Construction de profils de préférences contextuelles basée sur l'extraction de motifs séquentiels

Arnaud Giacometti, Dominique H. Li, Arnaud Soulet

Université de Tours, Laboratoire d'Informatique
3 place Jean Jaurès, 41000 Blois, FRANCE
{arnaud.giacometti, dominique.li, arnaud.soulet}@univ-tours.fr

Résumé. L'utilisation de préférences suscite un intérêt croissant pour personnaliser des réponses et effectuer des recommandations. En amont, l'étape essentielle est l'élicitation des préférences qui consiste à construire un profil de préférences en sollicitant le moins possible l'utilisateur. Dans cet article, nous présentons une méthode basée sur l'extraction de motifs séquentiels afin de générer des règles de préférences contextuelles à partir d'une base de paires de transactions. À partir de ces règles générées, qui ont une expressivité plus riche que celle des approches existantes, nous montrons comment construire et utiliser un profil modélisant les préférences de l'utilisateur. De plus, notre approche a l'avantage de bénéficier des nombreux algorithmes efficaces d'extraction de séquences fréquentes. L'évaluation de notre méthode sur des données réelles montre que les modèles de préférences construits permettent d'effectuer des recommandations justes à un utilisateur.

1 Introduction

Le besoin d'incorporer les préférences aux requêtes dans les technologies de l'information est un verrou crucial pour une grande variété d'applications allant de l'e-commerce aux moteurs de recherche personnalisés. Un utilisateur accédant à un système d'information peut avoir à reformuler plusieurs fois sa requête pour éliminer les résultats insatisfaisants et cheminer vers le résultat attendu. En particulier, cette expérience est très fréquente avec les recherches sur le Web en raison d'une abondance d'information et surtout, de l'hétérogénéité des utilisateurs. Une observation cruciale alors est que « différents utilisateurs considèreront comme pertinent des résultats différents » car leurs préférences divergent.

Cependant, la construction manuelle de modèles de préférences par l'utilisateur reste à la fois complexe et consommatrice de temps. De ce fait, l'apprentissage automatique de ses préférences, en s'appuyant sur les interactions entre lui et le système, joue un rôle critique dans de nombreuses applications. Ces interactions appelées, *feedbacks utilisateurs implicites*, ont une forme souvent rudimentaire indiquant si un utilisateur a réalisé une action particulière sur un objet (par exemple, le temps passé sur un objet). Ces feedbacks implicites sont souvent ambigus et d'une granularité peu fine mais ils sont plus faciles à obtenir en abondance dans un système réel. Dans cet article, nous faisons l'hypothèse que nous disposons d'un ensemble

d'objets et d'un ensemble de paires de préférences sur ces objets, c.-à-d., tel objet est préféré à tel autre. Notre objectif sera de construire un modèle de préférences à partir de ces données même si de telles préférences sont moins subtiles que des notes et peuvent parfois contenir des inconsistances.

Dans cet article, nous présentons une approche de construction de profils de préférences contextuelles basée sur l'extraction de motifs séquentiels, nommée Sprex (**S**equence-pattern-based **p**reference **r**ule **ex**traction). Cette approche se constitue de trois composants principaux : un *extracteur* de motifs séquentiels pour l'extraction des règles de préférence contextuelle, une *fonction de modélisation* qui permet de trier et de sélectionner un sous-ensemble de règles de préférence contextuelle afin de créer un profil, et une *fonction de préférence* qui permettent de prédire la préférence de l'utilisateur en utilisant le profil construit. L'approche Sprex étend l'expressivité des règles de préférence contextuelle par rapport à Agrawal et al. (2006). Auparavant, de telles règles permettaient seulement de préférer un item par rapport à un autre. Notre proposition permet de préférer un *ensemble* d'items par rapport à un autre. Ce gain en expressivité des règles permet de construire des profils enrichis plus proches des préférences de l'utilisateur. En outre, la réutilisation des outils d'extraction de motifs séquentiels fréquents permet à l'approche Sprex de générer les profils de préférences avec une très grande efficacité. L'étude expérimentale montre la rapidité de l'approche mais aussi son efficacité en terme de prédiction.

