UNIVERSITÉ FRANÇOIS - RABELAIS DE TOURS

École Doctorale MIPTIS

Laboratoire d'Informatique, Équipe BdTln

THÈSE présentée par :

Mouhamadou Saliou DIALLO

soutenue le : 27 Mars 2015

pour obtenir le grade de : **Docteur de l'université François - Rabelais de Tours et de l'université Gaston Berger de Saint-Louis du Sénégal**

Discipline / Spécialité : **Informatique**

Découvertes de règles de préférences contextuelles : application à la construction de profils utilisateurs

THÈSE dirigée par :

M. Arnaud GIACOMETTI	Professeur des universités, Université François - Rabelais de Tours
M. Cheikh Talibouya DIOP	Maître de conférences, HDR, Université Gaston Berger de Saint Louis - Sénégal

CO-ENCADRANTS :

M. Arnaud SOULET	Maître de conférences, Université François - Rabelais de Tours
M. Dominique LI	Maître de conférences, Université François - Rabelais de Tours

RAPPORTEURS :

M. Bruno CREMILLEUX	Professeur des universités, Université de Caen
M. Allel HADJALI	Professeur des universités, ENSMA de Poitiers

JURY :

M. Arnaud GIACOMETTI	Professeur des universités, Université François - Rabelais de Tours
M. Cheikh Talibouya DIOP	Maître de conférences, HDR, Université Gaston Berger de Saint Louis - Sénégal
M. Bruno CREMILLEUX	Professeur des universités, Université de Caen
M. Allel HADJALI	Professeur des universités, ENSMA de Poitiers

Certes rien n'est facile dans la vie mais aussi aucun obstacle ne résiste à qui sait prendre le temps de bien faire les choses. Le vie ne se trouve pas dans la connaissance ni dans la richesse mais dans l'acceptation de ce que l'on est avec amour et respect envers les autres.

(Lu quelque part sur Internet)

À Toute ma famille.

Remerciements

Résumé

L'utilisation de préférences suscite un intérêt croissant pour personnaliser des réponses aux requêtes et effectuer des recommandations ciblées. Pourtant, la construction manuelle de profils de préférences reste à la fois complexe et consommatrice de temps. Dans ce contexte, nous présentons dans cette thèse une nouvelle méthode automatique d'extraction de préférences basée sur des techniques de fouille de données.

L'approche que nous proposons est constituée de deux phases : (1) une phase d'extraction de toutes les règles de préférences contextuelles intéressantes et (2) une phase de construction du profil utilisateur. A la fin de la première phase, nous constatons qu'il y a des règles redondantes voir superflues ; la seconde phase permet d'éliminer les règles superflues afin d'avoir un profil concis et consistant. Dans notre approche, un profil utilisateur est constitué de cet ensemble de règles de préférences contextuelles résultats de la seconde phase. La consistance garantit que les règles de préférences spécifiant les profils sont en accord avec un grand nombre de préférences utilisateur et contredisent un petit nombre d'entre elles. D'autre part, la concision implique que les profils sont constitués d'un petit nombre de règles de préférences. Nous avons aussi proposé quatre méthodes de prédiction qui utilisent les profils construits.

Nous avons validé notre approche sur une base de données de films construite à partir de MovieLens et IMDB. La base de données contient 3 881 films notés par 6 040 utilisateurs. Ces derniers ont attribué 800 156 notes. Les résultats de ces expériences démontrent que la concision des profils utilisateurs est contrôlée par le seuil d'accord minimal et que même avec une forte réduction du nombre de règles, les qualités de prédiction des profils restent à un niveau acceptable. En plus des expérimentations montrant la qualité de prédiction de notre approche, nous avons montré également que les performances de notre approche peuvent rivaliser avec les qualités de prédiction de certaines méthodes de l'état de l'art, en particulier SVMRank.

Mots clés : elicitation de préférences, règles de préférences contextuelles, extraction de profil utilisateur, fouille de données.

Abstract

The use of preferences arouses a growing interest to personalize response to requests and making targeted recommandations. Nevertheless, manual construction of preferences profiles remains complex and time-consuming. In this context, we present in this thesis a new automatic method for preferences elicitation based on data mining techniques.

Our proposal is a two phase algorithm : (1) Extracting all contextual preferences rules from a set of user preferences and (2) Building user profile. At the end of the first phase, we notice that there is to much preference rules which satisfy the fixed constraints then in the second phase we eliminate the superfluous preferences rules. In our approach a user profile is constituted by the set of contextual preferences rules resulting of the second phase. A user profile must satisfy conciseness and soundness properties. The soundness property guarantees that the preference rules specifying the profiles are in agreement with a large set of the user preferences, and contradict a small number of them. On the other hand, conciseness implies that profiles are small sets of preference rules. We also proposed four predictions methods which use the extracted profiles.

We validated our approach on a set of real-world movie rating datasets built from MovieLens and IMDB. The whole movie rating database consists of 800,156 votes from 6,040 users about 3,881 movies. The results of these experiments demonstrates that the conciseness of user profiles is controlled by the minimal agreement threshold and that even with strong reduction, the soundness of the profile remains at an acceptable level. These experiment also show that predictive qualities of some of our ranking strategies outperform SVMRank in several situations.

Keywords : preference elicitation, contextual preference rule, user profile mining, data mining

Table des matières

Liste des tableaux

Liste des figures

Liste des algorithmes

Introduction générale

Contexte et motivations

Depuis une dizaine d'années, le problème de la représentation de préférences suscite un intérêt croissant dans le domaine de l'Intelligence Artificielle. Dans ce cadre, ont été proposés de nombreux formalismes permettant de représenter les préférences d'un utilisateur sur un ensemble d'objets, de la manière la plus concise et la plus fidèle possible, tout en permettant un raisonnement sur ces préférences.

Parmi les modèles proposés, on trouve deux grandes classes de modèles : les modèles quantitatifs, qui affectent un degré d'intérêt à chacun des objets [AW00, KI04], et les modèles qualitatifs, qui permettent notamment d'exprimer l'incomparabilité, au premier rang desquels les CP-nets, et plus généralement les modèles de préférences conditionnelles [Cho03, Kie02, Wil04, BBD+03]. La formulation qualitative est plus générale que la formulation quantitative : toutes les relations de préférences ne peuvent pas être exprimées avec des fonctions de scores ou à travers un degré d'intérêt.

En pratique, il est cependant constaté que la construction manuelle de modèles de préférences reste à la fois complexe et consommatrice de temps. De ce fait, l'apprentissage automatique de modèles de préférences, en s'appuyant sur les interactions entre un système et des utilisateurs, joue un rôle critique dans de nombreuses applications.

Il existe deux grandes approches pour apprendre automatiquement les préférences utilisateurs : l'apprentissage de préférences quantitatives et l'apprentissage de préférences qualitatives. Dans l'apprentissage de préférences quantitatives, l'objectif est d'apprendre une fonction de scores sur les données d'apprentissage ; le modèle appris assigne un score individuel à chaque objet, score qui peut être ainsi utilisé pour les classer [Joa06, FISS03, XL07, BSR+05]. Malheureusement, les modèles appris dans l'approche quantitative ne sont pas, toujours, facilement compréhensibles par l'utilisateur afin que ce dernier puisse les compléter ou bien les corriger.
Les travaux dans l'apprentissage de préférences qualitatives apprennent une relation binaire sur les données d'apprentissage [HEK03, Jea08, CKL+10, GLS14, DKSS11]. Cepen-

dant certains travaux ne considèrent pas les préférences contextuelles [HEK03, Jea08], d'autres ne peuvent pas travailler sur des données bruitées [CKL+10, YWLd08, SM06, DIV07].
Dans ce contexte, l'objectif principal de ce travail est de proposer de nouvelles méthodes de découvertes de préférences qualitatives satisfaisants les deux principaux critères suivants :

- Compréhensibilité : Les modèles de préférences construits doivent être compréhensibles par un utilisateur afin d'être complétés voir corrigés. Pour cette raison, nous construirons des modèles de préférences basés sur des règles de préférences conditionnelles exprimant de manière intelligible les préférences d'un utilisateur.
- Concision et consistance : la contrainte de concision signifie que le nombre de règles de préférences contextuelles dans le profil doit être gérable, tandis que la consistance garantit que les règles de préférences que composent les profils sont en accord avec un grand nombre de préférences utilisateurs et contredisent un petit nombre d'entre elles.

Pour construire de tels modèles, nous proposons un algorithme en deux phases :

- Une première phase d'extraction de toutes les règles de préférences pertinentes,
- Une seconde phase de sélection d'un sous-ensemble compréhensif et consistant de règles de préférences ; cette seconde phase sélectionnera de manière gloutonne les règles les plus pertinentes.

Nous présentons dans la section suivante les grandes lignes des propositions que nous avons effectuées.

Principales contributions

Comme nous l'avons indiqué dans la section précédente, notre contribution consiste principalement en un algorithme à deux phases ; ces deux phases sont décrites dans les sections suivantes :

Extraction de règles de préférences contextuelles

L'objectif de cette phase est d'extraire toutes les règles de préférences contextuelles qui satisfont un certain nombre de critères. Pour cela, nous avons proposé un algorithme en profondeur inspiré de Eclat [Zak00] et appelé ContPrefMiner. Après cette première phase, nous constatons qu'il y'a une grande quantités de règles extraites et, qu'il existe un certain nombre de règles redondantes et voire superflues. Il est par conséquent

nécessaire d'avoir une seconde phase permettant de sélectionner seulement les règles les plus pertinentes pour construire un modèle.

Construction de profil utilisateurs

Nous avons besoin d'une seconde phase pour construire le modèle à partir de l'ensemble des règles de préférences contextuelles extraites par ContPrefMiner. Le modèle construit est le profil d'un utilisateur, il est composé d'un ensemble de règles de préférences contextuelles[ART06] qui satisfont quelques critères intéressantes comme la *consistance* et la *concision*. La *consistance* garantie que les règles de préférences qui constituent le profil sont en accord avec un grand nombre de préférences utilisateurs et en contredisent un petit nombre. D'un autre coté, la *concision* indique que le profil construit est constitué d'un petit nombre de règles de préférences contextuelles. L'algorithme permettant de construire le modèle est nommé ProfMiner.

Méthodes d'utilisation des profils

En plus d'avoir introduit la méthode ProfMiner pour la construction du profil utilisateur, nous avons aussi proposé trois stratégies de classements qui permettent d'utiliser le profil construit dans le but de classer les objets. Les trois stratégies de classement que nous avons proposés sont décrites en détail dans les sections 3.4.1, 3.4.2, et 3.4.3.

Nous soutenons que notre approche basée sur les règles de préférences contextuelles a un réel avantage sur les autres méthodes : Le modèle construit est facilement compréhensible par l'utilisateur, il est aussi facilement gérable grâce à sa petite taille. En plus de la possibilité de gérer la taille du profil construit, la consistance du profil garantit que notre modèle à une bonne qualité de prédiction.

Organisation du mémoire

Ce manuscrit est composé de trois principales parties :

1. La première partie comporte deux chapitres sur l'état de l'art en rapport avec le thème que nous abordons.
2. La deuxième partie est composée également de deux chapitres concernant notre contribution sur la construction de profil utilisateur ainsi que les expérimentations permettant de valider notre approche.
3. La troisième partie est constituée d' annexes ; Nous présentons dans cette partie les pré-traitements que nous avons effectués sur les données afin d'avoir un for-

mat utilisable par notre approche. Nous présentons également une documentation de l'implémentation que nous avons effectuée et des tableaux contenant en détail certains résultats de la partie expérimentale.

Première partie

Le chapitre 1 présente quelques notions sur les relations binaires ainsi que leurs propriétés. Nous présentons également les deux approches qui existent pour formuler les préférences élémentaires. Nous présentons également les méthodes qui existent pour combiner les préférences élémentaires dans un modèle. Les notions présentées dans ce chapitre sont essentielles pour comprendre les notions présentées dans le chapitre 2.
Le chapitre 2 est consacré à l'étude des méthodes d'apprentissage de préférences. Nous distinguons les méthodes qui utilisent l'approche qualitative ainsi que les méthodes qui utilisent l'approche quantitative. Dans chaque approche nous distinguons les méthodes qui modélisent les préférences contextuelles et les méthodes qui ne prennent pas en compte le contexte pour modéliser les préférences .

Deuxième partie

Le chapitre 3 présente les contributions que nous avons effectuées dans cette thèse, à savoir : l'algorithme ContPrefMiner pour extraire les règles de préférences contextuelles qui satisfont certaines contraintes, l'algorithme ProfMiner pour construire le profil utilisateur à partir des règles extraites par ContPrefMiner et les trois méthodes de classements : Best-rules 3.4.1, Weighted-voting 3.4.2 et Range Voting 3.4.3.

Le chapitre 4 présente les expérimentations que nous avons effectuées pour valider notre approche. Dans les expérimentations, nous avons testé les performances des algorithmes ContPrefMiner et ProfMiner ainsi que les trois méthodes de prédiction. Les expérimentations effectuées sur ContPrefMiner concernent la quantité des règles extraites en fonction d'un certain nombre de contraintes ainsi que le temps d'exécution de l'algorithme. Les expérimentations sur ProfMiner concernent la concision et l'intelligibilité du profil. Les expérimentations sur les méthodes de classement concernent la qualité de prédiction du profil ainsi que les comparaisons avec certaines méthodes de prédiction.

En conclusion générale, nous présentons le bilan des travaux que nous avons effectués dans le cadre de la construction automatique de profil utilisateur basée sur l'extraction de règles de préférences contextuelles. Nous dégageons ensuite les perspectives envisageables pour ces travaux.

Troisième partie

L'annexe A présente en détail le protocole de pré-traitement que nous avons effectué. Nous présentons le pré-traitement effectué depuis la base de données jusqu'à l'obtention du format de données supportable par notre algorithme. Ce format est le même que celui utilisé par SVMrank.

Dans l'annexe B, nous présentons la documentation de l'implémentation que nous avons effectuée. Nous présentons un exemple d'exécution présentant des informations sur la phase d'extraction de règles de préférences, sur la phase de construction de profil et sur la phase de prédiction.

Dans l'annexe C, nous présentons les tableaux contenant en détail les performances obtenues avec SVMrank en faisant varier le paramètre C qui permet le compromis entre la largeur de la marge et le nombre d'erreur de classement. Nous montrons également un tableau contenant les performances obtenues en utilisant la stratégie de classement Best-rule et la méthode Range voting.

Première partie

Etat de l'art

Table des matières

Introduction

L'objectif de cette partie est de présenter un état de l'art sur les méthodes d'apprentissage de préférences. Ces méthodes utilisent principalement deux approches pour modéliser les préférences utilisateurs : à savoir l'approche qualitative ou bien l'approche quantitative. Les méthodes qui utilisent l'approche qualitative apprennent une relation de préférence binaire pour modéliser les préférences tandis que les méthodes qui utilisent l'approche quantitative apprennent une fonction de score sur les données pour modéliser les préférences. Les notions de relation de préférences binaires et de fonction de score seront présentées dans le chapitre 1.

Le chapitre 1 présente quelques notions sur les relations binaires ainsi que leurs propriétés, nous présentons aussi les deux approches qui existent pour formuler les préférences élémentaires. Nous présentons également les méthodes qui existent pour combiner les préférences élémentaires dans un modèle. Les notions présentées dans ce chapitre sont essentielles pour comprendre les notions présentées dans le chapitre 2.

Le chapitre 2 est consacré à l'étude des méthodes d'apprentissage de préférences. Nous distinguons les méthodes qui utilisent l'approche qualitative ainsi que les méthodes qui utilisent l'approche quantitative. Dans chaque approche, nous distinguons les méthodes qui modélisent les préférences contextuelles et les méthodes qui ne prennent pas en compte le contexte pour modéliser les préférences .

Chapitre 1

Définition des concepts

L'objectif de ce chapitre est de présenter quelques notions de bases nécessaires à la compréhension de l'état de l'art présenté dans le chapitre 2. Dans la section 1.1, nous présentons quelques notions de bases nécessaires à la compréhension des préférences. Cette section est composée de trois sous-sections. Dans la sous-section 1.1.1, nous rappelons quelques définitions sur la notion de relation binaire. Puis dans la sous-section 1.1.2, nous présentons les modèles de préférences élémentaires : nous présentons les formulations élémentaires quantitatives et les formulations élémentaires qualitatives. Enfin, dans la sous-section 1.1.3, nous présentons quelques méthodes permettant de combiner les préférences élémentaires dans un modèle. Les méthodes de combinaisons de préférences élémentaires que nous présentons sont les suivantes : les combinaisons prioritaires, le modèle de préférence pareto et enfin les réseaux de préférences conditionnelles. Comme exemple d'application, nous utilisons des informations sur des films suivis par des utilisateurs. La table 1.1 rassemble les informations sur ces films.

1.1 Notion de base sur les préférences

Dans cette section, nous présentons les notions de base sur les relations de préférences. Il existe deux grandes approches pour modéliser les préférences : l'approche qualitative et l'approche quantitative. Pour modéliser les préférences, l'approche qualitative utilise les relations binaires [Cho03, Kie02, Wil04] tandis que l'approche quantitative utilise les fonctions de scores [AW00, KI04]. Etant donné que dans notre travail, nous considérons les modèles de préférences utilisés dans les bases de données relationnelles, nous adoptons les notions qui sont utilisées dans ce domaine. Nous utilisons :

– R(A_1,...,A_d) pour décrire un schéma relationnel avec d attributs $A_i, 1 \leq i \leq d$ où chaque attribut A_i prend sa valeur dans un domaine $dom(A_i)$.

	mid	Title	Year	Genre	Director	Actor	Lang	Dur
t_1	m_1	Jumanji	1995	Act	Johnston, J.	Williams, R.	Eng	102
t_2	m_2	Grumpier Old Men	1995	Com	Matthau, W.	Deutch, H.	Eng	103
t_3	m_3	Waiting to Exhale	1995	Dram	Whitaker, F.	Hines, G.	Eng	105
t_4	m_4	Father of the Bride Part II	1995	Com	Shyer, C.	Martin, S.	Eng	108
t_5	m_5	Heat	1995	Act	Mann, M.	Pacino, A.	Eng	110
t_6	m_6	Sabrina	1995	Dram	Pollack, S.	Ford, H.	Fr	115
t_7	m_7	Tom and Huck	1995	Adv	Hewitt, P.	Thomas, J. T.	Eng	117
t_8	m_8	Sudden Death	1995	Act	Hyams, P.	Van Damme, J.C.	Eng	120
t_9	m_9	GoldenEye	1995	Adv	Campbell, M.	Brosnan, P.	Span	121
t_{10}	m_{10}	American President	1995	Rom	Reiner, R.	Douglas, M.	Eng	124
t_{11}	m_{11}	North	1995	Dram	Reiner, R.	Wood, E.	Eng	126
t_{12}	m_{12}	Copycat	1995	Dram	Amiel, Jon	Mulroney, D.	Eng	128

– **Lang** : Language
– **Dur** : Duration
– **com** : comedy
– **Act** : Actor
– **Dram** : Drama
– **Adv** : Adventure
– **Rom** : Romance

TABLE 1.1 – Instance d'une relation films

– $\mathcal{A} = \{A_1, A_2, \ldots, A_d\}$ pour désigner l'ensemble des attributs de R et $dom(\mathcal{A}) = dom(A_i) \times \ldots \times dom(A_d)$ le domaine de ses valeurs,
– t pour désigner un tuple $(u_1, \ldots, u_d) \in dom(\mathcal{A})$ de R,
– r pour désigner une instance (i.e un ensemble de tuples) de R,
– $B \subseteq \mathcal{A}$ un sous-ensemble de l'ensemble des attributs,
– t[B] pour indiquer la projection de t sur B,

Cette section est organisée de la façon suivante : Nous rappelons les définitions basiques sur les relations binaires (sous-section 1.1.1) puis nous montrons les approches qui existent pour modéliser les préférences élémentaires (sous-section 1.1.2). Enfin, nous présentons quelques mécanismes pour les combiner (sous-section 1.1.3).

1.1.1 Définitions basiques

Nous allons rappeler ici quelques définitions sur les relations binaires. Dans le travail de [MOU07], l'auteur à effectué un état de l'art sur la personnalisation de requêtes dans le domaine des bases de données relationnelles.
Dans ce travail, [MOU07] a présenté quelques définitions sur les concepts de bases et relation de préférences. Nous réutilisons dans cette sous-section quelques définitions présentées dans ce travail. Ces définitions sont adaptées à notre travail puisque nous tra-

vaillons sur les modèles de préférences utilisés dans les bases de données relationnelles. Il est à noter que les adaptations du travail de [MOU07] au notre ne sont faites qu'à partir de la sous-section 1.1.2.

Définition 1 (Relation binaire)**.** *Une relation binaire $\Re$ sur un ensemble E est un sous-ensemble du produit cartésien E×E : c'est à dire un ensemble de couples (x,y) d'éléments de E.*

Dans ce qui suit, nous utilisons la notation $x\Re y$ pour indiquer que le couple (x,y) appartient à la relation $\Re$. Pour la suite, nous présentons un ensemble de propriétés qu'une relation binaire peut avoir. Une relation binaire est dite :

- réflexive si pour tout $x \in$ E, $x\Re x$
- irreflexive si pour tout $x \in$ E, $\neg(x\Re x)$
- symétrique si pour tout $x, y \in$ E, $x\Re y \Rightarrow y\Re x$.
- antisymétrique si pour tout $x, y \in$ E, $x\Re y$ et $y\Re x \Rightarrow x = y$
- asymétrique si pour tout $x, y \in$ E, $x\Re y \Rightarrow \neg(y\Re x)$.
- complète si pour tout $x, y \in$ E, $x\Re y$ ou $y\Re x$
- transitive si pour tout $x, y, z \in$ E, $x\Re y$ et $y\Re z \Rightarrow x\Re z$.
- négativement transitive si pour tout $x, y, z \in$ E $\neg(x\Re y)$ et $\neg(y\Re z) \Rightarrow \neg(x\Re z)$.

La table 1.2 présente des types de relations binaire ainsi que certaines de leurs propriétés.

Relation binaire	$=$	$\neq$	$>$	$<$	$\leq$	$\geq$
réflexive	oui	non	non	non	oui	oui
irréflexive	non	oui	oui	oui	non	non
symétrique	oui	oui	non	non	non	non
antisymétrique	oui	non	non	non	oui	oui
asymétrique	non	non	oui	oui	non	non
complete	non	non	non	non	oui	oui
transitive	oui	non	oui	oui	oui	oui
négativement transitive	non	oui	oui	oui	oui	oui

Table 1.2 – Types de relations binaires

Définition 2 (Relation d'indifférence)**.** *Etant donnée une relation $\Re$ sur un ensemble E, la relation d'indifférence notée $\sim_\Re$ est définie pour tout x,y ∈ E par : $\sim_\Re$ si $x\Re y$ et $y\Re x$.*

Il est à noter que la relation d'indifférence $\sim_\Re$ est symétrique. On constate aussi qu'avec la relation d'indifférence, les couples (x,y) et (y,x) appartiennent à la relation $\Re$.

Définition 3 (Relation d'incompatibilité)**.** *Etant donnée une relation $\Re$ sur un ensemble E, la relation d'incompatibilité notée $||_{\Re}$ est définie pour tout $x, y \in E$ par : $x||_R y$ si $\neg(x\Re y)$ et $\neg(y\Re x)$.*

On note que la relation d'incompatibilité $x||_{\Re}y$ est symétrique. Avec cette relation, les couples (*x*,*y*) et (*y*,*x*) ne sont pas dans la relation $\Re$. Par la suite, toute relation de préférence sera définie par une relation binaire $\Re$. La notion de préférence permet ainsi de classer les éléments d'un ensemble E par ordre de préférence.

Définition 4 (Relation de préférence)**.** *Une relation de préférence sur un ensemble d'éléments E est une relation binaire.*

Dans ce cas, étant donnée une relation de préférences notée $\succ_{\Re}$:

1. $x \succ_{\Re} y$ signifie que x est strictement préféré à y,
2. $x \sim_{\Re} y$ signifie que x et y sont également préférés, et enfin
3. $x||_{\Re}y$ signifie qu'il n'y a ni indifférence, ni préférence entre x et y et on dit que x et y ne sont pas comparables.

Définition 5 (Relation d'ordre)**.** *On appelle relation d'ordre sur un ensemble E toute relation binaire sur E qui est réflexive, antisymétrique et transitive.*

Si la relation d'ordre est complète, elle est dite ordre total sinon l'ordre est partiel.

Définition 6 (Préordre)**.** *Un préordre sur un ensemble E est une relation binaire réflexive et transitive.*

Á une relation d'ordre on peut associer une relation obtenue en ôtant de celle-ci les couples d'éléments identiques.

Définition 7 (Relation d'ordre strict)**.** *Une relation d'ordre strict sur un ensemble E est une relation binaire irréflexive et transitive. L'ordre strict est dit faible, s'il est partiel et négativement transitif.*

En général, une relation de préférence est une relation d'ordre partiel strict. Pour la suite, on utilisera le symbole $\succeq$ pour représenter une relation d'ordre et le symbole $\succ$ pour représenter une relation d'ordre strict. Néanmoins, le passage d'une relation d'ordre à une relation d'ordre strict est toujours possible et on peut l'avoir comme suit :

1. Pour une relation d'ordre $\succeq$, on définit la relation d'ordre strict associée $\succ$ par : pour tout x,$y \in$ E,$x \succ y \equiv x \succeq y$ et $x \neq y$.
2. pour une relation d'ordre strict $\succ$, on définit la relation d'ordre associée $\succeq$ par : pour tout x,$y \in$ E,$x \succeq y \equiv x \succ y$ ou $x = y$.

Pour ordonner les éléments d'un ensemble, on peut utiliser une fonction d'utilité pour que les éléments intéressants de l'ensemble reçoivent des valeurs supérieurs à ceux qui le sont moins.

Définition 8 (Fonction d'utilité)**.** *Une fonction d'utilité u est une fonction définie de : E vers [0,1].*

$$U : E \longmapsto [0,1]$$

$$x \longmapsto u(x)$$

Il est à noter que pour toute fonction d'utilité u, on peut associer une relation d'ordre total $\succeq_u$ définie pour tout x,$y \in$ E par : $x \succeq_u y$ si $u(x) \geq u(y)$. Dans cette représentation par une fonction d'utilité, deux éléments de même utilité sont considérés comme indifférents.

Exemple 1. *En considérant les données du tableau 1.1, un utilisateur peut fournir les scores suivant sur les tuples de la table u(t_1)=0,8, u(t_2)=0,7,u(t_3)=0,6, u(t_5)=0,6. Dans ce cas on a par exemple $t_1 \succ_u t_2$ car u(t_1)=0,8 > u(t_2)=0,7. Etant donné que les scores des tuples t_3 et t_5 sont les mêmes, t_3 et t_5 sont indifférents.*

On constate, qu'en face d'un grand nombre d'éléments, cette façon de spécifier directement les scores sur les tuples peut être fastidieux pour l'utilisateur. Nous présentons dans la sous-section suivante les deux grandes approches pour modéliser les préférences utilisateurs.

1.1.2 Représentation de préférences

Comme nous l'avons indiqué à l'introduction de cette sous-section, une préférence peut être représentée en utilisant la formulation qualitative ou bien en utilisant la formulation quantitative. Dans la sous-section précédente (cf. sous-section 1.1.1), nous avons défini de façon générique la notion de relation binaire et la notion de fonction de scores en utilisant un ensemble E.
Dans ce qui suit, nous considérons cet ensemble E comme étant l'ensemble des tuples $t = (u_1, u_2, ..., u_d), u_i \in dom(A_i)$ de $R(A_1, A_2, ...A_d)$. Etant donnée cette considération, une relation de préférence ($\succ_p$) sur un schéma R sera l'ensemble des couples de tuples (t_i, t_j) de r tandis qu'une fonction de score sera défini de $dom(\mathcal{A})$ vers [0,1]. Par conséquent $t_i \succ_p t_j$ signifiera que t_i est préféré à t_j sous $\succ_p$.

Cette sous-section est organisée de la façon suivante : Nous présentons quelques travaux sur la formulation de préférences qualitatives dans la sous-sous-section 1.1.2.1 puis

dans la sous-sous-section 1.1.2.2, nous présentons les travaux sur la formalisation quantitative et enfin nous effectuons une synthèse en présentant les avantages et les inconvénients qui existent sur l'utilisation de l'une ou de l'autre approche (sous-sous-section 1.1.2.3).

1.1.2.1 Formulation qualitative

Il existe un certain nombre de travaux dans la formulation de préférences qualitatives [Cho03, Kie02, Wil04]. Dans les travaux de [Cho03], l'auteur utilise les formules logiques pour exprimer les contraintes que deux tuples doivent satisfaire pour que l'un soit préféré à l'autre. La définition d'une formule logique est la suivante [SKP11] :

Définition 9. *Etant donné un schéma relationnel R et une instance r de R, une formule de préférence $PF(t_i, t_j)$ définit une relation de préférences $\succ_{PF}$ sur R, tel que pour chaque paire de tuples t_i, $t_j \in r$, $t_i \succ_{PF} t_j$, si et seulement si , $PF(t_i, t_j)$ est vrai.*
PF est une disjonction de conjonction de forme normale (DFN) des conditions de prédicats. $PF = (Cond_{11} \wedge ... \wedge Cond_{1m}) \vee ... \vee (Cond_{p1} \wedge ... \wedge Cond_{pb})$, où chaque prédicat $Cond_{xy}$ a la forme : $(i)t_i[A_v]\theta_v t_j[A_v]$ ou (ii) $t_k[A_v]\theta_v co$, où $t_i, t_j \in r$, $k \in \{i,j\}$, $A_v \in \mathcal{A}$,$co \in dom(A_v)$ et $\theta_v \in \{=, \neq, <, >, \geq, \leq\}$ si A_v est un attribut numérique et $\theta_v \in \{=, \neq\}$ sinon.

En se basant sur cette formule, il est possible d'exprimer le choix sur plusieurs tuples en utilisant une seule spécification de préférences.
Considérons la relation de préférence exprimée par la formule suivante :

Exemple 2. *Soient deux tuples t_i, t_j appartenant à la table 1.1 : $t_i \succ_{PF_1} t_j$, si et seulement si, (t_i[genre] = t_j[genre] $\wedge$ t_i[Language]='English' $\wedge$ t_j[Language]='French').*

Cette formule de préférence exprime la préférence (PF_1) d'un tuple t_i à un tuple t_j si et seulement si les deux tuples ont le même genre et que la langue du film t_i est l'Anglais et que la langue du film t_j est le Français. Les couples de films qui satisfont à cette formule de préférence sont : (t_3, t_6) , (t_{11}, t_6) , (t_{10}, t_6).
On constate qu'avec cette exemple, les prédicats ne sont liés que par des conjonctions. Dans l'exemple suivant, nous allons illustrer l'utilisation des disjonctions entre les prédicats.

Exemple 3. *Etant donnée deux tuples $t_i, t_j \in r$, $t_i \succ_{PF_2} t_j$, si et seulement si, (t_i[genre] = t_j[genre] $\wedge t_i$[genre] = 'drama' $\wedge t_i$[year] > t_j[year]) $\vee$ (t_i[genre] = t_j[genre] $\wedge t_i$[genre] = 'action' $\wedge t_i$[actor]= 'Jean Claude Van damme' $\wedge t_i$[actor]$\neq$ t_j[actor]).*

Cette formule de préférence exprime la préférence (PF_2) : Préférer les films de drames récemment produits ou les films d'action dont l'acteur est 'Jean Claude Van damme'. Les couples de films qui satisfont à cette formule de préférence sont : (t_8, t_5),(t_8, t_1).
Puisque tous les films dans la table 1.1 sont produits dans la même année, il n'y a pas de films qui est plus récent qu'un autre. Le premier élément de la disjonction de préférence ne renvoie pas donc de résultats. Les préférences de t_8 sur t_5 et t_1 sont obtenues grâce au second élément de la disjonction.
D'autres auteurs ont travaillé sur la formulation qualitative des préférences. Dans [Wil04], les auteurs utilisent des règles de préférences conditionnelles pour modéliser les préférences. La forme générale d'une règle de préférence conditionnelle est :

"si [conjonction de conditions élémentaires] alors [décision]"

La partie condition indique dans quelle condition la règle est appliquée et la partie décision montre le choix à opérer sur les valeurs si la partie condition est satisfaite. Le formalisme proposé par Wilson [Wil04] est défini de la façon suivante :

Définition 10. *Soit $\mathcal{A}$ un ensemble d'attributs, $U \subseteq \mathcal{A}$, a_{i1} et $a_{i2} \in dom(A_i)$ $A_i \notin U$. Une règle de préférence contextuelle est sous la forme $u : a_{i1} \succ a_{i2}[W]$ tel que $u \in dom(U)$ et W un sous-ensemble de $S = \mathcal{A} - (U \cup \{A_i\})$.*

En considérant la règle de préférence $p = u : a_{i1} \succ a_{i2}[W]$, en utilisant la formule de préférence de la Définition 9, cette règle de préférence peut être représentée par le prédicat suivant : $t_i \succ_p t_j$ ssi $t_i[S-W] = t_j[S-W] \wedge t_i[U] = t_j[U] = u \wedge t_i[A_i] = a_{i1} \wedge t_j[A_i] = a_{i2}$.
De façon intuitif cette règle de préférence contextuelle indique que tout tuple t_i contenant u et a_{i1} est préféré à tout tuple t_j contenant u et a_{i2} indépendamment des valeurs dans $dom(W)$, sachant que les valeurs dans $dom(S-W)$ sont les mêmes pour les deux tuples.
En utilisant le formalisme proposé par Wilson (cf. Définition 10), la préférence (PF_1) représentée par la formule de préférence de l'exemple 2 est représentée par la règle de préférence suivante :

Exemple 4.

$$\top : Language =' English' \succ Language =' French'[\{Title, Year, Director, Actor\}]$$

où $\top$ ne représente aucun attribut.

Cette règle indique la préférence des films Anglais aux films Français sans tenir compte du titre du film, de l'année de production, du directeur du film ou bien de l'acteur.
Dans les travaux de [Kie02], l'auteur propose un langage formel pour modéliser les préférences qui forment une relation d'ordre partiel strict. Pour cela, l'auteur propose un

certain nombre de constructeurs de base ; Dans ce travail, nous présentons que deux constructeurs de base à savoir : le constructeur de base positif nommé POS et le constructeur de base négatif nommée NEG.
Un constructeur de base a un ou plusieurs arguments, le premier argument représente un attribut tandis que les autres arguments caractérisent la relation d'ordre partiel strict se référant au nom de l'attribut. Un constructeur POS est défini de la façon suivante :

Définition 11 (Préférence POS)**.** *Soient R un schéma relationnel,r une instance de R, POS-set un ensemble fini de valeurs préférées d'un attribut A de domaine $dom(A)$. La preference P est une POS(A, POS-set) préférence si $\forall\ t_i, t_j \in r$, $t_i \succ_P t_j$, si et seulement si, $t_i[A] \in$POS-set$\wedge t_j[A] \notin$POS-set, où POS-set $\subseteq dom(A)$.*

De façon intuitive, cela signifie que les valeurs de l'attribut qui sont dans POS-set sont préférées à toutes les autres valeurs de l'attribut qui ne sont pas dans POS-set.

Exemple 5. *Supposons qu'un utilisateur exprime ses préférences sur les films de l'acteur 'Jean Claude Van Damme'. Cette préférence peut être exprimée par le constructeur de préférence P=POS({Actor}, {Van Damme, J.C.}).*

Cette préférence indique que 'Van Damme, J.C.' est préféré à tous les autres acteur de dom(Acteur). La relation de préférence notée $\succ_P$ peut être définie en utilisant la formule de préférence suivante : $t_i \succ_{PF} t_j$ ssi (t_i[actor]= 'Jean Claude Van damme' $\wedge t_i$[actor]$\neq t_j$[actor]). Les couples de films qui satisfont cette relation de préférence sont : $(t_8, t_1), (t_8, t_2), (t_8, t_3), (t_8, t_4), (t_8, t_5), (t_8, t_6), (t_8, t_7), (t_8, t_9), (t_8, t_{10}), (t_8, t_{11}), (t_8, t_{12})$. Ce qui signifie que le film le plus préféré de la table 1.1 est : *Sudden Death*.
De façon similaire une NEG préférence est défini de la façon suivante :

Définition 12 (Préférence NEG)**.** *Soient R un schéma relationnel,r, un instance de R, NEG-set un ensemble fini de valeurs non préférées d'un attribut A de domaine $dom(A)$. La préférence P est une NEG(A, NEG-set) préférence si $\forall\ t_i, t_j \in r$, $t_i \succ_P t_j$, si et seulement si, $t_i[A] \notin$NEG-set $\wedge t_j[A] \in$ NEG-set, où NEG-set $\subseteq dom(A)$.*

De façon intuitive, cela signifie que les valeurs de l'attribut qui ne sont pas dans NEG-set sont préférées à toutes les autres valeurs de l'attribut qui sont dans NEG-set.

Exemple 6. *Supposons qu'un utilisateur exprime ses préférences sur les films de tout acteur sauf ceux de l' acteur 'Jean Claude Van Damme'. Cette préférence peut être exprimée par le constructeur de préférence P=NEG({Actor}, {Van Damme, J.C.}).*

Cette préférence indique que les films des autres acteurs sont préférés à ceux de 'Jean Claude Van Damme'. La relation de préférence notée $\succ_p$ peut être définie par les formules

de préférence de la relation suivante. $t_i \succ_{PF} t_j$ ssi (t$_j$[actor]= 'Jean Claude Van damme' $\wedge t_i$[actor]$\neq t_j$[actor]). Les couples de films qui satisfont cette relation de préférence sont les suivants :

$$(t_1,t_8),(t_2,t_8),(t_3,t_8),(t_4,t_8),(t_5,t_8),(t_6,t_8),(t_7,t_8),(t_8,t_9),(t_{10},t_8),(t_{11},t_8),(t_{12},t_8)$$

Dans ce cas, le film le moins préféré de la table 1.1 est Sudden Death.
On note que toutes les formulations de préférences définies dans cette sous-sous-section utilisent les relations de préférences binaires pour exprimer les préférences. Nous avons reformulé les propositions de [Kie02]et de [Wil04] grâce la proposition de [Cho03] ce qui montre que la proposition de [Cho03] est plus générale que les propositions de Kiebling [Kie02] et de Wilson [Wil04].

1.1.2.2 Formulation quantitative

Dans l'approche quantitative, les préférences sont exprimées en utilisant des fonctions qui associent des scores numériques à chaque tuple de la base de données [AW00]. Les scores expriment combien les tuples sont importants. Pour une fonction de score u (cf. Définition 8), un tuple t_i est préféré à un t_j ssi $u(t_i) > u(t_j)$.

Exemple 7. *Etant donnée une fonction de score $u(t_i)$=$0,001 \times t_i$[durée], les tuples t_1,t_2 et t_3 de la table1.1ont respectivement les scores 0,102, 0,103 et 0,105. Par conséquent t_1 est préféré à t_2 qui est préféré à t_3.*

Dans les travaux de [KI04], les auteurs proposent une définition en intention des fonctions de scores. Le principe est de fixer des contraintes que les tuples doivent satisfaire puis d'assigner un score de préférence à ses tuples. Les scores de préférences sont généralement dans l'intervalle [0,1]. Avec ce principe, une préférence utilisateur peut être définie de la façon suivante.

Définition 13. *Soit $R(A_1,...,A_d)$ un schéma relationnel et $dom(A_i)$ le domaine de l'attribut $A_i, 1 \leq i \leq d$. Une preference P sur R est une paire ($Condition_P$, $Score_P$) où*

1. *$Condition_P$ est de la forme $A_{n_1}\theta_{n_1}a_{n_1} \wedge ... \wedge A_{n_k}\theta_{n_k}a_{n_k}$ et spécifie une conjonction d'une simple condition de selection sur la valeur $a_{n_m} \in dom(A_{n_m})$ des attributs $A_{n_m}, n_m \in \{1,d\}$, où $\theta_{n_m} \in \{=,>,<,\leq,\geq,\neq\}$ pour les attributs numériques $\theta_{n_m} \in \{=,\neq\}$ pour les attributs catégoriels.*
2. *$Score_P$ appartient à un domaine numérique prédéfini.*

Intuitivement, cette définition indique que le(s) tuple(s) qui satisfont la condition $Condition_P$, se voient associer $Score_P$. Si un tuple satisfait les conditions de plusieurs préférences P, alors les scores qui lui sont assignés sont combinés pour former qu'un seul score[SKP11]. Si score(t) est le score assigné au tuple t, alors un tuple t_i est préféré à un tuple t_j, noté $t_i \succ t_j$ si et seulement si $score(t_i) > score(t_j)$.

