chen *et al.*, 2018).

Le filtrage basé sur le contenu, en revanche, a tendance à recommander des éléments dont les attributs et les caractéristiques sont similaires à d'autres éléments pour lesquels un utilisateur donné a montré un intérêt positif par le passé (Pazzani et Billsus, 2007). Cette approche nécessite l'utilisation de métadonnées relatives à chaque élément considéré ce qui peut parfois représenter un défi.

Dans une tentative de surmonter les limitations des techniques de filtrage collaboratif et de filtrage basé sur le contenu, des approches hybrides, essayant de combiner les deux, ont été introduites. Les travaux d'Adomavicus *et al.* ainsi que,

plus récemment, de Lu *et al.* ont passé en revue différentes méthodes utilisées dans le domaine des SR, en mettant en évidence leurs avantages et leurs inconvénients tout en donnant un aperçu des développements futurs dans le domaine (Adomavicius et Tuzhilin, 2005; Lu *et al.*, 2015).

Récemment, plusieurs extensions importantes des approches SR traditionnelles ont été introduites (Karimi *et al.*, 2018; Eirinaki *et al.*, 2018; Villegas *et al.*, 2018). Les principales d'entre elles sont les suivantes :

— Systèmes de recommandation basés sur la connaissance (*KBSR*, pour *Knowledge-based recommender systems*) ;
— Systèmes de recommandation sensibles au contexte (*CASR*, pour *Context-aware recommender systems*) ;
— Systèmes de recommandation basés sur la démographie (*DBSR*, pour *Demographic-based recommender systems*).

Les systèmes de recommandation basés sur la connaissance (Burke, 2000; Aggarwal, 2016) peuvent être utilisés efficacement pour recommander des produits hautement personnalisés (comme l'immobilier ou les automobiles). Contrairement aux méthodes classiques telles que le filtrage collaboratif ou le filtrage basé sur le contenu, un système de recommandation basé sur la connaissance cherche à obtenir les besoins explicites de l'utilisateur par sollicitation directe, permettant ainsi à l'utilisateur d'avoir plus de contrôle sur la recommandation tout en créant une retour d'information (*feedback*) interactif.

Les systèmes de recommandation contextuels (Villegas *et al.*, 2018) s'appuient sur plusieurs sources d'informations pour identifier un certain contexte et générer des recommandations plus précises (par exemple, recommander des maillots de bain au lieu de manteaux d'hiver en été).

Enfin, les systèmes de recommandation basés sur la démographie (Al-Shamri, 2016) regroupent les utilisateurs en fonction de leurs attributs démographiques disponibles (c'est-à-dire l'âge, le sexe, l'emplacement), en supposant que les personnes du même groupe (quartier) évaluent les éléments de la même manière. Cette approche a été introduite à l'origine pour améliorer la qualité des recommandations, mais elle s'est également avérée utile pour résoudre le problème du démarrage à froid (Safoury et Salah, 2013).

2.2 Systèmes de recommandation du prochain panier

Rappelons maintenant quelques travaux récents abordant la question de la recommandation du prochain panier d'épicerie.

Yu *et al.* ont introduit un modèle efficace, appelé Dynamic REcurrent bAsket Model (DREAM), basé sur des réseaux de neurones récurrents (*RNN*, pour *Recurrent neural network*). L'un des principaux avantages de DREAM est qu'il est non seulement capable d'apprendre une représentation dynamique du comportement d'un utilisateur mais aussi de prendre en compte les caractéristiques séquentielles globales entre les paniers (Yu *et al.*, 2016). Cependant, le modèle DREAM original de Yu *et al.* a été conçu pour seulement effectuer une classification binaire. Pour chaque produit disponible, le modèle génère un score de probabilité représentant la probabilité que ce produit soit inclus dans le prochain panier acheté par un client donné. Néanmoins, DREAM ne peut pas fournir de prédictions dans un contexte multi-magasins (c'est-à-dire multi-classes), consistant à prédire le magasin où le produit recommandé doit être acheté. De plus, dans leurs travaux, Yu *et al.* n'ont pas tenu compte de certaines caractéristiques importantes telles que les prix des produits, leur disponibilité ainsi que les offres spéciales hebdomadaires proposées dans les magasins locaux. Cela nous a motivés à généraliser le modèle

DREAM original à une tâche de classification multi-classes pour prédire à la fois si un produit donné doit être inclus dans le prochain panier du client et également dans quel magasin l'achat doit être effectué (pour plus de détails, voir le chapitre 4).

Che *et al.* ont décrit une nouvelle méthode de prédiction utilisant des réseaux de neurones récurrents basés sur l'attention afin de détecter et de modéliser les relations inter- et intra-panier. Les auteurs ont proposé de considérer tous les paniers disponibles de l'utilisateur concerné dans le but de modéliser ses préférences à long terme (relations inter-paniers) (Che *et al.*, 2019). Dans le même temps, le modèle d'attention intra-panier est destiné à agir à l'échelle des articles dans les paniers les plus récents de l'utilisateur pour prédire son comportement et ses préférences à court terme. Grâce à leur modèle d'attention adaptatif, Che *et al.* ont pu surpasser les méthodes de pointe pour la recommandation du prochain panier. Mais là encore, leur méthode ne s'applique que dans un contexte de classification binaire.

Faggioli *et al.* ont utilisé le facteur de récence pour prédire le prochain panier d'épicerie d'un consommateur en appliquant une méthode de prédiction basée sur le filtrage collaboratif dans un cadre général de recommandation par classement des meilleurs produits recommandables (*top-n products*). Pour démontrer l'efficacité de leur approche, les auteurs l'ont comparée à certains modèles de filtrage collaboratif de pointe. La méthode mise au point durant l'étude repose sur deux aspects jugés essentiels pour la recommandation du prochain panier d'un utilisateur : la popularité des produits considérés et la fenêtre de récence relative à l'achat de ces derniers (Faggioli *et al.*, 2020). Ces deux facteurs sont associés et intégrés à une approche de filtrage collaboratif qui s'est montrée capable de rivaliser en termes de performance avec les méthodes reconnues utilisées à titre de comparaison.

Les recommandations basées sur le contenu se sont également révélées efficaces dans le domaine de la recommandation du prochain panier et des coupons d'épicerie. Dans ce contexte, Xia *et al.* ont proposé un modèle de filtrage basé sur le contenu qui repose sur l'utilisation d'un arbre pour les recommandations de coupons (Xia *et al.*, 2017). Ces auteurs ont conformisé le processus de sélection des coupons afin de personnaliser la recommandation et ainsi augmenter le taux de clics. En utilisant les classificateurs Random Forest et XGBoost, Xia *et al.* ont pu améliorer le taux de clics estimé sur le coupon pour le faire passer de 1,20% à 7,80%.

Par ailleurs, Prokhorenkova *et al.* ont décrit et testé une nouvelle méthode statistique basée sur le modèle CatBoost de Yandex pour prédire si un client donné est sensé acheter certains produits sélectionnés. Cette étude a présenté certaines techniques algorithmiques majeures avec notamment le boosting ordonné (*ordered boosting*) qui est une alternative basée sur la permutation ainsi qu'un nouvel algorithme dédié au traitement des catégories (Prokhorenkova *et al.*, 2018a). La combinaison de ces techniques a permis à CatBoost de surpasser d'autres implémentations de boosting en termes de performance sur plusieurs ensembles de données.

Xiaotong Dou a considéré les données d'achat réelles et surtout déséquilibrées d'une plateforme d'e-commerce puis a utilisé le modèle CatBoost pour prédire si les clients achèteront ou non certains produits disponibles (Dou, 2020). La méthode proposée par Dou a obtenu une précision de 88,51% lors de la prédiction. Le modèle proposé a été en mesure de réduire efficacement les problèmes de surajustement courants avec les données déséquilibrées grâce à des arbres symétriques et adopte une approche algorithmique plus scientifique et interprétable pour les variables catégorielles, ce qui a eu pour conséquence de diminuer la perte d'informations lors de l'entraînement du modèle et d'améliorer la robustesse de ce dernier.

Lee *et al.* ont proposé d'utiliser des réseaux de neurones récurrents au lieu des techniques de filtrage collaboratif pour créer un système de recommandation de produits dit multi-périodes qui est lié à un marché alimentaire en ligne. Le système introduit par Lee *et al.* est capable de recommander des produits selon plusieurs périodes sur une séquence temporelle définie. Les auteurs ont montré que le système de recommandation proposé offrait une meilleure performance en termes de précision et de diversité dans une perspective multi-période comparativement à des systèmes basés sur le filtrage collaboratif (Lee *et al.*, 2020). De plus, le système proposé s'est également révélé robuste face aux schémas d'achat répétitifs des utilisateurs.

Zheng et Ding ont proposé un système de recommandation personnalisé et immersif basé sur un réseau de neurones à graphes (*IGNN*, pour *Immersive Graph Neural Network*), qui vise à augmenter le potentiel de commercialisation de divers produits, à améliorer l'expérience d'achat des utilisateurs, à promouvoir les ventes ainsi qu'à dynamiser la croissance du marché. L'« immersion » fait ici référence à l'impication totale de l'utilisateur dans son activité d'achat, à travers une expérience utilisateur lui permettant de se concentrer sur ses besoins actuels (Zheng et Ding, 2022). Cet aspect combine les facteurs qualitatifs et quantitatifs afin d'interpréter les besoins psychophysiologiques des utilisateurs dans un espace numérique. Les auteurs ont donc mis en place un environnement marketing immersif utilisant des modèles d'apprentissage profond et des réseaux de neurones en graphes (*GNN*). Les résultats de leurs travaux suggèrent que l'approche d'un tel marketing immersif parvient à particulièrement bien refléter les attributs et caractéristiques essentiels des produits. Le modèle proposé et les méthodes auxquelles il a été comparé ont été testés sur des ensembles de données publics. Le modèle mis au point par les auteurs a surpassé les autres méthodes en termes de précision et de rappel. Cependant, comme suggéré par les auteurs, le système

de recommandation proposé n'a pas été vérifié dans des applications pratiques. Ainsi, l'impact du modèle présenté sur des utilisateurs réels n'a pas été évalué.

Tahiri *et al.* ont récemment proposé d'utiliser à la fois un réseau de neurones récurrent et un réseau de neurones à propagation avant (*FFNN*, pour *Feed Forward Neural Network*) qu'ils ont combinés à une factorisation par matrices non-négatives (*NNMF*, pour *Non-negative Matrix Factorization*) et à des arbres de boosting de gradient (*GBT*, pour *Gradient Boosting Tree*) afin de construire des paniers d'épicerie intelligents pour les utilisateurs de la plateforme CircuitPromo. Tahiri *et al.* ont considéré des variables différentes et un nombre réduit de clients (par rapport à notre étude) pour décrire le comportement des utilisateurs de la plateforme. Leur meilleur résultat pour le F-score a été de 0,37 (Tahiri *et al.*, 2019). Ce résultat a été obtenu lorsque leur modèle de prédiction a été appliqué à un ensemble de données augmenté. Cependant, dans leur travaux, les auteurs n'ont effectué aucune analyse de partitionnement et n'ont pas pris en compte différents profils d'utilisateurs. Comme nous le verrons dans les prochaines sections, ce type d'analyse se révèle être d'une grande importance lorsqu'il s'agit d'améliorer les performances de prédiction du prochain panier d'un utilisateur. Aussi, Tahiri *et al.* n'ont pas comparé les résultats générés par leur modèle d'apprentissage profond avec ceux fournis par les algorithmes d'apprentissage machine traditionnels. Une telle comparaison est cruciale lorsque l'ensemble de données disponible est relativement petit. Enfin, le modèle d'apprentissage profond introduit par les auteurs n'est pas personnalisé car la même architecture du modèle a été utilisée pour tous les clients considérés.

Dans leur article, Gupta et Shrinath ont présenté un modèle basé sur le filtrage collaboratif conçu pour surmonter le problème du démarrage à froid. Pour y parvenir, les auteurs ont proposé de calculer la somme pondérée de quatre variables différentes (Gupta et Shrinath, 2022). La première d'entre elles représente la no-

tation des produits obtenue à l'aide de la factorisation pondérée par matrices non-négatives (*Weighted NNMF*) suite à laquelle une technique de propagation d'affinité a été appliquée. Les trois autres variables considérées sont des mesures de similarité liées à la notion de graphes et basées sur les métadonnées des utilisateurs ainsi que sur leurs habitudes d'achat. Gupta et Shrinath ont indiqué que leur modèle surpassait les approches existantes en se basant sur le taux de réussite (*HT*, pour *Hit Ratio*) et le gain cumulé actualisé normalisé (*nDCG*, pour *Normalized Discounted Cumulative Gain*) en guise de métriques.

Li *et al.* ont suggéré plusieurs nouvelles métriques pour mesurer le ratio de répétition/exploration dans les habitudes d'achats des clients afin d'évaluer les performances des systèmes de recommandation du prochain panier. Ils ont comparé et analysé les résultats de modèles de recommandation du prochain panier à la pointe de la technologie sur trois ensembles de données publics. Leur étude a été menée en mettant l'accent sur leurs nouvelles métriques afin d'aider à illustrer la portée et l'état actuel de la recherche centrée sur ce domaine en particulier, puis d'expliquer les progrès apportés par les approches existantes (Li *et al.*, 2021). Leurs travaux avaient également pour but de mettre en évidence les raisons derrière les progrès revendiqués par les méthodes étudiées. Li *et al.* ont indiqué que les futures recherches sur la recommandation du prochain panier devraient envisager une analyse du comportement de répétition et d'exploration (c'est-à-dire la découverte de nouveautés) pour obtenir des informations utiles et aider à concevoir des modèles non biaisés.

Le *et al.* ont proposé un cadriciel pour modéliser les séquences de paniers d'un utilisateur donné. Leur modèle de réseau hiérarchique, appelé Beacon et basé sur une architecture LSTM (pour *Long short-term memory*), est constitué de trois composants principaux, prenant en entrée une séquence de panier et une matrice de corrélation (Le *et al.*, 2019). Le premier composant, l'encodeur de panier,

produit des représentations de paniers sensibles à la corrélation après avoir capturé des relations et des réciprocités intra-panier entre les produits de ce dernier. La séquence de représentations des paniers est ensuite utilisée comme entrée pour un encodeur de séquences, le second composant, afin d'extraire des associations séquentielles inter-panier. La sortie de ce composant est associée à la matrice de corrélation, et les deux sont utilisées par le troisième composant, le *prédicteur*, pour générer le prochain panier de l'utilisateur prenant en compte la corrélation des produits. Ainsi, Le *et al.* ont pris en compte les dépendances corrélatives entre les produits dans le but d'améliorer la représentation des paniers individuels ainsi que la séquence plus globale des paniers de l'utilisateur.

Dans un récent article, (Ariannezhad *et al.*, 2022) ont réalisé une étude approfondie sur les habitudes de consommation récurrentes chez des utilisateurs dans le domaine de l'épicerie en ligne. À travers l'analyse de données transactionnelles issues à la fois de sources publiques et propriétaires, ils ont analysé les comportements d'achat et sont arrivés à la conclusion qu'une part significative des performances des systèmes de recommandation pour le prochain panier peut être attribuée aux produits que les utilisateurs ont déjà achetés par le passé.