L'article est organisé de la manière suivante. La section 2 définit les notions préliminaires. La section 3 introduit les travaux antérieurs associés à notre problématique. La section 4 présente l'approche Sprex. Pour cela, nous définissons formellement la notion de *séquences de préférence* et ensuite introduisons globalement notre approche ; enfin, nous présentons également les méthodes Sprex-Build et Sprex-Predict qui composent l'approche Sprex. La section 5 rapporte les expérimentations que nous avons menées sur des jeux de données réels, avant de conclure et de présenter les perspectives associées dans la section 6.

2 Définitions préliminaires et problématique

Nous définissons les notions préliminaires liées à notre problématique et à notre approche dans cette section.

Soit $\mathcal{R} = \{R_1, R_2, \ldots, R_n\}$ un ensemble d'attributs. Pour chaque attribut $R_i \in \mathcal{R}$, nous notons le domaine de valeurs de R_i par $dom(R_i)$. Nous définissons $dom(\mathcal{R}) = dom(R_1) \times dom(R_2) \times \ldots \times dom(R_m)$ et appelons *transaction* chaque élément de $dom(\mathcal{R})$. Soit $\mathcal{I} = \bigcup_{R_i \in \mathcal{R}} dom(R_i)$ l'ensemble de toutes les valeurs d'attributs, chaque valeur $i \in \mathcal{I}$ est un *item*. Soit $T = \{i_1, i_2, \ldots, i_n\}$ une transaction, chaque sous-ensemble $I \subseteq T$ est un *itemset*. Une *base de transactions* est un ensemble de transactions, chacune associée à un identifiant unique. Une *paire de transactions*, notée $\langle T_1, T_2 \rangle$, est un vecteur de deux transactions tel que $\langle T_1, T_2 \rangle \neq \langle T_2, T_1 \rangle$.

Dans cet article, nous modélisons les *préférences* de l'utilisateur comme un ordre partiel $\succ$ sur les itemsets tel que $X \succ Y$ indique que l'utilisateur préfère l'itemset X à l'itemset Y.

Définition 1 (Règle de préférence contextuelle étendue)**.** *Une préférence contextuelle est une règle de la forme $C \rightarrow X \succ Y$ décrivant que l'utilisateur préfère l'itemset X à l'itemset Y si le contexte, représenté par l'itemset C, est observé.*

Cette définition étend la notion de règle de préférence contextuelle utilisée par Agrawal et al. (2006) et de Amo et al. (2012). En effet, la définition usuelle se limite à des itemsets de taille 1 pour X et Y. En d'autres termes, l'expressivité de nos règles de préférences contextuelles est plus forte.

Exemple 1. *La règle de préférence* $\{viande\} \rightarrow \{carotte\} \succ \{riz\}$ *exprime qu'un utilisateur préfère des carottes au riz pour accompagner de la viande, et la règle de préférence* $\{viande, samedi\} \rightarrow \{carotte, vin\} \succ \{riz, soda\}$ *précise que le samedi, cet utilisateur préfère accompagner sa viande avec des carottes et du vin plutôt que du riz et du soda.* □

Intuitivement deux transactions sont comparables suivant la règle $C \rightarrow X \succ Y$ si les deux contiennent C et si une seule des deux contient X et l'autre Y. De manière formelle, étant données une règle de préférence contextuelle $r = C \rightarrow X \succ Y$ et une paire de transactions $p = \langle T_1, T_2 \rangle$, si $C \subseteq T_1 \cap T_2$, $X \subseteq T_1 \setminus (T_1 \cap T_2)$ et $Y \subseteq T_2 \setminus (T_1 \cap T_2)$, alors on dit que p *supporte* r, noté $r \vdash^+ p$; en revanche, si $C \subseteq T_1 \cap T_2$, $X \subseteq T_2 \setminus (T_1 \cap T_2)$ et $Y \subseteq T_1 \setminus (T_1 \cap T_2)$ alors on dit que p *contredit* r, noté $r \vdash^- p$. Par exemple, soient deux règles $r_1 = \{a, c\} \rightarrow \{b\} \succ \{e, f\}$ et $r_1 = \{a, c\} \rightarrow \{d\} \succ \{b\}$, on a $r_1 \vdash^+ p$ et $r_2 \vdash^- p$ si la paire de transactions $p = \langle \{a, b, c\}, \{a, c, d, e, f\} \rangle$.