Exemple 8. *La préférence (Genre = 'action' ∧ Actor='Van Damme J.C', 0.9) montre un grand intérêt sur les films dont l'acteur est 'Van Damme J.C', tandis que la préférence (Language = 'French', 0.1) indique un faible intérêt pour les films Français.*

Dans ce qui suit nous allons présenter les avantages et les inconvénients des formulations qualitatives et quantitatives.

1.1.2.3 Formulation qualitative VS Formulation quantitative

Les préférences qualitatives sont exprimées de façon relative en comparant les tuples en question[Cho03, Kie02]. Tandis qu'avec les approches quantitatives, les préférences sont exprimées de façon absolue : directement sur les tuples en question[AW00, KI04]. La formulation qualitative est plus générale que la formulation quantitative : toutes les relations de préférences ne peuvent pas être exprimées avec des fonctions de scores ou à travers un degré d'intérêt.

Exemple 9. *En considérant l'exemple 2, pour rappel les couples de films qui satisfont le prédicat*
$PF(t_i, t_j)$=(t_i[genre] = t_j[genre] ∧ t_i[Language]='English' ∧ t_j[Language]='French')
sont les suivants : (t_3, t_6) , (t_{11}, t_6), (t_{10}, t_6).

On constate dans cet exemple qu'il n'y a pas de préférence entre les tuples t_3, t_6, t_{11}, t_{10} et les autres tuples de la table 1.1. Par exemple, il n'y a pas de préférence entre ces tuples et le tuple t_4. Par conséquent, en utilisant la formulation quantitative avec les fonctions de scores, le score de t_4, t_6 ,t_{11}, t_{12} t_3 doit être le même. Si les scores des cinq tuples sont égaux alors les tuples t_6 ,t_{11}, t_{12} t_3 sont incomparables, ce qui est faux. Par conséquent, il n'y a pas de fonction de score qui capture cette relation de préférence qualitative.

Toutefois, avec les préférences qualitatives, il n'est pas possible de connaître combien un item est préféré par rapport à un autre. Par exemple, il n'est pas possible d'exprimer des préférences de la forme : 'j'aime beaucoup plus les comédies que les drames et encore plus les films d'action que les comédies.

Dans cette section, nous avons présenté les deux approches qui existent pour formuler les préférences élémentaires. Dans la section suivante, nous présentons les techniques pour combiner les préférences élémentaires de l'utilisateur dans un modèle.

1.1.3 Combinaison de préférences

Dans les sections précédentes, nous avons présenté les approches qui existent pour modéliser les préférences individuelles. Nous allons maintenant voir comment combiner ces préférences dans un modèle. Nous présentons dans la sous-sous-section 1.1.3.1 les modèles de préférences prioritaires puis dans la sous-sous-section 1.1.3.2, nous présentons les modèles de préférences Pareto. Enfin dans la sous-sous-section 1.1.3.3, nous présentons les réseaux de préférences conditionnelles.

1.1.3.1 Composition de préférences prioritaires

Etant données deux relations de préférences $\succ_{p_x}$ *et* $\succ_{p_y}$, dans la relation de préférence prioritaire, une des relations de préférence est prioritaire sur l'autre : soit $\succ_{p_x}$ est prioritaire sur $\succ_{p_y}$ soit c'est le contraire. La définition suivante présente la composition de préférences prioritaire sur deux relations de préférences $\succ_{p_x}$ et $\succ_{p_y}$.

Définition 14. *Soient $\succ_{p_x}$ et $\succ_{p_y}$ deux relations de préférences définies sur le même schéma relationnel R. La relation de préférence prioritaire $\succ_{p_x}$ & $\succ_{p_y}$ est définie sur R par : quelque soit t_i, t_j de R, $t_i \succ_{p_x}$ & $\succ_{p_y}$ t_j si et seulement si $(t_i \succ_{p_x} t_j) \vee (t_i \sim_{p_x} t_j \wedge t_i \succ_{p_y} t_j)$.*

La définition intuitive de la composition prioritaire est la suivante : utiliser $\succ_{p_y}$ si et seulement si $\succ_{p_x}$ n'est pas applicable.

Exemple 10. *Soient deux relations de préférences $\succ_{p_x}$ et $\succ_{p_y}$ tel que : $t_i \succ_{p_x} t_j$, si et seulement si, $t_i[genre] =' Drama' \wedge t_j[genre] =' Comedy'$. Et $t_i \succ_{p_y} t_j$, si et seulement si, $t_i[Language] =' French' \wedge t_j[Language] =' English'$.*
La relation de préférence prioritaire $\succ_{p_x}\&\succ_{p_y}$ peut être définie comme : $t_i \succ_{p_x}\&\succ_{p_y} t_j$, si et seulement si, $(t_i[genre] =' Drama' \wedge t_j[genre] =' Comedy') \vee (t_i[genre] \neq' drama' \wedge t_i[Language] =' French' \wedge t_j[Language] =' English') \vee (t_j[genre] \neq' Comedy' \wedge t_i[Language] =' French' \wedge t_j[Language] =' English')$.

En considérant la relation de préférence prioritaire $\succ_{p_x}\&\succ_{p_y}$, les couples de films qui satisfont cette relation de préférence sont : $(t_6, t_4), (t_6, t_2)$.

En appliquant la composition prioritaire sur des schémas relationnels différents, on obtient la relation de préférence Lexicographique [SKP11].

Définition 15. *Soient $\succ_{p_x}$ et $\succ_{p_y}$ deux relations de préférences définies sur les schémas relationnels R et R' dont les attributs prennent leurs valeurs dans dom(A) et dom(A') respectivement. La relation de préférence lexicographique $\succ_{p_x}\&\succ_{p_y}$ définie sur le produit cartésien $R\times R'$ est un sous ensemble de $dom(A)\times dom(A')$ tel que $(t_i, t'_i) \succ_{p_x}\&\succ_{p_y}(t_j, t'_j)$ si et seulement si $(t_i \succ_{p_x} t_j) \vee (t_i \sim_{p_x} t_j \wedge t'_i \succ_{p_y} t'_j)$, où t_i et t_j sont des tuples de R et t'_i et t'_j des tuples de R'.*

1.1.3.2 Modèle pareto

Dans la composition pareto, les relations de préférences dans le modèle ont une importance égale. La définition suivante présente la composition de préférence pareto sur deux relations de préférences $\succ_{p_x}$ et $\succ_{p_y}$.

Définition 16. *Soient $\succ_{p_x}$ et $\succ_{p_y}$ deux relations de preferences sur le même schema relationnel R. La relation de composition pareto $\succ_{p_x}\otimes\succ_{p_y}$ est définie sur la relation R comme suit : $\forall\, t_i, t_j$ de R, $t_i\succ_{p_x}\otimes\succ_{p_y}t_j$ ssi $(t_i \succ_{p_x} t_j\wedge\neg(t_j \succ_{p_y} t_i)) \vee (t_i \succ_{p_y} t_j \wedge\neg(t_j \succ_{p_x} t_i))$.*

Il est à noter que pour deux tuples t_1 et t_2 et une relation de préférence $\succ_p$, $\neg(t_1 \succ_p t_2) \equiv (t_2 \succ t_1) \vee (t_1 \sim_p t_2)$. De façon intuitive sous la composition pareto, un tuple t est préféré à un tuple u si t est au moins aussi préféré que u pour une relation de préférences et que t est absolument plus préféré que u pour l'autre relation de préférences.

Exemple 11. *La préférence pareto $p_x\otimes p_y$ (p_x et p_y sont définies dans l'exemple 10) peut être défini comme : $t_i\succ_{p_x}\otimes\succ_{p_y}t_j$, si et seulement si, ($t_i$ [genre] = 'drame' $\wedge$ $t_j[genre] =' comedy'\wedge$ $t_i[Language] =' French'\wedge$ $t_j[Language] =' English')\vee$ ($t_i[Language] =' French'\wedge$ $t_j[Language] =' English'\wedge$ $t_j[genre]\neq$ 'Drame') $\vee$ ($t_i[Language] =' French'\wedge$ $t_j[Language] =' English'\wedge$ $t_i[genre]\neq$ 'comedy').*

Le modèle de préférence pareto peut être aussi appliqué sur des relations qui sont définies sur des schémas différents [SKP11]. Soient $\succ_{p_x}$ et $\succ_{p_y}$ deux relations de préférences définies sur les schémas relationnels R et R' avec des domaines d'attributs dom(A) et dom(A') respectivement. La relation de préférence pareto multi-dimensionnelle $\succ_{p_x}\otimes\succ_{p_y}$ définie sur le produit cartésien R × R' est un sous ensemble de dom(A)× dom(A') tel que $(t_i, t'_i) \succ_{p_x}\otimes\succ_{p_y}$ (t_j, t'_j) si et seulement si $(t_i \succ_{p_x} t_j\wedge \neg(t'_j \succ_{p_y} t'_i)) \vee (t'_i \succ_{p_y} t'_j \wedge\neg(t_j \succ_{p_x}$

t_i)) où t_i et t_j sont des tuples de R et t'_i et t'_j sont des tuples de R'.

1.1.3.3 Réseaux de preferences conditionnelles (CP-nets)

Les réseaux de préférences conditionnelles CP-nets permettent de représenter de façon compacte et intuitive les relations de préférences. C'est un formalisme graphique qui permet de modéliser les préférences conditionnelles de façon qualitative [BBD+03].

Un CP-net sur un ensemble d'attributs $\mathcal{A} = \{A_1, ..., A_d\}$ est un graphe direct dans lequel il existe un noeud pour chaque attribut A_i dans $\mathcal{A}$. Si une flèche d'un attribut A_j vers un attribut A_i existe, alors A_j est dit parent de A_i, on note Pa(A_i) tous les parents d'une variable A_i.
Si $Pa(A_i)$ est l'ensemble des parents de A_i, les préférences sur l'attribut A_i dépendent de celles sur les attributs parents de A_i .

Les tables de préférences conditionnelles (CPT) décrivent les préférences sur les valeurs A_i en se basant sur la combinaison des valeurs des parents de A_i.
Une table de préférences conditionnelle de A_i, noté CPT(A_i), contient un ensemble de règles de préférences de la forme $z_i : a_{i1} \succ a_{i2}$; où $z_i \in$dom(Pa(A_i)) et a_{i1}, a_{i2} appartiennent à $dom(A_i)$.
Ces règles de préférences sont interprétées de la façon suivante : étant donnée z_i, a_{i1} est strictement préféré à a_{i2} (*ceteris paribus*).
En d'autres termes, cette règle de préférence indique qu'un tuple contenant z_i et a_{i1} est préféré à un tuple contenant z_i et a_{i2} ssi les valeurs des deux tuples sur les autres attributs sont les mêmes.
Il est à noter que ces règles de préférences sont une spécialisation des règles de Wilson présentées dans la Définition 10. Il suffit de constater que dans les règles présentées par les CP-nets, Pa(A_i)=U et $W = \emptyset$. Nous allons ci après introduire la définition formelle d'un CP-net [BBD+03].

Définition 17. *Un CP-net sur les attributs $\mathcal{A} = \{A_1, ..., A_d\}$ est un graphe orienté sur les attributs $\{A_1, A_2, ..., A_d\}$ où chaque noeud, A_i, est annoté avec une table de préférence conditionnelle, CPT(A_i). Chaque table de préférence conditionnelle CPT(A_i) associe un ordre total $\succ^i_{z_i}$ pour chaque instanciation z_i de Pa(A_i).*

Exemple 12. *En considérant la relation movies= {Genre, Director} tel que : $dom(Genre) = \{comedy, drama\}$ et $dom(Director) = \{W.Allen, M.Curtiz\}$. Considérons l'expression de préférence suivant :*
Préférer les films de comédie aux films de drames et préférer un film dirigé par W.Allen

à un film dirigé par M. Curtiz pour les films de comédie et pour les films de drames, préférer les films dirigés par M. Curtiz à ceux dirigés par W. Allen.

Le CP-net correspondant à cet exemple est représenté par la figure 1.1 ; l'élément le

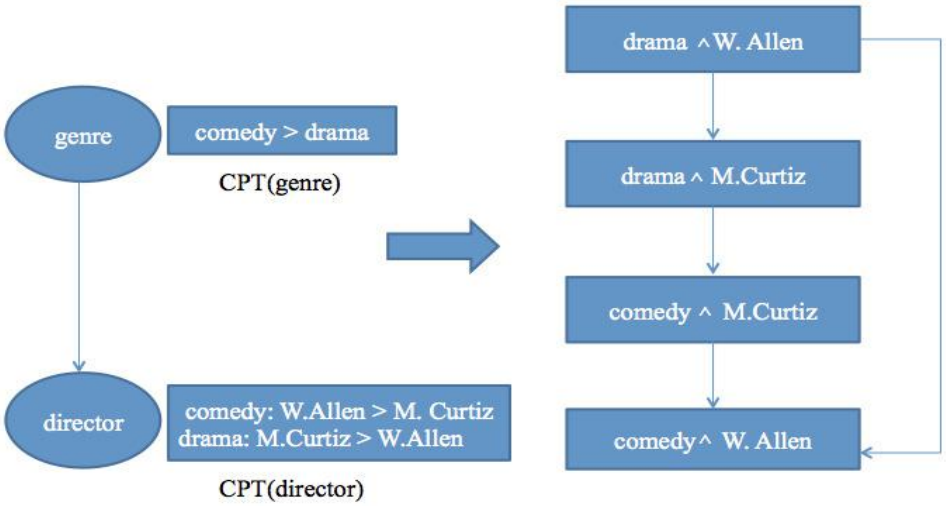


FIGURE 1.1 – (a) CP-Nets correspondant aux films ; (b) le graphe de préférences induit.

plus au dessus du graphe est l'élément le moins préféré tandis que l'élément le plus en bas du graphe est le meilleur élément. En considérant ce CP-net, l'élément (drama,W.Allen) est le moins préféré tandis que l'élément (Comedy,W.Allen) est le plus préféré.
Si on note N ce CP-net sur les attributs Genre et Director, N contient l'ensemble de préférences conditionnelles suivant :

$$N = \{comedy \succ drama, comedy : W.Allen \succ M.Curtiz, drama : M.Curtiz \succ W.Allen\}$$

Ces règles de préférence sont tout simplement les contraintes exprimées par les tables de préférences conditionnelles associés aux noeuds du CP-net.
Nous allons maintenant montrer comment comparer deux transactions en utilisant un CP-net.

Définition 18. *Soit N un CP-net sur $\mathcal{A}$, $A_i \in \mathcal{A}$ un attribut et $Pa(A_i) \subset \mathcal{A}$ les parents de A_i dans $\mathcal{A}$. Soit $Y = \mathcal{A} - (Pa(A_i) \cup \{A_i\})$. Soit $\succ^i_{z_i}$ l'ordre sur $dom(A_i)$ induit par $CPT(X_i)$ pour chaque instanciation z_i dans $dom(Pa(A_i))$ des parents de A_i. Finalement, soit $\succ$ une relation de préférences sur $dom(\mathcal{A})$. Une relation de préférences $\succ$ satisfait $\succ^i_{z_i}$ ssi pour chaque $y \in dom(Y)$ et chaque a_{i1}, $a_{i2} \in dom(A_i)$, si $ya_{i1}z_i \succ ya_{i2}z_i$ alors*

$a_{i1} \succ^{i}_{z_i} a_{i2}$.
Une relation de préférences $\succ$ satisfait la table de préférences conditionnelles CPT(A_i) ssi elle satisfait $\succ^{i}_{z_i}$ pour chaque z_i dans $dom(Pa(A_i))$. Une relation de préférences $\succ$ satisfait le CP-net N ssi elle satisfait CPT(A_i) pour chaque A_i.
Un CP-net est satisfiable ssi il existe une relation de préférences $\succ$ qui le satisfait.

Par conséquent, un réseaux N est satisfait par $\succ$ ssi $\succ$ satisfait chaque préférence conditionnelle exprimée dans les tables de préférences conditionnelles de N sous les conditions *ceteris-paribus*. [BBD+03] montre que tout CP-net acyclique est satisfiable.

Exemple 13. *La relation de préférences $\succ$ qui induit l'ordre :{comedy $\wedge$ W.Allen $\succ$ comedy $\wedge$ M.curtiz $\succ$ drama $\wedge$ M.curtiz $\succ$ drama $\wedge$ W.Allen} est la seule qui satisfait le CP-net de la figure 1.1.*

L'implication dans les CP-net est définie de la façon standard.

Définition 19. *Soit N un CP-net sur $\mathcal{A}$ et t_i,$t_j \in dom(\mathcal{A})$ deux tuples, N implique $t_i \succ t_j$ (i.e que l'objet t_i est préféré à t_j) noté $N \models t_i \succ t_j$ ssi $t_i \succ t_j$ est vrai dans toute relation de préférence qui satisfait N.*

Exemple 14. *Le CP-net de la figure 1.1 implique que comedyW.Allen $\succ$ comedyM.Curtiz puisque Cette préférence est vraie dans la seule relation de préférence qui satisfait le CP-net.*

Les préférences exprimées en utilisant le formalisme des CP-nets sont des préférences conditionnelles sous l'hypothèse *ceteris paribus*. La préférence exprimée sur la valeur d'un attribut dépend de la valeur d'autres attributs ; avec les CP-nets,
il n'y a pas de préférences contradictoires : il n'est pas possible d'appliquer deux règles en même temps.
Néanmoins, le pouvoir d'expression des CP-nets est limité, il n'est pas possible de représenter toutes les préférences. Cependant, pour les utilisateurs, il est plus facile d'utiliser le principe *ceteris paribus*.
Le formalisme proposé par Wilson [Wil04] constitue une extension des CP-nets, ce formalisme permet de manipuler des expressions plus générales. Pour rappel, les règles de Wilson sont sous la forme $u : x \succ x'[W]$ (cf. Définition 10). Il est montré que ces expressions généralisent celles des CP-nets qui sont obtenues si W= $\emptyset$.
En considérant l'exemple suivant :

Exemple 15. *Préférer les films de comédies aux films de drames sans tenir compte du directeur du film.*

Cette expression de préférence n'est pas représentable avec les CP-nets puisqu'ils utilisent le principe du *ceteris paribus*. Par contre en utilisant le formalisme de Wilson, cette préférence peut être représentée de la façon suivante. $\top : comedy \succ drame[Director]$.

1.2 Conclusion

Dans ce chapitre, nous avons présenté la définition de quelques concepts fondamentaux sur la notion de préférences utilisateur. Nous avons commencé par rappeler quelques éléments sur la théorie des ensembles (cf. sous-section 1.1.1). Puis nous avons présenté comment formuler les préférences avec les deux principales approches qui existent dans la littérature (cf. sous-section 1.1.2). Nous avons montré que la formulation qualitative [Cho03, Kie02] utilise les relations binaires tandis que la formulation quantitative [AW00, KI04] utilise les fonctions de scores pour modéliser les préférences. Nous avons ensuite montré dans la partie synthèse (cf. sous-sous-section 1.1.2.3) que la formulation qualitative est plus générale que la formulation quantitative : toute relation de préférence ne peut être exprimée par une fonction de score ou à travers un degré d'intérêt.

Apres avoir présenté les préférences élémentaires (cf. sous-section 1.1.2), nous avons montré comment combiner ces préférences dans un modèle (cf. sous-section1.1.3). Dans la sous-sous-section 1.1.3.1, nous avons présenté les combinaisons prioritaire. Pour rappel, avec la combinaison prioritaire, une préférence est prioritaire. Dans la sous-sous-section 1.1.3.2, nous avons présenté les modèles de préférences Pareto. Avec les modèles de préférence Pareto, toutes les préférences qui sont dans le modèle sont d'importance égale. Enfin, dans la sous-sous-section 1.1.3.3, nous avons présenté les modèles de préférences conditionnelles en présentant notamment les réseaux de préférences conditionnelles[BBD+03] ainsi que leurs généralisation proposés par Wilson[Wil04].

Dans le chapitre suivant, nous allons présenter les problématiques d'apprentissage de préférences ainsi que quelques travaux qui ont été présentés dans ce sens.

Chapitre 2

Apprentissage pour le classement

Dans le chapitre 1 nous avons présenté les deux formalismes qui existent pour modéliser les préférences utilisateurs à savoir : la formulation qualitative et la formulation quantitative. Nous avons montré que dans la formulation qualitative, les préférences sont représentées par une relation binaire [Cho03, Kie02], tandis que dans la formulation quantitative les préférences sont représentées par une fonction de score[AW00, KI04].
De façon similaire, il existe naturellement deux approches pour apprendre les préférences : l'approche qualitative et l'approche quantitative. Les travaux qui apprennent les relations de préférences binaires utilisent l'approche qualitative [HEK03, Jea08, CKL+10, GLS14, DKSS11], tandis que ceux qui apprennent une fonction de score pour modéliser les préférences utilisent l'approche quantitative [Joa06, FISS03, XL07, BSR+05, DKSS11].

Dans ce chapitre, nous nous intéressons sur un domaine qui a retenu beaucoup d'attentions dans le domaine de la recherche d'informations et dans le domaine de l'apprentissage machine (ou *Machine Learning* en anglais) [FH10] : l'apprentissage pour le classement (ou *Learning to rank* en anglais). Le problème d'apprentissage pour le classement consiste à apprendre un modèle de préférences sur des données d'apprentissages puis d'utiliser le modèle appris pour prédire les préférences sur de nouvelles données (données test) ou sur les mêmes données mais dans un contexte diffèrent [FH10, DKSS11].

Ce chapitre est organisé de la façon suivante : Nous commençons par présenter les problématiques d'apprentissage pour le classement (cf. section 2.1). Dans la section 2.3, nous présentons les deux grandes approches qui existent dans le domaine de l'apprentissage pour le classement : à savoir l'approche qualitative et l'approche quantitative. Pour bien étudier les méthodes que nous présentons dans la section 2.3, nous avons défini des critères d'études dans la section 2.2. Dans la section 2.4, nous effectuons une synthèse des travaux que nous avons présentés.

2.1 Problème de classement

Le classement (ou *ranking* en anglais) est l'un des problèmes fondamental dans le domaine de la recherche d'information et dans le domaine de l'apprentissage machine. Le problème consiste à classer un ensemble d'éléments par ordre de préférences.

Nous avons vu dans le chapitre 1 que les préférences sur les éléments peuvent être exprimées en utilisant l'approche qualitative ou en utilisant l'approche qualitative. Par conséquent, le problème de classement revient alors à classer des éléments en se basant sur une relation binaire ou sur des scores individuels associés à chaque élément.

Exemple 16. *La figure 2.1 représente un classement des documents représentants les réponses à la requête ≪learning to rank≫ exprimée sur le moteur de recherche de Google.*

L'apprentissage pour le classement utilise les techniques de l'apprentissage machine pour résoudre des problèmes de classements. Son objectif est d'apprendre un modèle de préférences sur les données d'apprentissage et utiliser ce modèle pour prédire un classement sur de nouvels objets ou les mêmes objets mais dans un contexte différent [FH10]. La qualité de prédiction du modèle appris pourra être évalué en utilisant des mesures de performances. L'apprentissage pour le classement peut être utilisé dans plusieurs domaines :

- Recherche d'information,
- Traitement automatique de la langue naturelle
- Data Mining.

Les applications typiques de l'apprentissage pour le classement sont :

- Recherche de documents
- Filtrage collaboratif
- Résumé de texte.

Dans les sous-sections suivantes, nous présentons les trois problèmes de classements qui existent dans la littérature [FH10]. Nous allons voir que selon le type de problème de classement traité, nous avons des formes de données différentes et pouvons utiliser une approche qualitative ou une approche quantitative pour résoudre le problème.
Pour illustrer les trois problèmes de classements, nous utilisons l'exemple suivant :

Exemple 17. *Considérons deux utilisateurs qui expriment leurs préférences sur les trois premiers films de la table 1.1 du chapitre 1. Les informations sur les utilisateurs, les préférences quantitatives exprimées sur les films ainsi que les paires de préférences qui en résultent sont stockées dans les tables de schémas (Userid, Genre, Age, Adresse), (Userid, Movid, Note) et (Userid, Paires).*

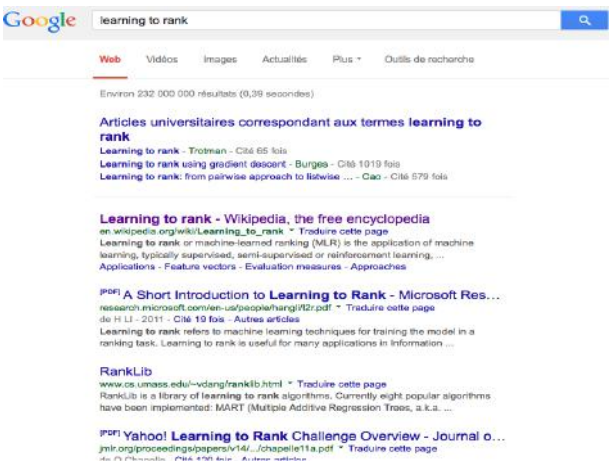


FIGURE 2.1 – Exemple de classement des résultats d'une requête

Uid	Genre	Age	Adr
u_1	M	25	Paris
u_2	F	25	Nice
u_3	F	25	Lyon

(a)

Uid	Movid	Note
u_1	m_1	Faible
u_1	m_2	Moyen
u_1	m_3	Fort
u_2	m_1	Fort
u_2	m_2	Moyen
u_2	m_3	Faible

(b)

Uid	Paires
u_1	$\langle m_3, m_2 \rangle$
u_1	$\langle m_3, m_1 \rangle$
u_1	$\langle m_2, m_1 \rangle$
u_2	$\langle m_1, m_2 \rangle$
u_2	$\langle m_1, m_3 \rangle$
u_2	$\langle m_2, m_3 \rangle$

(c)

TABLE 2.1 – (a) Utilisateurs (b) Préférences quantitatives (c) Préférences qualitatives

2.1.1 Classement de labels

Le classement de labels (ou *labels ranking* en anglais) est un problème d'apprentissage pour le classement dont la forme des données est constituée par des paires de labels $l_i \succ_x l_j$ indiquant que pour l'objet $x \in \mathcal{X}$, le label l_i est préféré au label l_j.
Le problème de classement consiste à ordonner les labels pour chaque objet x. Les labels peuvent être vus comme des entités sans contenu, tandis qu'un objet peut être vu comme un tuple d'une base de données.
Le problème du classement de labels peut être résolu en utilisant une approche quantitative [DMS04, HpRZ02, HpRZ03], en apprenant une fonction de score

$$f : \mathcal{X} \times \mathcal{L} \to \mathbb{R} \; tel \; que \; l_i \succ_x l_j \Rightarrow f(x, l_i) > f(x, l_j)$$

ou en utilisant l'approche qualitative [HFCB08], en apprenant pour chaque paire de labels (l_i, l_j) un prédicat $Q_{ij}(x)$ permettant de predire si $l_i \succ_x l_j$ ou $l_j \succ_x l_i$; le problème est résumé sur la figure 2.2.

Etant donnés :

– r une instance d'un schema relationnel (R)
– un ensemble de tuples $\{t_i | i = 1, 2, ..., n\} \subseteq r$
– un ensemble de labels $\mathcal{L} = \{l_i | i = 1, 2, ..., k\}$
– Pour chaque objet t : un ensemble de paires de preferences de la forme $l_i \succ_t l_j$

Trouver :

– une fonction de préférence qui associe à chaque $t \in r$ un classement $\succ_t$ dans $\mathcal{L}$

FIGURE 2.2 – classement de labels [FH10]

Pour illustrer le problème du classement de labels, nous considérons le scénario suivant :

Exemple 18. *Etant donné le scénario décrit dans l'exemple 17, nous considérons les utilisateurs comme étant des objets (ou tuples) et les films comme étant des labels (ou entités sans contenu).*

En considérant la table 2.1(b), on constate que les utilisateurs classent les films de la façon suivante :

$$u_1 : m_3 \succ m_2 \succ m_1$$

et

$$u_2 : m_1 \succ m_2 \succ m_3$$

Cet ordre signifie que l'utilisateur u_1 préfère d'abord le film identifié par m_3, puis celui identifié par m_2 et enfin celui identifié par m_1. On constate également que l'utilisateur u_2 préfère d'abord le film identifié par m_1 puis celui identifié par m_2 et enfin celui identifié par m_3. En considérant la figure 2.2, les données d'apprentissage sont alors les paires :

$$\{(m_3 \succ_{u_1} m_2), (m_3 \succ_{u_1} m_1), (m_2 \succ_{u_1} m_1), (m_1 \succ_{u_2} m_2), (m_1 \succ_{u_2} m_3), (m_2 \succ_{u_2} m_3)\}$$

Le problème du classement de labels consiste alors à apprendre un modèle de préférences permettant de prédire la façon dont un nouvel utilisateur (disons u_3) va classer les trois films $\{m_1, m_2, m_3\}$. Une étude complète sur le classement de labels est disponible sur [GV10].

Dans la section suivante, nous présentons un autre problème d'apprentissage pour le classement dont les données d'apprentissage sont constituées d'un ensemble d'objets et d'un ensemble de labels ; mais dans cette problématique, les labels associés aux objets sont ordonnés.

2.1.2 Classement d'instances

Le problème du classement d'instances (ou *Instance ranking* en anglais) est un sous problème d'apprentissage pour le classement dont les données d'apprentissage sont constitués par des objets et des labels ordonnées ; les labels ordonnés appartiennent à un ensemble fini de labels ordonnés.
Le problème de classement consiste à classer de nouveaux objets en utilisant le modèle appris sur les données d'apprentissages.
Pour résoudre le problème de classement d'instances, on utilise une approche quantitative. Dans le cas où le nombre de labels ordonnés $\mathcal{L}$ est égal à deux, ce problème est connu sous le nom de *bipartite ranking*. Dans le cas où le nombre de labels ordonnés $\mathcal{L}$ est supérieur à deux, le problème est connu sous le nom de $\mathcal{L}$*-partite ranking*[RA05] ou *multipartite-ranking*[Cin09]. Le problème de classement d'instance est résumé sur la figure 2.3.
Nous utilisons l'exemple suivant pour illustrer le problème.

Exemple 19. *Considérons les préférences de l'utilisateur u_1 sur les trois premiers films de la table 2.1(b). Nous considérons cette fois les films comme étant les objets sur lesquels l'apprentissage est effectué (cette fois-ci nous allons donc considérer les caractéristiques des films comme décrits dans le tableau 1.1). On constate qu'a chaque film est associé un label parmi les labels $\{Faible, Moyen, Fort\}$.*

Etant donnée cette considération, le problème consiste à apprendre une fonction de score

$$f : r \rightarrow \{Faible, Moyen, Fort\}.$$

Cette fonction de score doit permettre de classer de nouveaux films en leur assignant les trois scores proposés.

Exemple 20. *En considérant les données de la table 2.1(b), on constate qu' à un film peut être associé un label parmi les labels :$\{Faible, Moyen, Fort\}$. Les données d'apprentissage sont alors les paires (m_1,Faible),(m_2,Moyen) et (m_3, Fort).*
En considérant trois nouveaux films {m_3, m_4 et m_5} et en associant les scores Faible à m_3, Fort à m_4 et Moyen à m_5, la fonction de score apprise sur les trois premiers films doit assigner des scores aux films m_3, m_4 et m_5 de sorte que $f(m_4) = Fort, f(m_5) = Moyen, f(m_3) = Faible$.

Pour plus d'informations sur les travaux effectués sur le classement d'instances, le lecteur intéressé peut consulter les travaux de [WB11].

Dans les problèmes présentés dans les sous-sections 2.1.1 et 2.1.2, chaque objet est

Etant donnés :

- r une instance d'un schema relationnel (R)
- Un ensemble de tuples $\{t_i | i = 1, 2, ..., n\} \subseteq r$
- Un ensemble de Labels ordonnés $\mathcal{L} = \{l_1, l_2, ..., l_k\}$ tel que : $l_1 < l_2 < ... < l_k$
- Pour chaque instance d'apprentissage t_i un label associé l_i

Trouver :

- Une fonction de classement permettant d'ordonner un nouvel ensemble $\{t_j\}_{i=1}^{j}$ en se basant sur les labels ordonnés (éventuellement inconnu) associés aux instances

FIGURE 2.3 – Classement d'instance [FH10]

associé à un label (paires de labels) dans les données d'apprentissage. Dans la sous-section suivante, nous allons présenter une problématique d'apprentissage dont les objets constituants les données d'apprentissage ne sont pas associés à des labels.

2.1.3 Classement d'objets

Etant donnés :

- r une instance d'un schema relationnel (R)
- un ensemble fini de paires d'objets $t_i \succ t_j, (t_i, t_j) \in r \times r$

Trouver :

- une fonction de préférence permettant de prédire entre deux objets lequel est le préféré

FIGURE 2.4 – Classement d'objets [FH10]

Dans le problème de classement d'objets (ou *object ranking* en anglais), l'objectif est d'apprendre un modèle permettant de prédire entre deux objets donnés lequel est le préféré. Les données d'apprentissage sont constituées de paires d'objets ou d'un exemple de classements d'objets.
Le problème de classement d'objets peut être résolu en utilisant une approche quantitative [DKSS11], en apprenant une fonction de score :

$$f : r \to \mathbb{R} \; tel \; que \; t_i \succ t_j \Rightarrow f(t_i) > f(t_j).$$

ou en utilisant une approche qualitative [CSS99] : en apprenant un prédicat binaire $Q(t_i, t_j)$ permettant de prédire si $t_i \succ t_j$ ou $t_j \succ t_i$.

Exemple 21. *Pour illustrer le problème de classement d'objets, nous allons considérer les préférences de l'utilisateur u_1 sur les films comme décrit dans la table 2.1(c).*

En considérant les données de la table 2.1(c), les données d'apprentissage sont les paires $\langle m_3, m_2\rangle$ $\langle m_3, m_1\rangle$ $\langle m_2, m_1\rangle$. Le problème de classement d'objets consiste alors à apprendre un modèle permettant de prédire comment l'utilisateur u_1 va classer de nouveaux films en se basant sur les caractéristiques des films sur lesquels l'utilisateur a exprimé ses préférences.

2.1.4 Mesures d'évaluation pour le classement

Pour évaluer la qualité de prédiction des modèles de classements, plusieurs mesures de performances peuvent être utilisées. Les mesures de performances permettent d'évaluer la confrontation entre le classement effectué par le modèle et le classement cible. Les mesures de performances les plus utilisées sont :

- La Precision,
- Le rappel,
- La F-mesure
- Le Kindall
- La NDCG

Afin de pouvoir présenter la définition de ces mesures de performances, nous effectuons les considérations suivantes :

- Soit D un ensemble de paires d'éléments (objets ou labels) (x_1, x_2) tel que $x_1 \succ x_2$.
- Soit M le modèle appris sur les données d'apprentissage.
- Soit T, le nombre de paires d'éléments concordants : c'est à dire le nombre d'éléments classés de la même façon dans D et dans M.
- Soit F le nombre de paires d'éléments discordants : c'est à dire le nombre de paires d'éléments classés de façon différentes dans D et dans M.
- Soit I, le nombre de paires d'éléments que le modèle ne parvient pas à classer : le nombre de paires d'éléments pour lesquels M n'est pas capable de préciser la préférence de l'un sur l'autre.

En se basant sur ces notations, nous allons donner la définition de mesures :

$$Precision = \frac{T}{T + F}$$

$$Rappel = \frac{T}{T + F + I}$$

$$F - mesure = \frac{2 * (precision * rappel)}{precision + rappel}$$

$$Kindall = \frac{T - F}{T + F}$$

Une autre mesure qui est fréquemment utilisée est la Normalized Disconted cumulative Gain (NDCG). C'est une mesure qui évalue la qualité du classement effectué par le modèle. Elle est utilisée dans le domaine de la recherche d'information pour évaluer la qualité des algorithmes implémentés dans les moteurs de recherche [Wik14].

2.1.5 Synthèse

Dans cette section, nous avons présenté le problème de l'apprentissage pour le classement ainsi que certaines de ces applications. Nous avons présenté également les trois problèmes de classement qui existent dans le littérature : à savoir le problème du classement de labels (cf. sous-section 2.1.1), le problème de classement d'instances (cf. sous-section 2.1.2) et le problème de classement d'objets (cf. sous-section 2.1.3).

Nous avons vu que selon le type de problème de classement attaqué, la forme des données d'apprentissage est diffèrent. Pour le classement de labels, les données d'apprentissage sont constituées par des paires de labels pour chaque objet. Dans le problème de classement d'instances, les données d'apprentissage ne sont pas constituées par des paires, mais par des objets associés à des labels ordonnés. Dans le problème de classement d'objets, les objets ne sont pas associés à des labels, les données d'apprentissage sont constituées par des paires d'objets.

Nous avons vu que pour apprendre les modèles de classement sur les données d'apprentissage, pour le problème du classement de labels et de classement d'objets, il est possible d'utiliser l'approche qualitative [CSS99, HFCB08] ou l'approche quantitative [DMS04, HpRZ02, HpRZ03, XL07]. Tandis que pour le problème de classement d'instances, l'objectif est de pouvoir classer les instances grâce aux scores individuels associés à chaque instance. Par conséquent, il n'est pas utile d'apprendre une relation binaire sur les données d'apprentissage.

Pour évaluer la qualité de prédiction du modèle appris, plusieurs mesures de performances peuvent être utilisées. Les mesures les plus utilisées sont notamment celles issues de la recherche d'information : à savoir la *Précision*, le *Rappel*, la *NDCG* (cf. sous-section 2.1.4).

Dans ce qui suit (cf. sous-section 2.3), nous allons présenter quelques travaux sur l'apprentissage de préférences, plus précisément sur les méthodes qui font de l'apprentissage pour le classement. Pour bien étudier ces méthodes d'apprentissage de préférences, nous avons défini des critères d'études (cf. section 2.2).

2.2 Les critères d'études

Dans cette section, nous présentons les critères d'étude que nous avons définis. Ces critères sont bien en phase avec notre objectif : faire un état de l'art sur l'apprentissage de préférences pour le classement d'objets.
Il existe un certain nombre de travaux qui présentent un état de l'art sur l'apprentissage de préférences [KKA10, FH10, CP04]. Par contre, ces travaux ne présentent pas des critères de comparaison des méthodes proposées. Par exemple, dans les travaux de [FH10], les auteurs ne se limitent pas seulement sur à l'étude des travaux présentés dans le domaine de classement d'objets mais vont au delà en présentant les travaux existants dans le domaine du classement d'instances et du classement de labels. Ce travail n'est pas adapté à notre contexte car il est trop général.
Nous présentons ci-dessous les détails des critères que nous jugeons intéressants d'étudier pour la classement d'objets.

- **La forme des données d'apprentissage**
  Nous nous intéressons au format des données sur lesquelles travaillent les méthodes d'apprentissage de préférences que nous présentons. Sachant que les méthodes peuvent travailler sur des données ayant des formats différents : des paires d'objets avec l'information sur la préférence d'un objet sur un autre ou un exemple de classement d'objets.
  Dans le cas où les données d'apprentissage sont un exemple de classement d'objets, on à une relation d'ordre totale sur les objets constituants les données d'apprentissage. Dans le cas où les données d'apprentissage sont données sous forme de paires d'objets : il peut y avoir des cas d' inconsistance sur les préférences.

- **Le type d'approche de modélisation utilisé**
  Ce critère permet de distinguer l'approche utilisée pour modéliser les préférences :

est ce que l'approche est quantitative ou qualitative ? sachant que dans notre travail, nous avons utilisé l'approche qualitative pour modéliser les préférences.

- **Techniques utilisées pour apprendre les préférences**
  Les méthodes que nous présentons dans le section 2.3 utilisent différentes techniques de fouille pour apprendre les préférences. Nous précisons la technique de fouille utilisée par chaque méthode.