Dans ce contexte, les auteurs ont introduit un nouveau modèle de réseau de neurones, ReCANet, spécifiquement conçu pour prendre en compte les habitudes de consommation récurrentes des utilisateurs. Pour ce faire, le modèle utilise les informations relatives aux produits précedemment achetés pour prédire de manière plus précise quels seront les articles sélectionnés par l'utilisateur dans son prochain panier d'achat.

Les résultats obtenus par les auteurs ont mis en évidence que ReCANet surpasse les modèles actuels de recommandation pour le prochain panier en termes de rappel et de nDCG. Ils ont aussi effectué une étude d'ablation pour expliquer l'impact

de chaque composant de ReCANet sur ses performances globales, montrant que chacun de ses composants contribue de manière significative aux performances obtenues. Enfin, ils ont montré que le ratio de répétition d'un utilisateur, c'est-à-dire la fréquence à laquelle un utilisateur achète à nouveau le même article, a une influence directe sur l'efficacité de leur nouveau modèle.

Dans un autre article, (Ariannezhad *et al.*, 2023) se sont également penché sur la recommandation personnalisée en temps réel du panier actuel de l'utilisateur dans le contexte de l'achat en ligne, en particulier dans le domaine de l'épicerie. Ils ont mis en avant un autre modèle, PerNIR, basé sur le voisinage et qui prend en compte à la fois l'historique personnel de l'utilisateur et son panier actuel. Cette approche considère les intérêts à court terme de l'utilisateur, représentés par le panier en cours de constitution, et ses intérêts à long terme, reflétés par son historique d'achat. Les profils d'utilisateurs voisins sont également considérés afin de capturer un comportement d'achat « collaboratif ». Les résultats obtenus par les auteurs montrent que PerNIR surpasse d'autres approches avec une marge significative, offrant des gains de plus de 12% en termes de taux de succès par rapport à la deuxième meilleure approche utilisée qui correspond à leur précèdent modèle, ReCANet. Les auteurs ont également mis l'accent sur l'optimisation de leur nouvelle méthode, qui est en mesure de fournir rapidement des recommandations dans des situations en conditions réelles.

CHAPITRE III

PRÉSENTATION DE LA PLATEFORME CIRCUITPROMO ET DESCRIPTION DES DONNÉES

3.1 La plateforme CircuitPromo

CircuitPromo est une plateforme canadienne d'information sur l'épicerie disponible en anglais et en français. L'objectif principal de CircuitPromo est de fournir aux utilisateurs des informations à jour sur les meilleures offres d'épicerie proposées par les principaux détaillants en alimentation de leur région, leur permettant ainsi de comparer les produits disponibles et de créer des listes d'épicerie hebdomadaires personnalisées en fonction des informations fournies.

Les principales fonctionnalités de la plateforme CircuitPromo sont les suivantes. Elle permet aux utilisateurs de :

1. Rechercher et comparer les offres de spéciaux annoncés au niveau des épiceries locales préférées de l'utilisateur ;
2. Créer, enregistrer, gérer et imprimer des listes de courses hebdomadaires (voir Fig. 3.1) ;
3. Afficher une carte des épiceries et des pharmacies locales disponibles pour un code postal ou une adresse donnée ;
4. Comparer le prix d'un produit sélectionné dans un magasin en particulier

sur une période de 3 mois (voir Fig. 3.2) ;

5. Trouver des coupons d'épiceries canadiennes populaires,
6. Afficher le parcours d'achat optimal en fonction de la liste de courses hebdomadaire de l'utilisateur (voir Fig. 3.3) ;
7. Recevoir des alertes par e-mail lorsque les produits préférés de l'utilisateur sont en vente ;
8. Créer des listes de courses intelligentes et personnalisées en fonction d'un système de recommandation basé sur des algorithmes d'apprentissage automatique ou profond (voir Fig. 3.4). Cette recommandation est basée sur l'historique d'achat de l'utilisateur, la disponibilité des produits préférés de l'utilisateur et les spéciaux hebdomadaires offerts dans les épiceries et les pharmacies locales.

Une autre fonction de la plateforme CircuitPromo permet aux clients de comparer les prix d'un produit sélectionné dans différents magasins, ce qui facilite grandement l'identification de véritables aubaines. Les clients ont la possibilité de modifier leur zone de recherche en fonction de leur position géographique et de leurs besoins tout en affichant les produits d'épicerie disponibles. La distance de recherche depuis le domicile de l'utilisateur peut varier de 1 à 20 kilomètres et peut être précisée depuis l'interface d'accueil du site. De plus, les utilisateurs de CircuitPromo peuvent facilement créer, gérer et enregistrer leurs listes de courses, puis y accéder à tout moment.

Bien que les utilisateurs ne puissent pas acheter des articles auprès des détaillants directement via le site Web, ils peuvent ajouter des produits de différents magasins à leurs paniers. Une fois qu'une liste d'épicerie hebdomadaire est créée, le système recommande à l'utilisateur le chemin le plus court au départ de son domicile et qui passe par tous les commerces d'alimentation ou les pharmacies sélectionnés pour au final revenir à son domicile. Le tout, en prenant en compte le prix du

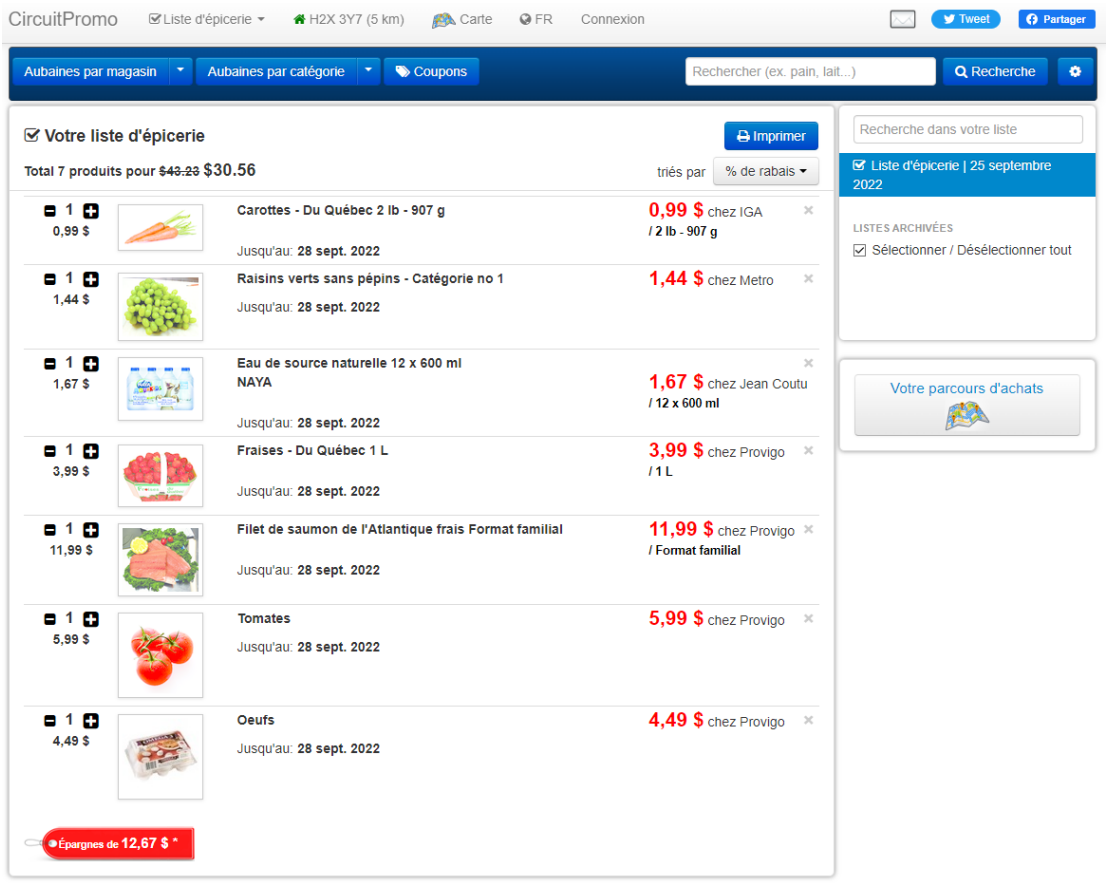

FIGURE 3.1 – Exemple de liste de courses d'un client sur le site CircuitPromo.

carburant dans le cas où l'utilisateur est véhiculé. Un algorithme efficace pour résoudre le problème généralisé du voyageur de commerce de Tasgetiren *et al.* a été implémenté pour cette fonctionnalité (Tasgetiren *et al.*, 2007). Celle-ci prend donc en compte les informations relatives au trafic local en temps réel fournies par l'API Google Maps ainsi que la position géographique des magasins les plus proches appartenant aux commerces sélectionnés (plusieurs magasins pour un détaillant sélectionné peuvent être disponibles dans une zone donnée).

Le caractère unique de CircuitPromo est dû à l'utilisation du système de recommandation intelligent permettant aux utilisateurs enregistrés d'obtenir des

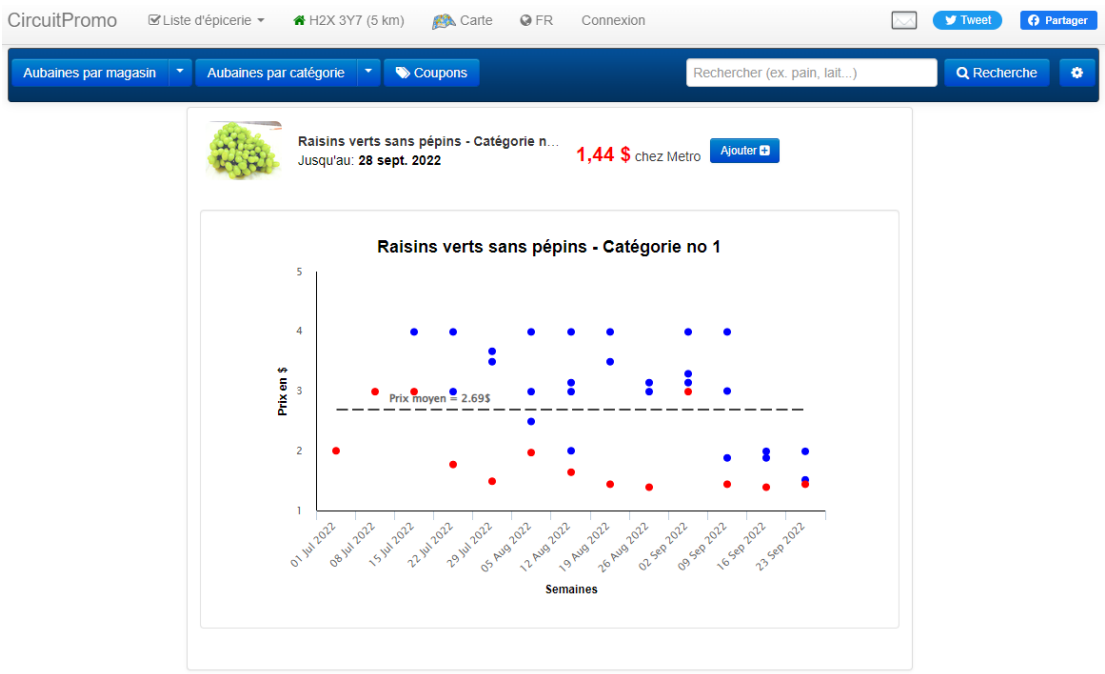

FIGURE 3.2 – Comparaison de prix pour un produit sélectionné (raisins verts sans pépins) dans les magasins locaux de Montréal sur une période de 3 mois affiché sur le site CircuitPromo. Les points rouges représentent les meilleures offres hebdomadaires pour le produit sélectionné. Les points bleus représentent les autres prix du produit disponibles au cours d'une semaine donnée. Des informations supplémentaires sur le prix et le magasin deviennent disponibles lorsque vous touchez un point.

recommandations d'épicerie hebdomadaires personnalisées basées sur l'utilisation de Random Forest, SVM ou de notre algorithme DREAM étendu basé sur un RNN-GRU qui a donné les meilleurs résultats de prédiction lors de nos expérimentations (voir le chapitre 5).

Un compte test aux coordonnées suivantes (login : test@test.com ; mot de passe : 123456) a été créé. Il peut être utilisé pour tester notre système de recommandation personnalisé et intégré à la plateforme CircuitPromo.

3.2 Description et prétraitement des données

Dans cette section, nous présentons le jeu de données qui a été utilisé pour notre étude. Nous avons considéré 831 utilisateurs de la plateforme Web CircuitPromo avec des quantités variables de listes d'épicerie hebdomadaires enregistrées (le nombre de paniers/listes par utilisateur varie entre 3 et 99). Toutes les données réelles considérées ici ont été anonymisées. Les listes d'épicerie utilisées lors de nos expériences comprenaient les produits d'épicerie que les utilisateurs prévoyaient d'acheter au cours d'une semaine donnée (la période couverte par ces données est comprise entre janvier 2017 et juin 2021). Les caractéristiques suivantes (c'est-à-dire les variables explicatives) de l'ensemble de données d'origine ont été prises en compte pour réaliser nos expériences :

- *user_id* (numérique) : identifiant unique de l'utilisateur ;
- *list_id* (numérique) : identifiant unique de la liste de courses ;
- *product_id* (numérique) : identifiant unique du produit ;
- *category* (catégorique) : catégorie du produit ;
- *price* (numérique) : le prix du produit ;
- *special* (numérique) : rabais appliqué sur le produit (en %) par rapport à son prix normal ;
- *distance_avg* (numérique) : distance moyenne entre le domicile de l'utilisateur et tous les magasins où le produit était disponible ;
- *disponibilité* (binaire) : disponibilité du produit dans différents magasins.

Nous avons complété cette liste de variables par une autre, nommée *total_bought*, qui représente le nombre total de fois qu'un produit donné a été acheté par tous les utilisateurs considéré.

Par ailleurs, la normalisation des données est une pratique courante et une étape

importante dans l'apprentissage automatique non supervisé et supervisé (Kotsiantis *et al.*, 2006), et l'exploration de données (García *et al.*, 2015). La normalisation des données s'est avérée théoriquement et empiriquement être une étape essentielle pour obtenir de meilleures prédictions à partir d'un modèle (Jain *et al.*, 2018; Pan *et al.*, 2016; Singh et Sharma, 2018). La normalisation permet d'amener toutes les variables sur une même échelle, les rendant mutuellement comparables et assurant ainsi un processus d'apprentissage plus stable et fournissant de meilleurs résultats pour les méthodes de partitionnement et d'apprentissage supervisé. Avant de fournir les données en entrée à nos modèles, nous avons également appliqué une méthode de standardisation à notre variable continue (c'est-à-dire la catégorie de produit), en la convertissant en un vecteur numérique. Nous avons utilisé la classe *feature_hasher* de scikit-learn (Pedregosa *et al.*, 2011; Buitinck *et al.*, 2013) pour encoder la variable *category*. La fonction principale de cette classe prend en entrée des chaînes de caractères et les convertit en vecteurs numériques à l'aide d'une fonction de hachage.

Dans notre étude, nous avons utilisé deux techniques de normalisation de données populaires : le z-score et la normalisation MinMax (Han *et al.*, 2011).

La normalisation dite *z-score* est une remise à l'échelle des données, de sorte que les données normalisées aient une moyenne de 0 et un écart type de 1 (Eq. 3.1) :

$$z(x_f) = \frac{x_f - \mu_f}{\sigma_f}, \tag{3.1}$$

où $z(x_f)$ est la valeur normalisée, x_f est la valeur originale observée de l'attribut f à une observation donnée, μ_f est la moyenne de f, et σ_f est la écart type de f.