Une *préférence de l'utilisateur* est une paire de transactions $\langle T, U \rangle \in \mathcal{P}$ qui spécifie que l'utilisateur préfère T à U, également noté $T \succ U$. Soit $\mathcal{D}$ une base de transactions décrivant des objets (p. ex. des films), une *base de préférences de l'utilisateur* $\mathcal{P} \subseteq \mathcal{D} \times \mathcal{D}$ est un ensemble de paires de transactions correspondant à un échantillon des *préférences de l'utilisateur* sur les objets de $\mathcal{D}$. À partir d'une base de préférences utilisateurs $\mathcal{P}$, nous définissons le *support* d'une règle de préférence contextuelle r, noté $supp_{\mathcal{P}}(r)$, comme le nombre de paires de transactions qui supportent r ; nous définissions également la *confiance* de la règle r, notée $conf_{\mathcal{P}}(r)$, comme le ratio du nombre de paires de transactions qui supportent r sur le nombre total de paires de transactions qui supportent ou contredisent r. On a alors $supp_{\mathcal{P}}(r) = |\{p \in \mathcal{P} \mid r \vdash^+ p\}|$ et $conf_{\mathcal{P}}(r) = |\{p \in \mathcal{P} \mid r \vdash^+ p\}| \,/\, |\{p \in \mathcal{P} \mid (r \vdash^+ p) \vee (r \vdash^- p)\}|$.

Définition 2 (Règle de préférence contextuelle minimale)**.** *Une règle de préférence contextuelle* $r = C \rightarrow X \succ Y$ *est minimale par rapport à une base de préférences utilisateurs* $\mathcal{P}$ *si et seulement s'il n'existe aucune règle* $r' = C' \rightarrow X' \succ Y'$, $r \neq r'$, *telle que* $C' \subseteq C$, $X' \subseteq X$ *et* $Y' \subseteq Y$ *avec* $supp_{\mathcal{P}}(r) = supp_{\mathcal{P}}(r')$ *et* $conf_{\mathcal{P}}(r) = conf_{\mathcal{P}}(r')$.

Un *modèle de préférences* sur une base de préférences utilisateurs $\mathcal{P}$ est un ensemble trié de règles de préférences contextuelles minimales, noté $\mathcal{M}_{\mathcal{P}}$. Étant donnée une base de préférences d'un utilisateur, on dit qu'un modèle de préférence est un *profil de préférences* de cet utilisateur. Le problème de la *construction de profil d'un utilisateur* est ainsi de construire un modèle de préférences à partir d'une base de préférences de l'utilisateur.

3 Travaux antérieurs

Étant donnée une paire d'objets (ou de transactions dans notre formalisme), l'objectif d'un modèle de préférence est de prédire lequel sera préféré par l'utilisateur. Les méthodes d'apprentissage de préférences se distinguent par la nature des préférences (qualitative ou quantitative) et le modèle sémantique sous-jacent (p. ex. les modèles de Pareto et les modèles de préférences conditionnelles/contextuelles).