- **Intelligibilité du modèle**
  Il est important pour le décideur de savoir sur quoi il se base pour prendre une décision. Par conséquent, il est important de construire un modèle de préférence intelligible. Nous utilisons ce critère pour préciser si la technique utilisée permet de construire un profil intelligible.

- **Préférences contextuelles**
  Ce critère nous permet de préciser si la méthode étudiée prend en compte les préférences contextuelles : est ce que la préférence sur la valeur d'un attribut dépend de la valeur d'un autre attribut dans la base de données ?

- **Mesures de qualité utilisées**
  Nous examinons si les méthodes que nous présentons proposent une mesure de qualité permettant d'évaluer la qualité de prédiction du modèle construit.

- **Caractéristiques de la relation de préférence induite par le modèle**
  Nous montrons également les caractéristiques de la relation de préférences construite (ordre partiel strict, ordre total..).

2.3 Méthodes de classement d'objets

Dans cette section, nous présentons quelques méthodes de classements d'objets. Nous présentons dans la sous-section 2.3.1 les méthodes qui utilisent l'approche quantitative [Joa06, FISS03, XL07, BSR+05, DKSS11]. Dans la sous-section 2.3.2, nous présentons les méthodes qui utilisent l'approche qualitative [HEK03, Jea08, CKL+10, GLS14, DKSS11].

2.3.1 Approche quantitative

Dans cette section, nous présentons des méthodes de découverte de préférences utilisant l'approche quantitative. Nous classons les travaux que nous présentons en deux catégories : les méthodes qui apprennent des modèles de préférences non-intelligibles [Joa06, FISS03, XL07, BSR+05] et les méthodes qui apprennent des modèles de préférences intelligibles[DKSS11]. Dans les travaux de [DKSS11], les auteurs utilisent l'approche quantitative et l'approche qualitative pour apprendre des règles de préférences. Les règles de préférences apprises sont diffèrent selon qu'on utilise l'approche qualitative ou l'approche quantitative.
Dans le cas de l'approche quantitative, les règles de préférences apprises spécifient une condition sur l'évaluation d'un seul objet sur des attributs particuliers. Cette évaluation conduit à l'augmentation ou à la diminution du degré de préférence de l'objet (cf. paragraphe 2.3.1.2).
Dans le cas de l'approche qualitative, les règles de préférences apprises spécifient des conditions sur la différence de l'évaluation entre deux objets sur des attributs particuliers (cf. paragraphe 2.3.2.1.3).

2.3.1.1 Modèles de préférences non-intelligibles

Nous allons présenter dans cette partie quelques travaux qui apprennent les préférences quantitatives non-intelligibles. Les travaux que nous allons présenter sont les suivants :

2.3.1.1.1 Rankboost Dans les travaux de Freund et al.[FISS03], les auteurs utilisent la technique de Boosting pour apprendre les préférences des utilisateurs.

Dans ce travail, les données d'apprentissage sont sous la forme de paires d'objets $(x, x') \in \mathcal{X} \times \mathcal{X}$ avec une information indiquant que l'objet x est préféré à l'objet x' ou vice versa. Cette information est fournie par une fonction de retour (ou *feedback function* en anglais) :

$$\Phi : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$$

Dans ce travail, les auteurs apprennent sur ces données d'apprentissage, une fonction de score de la forme :

$$f(x) = \sum_{t=1}^{T} \alpha_t f_t(x)$$

qui est une combinaison linéaire de fonctions de scores appelées *Weak ranker*. Les *Weak ranker* induisent des relations d'ordre partielle sur les objets : ces fonctions n'assignent pas toujours un score à tout objet. Il est possible qu'elles s'abstiennent d'assigner un

score à un objet.
Pour représenter l'abstention d'un *Weak ranker* sur un objet, les auteurs utilisent le symbole $\perp$ qui est incomparable à tout nombre réel. Les *Weak ranker* sont sous la forme $f_t : \mathcal{X} \rightarrow \bar{\mathbb{R}}$ où $\bar{\mathbb{R}}$ représente l'ensemble des nombres réelles plus le symbole $\perp$ et $\alpha_t \in \mathbb{R}$.
Lors du classement final, les auteurs veulent avoir un ordre total sur les objets, par conséquent la fonction f ne doit pas s'abstenir d'assigner un score à un objet même si f_t le fait. Dans le cas où f_t s'abstient à assigner un score à un objet, un nombre réel est associé à l'objet en question. Le modèle final est défini donc de la façon suivante : $f : \mathcal{X} \rightarrow \mathbb{R}$.
Le modèle construit n'est pas facilement compréhensible par l'utilisateur : on ne peut pas expliquer pourquoi un objet x à un score plus élevé qu'un objet x'. La relation de préférence induite par le modèle de préférence est un ordre total puisque le modèle est capable d'assigner un score à chaque objet.

Pour évaluer la qualité de prédiction du modèle construit, les auteurs utilisent les mesures de performances comme la *Précision* et le *Rappel*.
Les expérimentations effectuées sur des données réelles montrent la robustesse du modèle.

Une autre technique utilisée pour apprendre les préférences quantitatives est l'utilisation des SVMs.

2.3.1.1.2 RankSVM Dans les travaux de [Joa06], les auteurs utilisent les SVMs pour apprendre une fonction de score. Dans ce travail, les données d'apprentissage sont construites à partir des clics des utilisateurs sur les documents représentant les résultats d'une requête q. Les *clicthrough data* sont sous la forme de triplets (q, r, c) où q représente une requête, r représente le classement des documents représentant les résultats de la requête et c représente l'ensemble des documents sur lesquels l'utilisateur a cliqué : ce sont les documents préférés par l'utilisateurs.
L'auteur utilise les informations contenues dans les clics des utilisateurs pour construire les paires de documents constituant les données d'apprentissage pour une requête q.
Un document d_i est préféré à un document d_j noté $d_i \succ_{r*} d_j$ pour tout paire (d_i, d_j) tel que $d_i \in C$ et $d_j \notin C$ avec $1 \leq j < i$. r* représente le classement des documents représentant les résultats d'une requête selon les préférences de l'utilisateur.
Les données d'apprentissage sont un ensemble $S = \{(q_i, r_i^*)\}_{i=1}^n$ *où* q_i est une requête et r_i^* est le classement cible des documents représentants les résultats de la requête q_i.

L'auteur utilise une approche SVM pour apprendre une fonction de classement f_q permettant de prédire pour une requête q, un document d_a est préféré à un document d_b vice versa. La fonction de score apprise produit un classement r_{f_q} des documents représentant les résultats d'une requête. L'objectif est que la fonction de classement

apprise sur les données doit maximiser le kindall sur les données d'apprentissage

$$\tau_S(f) = \frac{1}{n}\sum_{i=1}^{n}\tau(r_{f(q_i)}, r_i^*).$$

Les auteurs ont montré que maximiser le kindall sur les données d'apprentissage augmente la qualité de la recherche d'information.
La fonction de score apprise doit appartenir à la famille de fonctions scores

$$F = \{f_{\vec{w}}(q)\ tq\ (d_i, d_j) \in f_{\vec{w}}(q) \iff \vec{w}\Phi(q, d_i) > \vec{w}\Phi(q, d_j)\}$$

$\Phi(q, d)$ est un score qui représente le degré d'adéquation entre la requête q et le document d, $\vec{w}$ est un vecteur de poids.
L'auteur à montré que construire la famille de fonction de scores F revient à trouver le vecteur poids , $\vec{w}$, de sorte que le plus grand nombre des inégalités suivantes soit vérifié.

$$\forall (d_i, d_j) \in r_1^* : \vec{w}\Phi(q_1, d_i) > \vec{w}\Phi(q_1, d_j)$$

$$\dots$$

$$\forall (d_i, d_j) \in r_n^* : \vec{w}\Phi(q_n, d_i) > \vec{w}\Phi(q_n, d_j)$$

Il est montré dans [HSV95] que le problème est NP-difficile. Cependant les auteurs montrent qu'il est possible d'avoir une solution approchée en introduisant une variable ressort (ou slack variables en anglais) $\xi_{i,j,k}$ et en minimisant la marge supérieure $\sum \xi_{i,j,k}$. Cela revient à résoudre le problème d'optimisation suivant :

Problème 1.

$$minimiser : V(\vec{w}, \vec{\xi}) = \frac{1}{2}\vec{w}.\vec{w} + C\sum \xi_{i,j,k}$$

Ce problème conduit aux inégalités suivantes :

$$\forall (d_i, d_j) \in r_1^* : \vec{w}\Phi(q_1, d_i) > \vec{w}\Phi(q_1, d_j) + 1 - \xi_{i,j,k}$$

$$\dots$$

$$\forall (d_i, d_j) \in r_n^* : \vec{w}\Phi(q_n, d_i) > \vec{w}\Phi(q_n, d_j) + 1 - \xi_{i,j,k}$$

$$\forall i \forall j \forall k : \xi_{i,j,k} \geq 0$$

C est le paramètre qui permet le compromis entre la largeur de la marge et le nombre d'erreur de classement. Le problème d'optimisation est convexe et n'a pas d'optima local.

On constate que ce problème peut être transformé sous la forme :

$$\vec{w}(\Phi(q_k, d_i) - \Phi(q_k, d_j)) \geq +1 - \xi_{i,j,k}.$$

Il devient apparent que le problème d'optimisation est équivalent à celui de la classification SVM sur la différence de paires de vecteurs $\Phi(q_k, d_i) - \Phi(q_k, d_j)$. En raison de cette similarité, les auteurs utilisent un adaptation de l'algorithme SVM^{light} [Joa99] pour l'apprentissage.
Les scores assignés aux documents ne dépendent pas d'un contexte, comme dans les travaux précédemment présentés le modèle construit n'est pas intelligible.
Les données d'apprentissage venant des clics des utilisateurs, le modèle est donc capable de travailler sur des données bruitées.
Pour évaluer le modèle, Joachim a effectué une série d'expérimentations permettant de montrer (1) que le modèle construit permet de maximiser le taux de kindall et (2) que le modèle construit augmente la qualité de la recherche d'information.

Dans le paragraphe 2.3.1.1.3, nous allons présenter un méthode qui utilise les réseaux de neurones pour apprendre les préférences [BSR+05].

2.3.1.1.3 RankNet Dans les travaux de Burges et al.[BSR+05], les auteurs utilisent les méthodes de descente de gradient pour apprendre une fonction de classement.
Les données d'apprentissage sont sous la forme de paires $(x, x') \in \mathcal{X} \times \mathcal{X}$ et $P_{xx'}$ la probabilité indiquant que x doit être préféré à x'.

Les auteurs utilisent un algorithme de descente de gradient pour apprendre un modèle probabiliste permettant de prédire la probabilité qu'un objet x soit préféré à un objet x'. Ce modèle est sous la forme :

$$P_{xx'} = \frac{e^{(f(x)-f(x'))}}{1 + e^{(f(x)-f(x'))}}$$

avec $f : \mathcal{X} \to \mathbb{R}$ une fonction de score tel que $x \succ x'$ ssi $f(x) > f(x')$. Pour implémenter cette méthode l'auteur utilise les réseaux de neurones.
Dans ce travail, les préférences ne sont pas contextuelles ; les scores affectés aux objets ne dépendent pas d'un contexte. Le modèle construit n'est pas facilement compréhensible par l'utilisateur : il n'est pas facile d'expliquer à un utilisateur pourquoi la probabilité qu'un objet soit préféré a un autre est égale à telle valeur.

La qualité de prédiction du modèle construit est évalué en terme de *NDCG* (normalized Discunted Cumulative Gain).

Les modèles de préférences présentées dans cette sous-sous-section 2.3.1.1 ne sont pas intelligibles : il n'est pas simple d'expliquer à un utilisateur (ou décideur) pourquoi tel objet à tel score. Dans la sous-sous-section suivante, nous présentons un travail qui apprend un modèle de préférence intelligible.

2.3.1.2 Modèles de préférences intelligibles

Nous allons présenter dans cette sous-sous-section, l'approche quantitative utilisée dans [DKSS11]pour apprendre les règles de décisions. Les auteurs travaillent sur des paires d'objets $(x, x') \in \mathcal{X} \times \mathcal{X}$, chaque objet étant décrit par m attributs $x = (x_1, x_2, ..., x_m)$. Les données d'apprentissage contiennent l'information sur la préférence d'un objet sur un autre. Cette information est fournie par la fonction de score

$$s : \mathcal{X} \times \mathcal{X} \to \{-1, 1\}$$

de sorte que :

- $s(x, x') = 1 \Leftrightarrow x \succ x'$
- $s(x, x') = -1 \Leftrightarrow x' \succ x$

L'objectif de cette approche est d'apprendre une fonction de score

$$f : \mathcal{X} \to \mathbb{R}$$

pour prédire la valeur de $s(x, x')$. La fonction de score apprise est représentée par un ensemble de règles de décisions de la forme :

$$if(x_{i_1} > \delta_{i_1}) \wedge ...(x_{i'_l} > \delta_{i'_l}) \wedge (x_{j_1} < \delta_{j_1}) \wedge ...(x_{j''_l} < \delta_{j''_l})$$

alors le score de x augmente de α.
$\delta(...) \in \mathbb{R}$, et l'ensemble des indices d'attributs $\{i_1, i_2, ..., i'_l\} \subseteq \{1, ..., m\}$ et $\{j_1, j_2, ..., j'_l\} \subseteq \{1, ..., m\}$ ne sont pas nécessairement disjoints.
Pour apprendre ces règles de décisions, les auteurs utilisent la technique de Boosting. L'avantage d'utiliser des règles de décisions dans un modèle de préférences réside dans la simplicité et la facilité de compréhension par les humains.
Pour pouvoir prédire la valeur de $s(x, x')$, les auteurs utilisent la différence des scores fournis par f(.) sur les objets x et x' i.e $f(x) - f(x')$ de sorte que :

- $f(x) - f(x') > 0 \Leftrightarrow x \succ x'$,
- $f(x) - f(x') < 0 \Leftrightarrow x' \succ x$
- $f(x) - f(x') = 0 \Leftrightarrow$ qu'il n'y a pas de préférence entre x et x'

Pour évaluer la qualité du modèle appris,les auteurs utilisent la fonction de perte L définie de la façon suivante :

$$L(s(x,x'), f(x) - f(x')) = \begin{cases} 0 & \text{si } s(x,x') \times (f(x) - f(x')) > 0 \\ 1/2 & \text{si } s(x,x') \times (f(x) - f(x')) = 0 \\ 1 & \text{si } s(x,x') \times (f(x) - f(x')) < 0 \end{cases}$$

La fonction de perte est utilisée lors de la phase de construction du modèle qui est composé d'un ensemble de règles de préférences contextuelles comme cité ci haut.
Lors de la phase de construction du modèle, les règles de préférences contextuelles sont rajoutées dans le modèle jusqu'à ce que la fonction de perte soit égal à 0 sur toutes les paires d'objets contenues dans les données d'apprentissage.

Pour évaluer la prédiction du modèle, les auteurs utilisent le *ranking error*. Cette mesure permet mesurer l'erreur existant entre la prédiction effectuée par le modèle f et l'information fournie par la fonction s. En effet une erreur est survenue lorsque $s(x,x') = -1$ et que le modèle classe x au dessus de x' ou lorsque $s(x,x') = 1$ et que le modèle classe x en dessous de x'.

2.3.1.3 Conclusion

Nous avons présenté dans cette section quelques travaux sur l'apprentissage de préférences quantitative [Joa06, FISS03, XL07, BSR+05, DKSS11]. L'avantage des méthodes d'apprentissage de préférences quantitatives est la capacité d'apprendre un modèle de préférence qui induit un ordre total sur les objets.
Par contre, les modèles de préférences apprises ne sont pas toujours facilement compréhensible : il n'est pas facile d'expliquer simplement à un utilisateur (ou décideur) pourquoi tel objet à tel score. À notre connaissance, la seule méthode quantitative qui apprend un modèle facilement interprétable est [DKSS11].

2.3.2 Approche qualitative

Dans cette sous-section, nous présentons des travaux qui utilisent l'approche qualitative pour apprendre les préférences des utilisateurs. Dans la sous-sous-section 2.3.2.1, nous présentons les travaux qui apprennent des préférences non-contextuelles [HEK03, Jea08, YWLd08, SM06, DIV07, DKSS11], et dans la sous-sous-section 2.3.2.2, nous présentons les travaux qui apprennent des préférences contextuelles [CKL+10, dABAdS13, GLS14, ZCY+12]. Pour rappel, nous avons défini la notion de préférence contextuelle dans la sous-sous-section 1.1.2.1 (cf. Définition 10).

2.3.2.1 Modèles de préférences non-contextuelles

2.3.2.1.1 Apprentissage de modèle pareto Dans les travaux de [Jea08], les auteurs apprennent un modèle de préférence pareto. Dans ce travail, les données d'apprentissage sont constituées de paires d'objets $(x, x') \in \mathcal{X} \times \mathcal{X}'$ tel que $\mathcal{X}$ contient l'ensemble des données préférées par l'utilisateur et $\mathcal{X}'$ contient l'ensemble des données non-préférées par l'utilisateur, $\mathcal{X} \cap \mathcal{X}' = \varnothing$.
Un objet x est représenté par un ensemble d'attributs sur lesquels les préférences des utilisateurs ne sont pas connues et un ensemble d'attributs sur lesquels les préférences des utilisateurs sont connus. Dans ce travail, l'objectif est d'apprendre les préférences sur les attributs indéterminés.

Les auteurs considèrent que la préférence $\succ$ d'un objet sur un autre est une composition des préférences sur les attributs des deux objets. Ainsi, un objet x est préféré à un objet x' noté $x \succ x'$ ssi $x.d_i \succ_i x'.d_i$ pour chaque attribut d_i appartenant à l'ensemble des attributs D $(1 \leq i \leq m)$. On a ainsi $\succ = (\succ_1 \otimes \succ_2 \ldots \otimes \succ_m)$, m représente le nombre d'attributs des objets.

Nous constatons que les préférences sur les attributs sont d'importances égales, la préférence sur la valeur d'un attribut ne dépend pas de la valeur d'un autre attribut.
Pour apprendre le modèle de préférence, les auteurs utilisent un algorithme glouton. Le modèle construit induit un ordre partiel strict sur les objets.

Dans ce travail, la qualité de prédiction du modèle n'est pas évaluée.
Dans les travaux de [HEK03], les auteurs apprennent aussi un modèle de préférence Pareto, en utilisant les techniques du clustring. Comme dans les travaux de [Jea08], le modèle construit induit un ordre partiel strict sur les objets, les préférences apprises ne sont pas contextuelles. Pour évaluer la qualité de prédiction du modèle, les auteurs utilisent la *Précision* et le *Rappel*.

2.3.2.1.2 Apprentissage de modèle lexicographique Un modèle de préférence lexicographique définit un ordre d'importance sur les attributs représentant un objet et ensuite utilise l'ordre défini sur ces attributs pour prédire que tel objet est préféré à tel autre [YWLd08].

Exemple 22. *Considérons le schéma film de l'exemple 12 et le modèle lexicographique* $\mathcal{L}$*={genre* $\succ$ *directeur }. Considérons également les préférences suivantes sur les valeurs des attributs genre et directeur :* p_1*= comedy* $\succ$ *drame et* p_2 *=W .Allen* $\succ$ *M.Curtis. En considérant les données de la table 2.2, la relation de préférence lexicographique* $\succ_{p_1}$ & $\succ_{p_2}$ *induit l'ordre suivant :* $f_1 \succ_{p_1}$ & $\succ_{p_2}$ $f_2 \succ_{p_1}$ & $\succ_{p_2}$ $f_3 \succ_{p_1}$ & $\succ_{p_2}$ f_4.

	genre	directeur
f_1	comedy	W. Allen
f_2	comedy	M. Curtiz
f_3	drame	W. Allen
f_4	drame	M. Curtiz

TABLE 2.2 – Instance d'une relation films

Un modèle de préférence lexicographique est un modèle de préférence intelligible, on peut facilement expliquer à quelqu'un pourquoi un objet est préféré à un autre en se basant sur le modèle. Dans le modèle de préférence lexicographique, les préférences ne sont pas contextuelles : la préférence sur la valeur d'un attribut ne dépend pas de la valeur d'un autre attribut. Apprendre automatiquement un modèle de préférence lexicographique consiste à identifier l'ordre d'importances sur les attributs représentant les objets. Un modèle de préférence lexicographique $\mathcal{L}$ est consistant avec une observation $O = (x, x')$, indiquant que l'objet x est préféré à l'objet x', ssi $\mathcal{L}$ induit que $x \succ_{\mathcal{L}} x'$.

Il existe un certain nombre de travaux sur l'apprentissage des modèles Lexicographiques [YWLd08, SM06, DIV07]. Dans les travaux de Yaman et al.[YWLd08], les observations sont constituées par un ensemble de variables et un ensemble de paires d'objets. L'objectif est de trouver un modèle de préférence lexicographique consistant avec les observations.
Contrairement aux travaux de [SM06, DIV07], Yaman et al. ne trouvent pas un seul modèle de préférence lexicographique consistant avec les observations, mais gardent une collection de modèles de préférences lexicographiques consistants avec les observations. Ces modèles de préférences seront ensuite combinés avec un système de vote pour indiquer que tel objet est préféré à tel autre.

Deux méthodes de vote ont été proposées par [YWLd08] : Le *variable voting* et le *Model voting*. Avec le variable voting, le vote est effectué par les variables ayant des valeurs différentes sur les deux objets. Avec le *Model voting*, les auteurs utilisent une approche bayésienne pour effectuer le vote. Ainsi, un objet x est préféré au un autre x' si et seulement si la majorité des modèles lexicographiques préfèrent x à x'. Pour apprendre les modèles lexicographiques, [YWLd08] utilisent un algorithme de vote. Dans ce travail, le modèle de préférence lexicographique n'est définie que sur un sous-ensemble des attributs représentant les objets, ce qui conduit à un modèle qui induit une relation d'ordre partiel strict sur les objets. Dans les travaux de [SM06, DIV07], les modèles de préférences

lexicographiques induisent un ordre total sur les observations : le modèle lexicographique est défini sur tous les attributs représentant l'objet.
Dans les travaux de [YWLd08, DIV07], les modèles de préférences lexicographiques apprises doivent être parfaitement consistant avec les observations : le modèle construit n'est pas donc robuste, ce qui n'est pas le cas dans [SM06].

2.3.2.1.3 Modèle à base de règles Dans ce paragraphe, nous présentons l'approche qualitative proposée dans [DKSS11] pour faire du classement d'objets. Comme dans la version quantitative présentée dans le paragraphe 2.3.1.2, les données d'apprentissage sont constituées de paires d'objets $(x,x') \in \mathcal{X} \times \mathcal{X}$ et une information précisant si x est préféré à x' ou vice versa. Cette information est fournie par la fonction :

$$s : \mathcal{X} \times \mathcal{X} \to \{-1, 1\}$$

Comme dans l'approche présentée dans le paragraphe 2.3.1.2, l'objectif est d'apprendre une fonction de préférence permettant de prédire la valeur de $s(x, x')$. Dans cette approche, le modèle appris n'évalue pas individuellement un objet, mais les paires d'objets à comparer. Ainsi, à chaque paire d'objet, le modèle associe un score qui est le résultat de l'évaluation de la paire. Le modèle de préférence appris est sous la forme :

$$f : \mathcal{X} \times \mathcal{X} \to \{-1, 0, 1\}$$

de sorte que :

- $f(x, x') = 1 \Leftrightarrow x \succ x'$,
- $f(x, x') = -1 \Leftrightarrow x' \succ x$
- $f(x, x') = 0 \Leftrightarrow$ qu'il n'y a pas de préférence entre x et x'

Le modèle construit peut être représenté par un ensemble de règles de préférences contextuelles de la forme :

$$if(x_{i_1} - x'_{i_1} > \delta_{i_1}) \wedge ...(x_{i'_l} - x'_{i'_l} > \delta_{i'_l}) \wedge (x_{j_1} - x'_{j1} < \delta_{j_1}) \wedge ...(x_{j''_l} - x'_{j''_l} < \delta_{j''_l})$$

alors $x \succ x'$. où $\delta(...) \in \mathbb{R}$ et l'ensemble des indices d'attributs $\{i_1, i_2, ..., i'_l\} \subseteq \{1, ..., m\}$ et $\{j_1, j_2, ..., j'_l\} \subseteq \{1, ..., m\}$ ne sont pas nécessairement disjoints.
L'approche proposée peut travailler à la fois sur des attributs catégoriels que sur des attributs nominaux. Dans le cas des attributs catégoriels : attributs ayant des valeurs ordinales finis codées dans $\mathbb{R}$, les auteurs considèrent que $\delta() = 0$. Cette condition est surtout intéressante pour les règles de préférences contextuelles considérées dans l'approche qualitative. Si les attributs sont nominaux, les auteurs effectuent une binairisation de chacun de ces attributs. Les attributs nominaux seront ensuite remplacés par les nou-

veaux attributs et en traitant ces derniers comme des attributs ordinaux.
Pour évaluer le modèle appris, les auteurs utilisent la fonction L de la forme :

$$L(s(x,x'),f(x,x')) = \begin{cases} 0 & \text{si } s(x,x') \times f(x,x') > 0 \\ 1/2 & \text{si } s(x,x') \times f(x,x') = 0 \\ 1 & \text{si } s(x,x') \times f(x,x') < 0 \end{cases}$$

Lors de la phase de construction du profil, les règles de décision sont rajoutées dans le profil jusqu'à ce que $L = 0$.
La fonction de préférence apprise sur les données d'apprentissage, induit une relation de préférence réflexive mais pas nécessairement transitive, l'objectif étant d'avoir une relation d'ordre total, les auteurs utilisent une procédure de classement nommée NetFlowScore [Bou92, BP92, Vin92]. Cette procédure utilise les résultats de la fonction de préférence appliquée aux données pour assigner un score à chaque objet. La procédure de classement pour un objet x est défini de la façon suivante :

$$NFS(x) = \sum_{x' \in X \quad x' \neq x} \left[sgn(f(x,x')) - sgn(f(x',x)) \right]$$

Cette procédure permet d'avoir un ordre total sur les données test. Pour évaluer la qualité de prédiction du modèle $f(.,.)$, les auteurs utilisent le *ranking error* comme indiqué dans le paragraphe 2.3.1.2.

2.3.2.2 Modèles de préférences contextuelles

2.3.2.2.1 Apprentissage de CP-nets Dans les travaux de [CKL+10], les auteurs examinent le problème d'apprentissage de CP-nets selon plusieurs dimensions. Dans ce travail, les données d'apprentissage sont constituées par des paires d'objets qui représentent une relation de préférence cible que l'utilisateur veut atteindre.
Etant donnée ces considérations, les auteurs identifient deux scénarios :

(1) La relation de préférence que le l'utilisateur veut avoir est représentable par un CP-net.

Exemple 23. *La relation de préférence suivante : {comedyW.Allen ≻ comedyM.curtiz, comedyW.Allen ≻ dramaW.Allen, dramaM.curtiz ≻ comedyM.curtiz, dramaM.curtiz ≻ dramaW.Allen }*
peut être représenté par le cp-net suivant :
{ comedy : W.allen ≻ M.curtiz, drama : M.curtiz ≻ W.Allen, W.Allen :comedy ≻ drama, M.curtiz :drama ≻ comedy}

Dans ce cas, les auteurs utilisent la technique de l'apprentissage passif pour iden-

tifier le CP-net correspondant. Dans la technique de l'apprentissage passif, l'apprenant observe le comportement de l'utilisateur sur une situation donnée et puis extrait la structure de préférence correspondante à ce comportement.

(2) La relation de préférence cible n'est pas représentable par un CP-net.

Exemple 24. *L'ordre suivant : { comedyW.allen ≻ dramaM.curtiz ≻ comedyM.curtiz ≻ dramaW.Allen } ne peut pas être représenté par un cp-net. mais on peut constater que le CP-net du cas (1) induit le même ordre que l'exemple.*

Dans ce cas les auteurs utilisent la technique de l'apprentissage actif, dans cette technique l'apprenant teste le comportement de l'utilisateur dans certaines situations bien choisies puis apprend les préférences utilisateurs correspondantes. Avec la technique de l'apprentissage actif, il y a interaction entre l'apprenant et l'utilisateur.

Dans le cas du scénario (1), le modèle de préférence construit induit un ordre partiel stricte sur les objets. Dans le cas du scénario (2), les auteurs utilisent l'apprentissage active : qui consiste à interagir avec l'utilisateur a travers des requêtes de la forme est-ce que « x est préféré a y ? » et ensuite utiliser ces informations pour trouver un CP-net qui induit une relation proche de la relation cible. L'objectif est de construire un CP-net en effectuant le moins d'interaction possible avec l'utilisateur. Le modèle de préférence construit induit un ordre total sur les objets.

Comme nous l'avons vu dans le paragraphe 1.1.3.3, les CP-nets permettent d'exprimer des préférences contextuelles et sont utilisés dans les conditions (*ceteris paribus*). Les préférences sur la valeur d'une variable dépend de la valeur sur les autres variables. Les CP-nets sont un modèle de préférences intelligible, on peut expliquer facilement à un utilisateur pourquoi un objet est préféré à un autre en utilisant un CP-net.

2.3.2.2.2 CPrefMiner Dans les travaux de [dABAdS13], les auteurs apprennent un modèle bayésien de préférences composé d'un ensemble de règles de préférences contextuelles probabilistes. Les données d'apprentissage sont des bituples (x_1, x_2) tel que $x_1 \succ x_2$. Dans ce travail, les auteurs proposent un algorithme nommé $CPrefMiner$ qui est une approche qualitative pour apprendre un modèle bayésien de préférences.
Un réseaux bayésien de préférences est défini de la façon suivante :

Définition 20. *Un réseau bayésien de préférences (ou Bayesian Préférence Network (BNP) en anglais) sur un schéma relationnel $R(A_1, \ldots, A_n)$ est une paire (G, θ) où :*

1. *G est un graphe directe acyclique dont les neouds sont des attributs dans $\{A_1, \ldots, A_n\}$ et les arcs montrent les dépendances entre les attributs.*

2. *θ est un mapping qui associe à chaque noeud du graphe G, une table de préférences conditionnelles, qui est un ensemble fini de probabilités conditionnelles de la forme $P[E_2|E_1]$ où (i) E_1 est un évènement de la forme $(A_{i_1} = a_{i_1})\wedge, \ldots, \wedge(A_{i_k} = a_{i_k})$ tel que quelque soit $j \in \{1, \ldots, K\}$ $a_{i_j} \in dom(A_{i_j})$, et (ii) E_2 est un évènement de la forme $(B = b_1)$ est préféré à $(B = b_2)$, où B est un attribut de R, $B \neq A_{i_j}$ quelque soit $j \in \{1, \ldots, k\}$ et b_1, $b_2 \in dom(B)$, $b_1 \neq b_2$.*

Exemple 25. *Considérons un schéma relationnel $R(A, B, C, D)$ avec les domaines d'attributs $dom(A) = \{a_1, a_2, a_3\}$, $dom(B) = \{b_1, b_2\}$, $dom(C) = \{c_1, c_2\}$ et $dom(D) = \{d_1, d_2\}$. La figure 2.5 suivante représente un réseau bayésien de préférences.*

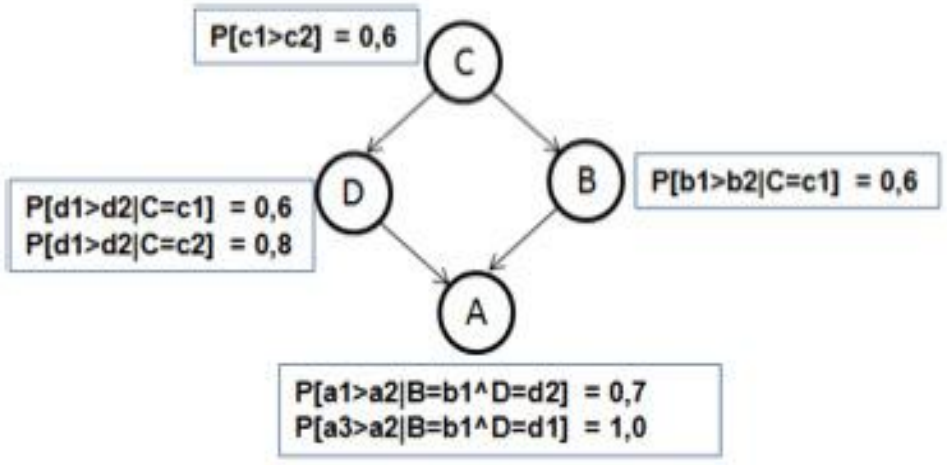


FIGURE 2.5 – Réseau Bayesien de Préférences [dABAdS13]

Une règle de préférence contextuelle probabiliste est sous la forme $P[E_1|E_2]$ où E_2 est le contexte de la règle et E_1 est la préférence. Une règle de préférence contextuelle probabiliste associé à un noeud x sur un graphe G représente le degré de confiance de préférer une valeur de X sur une autre en considérants les valeurs supposés des parents de X.

Exemple 26. *La règle de préférence contextuelle probabiliste $P[D = d_1 > D = d_2|C = c1] = 0,6$ signifie que la probabilité de $D = d_1$ soit préféré à $D = d_2$ sachant que $C = c_1$ est de 60%*

Définition 21. *Dans ce travail, un réseau bayésien de préférence induit qu'un tuple t_1 est préféré à un tuple t_2 si et seulement si la probabilité que t_1 soit préférée à t_2 est supérieure à la probabilité que t_2 soit supérieur à t_1.*

Exemple 27. *En considérant le schéma relationnel de l'exemple 25, le réseau bayésien de préférences présenté par la figure 2.5 induit que* $t_1 = (a_1, b_1, c_1, d_1) \succ_{BPN} t_2 = (a_2, b_2, c_1, d_2)$ *pour les raisons suivantes :*

1. *Les tuples* t_1 *et* t_2 *ont la même valeur sur les attributs C et des valeurs différentes sur les attributs A,B,D. On peut donc utiliser une règle ayant comme contexte* $C = c_1$
2. *Les deux règles ayant comme contexte* $C = c_1$ *sont associées aux attributs B et D ; par conséquent, pour avoir* $P(t_1 \succ_{BNP} t_2) > P(t_2 \succ_{BNP} t_1)$ *il suffit d'avoir :* $P(b_1 > b_2|c_1) \times P(d_1 > d_2|c_1) > P(b_2 > b_1|c_1) \times P(d_2 > d_1|c_1)$.
3. *On a :* $P(b_1 > b_2|c_1) \times P(d_1 > d_2|c_1) = 0,6 * 0,6$ *et* $P(b_2 > b_1|c_1) \times P(d_2 > d_1|c_1) = 0,4 * 0,4$
4. *On a :* $P(b_1 > b_2|c_1) \times P(d_1 > d_2|c_1) = 0,36 > P(b_2 > b_1|c_1) \times P(d_2 > d_1|c_1) = 0,16$, *par conséquent,* $t_1 \succ_{BPN} t_2$

Le modèle de préférence construit est compacte et produit une relation d'ordre strict sur les bi-tuples. Pour évaluer la qualité de prédiction du modèle, les auteurs utilisent la p*récision* et le *rappel*.

2.3.2.3 Conclusion

Dans cette section, nous avons présenté quelques travaux sur l'apprentissage de préférences qualitatives [HEK03, Jea08, YWLd08, SM06, DIV07, CKL+10, DKSS11]. Nous avons présenté ces travaux en les classant en deux catégories selon qu'ils apprennent les préférences contextuelles ou non.
Dans les travaux sur les modèles de préférences Pareto [HEK03, Jea08] et lexicographique [YWLd08, SM06, DIV07], les préférences contextuelles ne sont pas considérées, ce qui limite leurs expressivité. En plus, les modèles construits dans [YWLd08, DIV07] ne sont pas robustes.
Dans [DKSS11], des règles de préférences assez similaires aux nôtres ont été proposées, mais leurs extractions n'est pas facile. Dans [CKL+10, dABAdS13], les préférences contextuelles sont prise en compte. Dans [CKL+10], il n'est pas possible de comparer des transactions de taille différentes du fait de l'utilisation du principe du *ceteris-paribus* ; il n'est pas possible non plus de travailler sur des données bruitées.
Dans [dABAdS13], le modèle de préférence contextuelle construit fournit une qualité de prédiction inférieure à la nôtre.

2.4 Synthèse sur les méthodes de découverte de préférences

Dans cette section, nous présentons une synthèse sur les méthodes d'apprentissage de préférences que nous avons présentées précédemment. Nous effectuons cette synthèse en nous basant sur les critères d'étude que nous avons défini dans la section 2.2. Le tableau 2.4 présente un récapitulatif de ces méthodes.

- **La forme des données :**
  Dans ce travail, nous nous sommes consacré au classement d'objets : les techniques présentées travaillent sur des paires d'objets (x_1, x_2) signifiant que l'objet x_1 doit être préféré à l'objet x_2 [Joa06, YWLd08, SM06, DIV07, CKL+10, FISS03, DKSS11]. D'autres méthodes travaillent également sur des ensembles d'objets représentant ceux préférés par l'utilisateur et ceux non préférés par l'utilisateur [HEK03, Jea08]. Avec les ensembles, il n'y a pas d'inconsistance sur les paires d'objets.

- **Type d'approche d'apprentissage utilisé :**
  Pour apprendre les préférences des utilisateurs, deux approches existent : l'approche quantitative et l'approche qualitative. Les travaux utilisant l'approche quantitative apprennent une fonction de score sur les données d'apprentissage. Cette fonction de score permet d'évaluer individuellement chaque objet [BSR+05, FISS03, Joa06, DKSS11]. Les travaux qui utilisent l'approche qualitative apprennent une relation binaire sur les objets. Cette relation binaire apprise sur les objets permet d'évaluer directement une paire d'objets [YWLd08, SM06, DIV07, CKL+10, DKSS11].
  L'approche qualitative est plus générale que l'approche quantitative (cf. section 1.1.2.3), l'avantage qui existe en utilisant l'approche quantitative est d'avoir un ordre total sur les objets par contre, le modèle construit n'est pas toujours facilement compréhensible par l'utilisateur.
  Plusieurs techniques de fouille peuvent être utilisées pour apprendre une fonction de score ou une relation binaire.

- **Techniques utilisées pour apprendre les préférences :**
  Pour pouvoir prédire les préférences des utilisateurs sur de nouvelles données, il est nécessaire de construire un modèle sur les observations. Plusieurs techniques sont utilisées pour construire un modèle de préférences tel que le Boosting [FISS03, DKSS11], les SVMs [Joa06], les réseaux de neurones [BSR+05],

les réseaux bayesiens [dABAdS13] etc.

– **Intelligibilité du modèle :**
Il est important pour le décideur de savoir sur quoi il se base pour prendre une décision. Par conséquent, il est important d'exhiber les préférences des utilisateurs à travers un profil. La plupart des approches quantitatives que nous avons étudiées dans ce chapitre ne présentent pas les préférences des utilisateurs à travers un profil [BSR+05, FISS03, Joa06]. Les modèles de préférences qualitatives présentés produisent un modèle de préférence intelligible mais présentent quelques limites : certaines méthodes ne modélisent pas les préférences contextuelles [HEK03, Jea08, YWLd08, SM06, DIV07, DKSS11] d'autres ne peuvent pas travailler sur des données bruitées [CKL+10].

– **Préférences contextuelles :**
Les préférences d'un utilisateur peuvent dépendre d'un contexte [CKL+10, dABAdS13] : le choix du film à regarder peut dépendre du temps qu'il fait, de la personne qui nous accompagne .... Cependant toutes les méthodes de découverte de préférences ne modélisent pas les préférences contextuelles, ce qui limite leurs expressivité [BSR+05, FISS03, Joa06, HEK03, Jea08].

– **Mesures de qualité utilisées**
Pour évaluer la qualité de prédiction du modèle construit, les auteurs utilisent assez souvent des mesures issuent de la recherche d'information. Les mesures de qualité utilisées sont souvent réadaptées selon les problèmes. Les mesures les plus utilisées sont : *la NDCG*[BSR+05], *la Précision* [HEK03, FISS03, dABAdS13] et le *Rappel*[HEK03, dABAdS13, FISS03].