La normalisation MinMax est effectuée à l'aide de la formule suivante (Eq. 3.2) :

$$x'_f = \frac{x_f - min(x_f)}{max(x_f) - min(x_f)}, \tag{3.2}$$

où x'_f est la valeur normalisée et x_f est la valeur d'origine observée de l'attribut f à une observation donnée, $min(x_f)$ est la valeur minimale de l'attribut f sur toutes les observations, et $max(x_f)$ est la valeur maximale de l'attribut f sur toutes les observations.

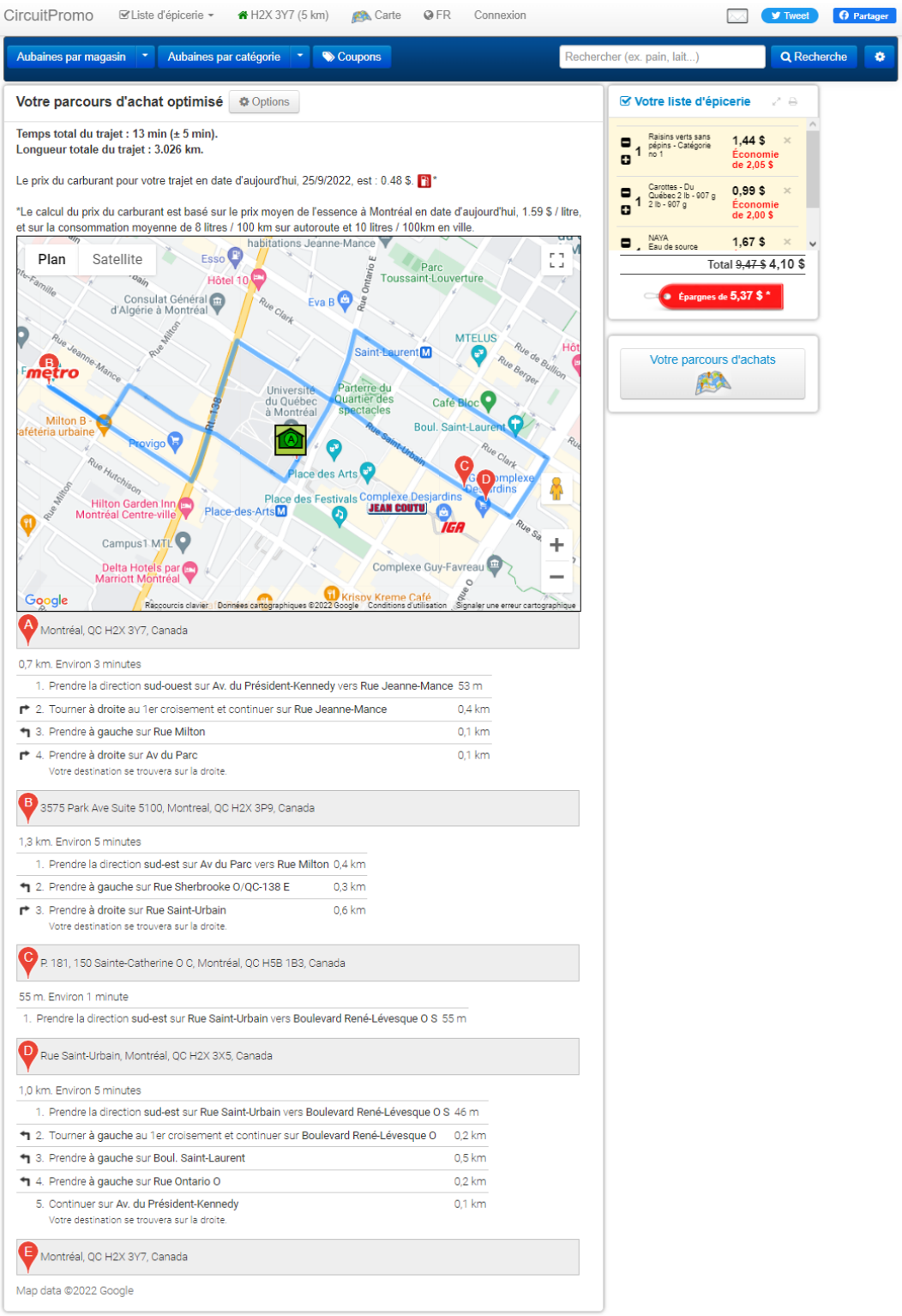

FIGURE 3.3 – Un parcours d'achat optimal en temps réel, basé sur la liste de courses hebdomadaire de l'utilisateur, affiché sur la plateforme CircuitPromo.

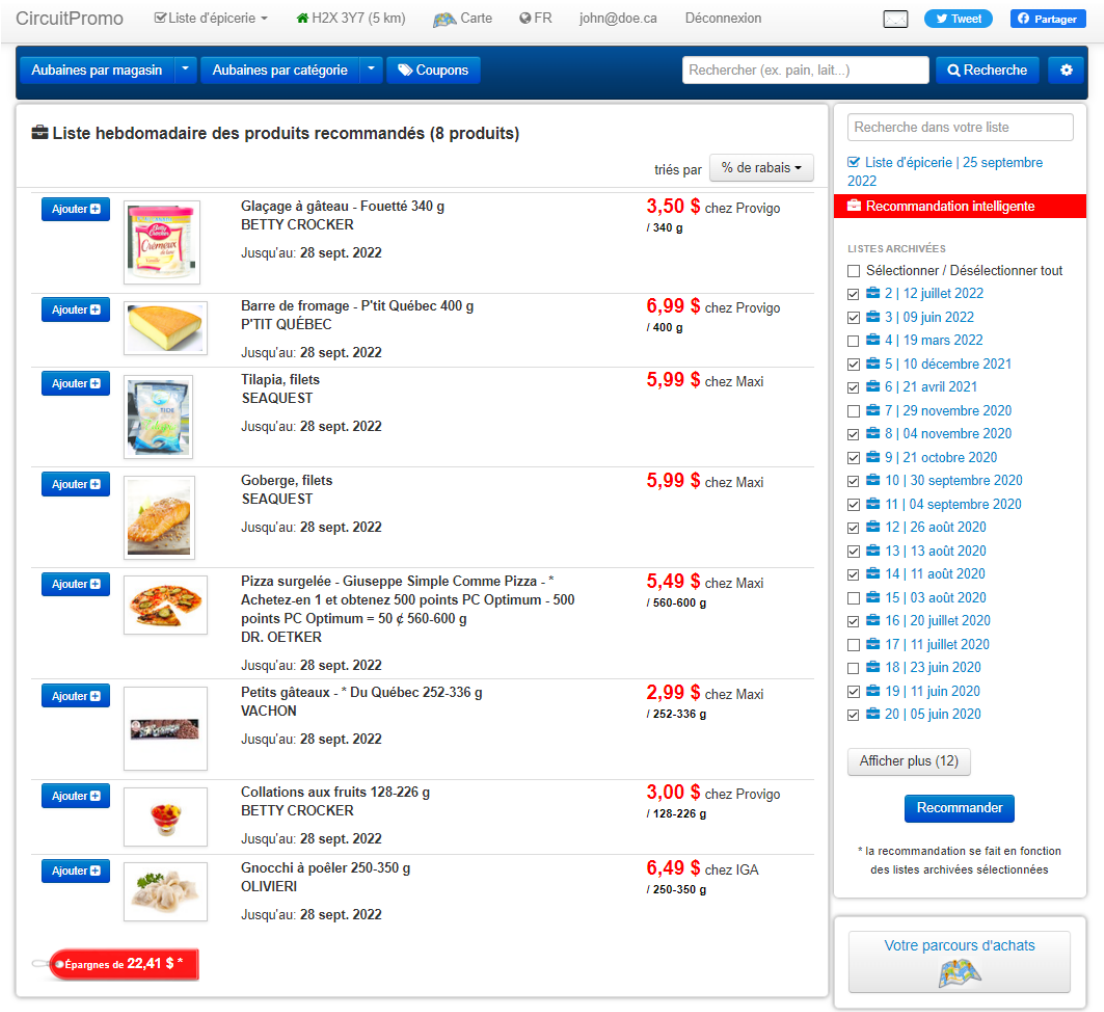

FIGURE 3.4 – Capture d'écran de l'interface du système de recommandation de CircuitPromo.

CHAPITRE IV

MÉTHODOLOGIE

4.1 Méthodes de partionnement

Le partitionnement (*clustering*) fait partie du domaine de l'analyse des données visant à trouver des groupes d'objets homogènes au sein des données. Les algorithmes de partitionnement sont divisés en fonction des formats de données d'entrée et des formats de structure des groupes de sortie. Un format de données générique est ce qu'on appelle la matrice objet-attribut $X = (x_{if})$, dans laquelle les lignes x_i $(i = 1, 2, ..., N)$ correspondent à des objets donnés (des clients dans notre cas) et les colonnes f $(f = 1, ..., F)$ correspondent à des attributs (caractéristiques) caractérisant ces objets (par exemple, le prix du produit, le rabais appliqué au produit s'il est en spécial ou encore la catégorie dans notre cas). Un format générique de structure de groupe (*cluster)* représente une partie de l'ensemble des objets rassemblés dans des groupes non superposés $S_1, S_2, ..., S_K$. Le nombre de groupes K doit être supérieur ou égal à 2, mais sans être trop élevé, de sorte que $K << N$ et que les groupes soient généralement des représentations agrégées de la matrice de données X.

Deux méthodes de regroupement de données, les algorithmes K-means (MacQueen *et al.*, 1967) (ou *K-moyennes*) et Ward (Ward Jr, 1963), ont été appliquées pour notre étude.

La structure du groupe dans l'algorithme de *K-moyennes* (MacQueen *et al.*, 1967; de Amorim et Makarenkov, 2016) est spécifiée par une partition S de l'ensemble d'objets en K groupes non superposés, $S = \{S_1, S_2, ..., S_K\}$. Chaque partition S est caractérisée par la liste des objets appartenant à chacun de ses groupes S_k $(k = 1, ..., K)$ et les centroïdes de ces derniers $c_k = (c_1, c_2, ..., c_K)$. L'enjeu dans ce cas de figure est alors de trouver une partition $S = \{S_1, S_2, ..., S_K\}$ et des centroïdes de groupes $c_k = (c_1, c_2, ..., c_K)$ qui minimisent le critère de la somme des carrés. L'algorithme *K-moyennes* suit le schéma dit de minimisation alternée pour trouver une partition de K groupes qui minimise le critère (4.1) :

$$W(S, c) = \sum_{k=1}^{K} \sum_{x_i \in S_k} \sum_{f=1}^{F} (x_{if} - c_{kf})^2, \tag{4.1}$$

où x_{if} est la valeur de l'attribut f pour l'objet x_i, et c_{kf} est la valeur de l'attribut f au centroïde c_k.

Partant d'une partition initiale aléatoire et d'un ensemble de centroïdes c, l'algorithme essaie de trouver une partition optimale S qui minimise la somme des carrés $W(S, c)$ pour un c donné, puis trouve le vecteur c' qui minimise $W(S, c)$. La procédure est répétée jusqu'à convergence, c'est-à-dire jusqu'à ce que c' coïncide avec c. En pratique, la méthode converge rapidement vers un minimum local qui dépend beaucoup du choix de la partition de départ.

L'*algorithme de partitionnement Ward* (Ward Jr, 1963; Cordeiro de Amorim *et al.*, 2016) suit l'approche hiérarchique dite agglomérative. À chaque étape, cet algorithme considère une partition courante $S = \{S_1, S_2, ..., S_K\}$ avec K groupes et leurs centres $c = \{c_1, c_2, ..., c_K\}$, et fusionne deux groupes, S_k et S_l, en un nouveau groupe $S_{kl} = S_k \cup S_l$, avec pour centre $c(k, l) = (N_k c_k + N_l c_l)/(N_k + N_k)$, où N_k et N_l sont respectivement les cardinalités des groupes S_k et S_l. Les groupes

à fusionner sont sélectionnés pour que l'augmentation de la valeur de $\Delta(k, l)$ (Eq. 4.2) atteigne son minimum sur tous les k et l (avec $k \neq l$) :

$$\Delta(k, l) = W(S(k, l), c(k, l)) - W(S, c), \tag{4.2}$$

où $S(k, l)$ désigne la nouvelle partition avec $m - 1$ groupes obtenus à partir de S en fusionnant S_k et S_l (c'est-à-dire $S_{kl} = S_k \cup S_l$), et $c(k, l)$ désigne le centroïde de cette nouvelle partition. Les quantités $\Delta(k, l)$ sont toutes positives car la valeur du critère (4.1) diminue à mesure que le nombre de groupes K augmente, de sorte qu'il devient nul à $K = N$. Il est ensuite facile d'obtenir la formule suivante exprimant explicitement $\Delta(k, l)$ par le biais des groupes fusionnés :

$$\Delta(k, l) = \frac{N_k N_l}{N_k + N_l} d(c_k, c_l), \tag{4.3}$$

où $d(c_k, c_l)$ est la distance euclidienne entre les centroïdes c_k et c_l. Cette formule montre que le critère d'erreur quadratique moyenne tend à fusionner les groupes dont les centres sont les plus proches et dont les tailles sont les plus déséquilibrées. L'algorithme générique de partitionnement Ward commence par une partition triviale composée de tous les singletons (chacun étant son propre centre), puis fusionne entre eux, un par un, les groupes avec la distance Ward la plus faible (4.3), jusqu'à ce que tous les objets tombent dans un groupe unique les comprenant tous.

4.2 Algorithmes d'apprentissage automatique supervisé et profond

Dans cette section, nous présentons les principales caractéristiques des algorithmes d'apprentissage automatique traditionnels supervisés et d'apprentissage profond utilisés et comparés dans nos travaux. Leurs implémentations scikit-learn et PyTorch ont été utilisées pour nos expériences de calcul. Les résultats obtenus sont présentés dans le chapitre 5, intitulé « résultats et discussion ». Il est important de noter que tous les algorithmes d'apprentissage ont été appliqués de manière personnalisée, c'est-à-dire qu'un modèle d'apprentissage automatique ou profond distinct a été construit pour chacun des 831 utilisateurs réels pris en compte dans nos expériences.

Decision Trees (DT) : Les arbres de décision sont des modèles hiérarchiques basés sur une succession de règles de décision simples (Breiman *et al.*, 1984). Chaque arbre de décision comprend une racine, des nœuds, des branches et des feuilles. Chaque nœud correspond à un test pour un attribut donné, tandis que les branches représentent le résultat de ce test. Une décision est prise en atteignant une feuille qui correspond à la classe prédite. Les règles de décision sont déduites sur la base des données d'apprentissage et des caractéristiques (donc les variables ou encore attributs). Une approche populaire pour construire un arbre de décision est la minimisation des impuretés à chaque nœud basée sur la mesure d'impureté de Gini (4.4) qui vise à réduire la probabilité de faire des erreurs lors de la classification. La mesure d'impureté de Gini est définie comme suit :

$$Gini(Z) = 1 - \sum_{k=1}^{K} {P_k}^2, \tag{4.4}$$

où Z est un ensemble d'apprentissage contenant K classes, k est une classe donnée, et P_k est la proportion d'objets appartenant à la classe k.