Les approches d'élicitation de préférences *quantitatives* visent à associer un score à chaque objet afin d'ordonner un ensemble d'objets comme dans l'approche d'Agrawal et Wimmers (2000) par exemple. Pour cette raison, ces méthodes quantitatives qui établissent un classement sont aussi appelées « learning to rank ». Plusieurs méthodes efficaces pour apprendre un classement ont été proposées notamment dans le domaine de la recherche d'information. Par exemple, Joachims (2006) a proposé un cadre d'apprentissage de fonctions de score en exploitant une approche type Support Vector Machine (SVM), ce qui a donné lieu à l'algorithme SVMRank. Les algorithmes RankBoost proposé par Freund et al. (2003) et AdaRank proposé par Xu et Li (2007) optimisent le classement en s'appuyant sur une technique de *boosting* lors de l'apprentissage automatique. En effet, les méthodes quantitatives type SVMRank produisent le plus souvent des vecteurs numériques peu compréhensibles pour un utilisateur final et l'utilisateur ne peut ni interpréter le profil construit, ni le modifier. De plus, les prédictions effectuées ne peuvent être expliquées simplement. Sprex peut fournir des règles intelligibles pour justifier la prédiction qu'il a effectuée.

Les approches *qualitatives* ont pour objectif de produire un profil qui permettra déterminer une préférence entre deux objets. En se basant sur le modèle de préférences de Pareto, Kießling (2002); Holland et al. (2003) ont proposés des approches pour extraire les préférences utilisateurs. Plus récemment, introduit par Jiang et al. (2008), les exemples de préférences sont catégorisés en deux classes : les exemples supérieurs (ceux préférés par l'utilisateur) et les exemples inférieurs (ceux non-préférés par l'utilisateur). A partir de ces informations, un ordre sur toute paire de tuples est inféré et peut donc permettre de comparer deux nouveaux tuples.

Historiquement, les premières approches d'élicitation de préférences contextuelles (conditionnelles) ont été proposées dans le domaine de l'intelligence artificielle. Par exemple, les **CP-Nets** (réseaux de préférences conditionnelles) proposés par Boutilier et al. (2004) permettent à l'utilisateur de modéliser ses préférences contextuelles sous la forme d'un réseau assez compact. A l'inverse, notre méthode cherche à traiter des grands volumes de données contenant des incohérences car issus de feedbacks implicites.

Notre travail exploitent clairement des préférences contextuelles proches de celles proposées par Agrawal et al. (2006) pour classer des tuples ou par Stefanidis et al. (2007) où le contexte est modélisé comme un ensemble d'attributs multidimensionnels. Toutefois, Agrawal et al. (2006); Stefanidis et al. (2007) ne présentent pas de méthodes d'extraction de préférences à partir de données. Enfin, dans le travail récent présenté par de Amo et al. (2012), les préférences contextuelles sont formalisées et extraites en adaptant le cadre des règles d'association afin de construire les profils utilisateurs. Notre définition de règles de préférences contextuelle est plus générale et s'appuie sur l'extraction de motifs séquentiels.

4 L'approche Sprex

Dans cette section, nous présentons notre approche Sprex. Dans un premier temps, nous définissons formellement la correspondance entre préférences et motifs séquentiels. L'extraction de règles de préférences contextuelles, dans ce contexte, consiste alors à extraire les séquences de préférence fréquentes. Nous présentons ensuite les deux principales phases de l'approche Sprex : Sprex-Build et Sprex-Predict.

4.1 Représentation séquentielle des préférences

Soit $\mathcal{I}$ l'ensemble de tous les items, un *item de préférence* est une paire $L{:}i$ où $L \in \{\texttt{C}, \texttt{P}, \texttt{N}\}$ est un *label* (parmi **C**ontexte, **P**référé et **N**on-préféré) et $i \in \mathcal{I}$ est un item. Étant données deux transactions T et U telles que $T \succ U$, l'*itemset contextuel* est un ensemble d'items de préférence $\{\texttt{C}{:}c_1, \texttt{C}{:}c_2, \ldots, \texttt{C}{:}c_k\}$ où $\{c_1, c_2, \ldots, c_k\} \equiv T \cap U$; l'*itemset préféré* est l'ensemble $\{\texttt{P}{:}x_1, \texttt{P}{:}x_2, \ldots, \texttt{P}{:}x_m\}$ où $\{x_1, x_2, \ldots, x_m\} \equiv T \setminus (T \cap U)$; l'*itemset non-préféré* est l'ensemble $\{\texttt{N}{:}y_1, \texttt{N}{:}y_2, \ldots, \texttt{N}{:}y_n\}$ où $\{y_1, y_2, \ldots, y_n\} \equiv U \setminus (T \cap U)$. Les itemsets contextuel, préféré et non-préféré sont appelés les *itemsets de préférence*. Nous définissons trois *fonctions de label* $\{\lambda_C, \lambda_P, \lambda_N\}$. Chacune génère un itemset de préférence à partir d'un itemset usuel : par exemple, si $I = \{i_1, i_2, \ldots, i_n\}$, alors $\lambda_C(I) = \{\texttt{C}{:}i_1, \texttt{C}{:}i_2, \ldots, \texttt{C}{:}i_n\}$, etc.