– **Caractéristiques de la relation de préférence induite par le modèle**
L'avantage de construire un modèle de préférences est de pouvoir prédire de nouvelles préférences en se basant sur le modèle construit. Le modèle construit doit permettre classer des objets en se basant sur les préférences utilisateur. L'avantage d'utiliser une approche quantitative est d'avoir un ordre total sur les objets lors de la prédiction. Par contre, les modèles construits avec l'approche quantitative ne sont pas toujours intelligibles.

	Modèle	Réf	FD	TU	Profil	Cont	IM	RO
Quantitatif	RankNet	[BSR+05]	paires	Descente de gradient	non	non	non	OT
	Ranboost	[FISS03]	paires	Boosting	non	non	non	OT
	Modèle à Base de Règles	[DKSS11]	paires	Boosting	oui	non	oui	OT
	RankSVM	[Joa06]	paires	SVM	non	non	non	OT
Qualitatif	Pareto	[HEK03]	ensemble	clustering	oui	non	oui	OPS
		[Jea08]	ensemble	méthode glouton	oui	non	oui	OPS
	Lexicographique	[YWLd08]	paires	algorithme de vote	non	non	oui	OPS
		[SM06]	paires	méthode glouton	non	non	non	OT
		[DIV07]	paires	algorithme à base d'oracle	non	non	non	OT
	CP-Net	[CKL+10]	paires	apprentissage passive- apprentissage active	oui	oui	non	OT
	Modèle à Base de Règles	[DKSS11]	paires	Boosting	oui	non	oui	OT
		[dABAdS13]	paires	réseaux bayesien	non	oui	oui	OPS

TABLE 2.3 – récapitulatif sur les méthodes d'apprentissage de préférences

- **OT** : Ordre total
- **OPS** : Ordre partiel Strict
- **FD** : représente la forme des données
- **TU** : Technique de fouille utilisée
- **Profil** : vérifie si un profil est exhibé
- **IM** : vérifie si le modèle construit est facilement compréhensible par l'utilisateur
- **RO** : represente la relation d'ordre induite par le modèle

Conclusion

Nous avons présenté dans cette partie, un état de l'art sur l'apprentissage de préférences. Nous avons présenté dans le chapitre 1, les notions essentielles pour comprendre les préférences utilisateurs. Nous avons présenté également les deux approches qui existent pour modéliser les préférences utilisateurs à savoir : l'approche qualitative et l'approche quantitative. Nous avons vu que dans l'approche qualitative, les préférences sont modélisées par une relation de préférence binaire, tandis que dans l'approche quantitative, les préférences sont modélisées par une fonction de score.

Dans le chapitre 2, nous avons présenté les approches qui existent dans le domaine de l'apprentissage de préférences. Plus précisément, nous avons présenté les méthodes d'apprentissage pour le classement. Pour cela, nous avons commencé par présenter les trois problèmes de classement qui existent dans la littérature à savoir le classement de labels (cf. sous-section 2.1.1), le classement d'instances (cf. sous-section 2.1.2) et le classement d' objets (cf. sous-section 2.1.3). Après la présentation des trois problèmes de classement, nous nous sommes consacrés au problème de classement d'objets en présentant en détail quelques travaux qui abordent ce problème.

Nous avons présenté ces travaux en distinguant ceux qui utilisent l'approche qualitative et ceux qui utilisent l'approche qualitative. Nous avons vu que l'approche quantitative assigne un score à chaque objet et ensuite utilise les scores assignés pour classer les objets. Pour assigner ces scores aux objets on utilise une fonction de score qui peut être apprise en utilisant les techniques comme le Boosting [FISS03], les réseaux de neurones [BSR+05], les SVMs [Joa06]. Malheureusement, les modèles appris ne sont pas toujours facilement compréhensibles par l'utilisateur pour que ce dernier puisse les compléter ou bien les corriger.

Dans l'approche qualitative, on utilise les relations de préférences binaires pour modéliser les préférences sur les objets. Nous avons divisé les approches quantitatives en

deux catégories, celles qui ne considèrent pas les contextes pour modéliser les préférences [HEK03, Jea08, YWLd08, SM06, DIV07, DKSS11] et celles qui considèrent les contextes pour modéliser les préférences. Les méthodes qui ne considèrent pas les contextes pour modéliser les préférences ne sont pas assez expressives car les préférences d'un utilisateur peuvent varier selon les contextes. Des méthodes qui considèrent les contextes pour modéliser les préférences sont présentées dans [CKL+10, dABAdS13].

Dans les travaux de [dABAdS13], les auteurs utilisent les réseaux bayésien pour extraire les règles de préférences contextuelles par contre, le qualité de prédiction de prédiction est inférieure à la notre. Dans les travaux présentés dans [CKL+10], il n'est pas possible de comparer des transactions qui ont des tailles différentes, il n'est pas possible non plus de travailler avec des règles contradictoires.
Dans les travaux de [DKSS11], des règles de préférences assez similaires que les nôtres on été proposées : Des règles comportant une partie condition et une partie décision. De la forme : "si [conjonction de conditions élémentaires] alors [décision]" (cf. section 2.3.1.2 et paragraphe 2.3.2.1.3). Par contre, ces règles ne sont pas des règles de préférences contextuelles en considérant la définition 10. Même si la parti condition est une conjonction de conditions élémentaires sur les valeurs des attributs, la partie décision représente des objets ou tuples et non la préférences sur la valeur d'un autre attributs. En plus, l'extraction de ces règles n'est pas aussi aisée : il est nécessaire de comparer les valeurs des attributs et de contraindre ces valeurs.

Dans le chapitre suivant, nous présentons notre contribution qui consiste à construire un profil utilisateur en se basant sur ses préférences. Nous utilisons une approche qualitative et des règles de préférences contextuelles pour construire un profil utilisateur. Notre approche construit un modèle facilement interprétable par les utilisateur, ce qui n'est pas toujours le cas avec les travaux présentés dans l'approche quantitative. Dans notre travail, nous modélisons les préférences contextuelles, contrairement à certains approches qualitatives [HEK03, Jea08, YWLd08, SM06, DIV07]. Le modèle que nous construisons est aussi robuste et est capable de travailler sur des transactions de taille différentes contrairement à [CKL+10]. Les règles de préférences présentées dans [DKSS11] sont assez similaires aux nôtres mais nous proposons des règles plus compactes. Par rapport à Sandra et al. [dABAdS13], notre approche fournie une qualité de prédiction meilleure.

Deuxième partie

Construction du profil Utilisateur

Table des matières

Introduction

Dans la partie précédente, nous avons montré l'insuffisance des méthodes d'apprentissage pour le classement : les méthodes qui utilisent l'approche quantitative ne produisent pas toujours un modèle facilement compréhensible par l'utilisateur, tandis que certaines méthodes qui utilisent l'approche qualitative souffrent de problème d'expressivité car ne modélisant pas les préférences contextuelles. Dans cette partie, nous présentons la méthode que nous avons proposée pour apprendre les préférences utilisateurs. Nous utilisons une approche qualitative et adaptons le cadre des règles d'associations pour construire un modèle de préférences facilement interprétable.

Cette partie est composée de deux chapitres : Dans le chapitre 3, nous présentons nos contributions à savoir : la proposition d'une approche à deux phases pour la modélisation de préférences utilisateurs. Nous présentons également les méthodes de prédiction que nous avons proposées pour prédire les préférences des utilisateurs sur de nouveaux objets ; ces méthodes de prédictions se basent sur le modèle construit.

Dans le chapitre 4, nous présentons les expérimentations que nous avons effectuées pour évaluer l'approche que nous avons proposée. Les expérimentations effectuées sur des données réelles montrent la faisabilité de notre approche. Elles montrent également que la qualité de prédiction de notre approche peut rivaliser avec celles de certaines méthodes proposées dans l'état de l'art. Les propositions que nous avons effectuées sont valorisées à travers des publications (cf.[DGS+12, dADD+12, dADD+13, dADD+14]).

Chapitre 3

Construction de profil utilisateur

Dans ce chapitre, nous présentons notre contribution à savoir : la construction d'un profil utilisateur à partir d'un ensemble de règles de préférences contextuelles. Pour construire le profil utilisateur, nous utilisons une approche à deux phases qui consiste à : (1) extraire toutes les règles de préférence contextuelles qui satisfont des contraintes fixées et (2) construire un profil utilisateur à partir des règles de préférences contextuelles.
Nous présentons également les stratégies d'utilisation des profils que nous proposons.

Ce chapitre est organisé de la façon suivante : Dans la section 3.1, nous présentons la formalisation du problème que nous comptons résoudre : Il s'agit de définir les concepts nécessaires à la compréhension du problème ; nous présentons également dans cette section, les deux problèmes que nous comptons résoudre.
Dans la section 3.2, nous présentons l'algorithme que nous avons proposé pour extraire toutes les règles de préférences contextuelles. Dans la section 3.3, nous présentons l'algorithme permettant la construction du profil utilisateur. Finalement, dans la section 3.4, nous présentons les techniques que nous avons proposées pour faire de la prédiction.

3.1 Formalisation du problème

Dans cette section nous formalisons le problème d'extraction que nous comptons résoudre. Comme nous l'avons indiqué à l'introduction de ce chapitre, cette tâche est subdivisée en deux sous-tâches :

(1) L'extraction de l'ensemble S des règles de préférences d'une base de préférences et,

(2) La construction du profil utilisateur à partir de l'ensemble S.

Cette section est organisée de la façon suivante : Nous formalisons les deux sous-tâches respectivement dans les sous-sections 3.1.2 et 3.1.3. Dans la sous-section 3.1.1, nous définissons les concepts nécessaires pour la compréhension des deux sous-tâches.

Notation	Description
S	Ensemble des règles de préférences contextuelles
$\mathcal{I}$	Un ensemble de littéraux appelés Items
X	Un itemset
$\mathcal{L}$	Le langage des itemsets
$\mathcal{D}$	Une base de données transactionnelles
$\mathcal{P}$	Une base de préférence
σ	seuil minimum de support
κ	seuil minimum de confiance
π	Une règle de preference contextuelle
$\mathcal{CP}_{\sigma,\kappa}$	Un ensemble de règles de préférences contextuelles
$\mathcal{MCP}_{\sigma,\kappa}$	représente l'ensemble des règles de préférences contextuelles minimales fréquentes et valides.
$\Pi_{\langle t,u\rangle}$	représente l'ensemble des règles de préférences contextuelles qui couvrent (t,u)

TABLE 3.1 – Notations utilisées dans ce chapitre

3.1.1 Règles de préférence contextuelle et base de préférences.

Dans cette sous-section, nous présentons les définitions nécessaires permettant de comprendre la formulation du problème d'extraction de préférences et de construction de profil utilisateurs. Ces définitions seront détaillées et seront également illustrées grâce aux données contenues dans la table 3.2. La table 3.2(a) fait la correspondance entre les lettres A, B, C, D, E et ce qu'elles représentent. Ces lettres représentent les valeurs des attributs *Directeur, Acteur, Genre* du schéma de la table 1.1 présenté dans le chapitre 1. Les lettres A, B, C, D et E représentent respectivement : *Steven-Spielberg*, *Tom Hanks*, *Action*, *Leonardo Di caprio*, et *War*.

Définition 22 (item)**.** *Etant donné un ensemble d'attributs $\mathcal{A}$, un item défini sur $\mathcal{A}$ est un couple d'attribut-valeur (A_i, A) tel que A_i est un élément de $\mathcal{A}$ et $A \in dom(A_i)$.*

Exemple 28. *La table 3.2(a) contient les items (Directeur, A), (Acteur,B), (genre, C), (Acteur, D) et (Genre, E).*

Si aucune ambiguité n'est possible, nous allons tout simplement noter les items par leur valeur. Un itemset est défini de la façon suivante :

Définition 23 (ItemSet)**.** *Etant donné un ensemble d'attributs $\mathcal{A}$, un itemset X défini sur $\mathcal{A}$ est un ensemble d'items définis sur $\mathcal{A}$.*

Exemple 29. *La table 3.2(b) contient 5 itemsets qui sont : {A,C,D},{A,B,D}, {A,B,C,E}, {C,D},{A,B}.*

Nous allons maintenant définir ce que c'est une transaction :

Définition 24 (Transaction)**.** *Une transaction est un itemset désigné par un identifiant unique t_{id}. L'ensemble de tous les identifiants de transaction sera désigné par l'ensemble $\mathcal{T}$.*

Exemple 30. *La table 3.2(b) contient 5 transactions identifiées par $t_1, \ldots, t_5$.*

Définition 25 (Base de données transactionnelle)**.** *Une base de données transactionnelle $\mathcal{D}$ est un ensemble de couples formés d'un identifiant de transaction t_{id} et de la transaction proprement dite.*

Exemple 31. *La table 3.2(b) représente une base de données transactionnelle de cinq transactions sur 5 items.*
$\mathcal{I}$={A,B,C,D,E}
$\mathcal{T}$= {t_1,t_2,t_3,t_4,t_5}
$\mathcal{D} = \{(t_1, \{A, C, D\}), (t_2, \{A, B, D\}), (t_3, \{A, B, C, E\}), (t_4, \{C, D\}), (t_5, \{A, B\})$

Définition 26 (Base de préférences)**.** *Une base de préférences $\mathcal{P} \subseteq \mathcal{D} \times \mathcal{D}$ est un ensemble de paires de transactions représentant un échantillon de préférences utilisateurs sur la base $\mathcal{D}$.*

Intuitivement, une préférence utilisateur $\langle t, u \rangle \in \mathcal{P}$ signifie que l'utilisateur préfère la transaction t à la transaction u. Etant donnée une préférence utilisateur $\langle t, u \rangle \in \mathcal{P}$, t est appelé la transaction préférée (pour l'utilisateur) et u est appelé la transaction non préférée.

Exemple 32. *La table 3.2(c) montre un ensemble de 5 préférences utilisateurs notées $p_1, \ldots, p_5$.*

La base de préférence et la base de transactions sont toutes les deux représentées par un graphe illustré à la figure 3.1. Nous tenons à préciser qu'en général, $\mathcal{P}$ n'est pas nécessairement *transitive*, puisque la base de préférences (ici, les préférences utilisateurs) sont collectées de façon implicite.

$\mathcal{I}$

item	Valeur
A	Steven Spielberg
B	Tom Hanks
C	Action
D	Leonardo DiCaprio
E	War

(a) Signification des items

$\mathcal{D}$

Tid	Transactions				
t_1	A		C	D	
t_2	A	B		D	
t_3	A	B	C		E
t_4			C	D	
t_5	A	B			

(b) Base de données transactionnelle

$\mathcal{P}$

Pid	Preference utilisateur
p_1	$\langle t_1, t_3 \rangle$
p_2	$\langle t_2, t_3 \rangle$
p_3	$\langle t_2, t_4 \rangle$
p_4	$\langle t_3, t_4 \rangle$
p_5	$\langle t_4, t_5 \rangle$

(c) Base de préférences

TABLE 3.2 – Une base de données transactionnelle et une base de préférences

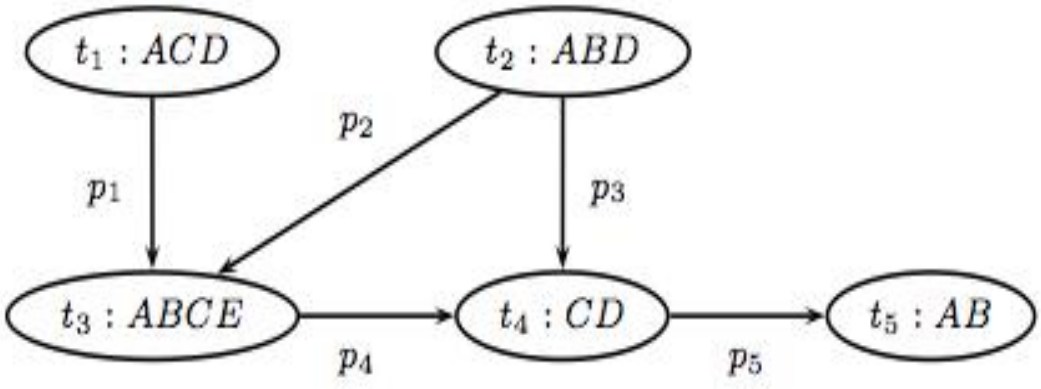


FIGURE 3.1 – Un graphe de preferences

L'objectif principal de cette thèse est d'extraire un profil utilisateur à partir de la base de préférences fournie par l'utilisateur. Un profil utilisateur est représenté par un ensemble de règles de préférences contextuelles vérifiant un certain nombre de propriétés intéressantes. Dans notre contexte, une règle de préférence contextuelle est définie de la façon suivante.

Définition 27 (Règle de préférence contextuelle [ART06])**.** *Une règle de préférence contextuelle est sous la forme* $i^+ \succ i^- \mid X$ *où* X *est un itemset de* $\mathcal{L}$*,* i^+ *et* i^- *sont des items de* $\mathcal{I} \setminus X$*.*

En gros, une règle de préférence contextuelle $i^+ \succ i^- \mid X$ signifie que quand le contexte X apparait alors l'item i^+ est préféré l'item i^-. La partie gauche de la règle de préférence

représente le choix tandis que la partie droite représente le contexte.

Exemple 33. *Par exemple, la règle de préférence contextuelle $D \succ E \,|\, AB$ signifie que le contexte AB conduit à choisir l'item D plutôt que l'item E.*

En d'autres termes, la règle de préférence contextuelle $D \succ E \,|\, AB$ indique que si deux transactions contiennent le contexte AB, alors la transaction contenant l'item D est préférée à celle contenant l'item E.
$\mathcal{CP}(\mathcal{L})$ représente l'ensemble des règles de préférences contextuelles basées sur $\mathcal{L}$ (assez souvent, nous omettons le langage lorsqu'il est implicite dans le contexte). L'intérêt d'avoir une règle de préférence contextuelle, $i^+ \succ i^- \mid X$, est la capacité de comparer des transactions.
Une transaction t est préférée à une transaction u selon $\pi : i^+ \succ i^- \mid X$ noté $t \succ_\pi u$, si $(X \cup \{i^+\} \subseteq t) \wedge (X \cup \{i^-\} \subseteq u) \wedge (i^- \notin t) \wedge (i^+ \notin u)$.

Exemple 34. *Par exemple, la transaction ACD est préférée à la transaction $ABCE$ selon la règle de préférence contextuelle $D \succ E \,|\, A$.*

Définition 28. *Une règle de préférence contextuelle π est en accord avec une préférence utilisateur $\langle t, u\rangle \in \mathcal{P}$ si la règle de préférence π conduit à préférer t à u. L'accord d'une règle de préférence contextuelle π dans $\mathcal{P}$ est définie comme suit :*

$$agree(\pi, \mathcal{P}) = \{\langle t, u\rangle \in \mathcal{P} \mid t \succ_\pi u\}$$

Exemple 35. *La règle de préférence contextuelle $D \succ E \,|\, A$ est en accord avec la préférence utilisateur $p_1 = \langle t_1, t_3\rangle$ de la table 3.2(c). L'accord de la règle $D \succ E \,|\, A$ dans $\mathcal{P}$ est l'ensemble $\{p_1, p_2\}$.*

Définition 29. *Une règle de préférence contextuelle π est en désaccord avec une préférence utilisateur $\langle t, u\rangle \in \mathcal{P}$ si la règle de préférence π conduit à préférer u à t. La* contradiction *d'une règle de préférence contextuelle π dans $\mathcal{P}$ est définie comme suit :*

$$contradict(\pi, \mathcal{P}) = \{\langle t, u\rangle \in \mathcal{P} \mid u \succ_\pi t\}$$

Exemple 36. *La règle de préférence contextuelle $B \succ D \mid C$ est en désaccord avec la préférence utilisateur $p_1 = \langle t_1, t_3\rangle$ de la table 3.2(c). La contradiction de la règle de préférence $B \succ D \,|\, C$ est $\{p_1\}$.*

Définition 30. *Si une règle de préférence contextuelle π est en accord ou en désaccord avec une préférence utilisateur $\langle t, u\rangle \in \mathcal{P}$, la règle de préférence contextuelle couvre la préférence utilisateur. La* couverture *d'une règle de préférence contextuelle π dans $\mathcal{P}$ est*

définie comme suit :

$$cov(\pi, \mathcal{P}) = agree(\pi, \mathcal{P}) \cup contradict(\pi, \mathcal{P})$$

Exemple 37. *Par exemple, la préférence utilisateur* $p_1 = \langle t_1, t_3 \rangle$ *dans la table 3.2(c) est couverte à la fois par la règle* $D \succ E \,|\, A$ *(qui est en accord avec la préférence) et la règle* $B \succ D \,|\, C$ *(qui est en contradiction avec la préférence). La couverture de la règle* $B \succ D \,|\, C$ *dans* $\mathcal{P}$ *est* $\{p_1, p_4\}$.

Comme nous l'avons indiqué au début de ce chapitre, notre problème consiste à extraire un profil utilisateur à partir d'une base de préférences fournie par un utilisateur. Ce problème est subdivisé en deux sous tâches qui consistent à : (1) extraire toutes les règles de préférences contextuelles intéressantes et minimales et (2) de construire un profil utilisateur à partir des règles extraites.

3.1.2 Problème d'extraction de règles de préférences

Dans cette sous-section, nous présentons la première sous tâche de notre problème, à savoir le problème d'extraction de règles de préférences.
Dans cette tâche, nous avons adapté le cadre des règles d'associations, plus précisément, celui de support-confiance des règles d'associations. Dans cette adaptation, nous avons considéré que le contexte X et la préférence $i^+ \succ i^-$ (cf. Définition 27) correspondent respectivement à l'antécédent et au conséquent d'une règle d'association.
Nous avons défini, de façon analogue, le concept de *support*, *confiance* et de la *minimalité* comme critères intéressants pour éliminer les règles de préférences contextuelles non-intéressantes.

Exemple 38. *En considérant l'exemple jouet présenté dans la table 3.2, la règle de préférences contextuelles* $D \succ E \,|\, A$ *est plus intéressante que* $D \succ E \,|\, B$ *parce que* $D \succ E \,|\, A$ *est en accord avec deux préférences (*p_1 *et* p_2*) tandis que* $D \succ E \,|\, B$ *est en accord avec une seule préférence utilisateur (*p_2*). Cette information est fournie par le* support *d'une règle de préférence contextuelle* π *qui estime la probabilité que* π *soit en accord avec une paire* $\mathcal{P}$.

Définition 31 (Support)**.** *Le support d'une règle de préférence contextuelle* π *dans* $\mathcal{P}$ *est défini comme suit :*

$$supp(\pi, \mathcal{P}) = \frac{|agree(\pi, \mathcal{P})|}{|\mathcal{P}|}$$

Exemple 39. *Par exemple, comme les transactions* ACD *et* ABD *sont préférées à la transaction* $ABCE$ *selon la règle* $D \succ E \mid A$*, nous obtenons* $supp(D \succ E \mid A, \mathcal{P}) = |\{p_1, p_2\}|/|\mathcal{P}| = 0.4$. *De façon similaire,* $supp(D \succ E \,|\, B, \mathcal{P}) = |\{p_2\}|/|\mathcal{P}| = 0.2$.

L'intérêt d'une règle de préférences contextuelles augmente avec son support : la règle de préférences contextuelles $D \succ E \,|\, A$ est donc plus intéressante que la règle $D \succ E \,|\, B$.
Maintenant nous avons besoin d'évaluer le désaccord entre une règle de préférence contextuelle et une base de préférence. Ce qui veut dire, parmi toutes les préférences utilisateur que couvrent une règle, combien sont en accord avec cette règle ? Pour arriver à cette fin, nous définissons la *confiance* d'une règle de préférence contextuelle π en considérant une base de préférences $\mathcal{P}$ comme suit :

Définition 32 (Confiance)**.** *La confiance de π en considérant $\mathcal{P}$ est définie comme suit :*

$$conf(\pi, \mathcal{P}) = \frac{|agree(\pi, \mathcal{P})|}{|cov(\pi, \mathcal{P})|}$$

En d'autres termes, la confiance évalue si une règle de préférence contextuelle contredit plusieurs préférences utilisateurs. Ce critère montre que $D \succ E \mid A$ est plus *valide* que $D \succ E \,|\, \emptyset$ parce que $conf(D \succ E \,|\, A, \mathcal{P}) = 2/2 = 1$ alors que $conf(D \succ E \,|\, \emptyset, \mathcal{P}) = 2/3$.
Une règle de préférence contextuelle excédant un seuil minimum de support σ (resp. un seuil minium de confiance κ) est dite *frequente* (resp. *valide*). L'ensemble de toutes les règles fréquentes et valides par rapport au seuil minimum de support σ et κ est $\mathcal{CP}_{\sigma,\kappa}(\mathcal{L}, \mathcal{P})$ (ou $\mathcal{CP}_{\sigma,\kappa}$ en bref).
A ce stade, le support et la confiance écartent respectivement les règles de préférences contextuelles non-fréquentes et non valides. Ces mesures d'intérêt évaluent la qualité intrinsèque d'une règle de préférence contextuelle, mais ne considèrent pas la rédondance entre plusieurs règles de préférences contextuelles de $\mathcal{CP}_{\sigma,\kappa}$.

Exemple 40. *Etant donné l'exemple de la table 3.2, nous constatons que $D \succ E \mid B$ et $D \succ E \mid AB$ ont le même support et la même confiance. Intuitivement, la règle de préférence contextuelle $D \succ E \mid B$ est plus pertinente que $D \succ E \mid AB$ parce que son contexte est plus petit.*

Pour cette raison nous introduisons la notion de règles de préférences contextuelles *minimales.*

Définition 33 (règle de préférence minimale)**.** *Une règle de préférences contextuelles $i^+ \succ i^- \,|\, X$ est minimale dans $\mathcal{P}$ ssi il n'existe pas de règles de préférences contextuelles $i^+ \succ i^- \,|\, Y$ tel que $Y \subset X$ et $supp(i^+ \succ i^- \,|\, Y, \mathcal{P}) = supp(i^+ \succ i^- \,|\, X, \mathcal{P})$ et $conf(i^+ \succ i^- \,|\, Y, \mathcal{P}) = conf(i^+ \succ i^- \,|\, X, \mathcal{P})$.*

Exemple 41. *Typiquement, la règle de préférence contextuelle $D \succ E \,|\, B$ est minimale, mais $D \succ E \,|\, AB$ n'est pas minimale.*

Pour ce qui suit, $\mathcal{MCP}_{\sigma,\kappa}(\mathcal{L},\mathcal{P})$ (ou $\mathcal{MCP}_{\sigma,\kappa}$) représente l'ensemble des règles de préférences contextuelles minimales avec un support et une confiance respectivement plus grands que σ et κ.
Dans la pratique, ce critère de minimalité réduit le nombre de règles de préférences contextuelles. Etant donné un échantillon de base de préférences, le premier problème que nous considérons concerne l'extraction de toutes les règles de préférences contextuelles *intéressante*, i.e ces règles qui sont minimales et ayant un support et une confiance acceptable. Plus précisément :

Problème 2 (Extraction de règles de preferences)**.** *Etant donnée une base de préférences $\mathcal{P}$, un seuil minimum de support σ et un seuil minimum de confiance κ, trouver l'ensemble $\mathcal{MCP}_{\sigma,\kappa}$ des règles de préférences contextuelles minimale.*

Etant donné un ensemble d'items $\mathcal{I}$ où $k = |\mathcal{I}| \geq 2$, il existe $k \times (k-1) \times 2^{k-2}$ règles de préférences contextuelles distinctes. Clairement, une énumération naïve des règles de préférences pour calculer $\mathcal{MCP}_{\sigma,\kappa}$ est infaisable et quelques critères d'élagage sont nécessaires pour réduire l'espace de recherche. Dans notre première contribution[dADD+12], nous avons utilisé un algorithme par niveau pour calculer $\mathcal{MCP}_{\sigma,\kappa}$, tandis que dans [dADD+14] nous utilisons la propriété d'antimonotonie du support de $\mathcal{MCP}_{\sigma,0}$ pour réduire l'espace de recherche.

3.1.3 Problème de construction de profil utilisateur

Dans la sous-section suivante, nous présentons le second problème qui consiste à construire le profil utilisateur à partir des règles de préférences contextuelles extraites dans la première phase.
Dans notre approche, un profil utilisateur est représenté par un ensemble de règles de préférences *concis* et *consistants* selon l'échantillon de préférences que l'utilisateur a précédemment fourni.
La *concision* d'un ensemble de règles de préférences est évaluée par sa cardinalité tandis que la *consistance* d'un ensemble de règles est évaluée par une fonction coût.
Dans la sous-section 3.1.1, nous avons défini l'accord et la contradiction d'une règle de préférence contextuelle π. Nous allons maintenant définir les notions d'accord, de contradiction et de couverture sur un ensemble de règles de préférences contextuelles Π.

Définition 34. *Soient une base de préférences $\mathcal{P}$ et un ensemble de règles de préférences contextuelle Π. L' ensemble des préférences utilisateur en accord avec Π est défini comme suit :*

$$agree(\Pi,\mathcal{P}) = \cup_{\pi\in\Pi} agree(\pi,\mathcal{P})$$

Exemple 42. *En considérant l'ensemble des règles de préférences contextuelles* $\Pi = \{D \succ E \mid A, B \succ D \mid C\}$*, l'ensemble des préférences utilisateur dans* $\mathcal{P}$ *en accord avec* Π *est* $agree(\Pi, \mathcal{P}) = \{ p_1, p_2, p_4\}$.

Définition 35. *Soient une base de préférences* $\mathcal{P}$ *et un ensemble de règles de préférences contextuelle* Π*. L'ensemble des préférences utilisateur en contradiction avec* Π *est défini comme suit :*

$$contradict(\Pi, \mathcal{P}) = \cup_{\pi \in \Pi} contradict(\pi, \mathcal{P})$$

Exemple 43. *En considérant l'ensemble des règles de préférences contextuelles* $\Pi = \{D \succ E \mid A, B \succ D \mid C\}$*, l'ensemble des préférences utilisateur dans* $\mathcal{P}$ *en contradiction avec* Π *est* $contradict(\Pi, \mathcal{P}) = \{ p_1\}$.

Définition 36. *Soient une base de préférences* $\mathcal{P}$ *et un ensemble de règles de préférences contextuelle* Π*. L'ensemble des préférences utilisateur couvertes par* Π *est définie comme suit :*

$$cover(\Pi, \mathcal{P}) = agree(\Pi, \mathcal{P}) \cup contradict(\Pi, \mathcal{P})$$

Exemple 44. *En considérant l'ensemble des règles de préférence contextuelles* $\Pi = \{D \succ E \mid A, B \succ D \mid C\}$ *, l'ensemble des préférences utilisateur dans* $\mathcal{P}$ *couverte par* Π *est* $cover(\Pi, \mathcal{P}) = \{ p_1, p_2, p_4\}$.

En utilisant cette notation, nous pouvons maintenant définir le coût d'un profil utilisateur comme suit :

Définition 37 (Coût)**.** *Etant donnée une base de préférence* $\mathcal{P}$ *et un ensemble de règles de préférences contextuelle* Π*, le coût de* Π *selon* $\mathcal{P}$*, noté* $Cost(\Pi, \mathcal{P})$*, est défini par :*

$$Cost(\Pi, \mathcal{P}) = \frac{|\mathcal{P} \setminus cover(\Pi, \mathcal{P})| + |contradict(\Pi, \mathcal{P})|}{|\mathcal{P}|}$$

Intuitivement, le coût d'un profil utilisateur Π représente le pourcentage de préférences utilisateur dans $\mathcal{P}$ qui n'est couverte par aucune règle dans Π ou qui est en contradiction avec quelques règles dans Π. En utilisant cette définition, nous considérons que le profil utilisateur est *consistant* si son coût est minimal.
Nous pouvons maintenant définir précisément le second problème principal i.e la construction du profil utilisateur qui est *concis* et *consistant* en considérant un ensemble de préférences utilisateur.

Problème 3 (Construction de profil utilisateur)**.** *Etant données une base de préférences* $\mathcal{P}$ *et un ensemble de règles de préférences contextuelles* S*, sélectionner un profil* $\Pi \subseteq S$ *qui est concis et qui minimise de coût.* Π *est appelé le profil utilisateurassocié à* $\mathcal{P}$.

Il est à noter que dans la déclaration du problème, S peut être n'importe quel ensemble de règles de préférences. Dans la pratique, S sera l'ensemble des règles de préférences contextuelles minimales, comme défini dans le problème 3.2. Il est aussi important de noter qu'avec un gros volume de données, la construction d'un profil utilisateur qui est concis au maximum est un problème difficile. En plus, il peut être facile de voir que notre problème est similaire au problème du *Positive-Negative Partial Set Cover* ($\pm psc$) introduit dans [Mie08].

Théoreme 1. *Soit le problème de couverture de préférences défini comme suit :*

- **Entrée :** *un tuple* $\langle \mathcal{P}, \mathcal{R}, \lambda \rangle$ *où* $\mathcal{P}$ *est une base de préférence,* $\mathcal{R}$ *est un ensemble de règles de preferences contextuelles et* $0 \leq \lambda \leq 1$.
- **Question :** *Y a-t-il* $\Pi \subseteq \mathcal{R}$ *tel que* $Cost(\Pi, \mathcal{P}) \leq \lambda$ *?*

Le problème de couverture de préférences est NP-complet.

Démonstration. Nous définissons une réduction polynomial du problème de décision $\pm psc$ introduit dans [Mie08] dans le problème de *couverture de* préférences. Une instance de $\pm psc$ est un tuple $\langle N, P, \mathcal{Q}, \lambda \rangle$ où N et P sont des ensembles disjoints d'éléments positifs et négatifs, $\mathcal{Q} = \{Q_1, \ldots, Q_m\} \subseteq 2^{N \cup P}$ est une collection de $N \cup P$ et $0 \leq \lambda \leq 1$. Le but est de trouver une collection $\mathcal{D} \subseteq \mathcal{Q}$ tel que $Cost_{\pm}(\mathcal{D}, N, P) \leq \lambda$ où :

$$Cost_{\pm}(\mathcal{D}, N, P) = \frac{|P \setminus (\cup_{Q_i \in \mathcal{D}} Q_i)| + |N \cap (\cup_{Q_i \in \mathcal{D}} Q_i)|}{|P|}$$

Etant donnée une instance $\langle N, P, \mathcal{Q}, \lambda \rangle$ de $\pm psc$, la transformation vers le problème de *couverture de* préférences est définie comme suit : pour chaque élément $e \in (\cup_{Q_i \in \mathcal{Q}} Q_i)$, soit $X_e = \{i \in [1, m] \mid e \in Q_i\}$. Si $e \in N$, nous définissons la préférence utilisateur $p_e = \langle X_e \cup \{-\}, X_e \cup \{+\} \rangle$. D'autre part, si $e \in P$, nous définissons le profil utilisateur $p_e = \langle X_e \cup \{+\}, X_e \cup \{-\} \rangle$. De plus, pour chaque $i \in [1, m]$ et $e \in N$, nous définissons les règles de préférences contextuelles $\pi_i = + \succ - \mid i$ et $r_e = - \succ + \mid X_e$. Par construction, il est à noter que chaque règle π_i est en accord avec la préférence utilisateur p_e ssi $e \in P \cap Q_i$ et contredit une préférence utilisateur p_e ssi $e \in N \cap Q_i$. D'autre part, chaque règle r_e est seulement en accord avec la préférence utilisateur p_e. Etant donnée une instance $\langle N, P, Q, \lambda \rangle$ de $\pm psc$, nous pouvons maintenant définir l'instance $\langle \mathcal{P}, \mathcal{R}, \lambda \rangle$ du problème de *couverture de* préférences où $\mathcal{P} = \{p_e \mid e \in (\cup_{Q_i \in \mathcal{Q}} Q_i)\}$ et $\mathcal{R} = \{\pi_i \mid i \in [1, m]\} \cup \{r_e \mid e \in N\}$. La transformation peut être calculée en un temps polynomial. Maintenant, nous avons à montrer que $\langle N, P, \mathcal{Q}, \lambda \rangle$ est une instance de $\pm psc$ si et seulement si le tuple $\langle \mathcal{P}, \mathcal{R}, \lambda \rangle$ obtenue par transformation est une instance du problème de *couverture de* préférences.

Supposons que $\langle N, P, \mathcal{Q}, \lambda \rangle$ soit une instance de $\pm psc$, cela signifie qu'il existe une collection $\mathcal{D} \subseteq \mathcal{Q}$ tel que $Cost_{\pm}(\mathcal{D}, N, P) \leq \lambda$. Soit $\Pi = \{\pi_i \in \mathcal{R} \mid Q_i \in \mathcal{D}\} \cup \{r_e \mid e \in N\}$.

Nous pouvons voir que :

- $contradict(\Pi, \mathcal{P}) = \{p_e \in \mathcal{P} \mid e \in N \cap (\cup_{Q_i \in \mathcal{D}} Q_i)\}$, ce qui montre que $|contradict(\Pi, \mathcal{P})| = |N \cap (\cup_{Q_i \in \mathcal{D}} Q_i)|$, et
- $cover(\Pi, \mathcal{P}) = agree(\Pi, \mathcal{P}) \cup contradict(\Pi, \mathcal{P}) = (\{p_e \in \mathcal{P} \mid e \in P \cap (\cup_{Q_i \in \mathcal{D}} Q_i)\} \cup \{p_e \in \mathcal{P} \mid e \in N\}) \cup \{p_e \in \mathcal{P} \mid e \in N \cap (\cup_{Q_i \in \mathcal{D}} Q_i)\} = \{p_e \in \mathcal{P} \mid e \in \cup_{Q_i \in \mathcal{D}} Q_i\} \cup \{p_e \in \mathcal{P} \mid e \in N\})$ ce qui montre que $|\mathcal{P} \setminus cover(\Pi, \mathcal{P})| = |P \setminus (\cup_{Q_i \in \mathcal{D}} Q_i)|$.

Par conséquent, nous avons :$cost(\Pi, \mathcal{P}) = cost_{\pm}(\mathcal{D}, N, P)) \leq \lambda$ ce qui prouve que $\langle \mathcal{P}, \mathcal{R}, \lambda \rangle$ est une instance du problème de *couverture de* préférences. Nous devons maintenant prouver l'inverse, i.e. que si$\langle \mathcal{P}, \mathcal{R}, \lambda \rangle$ est une instance de *couverture de* préférences, alors $\langle N, P, \mathcal{Q}, \lambda \rangle$ est une instance de $\pm psc$. Si $\langle \mathcal{P}, \mathcal{R}, \lambda \rangle$ est une instance du problème de *couverture de* préférences, cela signifie qu'il existe un profil utilisateur $\Pi \subseteq \mathcal{R}$ tel que $cost(\Pi, \mathcal{P}) \leq \lambda$. Soit $\Pi^* = \Pi \cup \{r_e \mid e \in N\}$ et $\mathcal{D} = \{Q_i \in \mathcal{Q} \mid \pi_i \in \Pi\}$. En utilisant le même principe que précédemment, nous pouvons prouver facilement que $cost_{\pm}(\mathcal{D}, N, P) = cost(\Pi^*, \mathcal{P}) \leq cost(\Pi, \mathcal{P}) \leq \lambda$, ce qui montre que $\langle N, P, \mathcal{Q}, \lambda \rangle$ est une instance de $\pm psc$ et complète la preuve. □

Puisque la construction d'un profil utilisateur est NP-difficile, nous proposons dans la section 3.3 une approche heuristique basée sur la même idée que les méthodes de classification associative telle queCBA [LHM98]. Plus précisément, étant donnée une base de préférences, et un ensemble de règles de préférences intéressantes, nous présentons un algorithme glouton appelé ProfMiner permettant de construire un profil *concis* et *consistant*.

3.1.4 Conclusion

Dans cette section, nous avons présenté la formulation du problème que nous comptons résoudre dans cette thèse. Nous avons indiqué que le problème de construction de profil est composé de deux sous-tâches.

1. Une phase d'extraction de toutes les règles de préférences contextuelles minimales dont le support est supérieur à un seuil minimum de support fixé et dont la confiance est supérieure à un seuil minimum de confiance fixé. Cet ensemble est nommé $\mathcal{MCP}_{\sigma,\kappa}$.
2. La seconde sous-tâche consiste à sélectionner un sous ensemble de l'ensemble des règles, extraites au niveau de la première phase, pour constituer le profil utilisateur. Ce sous ensemble sélectionné à partir des règles de préférences extraites au niveau de la première phase doit être concis et consistant.

Dans la section 3.1, nous avons présenté les concepts nécessaires pour la compréhension des deux sous tâches.