Random Forest (RF) : les forêts d'arbres décisionnels constituent une classe d'algorithmes d'apprentissage par ensemble qui fonctionne en construisant une multitude d'arbres de décision lors de la phase d'entraînement. Pour les tâches de classification, la classe sélectionnée par la majorité des arbres est donnée en sortie (Breiman, 2001). Plus précisément, chaque arbre de décision est construit sur un sous-échantillon de l'ensemble d'apprentissage, suivant un méta-algorithme appelé *bootstrap aggregation* qui vise à minimiser la variance et à éviter le surentraînement du modèle. Les arbres de décision sont particulièrement sensibles aux données sur lesquelles ils sont construits. Des modifications mineurs apportées à l'ensemble de données utilisé lors de l'entraînement peuvent avoir pour conséquence des structures arborescentes très différentes. L'algorithme Random Forest tire profit de ce phénomène en permettant à chaque arbre de décision d'utiliser un échantillon pris au hasard depuis l'ensemble de données avec remplacement (*bagging*). Il en résulte alors des arbres différents. La décision finale pour une observation est prise sur la base d'un vote majoritaire entre les résultats de tous les arbres de décision. L'étape consistant à combiner tous ces résultats et à générer la sortie correspond à l'agrégation. Les principaux avantages de l'algorithme Random Forest sont qu'il est connu pour être résistant aux potentielles valeurs aberrantes (*outliers*) ainsi que pour être facilement parallélisable.

Gradient Boosting Trees (GBT) : les arbres basées sur le boosting de gradient représentent une méthode d'apprentissage par ensemble utilisant des arbres de décision en guise d'apprenants faibles (*weak learners*) en association avec l'optimisation de la descente de gradient (similaire aux réseaux de neurones) pour obtenir la meilleure solution aux problèmes de classification ou de régression (Friedman, 2001; Friedman, 2002). Contrairement à Random Forest, qui repose sur le bagging, l'algorithme GBT est basé, comme son nom l'indique, sur le *boosting*. L'algorithme GBT étant itératif, il tente de minimiser la fonction de perte en ajustant

séquentiellement un nouvel arbre à chaque étape et en corrigeant l'erreur de prédiction des étapes précédentes. Il existe différentes implémentations de GBT, et certaines d'entre elles fonctionnent souvent mieux que d'autres dans la pratique. Dans cette étude, nous avons également utilisé les implémentations scikit-learn de l'algorithmes GBT nommées XGBoost et Catboost (Chen et Guestrin, 2016; Prokhorenkova *et al.*, 2018b; Dorogush *et al.*, 2018). Des travaux antérieurs au sein de la littérature relative à l'apprentissage automatique classique et à l'apprentissage profond ont montré que les méthodes d'apprentissage par ensemble (boosting et bagging) de plusieurs apprenants faibles peuvent considérablement améliorer les performances de l'algorithme de base. De plus, le boosting a tendance à surpasser le bagging sur des ensembles de données qui contiennent une couverture de données inégale, d'où notre choix concernant les algorithmes GBT classique, XGBoost et CatBoost.

Naive Bayes (NB) : la classification bayesienne naïve est la forme la plus simple d'un réseau bayésien. Cette approche probabiliste est basée sur le théorème de Bayes (4.5), défini comme suit :

$$P(y|X) = \frac{P(y)P(X|y)}{P(X)}, \tag{4.5}$$

où y est la classe et X est l'ensemble des caractéristiques. Toutefois, l'un des principaux inconvénients de Naive Bayes est qu'il suppose que toutes les caractéristiques considérées sont indépendantes, ce qui se produit rarement dans des scénarios réels. Néanmoins, Naive Bayes est connu pour fournir des résultats compétitifs dans certains cas, en particulier dans la détection de spam et dans l'analyse des sentiments (Zhang, 2005; Diab et El Hindi, 2017).

Support Vector Machines (SVM) : une machine à vecteurs de support est un type d'algorithmes qui tente de séparer l'ensemble de données considéré à l'aide d'une droite appelée *hyperplan*. Alors qu'une infinité d'hyperplans différents

peuvent exister pour cette tâche, SVM choisit celui qui maximise la marge entre les observations représentatives appartenant à chaque classe. Ces observations sont appelées vecteurs de support (Cortes et Vapnik, 1995). SVM introduit le concept de marges souples (*soft margins*) pour traiter les valeurs aberrantes ou les données non linéaires, permettant ainsi de choisir un hyperplan tout en autorisant quelques erreurs afin d'obtenir une meilleure séparation finale (Noble, 2006). Cependant, les données ne sont pas souvent linéairement séparables, même lorsque des marges souples sont utilisées. Dans ce cas, il est possible de transformer les données en considérant un espace de dimension supérieure, ce qui permet une meilleure séparation des classes. Ceci est réalisable grâce à l'utilisation de fonctions noyau (*kernel functions*) telles que la fonction de base radiale (*rbf*, pour *radial basis function*) définie comme suit :

$$K(x_i, x_j) = exp(-\gamma||x_i - x_j||^2), \tag{4.6}$$

où γ est le coefficient de la fonction noyau défini au préalable par l'utilisateur (Bergman, 1970; Claesen *et al.*, 2014). Le choix de la fonction noyau la plus appropriée est généralement effectué en se basant sur des expériences itératives et donc par une approche empirique reposant sur des essais et des erreurs.

Régression logistique : la régression logistique est un modèle de classification simple utilisant une fonction logistique (Eq. 4.7) pour modéliser la probabilité de tous les résultats d'un seul essai (McFadden *et al.*, 1973). Elle est généralement de la forme suivante :

$$f(x) = \frac{1}{1 + e^{-(x-\mu)/s}} \tag{4.7}$$

où μ est un paramètre de localisation et s est un paramètre d'échelle proportionnel à la variance.

Perceptron multi-couches (MLP, pour multilayer perceptron) : le perceptron est un classificateur binaire et le type le plus simple de réseau de neurones (Rosenblatt, 1961). La classification d'un perceptron est obtenue en calculant le produit scalaire des données d'entrée $(x_1, x_2, ..., x_n)$ et des poids $(w_1, w_2, ..., w_n)$, et en ajoutant un biais b au résultat. La somme pondérée obtenue est par la suite passée à une fonction d'activation (aussi appelée fonction de transfert ou de seuillage) qui fournit la classification finale (4.8) :

$$f(\mathbf{x}) = \begin{cases} 1, & \text{si } \ \mathbf{w} \cdot \mathbf{x} + b \geq 0, \\ 0, & \text{sinon.} \end{cases} \tag{4.8}$$

La phase d'apprentissage du perceptron repose sur le fait de trouver les valeurs optimales des poids selon un processus itératif de comparaison de la sortie attendue y à la sortie prédite y'. Ce processus est répété jusqu'à ce que l'algorithme converge sur l'ensemble des données considérées. Différentes fonctions d'activation existent et leur utilisation va principalement dépendre du contexte et du problème à résoudre.

Un perceptron multi-couches est un réseau de neurones artificiel possédant une couche d'entrée, une couche cachée ainsi qu'une couche de sortie. Ce réseau est composé de neurones interconnectés qui sont des perceptrons. De ce fait, alors qu'un perceptron simple n'est capable d'effectuer qu'une classification binaire associée à une séparation linéaire des données, un perceptron multi-couches est capable de capturer des relations complexes et d'effectuer une classification multi-classes. Les données sont transmises via la couche d'entrée, puis traitées via la couche cachée, avant que la couche de sortie ne donne le résultat final. La phase d'apprentissage d'un MLP est similaire à celle d'un perceptron simple : c'est également un processus itératif visant à trouver le vecteur optimal de poids $\mathbf{w}$ en comparant la classe prédite y' avec la classe réelle y . Cependant, compte tenu

de sa nature plus sophistiquée et de la présence d'une couche cachée, le MLP s'appuie sur la rétro-propagation pour gérer les erreurs pendant la phase d'apprentissage (Rumelhart *et al.*, 1985).

Modèle RNN-GRU proposé : un réseau de neurones récurrents (RNN) est un réseau d'apprentissage profond conçu pour intégrer des données séquentielles dépendant du temps. Dans cette étude, nous avons utilisé une architecture basée sur un réseau de neurones récurrents à portes (GRU) afin de représenter les paniers des utilisateurs. Précisément, nous avons généralisé le modèle DREAM (Dynamic REcurrent bAsket Model) proposé par Yu et ses collaborateurs pour prédire le contenu du prochain panier. De plus, nous avons utilisé certaines variables supplémentaires telles que les prix des produits, la disponibilité des produits et les promotions hebdomadaires proposées dans les magasins locaux. Ces attributs n'ont pas été pris en compte par Yu et ses collaborateurs. Nous avons également appliqué quelques modifications importantes au modèle DREAM original dans le but de l'adapter à une classification multi-classes, puisque seul un problème de classification binaire est considéré dans (Yu *et al.*, 2016). Nous avons ainsi intégré à travers PyTorch chaque produit disponible à l'aide d'une couche d'intégration (*Embedding layer*). Le produit considéré est ensuite concaténé avec le reste des variables passées par un perceptron à deux couches. Ainsi, chaque produit est représenté par un vecteur augmenté (voir Fig. 4.1 pour un aperçu schématisé de l'architecture du modèle proposé).

Plus spécifiquement, notre architecture RNN contient deux couches GRU de 64 neurones chacune. L'optimisation des paramètres a été effectuée par l'optimiseur *RMSProp* avec PyTorch. Nous avons sélectionné le taux d'apprentissage optimal en utilisant une validation croisée à 5 blocs. Pour éviter le sur-apprentissage, nous avons entraîné le modèle avec un drop-out de 0,1.

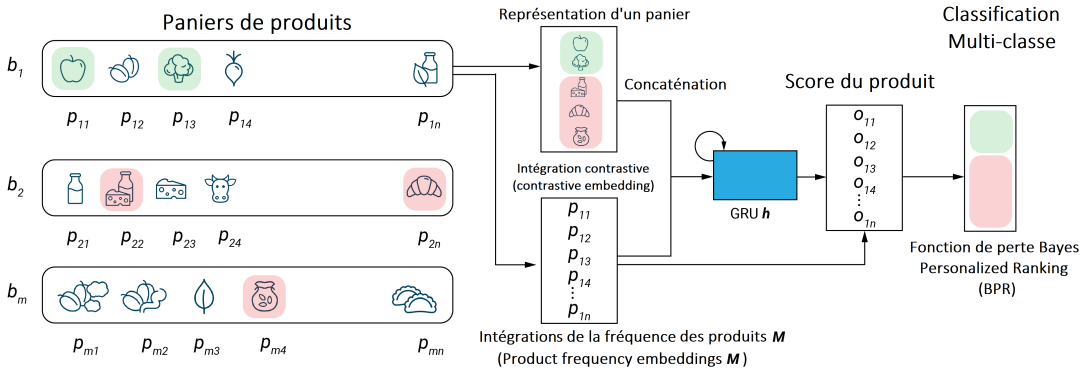

FIGURE 4.1 – Architecture de notre modèle DREAM RNN-GRU étendu pour la prédiction du prochain panier dans un contexte multi-classe.

Étant donné que dans ce travail nous étudions les avantages induits par l'entraînement d'un seul modèle par utilisateur, aucune intégration explicite d'utilisateur n'a besoin d'être construite. Chaque panier b_i est traité comme une permutation arbitraire de produits $b_i = \{p_{i,1}, .., p_{i,n}\}$ encodée dans l'espace augmenté des caractéristiques, qui est ensuite condensé par les cellules GRU en $h_{i,t}$. L'intégration cachée (*hidden embedding*) $h_{i,t}$ agit comme une représentation implicite de l'utilisateur, car elle stocke des informations concernant les paniers de l'utilisateur.

Pour obtenir le score d'affinité du produit, c'est-à-dire le score proportionnel à la probabilité que le produit $p_{i,t}$ soit inclus dans un panier contenant les produits $p_{i,1}, ..p_{i,t-1}, p_{i,t+1}, .., p_{i,n}$, nous multiplions la matrice d'intégration du produit M par l'intégration cachée $h_{i,t}$:

$$o_{i,t} = M^T h_{i,t} \tag{4.9}$$

Un score $o_{i,t}$ plus élevé indique que l'utilisateur est plus susceptible d'acheter l'article correspondant. La fonction de perte appelée « Bayesian Personalized Ranking » (BPR) est utilisée pour approximer et maximiser la probabilité suivante :

$$p(b_i, v \succ v') = \sigma(o_{i,v} - o_{i,v'}), \tag{4.10}$$

où v désigne un élément positif inclus dans le panier b_i, v' désigne un élément négatif non inclus dans le panier b_i, et $\sigma(x)$ est la fonction d'activation logistique à faire correspondre sur l'espace de probabilités. Pour mettre en œuvre cet objectif, nous avons échantillonné un certain nombre d'éléments négatifs qui étaient absents dans le panier considéré b_i (lors nos expérience, le nombre d'éléments négatifs était égal au nombre d'éléments positifs dans b_i), et avons maximisé la probabilité liée à l'espérance mathématique (Eq. 4.11) sur tous les produits et paniers :

$$\ell_{RNN} = \mathbb{E}_{b_i}[\mathbb{E}_{v,v'}[p(b_i, v \succ v')]]. \tag{4.11}$$

Pour traiter des données séquentielles, comme dans le cas de la recommandation d'un panier de consommation, il est courant d'utiliser des mécanismes de prédiction récurrents (par exemple un LSTM, un GRU ou un RNN classique). Bien que d'autres alternatives existent, elles ont soit tendance à sous-performer dans les tâches de recommandation (par exemple, les convolutions causales à une dimension), soit nécessitent une grande quantité de données de bonne qualité (par exemple, la couche d'auto-attention dans le cas d'un transformateur). Étant donné que nos données ne nécessitent pas la prédiction de séquences extrêmement longues, l'utilisation d'une cellule GRU est un choix de conception approprié qui équilibre les performances de prédictions et la vitesse d'entraînement.

4.3 Optimisation des paramètres et validation croisée

L'optimisation des paramètres est une étape décisive lors de la construction d'un modèle d'apprentissage automatique, car la plupart des modèles dépendent fortement des paramètres sélectionnés et offrent des performances nettement meilleures lorsqu'ils sont correctement optimisés. Ignorer l'optimisation des paramètres peut conduire à la sélection d'une solution sous-optimale à l'issue de l'expérimentation.

Il existe plusieurs méthodes pour optimiser les paramètres du modèle telles que Grid Search, Random Search ou l'optimisation bayésienne (Bergstra et Bengio, 2012; Hutter *et al.*, 2011).

Grid Search utilise une grille de paramètres et effectue des tests exhaustifs avec toutes les combinaisons de paramètres afin de sélectionner finalement celle qui donne les meilleurs résultats pour les données disponibles. Dans cette étude, nous avons utilisé Random Search comme technique d'optimisation des paramètres pour les modèles répertoriés dans la sous-section 4.2 de ce même chapitre. Comme pour Grid Search, Random Search considère une grille de paramètres et de valeurs. Cependant, Random Search effectue des essais sur des combinaisons aléatoires de paramètres au lieu d'effectuer une recherche exhaustive. Cela permet d'utiliser des distributions au lieu de valeurs spécifiques pour les paramètres continus et assure une meilleure gestion du temps et des ressources (le temps d'exécution n'est pas nécessairement lié à la quantité de paramètres/valeurs car le nombre de combinaisons de paramètres à tester peut être fixé par l'utilisateur). Il a été démontré que cette approche surpasse celle de Grid Search en termes de résultats et de temps d'exécution (Bergstra et Bengio, 2012).