De manière plus simple, on note $C_C = \lambda_C(C)$, $X_P = \lambda_P(X)$ et $Y_N = \lambda_N(Y)$. Nous appelons alors la séquence $\langle C_C P_P N_N \rangle$ une *séquence de préférence*, il s'agit d'une liste ordonnée des itemsets de préférence sur les itemsets C, P et N qui indique la préférence de la transaction $T = C \cup P$ sur la transaction $U = C \cup N$. Étant donnée une base de préférences de l'utilisateur $\mathcal{P}$, il existe une base de séquences de préférence, notée $\mathcal{P}_S$, dont chaque séquence (associée à un identificateur unique) correspond à une préférence unique $p \in \mathcal{P}$.

Exemple 2. *Soient $T_1 = \{a, b, c, d, e, f\}$ et $T_2 = \{a, g, h, f\}$ deux transactions. Selon la préférence de l'utilisateur $T_1 \succ T_2$, on a l'itemset contextuel $\{\texttt{C}{:}a, \texttt{C}{:}f\}$, l'itemset préféré $\{\texttt{P}{:}b, \texttt{P}{:}c, \texttt{P}{:}d, \texttt{P}{:}e\}$ et l'itemset non-préféré $\{\texttt{N}{:}g, \texttt{N}{:}h\}$. La séquence de préférence construite à partir de cette préférence utilisateur est alors $\langle \{\texttt{C}{:}a, \texttt{C}{:}f\}\{\texttt{P}{:}b, \texttt{P}{:}c, \texttt{P}{:}d, \texttt{P}{:}e\}\{\texttt{N}{:}g, \texttt{N}{:}h\} \rangle$.* □

Selon le principe de l'extraction de motifs séquentiels proposée par Agrawal et Srikant (1995), nous définissons les relations suivantes sur les séquences de préférence. Etant donnée deux séquences de préférence $s = \langle I_C I_P I_N \rangle$ (où I_C, I_P, et I_N sont respectivement les itemsets contextuel, préféré et non-préféré) et $s' = \langle I'_C I'_P I'_N \rangle$, si $I_C \subseteq I'_C$, $I_P \subseteq I'_P$, et $I_N \subseteq I'_N$, alors s est une *sous-séquence* de s', dénoté par $s \sqsubseteq s'$. Etant donnée une base de séquences de préférence $\mathcal{P}_S$, le *support* de la séquence de préférence s, dénoté par $supp_{\mathcal{P}_S}(s)$, est le nombre de séquences de préférence de $\mathcal{P}_S$ qui contiennent s : $supp_{\mathcal{P}_S}(s) = |\{s' \in \mathcal{P}_S \mid s \sqsubseteq s'\}|$. Etant donné un seuil minimal de support σ, une séquence s est alors dite *fréquente* si $supp_{\mathcal{P}_S}(s) \geq \sigma$, on parle de *séquence fréquente de préférence*.

Soit $r = C \rightarrow X \succ Y$ une règle de préférence contextuelle, où $C = \{c_1, c_2, \ldots, c_k\}$, $X = \{x_1, x_2, \ldots, x_m\}$ et $Y = \{y_1, y_2, \ldots, y_n\}$, alors la règle r peut être réécrite en une séquence de préférence $\langle C_C X_P Y_N \rangle$.