Dans ce qui suit, nous allons présenter les solutions que nous avons proposées pour extraire les règles de préférences contextuelles minimales (cf. section 3.2) et pour construire le profil utilisateur (cf. section 3.3).

3.2 Extraction de règles de préférence contextuelle

Comme nous l'avons indiqué dans la sous-section 3.1.2, pour faire face au problème 2, nous utilisons un critère d'élagage durant le parcourt en profondeur de l'espace de recherche $\mathcal{CP}$. Pour cette raison, nous énumérons les contextes en se basant sur un ordre $\preceq_{\mathcal{L}}$. Plus précisément, nous utilisons un ordre total arbitraire $<_{\mathcal{I}}$ sur l'ensemble de tous les items et nous supposons que les contextes sont écrits en se basant sur cet ordre.

Définition 38. *Un contexte $X = i_1 i_2 \dots i_l$ est appelé un* contexte préfixe *de $Y = i_1 i_2 \dots i_m$, noté par $X \preceq_{\mathcal{L}} Y$, où $l \leq m$ et les items dans les deux contextes sont listés en se basant sur l'ordre $<_{\mathcal{I}}$.*

Exemple 45. *Par exemple, les contextes préfixe de AB sont $\emptyset$, A et AB, mais non B.*

L'ordre partiel $\preceq_{\mathcal{L}}$ est utile pour bénéficier du support et de la minimalité pour réduire l'espace de recherche.

Propriété 1 (Critère d'élagage)**.** *Si une règle de préférence contextuelle $i^+ \succ i^- \mid X$ n'est pas fréquente ou n'est pas minimale, alors il n'existe aucune règle de préférence contextuelle $i^+ \succ i^- \mid Y$ tel que $X \preceq_{\mathcal{L}} Y$ fréquente et minimale.*

Démonstration. Etant donnés i^+ et i^-, le problème est d'extraire les contextes i.e., les itemsets. Nous pouvons par conséquent utiliser le formalisme proposé dans [SC08]. Comme le support et la confiance sont deux fonctions condensables[1], l'ensemble des contextes minimaux est fermé par le bas (en considérant l'inclusion). Nous concluons par conséquent que la propriété 1 est correcte puisque tous les motifs comparables en considérant l'ordre $\preceq_{\mathcal{L}}$ sont également comparables en considérant l'inclusion. □

La propriété 1 signifie que la contrainte de fréquence et la contrainte de minimalité sont anti-monotones en considérant $\preceq_{\mathcal{L}}$ [MT97]. Il est naturel d'intégrer ces techniques dans l'algorithme en profondeur comme proposé par l'algorithme 3.1.

En fait, CONTPREFMINER construit toutes les paires d'items (i_1, i_2) et son sous-programme CONTENUM (voir algorithme 3.2) énumère en profondeur toutes les règles de préférences contextuelles sur $i_1 \succ i_2$ ou $i_2 \succ i_1$.

1. Une fonction f est condensable si pour chaque itemset $X \subseteq Y$, $f(X \cup \{i\}) = f(X)$ implique que $f(Y \cup \{i\}) = f(Y)$.

Algorithm 3.1 ContPrefMiner

Entrées Une base de préférence $\mathcal{P}$, un seuil minimum de support σ, un seuil minimum de confiance κ

Sortie Toutes les règles de préférences, ayant un contexte-préfixe, minimal excédant σ et κ

1: $S := \emptyset$
2: **Pour Tout** paire d'items $(i_1, i_2) \in \mathcal{I} \times \mathcal{I}$ **Faire**
3: $s_{12} := supp(i_1 \succ i_2 \,|\, \emptyset, \mathcal{P})$
4: $s_{21} := supp(i_2 \succ i_1 \,|\, \emptyset, \mathcal{P})$
5: **Si** $s_{12} \geq \sigma$ **ou** $s_{21} \geq \sigma$ **alors**
6: **si** $s_{12} \geq \sigma$ & $s_{12}/(s_{12} + s_{21}) \geq \kappa$ **Alors**
7: $S := S \cup \{i_1 \succ i_2 \,|\, \emptyset\}$
8: **fin si**
9: **si** $s_{21} \geq \sigma$ & $s_{21}/(s_{12} + s_{21}) \geq \kappa$ **Alors**
10: $S := S \cup \{i_2 \succ i_1 \,|\, \emptyset\}$
11: **fin si**
12: $S := S \cup$ ContEnum$((i_1, i_2), \emptyset, \mathcal{I} \setminus \{i_1, i_2\}, \mathcal{P}, \sigma, \kappa)$
13: **Fin Si**
14: **Fin Pour**
15:
16: **Retourner** S

La ligne 5 dans ContPrefMiner et ContEnum arrête l'énumération des règles de préférences si les deux supports sont plus petits que le seuil de support minimum (propriété 1). La minimalité du contexte-préfixe est seulement évaluée à la ligne 5 de ContEnum puisque toutes les règles de préférences basées sur $\emptyset$ sont minimale.
En plus, une règle de préférence n'a pas un contexte-prefixe minimale dès qu'une règle de préférence contextuelle concluant sur $i_1 \succ i_2$ et $i_2 \succ i_1$ possédant un contexte plus petit conduit au même support.
Finalement, l'ordre arbitraire $<_{\mathcal{I}}$ est utilisé à la ligne 12 de ContEnum pour énumérer toutes les prochaines règles de préférences contextuelles en se basant sur $\preceq_{\mathcal{L}}$ comme expliqué plus haut.

Une des forces de l'approche à deux phases repose sur la consistance et la complétude de la phase d'extraction de règles de préférences [KCFS08] comme c'est la cas avec ContPrefMiner :

Théoreme 2 (Consistance et complétude)**.** *Etant donnée une base de préférence $\mathcal{P}$, un seuil minimum de support σ et κ,* ContPrefMiner *retourne toutes les règles de préférences contextuelles de $\mathcal{CP}_{\sigma,\kappa}$ qui sont minimales en considérant $\preceq_{\mathcal{L}}$.*

Démonstration. Etant donnée une règle $\pi = i^+ \succ i^- \mid X \in \mathcal{CP}_{\sigma,\kappa}$, il y' a au moins une paire d'items (i^+, i^-) ou (i^-, i^+) qui est donné comme paramètre de ContEnum avec un contexte vide . Alors, ContEnum exécute le parcourt en profondeur habituel de l'espace de recherche pour construire le contexte X de π. Il est à noter que les conditions d'arrêts

Algorithm 3.2 CONTENUM

ENTRÉES Une paire d'items (i_1, i_2), un contexte X, un ensemble d'items candidats $\mathcal{C}$, une base de préférences $\mathcal{P}$, un seuil minimum de support σ, un seuil minimum de confiance κ

SORTIE Toutes les règles de préférences contextuelles ayant un prefixe-minimal excédant σ et κ concluant sur $i_1 \succ i_2$ ou $i_2 \succ i_1$ dont le contexte contient X comme préfixe

1: $S := \emptyset$
2: **POUR TOUT** $i \in \mathcal{C}$ **FAIRE**
3: $s_{12} := supp(i_1 \succ i_2 \,|\, X \cup \{i\}, \mathcal{P})$
4: $s_{21} := supp(i_2 \succ i_1 \,|\, X \cup \{i\}, \mathcal{P})$
5: **SI** $((s_{12} \geq \sigma) \vee (s_{21} \geq \sigma)) \wedge ((s_{12} < supp(i_1 \succ i_2 \,|\, X)) \vee (s_{21} < supp(i_2 \succ i_1 \,|\, X)))$ **ALORS**
6: **si** $(s_{12} \geq \sigma) \wedge (s_{12}/(s_{12} + s_{21}) \geq \kappa)$ **Alors**
7: $S := S \cup \{i_1 \succ i_2 \,|\, X \cup \{i\}\}$
8: **fin si**
9: **si** $(s_{21} \geq \sigma) \wedge (s_{21}/(s_{12} + s_{21}) \geq \kappa)$ **Alors**
10: $S := S \cup \{i_2 \succ i_1 \,|\, X \cup \{i\}\}$
11: **fin si**
12: $S := S \cup$ CONTENUM$((i_1, i_2), X \cup \{i\}, \{c \in \mathcal{C} | i <_{\mathcal{I}} c\}, \mathcal{P}, \sigma, \kappa)$
13: **FIN SI**
14: **FIN POUR**
15:
16: **RETOURNER** S

de la ligne 5 sont sûres grâce au critère d'élagage des règles minimales fréquentes (cf. Propriété 1). Alors, l'approche d'extraction est complète.
Finalement, de la ligne 5 à la ligne 10, nous avons le code garantissant la consistance en supprimant les règles dont la confiance n'est pas suffisante. □

Une règle de préférence contextuelle ayant un contexte-prefixe minimal est seulement minimale en se basant sur son contexte.

Exemple 46. *Par exemple, la minimalité du contexte-prefixe de $D \succ E \,|\, AB$ ne considère pas la règle $D \succ E \,|\, B$ parce que B n'est pas un contexte-prefixe de AB.*

Alors, la minimalité du contexte-prefixe est moins restrictive que la minimalité et l'ensemble des règles de préférences ayant un contexte-prefixe minimal est un sur-ensemble de $\mathcal{MCP}_{\sigma,\kappa}$.
Dans l'approche en profondeur, la minimalité du contexte-prefixe est plus facile à évaluer que la minimalité. Nous montrons dans la section 3.3 que l'utilisation des règles de préférences contextuelles ayant un contexte-prefixe minimal au lieu des règles de préférences minimale n'affecte pas la construction du profil.
Dans la pratique, le nombre de règles de préférences contextuelles minimales est souvent une très grande collection excédant la taille du jeu de données original. L'utilisateur final ne peut pas directement analyser cette sortie importante. En fait, les motifs les plus pertinents sont perdus parmi beaucoup d'informations triviaux, redondantes et d'information

bruitées.

Exemple 47. *Par exemple, même avec le petit jeu de données présenté sur la table 3.2, nous extrayons 20 règles de préférences contextuelles ayant une confiance plus grande que 0.5.*

La solution la plus fréquente consistant à augmenter le support ou la confiance n'est pas satisfaisante.
En augmentant le seuil minimum de support les motifs extraits deviennent très généraux et seront souvent triviaux. L'augmentation du seuil de confiance minimal est contreproductive si on travail sur des données bruitées ; en plus, elle ne réduit pas assez le nombre de motifs.

Exemple 48. *Par exemple en considérant le jeu de données de la table 3.2, il y'a 8 règles de préférences contextuelles avec* $\kappa = 1$.

Dans la section suivante, nous allons présenter la solution que nous avons proposée pour réduire le nombre de règles de préférences contextuelles extraites par l'algorithme ContPrefMiner. L'objectif est de construire un profil utilisateur à partir des règles de préférence contextuelles extraites par ContPrefMiner sans augmenter le seuil minimum de support σ et de confiance κ.

3.3 Construction de profil utilisateur

Comme nous l'avons formalisé par le problème 3, le but de l'étape de la construction de profil est de trouver le plus petit ensemble de règles de préférences contextuelles qui est en accord avec toutes les préférences (ce qui veut dire un profil qui contredit un nombre minimum de préférences utilisateur).
Pour ce faire, la construction du profil utilisateur répète les deux principes suivants :

1. sélectionner la meilleure règle de préférence contextuelle,
2. supprimer, dans S, les règles de préférences contextuelles non nécessaires

Ces deux principes sont répétés sur les règles retournées par ContPrefMiner jusqu'à ce que toutes les préférences utilisateur dans la base de préférences soient en accord avec au moins une règle de préférences dans le profil.
En effet, le principe utilisé pour la construction du profil utilisateur est vraiment intéressant puisque, même si le critère de minimalité supprime plusieurs règles de préférences contextuelles redondantes, quelques règles de préférences contextuelles superflues restent parmi celles retournées par ContPrefMiner.

Exemple 49. *Par exemple, dans notre exemple jouet, la règle de préférence contextuelle* $D \succ B \,|\, A$ *(Seulement en accord avec* p_1*) peut être supprimée de* $\mathcal{MCP}_{0.2,0.6}$ *(cf. table 3.3) puisque* $D \succ E \,|\, A$ *est déjà en accord avec* p_1 *et a un meilleur support (avec la même confiance).*

Plus généralement, une règle de préférence contextuelle π est pertinente pour le profil en construction, si elle est en accord avec des préférences utilisateur de $\mathcal{P}$ qui ne sont pas déjà en accord avec d'autres meilleures règles. Il est à noter que ce genre de processus itératif pour construire un modèle est assez classique dans la littérature [BZ07].

3.3.1 Ordonner les règles de preferences contextuelles

La principale stratégie de l'algorithme PROFMINER permettant la construction du profil utilisateur est la capacité à sélectionner la *meilleure* règle de préférence contextuelle permettant de décider entre deux transactions, quelle est la préférée. La définition suivante introduit un ordre total sur l'ensemble des règles de préférences contextuelles $\mathcal{MCP}$.

Définition 39 (*Best rule order*)**.** *L'ordre de la meilleure règle sur* $\mathcal{MCP}$*, noté par* $>_{best}$*, est un ordre total défini pour toutes règles de préférences contextuelles* π *et* π' *comme :*

$$\pi >_{best} \pi' \Leftrightarrow \begin{cases} conf(\pi) > conf(\pi') \ ou \\ conf(\pi) = conf(\pi') \ et \ supp(\pi) > supp(\pi') \ ou \\ conf(\pi) = conf(\pi') \ et \ supp(\pi) = supp(\pi') \\ \quad et \ |context(\pi)| < |context(\pi')| \ ou \\ conf(\pi) = conf(\pi') \ et \ supp(\pi) = supp(\pi') \\ \quad et \ |context(\pi)| = |context(\pi')| \ et \ \pi <_{\mathcal{CP}} \pi' \end{cases}$$

Comme le profil doit contredire au plus un très petit nombre de préférences utilisateur (dans le but d'avoir une précision élevée), les critères suivants sont utilisées par ordre de priorité pour classer les règles de préférences : la confiance, le support, la taille des contextes. Le quatrième critère (où $<_{\mathcal{CP}}$ est un ordre total arbitraire) est seulement utilisé pour décider définitivement entre deux règles indifférenciées par les trois premiers critères. L'ordre $<_{\mathcal{CP}}$ est un ordre lexicographique classique sur $\mathcal{CP}$ se basant sur $<_{\mathcal{I}}$ où chaque règle de préférence contextuelle $i^+ \succ i^- \,|\, X$ est considérée comme un mot $i^+ i^- X$. Deux règles de préférences contextuelles sont ordonnées par i^+, ensuite i^- et finalement, par le contexte X en utilisant l'ordre $<_{\mathcal{I}}$.

La partie gauche de la table 3.3 illustre l'ordre induit par $>_{best}$ sur les règles de préférence minimales extraites (avec $\sigma = 0.2$ et $\kappa = 0.6$) sur notre exemple jouet. Il est à noter que l'ordre arbitraire $<_{\mathcal{CP}}$ justifie le fait de classer $A \succ C \,|\, D$ avant $A \succ D \,|\, C$ et $B \succ C \,|\, D$ comme le fait de classer $D \succ B \,|\, A$ avant $D \succ E \,|\, AC$.

	$\mathcal{MCP}_{0.2,0.6}$			Construction de Profil			
Cont. pref.	*supp*	*conf*	couverture	étape 1	étape 2	étape 3	étape 4
$D \succ E \mid A$	0.4	1	p_1,p_2	✔			
$D \succ C \mid \emptyset$	0.2	1	p_2	✗			
$A \succ C \mid D$	0.2	1	p_3		✔		
$A \succ D \mid C$	0.2	1	p_4			✔	
$B \succ C \mid D$	0.2	1	p_3		✗		
$D \succ B \mid A$	0.2	1	p_1	✗			
$D \succ E \mid B$	0.2	1	p_2	✗			
$D \succ E \mid AC$	0.2	1	p_1	✗			
$D \succ B \mid \emptyset$	0.4	2/3	p_1,p_5				✔
$D \succ E \mid \emptyset$	0.4	2/3	p_1,p_2	✗			

Table 3.3 – Règles de $\mathcal{MCP}_{0.2,0.6}$ ordonnées en fonction de $>_{best}$ et la construction de profil avec $(k = 1)$

3.3.2 L'algorithme ProfMiner

Etant donnée une base de préférences $\mathcal{P}$, un ensemble de règles de préférences contextuelles S, un seuil minimum d'accord k, ProfMiner retourne un profil utilisateur Π en sélectionnant les règles de préférences contextuelles pertinentes de S (cf. algorithme 3.3). Il est à noter que le seuil d'accord k nous permet d'ajuster la taille du profil utilisateur comme indiqué dans le problème 3. Plus le seuil minimum d'accord k est grand, plus le profil est petit.

Après avoir initialisé le profil (ligne 1), la principale itération (ligne 2-7) sélectionne la meilleure règle de préférence en se basant sur $>_{best}$ (ligne 3) et l'ajoute dans le profil (ligne 4) jusqu'à ce que S devienne vide (ligne 2). Cette condition est assurée par la réduction de $\mathcal{P}$ (ligne 5) et la réduction de S (ligne 6). En plus, une règle de préférence contextuelle π est considérée comme non pertinente (durant la construction du profil) si π n'est pas en accord avec au moins k préférences utilisateurs restantes (i.e., non encore en accord avec d'autres règles de préférences du profil). La partie droite de la table 3.3

Algorithm 3.3 ProfMiner

Entrées Une base de préférences $\mathcal{P}$, un ensemble de règles de préférences S, un seuil minimum d'accord k

Sortie Un profil utilisateur Π

1: $\Pi := \emptyset$
2: **Tant que** $S \neq \emptyset$ **Faire**
3: $\quad \pi_{best} = \max_{>_{best}} S$
4: $\quad \Pi := \Pi \cup \{\pi_{best}\}$
5: $\quad \mathcal{P} := \{\langle t, u\rangle \in \mathcal{P} | t \not\succ_{\pi_{best}} u\}$
6: $\quad S := \{\pi \in S | supp(\pi, \mathcal{P}) \geq k/|\mathcal{P}|\}$
7: **Fin Tant que**
8: **Retourner** Π

illustre PROFMINER sur notre exemple jouet (cf. table 3.2) avec $S = \mathcal{MCP}_{0.2,0.6}$ et $k = 1$. A la première itération, la ligne 3 sélectionne $D \succ E \mid A$ (symboles ✔) sur la meilleure règle en se basant sur $>_{best}$ (cf. table 3.3).
La ligne 5 supprime la préférence utilisateur p_1 et p_2 et ensuite, la ligne 6 supprime 5 règles de préférences contextuelles de S (symbole ✘). Il est à noter que $D \succ B \mid \emptyset$ est préservée puisqu'elle couvre aussi p_5.
La seconde itération ajoute $A \succ C \mid D$ au profil parce que c'est la meilleure règle de préférence contextuelle qui reste.
Ce processus s'arrête à la fin de la $4^{\text{ième}}$ itération parce que S est vide (cf. ligne 2 de PROFMINER).
A la fin de la quatrième itération, le profil final est $\{D \succ E \mid A, A \succ C \mid D, A \succ D \mid C, D \succ B \mid \emptyset\}$.

Dans la section 3.2, nous avons montré que CONTPREFMINER n'extrait pas toutes les règles de préférences contextuelles minimales mais toutes les règles de préférence ayant un contexte-préfixe minimal. Cependant, selon qu'on utilise les règles de préférences ayant un contexte-préfixe minimal ou les règles de préférences contextuelles minimales comme entrée de PROFMINER, le résultat reste le même.

Propriété 2 (Equivalence)**.** *Etant données une base de préférences $\mathcal{P}$, un seuil minimal σ, κ et k, le résultat de* PROFMINER *appliquée à S est égale à celle appliquée à $\mathcal{MCP}_{\sigma,\kappa}$ pour tout ensemble S tel que $\mathcal{MCP}_{\sigma,\kappa} \subseteq S \subseteq \mathcal{CP}_{\sigma,\kappa}$.*

Démonstration. Etant donnée une règle non-minimale $\pi \in S$, il existe une règle minimale $\pi_m \in S$ tel que $conf(\pi) = conf(\pi_m)$ et $supp(\pi) = supp(\pi_m)$ par définition. Comme le contexte de π_m est plus petit que celui de π, la règle minimale est toujours sélectionnée. Toutes les règles retournées par PROFMINER appartiennent à $\mathcal{MCP}_{\sigma,\kappa}$ et donc la propriété 2 est vraie. □

La propriété 2 signifie que l'utilisation des règles de préférences contextuelles intéressantes minimales ou les règles intéressantes avec un contexte préfixe minimal ou toutes les règles de préférences intéressantes, conduisent au même profil utilisateur. En plus, comme la relation $>_{best}$ classe une règle de préférence contextuelle minimale avant une règle non-minimale, la ligne 4 sélectionne une règle de préférence contextuelle minimale comme π_{best}. Alors, les lignes 5 et 6 écartent les règles de préférences contextuelles non-minimales correspondant à π_{best}.

Biensure, plus S est petit plus le temps d'exécution de PROFMINER est petit. Il sera alors mieux de considérer l'ensemble $\mathcal{MCP}_{\sigma,\kappa}$ comme entrée, mais les règles de préférences contextuelles minimales sont coûteux à calculer lors du parcours en profondeur. Nous

pensons que les règles ayant les contexte-prefixe minimaux sont le meilleure compromis puisqu'elles sont faciles à extraire et en même temps, elles sont moins nombreuses que $\mathcal{CP}_{\sigma,\kappa}$.

3.3.3 Conclusion

Dans cette section, nous avons présenté la solution que nous avons proposée pour la construction du profil utilisateur. Nous avons montré que cette solution consiste en une approche à deux phases qui consiste (1) à extraire toutes les règles de préférences contextuelles minimales et (2) à construire un profil utilisateur à partir des règles de préférences extraites au niveau de la première phase.
Dans la section suivante, nous allons présenter les stratégies que nous avons proposées pour prédire entre deux transactions (t, u) laquelle est la plus préférée.

3.4 Méthodes d'utilisation du profil

Pour prédire entre deux transactions (t, u) laquelle est la préférée, il est nécessaire d'utiliser les règles de préférences qui couvrent ces transactions dans le profil Π.
$\Pi_{\langle t,u\rangle}$ représente l'ensemble des règles de préférences contextuelles qui couvrent (t, u), i.e. $\Pi_{\langle t,u\rangle} = \{\pi \in \Pi \,|\, t \succ_\pi u \vee u \succ_\pi t\}$.
Quand une règle de préférence $\pi \in \Pi_{\langle t,u\rangle}$ induit que t est préférée à u, la confiance de π représente le degré de préférence de t sur u. Le poids d'une règle de préférence π pour évaluer une paire $\langle t, u\rangle$ est alors défini par $w_\pi(t, u) = conf(\pi, \mathcal{P})$ si $t \succ_\pi u$ ou $w_\pi(t, u) = 1 - conf(\pi, \mathcal{P})$ si $u \succ_\pi t$.
Dans ce contexte, le challenge consistant à classer les transactions est de sélectionner les bonnes règles dans $\Pi_{\langle t,u\rangle}$ et puis d'agréger leurs poids.

3.4.1 Best rule

Supposons que nous voulons utiliser le profil $\Pi = \{D \succ E \,|\, A, A \succ C \,|\, D, A \succ D \,|\, C, D \succ B \,|\, \emptyset\}$ pour prédire quelle est la transaction la préférée entre CDE et AC (il est à noter que ces transactions ne sont pas présentes sur les jeux de données initiales, cf. table 3.2).
Une façon naturelle de déterminer si une transaction t est préférée à une autre u est d'utiliser la *meilleure* règle de préférence contextuelle $\pi \in \Pi_{\langle t,u\rangle}$.
L'argument de la section précédente justifie encore le choix de l'ordre $>_{best}$ pour la définition de *la meilleure règle* (cf. Définition 39).

Exemple 50. *La meilleure règle de préférence contextuelle selon* $>_{best}$ *couvrant la paire* $\langle CDE, AC\rangle$ *est* $A \succ D \mid C$. *Cette règle induit que* AC *est préférée à* CDE *avec une confiance de* 1.

Plus généralement, d'après cette stratégie de classement, nous indiquons qu'une transaction $t \in \mathcal{L}$ est préférée à une transaction $u \in \mathcal{L}$ d'après un profil utilisateur Π, noté par $t \succ_{\Pi}^{br} u$, s'il existe une meilleure règle dans Π donnant $t \succ_b u$.
La figure 3.2 illustre la relation de préférence $\succ_{\Pi}^{br}$ induite par la meilleure règle sur les transactions de $\mathcal{D}$. En plus, chaque arc $\langle t, u\rangle$ est labélisé par la confiance de la meilleure règle π qui est en accord avec la préférence $\langle t, u\rangle$. Pour la suite, nous notons par br_{Π}, la fonction de préférence définie pour chaque transaction $(t, u) \in \mathcal{L} \times \mathcal{L}$ par :

$$br_{\Pi}(t,u) = \begin{cases} w_{\pi}(t,u) & \text{s'il existe une meilleure règle } \pi \text{ dans } \Pi_{\langle t,u\rangle} \\ 0.5 & \text{Sinon} \end{cases}$$

br_{Π} est une fonction de préférence [CSS99]. Une valeur de br_{Π} qui est proche de 1 est interprétée comme une forte recommandation que t doit être classée avant u en considérant le profil de l'utilisateur Π.

Exemple 51. $br_{\Pi}(AC, CDE) = 1$ *puisque la confiance de la règle de préférence contextuelle* $A \succ D \mid C$ *est égale à 1.*

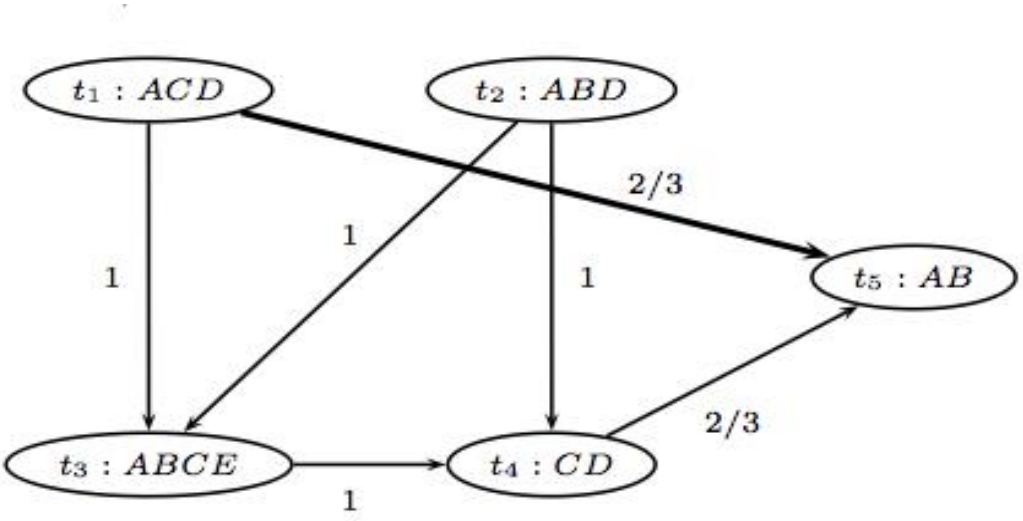


FIGURE 3.2 – La relation de préférence $\succ_{\Pi}^{br}$ induite par la meilleure règle

Avec cette stratégie, pour décider entre deux transactions (t,u) laquelle est la préférée, on utilise seulement la meilleure règle qui couvre la paire de transaction même s'il y en

à plusieurs autres qui couvrent la même transaction et peuvent conduire à reconsidérer la décision prise avec la meilleure règle.
Dans la section suivante, nous allons présenter la seconde stratégie de classement que nous avons proposée. Avec cette stratégie, on ne considère pas seulement la meilleure règle qui couvre une paire de transaction pour prendre une décision, mais on organise un vote sur les décisions prises par toutes les règles qui couvrent la paire de transaction.

3.4.2 Weighted voting

Il peut sembler contreproductif de considérer seulement la meilleure règle quand le profil peut contenir 10 autres règles dont la conclusion est contraire.
En se basant sur cette observation, nous proposons une seconde alternative qui consiste à utiliser une stratégie nommée "*Weighted Voting*" comme proposé dans [LHP01].
En comparant deux transactions, l'idée est de ne pas utiliser seulement la meilleure règle qui couvre cette paire, mais l'ensemble des règles qui la couvre (i.e., $\Pi_{\langle t,u\rangle}$).
Evidemment, chaque règle est pondérée par la confiance dans la recommandation finale en utilisant w_π. Dans certains cas, plusieurs règles minoritaires peuvent permettre de reconsidérer la recommandation de la meilleure règle. Nous définissons la fonction de préférence wv_Π comme :

$$wv_\Pi(t,u) = \begin{cases} \frac{1}{|\Pi_{\langle t,u\rangle}|} \sum_{\pi \in \Pi_{\langle t,u\rangle}} w_\pi(t,u) & \text{si } \Pi_{\langle t,u\rangle} \neq \emptyset \\ 0.5 & \text{sinon} \end{cases}$$

Finalement, en utilisant la fonction de préférence wv_Π, nous pouvons définir la relation de préférence $\succ_\Pi^{wv}$ comme suit : $t \succ_\Pi^{wv} u$ ssi $wv_\Pi(t,u) > 0.5$.
A cause de la simplicité de notre exemple jouet, la relation de préférence $\succ_\Pi^{wv}$ est exactement la même que $\succ_\Pi^{br}$.
Le but principal d'un profil est d'ordonner des transactions, pour cette raison, le premier désavantage majeur de $\succ_\Pi^{br}$ et de $\succ_\Pi^{wv}$ est la possibilité d'*inconsistence*.

Exemple 52. *Par exemple, la relation $\succ_\Pi^{br}$ basée sur le profil $\{A \succ B \,|\, \emptyset, C \succ D \,|\, \emptyset, D \succ E \,|\, \emptyset\}$ indique que $AE \succ_\Pi^{br} BC$ et $BC \succ_\Pi^{br} D$, mais $D \succ_\Pi^{br} AE$.*

Cette inconsistance rend difficile l'ordre sur les transactions AE, BC et D. Le second plus grand inconvénient de $\succ_\Pi^{br}$ et $\succ_\Pi^{wv}$ est leurs *éparcité* i.e., plusieurs transactions ne sont pas comparables.

Exemple 53. *Par exemple, la relation $\succ_\Pi^{br}$ basée sur le profil $\{D \succ E \,|\, A, A \succ C \,|\, D, A \succ D \,|\, C, D \succ B \,|\, \emptyset\}$ ne peut pas décider si la transaction AC est préférée à BCE ou vice versa.*

Par conséquent sur la section suivante, nous introduisons une nouvelle stratégie de classement, appelée range voting.

3.4.3 Range voting

Indépendamment de notre méthode de construction du profil et des méthodes d'ordonnancement, réparer et compléter une relation de préférence pour faire face à l'inconsistance et l'éparsité est un défi [MFJ+06].
Au lieu de comparer directement deux transactions grâce à la relation $\succ_{\Pi}^{br}$ ou $\succ_{\Pi}^{wv}$, nous proposons d'organiser un *range voting* pour prendre une décision " qui est le meilleur candidat ?".
Un range voting est un système de vote pour une élection à un siège pour lequel les votants dans T notent chaque candidat avec un nombre entre 0 et 1 en utilisant une fonction de préférence comme br_{Π}ou wv_{Π}.
Les scores de chaque candidat sont cumulés, et le candidat ayant la valeur la plus élevée est le gagnant. Considérons deux transactions ACD et CD comme candidats, la transaction $t_3 : ABCE$ donne simplement un point à ACD (parce que $br_{\Pi}(ACD, ABCE) = 1$ et $br_{\Pi}(CD, ABCE) = 0$) ; $t_5 : AB$ donne $2/3$ à chaque candidat ACD et AD ; et ainsi de suite.
Avec ce vote ACD reçoit plus de votes et ensuite, est préférée à CD. L'ordre induit par le range voting formalise ce processus :

Définition 40 (Range voting order)**.** *Etant donnée une fonction de préférence $pref_{\Pi}$, l'ordre induit par le range voting pour $T \subseteq \mathcal{L}$ est défini comme :*

$$t \succ_{\Pi}^{T,pref} u \Leftrightarrow \sum_{v \in T} pref_{\Pi}(t, v) > \sum_{v \in T} pref_{\Pi}(u, v)$$

Maintenant, cette nouvelle relation de préférences nous permet d'ordonner ACD avant CD parce que $\sum_{v \in T} br_{\Pi}(ACD, v) = 0.5 + 0.5 + 1 + 0.5 + 2/3 = 19/6$ est plus grand que $\sum_{v \in T} br_{\Pi}(CD, v) = 0.5 + 0 + 0 + 0.5 + 2/3 = 10/6$ avec $T = \mathcal{D}$.
La figure 3.3 représente l'ordre $\succ_{\Pi}^{\mathcal{D},br}$ induit par le range voting sur l'exemple jouet $\mathcal{D}$. La première observation importante est que l'ordre induit par le range voting $\succ_{\Pi}^{T,br}$ conduit à comparer plusieurs transactions (voir les lignes gras) comme désiré pour faire face à l'éparcité.
En effet, comme le range voting satisfait le critère de resolvabilité [Woo94] qui assure une faible possibilité d'indécision dans le vote,$\succ_{\Pi}^{T,br}$ permet de classer plus de transactions que $\succ_{\Pi}^{br}$.
De plus, même si le range voting reste consistant avec plusieurs préférences utilisateurs

et la fonction de préférence initiale $pref_{\Pi}$, il existe une préférence inversée entre AB et CD (voir les lignes en pointillé). Sans cette inversion la relation ne serait pas transitive. Il est à noter que les transactions de $\mathcal{D}$ sont utilisées comme des votants par la fonction

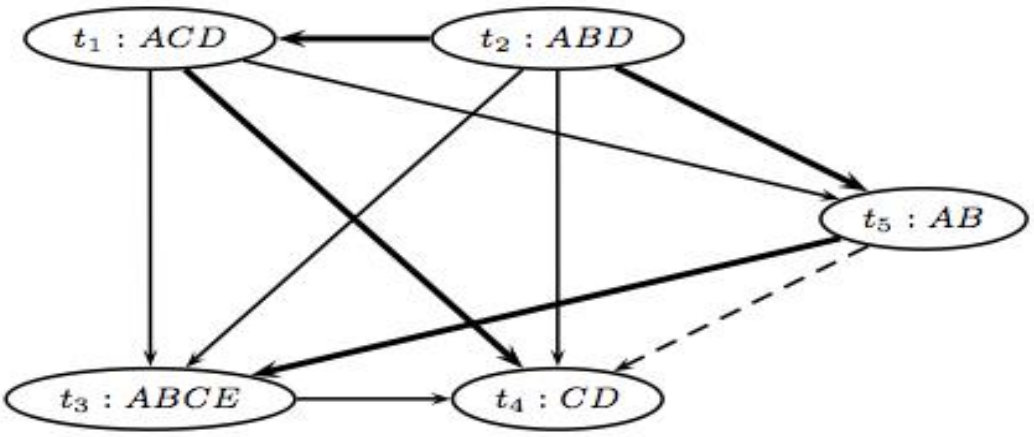


FIGURE 3.3 – L'ordre induit par la strategie du Range Voting $\succ_{\Pi}^{T,br}$

de range voting pour ordonner $\mathcal{D}$ dans la figure 3.3.
Pour ce qui suit, quand nous avons à ordonner des transactions de T, nous utilisons le même ensemble T comme votant (e.g., dans nos expérimentations, les données test sont utilisées comme des votants). En plus, cet ensemble de votants T garantit que l'ordre induit par $\succ_{\Pi}^{T,pref}$ peut être total.
Nous présentons maintenant le résultat principal sur les relations de préférences $\succ_{\Pi}^{T,pref}$. En plus, même si la fonction de préférence initiale n'induit pas un ordre, la relation de préférence correspondante $\succ_{\Pi}^{T,pref}$ est transitive et par conséquent, est un ordre :

Propriété 3 (Ordre partiel strict)**.** *La relation* $\succ_{\Pi}^{T,pref}$ *est un ordre partiel sur* $\mathcal{L}$*.*

Démonstration. **Relation irreflexive :** Pour chaque transaction t, on a $\sum_{v\in T} pref_{\Pi}(t,v) \not> \sum_{v\in T} pref_{\Pi}(t,v)$. **Relation Transitive :** étant données trois transactions s, t et u telles que $s \succ_{\Pi}^{T,pref} t$ et $t \succ_{\Pi}^{T,pref} u$, la définition 40 donne $\sum_{v\in T} pref_{\Pi}(s,v) > \sum_{v\in T} pref_{\Pi}(t,v) > \sum_{v\in T} pref_{\Pi}(u,v)$. Par conséquent, nous avons $\sum_{v\in T} pref_{\Pi}(s,v) > \sum_{v\in T} pref_{\Pi}(u,v)$ signifiant que $s \succ_{\Pi}^{T,pref} u$. □

Cette propriété cruciale explique pourquoi le graphe produit par la figure 3.3 correspond à un ordre partiel strict sur les transactions.

3.4.4 Relation entre la fonction de coût et les mesures de précision et rappel

Le but de cette section est de relier notre méthode de construction de profil basée sur la fonction de coût (cf. définition 37) avec les mesures de précision et rappel utilisées pour évaluer les méthodes de classifications.
Dans les sous-sections précédentes nous avons proposé trois méthodes de classement qui peuvent être associés à un ensemble de règles de préférences Π (cf. sous-sections 3.4.1, 3.4.2, 3.4.3). La qualité prédictive d'une relation de préférence est évaluée par les mesures de précision et de rappel introduites dans la Définition 41 ci-dessous. Ces deux mesures seront utilisées dans le chapitre 4 pour comparer la qualité de prédiction de nos méthodes de classement avec celle de SVMRANK.

Définition 41 (Précision et rappel)**.** *Etant donnée une base de préférence $\mathcal{P}$ et une relation de préférence $\succ$ sur les transactions apparaissant dans $\mathcal{P}$. La précision de $\succ$ en considérant $\mathcal{P}$ noté $Prec(\succ, \mathcal{P})$, est définie par :*

$$Prec(\succ, \mathcal{P}) = \frac{|\{\langle t, u\rangle \in \mathcal{P} | t \succ u\}|}{|\{\langle t, u\rangle \in \mathcal{P} | t \succ u \vee u \succ t\}|}$$

En plus, le rappel de $\succ$ en consideration $\mathcal{P}$, noté $Rec(\succ, \mathcal{P})$, est definie par :

$$Rec(\succ, \mathcal{P}) = \frac{|\{\langle t, u\rangle \in \mathcal{P} | t \succ u\}|}{|\mathcal{P}|}$$

Il est à noter que les définitions de la précision et du rappel, ne sont pas différentes de celles présentées dans la sous-section 2.1.4 : il suffit de noter que $|\{\langle t, u\rangle \in \mathcal{P} | t \succ u\}| = T$, $|\{\langle t, u\rangle \in \mathcal{P} | t \succ u \vee u \succ t\}| = T + F$ et $|\mathcal{P}| = T + F + I$.
Dans notre approche, les relations de préférences décrites dans les sous-sections précédentes (cf. sous-sections 3.4.1, 3.4.2,3.4.3) et le profil ne sont pas des notions différentes. La définition suivante tente d'établir ce lien :

Définition 42 (Compatibilité)**.** *Une relation de préférence $\succ$ est compatible avec un ensemble de règles de préférences Π ssi pour chaque paire $\langle t, u\rangle \in \mathcal{P}$, $t \not\succ u$ implique qu'il existe $\pi \in \Pi$ tel que $u \succ_\pi t$ ou qu'il n'existe pas des règles $\pi \in \Pi$ telle que $t \succ_\pi u$.*

Cette propriété existe pour les stratégies best rule et weighted voting mais pas pour la stratégie de range voting. Maintenant nous donnons un résultat important montrant pourquoi le profil utilisateur conduit à un bonne méthode de ranking :

Proposition 1. *Soit Π un ensemble de règles de préférences, $\succ$ une relation de préférence*

compatible avec Π. *La relation suivante existe :*

$$1 - Prec(\succ, \mathcal{P}) \leq 1 - Rec(\succ, \mathcal{P}) \leq Cost(\Pi, \mathcal{P})$$

Démonstration. Montrons que $1 - Rec(\succ, \mathcal{P}) \leq Cost(\Pi, \mathcal{P})$. Considérons les deux transactions t et u tel que $t \not\succ u$. Comme $\succ$ est compatible avec Π, cela signifie que (i) il existe $\pi \in \Pi$ tel que $u \succ_\pi t$ ou (ii) qu'il n'existe pas de règles $\pi \in \Pi$ tel que $t \succ_\pi u$. Dans le cas de (i), $\langle t, u\rangle$ appartient à $contradict(\Pi, \mathcal{P})$ et dans le cas (ii), $\langle t, u\rangle$ appartient à $\mathcal{P} \setminus cover(\Pi, \mathcal{P})$. Nous obtenons que $\{\langle t, u\rangle \in \mathcal{P} | t \not\succ u\}$ est inclus dans $(\mathcal{P} \setminus cover(\Pi, \mathcal{P})) \cup contradict(\Pi, \mathcal{P})$. Nous allons alors conclure que $1 - Rec(\succ, \mathcal{P}) \leq Cost(\Pi, \mathcal{P})$. Comme par definition $Prec(\succ, \mathcal{P}) \geq Rec(\succ, \mathcal{P})$, il est facile de conclure que $1 - Prec(\succ, \mathcal{P}) \leq 1 - Rec(\succ, \mathcal{P})$. □

Cette proposition signifie que minimiser la fonction de coût conduit à maximiser à la fois la précision et le rappel. Par conséquent, l'objectif de la construction de profil est consistant avec la prédiction pour une relation de préférence compatible.