Lors de nos expérimentation avec Random Search, nous avons effectué une validation croisée (k-fold) pour nous assurer que le meilleur modèle sélectionné ne

surajuste pas les données (Larson, 1931; Stone, 1974). La validation croisée k-fold est une technique de validation de modèle courante en apprentissage automatique qui consiste à diviser les données en k sous-échantillons de taille égale. Un seul sous-échantillon est ensuite conservé à des fins de validation car le modèle est entraîné sur le reste des données (c'est-à-dire sur les sous-échantillons $k-1$ restants). Ce processus est répété k fois en utilisant chaque sous-échantillon exactement une fois pour validation. L'évaluation finale du modèle est la moyenne des résultats k. Dans nos expériences, nous avons fixé la valeur de k à 5 (qui est un nombre de sous-échantillons couramment utilisé) (McLachlan *et al.*, 2005). En utilisant cette méthodologie, nous avons pu optimiser les paramètres des algorithmes d'apprentissage automatiques décrits dans la sous-section 4.2, en veillant à ce que les résultats présentés dans les tableaux 5.1, 5.2 et 5.3 ne soient pas dus à un surajustement des données.

4.4 Système de recommandation de CircuitPromo

Afin de déterminer les produits à recommander à un utilisateur donné par le système de recommandation CircuitPromo, nous classons tous les produits en fonction de l'historique d'achat de l'utilisateur, des informations sur les promotions en cours et de la disponibilité des produits dans chacune des épiceries locales considérées. Afin de pouvoir classer efficacement les produits, l'utilisation de feedbacks positifs et négatifs est nécessaire. Un modèle d'apprentissage automatique ou profond personnalisé (c'est-à-dire un modèle par utilisateur) est créé et mis à jour chaque semaine dans notre système.

Alors que nous considérons les produits achetés par un utilisateur donné comme un feedback positif, nous considérons tous les produits qui étaient disponibles pour cet utilisateur au moment de la commande mais qu'il n'a pas acheté comme étant un feedback négatif. Pour une commande de taille S, si T est le nombre total de produits disponibles pour l'utilisateur au moment de la commande, alors le feedback négatif N pour cette commande est $N = T - S$.

En règle générale, N représente des milliers de produits, tandis que S varie généralement de 5 à 50. Cette différence de taille entre les feedback positifs et négatifs conduit à des données d'entraînement déséquilibrées et peut entraîner une perte importante de performances. Comme pour les travaux de (Xia *et al.*, 2017), nous avons décidé d'utiliser une méthode de sous-échantillonnage pour équilibrer les données de l'utilisateur au lieu de considérer tous les éléments disponibles et ignorés comme feedback négatif. Les méthodes de sous-échantillonnage se sont avérées efficaces pour les classifications binaires et multi-classes (Hasanin et Khoshgoftaar, 2018; Arafat *et al.*, 2017).

Par ailleurs, le nombre de produits recommandés par les modèles d'apprentissage

est souvent supérieur à la taille moyenne d'une liste d'épicerie, S_u, calculée pour un utilisateur donné u, pour la recommandation finale. Afin de pallier à cette éventualité, seuls les articles de cette liste S_u avec les scores de confiance les plus élevés sont retenus pour notre approche. Ce score de confiance est calculé comme étant la probabilité de la classe prédite pour une observation donnée ; il peut par exemple être obtenu en utilisant la fonction *predict_proba* de la bibliothèque scikit-learn.

CHAPITRE V

RÉSULTATS ET DISCUSSION

5.1 Analyse du partitionnement

5.1.1 Caractéristiques considérées

Comme mentionné plus tôt dans le mémoire, nous avons d'abord utilisé le partitionnement pour identifier les profils des utilisateurs de CircuitPromo. Pour ce faire, nous avons considéré les caractéristiques suivantes :

- *avg_price* (numérique) : le prix moyen des produits achetés par un utilisateur spécifique ;
- *avg_special* (numérique) : pourcentage de remise moyen sur les produits achetés par un utilisateur spécifique ;
- *avg_list_size* (numérique) : la taille moyenne de la liste de courses d'un utilisateur spécifique ;
- *pca_category* (numérique) : cette caractéristique tient compte de la catégorie de produits sélectionnés par un utilisateur spécifique. Ici, nous avons construit une matrice 831×24 (831 étant le nombre d'utilisateurs et 24, le nombre de catégories disponibles) reflétant le choix de l'utilisateur parmi les différentes catégories de produits. Chaque valeur de cette matrice représente le nombre d'éléments d'une catégorie donnée achetés par un uti-

lisateur spécifique. Nous avons effectué l'analyse en composantes principales (*PCA*, pour *Principal Component Analysis*) afin de réduire la dimension matricielle et déterminer le pourcentage de variance représenté par les grands axes principaux. Le premier axe PCA (principal) représentait 72,6% de la variance totale et le deuxième axe 12,1%, alors que la variance expliquée par les axes restants était négligeable. Ainsi, nous avons décidé de conserver pour notre analyse du partitionnement une seule caractéristique transformée représentant la catégorie du produit. Elle correspond aux valeurs normalisées du premier axe principal ;

— *avg_fidelity_ratio* (numérique) : la moyenne du ratio de fidélité basé sur la quantité (RFQ) et du ratio de fidélité basé sur le prix (RFP) définis respectivement dans les équations (5.1) et (5.2). Ici, $avg\_fr_u = (RFQ_u + RFP_u)/2$, où u est un utilisateur donné, avg_fr_u est le ratio de fidélité moyen, RFQ est le ratio de fidélité basé sur la quantité et RFP est le ratio de fidélité basé sur le prix.

Le ratio de fidélité basé sur la quantité (RFQ) et le ratio de fidélité basé sur le prix (RFP) définis ci-dessous sont tous deux destinés à donner un aperçu de la fidélité du client à son magasin préféré.

Une valeur RFQ proche de 1 indique qu'un consommateur donné a tendance à faire ses courses dans le même magasin (son préféré), alors qu'une valeur RFQ proche de 0 indique que le client a tendance à faire ses courses dans plusieurs magasins différents, sans préférence particulière. Le ratio de fidélité basé sur la quantité est défini comme suit :

$$RFQ_u = \begin{cases} \dfrac{X_{max,u}}{X_{total,u}} = 1, & \text{si } n = 1 \\ \dfrac{X_{max,u} - \frac{1}{(n-1)} \sum_{i=2}^{n} X_{i,u}}{X_{total,u}}, & \text{si } n > 1 \end{cases} \tag{5.1}$$

où u représente un utilisateur donné, n est le nombre total de magasins où l'utilisateur u ($n \in \mathbb{N}^*$) a acheté au moins un produit, $X_{max,u}$ est le nombre total de produits achetés par l'utilisateur u dans son magasin préféré (c'est-à-dire là où il a effectué la plupart de ses achats), et $X_{total,u}$ ($X_{total,u} = X_{max,u} + \sum_{i=2}^{n} X_{i,u}$) est le nombre total de produits achetés par l'utilisateur u sur l'ensemble des magasins où il a acheté au moins un produit.

De même, le ratio de fidélité basé sur le prix (RFP) dépend du prix total des produits achetés par le client dans son magasin préféré. Le ratio de fidélité basé sur les prix est défini comme suit :

$$RFP_u = \begin{cases} \dfrac{P_{max,u}}{P_{total,u}}, & \text{si } n = 1 \\ \dfrac{P_{max,u} - \frac{1}{(n-1)} \sum_{i=2}^{n} P_{i,u}}{P_{total,u}}, & \text{si } n > 1 \end{cases} \tag{5.2}$$

où u représente un utilisateur donné, n est le nombre total de magasins où l'utilisateur u ($n \in \mathbb{N}^*$) a acheté au moins un produit, $P_{max,u}$ est le prix total de tous les produits achetés par l'utilisateur u dans son magasin préféré, et $P_{total,u}$ ($P_{total,u} = P_{max,u} + \sum_{i=2}^{n} P_{i,u}$) est le prix total de tous les produits achetés par l'utilisateur u dans tous les magasins où il a acheté au moins un produit.

5.1.2 Nombre optimal de groupes pour le partitionnement

Les données d'entrée pour l'étape de partitionnement sont représentées dans notre étude par une matrice de 831 observations (correspondant à l'ensemble des 831 utilisateurs provenant de la plateforme CircuitPromo) et 5 caractéristiques. Avant d'effectuer le partitionnement, nous avons normalisé les données disponibles. Nous avons testé les normalisations Z-score et MinMax. Les résultats présentés ci-dessous ont été obtenus en utilisant la normalisation MinMax car elle a fourni des résultats de partitionnement légèrement meilleurs. L'étape de partitionnement a été effectuée en utilisant à la fois l'algorithme Ward (Ward Jr, 1963), qui est l'un des algorithmes de partitionnement hiérarchiques les plus populaires, et K-means (MacQueen *et al.*, 1967), qui est l'un des algorithmes de partitionnement les plus populaire. Ces deux algorithmes ont été utilisés à travers leur implémentation scikit-learn. Les paramètres scikit-learn par défaut pour ces deux algorithmes ont été maintenus.

Afin de déterminer le nombre de groupes optimal pour notre ensemble de données, nous avons utilisé deux indices de validité des groupes populaire : Silhouette (Rousseeuw, 1987) et Davies-Bouldin (DB) (Davies et Bouldin, 1979).

Le coefficient de Silhouette est défini comme suit. Étant donné une partition P d'un ensemble de données X avec des objets N, le coefficient de silhouette $s(x_i)$, pour l'objet $x_i \in X$, représente le degré de correspondance entre x_i et la partition. La distance moyenne entre l'objet x_i et son groupe C_k peut être définie comme suit :

$$a(i) = \frac{1}{|C_k|} \sum_{j \in C_k} \sqrt{d(x_i, x_j)}, \tag{5.3}$$

et la distance à un objet le plus proche dans un autre groupe comme suit :

$$b(i) = \min_{C_k : x_i \notin C_k} \Big\{ \frac{1}{|C_k|} \sum_{j \in P_k} \sqrt{d(x_i, x_j)} \Big\}. \tag{5.4}$$

Le coefficient de Silhouette d'un objet $s(x_i)$ est défini comme étant la différence relative entre $a(x_i)$ et $b(x_i)$:

$$s(x_i) = \frac{b(x_i) - a(x_i)}{\max\{a(x_i), b(x_i)\}}. \tag{5.5}$$

La valeur globale du coefficient de Silhouette est alors définie comme suit :

$$s(P) = \frac{1}{N} \sum_{x_i \in X} (x_i) \tag{5.6}$$

Elle représente l'étendue de la cohérence de la partition P. La valeur maximale de $s(P)$ correspond au "bon" nombre de groupes.

L'indice de Davies-Bouldin est la similarité moyenne entre chaque groupe C_i pour $i = 1, ..., k$ et son équivalent le plus similaire C_j. Il est calculé comme suit :

$$DB = \frac{1}{k} \sum_{i=1}^{k} \underset{i \neq j}{max}\ S_{ij}, \tag{5.7}$$

où S_{ij} est la valeur de similarité entre les groupes, calculée comme étant $(d_i + d_j)/\delta_{ij}$, où d_i et d_j sont respectivement les distances moyennes entre les objets composant les groupe C_i et C_j et les centroïdes de ces groupes. δ_{ij} est la distance entre les centroïdes des groupes C_i et C_j. La valeur minimale de l'index DB correspond au "bon" nombre de groupes.

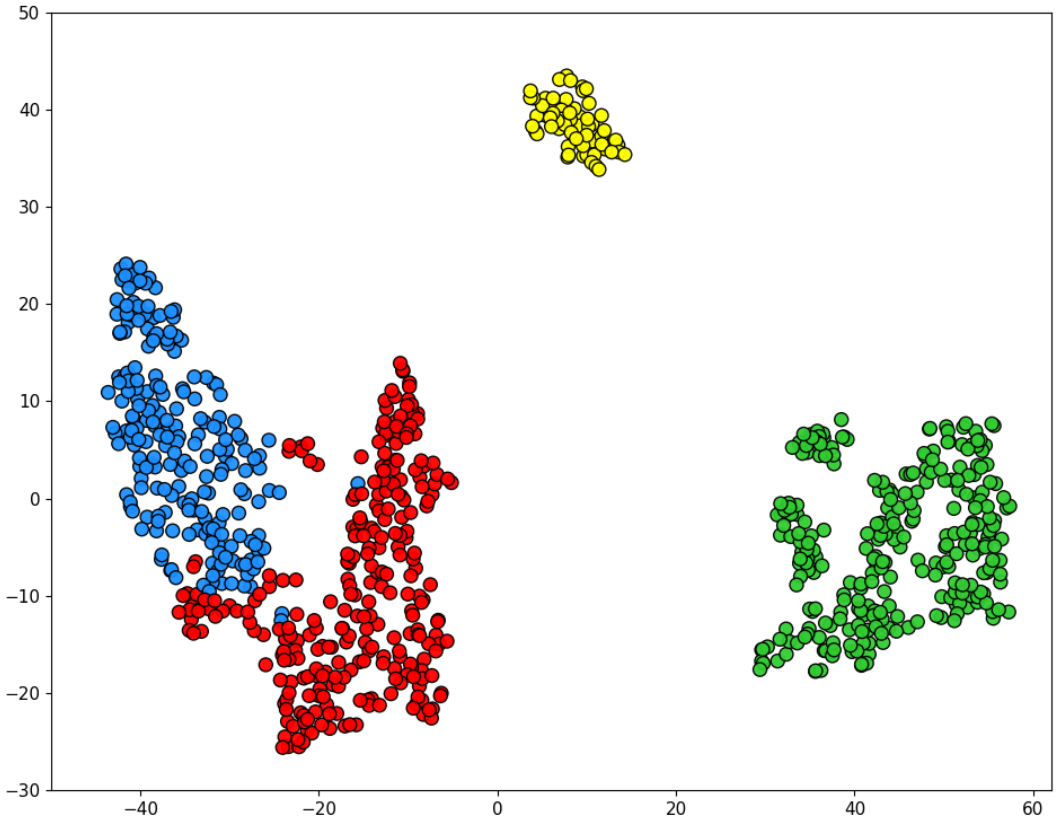

FIGURE 5.1 – Résultats du partitionnement : solution de partitionnement fournie par l'algorithme de Ward pour K=4 groupes (c'est-à-dire le meilleur nombre de groupes selon les indices de validité des groupes de Silhouette et de Davies-Bouldin). La réduction de la dimensionnalité après le partitionnement a été réalisée avec t-SNE (à des fins de visualisation). Les 4 groupes de clients trouvés par le partitionnement hiérarchique basé sur Ward sont représentés par des couleurs différentes (Groupe 1 de 278 utilisateurs - en rouge, Groupe 2 de 276 utilisateurs - en vert, Groupe 3 de 214 utilisateurs - en bleu, Groupe 4 de 63 utilisateurs - en jaune.