3.4.5 Conclusion

Dans cette section, nous avons présenté les stratégies d'utilisation du profil construit afin de faire de la recommandation. Nous avons présenté la méthode *Best-rule* qui consiste à choisir la meilleure règle dans le profil pour décider parmi deux transactions (t,u) la quelle est la préférée.
Nous avons également présenté une seconde stratégie d'utilisation du profil qui n'utilise pas seulement la meilleure règle qui couvre le profil pour trancher entre deux transactions laquelle est la préférée pour l'utilisateur, mais organise un vote parmi toutes les règles de préférences qui couvrent une paire de transactions.
La troisième stratégie consiste à organiser un *range voting* pour décider quelle transaction est la plus préférée.

3.5 Conclusion

Dans ce chapitre nous avons présenté le problème de construction de profil que nous avons résolu. Nous avons commencé par formaliser le problème que nous comptons résoudre (section 3.1). Nous avons montré que nous comptons résoudre ce problème en utilisant une approche à deux phases qui consiste à :

(1) Extraire toutes les règles de préférences contextuelles intéressantes et minimales $\mathcal{MCP}_{\sigma,\kappa}$ (cf. section 3.2)

(2) Eliminer les règles de préférences contextuelles redondantes au niveau de la première phase afin de construire un profil utilisateur (cf. section 3.3)

Nous avons également présenté les trois stratégies d'utilisation du profil que nous avons utilisées pour faire une prédiction. Nous avons présenté la méthode Best-rule inspirée de CBA (cf. sous-section 3.4.1) puis la méthode Weighted Voting inspirée de CMAR (cf. sous-section 3.4.2) et enfin nous avons présenté la méthode du Range voting (cf. sous-section 3.4.3).
Dans le chapitre suivant, nous allons présenter les expérimentations que nous avons effectuées pour valider notre approche.

Chapitre 4

Expérimentation

Dans ce chapitre nous présentons les expérimentations que nous avons effectuées dans le but d'évaluer les performances de notre approche. Le prototype avec lequel nous avons effectué les expérimentations est développé en C++. Les expérimentations sont conduites sur un PC tournant sous Linux, avec 32 GB de mémoire et 16 processeurs 4 cores de type Genuine Intel(R) Xeon(R) ayant chacun une vitesse de 2.40 GHz.
Après la présentation des jeux de données dans la section 4.1, nous décrivons différents types d'expérimentations.

Dans la section 4.2, nous évaluons la performance de ContPrefMiner, l'algorithme proposé pour extraire les règles de préférences contextuelles intéressantes. Plus précisément, nous présentons à la fois le nombre de règles de préférences contextuelles extraites et le temps d'extraction en considérant un seuil minimum de support et un seuil minimum de confiance. Nous évaluons également comment le temps d'extraction varie quand la taille de la base de préférences augmente.

Dans la section 4.3, nous présentons une étude qualitative du profil utilisateur construit par ProfMiner. En particulier, nous analysons la concision des profils utilisateurs en se basant sur le seuil d'accord minimal. Nous donnons aussi quelques exemples de profils extraits, et nous étudions la distribution des règles de préférences dans le profil utilisateur par rapport à la taille des contextes.

Dans la section 4.4, nous évaluons la qualité de prédiction du profil, en terme de précision, rappel et de f-mesure, en utilisant différentes stratégies de classements (*Best Rule*, *Weighted Voting* et *Range Voting*). Nous comparons aussi, dans cette section, la qualité prédictive de notre approche avec deux méthodes de l'état de l'art : à savoir les méthodes SVMRank [Joa02] et Order-by-Pref [CSS99]. Finalement, dans la section 4.5, nous présentons la conclusion de ce chapitre.

User Id.	$\lvert\mathcal{D}\rvert$	$\lvert\mathcal{I}\rvert$	$\lvert\mathcal{P}\rvert$
user850	504	4 646	90 740
user5555	505	4 074	97 979
user1701	509	4 515	87 919
user5317	513	4 477	89 989
user438	520	4 193	98 762
user48	541	4 232	93 365
user1125	544	4 391	110 026
user1496	551	4 622	109 650
user4867	552	4 640	97 860
user4411	553	4 626	111 208
user3942	559	4 569	119 600
user5107	564	4 843	116 700
user533	565	4 468	122 757
user3884	571	4 893	124 974
user148	582	4 591	120 291
user4054	583	4 854	117 342
user123	584	4 491	111 993
user5682	588	4 492	127 827
user692	591	4 611	111 754
user411	597	4 939	129 933

TABLE 4.1 – Base de preferences sur les films

4.1 Présentation des données

Dans cette section, nous présentons le schéma de la base de données sur laquelle nous avons effectué nos expérimentations. Le modèle de la base de données est représenté sur la figure 4.1(a). Cette base de données est composée de trois tables contenant des informations sur des films (acteur, producteur, etc..), les personnes qui ont regardé ces films et les notes que ces personnes ont attribué à ces films.

- La table *Film* : donne la liste de tous les films de la base décrits par leur titre, genre etc...
- La table *Personne* : donne la liste des personnes qui ont notées des films. Ces personnes sont décrites par leur identifiant, leur genre, leur age ...
- La table *Notes* : donne les notes fournies par les utilisateurs aux films. Cette table contient l'identifiant de l'utilisateur, l'identifiant du film noté, la note attribuée ainsi que l'horodatation.

La table 4.2 montre le nombre d'enregistrements que contient chaque table, tandis que la figure 4.1(b) montre la distribution des utilisateurs par rapport au nombre de notes. On y constate par exemple que 3095 utilisateurs ont noté entre 1 et 100 films. 140 utilisateurs ont noté entre 500 et 600 films ...

table	Nombre d'enregistrements
Films	3881
Utilisateur	6040
Notes	800156

Table 4.2 – Taille des enregistrements

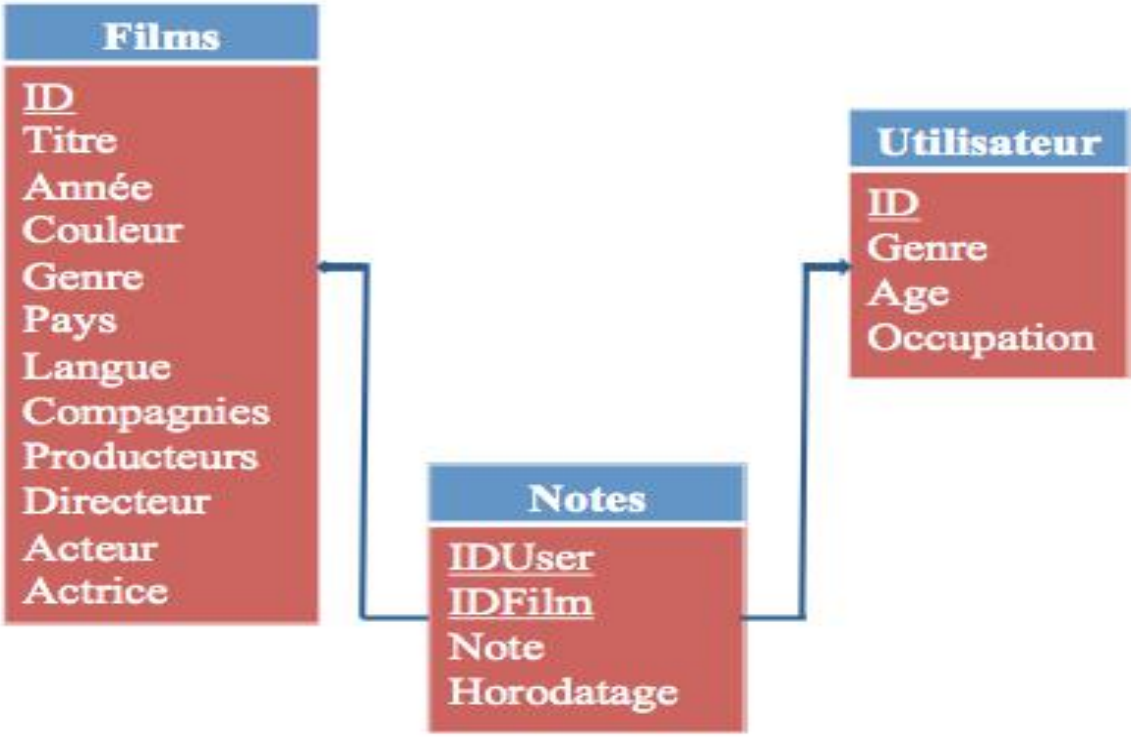
Les données contenues dans la base de données sont des données réelles basées sur APMD-Workbench [PKB09] construit automatiquement de MovieLens [1] et IMDB [2].

Ces jeux de données sont constitués de 800156 notes (comprises entre 1 et 5), de 6040 utilisateurs et de 3881 films (cf. table 4.2). Chaque utilisateur ayant noté au moins 20 films, et chaque films est décrit par un ensemble d'attributs tel que le genre, le directeur, l'acteur, etc.
Dans le but d'évaluer notre approche, nous extrayons de la base de données initiale 20 jeux de données de films notés par 20 utilisateurs différents. Ces utilisateurs sont choisis par hasard parmi les 140 utilisateurs qui ont noté entre 500 et 600 films. Il est à noter que ces jeux de données n'ont pas été directement utilisables par notre algorithme, il a fallu effectuer un pré-traitement [3] pour avoir des données que notre algorithme peut manipuler.

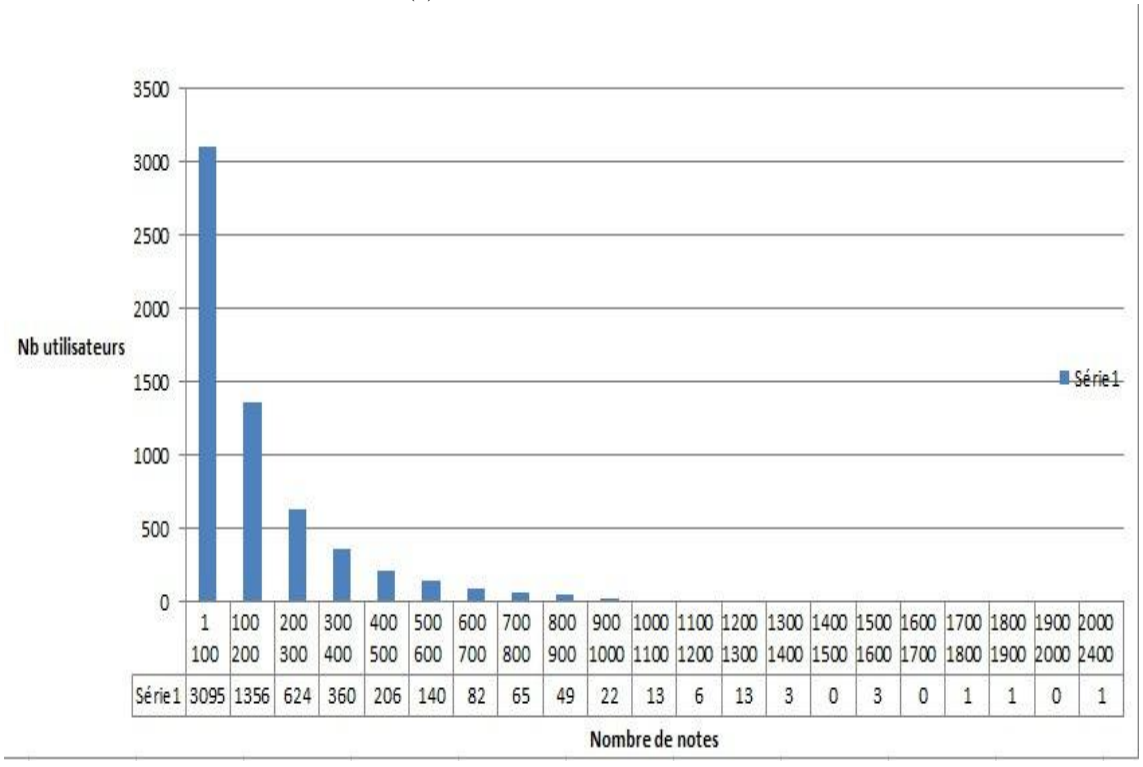
Dans le but de comparer notre approche avec la méthode de référence SVM-Rank [Joa02], nous utilisons le même format [4], i.e. chaque ligne des 20 jeux de données contient d'abord une note d'un film, et la liste des paires d'attributs/valeurs représentant les caractéristiques des films notés (genre, directeur, année, acteur, ...).

Dans la table 4.1, on peut voir les principales caractéristiques de chaque jeu de données : le nombre de films notés par chaque utilisateur $|\mathcal{D}|$, le nombre d'attributs distincts $|\mathcal{I}|$ utilisés pour décrire les films notés et le nombre de paires de films comparables $|\mathcal{P}|$ (deux films sont dits *comparables* si leurs notes associées sont différentes).

1. www.movielens.org
2. www.imdb.com
3. cf. Annexe C pour le détail du pré-traitement
4. cf. Annexe C pour le format des données utilisées



(a) Modèle de la base de données



(b) Nombre d'utilisateurs en fonction des notes

FIGURE 4.1 – Présentation de la base de données

Dans notre approche et nos expérimentations, il est important de noter que l'algorithme ContPrefMiner est exécuté sur un ensemble de paires d'objets comparables et non sur un ensemble d'objets notés.

4.2 Extraction de règles de préférences

Dans cette section, nous étudions le comportement de l'algorithme, ContPrefMiner, proposé pour extraire les règles de préférences contextuelles. Nous rappelons que la phase d'extraction des règles de préférences constitue la première phase de notre approche à deux phases.
Nous étudions le comportement de l'algorithme proposé en terme de quantité de règles extraites pour différents seuil de supports et de confiances (cf. sous-section 4.2.1) ; nous présentons également le comportement de l'algorithme et en terme de temps d'exécution en fonction des support et confiance et en fonction de la taille de la base de préférences(cf. sous-section 4.2.2)

4.2.1 Quantité des règles

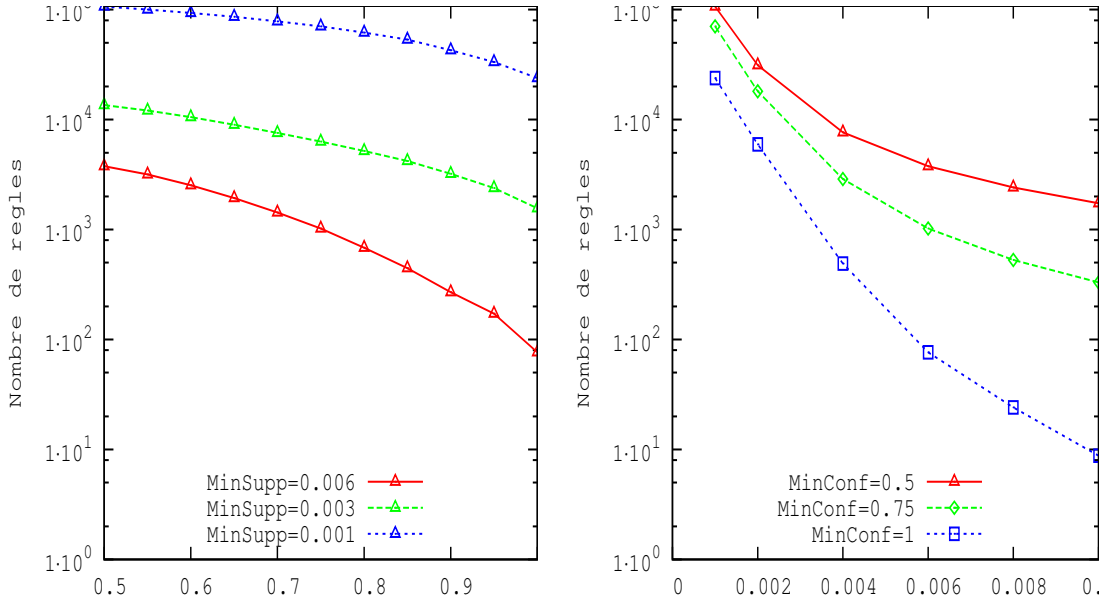
Dans cette sous-section, nous présentons le nombre de règles de préférences extraites en fonction de la variation du support et de la confiance.
Les figures 4.2 (a) et 4.2 (b) montrent le nombre moyen de règles de préférences contextuelles extraites par ContPrefMiner sur les 20 jeux de données. Cette expérimentation est effectuée en augmentant les seuils minimum de support et de confiance.

Comme les seuils minimum de support et de confiance augmentent, nous pouvons voir que le nombre de règles de préférences contextuelles extraites diminue exponentiellement. Ce phénomène est naturellement comparable au phénomène obtenu quand on extrait n'importe quel type de motifs locaux intéressants.

4.2.2 Temps d'execution

Dans cette sous-section, nous présentons le temps d'exécution de ContPrefMiner en fonction de la variation du support et de la confiance.

La figure 4.3 montre le temps d'exécution de ContPrefMiner quand (a) le seuil minimum de support varie de $0,1\%$ à 1% (pour différentes valeures de confiance fixées) et quand (b) le seuil de confiance varie de 50% à 100% (pour différentes valeurs de support fixées).
Nous voyons d'abord que le temps d'exécution est quasiment indépendant de la valeur



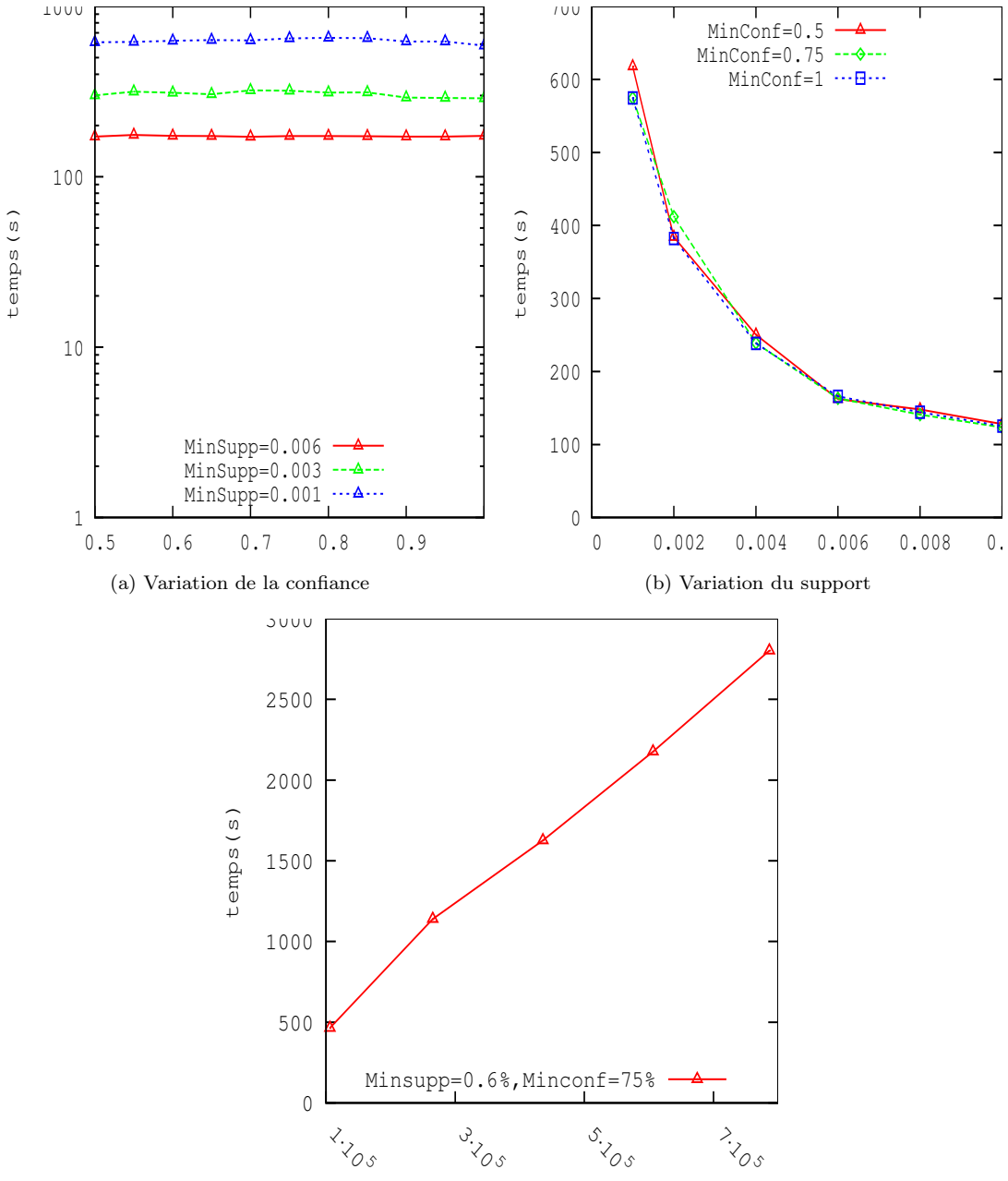
(a) Variation de la confiance

(b) Variation du support

FIGURE 4.2 – Performances de CONTPREFMINER en fonction de :

de confiance minimum, ce qui signifie que l'ensemble des règles de préférences exploré n'est pas élagué par la contrainte de confiance .
D'autre part, nous pouvons noter que l'effet du support est important. En effet, le temps d'exécution diminue exponentiellement avec le support minimum, ce qui montre que CONTPREFMINER exploite effectivement la contrainte du support pour élaguer l'ensemble des règles de préférences explorées.
Finalement, la figure 4.3(c) montre comment le temps d'exécution varie quand la taille de la base de préférence augmente (pour des seuils minimum de support et de confiance fixés).

Dans cette expérimentation, nous avons utilisé les jeux de données d'un utilisateur qui a noté plus de 1000 films. Plus précisément, la base de préférence de cet utilisateur contient plus de 700000 préférences utilisateurs, et nous avons évalué le temps d'exécution de CONTPREFMINER pour différentes fractions de cette base de préférences.
La figure 4.3(c) montre que le temps d'exécution augmente linéairement avec la taille de la base de préférences ; elle montre aussi que CONTPREFMINER peut extraire toutes les règles de préférences intéressantes d'une base contenant plus de 700000 préférences en moins d'une heure.



(a) Variation de la confiance

(b) Variation du support

(c) Taille de la base de préférence

Figure 4.3 – Pourcentage des règles en fonction de la taille des contextes

4.3 Construction de profil utilisateur

Dans cette section, nous présentons les expérimentations que nous avons effectuées pour évaluer la qualité du profil construit. Ces expérimentations vont montrer que le profil construit peut être aussi concis que désiré par l'utilisateur (sous-section 4.3.1) et qu'il est facilement interprétable (sous-section 4.3.2)

4.3.1 Concision du profil

Dans cette sous-section, nous analysons la concision du profil utilisateur en se basant sur le seuil minimum d'accord.

La figure 4.4 affiche le nombre moyen des règles de préférences contextuelles extraites (des 20 jeux de données sur les utilisateurs) quand le seuil minimum d'accord k varie de 1 à 3500. D'abord, il est important de noter que même avec $k = 1$, le nombre moyen de règles de préférences contextuelles dans le profil utilisateur est réduit drastiquement par ProfMiner. En effet, pour $k = 1$, l'algorithme ProfMiner réduit le nombre de règles de préférences extraites par ContPrefMiner de 490 à $203,9$.

La figure 4.4 montre que la taille du profil utilisateur décroît rapidement avec l'augmentation de k. Il est à noter que si k est plus grand que 500, la taille moyenne des profils utilisateur est plus petit que 50. De plus, cette figure montre que le profil utilisateur peut être aussi concis que désiré par l'utilisateur, ce qui est très important pour présenter un profil compréhensible.

4.3.2 Intelligibilité du profil

Dans cette sous-section, nous montrons que notre profil est facilement interprétable par l'utilisateur. Nous montrons également comment les règles de préférences se repartissent par rapport à la taille des contextes.
Les tables 4.3 et 4.4 montrent les 10 meilleures règles de préférences obtenues à partir des bases de préférences *User48* et *User1125*. Ces règles de préférences sont obtenues en utilisant ProfMiner avec un seuil d'accord $k = 1$.

La table 4.3 montre que l'utilisateur *User48* n'aime pas les films de sports (cf. règles $1, 2, 3, 6, 7, 9$). Les règles 4 et 8 montrent que l'utilisateur préfère les films de *20th Century Fox* que les films des compagnies *Paramount Pictures*ou *Metro Goldwyn Mayer (MGM)*. La règle 5 montre qu'entre deux films de comédies, l'utilisateur *User48* préfère les films de la compagnie *TriStar Pictures* à ceux de la compagnie *Walt Disney Pictures*.

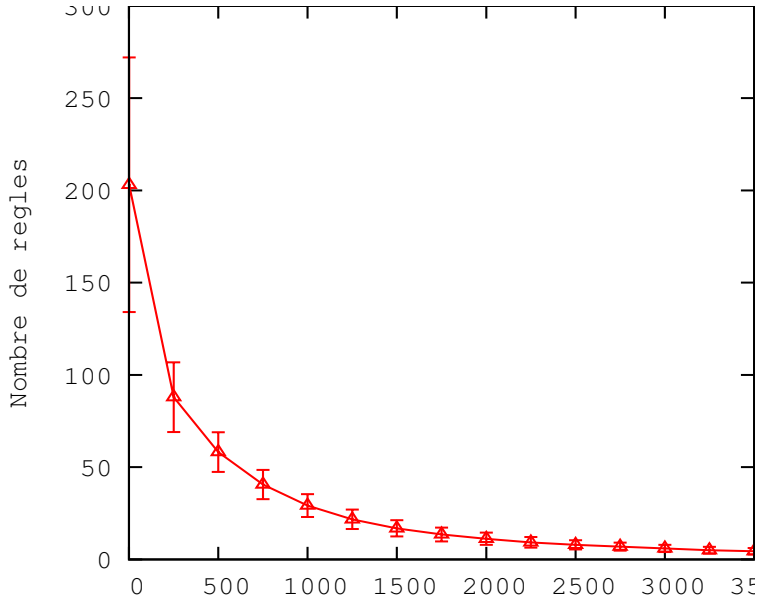


Figure 4.4 – Nombre de règles de préférences en fonction de k

La table 4.3 montre que l'utilisateur *User1125* préfère les films de *Guerre* aux films de *Thriller* (cf. règles $3, 4, 7$). Nous pouvons aussi voir qu'entre deux films de comédies, il/elle préfère le film de la compagnie *TriStar Pictures* à ceux de la compagnie *Warner Bros Pictures*. Finalement, la règle 5 montre que *User1125* préfère inconditionnellement les films de la compagnie *Lucasfilm Ltd.* à ceux de la compagnie *Touchstone Pictures*.

Les figures 4.5(a), (b) et (c) montrent la distribution des règles de préférence sur un profil utilisateur en se basant sur la taille de leur contextes (avec des valeurs différentes de k).
D'abord, nous pouvons voir figure 4.5(c) que pour tous les utilisateurs, la taille des contextes des règles de préférence est toujours entre 0 et 4. Plus précisément, la figure 4.5(a) montre que dans le profil utilisateur *User48*, les règles les plus nombreuses sont celles ayant la taille de contexte égale à 1, i.e avec $k = 1$, $50,47\%$ des règles de préférence contextuelles ont un contexte de taille égale à 1.

Dans le profil de l'utilisateur *User1125*, la taille des contextes est même plus importante. En effet, avec $k = 1500$, $59,42\%$ des règles de préférences ont une taille des contexte égale à 2. Par conséquent, ces histogrammes montrent l'importance de l'usage des contextes dans le but d'obtenir des profils qui sont consistants et concis (cf. section 4.4). Ces histogrammes montrent aussi que plus la valeur de k est grande, plus la taille des contextes augmente. Ce phénomène peut être expliqué facilement par le rôle du paramètre k dans ProfMiner. En effet, quand k augmente, ProfMiner extrait des règles de préférence contextuelles moins redondantes, ce qui signifie qu'il insère dans le profil

Contextual preference rule	*supp*	**conf**
1.GEN :War ≻ GEN :Sport | GEN :Drama COU :United States	0.41%	100%
2.GEN :History ≻ GEN :Sport | GEN :Drama	0.22%	100%
3.GEN :War ≻ GEN :Sport | YEA :1990s	0.18%	100%
4.COM :20th Century Fox ≻ COM :Paramount Pictures | GEN :Sci-Fi	0.13%	100%
5.COM :TriStar Pictures ≻ COM :Walt Disney Pictures | GEN :Comedy	0.12%	100%
6.GEN :Musical ≻ GEN :Sport | GEN :Family	0.12%	100%
7.GEN :History ≻ GEN :Sport | YEA :1990s	0.12%	100%
8.COM :20th Century Fox ≻ COM :Metro-Goldwyn-Mayer (MGM) | GEN :Drama	0.12%	100%
9.GEN :Western ≻ GEN :Sport | GEN :Drama	0.11%	100%
10.COM :Warner Bros. Pictures ≻ COM :Chartoff-Winkler Prod. | GEN :Drama	0.13%	99.21%

TABLE 4.3 – 10 meilleures règles extraites de la base User48 ($k = 1$).

Contextual preference rule	*supp*	*conf*
1.COU :Germany ≻ COU :United States | ∅	0.37%	100%
2.COM :Jersey Films ≻ COM :20th Century Fox | YEA :1990s	0.11%	92.86%
3.GEN :War ≻ GEN :Thriller | GEN :Action GEN :Drama YEA :1990s LAN :English	0.10%	92.62%
4.GEN :War ≻ GEN :Thriller | GEN :Action GEN :Drama YEA :1990s	0.11%	90.77%
5.COM :Lucasfilm Ltd. ≻ COM :Touchstone Pictures | ∅	0.10%	89.6%
6.COM :TriStar Pictures ≻ COM :Warner Bros. Pictures | GEN :Comedy	0.11%	89.55%
7.GEN :War ≻ GEN :Thriller | GEN :Action GEN :Drama LAN :English	0.15%	89.53%
8.COM :Jersey Films ≻ COM :Paramount Pictures | YEA :1990s	0.11%	89.21%
9.COM :Jersey Films ≻ COM :20th Century Fox | ∅	0.15%	89.13%
10.COM :Jersey Films ≻ COM :Universal Pictures | ∅	0.10%	88.37%

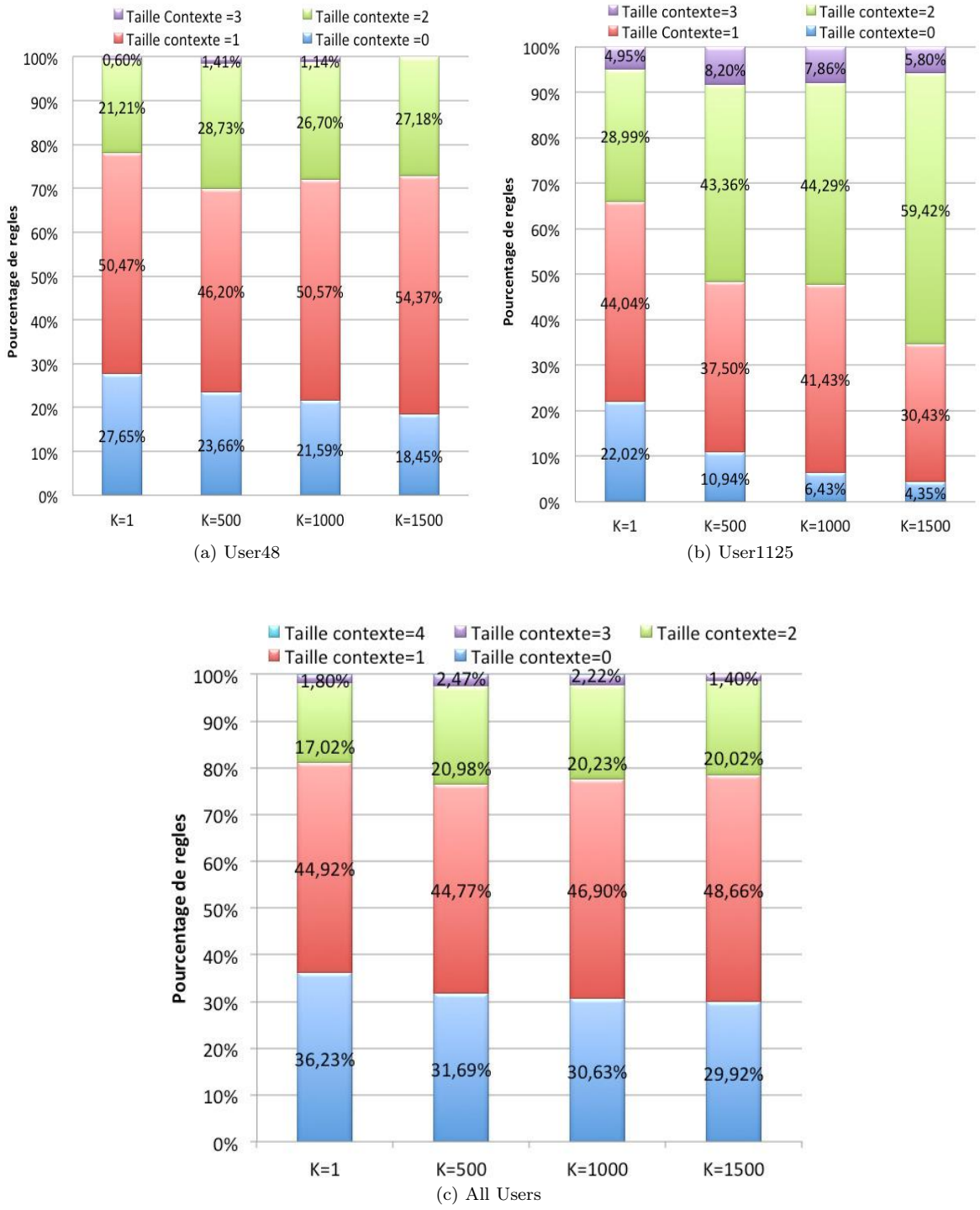
TABLE 4.4 – 10 meilleures règles extraites de la base User1125 ($k = 1$).

utilisateur des règles de préférence plus spécifiques (des règles avec des contextes plus larges).

4.4 Qualité de prédiction du profil

Dans le but d'évaluer la qualité de prédiction du profil utilisateur, nous avons effectué une validation croisée *stratifiée* de 5-blocs. En considérant une base de données $\mathcal{D}$ i.e. une liste de films notés avec leurs attributs, le principe de la validation croisée *stratifiée* de 5-blocs est le suivant :

1. Partitionner les jeux de données $\mathcal{D}$ en $K = 5$ sous ensembles disjoints $\mathcal{D}_1, ..., \mathcal{D}_K$ qui sont approximativement de même taille et stratifiés en considérant les notes (chaque sous ensemble D_k garde la même proportion de films notés i, $i = 1, ..., 5$).
2. Pour chaque $k = 1, ..., K$, soit $\mathcal{P}_k$ l'ensemble des préférences utilisateurs $\langle m_1, m_2 \rangle$ où m_1 et m_2 sont les films notés dans $\mathcal{D}_k$ tel que la note fournie à m_1 est plus grande que la note fournie à m_2.
3. Exécuter K itérations d'apprentissage et de test, tel que lors de chaque itération k, la base de préférences fournie pour l'apprentissage est : $\mathcal{P}_1 \cup ... \cup \mathcal{P}_{k-1} \cup \mathcal{P}_{k+1} \cup ... \cup \mathcal{P}_K$ et la base de données test est $\mathcal{P}_k$.



(a) User48 (b) User1125

(c) All Users

Figure 4.5 – Pourcentage de règles en fonction de la taille des contextes.

Ainsi, il est important de noter qu'à chaque itération, les préférences utilisateur impliquées dans la base test ne contiennent aucun film impliqué dans la base d'apprentissage. En plus, il est à noter que dans nos expérimentations, ce sont les valeurs moyennes des métriques (e.g., précision, rappel et f-mesure) sur les 20 fichiers qui sont utilisées.

Finalement, dans le but de comparer la qualité de prédiction de notre approche avec celle de SVMRANK, nous avons exécuté cette méthode avec différentes valeurs du paramètre C . Pour rappel (cf. paragraphe 2.3.1.1.2), C est le paramètre qui permet le compromis entre la largeur de la marge et le nombre d'erreur de classement i.e $C \in \{0,001, 0,003, 0,005, 0,01\}$. Apres l'exécution de SVMRANK, nous gardons les meilleurs résultats pour représenter la *baseline* (ce qui est favorable à SVMRANK).

4.4.1 Best-rule versus Weighted Voting

Dans cette sous-section, nous évaluons la qualité de prédiction du profil utilisateur construit par PROFMINER en utilisant les stratégies de classements *Best Rule (BR)* et *Weighted Voting (WV)* ; ces stratégies de classements sont introduites dans la section 3.4. La figure 4.6 montre la moyenne des précision, rappel et f-mesure des 20 profils utilisateur quand le seuil minimum d'accord k varie entre 1 et 3500.

Il est à noter que les performances des deux stratégies de classements sont très similaires. Ce résultat peut être expliqué par le principe de construction utilisé par PROFMINER pour construire le profil utilisateur. En effet, à chaque itération, PROFMINER supprime les règles de préférences contextuelles non-pertinante dans le but de réduire la redondance entre les règles de préférences. C'est ainsi que, par construction, chaque préférence utilisateur est couverte à la fin par peu de règles de préférences contextuelles, ce qui implique que le *weighted voting* est principalement conduit sur un ensemble de règles de préférence contextuelles extrêmement petit (la plus part du temps, seulement une règle, qui est la meilleure).

La seconde observation majeure est que la qualité de prédiction des profils extraits peut être assez élevée. Plus précisément, la précision reste toujours élevée, tandis que le rappel et la f-mesure dépendent profondément du seuil minimal d'accord, i.e la taille du profil utilisateur.
De plus, nous pouvons voir que la précision du profil utilisateur est toujours plus grande que la précision obtenue en utilisant SVMRANK. Cependant, en utilisant la méthode *Best Rule (BR)* et *Weighted Voting (WV)*, nous notons que le rappel et la f-mesure peuvent décroitre fortement quand la taille du profil utilisateur diminue, i.e, quand k augmente.

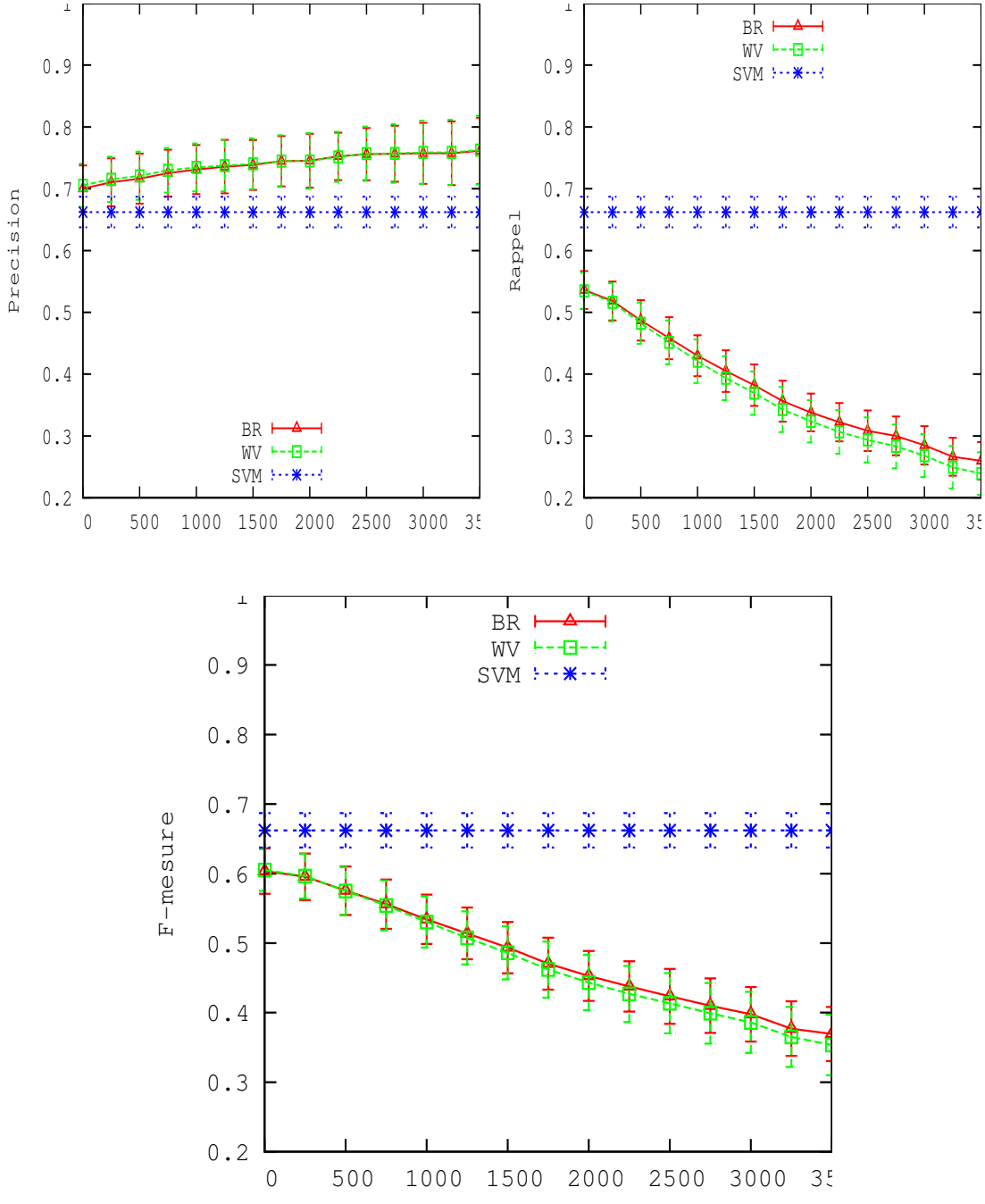


Figure 4.6 – Efficacité des strategie Best Rule et Weighted Voting

4.4.2 Best-rule versus Range Voting

Nous avons présenté dans la sous-section 3.4.3 la stratégie de classement *Range Voting (RV)* permettant de résoudre le problème de l'inconsistance et de l'éparcité des relations de préférences induites par les stratégies de classement *Best Rule (BR)* et *Weighted Voting (WV)*. Nous évaluons dans cette sous-section, les avantages de cette nouvelle stratégie de classement comparée à la méthode *Best Rule (BR)*(sachant que les méthodes BR et WV sont comparables).

La figure 4.7 montre la moyenne des précision, rappel et f-mesure sur les 20 profils d'utilisateur, en utilisant soit la méthode *Best Rule (BR)* ou la méthode *Range Voting (RV)*. La première observation importante est que la précision est toujours restée assez élevée, i.e. plus grande que la précision obtenue en utilisant SVMRANK. Par contre, si le rappel décroît fortement avec la méthode BR (quand k augmente), ce n'est plus le cas avec la méthode RV.

En effet, avec une valeur du seuil d'accord k plus petite que 2000, le rappel reste toujours plus grand que le rappel obtenu en utilisant SVMRANK. Ces résultats sont très importants ; ils montrent que même avec un petit profil (avec $k = 2000$, le nombre de règles de préférences dans le profil est plus petit que 30, cf. figure 4.4), la qualité prédictive (à la fois, la précision et rappel) obtenue est très élevée.

Finalement, la figure 4.9 montre la différence de précision, de rappel et de f-mesure en utilisant soit les stratégies *Best Rule (BR)* ou la stratégie *Range Voting (RV)*. Elle montre que si la différence de prédiction est en faveur de la stratégie BR quand k augmente, la différence de rappel et de f-mesure soulignent l'intérêt de la stratégie du RV.

4.4.3 Range Voting versus Order-by-pref

Dans [CSS99], les auteurs introduisent une nouvelle méthode, intitulée ORDER-BY-PREF (OBP), pour construire un ordre total qui est en accord au mieux avec la relation de préférence apprise. Cela signifie que cette méthode peut être utilisée comme une stratégie de classement en se basant sur la stratégie de classement de Best Rule. Ainsi, dans cette sous-section, nous comparons la qualité de prédiction des stratégies RV et OBP.

La Figure 4.8montre la moyenne de la précision, du rappel et de la f-mesure sur les 20 profils utilisateur, en utilisant soit la stratégie du *Range Voting (RV)* ou la stratégie ORDER-BY-PREF. Il montre que la stratégie du range voting RV dépasse toujours la stratégie OBP. Plus précisément, nous pouvons voir que la différence de précision est toujours en faveur de la stratégie de RV, et que la différence de précision augmente quand

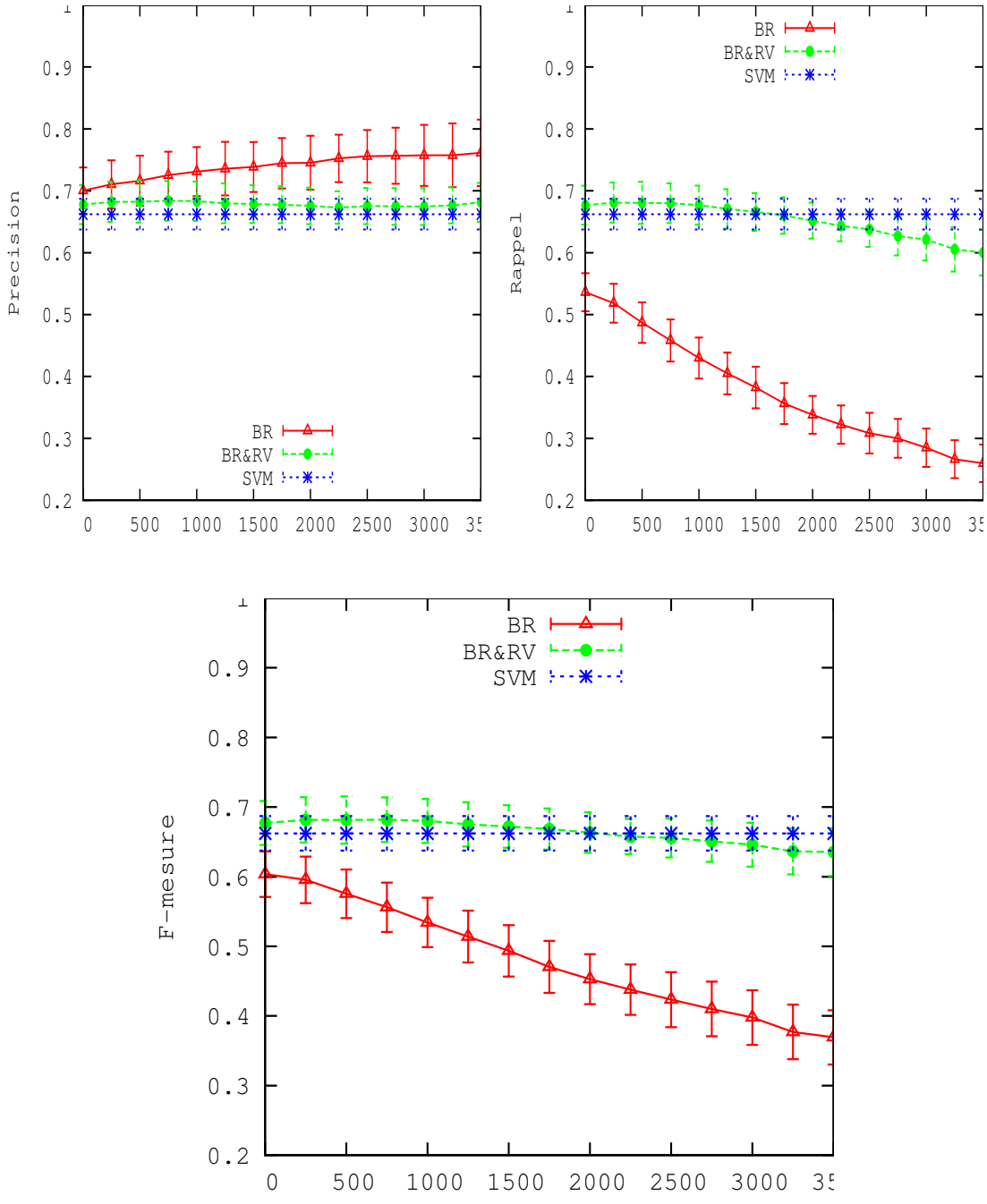


Figure 4.7 – Efficacité des strategie de classement

k augmente, (ce qui veut dire quand la taille du profil utilisateur diminue).

Finalement, nous caractérisons expérimentalement l'ordre $\succ_{\Pi_T}$ induit par les stratégies RV ou OBP comparées à la relation de préférence $\succ_{\Pi}^{br}$ induite par la stratégie BR. La Figure 4.10 montre *le taux d'accord* et le *taux de complétude* de $\succ_{\Pi_T}$ en considérant $\succ_{\Pi}^{br}$. Le taux d'accord défini dans [CSS99] mesure la proportion de décisions de $\succ_{\Pi}^{br}$ restant vrai avec $\succ_{\Pi_T}$:

$$Agree_T(\succ_{\Pi_T}, \succ_{\Pi}^{br}) = \frac{|\{\langle t,u\rangle \in T \times T | t \succ_{\Pi_T} u \wedge t \succ_{\Pi}^{br} u\}|}{|\{\langle t,u\rangle \in T \times T | t \succ_{\Pi}^{br} u\}|}$$

Le *taux de complétude* mesure la proportion de décisions prise par $\succ_{\Pi_T}$ parmi les indécisions de $\succ_{\Pi}^{br}$:

$$Comp_T(\succ_{\Pi_T}, \succ_{\Pi}^{br}) = \frac{|\{\langle t,u\rangle \in T \times T | t \succ_{\Pi_T} u \wedge t \sim_{\Pi}^{best} u\}|}{|\{\langle t,u\rangle \in T \times T | t \sim_{\Pi}^{best} u\}|}$$

où $t \sim_{\Pi}^{best} u \equiv t \not\succ_{\Pi}^{br} u \wedge u \not\succ_{\Pi}^{br} t$.

Nous pouvons voir sur la figure 4.10(a) que le *le taux d'accord* entre les stratégies RV et BR sont élevés signifiant que $\succ_{\Pi_T}$ corrige l'inconsistance de $\succ_{\Pi}^{br}$ en restant similaire avec $\succ_{\Pi}^{br}$. De plus, la performance de la stratégie du RV est similaire à la performance de OBP. D'autre part, le *taux de complétude* affiché par la figure 4.10(b) indique que la stratégie de RV ne conduit pas à un ordre total (qui est, par définition, le cas avec la stratégie OBP), mais ce taux est toujours plus grand que 0.8.

En d'autre termes, en utilisant la stratégie du RV, l'ordre $\succ_{\Pi_T}$ est moins éparse que la relation de préférence $\succ_{\Pi}^{br}$. Plus précisément, le nombre d'indécisions est plus grand quand la taille du profil utilisateur est faible.

4.5 Conclusion

Dans ce chapitre, nous avons présenté les expérimentations effectuées pour valider notre approche. Nous avons testé les performances des algorithmes développés pour construire les profils utilisateur à savoir : les algorithmes ContPrefMiner et ProfMiner. Nous avons également testé les trois méthodes de prédiction que nous avons développées.

Les expérimentations ont montré ContPrefMiner utilise plus la contrainte de support pour élaguer les règles de préférences contextuelles, tandis que la contrainte de confiance influe peu sur le temps d'extraction des règles de préférence contextuelles. Les expérimentations effectuées sur ProfMiner montrent que les profils construits sont

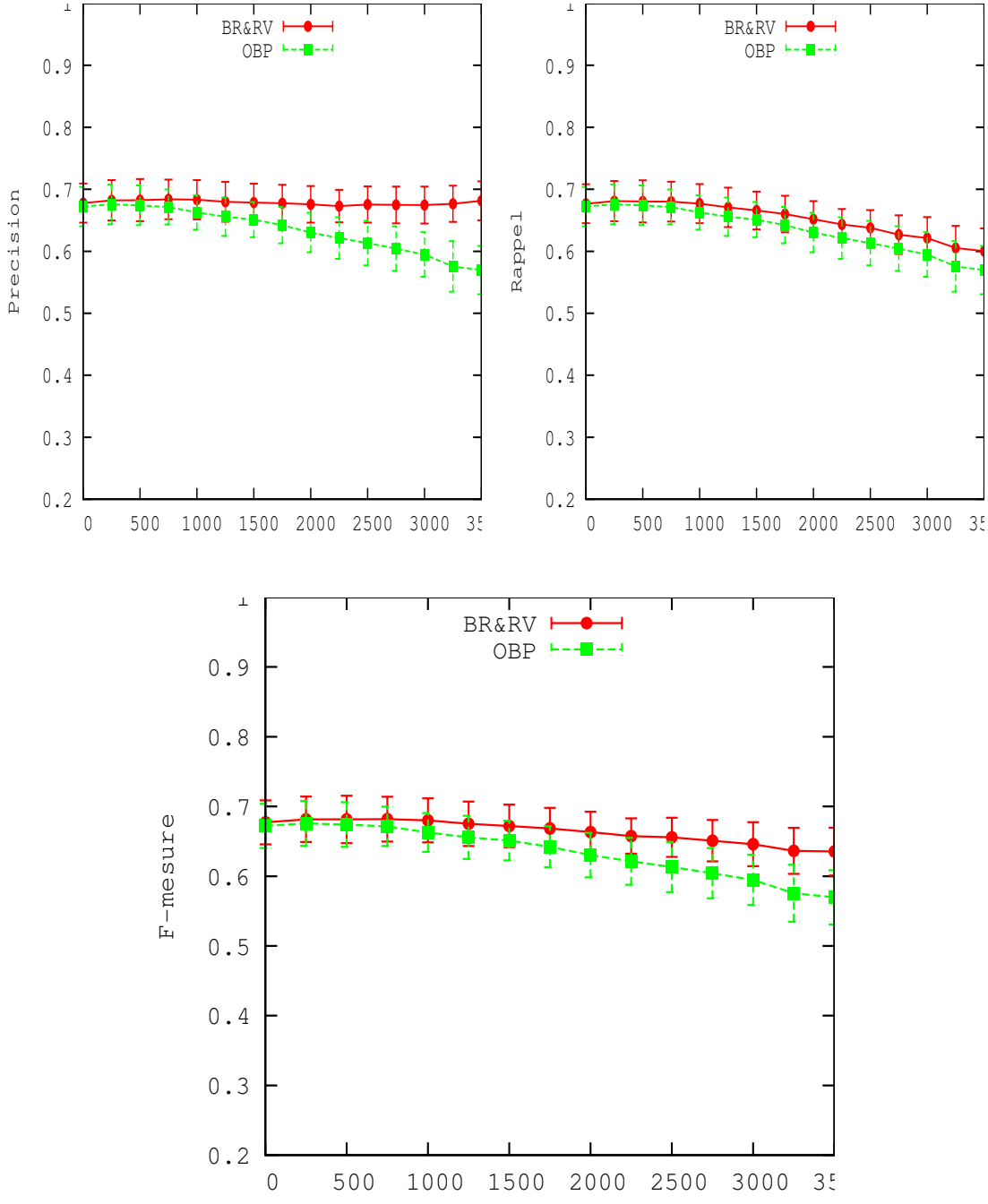


Figure 4.8 – Efficacité des stratégies Range Voting et Order-by-Pref

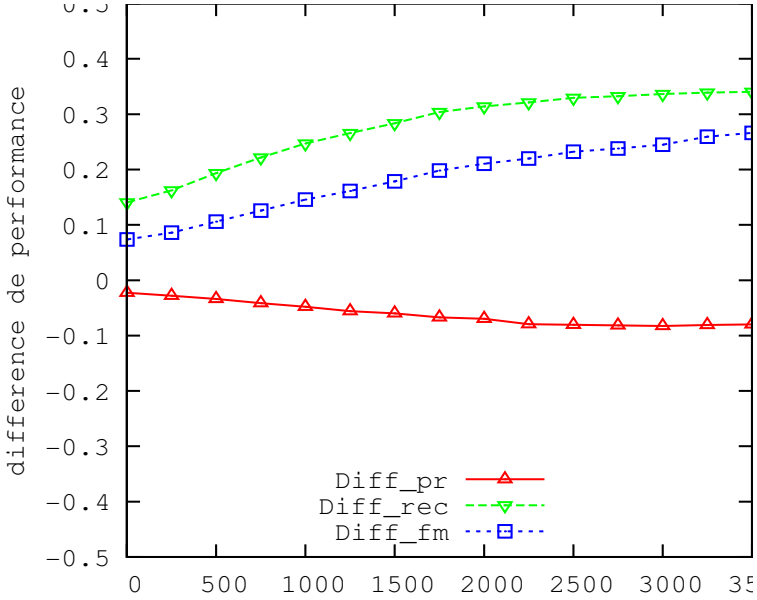
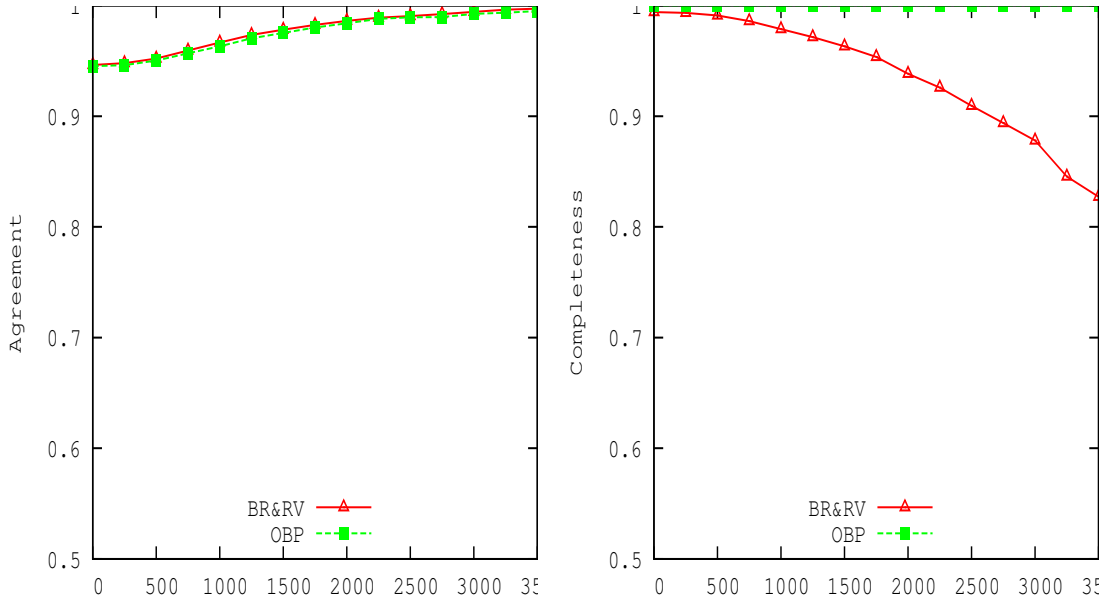


FIGURE 4.9 – Differences de precision (Diff_pr), rappel (Diff_rec) et F-mesure (Diff_fm) entre les strategie Range Voting et Best Rule.



(a) Accord (b) Complétude

FIGURE 4.10 – Taux d'accord et de complétude de $\succ_{\Pi_T}$ en fonction de $\succ_{\Pi}^{br}$

facilement compréhensibles par l'utilisateur. Elles montrent aussi que le seuil d'accord k permet d'avoir des profils aussi concis que désiré. Ces expérimentations ont montré également l'importance d'utiliser les contextes dans le but d'avoir des profils consistants.

Les expérimentations sur les méthodes de prédiction associées au profil extraits, montrent que le profil conduit à une bonne qualité de prédiction : la précision est toujours élevée pour les trois méthodes de classements, tandis que pour la méthode *Best Rule (BR)* et *Weighted Voting (WV)*, le rappel dépend du nombre de règles dans le profil.
Les expérimentations effectuées sur le *Range Voting (RV)* ont montré que cette méthode de prédiction amorti la chute du rappel ce qui montre qu'avec le *Range Voting* , on peut avoir peu de règles dans le profil et avoir une bonne qualité de prédiction.

Nous avons également comparé la qualité de prédiction de notre approche avec deux méthodes de l'état de l'art à savoir la méthode OBP [CSS99] et SVMRank [Joa02] montrant que la qualité de prédiction de notre approche peut concurrencer certaines méthodes de l'état de l'art.

Conclusion

Dans le chapitre 3, nous avons présenté nos contributions à savoir : une approche à deux phases pour l'extraction des règles de préférences contextuelles et la construction du profil utilisateur. Nous avons également présenté les méthodes de prédiction que nous avons proposées.
Les expérimentations présentés dans la chapitre 4 montrent l'efficacité de notre approches : les profils construits peuvent être aussi concis que désirés, ils sont aussi facilement interprétables et ont une bonne qualité de prédiction.
Les expérimentations ont montré que la qualité de prédiction de nos profils peuvent rivaliser avec certaines méthodes de l'état de l'art : SVMrank [Joa02] et OrderbyPref [CSS99]

Conclusion générale et perspectives

Dans le cadre de cette thèse, nous nous sommes intéressés à la construction de profil utilisateur en se basant sur ses préférences. Nous avons également défini quelques méthodes de prédictions utilisant le profil construit (cf. chapitre 3). Les méthodes de prédictions proposées permettent de prédire les futures préférences de l'utilisateur en se basant sur les préférences antérieures de ce dernier (cf. section 3.4).

L' état de l'art, sur les méthodes de découverte de préférences, que nous avons effectué a montré les limites des approches présentées (cf. chapitre 2). Les limites que nous avons décelées sont notamment, pour les approches quantitatives, la difficulté pour l'utilisateur de comprendre le modèle construit [Joa06, FISS03, XL07, BSR+05]. Tandis que pour certaines approches qualitatives, les auteurs ne considèrent pas les préférences contextuelles [HEK03, Jea08], d'autres ne peuvent pas travailler sur des données bruitées [CKL+10].

Nous avons proposé dans cette thèse, une approche qualitative permettant de construire un profil utilisateur lisible et facilement compréhensible par l'utilisateur. Nous avons également proposé trois méthodes de prédiction montrant que le modèle construit permet d'avoir une bonne qualité de prédiction.
Dans la suite, nous présentons un bilan des travaux que nous avons effectués ainsi que quelques perspectives.

Bilan

Algorithme d'extraction de preferences Comme nous l'avons exprimé précédemment, pour construire le modèle de préférences, nous avons utilisé une approche à deux phases. La première phase constitue la phase d'extraction des règles de préférences contextuelles. Pour extraire ces règles de préférences, nous avons proposé un algorithme en profondeur inspiré d'Eclat [Zak00] nommé ContPrefMiner. Le but cet algorithme est d'extraire toutes les règles de préférences contextuelles minimales satisfaisant un seuil minimum de support et un seuil minimum de confiance fixé.

Algorithme de construction de profil Pour la seconde phase de notre approche, nous avons proposé un algorithme nommé PROFMINER permettant de construire un profil utilisateur. Après la première phase (phase d'extraction des règles de préférences), une seconde phase est nécessaire pour éliminer les règles de préférences redondantes et superflues. Notre algorithme pour construire le profil utilisateur est inspiré de CBA [LHM98]. Le but de l'algorithme est de construire un profil utilisateur *concis* et *consistant* à partir de l'ensemble des règles de préférences contextuelles extraites. L'algorithme proposé suit une approche heuristique étant donnée que le problème de la construction du profil utilisateur est NP-complet (cf. Problème 3).

Des profils de qualité Les profils que nous construisons sont facilement interprétables puisque nous avons utilisé des règles de préférences (cf. table 4.3 et table 4.4). Le seuil d'accord K que nous avons introduit permet de gérer la taille du profil, ainsi nous pouvons avoir un profil aussi concis que désiré. Malgré la concision des profils que nous construisons, les expérimentations ont montré que ces profils permettent d'avoir une bonne qualité de prédiction (cf. chapitre 4).

Méthodes d'utilisation du profil Nous avons développé trois méthodes de prédiction montrant que notre modèle permet d'avoir une bonne qualité de prédiction. Les méthodes prédictives que nous avons développées sont les méthodes Best-rule, Weighted Voting et RangeVoting (cf. section 3.4). Les expérimentations que nous avons effectuées, avec ces méthodes prédictives, montent que les performances obtenues peuvent rivaliser avec SVMrank [Joa06] (cf. chapitre 4).

Un prototype pour les expériences Nous avons développé un prototype permettant d'extraire des règles de préférences contextuelles et de construire des profils utilisateur (cf. Annexe B). L'implémentation permet d'afficher plusieurs informations au niveau de la première phase (phase d'extraction de toutes les règles de préférences). Ces informations sont entre autres, le support moyen des règles extraites ainsi que la taille moyenne des contextes des règles de préférence extraites.
Au niveau de la deuxième phase (phase de construction du profil), l'implémentation affiche la taille moyenne du profil (le nombre de règles de préférence contextuelle) ainsi que les mesures permettant d'évaluer la qualité de prédiction du profil construit (i.e *précision, rappel et f-mesure* ...).

Une validation solide Nous avons proposé une série d'expérimentations permettant de valider notre approche (cf. chapitre 4). Ces expérimentations permettent de valider

les propositions que nous avons effectuées dans cette thèse. Les expérimentations effectuées sur l'algorithme ContPrefMiner concernent la quantité des règles extraites en fonction des contraintes de support et de confiances (cf. section 4.2). Tandis que les expérimentations que nous avons effectuées sur ProfMiner concernent la concision du profil et l'intelligibilité du profil (cf. section 4.3). Nous avons également effectué des expérimentations pour évaluer la qualité de prédiction des stratégies de classement que nous avons développées (cf. section 4.4). Ces expérimentations montrent que les stratégies de classement proposées fournissent une bonne qualité de prédiction. Nous avons constaté également que les performances de la méthode range voting peuvent rivaliser avec les performances de SVMRank [Joa02].

Troisième partie

Annexes

Table des matières

Annexe A

Pré-traitement

Nous présentons dans cette partie les pré-traitements que nous avons effectuées pour obtenir des données utilisables par notre prototype. Cette partie est organisée de la façon suivante : Dans la partie Introduction A.1, nous présentons le format de données que nous voulons obtenir. Dans la section A.2, nous présentons le code de création de la base de données sur laquelle nous travaillons. Dans la section A.3, nous présentons les différentes étapes de la phase de pré-traitement.

A.1 Introduction

Les données sur lesquelles nous travaillons contiennent des informations sur des films et sont stockées dans une base de données relationnelle sous forme attribut valeur. Notre objectif est de les transformer pour obtenir des données, ayant le format de SVMRank, que nous utiliserons comme entrée de notre algorithme. Le format d'une ligne d'un fichier SVMRank est le suivant :

<target> qid :<qid> <feature> :<value> <feature> :<value> ... <feature> :<value> .

<target> .=. <float> représente le score affecté à l'objet.
<qid> .=. <positive integer> représente l'identifiant de l'utilisateur qui a noté l'objet.
<feature> .=. <positive integer> représente un nombre qui correspond à la valeur d'un attribut.
<value> .=. <float> est une valeur qui permet de vérifier la présence où l'absence de la valeur d'un attribut dans un objet, cette valeur prends 1 ou 0.

Chaque ligne représente un objet, un score affecté a l'objet, un identifiant d'un utili-

sateur et une liste d'attributs (nombre : 0 ou 1). 0 signifie que la valeur de l'attribut ne fait pas parti de cet objet (ou insatisfait). 1 signifie que la valeur de l'attribut fait parti de cet objet (ou satisfait).
La ligne suivante représente un objet noté par l'utilisateur 3781, la note attribuée a l'objet est 3 ; l'objet contient 15 valeurs d'attributs (dont 5 satisfaits pour cet objet). Les attributs satisfaits sont ceux qui sont suivis par un 1 après les deux points.

3 qid :3781 1 :0 2 :0 3 :0 4 :0 7 :1 8 :1 9 :1 10 :1 11 :1 12 :0 13 :0 14 :0 16 :0 17 :0 18 :0

Il est à noter que les lignes contiennent le même nombre d'attributs.

A.2 Présentation de la base de données

Voici le code SQL que nous avons utilisé pour construire les tables de la base de données.

```
CREATE TABLE ‘mpmovies‘ (
‘ID‘ int(11) NOT NULL,
‘Title‘ varchar(256) DEFAULT NULL,
‘Year‘ varchar(8) DEFAULT NULL,
‘Colors‘ varchar(128) DEFAULT NULL,
‘Genre‘ varchar(128) DEFAULT NULL,
‘Countries‘ varchar(128) DEFAULT NULL,
‘Languages‘ varchar(128) DEFAULT NULL,
‘Companies‘ varchar(512) DEFAULT NULL,
‘Producers‘ varchar(512) DEFAULT NULL,
‘Directors‘ varchar(512) DEFAULT NULL,
‘Actors‘ varchar(512) DEFAULT NULL,
‘Actresses‘ varchar(512) DEFAULT NULL,
PRIMARY KEY (‘ID‘)
) ENGINE=InnoDB DEFAULT CHARSET=utf8$$

CREATE TABLE ‘mpratings‘ (
‘User‘ int(11) NOT NULL,
‘Movie‘ int(11) NOT NULL,
‘Rating‘ int(11) DEFAULT NULL,
‘Timestamp‘ int(11) DEFAULT NULL,
PRIMARY KEY (‘User‘,‘Movie‘)
) ENGINE=InnoDB DEFAULT CHARSET=utf8$$
```

```
CREATE TABLE ‘mpusers‘ (
‘ID‘ int(11) NOT NULL,
‘Gender‘ varchar(1) DEFAULT NULL,
‘Age‘ varchar(5) DEFAULT NULL,
‘occupation‘ varchar(64) DEFAULT NULL,
PRIMARY KEY (‘ID‘)
) ENGINE=InnoDB DEFAULT CHARSET=utf8$$
```

On constate ainsi que la base de données contient trois tables :

- Une table nommée **mpusers** qui contient des informations sur les utilisateurs,
- Une table **mpmovies** qui contient des informations sur les films,
- Une table **mpratings** qui contient les notes attribuées aux films par les utilisateurs

Autres caractéristiques de la base :

- Nombre d'utilisateurs : 6040
- Nombre de films : 3881
- Nombre de films notés : 3704

La figure A.1 représente le nombre de notes en fonction des utilisateurs.
On constate sur cette figure que 3095 utilisateurs ont regardé entre 1 et 100 films, 22 utilisateurs ont regardé entre 900 et 1000 films ...

A.3 Pré-traitement des données

Dans cette section, nous montrons les pré-traitements que nous avons effectués pour obtenir des données que nous pouvons manipuler avec notre algorithme.
Nous rappelons que l'objectif de ce pré-traitement est d'avoir des données qui ont le format de SVMRank. Pour cela, un ensemble de scripts à été développé pour arriver à cet objectif. Nous allons ainsi décrire toutes les étapes du pré-traitement : de la base de données jusqu'à l'obtention des fichiers au format SVMRank. Nous allons principalement montrer les étapes suivantes :

- Exporter les données des trois tables citées plus haut vers un fichier texte nommé **"movpref.txt"**.
- Transformer les données du fichier **"movpref.txt"** pour pouvoir associer à chaque valeur d'un attribut, un nombre entier positif.

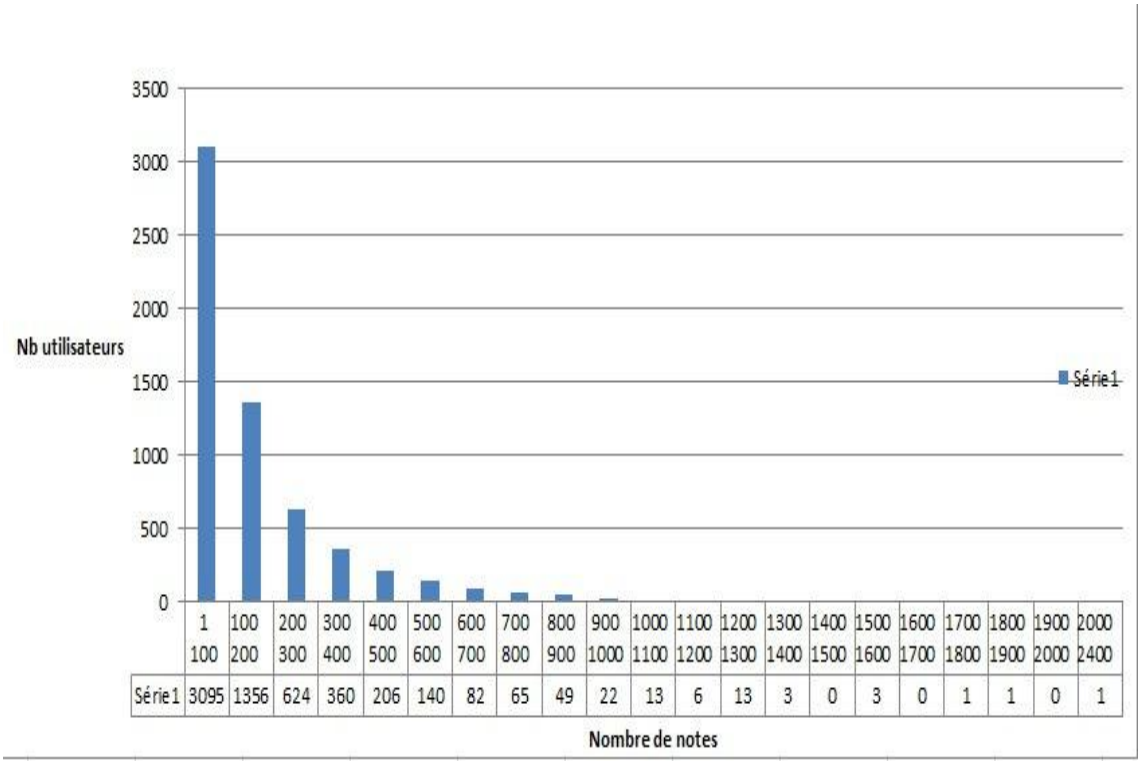


FIGURE A.1 – Notes des utilisateurs

Á la fin de ce pré-traitement, nous obtenons deux fichiers, nommées : **"movpref.bin.txt"** et **"movpref.map.txt"**. Chaque ligne du fichier **"movpref.bin.txt"** contient un objet dont les valeurs des attributs sont représentées par des nombres entiers, tandis que le fichier **"movpref.map.txt"** contient le dictionnaire qui associe à chaque nombre, la valeur de l'attribut correspondant.

– En fin, les données au format SVMRank sont obtenues à partir du fichier **"movpref.bin.txt"**grâce à un script python.

Pour illustrer les résultats du pré-traitement, nous allons utiliser les résultats obtenus sur l'utilisateur 21.

A.3.1 Exporter les données

./mp-export.php >movpref.txt.

Cette commande permet d'exporter les données depuis la base de données **"movpref"** vers un fichier texte. Lors de l'exportation des données,une jointure des trois tables(mpusers, mpmovies et mpratings) est effectuée.La table temporaire que je nomme "UserMovRat" (qui est la table résultante de la jointure des trois tables) contient les attributs 11 attributs,les attributs sont les suivants :
UserMovRat(User,Timestamp,Movies,Rating,Title,Year,Colors,Genre,Countries,Languages, Companies,Producers,Directors,Actors,Actress).
Voici un extrait du fichier "movpref.txt", il contient le résultat de la jointure des trois tables.

(certains attributs sont représenté sous formes de nombres). (mettre les attributs manuellement,expliquer le format)

[USR]21 [Time]978138933 [MOV]585 [RAT] 3 []Brady Bunch Movie, The (1995) 1990s Color [GEN]Comedy United States [LAN]English [COM]Paramount Pictures [PRO]Kirkpatrick, David (I) [PRO]Schwartz, Lloyd J. [PRO]Schwartz, Sherwood [DIR]Thomas, Betty (I) [ACT]Cole, Gary (I) [ACT]Barnes, Christopher Daniel [ACT]Sutera, Paul [ACT]Long, Shelley [ACT]Taylor, Christine (I) [ACT]Cox, Jennifer Elise.

On constate que certaines valeurs d'attributs sont préfixées par une abréviation du nom de l'attribut correspondante.Par exemple [GEN] correspond à l'attribut genre,[LAN] correspond à l'attribut Languages. Par contre on constate que les valeurs des attributs **User,Timestamp,Movies,Rating,Title et Year** ne sont pas préfixées

par leur nom d'attributs respectifs.

Cette ligne représente un film noté par l'utilisateur 21.On constate que les attributs Producers ici abrégé [PRO]et Acteurs([ACT]) sont multivalués, on peut lire entre autre sur cette ligne que le film est produit par :

- Kirkpatrick, David (I)
- Schwartz, Lloyd J.
- Schwartz, Sherwood

Les acteurs de ce film sont :

- Cole, Gary (I)
- Barnes, Christopher Daniel
- Sutera, Paul
- Long, Shelley
- Taylor, Christine (I)
- Cox, Jennifer Elise.

A.3.2 itemization

(expliquer La remarque) **./mp-itemize.py movpref<movpref.txt**.

Cette commande permet de créer les fichiers **"movpref.bin.txt"** et **"movpref.map.txt"** cités plus haut. Etant donnée la jointure des trois tables de la base de données,chaque ligne du fichier **"movpref.bin.txt"** contient au moins 11 items ; puisque certains attributs sont multivalués, le nombre d'items de chaque ligne du fichier dépasse très souvent 11 items.Lors du pré-traitement toutes les valeurs d'un attribut seront prise en compte. Chaque ligne du fichier **"movpref.bin.txt"**contient un objet qui est représenté par des nombres qui correspondent aux valeurs des attributs de cet objet. Voici une ligne du fichier **"movpref.bin.txt"** :

21 978138933 585 3 7916 2 3 43 7 8 29 2889 7917 7918 1581 7919 7920 7921 5291 4326 7922.

Chaque ligne de ce fichier représente un film noté par un utilisateur, la première valeur de chaque ligne représente l'identifiant de l'utilisateur qui a noté le film. Par exemple la valeur 21 sur la ligne précédente signifie que c'est l'utilisateur 21 qui à regardé le film. En regardant le fichier **movpref.map.txt**, on constate les correspondances suivantes :

7916 Brady Bunch Movie, The (1995)
2 1990s

3 Color
43 [GEN]Comedy
7 United States
8 [LAN]English
29 [COM]Paramount Pictures
2889 [PRO]Kirkpatrick, David (I)
7917 [PRO]Schwartz, Lloyd J.
7918 [PRO]Schwartz, Sherwood
1581 [DIR]Thomas, Betty (I)
7919 [ACT]Cole, Gary (I)
7920 [ACT]Barnes, Christopher Daniel
7921 [ACT]Sutera, Paul
5291 [ACT]Long, Shelley
4326 [ACT]Taylor, Christine (I)
7922 [ACT]Cox, Jennifer Elise

il est à rappeler que le fichier **"movpref.map.txt"** est un dictionnaire, il a le format suivant : **numeroitem valeurattribut.**
numeroitem correspond à un nombre qu'on associe à la valeur d'un attribut de la base de données.
valeurattribut correspond à une chaine de caractère quelconque. Par exemple la valeur 43 est associée à l'attribut **Genre="Comedy"**.