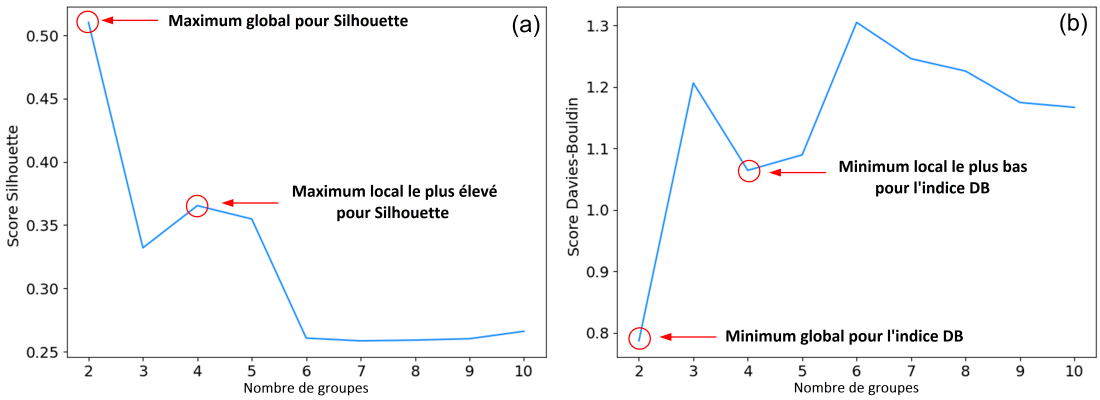

FIGURE 5.2 – Variation des scores de validité des groupes de Silhouette et de Davies-Bouldin en fonction du nombre de groupes.

Bien que la valeur la plus élevée du coefficient de Silhouette et la valeur la plus faible de l'indice de Davies-Bouldin ont été trouvées avec pour solution $K = 2$ groupes, nous présentons ici la solution la plus intéressante trouvée pour $K = 4$ groupes (voir Fig. 5.1). Cette solution correspond au maximum local le plus élevé de l'indice Silhouette et au minimum local le plus bas de l'indice de Davies-Bouldin (voir Fig. 5.2). Nous avons utilisé le t-distributed Stochastic Neighbor Embedding (tSNE) (Van der Maaten et Hinton, 2008) comme méthode de réduction de la dimensionnalité pour visualiser la solution de partitionnement fournie par l'algorithme de Ward (voir Fig. 5.1). Lors de nos expériences, nous avons utilisé des valeurs spécifiques pour certains paramètres de la méthode tSNE : une perplexité de 30, un taux d'apprentissage de 925 et un paramètre d'initialisation tSNE basé sur l'analyse en composantes principales (Makarenkov et Legendre, 1999) afin de préserver la forme générale des données.

Il convient de mentionner que la solution de partitionnement fournie par K-means (pour $K = 4$ groupes) était similaire, mais avait un chevauchement de groupes légèrement plus important, par rapport à celui trouvé par Ward. De ce fait, cette solution n'est pas présentée ici.

Les quatre profils d'utilisateurs illustrés à la Fig. 5.1 sont les suivants :

— Groupe 1 (en rouge sur la Fig. 5.1) : comprend les clients qui sont modérément sensibles aux offres spéciales et qui achètent généralement leurs produits d'épicerie dans le même magasin (c'est-à-dire qui ont des ratios de fidélité élevés) ;

— Groupe 2 (en vert sur la Fig. 5.1) : correspond au groupe le plus diversifié, composé de clients qui achètent leurs produits d'épicerie dans différents magasins (c'est-à-dire qui ont des ratios de fidélité faibles). Les membres de ce groupe sont généralement sensibles aux offres spéciales ;

— Groupe 3 (en bleu sur la Fig. 5.1) : comprend des clients qui achètent généralement les mêmes produits (ou des produits similaires) dans le même magasin (c'est-à-dire qui ont des taux de fidélité élevés), mais ne réagissant presque pas aux promotions ;

— Le groupe 4 (en jaune sur la Fig. 5.1) : comprend les clients qui sont très sensibles aux promotions et qui achètent leurs produits d'épicerie dans le même magasin (c'est-à-dire qui ont des ratios de fidélité élevés).

5.2 Évaluation et comparaison d'algorithmes d'apprentissage automatique supervisé et profond

Pour évaluer les performances des algorithmes traditionnels d'apprentissage automatique et celles de notre algorithme d'apprentissage profond, nous avons utilisé le F-score, qui est une métrique populaire et fiable utilisée afin d'évaluer les méthodes de classification (Rustam *et al.*, 2019; Lipton *et al.*, 2014; Joachims, 2005). Le F-score est la moyenne harmonique de la précision et du rappel. Il est défini comme suit :

$$F = \frac{2 \times Precision \times Rappel}{Precision + Rappel}, \tag{5.8}$$

où le rappel est défini par la formule $\frac{TP}{TP+FN}$ et la précision par la formule $\frac{TP}{TP+FP}$. Dans ces deux formules, TP représente les vrais positifs (échantillons positifs correctement classés), TN représente les vrais négatifs (échantillons négatifs correctement classés), les FP représente les faux positifs (échantillons négatifs classés comme positifs) et FN représente les faux négatifs (échantillons positifs classés comme négatifs).

En plus de cette métrique, nous avons également compilé les résultats relatifs au rappel et au taux de succès (*accuracy rate*, avec pour formule $\frac{TP+TN}{TP+FP+TN+FN}$) obtenus par les différentes méthodes comparées afin d'avoir un meilleur aperçu des performances réalisées. Le rappel calcul le rapport entre les observations classées comme étant positives et toutes les observations appartenant effectivement à la classe réelle. Le taux de succès permet d'obtenir le rapport entre le nombre d'observations correctement classés et le nombre d'observations total de l'ensemble de données considéré.

5.3 Résultats

TABLEAU 5.1 – F-scores obtenus par les méthodes ML/DL pour tous les utilisateurs du site Web CircuitPromo ainsi que pour chacun des quatre groupes d'utilisateurs identifiés. Les meilleurs résultats sont mis en évidence en gras.

Méthodes ML/DL	**Tous les utilisateurs**	**Groupe 1**	**Groupe 2**	**Groupe 3**	**Groupe 4**
	831 util.	278 util.	276 util.	214 util.	63 util.
Decision Tree	0.418	0.473	0.306	0.487	0.425
Random Forest	0.516	**0.583**	0.355	0.643	0.508
GBT	0.435	0.496	0.313	0.509	0.444
CatBoost	0.465	0.534	0.288	0.619	0.414
XGBoost	0.438	0.501	0.312	0.521	0.431
Naive Bayes	0.268	0.352	0.203	0.221	0.347
SVM-RBF	0.514	0.579	0.337	**0.662**	0.482
Rég. Logistique	0.503	0.562	0.363	0.616	0.470
MLP	0.437	0.489	0.296	0.547	0.454
RNN (GRU)	**0.559**	0.568	**0.506**	0.605	**0.597**
F-score moyen	0.455	0.514	0.328	0.543	0.457

Les résultats du F-score concernant les algorithmes traditionnels d'apprentissage automatique et notre algorithme d'apprentissage profond sont présentés dans le tableau 5.1. Dans ce tableau, les résultats moyens globaux du F-score (obtenus sur l'ensemble des 831 utilisateurs de la plateforme CircuitPromo) sont indiqués. On y retrouve également les performances de chaque méthode sur les différents groupes issus de l'étape de partitionnement.

Nous pouvons observer que trois algorithmes se démarquent en performant mieux que le reste des méthodes. Ces derniers offrent les meilleures performances concernant le F-score pour au moins un groupe d'utilisateurs. Le meilleur résultat global consistant en un F-score de 0,559 a été obtenu par notre modèle RNN-GRU. Ce modèle a également fourni les meilleurs résultats moyens pour les utilisateurs du groupe 2 (avec un F-score de 0,506) et ceux du groupe 4 (avec un F-score de 0,597), dont le comportement est le plus difficile à prédire. Random Forest a donné les meilleurs résultats pour les utilisateurs du groupe 1 (avec un F-score de 0,583), tandis que SVM a fourni les meilleurs résultats pour les utilisateurs du groupe 3 (avec un F-score de 0,662 ; le comportement des utilisateurs de ce groupe était le plus facile à prédire). Nous pouvons également remarquer que les algorithmes de base tels que Naive Bayes et Decision Tree ont constamment sous-performé sur tous les groupes.

Les tableaux 5.2 et 5.3 présentent respectivement les résultats de rappel et du taux de succès fournis par les algorithmes d'apprentissage automatique et profond considérés dans notre étude. Ces résultats concordent avec ceux obtenus pour le F-score et qui sont rapportés dans le tableau 5.1 car ici encore, l'algorithme RNN-GRU surpasse les autres méthodes pour l'ensemble des 831 utilisateurs.

TABLEAU 5.2 – Valeurs du rappel fournies par des méthodes ML/DL supervisées pour tous les utilisateurs du site Web CircuitPromo ainsi que pour chacun des quatre groupes d'utilisateurs identifiés. Les meilleurs résultats sont mis en évidence en gras.

Méthodes ML/DL	**Tous les utilisateurs**	**Groupe 1**	**Groupe 2**	**Groupe 3**	**Groupe 4**
	831 util.	278 util.	276 util.	214 util.	63 util.
Decision Tree	0.602	0.625	0.481	0.714	0.596
Random Forest	0.689	**0.738**	0.497	**0.872**	0.665
GBT	0.627	0.653	0.503	0.749	0.609
CatBoost	0.605	0.661	0.390	0.824	0.530
XGBoost	0.633	0.671	0.503	0.755	0.598
Naive Bayes	0.564	0.587	0.529	0.556	0.611
SVM-RBF	0.588	0.661	0.383	0.758	0.557
Rég. Logistique	0.643	0.701	0.490	0.778	0.576
MLP	0.587	0.648	0.413	0.719	0.599
RNN (GRU)	**0.729**	0.723	**0.648**	0.834	**0.754**
Rappel moyen	0.626	0.666	0.483	0.755	0.609

TABLEAU 5.3 – Valeurs du taux de succès (en %) fournies par des méthodes ML/DL supervisées pour tous les utilisateurs du site Web CircuitPromo ainsi que pour chacun des quatre groupes d'utilisateurs identifiés. Les meilleurs résultats sont mis en évidence en gras.

Méthodes ML/DL	**Tous les utilisateurs**	**Groupe 1**	**Groupe 2**	**Groupe 3**	**Groupe 4**
	831 util.	278 util.	276 util.	214 util.	63 util.
Decision Tree	40.5	46.5	30.1	44.8	42.4
Random Forest	49.3	55.6	33.3	60.9	48.9
GBT	41.5	47.5	30.1	47.4	42.9
CatBoost	47.4	54.8	29.8	60.3	45.3
XGBoost	41.8	48.4	29.5	48.0	42.4
Naive Bayes	34.7	42.0	26.2	35.9	40.7
SVM-RBF	50.3	**56.6**	33.6	**63.8**	46.8
Rég. Logistique	47.3	53.1	33.9	57.2	44.4
MLP	42.1	47.3	29.5	50.0	44.9
RNN (GRU)	**53.3**	54.1	**48.4**	57.1	**57.8**
Taux moyen	44.8	50.4	32.4	52.5	45.6

Notre modèle RNN-GRU (DREAM généralisé) apprend une représentation dynamique d'un utilisateur donné et capture les caractéristiques séquentielles globales existant parmi les paniers de l'utilisateur. Ceci en fait une approche optimisée pour la tâche de recommandation personnalisée du prochain panier. Aussi, notre méthode est capable, dans l'ensemble, de correctement modéliser le comportement du groupe d'utilisateurs le plus diversifié, c'est-à-dire ceux qui forment le groupe 2 et qui achètent leurs courses dans différents magasins tout en étant sensibles aux promotions.

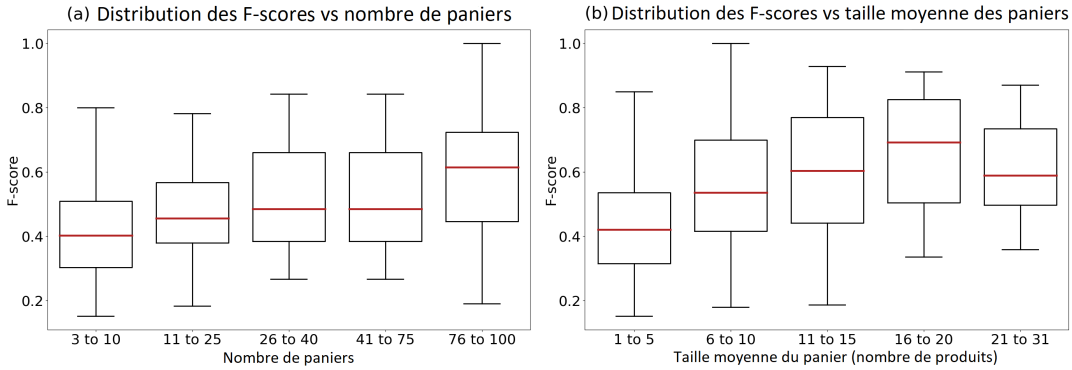

FIGURE 5.3 – Boîtes à moustache construites pour les résultats de la prédiction basée sur Random Forest : (a) variation du F-score par rapport au nombre de paniers ; (b) variation du F-score par rapport au nombre de produits par panier.

Les figures 5.3 et 5.4 illustrent respectivement l'impact du nombre de paniers et de la taille moyenne du panier sur les performances de prédiction de Random Forest (le meilleur algorithme d'apprentissage automatique traditionnel) et sur celles de notre méthode RNN-GRU. Nous pouvons observer que Random Forest et RNN-GRU fonctionnent mieux pour les utilisateurs avec un grand nombre de paniers (75 et plus), bien que l'impact du nombre de paniers soit plus important pour Random Forest (voir Fig. 5.3a).

Par ailleurs, une taille de panier moyenne plus grande ne se traduit pas toujours

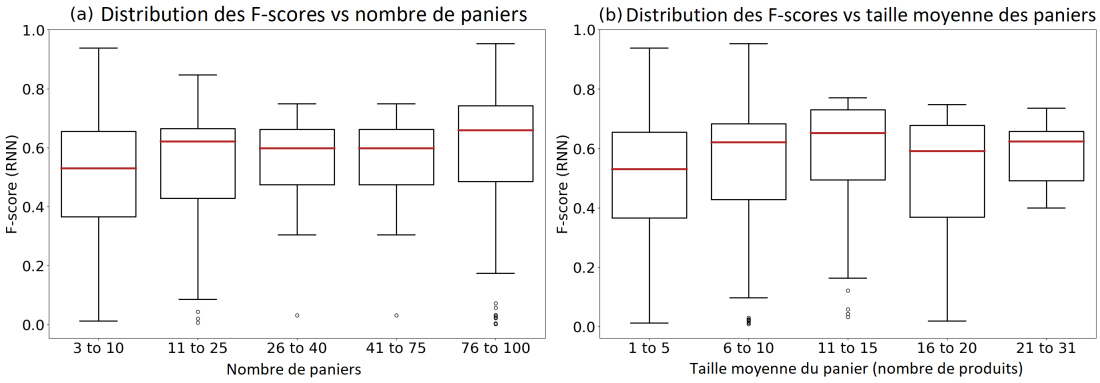

FIGURE 5.4 – Boîtes à moustache construites pour les résultats de la prédiction basée sur notre modèle RNN-GRU : (a) variation du F-score par rapport au nombre de paniers ; (b) variation du F-score par rapport au nombre de produits par panier.

par une meilleure performance de prédiction. Par exemple, Random Forest (voir Fig. 5.3b) est moins efficace pour les utilisateurs ayant un panier moyen supérieur à 20 articles que pour ceux dont le panier moyen varie entre 16 et 20 articles. Cela peut être dû à des relations complexes entre les articles dans les paniers. Les performances de RNN-GRU semblent moins affectées par la taille du panier, bien que cet algorithme fonctionne mieux pour les utilisateurs ayant plus de 5 articles en moyenne dans leur panier.