A.3.3 Format SVMrank

./mp-svmrank.py 21 < movpref.bin.txt

Cette commande permet de créer un fichier au format SVMrank pour l'utilisateur 21. Chaque ligne représente un film regardé par un utilisateur, un score affecté au film, un identifiant d'un utilisateur et une liste d'attributs(Nombre :0 ou 1). La ligne ci dessous représente un film noté par l'utilisateur 21, la note attribuée au film est 3 ; le nombre d'attributs du film est :124(seul 17 attributs sont satisfaits).

3 qid :21 2 :1 3 :1 4 :0 5 :0 6 :0 7 :1 8 :1 25 :0 29 :1 40 :0 41 :0 42 :0 43 :1 44 :0 45 :0 46 :0 49 :0 51 :0 57 :0 58 :0 94 :0 123 :0 125 :0 127 :0 128 :0 130 :0 160 :0 161 :0 187 :0 193 :0 200 :0 271 :0 289 :0 324 :0 335 :0 346 :0 365 :0 431 :0 476 :0 477 :0 478 :0 479 :0 480 :0 481 :0 489 :0 626 :0 669 :0 694 :0 746 :0 778 :0 810 :0 890 :0 939 :0 1080 :0 1214 :0 1258 :0 1296 :0 1297 :0 1298 :0 1301 :0 1318 :0 1341 :0 1375 :0 1376 :0

1400 :0 1422 :0 1581 :1 1648 :0 1917 :0 1920 :0 2085 :0 2287 :0 2288 :0 2289 :0 2290 :0 2291 :0 2292 :0 2293 :0 2306 :0 2542 :0 2808 :0 2889 :1 2934 :0 2953 :0 3154 :0 3772 :0 4003 :0 4018 :0 4326 :1 4623 :0 4741 :0 4937 :0 4938 :0 4939 :0 4940 :0 4941 :0 4942 :0 5291 :1 5385 :0 6470 :0 6638 :0 6639 :0 6640 :0 6641 :0 6642 :0 6643 :0 7916 :1 7917 :1 7918 :1 7919 :1 7920 :1 7921 :1 7922 :1 7923 :0 7924 :0 7925 :0 7926 :0 7927 :0 7928 :0 7929 :0 7930 :0 7931 :0 7932 :0 7933 :0

On constate sur cette ligne que le nombre d'attribut dépasse largement 11 ; en plus des attributs multivalués ; et étant donnée que chaque ligne doit contenir le même nombre d'attributs,chaque ligne représentant un film regardé par un utilisateur va contenir tous les attributs de tous les films regardé par cet utilisateur.Pour indiquer que la valeur d'un attribut n'est pas présent dans un objet, on met un Zero après la valeur de l'attribut correspondant tandis que si l'attribut est satisfait, on met un 1 après l'attribut.

Annexe B

Documentation ProfMiner

B.1 Overview

Input data Basically ProfMiner takes as input a training database consisting in objects described by a set of features, plus user preferences over these objects (see option -i). Two file formats are available (see option -q) : *quantitative preferences* and *pairwise preferences*. Quantitative preferences associate one rating per object as user preference. Pairwise preferences consist in a list of object pairs where the first object is preferred to the second one. As pairwise preferences are more expressive than quantitative ones, a pairwise preference database is automatically generated starting from quantitative preferences.

Contextual preference rule mining The first stage of ProfMiner is to mine exhaustively all the relevant contextual preference rules. A contextual preference rule is of the form $i^+ \succ i^- \,|\, X$ where X is an itemset, i^+ and i^- are items not belonging to X. The left-hand side of a preference rule specifies the choice while the right-hand side is the context. For instance, $D \succ E \,|\, AB$ means that the context AB leads to choose the item D to the item E. Several options allows to define what a relevant rule is using a minimal support (see -g and -s), a minimal confidence (see -c), a minimaliy (see -a) and a field (see -f) constraints.

Profile construction The second stage of ProfMiner is to build a *user profile*. A profile is a set of preference rules that aims at being *concise* and *sound* with respect to his/her preferences. ProfMiner offers several methods for this construction (see option -r).

Test and statistics The efficiency of user profile built by PROFMINER can be evaluated by means of a test database (option -T) or by means of a k fold cross-validation (option -C). In both cases, use -S for reporting many statistics. Note that several methods for prediction are available by combining -r and -R.

Information about PROFMINER A short help is available with -h while the version number is obtained with -v. Finally, a verbose mode -V gives a lot of information during the process.

B.2 Input Data

B.2.1 Pairwise Format

Use -i file.bin file.pref (or –input) to specify file.bin and file.pref as inputs and the option -q (or –quantitative) must not be used because the default format is the pairwise one.

Let us recall that pairwise preferences consist in a list of object pairs where the first object is preferred to the second one. In practice, there are two files :

1. .bin file : each line of the file represents an object (or a transaction). An object is a list of features/items encoded by numbers.
2. .pref file : each line of the file represents a pair. The first number is the preferred transaction and the second number is the non-preferred transaction. A number 1 refers to the first transaction of the .bin file.

Note that lines beginning by a # are ignored.

Let us illustrates this file format with a simple example :

```
$ cat data/data.bin
# data.bin
1 2 4 5
1 3 5
1 2 3 4 6
2 3 4 5
$ cat data/data.pref
# data.pref
1 3
1 4
2 3
3 4
```

B.2.2 Quantitative Format

Use -i file.svm to specify the file file.svm as input and the option -q must be used for activating the quantitative file format.

The quantitative file format is the format used by SVMrank. More precisely, each line represents an object with a rating, a user identifier, a list of features (NUMBER :0 or 1). The below line represents an object of the user 3781 with the rating 3 containing 15 features (only 5 features are satisfied for this object).

```
3 qid:3781 1:0 2:0 3:0 4:0 7:1 8:1 9:1 10:1 11:1 12:0 13:0 14:0 16:0 17:0 18:0
```

Note that :

- Lines beginning by a # are ignored.
- All the lines must have the same features.
- A feature/item is a number.
- Only one user per file.

B.2.3 Dictionary

Use -t file.txt (or –translation) to specify the dictionary file.txt as translation.

A dictionary allows to translate each item by its corresponding translation. Each line must be formatted as follow :

```
item_number item_translation
```

item_number corresponds to one item number among those that are listed in the .bin file if option -q is not used. If the option -q is used (quantitative dataset), the item_number corresponds to a feature number (and not a column number).

item_translation is any string of characters. Note that it is possible to specify a field name as prefix for item_translation with the syntax ATT :val or ATT=val or ATT ;val.

For instance, this is a short excerpt of a dictionary having several fields :

```
$ head data/dict.txt
1       GEN:Adventure
2       GEN:Drama
3       GEN:History
4       ACT:Hanks/Tom
5       ACT:Quinlan/Kathleen
6       DIR:Howard/Ron (I)
7       DEC:1990
8       LAN:English
```

```
9       COL:Color
10      GEN:Action
```

B.3 Contextual Rule Mining

B.3.1 Constraints

There are 4 different notions which are useful for defining constraints :

- **Support** : the *support* of a contextual preference rule π which estimates the probability that π agrees with a pair
- **Confidence** : the *confidence* of a contextual preference rule π measures the proportion of user preferences in agreement with π among pairs covered by π
- **Minimal** : a contextual preference rule $i^+ \succ i^- \mid X$ is *minimal* iff there is no contextual preference rule $i^+ \succ i^- \mid Y$ such that $Y \subset X$ with the same support and the same confidence than $i^+ \succ i^- \mid X$
- **Field constraint** : $i^+ \succ i^- \mid X$ satisfies the field constraint iff i^+ and i^- are derived from a same filed

Use -g (–gamma) to set a minimal absolute frequency threshold or use -s (–support) to set a minimal support threshold. By default, the minimal absolute frequency threshold is 1.

Use -c (–confidence) to set a minimal confidence threshold. By default, this threshold is 0.5.

By default, PROFMINER mines only contextual preference rules that are minimal. Use -a (–all) to mine all the contextual preference rules including non-minimal ones. In fact, the prototype mines all the *prefix-minimal* rules instead of minimal ones.

Use -f (–field) to restrain the language of contextual preference rules to those having the choice part based on the same field. Obviously, this option only works when a translation (option -t) is simultaneously used where the dictionary satisfies a particular syntax (see above).

B.4 Profile Construction

B.4.1 Primary Ranker for Construction

The construction of the user profile depends on the primary ranker :

– all-rules and hedge : All the rules are preserved and sorted with respect to the condifence (descending order), the support (descending order) and the cardinality (ascending order).
– first-rule k l and harmony k l : Note that k and l are two parameters of the approach. The construction iterates two main principles over the contextual preference rules returned by the first stage until all user preferences in the database are in agreement with at least l preference rule in the profile : (1) select the best contextual preference rule and (2) remove the unnecessary contextual preference rules (i.e., a rule that doesn't cover at least k remaining pairs).

Use -r (–ranker) to select the right primary ranker. By default, the primary ranker is first-rule 1 1.

B.4.2 User Profile Format

The command bin/profminer -i data/data.bin data/data.pref returns the user profile built with the default primary ranker i.e., first-rule 1 1 :

```
$ bin/profminer -i data/data.bin data/data.pref
5 > 6 | 1  : 0.5 : 1
1 > 3 |  : 0.25 : 1
1 > 5 |  : 0.25 : 1
```

For instance, the first line represents the contextual preference rule item $5 \succ$ item 6 | item 1 has a support of 0.5 and a confidence of 1. The two other rules have no context. Note that the rules are sorted according to the condifence (descending order), the support (descending order) and the cardinality (ascending order). Finally an arbitrary order is used for ordering the two last rules.

B.5 Prediction

B.5.1 Primary Ranker

– all-rules and first-rule : The best rule that matches a pair of transactions/objects is used for predicting the preferred and the non-preferred transactions.
– harmony : Each rule that matches a pair of transactions/objects adds its confidence to its preferred transaction. Finally, the transaction having the highest sum is the preferred transaction.

Let us recall that -r (–ranker) allows to select the desired primary ranker.

B.5.2 Secondary Ranker

Use -R (–Ranker) to select the desired secondary ranker. A secondary ranker is a ranker that benefits from antoher ranker (named primary ranker) for making decision. by default, the range voting is used.

- none : No secondary ranker is used. With this option, all the predictions are based on the primary ranker.
- range-voting (default option) : A range voting is organised for electing the best transaction where the voters are all the transactions to order.
- order-by-pref : This argument implements the *Learning to order thing* method.

B.6 Test

B.6.1 Simple Test File

Use -T (–test) to specify a pairwise dataset (two files, if -q is not used) or a quantitative dataset (one file, if -q is used) for evaluating the model trained from the file(s) by means of -i.

B.6.2 Cross-Validation

Use -C k (–cross-validation) (where k is number) to perform a k fold cross-validation over the file(s) by means of -i. If k is omitted, then 10 fold are used.

B.7 Statistics

Use -S (–statistics) to compute and print statistics. By default, there is no statistic.

The statistic report is divided into at most 4 parts (depending on the other options) as illustrated below. Note that all the measures obtained from the different folds are averaged for taking into account cross-validation (option -C).

1. **Taining data** (Lines 5-8, below) : it gives information about input data. More precisely, the number of instances (two instances for one pair) and the number of distinct instances are indicated.
2. **Phase 1 : Contextual preference extraction** (Lines 11-20) : it summarizes the mining step by reporting parameters (minimal support and minimal confidence threshols) and the size of data (the number of preference pairs). The second part shows the number of mined contextual preference rules. It also gives the average

length of contexts, the average support and the average confidence of the whole collection.

3. **Phase 2 : Profile construction** (Lines 23-30) : This part provides information about the construction of the user profile. It first recalls what is the primary ranker (with its parameters). After, the size of the profile is given. In the same way as for the mining step, the average length, support and confidence are reported.
4. **Test** (if -T or -C) (Lines 33-51) : A first part displays information about inputs : the secondary ranker (if none, the primary ranker indicated above is used), the number of folds (only one 1 if -T) and the number of (distinct) intances of test file. Then, there are different measures for evaluating the quality of the model :
   - If there is a secondary ranker, the agreement measures its agreement rate with the primary ranker. The completeness indicates the percentage of pairs that are ordered.
   - The classical recall, precision and F-measure are then given.
   - Finally, the accuracy is the percentage of right predictions of the model as recall. But, when the model can't make a decision, it chooses randomly. Note that the accuracy is always greater than the recall and then, it is the right measure for comparing ProfMiner to another.
   - Total time gives the execution time.

```
$ bin/profminer -q -i data/data3568.svm -s 0.01 -S -C
 0 # ==========
 1 # Statistics
 2 # ==========
 3 #
 4 #
 5 # Training data
 6 # -------------
 7 # Average number of Training instances: 4735.6
 8 # Average number of distinct Training instances: 165.2
 9 #
10 #
11 # Phase 1: Contextual preference extraction
12 # ----------------------------------------
13 # Minimal support: 0.01
14 # Minimal confidence: 0.5
15 # Average number of preference pairs: 2367.8
16 #
17 # Average number of contextual preferences: 1285.2
18 # Average length: 0.800809
```

```
19 # Average support: 0.0265759
20 # Average confidence: 0.819697
21 #
22 #
23 # Phase 2: Profile construction
24 # -----------------------------
25 # Primary ranker: first-rule 1 1
26 #
27 # Average size of profile: 301.3
28 # Average length: 0.594092
29 # Average support: 0.0217645
30 # Average confidence: 0.861763
31 #
32 #
33 # Test
34 # ----
35 # Secondary ranker: range-voting
36 # Number of folds: 10
37 # Average number of test instances: 38.4
38 # Average number of distinct test instances: 14.8
39 #
40 # Average agreement: 0.938527
41 # Average completeness: 0.185635
42 # Average answer: 0.932408
43 #
44 # Average precision: 0.629365
45 # Average recall: 0.629365
46 # Average F-measure: 0.629365
47 #
48 # Average accuracy: 0.629365
49 #
50 #
51 # Total time: 14.87
```

B.8 Command Line Helps

B.8.1 Verbose Mode

Use -V (or –verbose) to activate the verbose mode.

– -V 0 (default) : verbose mode disabled

– -V 1 or -V : light verbose mode activated that indicates each stage

– -V 2 : medium verbose mode activated that gives more information during the test stage (precision, recall and accuracy for each fold)
– -V 3 : heavy verbose mode activated that fills the screen with a progress bar

B.8.2 Short Help

Use -h (or –help) for returning a brief help of ProfMiner

```
$ bin/profminer -h
Use: profminer -i <FILE> ...  [OPTION] ...
Prof. Miner mines a profile of contextual preferences
Example: profminer -i dataset.bin dataset.pref

Options:
 -h, --help                        give this message
 -v, --version                     give the version number
 -i, --input <FILE> ...            binary dataset and preference pairs to mine
 -q, --quantitative                quantitative dataset (SVM rank format)
 -o, --output <FILE>               output of profile (=cout)
 -g, --gamma <NUMBER>              minimal absolute frequency threshold (=1 or -s)
 -s, --support <[0,1]>             minimal support threshold (=1/|D| or -g)
 -c, --confidence <[0,1]>          minimal confidence threshold (= 0.5)
 -r, --ranker <NAME> ...           primary ranker (= first-rule 1 1)
 -R, --Ranker <NAME>               secondary ranker (= range-voting)
 -a, --all                         mine all the preference rules (even non-minimal ones)
 -V, --verbose ...                 verbose mode (= 1)
 -C, --cross-validation ...        perform a cross-validation (= 10)
 -T, --test <FILE> ...             predict preferences on test file
 -S, --statistics                  show the statistics
 -f, --field                       item+ and item- come from the same field
 -t, --translation <FILE>          translation of items

Report bug to <arnaud.soulet@univ-tours.fr>.
```

B.8.3 Version number

Use -v (or –version) for returning the version number of ProfMiner

```
$ bin/profminer -v
 profminer 0.0.7
```

Annexe C

Tableaux Résultats

C.1 Protocol Expérimental

Ces expérimentations ont pour premier objectif de montrer l'avantage du range-voting par rapport à la méthode sans range-voting et pour second objectif de comparer les performances obtenues par ProfMiner à celles obtenus par SVMRank sur les mêmes données.Les tests sont effectués sur 20 fichiers (voir tableau1)avec une validation croisée stratifiée de 5 blocs. Les tests sur ProfMiner sont effectués avec range-voting et sans range-voting en faisant varier le seuil d'accord K de 1 à 3500 en fixant minsupp à 0.006 et minconf à 0.75.

Le tableau1 contient les caractéristiques des fichiers sur lesquels les expérimentations ont été effectuées :à savoir le nombre de films, le nombre d'attributs et le nombre de préférences utilisateurs. Le tableau 2 contient les résultats obtenus avec SVM_Rank en faisant varier le paramètre C. On constate sur ce tableau qu'en moyenne la performance maximale pour SVMRank est obtenue avec C=0.001(voir tableau2 dernière ligne, seconde colonne).

Les tableaux 3 et 4 contiennent (pour chaque seuil d'accord K)les moyennes des performances obtenues avec ProfMiner sur les 20 utilisateurs. Ces résultats sont utilisés pour tracer les courbes de la figure 1 .La figure 1(a) représente la précision (avec et sans range-voting)en fonction du seuil d'accord.Sur la figure1(b) nous avons le rappel(avec et sans range-voting) en fonction du seuil d'accord K ; on constate sur cette figure que la rappel sans range voting dépend fortement de la taille du profil.(on constate une décroissance forte du rappel sans range voting lorsque le seuil d'accord augmente),tandis qu'avec le range voting même avec un seuil d'accord K élevé(une forte réduction du nombre de règles dans le profil) on parvient qu'en même à avoir un rappel intéressant(rappel supé-

rieur à 60%). La figure 1(c) montre l'accuracy(avec et sans range voting) en fonction du seuil d'accord K.

La figure 1(d) représente la différence performances entre ProfMiner et SVMrank.Cette différence représente la variation de l'accuracy obtenue avec rangeVoting en fonction du seuil d'accord K auquel on retranche la moyenne des précisions obtenues avec SVMrank pour C=0.001.Il est à rappeler que c'est avec C=0.001 qu'on obtient la meilleure moyenne des précisions avec SVMrank.On constate sur cette figure,même après une forte réduction du nombre de règles dans le profil(K=2500, ce qui représente une réduction de plus de 95% de la taille du profil) on obtient une accuracy qui est meilleure que la meilleure performance moyenne obtenue avec SVMrank :la courbe différence entre K=1 et K=2500 est au dessus de la ligne Zero.
La figure 2 représente la taille moyenne du profile(figure (a)) et la taille moyenne des contextes des règles(figures 2(b)). On constate qu'en augmentant le taux de réduction des règles, la taille du profile diminue considerablement. La remarque le plus importante à faire pour ces expérimentations est : même avec un seuil d'accord très grand K=2500(ce qui représente une forte réduction du nombre de règles dans le profile) on obtient non seulement de bonnes performances en terme de précision, de rappel et d'accuracy mais en plus notre accuracy est meilleure que la meilleure performance de SVMrank (voir figure 1(d)). Il est à constater qu'avec K=2500, nous avons moins de 10 règles dans le profile.

Fichier	nbfilms	features	nbPref
user850	504	4646	90740
user5555	505	4074	97979
user1701	509	4515	87919
user5317	513	4477	89989
user438	520	4193	98762
user48	541	4232	93365
user1125	544	4391	110026
user1496	551	4622	109650
user4867	552	4640	97860
user4411	553	4626	111208
user3942	559	4569	119600
user5107	564	4843	116700
user533	565	4468	122757
user3884	571	4893	124974
user148	582	4591	120291
user4054	583	4854	117342
user123	584	4491	111993
user5682	588	4492	127827
user692	591	4611	111754
user411	597	4939	129933

Table C.1 – Caractéristique des fichiers

Svm_rank	precision						
	C=0.001	C=0.003	C=0.005	C=0.01	C=10	C=20	C=30
user850	0.6740046	0.678051	0.67346114	0.6723714	0.6723714	0.6723714	0.6723714
user5555	0.6765933	0.6766862	0.67679065	0.67724967	0.67724967	0.67724967	0.67724967
user1701	0.6902069	0.68907905	0.6909035	0.6909035	0.6909035	0.6909035	0.6909035
user5317	0.62154907	0.62021625	0.6153168	0.6157626	0.6157626	0.6157626	0.6157626
user438	0.69177425	0.6913567	0.6925172	0.6925172	0.6925172	0.6925172	0.6925172
user48	0.7408855	0.7406841	0.7406841	0.7406841	0.7406841	0.7406841	0.7406841
user1125	0.6039489	0.5939	0.5922311	0.59173	0.59173	0.59173	0.59173
user1496	0.6223698	0.62467486	0.62494844	0.62421507	0.62421507	0.62421507	0.62421507
user4867	0.6275599	0.62711656	0.62665296	0.62665296	0.62665296	0.62665296	0.62665296
user4411	0.69377595	0.6894458	0.68974584	0.689968	0.689968	0.689968	0.689968
user3942	0.70216125	0.7006396	0.7002262	0.7003093	0.7003093	0.7003093	0.7003093
user5107	0.6983542	0.69634783	0.69578296	0.69578296	0.69578296	0.69578296	0.69578296
user533	0.6569757	0.6522415	0.6508527	0.6510156	0.6510156	0.6510156	0.6510156
user3884	0.6634143	0.664092	0.66401035	0.66433054	0.66433054	0.66433054	0.66433054
user148	0.60271114	0.60153294	0.60094434	0.59878165	0.59878165	0.59878165	0.59878165
user4054	0.6252003	0.6269011	0.62866026	0.6283688	0.6283688	0.6283688	0.6283688
user123	0.693036	0.695398	0.69499886	0.69499886	0.69499886	0.69499886	0.69499886
user5682	0.6376245	0.63706267	0.6345359	0.63378483	0.63378483	0.63378483	0.63378483
user692	0.7095787	0.70899624	0.70895725	0.7060044	0.7060044	0.7060044	0.7060044
user411	0.6277881	0.63012034	0.6304206	0.6304206	0.6304206	0.6304206	0.6304206
Moyenne	0.66297567	0.66222715	0.6616321	0.6612926	0.6612926	0.6612926	0.6612926

TABLE C.2 – SVM_Rank validation croisée

K	Sans range-voting				
	RED	Precision	Rappel	F-mesure	Accuracy
1	0.0	0.7002377	0.5362373	0.6035842	0.6544236
250	0.54848427	0.7104238	0.51846683	0.59549546	0.6543097
500	0.69686246	0.7163671	0.48695284	0.5754024	0.6472591
750	0.7879543	0.7251226	0.45817095	0.5560237	0.64226586
1000	0.84798735	0.7310912	0.42980736	0.53426725	0.63610584
1250	0.8873275	0.7357913	0.40494338	0.5139359	0.62996733
1500	0.9131503	0.73848253	0.3820935	0.49347454	0.6237418
1750	0.93041945	0.7444829	0.35621566	0.47046095	0.6166519
2000	0.9426996	0.74533385	0.3378804	0.4527553	0.6109098
2250	0.95301133	0.75230443	0.32221618	0.4376832	0.6071441
2500	0.95974123	0.75604904	0.30842814	0.4233831	0.60303694
3000	0.96968174	0.7572673	0.28479436	0.39766973	0.5945041
3250	0.9746207	0.7574098	0.26625314	0.37701356	0.5886715
3500	0.97760886	0.7612792	0.2596336	0.3692918	0.58703727

Table C.3 – Moyenne des performances sans range voting pour minsupp=0.006 et minconf =0.75

K	Avec range-voting				
	RED	Precision	Rappel	F-mesure	Accuracy
1	0.0	0.67770797	0.6767169	0.6772118	0.67737114
250	0.54848427	0.68221104	0.6808968	0.681553	0.6817756
500	0.69686246	0.68247265	0.6804007	0.6814343	0.68177503
750	0.7879543	0.683727	0.6800097	0.6818611	0.68253016
1000	0.84798735	0.68323386	0.6768844	0.680034	0.6814676
1250	0.8873275	0.6797558	0.6706894	0.6751777	0.67737544
1500	0.9131503	0.67853844	0.66577923	0.67206454	0.67537534
1750	0.93041945	0.67737496	0.66004336	0.66853255	0.6731392
2000	0.9426996	0.675652	0.65179616	0.66339076	0.6698657
2250	0.95301133	0.67287517	0.64327514	0.6575482	0.6657926
2500	0.95974123	0.67540586	0.637821	0.6556642	0.66617626
3000	0.96968174	0.67466724	0.62128806	0.64586765	0.6616302
3250	0.9746207	0.6765657	0.6052353	0.6364436	0.65856045
3500	0.97760886	0.6813756	0.6000158	0.63532877	0.66021997

TABLE C.4 – Moyenne des performances avec range voting pour minsupp=0.006 et minconf =0.75

Bibliographie

[ART06] Rakesh Agrawal, Ralf Rantzau, and Evimaria Terzi. Context-sensitive ranking. In *SIGMOD*, pages 383–394, 2006.

[AW00] Rakesh Agrawal and Edward L. Wimmers. A framework for expressing and combining preferences. *SIGMOD Rec.*, 29(2) :297–306, May 2000.

[BBD+03] Craig Boutilier, R. Brafman, C. Domshlak, H. Hoos, and D. Poole. CP-nets : A Tool for Representing and Reasoning with Conditional Ceteris Paribus Preference Statements, 2003.

[BGM+05] Ladjel Bellatreche, Arnaud Giacometti, Patrick Marcel, Hassina Mouloudi, and Dominique Laurent. A personalization framework for olap queries. In *DOLAP*, 2005.

[Bou92] D. Bouyssou. Ranking methods based on valued preference relations : A characterization of the net flow network. *European Journal of Operational Research*, 60 :61–67, 1992.

[BP92] D. Bouyssou and P. Perny. Ranking methods for valued preference relations : a characterization of a method based on entering and leaving flows. *European Journal of Operational Research*, 61(1-2) :186–194, 1992.

[BSR+05] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*, ICML '05, pages 89–96, New York, NY, USA, 2005. ACM.

[BZ07] Björn Bringmann and Albrecht Zimmermann. The chosen few : On identifying valuable patterns. In *ICDM*, pages 63–72. IEEE Computer Society, 2007.

[Cho03] Jan Chomicki. Preference formulas in relational queries. *ACM Trans. Database Syst.*, 28(4) :427–466, December 2003.

[Cin09] Binary decomposition methods for multipartite ranking. *Machine Learning and Knowledge Discovery in Databases*, pages 359–374, 2009.

[CKL+10] Yann Chevaleyre, Frédéric Koriche, Jérôme Lang, Jérôme Mengin, and Bruno Zanuttini. Learning ordinal preferences on multiattribute domains : The case of CP-nets. In Johannes Fürnkranz and Eyke Hüllermeier, editors, *Preference Learning*, pages 273–296. Springer, 2010.

[CP04] Li Chen and Pearl Pu. Survey of Preference Elicitation Methods, 2004.

[CSS99] William W. Cohen, Robert E. Schapire, and Yoram Singer. Learning to order things. *J. Artif. Int. Res.*, 10(1) :243–270, May 1999.

[dABAdS13] Sandra de Amo, Marcos L. P. Bueno, Guilherme Alves, and Nádia Félix F. da Silva. Mining user contextual preferences. *JIDM*, 4(1) :37–46, 2013.

[dADD+12] Sandra de Amo, Mouhamadou Saliou Diallo, Cheikh Talibouya Diop, Arnaud Giacometti, Haoyuan D. Li, and Arnaud Soulet. Mining contextual preference rules for building user profiles. In Alfredo Cuzzocrea and Umeshwar Dayal, editors, *DaWaK*, volume 7448 of *Lecture Notes in Computer Science*, pages 229–242. Springer, 2012.

[dADD+13] Sandra de Amo, Mouhamadou Saliou Diallo, Cheikh Talibouya Diop, Arnaud Giacometti, Dominique H. Li, and Arnaud Soulet. Extraction de règles de préférence contextuelle. *Journée fouille de données du GdR I3 (Saint etienne France)*, le 8 avril 2013.

[dADD+14] Sandra de Amo, Mouhamadou Saliou Diallo, Cheikh Talibouya Diop, Arnaud Giacometti, Haoyuan D. Li, and Arnaud Soulet. Contextual preference mining for user profile construction. *submitted to The International Journal of Information Science (JIS)*, 2014.

[DGS+12] Mouhamadou Saliou Diallo, Arnaud Giacometti, Arnaud Soulet, Haoyuan D. Li, Cheikh Talibouya Diop, and Sandra de Amo. Profminer : algorithme de construction de profil utilisateur. *In : 4ème Colloque National sur la Recherche en Informatique et ses Applications (CNRIA 2012), 2012, THIES-BAMBEY, SENEGAL.*, 2012.

[DIV07] J ?zsef Dombi, Csan ?d Imreh, and N ?ndor Vincze. Learning lexicographic orders. *European Journal of Operational Research*, 183(2) :748–756, 2007.

[DKSS11] Krzysztof Dembczynski, Wojciech Kotlowski, Roman Slowinski, and Marcin Szelag. Learning of rule ensembles for multiple attribute ranking problems. In Johannes Fürnkranz and Eyke Hüllermeier, editors, *Preference Learning*, pages pp 217–247. Springer Berlin Heidelberg, 2011.

[DMS04] Ofer Dekel, Christopher Manning, and Yoram Singer. Log-linear models for label ranking. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf,

editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

[FH10] Johannes Fürnkranz and Eyke Hüllermeier. Preference learning : An introduction. In Johannes Fürnkranz and Eyke Hüllermeier, editors, *Preference Learning*, pages 1–17. Springer-Verlag, 2010.

[FISS03] Yoav Freund, Raj Iyer, Robert E. Schapire, and Yoram Singer. An efficient boosting algorithm for combining preferences. *J. Mach. Learn. Res.*, 4 :933–969, December 2003.

[GLS14] Arnaud Giacometti, Dominique H. Li, and Arnaud Soulet. Construction de profils de préférences contextuelles basée sur l'extraction de motifs séquentiels. *14èmes Journées Francophones "Extraction et Gestion des Connaissances" Du 28 au 31 Janvier 2014 à Rennes*, 2014.

[GV10] T. Gärtner and Shankar Vembu. Label Ranking Algorithms : A Survey. In Eyke H. Johannes Fürnkranz, editor, *Preference Learning*. Springer–Verlag, 2010.

[HEK03] Stefan Holland, Martin Ester, and Werner Kießling. Preference mining : A novel approach on mining user preferences for personalized applications. In *PKDD*, pages 204–216, 2003.

[HFCB08] Eyke Hüllermeier, Johannes Fürnkranz, Weiwei Cheng, and Klaus Brinker. Label ranking by learning pairwise preferences. *Artif. Intell.*, 172(16-17) :1897–1916, November 2008.

[HpRZ02] Sariel Har-peled, Dan Roth, and Dav Zimak. Constraint classification : A new approach to multiclass classification and ranking. In *In Advances in Neural Information Processing Systems 15*, pages 365–379, 2002.

[HpRZ03] Sariel Har-peled, Dan Roth, and Dav Zimak. Constraint classification for multiclass classification and ranking. In *In Proceedings of the 16th Annual Conference on Neural Information Processing Systems, NIPS-02*, pages 785–792. MIT Press, 2003.

[HSV95] K.U. Hoffgen, H.U. Simon, and K.S. Vanhorn. Robust trainability of single neurons. *Journal of Computer and System Sciences*, 50(1) :114 – 125, 1995.

[Jea08] Bin Jiang and et al. Mining preferences from superior and inferior examples, 2008.

[Joa99] Thorsten Joachims. Advances in kernel methods. chapter Making Large-scale Support Vector Machine Learning Practical, pages 169–184. MIT Press, Cambridge, MA, USA, 1999.

[Joa02] Thorsten Joachims. Optimizing search engines using clickthrough data. In *KDD*, pages 133–142. ACM, 2002.

[Joa06] Thorsten Joachims. Training linear SVMs in linear time. In *KDD*, 2006.

[KCFS08] Arno J. Knobbe, Bruno Crémilleux, Johannes Fürnkranz, and Martin Scholz. From local patterns to global models : The LeGo approach to data mining. In *Proceedings of the LeGo Workshop*, pages 1–16, 2008.

[KI04] Georgia Koutrika and Yannis Ioannidis. Personalization of queries in database systems. In *Proceedings of the 20th International Conference on Data Engineering*, ICDE '04, pages 597–, Washington, DC, USA, 2004. IEEE Computer Society.

[Kie02] Werner Kießling. Foundations of preferences in database systems. In *VLDB '02 : Proceedings of the 28th international conference on Very Large Data Bases*, pages 311–322. VLDB Endowment, 2002.

[KKA10] Toshihiro Kamishima, Hideto Kazawa, and Shotaro Akaho. A survey and empirical comparison of object ranking methods. In Johannes Fürnkranz and Eyke Hüllermeier, editors, *Preference Learning*, pages 181–201. Springer-Verlag, 2010.

[LHM98] B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. In *Proceedings of the 4th international conference on Knowledge Discovery and Data mining (KDD'98)*, pages 80–86. AAAI Press, August 1998.

[LHP01] Wenmin Li, Jiawei Han, and Jian Pei. CMAR : Accurate and efficient classification based on multiple class-association rules. In *ICDM*, pages 369–376, 2001.

[MFJ+06] Jian Ma, Zhi-Ping Fan, Yan-Ping Jiang, Ji-Ye Mao, and Louis Ma. A method for repairing the inconsistency of fuzzy preference relations. *Fuzzy Sets and Systems*, 157(1) :20–33, 2006.

[Mie08] Pauli Miettinen. On the positive-negative partial set cover problem. *Inf. Process. Lett.*, 108(4) :219–221, 2008.

[MOU07] HASSINA MOULOUDI. *Personnalisation de requêtes et visualisations OLAP sous contraintes.* PhD thesis, Université François-Rabelais-Tours, 2007.

[MT97] Heikki Mannila and Hannu Toivonen. Levelwise search and borders of theories in knowledge discovery. *Data Min. Knowl. Discov.*, 1(3) :241–258, 1997.

[PKB09] Verónika Peralta, Dimitre Kostadinov, and Mokrane Bouzeghoub. APMD-workbench : A benchmark for query personalization. In *Proceedings of the CIRSE Workshop*, pages 38–41, 2009.

[RA05] Shyamsundar Rajaram and Shivani Agarwal. Generalization bounds for k-partite ranking. In S. Agarwal, C. Cortes, and R. Herbrich, editors, *Proceedings of the NIPS 2005 Workshop on Learning to Rank*, pages 28–23, Whistler, BC, Canada, 2005.

[SC08] Arnaud Soulet and Bruno Crémilleux. Adequate condensed representations of patterns. *Data Min. Knowl. Discov.*, 17(1) :94–110, 2008.

[SKP11] Kostas Stefanidis, Georgia Koutrika, and Evaggelia Pitoura. A survey on representation, composition and application of preferences in database systems. *ACM Trans. Database Syst.*, 36(3) :19 :1–19 :45, August 2011.

[SM06] Michael Schmitt and Laura Martignon. On the complexity of learning lexicographic strategies. *Journal of Machine Learning Research*, 7 :55–83, 2006.

[Val84] L. G. Valiant. A theory of the learnable. *Communications of the ACM, 27(11) :1134–1142*, 1984.

[Vin92] Ph. Vincke. Exploitation of a crisp relation in a ranking problem. *Theory and Decision*, 32(3) :221–240, 1992.

[WB11] Willem Waegeman and BernardDe Baets. A survey on roc-based ordinal regression. In *Preference Learning*, pages 127–154. Springer Berlin Heidelberg, 2011.

[Wik14] Wikipedia. Discounted cumulative gain, 8 Novembre 2014 2014.

[Wil04] Nic Wilson. Extending CP-Nets with stronger conditional preference statements. In *AAAI*, pages 735–741, 2004.

[Woo94] Douglas R. Woodall. Properties of preferential election rules. *Voting matters*, 3 :8–15, 1994.

[XL07] Jun Xu and Hang Li. Adarank : a boosting algorithm for information retrieval. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 391–398, New York, NY, USA, 2007. ACM.

[YWLd08] Fusun Yaman, Thomas J. Walsh, Michael L. Littman, and Marie desJardins. Democratic approximation of lexicographic preference models. In *Proceedings of the 25th international conference on Machine learning*, ICML '08, pages 1200–1207, New York, NY, USA, 2008. ACM.

[Zak00] Mohammed Javeed Zaki. Scalable algorithms for association mining. *IEEE Trans. Knowl. Data Eng.*, 12(3) :372–390, 2000.

[ZCY+12] Hengshu Zhu, Enhong Chen, Kuifei Yu, Huanhuan Cao, Hui Xiong, and Jilei Tian. Mining personal context-aware preferences for mobile users. In Mohammed Javeed Zaki, Arno Siebes, Jeffrey Xu Yu, Bart Goethals, Geoffrey I. Webb, and Xindong Wu, editors, *ICDM*, pages 1212–1217. IEEE Computer Society, 2012.

Résumé :

L'utilisation de préférences suscite un intérêt croissant pour personnaliser des réponses aux requêtes et effectuer des recommandations ciblées. Pourtant, la construction manuelle de profils de préférences reste à la fois complexe et consommatrice de temps. Dans ce contexte, nous présentons dans cette thèse une nouvelle méthode automatique d'extraction de préférences basée sur des techniques de fouille de données.

L'approche que nous proposons est constituée de deux phases : (1) une phase d'extraction de toutes les règles de préférences contextuelles intéressantes et (2) une phase de construction du profil utilisateur. A la fin de la première phase, nous constatons qu'il y a des règles redondantes voir superflues ; la seconde phase permet d'éliminer les règles superflues afin d'avoir un profil concis et consistant. Dans notre approche, un profil utilisateur est constitué de cet ensemble de règles de préférences contextuelles résultats de la seconde phase. La consistance garantit que les règles de préférences spécifiant les profils sont en accord avec un grand nombre de préférences utilisateur et contredisent un petit nombre d'entre elles. D'autre part, la concision implique que les profils sont constitués d'un petit nombre de règles de préférences. Nous avons aussi proposé quatre méthodes de prédiction qui utilisent les profils construits.

Nous avons validé notre approche sur une base de données de films construite à partir de MovieLens et IMDB. La base de données contient 3 881 films notés par 6 040 utilisateurs. Ces derniers ont attribué 800 156 notes. Les résultats de ces expériences démontrent que la concision des profils utilisateurs est contrôlée par le seuil d'accord minimal et que même avec une forte réduction du nombre de règles, les qualités de prédiction des profils restent à un niveau acceptable. En plus des expérimentations montrant la qualité de prédiction de notre approche, nous avons montré également que les performances de notre approche peuvent rivaliser avec les qualités de prédiction de certaines méthodes de l'état de l'art, en particulier SVMRank.

Mots clés : elicitation de préférences, règles de préférences contextuelles, extraction de profil utilisateur, fouille de données.

Abstract :

The use of preferences arouses a growing interest to personalize response to requests and making targeted recommandations. Nevertheless, manual construction of preferences profiles remains complex and time-consuming. In this context, we present in this thesis a new automatic method for preferences elicitation based on data mining techniques.

Our proposal is a two phase algorithm : (1) Extracting all contextual preferences rules from a set of user preferences and (2) Building user profile. At the end of the first phase, we notice that there is to much preference rules which satisfy the fixed constraints then in the second phase we eliminate the superfluous preferences rules. In our approach a user profile is constituted by the set of contextual preferences rules resulting of the second phase. A user profile must satisfy conciseness and soundness properties. The soundness property guarantees that the preference rules specifying the profiles are in agreement with a large set of the user preferences, and contradict a small number of them. On the other hand, conciseness implies that profiles are small sets of preference rules. We also proposed four predictions methods which use the extracted profiles.

We validated our approach on a set of real-world movie rating datasets built from MovieLens and IMDB. The whole movie rating database consists of 800,156 votes from 6,040 users about 3,881 movies. The results of these experiments demonstrates that the conciseness of user profiles is controlled by the minimal agreement threshold and that even with strong reduction, the soundness of the profile remains at an acceptable level. These experiment also show that predictive qualities of some of our ranking strategies outperform SVMRank in several situations.

Keywords : preference elicitation, contextual preference rule, user profile mining, data mining