Même si notre modèle RNN-GRU ressort avec la meilleure performance globale, les résultats que nous avons obtenus mettent en évidence que différents modèles de prédiction devraient être appliqués selon le profil de l'utilisateur concerné. Une approche généralisée pour l'ensemble des utilisateurs n'est pas forcément optimale dans le contexte de la prédiction du prochain panier. L'étape de partitionnement est, dans ce sens, essentielle puisqu'elle permet d'aboutir à des groupes distincts et sans chevauchement, notamment grâce à notre formule du rapport de fidélité moyen qui prend en compte les caractéristiques du ratio de fidélité basé sur la

quantité (RFQ) et du ratio de fidélité basé sur le prix (RFP).

Comme appuyé par les résultats présentés dans cette section, notre modèle d'apprentissage profond se veut plus performant pour les utilisateurs appartenant aux groupes 2 et 4. Le groupe 2 est particulièrement difficile à prédire, compte tenu du fait que ses utilisateurs ont des ratios de fidélité faibles et qu'ils sont généralement plutôt sensibles aux spéciaux. Le groupe 4 quant à lui regroupe des utilisateurs dont les ratios de fidélité sont très élevés et qui sont très sensibles aux promotions. Une des raisons qui pourrait alors expliquer cette performance serait la capacité de notre modèle à tenir compte des dépendances séquentielles. L'historique d'achat jouerait alors un rôle important dans le cadre de cette méthode et les précédents paniers de l'utilisateur seraient particulièrement importants pour ces deux groupes d'utilisateurs.

Le modèle SVM est en mesure de cerner des relations non-linéaires entre les attributs, notamment en les projetant dans un espace de dimensions supérieures. Le groupe 3 semble être le plus stable et le prévisible de tous puisque ses clients ont tendance à acheter généralement les mêmes produits ou des produits similaires, et ce, dans le même magasin. Les données concernant ce groupe pourraient avoir des frontières de décision bien définies que le modèle SVM est capable de déterminer efficacement, d'où une meilleure performance de ce modèle uniquement sur le groupe 3.

Enfin, concernant Random Forest, les utilisateurs appartenant au groupe 1 ont des ratios de fidélité élevés. Étant peu sensibles aux spéciaux et achetant leurs produits dans le même magasin, leur comportement d'achat peut indiquer que certaines caractéristiques interagissent de façon complexe et non-linéaire. La combinaison d'arbres de décision semble être capable de bien rendre compte de ces interactions afin d'aboutir à une meilleure prédiction pour ce groupe d'utilisateurs.

Par conséquent, en prenant en compte les historiques d'achat, les profils, les préférences et les promotions hebdomadaires disponibles pour les utilisateurs de CircuitPromo, les résultats obtenus ici indiquent que dans notre situation, une approche personnalisée appliquant selon le contexte soit le modèle d'apprentissage automatique le plus pertinent, soit notre modèle d'apprentissage profond est une solution congruente et adéquate.

CHAPITRE VI

CONCLUSION

Dans ce mémoire, nous avons présenté un nouveau système de recommandation personnalisé implémenté sur la plateforme Web CircuitPromo, conçue pour recommander les meilleures offres d'épicerie hebdomadaires aux clients canadiens. Notre système applique le modèle de prédiction d'apprentissage automatique ou d'apprentissage profond le plus approprié (voir Figure 6.1) pour fournir au client considéré une liste de courses hebdomadaire sur mesure, accompagnée d'une liste des magasins dans lesquels il doit se rendre pour acheter les produits recommandés. Notre système prend en compte plusieurs attributs liés à l'historique d'achat du client ainsi que des caractéristiques liées au prix actuel et à la disponibilité des produits dans les épiceries locales. L'un des avantages de notre système de recommandation est qu'il peut recommander au client des produits qu'il n'a jamais achetés auparavant, ce qui peut être utile pour découvrir de nouveaux produits pertinents ou être tenu au courant des offres à durée limitée.

Nos résultats démontrent que différentes méthodes d'apprentissage automatique ou d'apprentissage profond doivent être appliquées pour différents groupes d'utilisateurs (voir les résultats dans les tableaux 5.1, 5.2 et 5.3). Afin d'identifier ces groupes d'utilisateurs en fonction de leur comportement d'achat, nous avons réalisé deux expériences de partionnement avec les algorithmes de K-moyennes et de

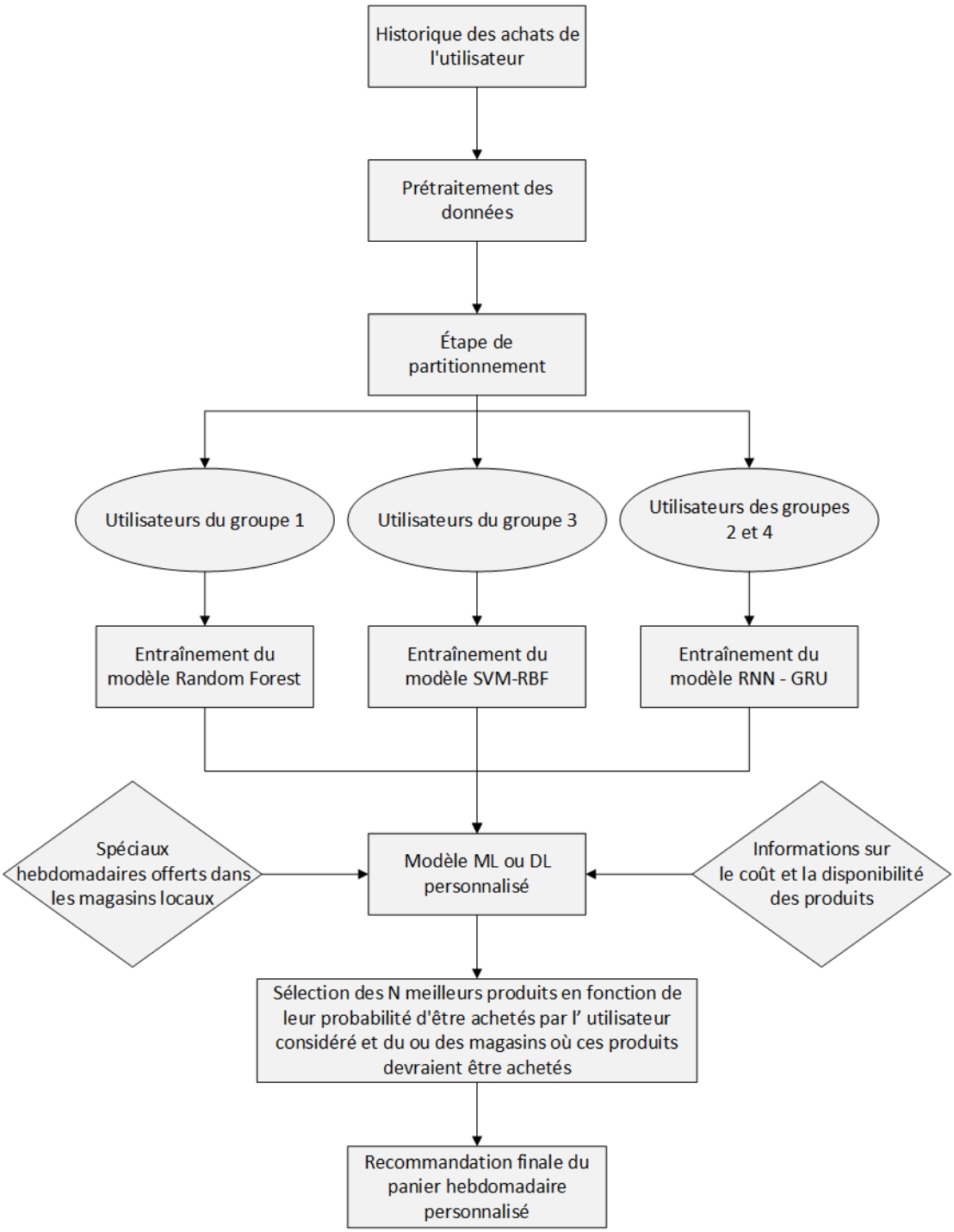

FIGURE 6.1 – Vue d'ensemble du système de recommandation proposé.

Ward (partitionnement hiérarchique), tous deux connus pour leur simplicité et leur rapidité. Notre étape de partitionnement, effectuée à l'aide de l'algorithme de Ward, a divisé l'ensemble des 831 clients canadiens considérés dans ce travail en quatre groupes selon leurs habitudes d'achat. Dans cette étude, nous avons également introduit la formule du rapport de fidélité moyen utilisée dans notre étape de partitionnement (voir sous-section 5.1.1 du chapitre 5). Cette caractéristique a été définie comme la moyenne entre le ratio de fidélité basé sur la quantité (RFQ) et le ratio de fidélité basé sur le prix (RFP) introduits respectivement via les équations (5.1) et (5.2). Nous avons ensuite utilisé différents modèles d'apprentissage automatique traditionnels et un nouveau modèle d'apprentissage profond pour fournir les recommandations du prochain panier. Fait intéressant, les valeurs moyennes des F-scores obtenues pour les utilisateurs de différents groupes étaient assez différentes (voir Tableau 5.1). Elles variaient de 0,328 (pour les utilisateurs du groupe 2 - qui font leurs courses dans différents magasins et sont sensibles aux promotions) à 0,543 (pour les utilisateurs du groupe 3 - qui achètent généralement les mêmes produits ou des produits similaires dans le même magasin, et sont peu sensibles aux spéciaux). Nous pouvons également observer que certaines méthodes d'apprentissage automatique sont bien meilleures que d'autres pour recommander des éléments à un groupe spécifique d'utilisateurs. Ainsi, il serait plausible d'appliquer différentes méthodes de prédiction pour différents groupes de clients : Random Forest pour les clients du groupe 1, RNN-GRU pour les clients des groupes 2 et 4, et SVM-RBF pour les clients du groupe 3. Globalement, les meilleur résultats ont été obtenus par notre modèle RNN-GRU. En termes de F-score moyen, il a surpassé Random Forest, le deuxième modèle le plus performant, de 0,043. RNN-GRU a également donné les résultats les plus cohérents à travers tous les groupes. L'organigramme illustré dans la Figure 6.1 fournit un aperçu général de notre système de recommandation.

Dans le cas des systèmes de recommandation, des modèles tels que les réseaux de neurones récurrents à portes (RNN-GRU) et les transformers peuvent être utilisés pour capter des dépendances séquentielles dans les données. Les transformers peuvent en effet voir leur architecture de traitement du langage naturel, qui se charge habituellement de prédire le prochain caractère, modifiée pour effectuer une tâche de prédiction du prochain produit dans le nouveau panier d'un utilisateur, ce qui les classe dans la catégorie du filtrage collaboratif, basé dans ce cas de figure sur les éléments du panier.

Toutefois, les transformers sont généralement plus coûteux en termes de ressources, notamment en temps de traitement et en mémoire. Lorsque l'historique d'achat contient beaucoup de paniers, le temps de calcul peut devenir bien plus long. Par ailleurs, les transformers traitent toutes les entrées en parallèle et n'ont pas de concept relatif à l'ordre des événements dans une séquence. Ce souci peut être résolu de plusieurs façons, par exemple en ajoutant des attributs de positionnement ou d'ordre. Un RNN-GRU a l'avantage inhérent de traiter les séquences dans l'ordre, ce qui, dans notre cas, était un avantage non négligeable. Pour ces raisons, nous avons fait le choix d'opter pour le RNN-GRU, à travers une extension du modèle DREAM.

La supériorité du modèle RNN-GRU proposé ici indique que dans un contexte d'épicerie le comportement temporel de l'utilisateur (qui révèle ses intérêts dynamiques à différents moments), ainsi que les caractéristiques séquentielles des paniers (qui reflètent les interactions entre tous les paniers de l'utilisateur au fil du temps), sont deux facteurs de prédiction cruciaux pour la recommandation du prochain panier. Nos résultats de prédiction prometteurs s'expliquent par la nature des données : en effet, les données d'épicerie sont souvent très répétitives car les utilisateurs ont tendance à acheter régulièrement un ensemble d'articles similaires (comme des produits de première nécessité), développant ainsi des habitudes

constantes.

Il est important de noter qu'en termes de F-score, notre modèle RNN-GRU personnalisé a surpassé le récent modèle général basé sur LSTM et proposé par Tahiri *et al.* de 0,339 lorsque nous avons utilisé les nouvelles données disponible sur la plateforme CircuitPromo. De plus, concernant les données augmentées considérées par Tahiri *et al.*, notre meilleur résultat F-score était supérieur de 0,189 à celui de leur étude. Le modèle introduit dans notre travail est personnalisé (c'est-à-dire que les paramètres du modèle sont ajustés pour chaque utilisateur). Plus précisément, notre modèle actuel équivaut à entraîner un seul modèle agrégé (comme celui de Tahiri *et al.*) pour tous les utilisateurs, et à conditionner les entrées sur l'intégration du comportement de l'utilisateur. Ceci explique ses performances supérieures à celles du modèle agrégé de Tahiri *et al.*

L'implémentation Python de tous les algorithmes de partitionnement et d'apprentissage automatique et profond utilisés dans notre travail ainsi que l'ensemble des données anonymisées des 831 utilisateurs considérés sont disponibles dans notre dépôt GitHub [1]. Pour plus de détails, l'article de revue (Chabane *et al.*, 2022) est disponible sur le site de la revue PLOS One [2] et l'article de conférence (Chabane *et al.*, 2023) est disponible sur le site de Springer [3].

L'une des limites de notre approche réside dans la plateforme web elle-même. En effet, CircuitPromo ne permet pas aux utilisateurs d'acheter directement les produits. Ainsi, nous n'avons aucune assurance que les utilisateurs ont effectivement acheté les articles inclus dans leurs listes d'épicerie. Nous ne pouvons pas

1. https://github.com/Achrafb11/Smartshopping

2. https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0278364

3. https://link.springer.com/book/9783031090356

non plus suivre les stocks des différents magasins pour éventuellement informer les utilisateurs des pénuries avant d'ajouter des produits à leurs listes d'épicerie. Notre système de recommandation est également sensible au problème du démarrage à froid et il n'est pas encore en mesure de prédire la quantité exacte de chaque article recommandé dans le prochain panier de l'utilisateur. Nous prévoyons de remédier à ces limitations dans nos travaux futurs, dans lesquels nous explorerons également l'impact du caractère saisonnier des produits sur les habitudes d'achat en épicerie, ce qui pourrait également conduire à de meilleures recommandations. Il pourra également être question de mettre en place un modèle sur la base d'un transfomer et de comparer les résultats nouvellement obtenus avec ceux de notre travail actuel.

RÉFÉRENCES

Adomavicius, G. et Tuzhilin, A. (2005). Toward the next generation of recommender systems : A survey of the state-of-the-art and possible extensions. *IEEE transactions on knowledge and data engineering*, *17*(6), 734–749.

Aggarwal, C. C. (2016). Knowledge-based recommender systems. In *Recommender systems* 167–197. Springer.

Al-Shamri, M. Y. H. (2016). User profiling approaches for demographic recommender systems. *Knowledge-Based Systems*, *100*, 175–187.

Arafat, M. Y., Hoque, S. et Farid, D. M. (2017). Cluster-based under-sampling with random forest for multi-class imbalanced classification. Dans *2017 11th International Conference on Software, Knowledge, Information Management and Applications (SKIMA)*, 1–6. IEEE.

Ariannezhad, M., Jullien, S., Li, M., Fang, M., Schelter, S. et de Rijke, M. (2022). Recanet : A repeat consumption-aware neural network for next basket recommendation in grocery shopping. Dans *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, p. 1240–1250., New York, NY, USA. Association for Computing Machinery. `http://dx.doi.org/10.1145/3477495.3531708`. Récupéré de `https://doi.org/10.1145/3477495.3531708`

Ariannezhad, M., Li, M., Schelter, S. et de Rijke, M. (2023). A personalized neighborhood-based model for within-basket recommendation in grocery shopping. Dans *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, WSDM '23, p. 87–95., New York, NY, USA. Association for Computing Machinery. `http://dx.doi.org/10.1145/3539597.3570417`. Récupéré de `https://doi.org/10.1145/3539597.3570417`

Bergman, S. (1970). *The kernel function and conformal mapping*, volume 5. American Mathematical Soc.

Bergstra, J. et Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of machine learning research*, *13*(2).

Bhattacharya, S., Floréen, P., Forsblom, A., Hemminki, S., Myllymäki, P., Nurmi, P., Pulkkinen, T. et Salovaara, A. (2012). Ma$$iv€– an intelligent mobile grocery assistant. Dans *2012 Eighth International Conference on Intelligent Environments*, 165–172. IEEE.

Breiman, L. (2001). Random forests. *Machine learning*, *45*(1), 5–32.

Breiman, L., Friedman, J. H., Olshen, R. A. et Stone, C. J. (1984). Classification and regression trees. belmont, ca : Wadsworth. *International Group*, *432*, 151–166.

Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., Niculae, V., Prettenhofer, P., Gramfort, A., Grobler, J., Layton, R., VanderPlas, J., Joly, A., Holt, B. et Varoquaux, G. (2013). API design for machine learning software : experiences from the scikit-learn project. Dans *ECML PKDD Workshop : Languages for Data Mining and Machine Learning*, 108–122.

Burke, R. (2000). Knowledge-based recommender systems. *Encyclopedia of library and information systems*, *69*(Supplement 32), 175–186.

Chabane, N., Bouaoune, A., Tighilt, R., Abdar, M., Boc, A., Lord, E., Tahiri, N., Mazoure, B., Acharya, U. R. et Makarenkov, V. (2022). Intelligent personalized shopping recommendation using clustering and supervised machine learning algorithms. *PLOS ONE*, *17*(12), 1–30. `http://dx.doi.org/10.1371/journal.pone.0278364`

Chabane, N., Bouaoune, A., Tighilt, R., Tahiri, N., Mazoure, B. et Makarenkov, V. (2023). Using clustering and machine learning methods to provide intelligent grocery shopping recommendations. Dans *Classification and Data Science in the Digital Age, Proceedings of IFCS 2022*, Studies in Classification, Data Analysis, and Knowledge Organization. Springer Cham.

Che, B., Zhao, P., Fang, J., Zhao, L., Sheng, V. S. et Cui, Z. (2019). Inter-basket and intra-basket adaptive attention network for next basket recommendation. *IEEE Access*, *7*, 80644–80650.

Chen, J., Wang, H., Yan, Z. *et al.* (2018). Evolutionary heterogeneous clustering for rating prediction based on user collaborative filtering. *Swarm and Evolutionary Computation*, *38*, 35–41.

Chen, T. et Guestrin, C. (2016). Xgboost : A scalable tree boosting system.

Dans *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 785–794.

Claesen, M., Smet, F. D., Suykens, J. A. K. et Moor, B. D. (2014). Fast prediction with svm models containing rbf kernels.

Cordeiro de Amorim, R., Makarenkov, V. et Mirkin, B. (2016). A-ward_pβ : Effective hierarchical clustering using the minkowski metric and a fast k-means initialisation. *Information Sciences*, *370-371*, 343–354. Récupéré de `https://www.sciencedirect.com/science/article/pii/S0020025516305606`

Cortes, C. et Vapnik, V. (1995). Support-vector networks. *Machine learning*, *20*(3), 273–297.

Davies, D. L. et Bouldin, D. W. (1979). A cluster separation measure. in : Ieee transactions on pattern analysis and machine intelligence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *PAMI-1, no. 2*, 224–227.

de Amorim, R. C. et Makarenkov, V. (2016). Applying subclustering and lp distance in weighted k-means with distributed centroids. *Neurocomputing*, *173*, 700–707. Récupéré de `https://www.sciencedirect.com/science/article/pii/S0925231215011650`

Deshpande, M. et Karypis, G. (2004). Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS)*, *22*(1), 143–177.

Diab, D. M. et El Hindi, K. M. (2017). Using differential evolution for fine tuning naïve bayesian classifiers and its application for text classification. *Applied Soft Computing*, *54*, 183–199.

Dorogush, A. V., Ershov, V. et Gulin, A. (2018). Catboost : gradient boosting with categorical features support. *arXiv preprint arXiv :1810.11363*.

Dou, X. (2020). Online purchase behavior prediction and analysis using ensemble learning. Dans *2020 IEEE 5th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA)*, 532–536.

Eirinaki, M., Gao, J., Varlamis, I. et Tserpes, K. (2018). Recommender systems for large-scale social networks : A review of challenges and solutions.

Faggioli, G., Polato, M. et Aiolli, F. (2020). Recency aware collaborative filtering for next basket recommendation. Dans *Proceedings of the 28th ACM Conference on User Modeling, Adaptation and Personalization*, p. 80–87., New York, NY, USA. Association for Computing Machinery. Récupéré de

https://doi.org/10.1145/3340631.3394850

Friedman, J. H. (2001). Greedy function approximation : a gradient boosting machine. *Annals of statistics*, 1189–1232.

Friedman, J. H. (2002). Stochastic gradient boosting. *Computational statistics & data analysis*, *38*(4), 367–378.

García, S., Luengo, J. et Herrera, F. (2015). *Data preprocessing in data mining*. Springer.

Gupta, A. et Shrinath, P. (2022). A novel recommendation system comprising wnmf with graph-based static and temporal similarity estimators. *International Journal of Data Science and Analytics*, 1–15.

Han, J., Pei, J. et Kamber, M. (2011). *Data mining : concepts and techniques*. Elsevier.

Hasanin, T. et Khoshgoftaar, T. (2018). The effects of random undersampling with simulated class imbalance for big data. Dans *2018 IEEE International Conference on Information Reuse and Integration (IRI)*, 70–79. IEEE.

Hutter, F., Hoos, H. H. et Leyton-Brown, K. (2011). Sequential model-based optimization for general algorithm configuration. Dans *International conference on learning and intelligent optimization*, 507–523. Springer.

Jain, S., Shukla, S. et Wadhvani, R. (2018). Dynamic selection of normalization techniques using data complexity measures. *Expert Systems with Applications*, *106*, 252–262.

Joachims, T. (2005). A support vector method for multivariate performance measures. Dans *Proceedings of the 22nd international conference on Machine learning*, 377–384.

Karimi, M., Jannach, D. et Jugovac, M. (2018). News recommender systems–survey and roads ahead. *Information Processing & Management*, *54*(6), 1203–1227.

Koren, Y. et Bell, R. (2015). Advances in collaborative filtering. In *Recommender systems handbook* 77–118. Springer.

Kotsiantis, S., Kanellopoulos, D. et Pintelas, P. (2006). Data preprocessing for supervised leaning. *International Journal of Computer Science*, *1*(2), 111–117.

Larson, S. C. (1931). The shrinkage of the coefficient of multiple correlation. *Journal of Educational Psychology*, *22*(1), 45.

Le, D.-T., Lauw, H. W. et Fang, Y. (2019). Correlation-sensitive next-basket recommendation. Dans *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, IJCAI'19, p. 2808–2814. AAAI Press.

Lee, H. I., Choi, I. Y., Moon, H. S. et Kim, J. K. (2020). A multi-period product recommender system in online food market based on recurrent neural networks. *Sustainability*, *12*(3). Récupéré de `https://www.mdpi.com/2071-1050/12/3/969`

Li, M., Jullien, S., Ariannezhad, M. et de Rijke, M. (2021). A next basket recommendation reality check. arXiv :2109.14233, `http://dx.doi.org/10.48550/ARXIV.2109.14233`. Récupéré de `https://arxiv.org/abs/2109.14233`

Lika, B., Kolomvatsos, K. et Hadjiefthymiades, S. (2014). Facing the cold start problem in recommender systems. *Expert Systems with Applications*, *41*(4), 2065–2073.

Lipton, Z. C., Elkan, C. et Naryanaswamy, B. (2014). Optimal thresholding of classifiers to maximize f1 measure. Dans *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 225–239. Springer.

Lu, J., Wu, D., Mao, M., Wang, W. et Zhang, G. (2015). Recommender system application developments : a survey. *Decision Support Systems*, *74*, 12–32.

MacQueen, J. *et al.* (1967). Some methods for classification and analysis of multivariate observations. Dans *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, 281–297. Oakland, CA, USA.

Makarenkov, V. et Legendre, P. (1999). Une méthode d'analyse canonique non linéaire et son application à des données biologiques. *Mathématiques et sciences humaines. Mathematics and social sciences*, (147).

McFadden, D. *et al.* (1973). Conditional logit analysis of qualitative choice behavior.

McLachlan, G. J., Do, K.-A. et Ambroise, C. (2005). Analyzing microarray gene expression data.

Melville, P. et Sindhwani, V. (2017). Recommender systems. In *Encyclopedia of Machine Learning and Data Mining*. Routledge.

Newcomb, E., Pashley, T. et Stasko, J. (2003). Mobile computing in the retail

arena. Dans *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 337–344. ACM.

Noble, W. S. (2006). What is a support vector machine ? *Nature biotechnology*, *24*(12), 1565–1567.

Pan, J., Zhuang, Y. et Fong, S. (2016). The impact of data normalization on stock market prediction : using svm and technical indicators. Dans *International Conference on Soft Computing in Data Science*, 72–88. Springer.

Park, D. H., Kim, H. K., Choi, I. Y. et Kim, J. K. (2012). A literature review and classification of recommender systems research. *Expert systems with applications*, *39*(11), 10059–10072.

Park, Y.-J. et Chang, K.-N. (2009). Individual and group behavior-based customer profile model for personalized product recommendation. *Expert Systems with Applications*, *36*(2), 1932–1939.

Pazzani, M. J. et Billsus, D. (2007). Content-based recommendation systems. In *The adaptive web* 325–341. Springer.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. et Duchesnay, E. (2011). Scikit-learn : Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V. et Gulin, A. (2018a). Catboost : unbiased boosting with categorical features. Dans S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, et R. Garnett (dir.). *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V. et Gulin, A. (2018b). Catboost : unbiased boosting with categorical features. *Advances in neural information processing systems*, *31*.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P. et Riedl, J. (1994). Grouplens : An open architecture for collaborative filtering of netnews. Dans *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, 175–186.

Ricci, F., Rokach, L. et Shapira, B. (2015). Recommender systems : introduction and challenges. In *Recommender systems handbook* 1–34. Springer.

Rosenblatt, F. (1961). *Principles of neurodynamics. perceptrons and the theory of brain mechanisms*. Rapport technique, Cornell Aeronautical Lab Inc Buffalo NY.

Rousseeuw, P. J. (1987). Silhouettes : a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, *20*, 53–65.

Rumelhart, D. E., Hinton, G. E. et Williams, R. J. (1985). *Learning internal representations by error propagation*. Rapport technique, California Univ San Diego La Jolla Inst for Cognitive Science.

Rustam, F., Ashraf, I., Mehmood, A., Ullah, S. et Choi, G. S. (2019). Tweets classification on the base of sentiments for us airline companies. *Entropy*, *21*(11), 1078.

Safoury, L. et Salah, A. (2013). Exploiting user demographic attributes for solving cold-start problem in recommender system. *Lecture Notes on Software Engineering*, *1*(3), 303–307.

Shardanand, U. et Maes, P. (1995). Social information filtering : Algorithms for automating "word of mouth". Dans *Proceedings of the SIGCHI conference on Human factors in computing systems*, 210–217.

Shi, Y., Larson, M. et Hanjalic, A. (2014). Collaborative filtering beyond the user-item matrix : A survey of the state of the art and future challenges. *ACM Computing Surveys (CSUR)*, *47*(1), 1–45.

Singh, A. et Sharma, A. (2018). Maicbr : A multi-agent intelligent content-based recommendation system. In *Information and Communication Technology for Sustainable Development* 399–411. Springer.

Stone, M. (1974). Cross-validatory choice and assessment of statistical predictions. *Journal of the royal statistical society : Series B (Methodological)*, *36*(2), 111–133.

Tahiri, N., Mazoure, B. et Makarenkov, V. (2019). An intelligent shopping list based on the application of partitioning and machine learning algorithms. Dans *proceedings of the 18th Python in Science Conference (SCIPY 2019)*.

Tasgetiren, M. F., Suganthan, P. N., Pan, Q.-K. et Liang, Y.-C. (2007). A genetic algorithm for the generalized traveling salesman problem. Dans *2007 IEEE Congress on Evolutionary Computation*, 2382–2389.

Van der Maaten, L. et Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, *9*(11).

Verma, V. et Aggarwal, R. K. (2020). Neighborhood-based collaborative recommendations : An introduction. In *Applications of Machine Learning* 91–110. Springer.

Villegas, N. M., Sánchez, C., Díaz-Cely, J. et Tamura, G. (2018). Characterizing context-aware recommender systems : A systematic literature review. *Knowledge-Based Systems*, *140*, 173–200.

Vincent-Wayne, M. et Aylott, R. (1998). An exploratory study of grocery shopping stressors. *International Journal of Retail & Distribution Management*, *26*(9), 362–373.

Walters, R. et Jamil, M. (2002). Measuring cross-category specials purchasing : theory, empirical results, and implications. *Journal of Market-Focused Management*, *5*(1), 25–42.

Ward Jr, J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, *58*(301), 236–244.

Xia, Y., Di Fabbrizio, G., Vaibhav, S. et Datta, A. (2017). A content-based recommender system for e-commerce offers and coupons. Dans *Proc. SIGIR Workshop eCommerce*.

Yu, F., Liu, Q., Wu, S., Wang, L. et Tan, T. (2016). A dynamic recurrent model for next basket recommendation. SIGIR '16, p. 729–732., New York, NY, USA. Association for Computing Machinery.

Zhang, H. (2005). Exploring conditions for the optimality of naive bayes. *International Journal of Pattern Recognition and Artificial Intelligence*, *19*(02), 183–198.

Zheng, Q. et Ding, Q. (2022). Exploration of consumer preference based on deep learning neural network model in the immersive marketing environment. *Plos one*, *17*(5), e0268007.

Zhou, L., Dai, L. et Zhang, D. (2007). Online shopping acceptance model-a critical survey of consumer factors in online shopping. *Journal of Electronic commerce research*, *8*(1).