Propriété 1. *Avec la paire de transactions $\langle T, U \rangle$ où $T \succ U$ représentée sous la forme d'une séquence de préférence $\langle I_C I_P I_N \rangle$, on obtient les relations suivantes :*

$$C \rightarrow X \succ Y \vdash^{+} \langle T, U \rangle \iff \langle C_C X_P Y_N \rangle \sqsubseteq \langle I_C I_P I_N \rangle\,, \tag{1}$$

$$C \rightarrow X \succ Y \vdash^{-} \langle T, U \rangle \iff \langle C_C Y_P X_N \rangle \sqsubseteq \langle I_C I_P I_N \rangle\,. \tag{2}$$

■

Ainsi, étant donnée une base de séquences de préférence $\mathcal{P}_S$ construite à partir des préférences utilisateurs $\mathcal{P}$, alors le support de la règle $r = C \rightarrow X \succ Y$ par rapport à $\mathcal{P}$ est équivalent au support de la séquence de préférence $\langle C_C X_P Y_N \rangle$ par rapport à $\mathcal{P}_S$. De plus, le nombre de paires de transactions de $\mathcal{P}$ qui contredisent la règle r est équivalent au nombre de séquences de préférence de $\mathcal{P}_S$ qui contiennent la séquence $\langle C_C Y_P X_N \rangle$.

4.2 Phase de construction : Sprex-Build

La construction du modèle de préférences Sprex-Build est décrite par l'algorithme 1 où un modèle de préférence $\mathcal{M}_\mathcal{P}$ (c.-à-d., le profil de l'utilisateur) est généré à partir d'une base de préférences utilisateurs $\mathcal{P}$ avec un seuil minimal de support σ, un seuil minimal de confiance δ, et une fonction de modélisation π. Sprex-Build génère d'abord une base de séquences de préférence $\mathcal{P}_S$ à partir des préférences utilisateurs $\mathcal{P}$ en utilisant la correspondance vue dans la section précédente (ligne 1 à 4). Ensuite, à la ligne 5, la collection des séquences fréquentes de préférence $\mathcal{F}$ est extraite depuis $\mathcal{P}_S$ (selon un seuil σ) en utilisant n'importe quel algorithme d'extraction de motifs séquentiels (dans notre partie expérimentale, nous utiliserons l'approche PatternGrowth). Pour chaque séquence fréquente de préférence s de l'ensemble $\mathcal{F}$, Sprex-Build calcule sa confiance $conf_{\mathcal{P}_S}(s)$ et l'ajoute au modèle $\mathcal{M}$ si $conf_{\mathcal{P}_S}(s) \geq \delta$. Enfin, une fonction de modélisation π définie par l'utilisateur est utilisée pour construire le modèle de préférence final.

Algorithme 1 : Sprex-Build

Entrées : Les préférences de l'utilisateur $\mathcal{P}$, un seuil minimal de support σ, un seuil de confiance minimal δ, et une fonction de modélisation π.
Sorties : Modèle de règles de préférences contextuelles $\mathcal{M}_\mathcal{P}$.

1 $\mathcal{P}_S = \emptyset$;
2 **pour chaque** $\langle T, U\rangle \in \mathcal{P}$ **faire**
3 $\quad C_C = \lambda_C(T \cap U)$, $P_P = \lambda_P(T \setminus T \cap U)$, $N_N = \lambda_N(U \setminus T \cap U)$;
4 $\quad \mathcal{P}_S = \mathcal{P}_S \cup \langle C_C P_P N_N\rangle$;
5 $\mathcal{F} = FrequentSequenceMining(\mathcal{P}_S, \sigma)$;
6 $\mathcal{M} = \emptyset$;
7 **pour chaque** $s \in \mathcal{F}$ **faire**
8 $\quad$ **si** $conf_{\mathcal{P}_S}(s) \geq \delta$ **alors**
9 $\quad\quad \mathcal{M} = \mathcal{M} \cup s$;
10 $\mathcal{M}_\mathcal{P} = \pi(sort(\mathcal{M}))$;
11 **retourner** $\mathcal{M}_\mathcal{P}$;

Comme précédemment décrit, étant donné une paire de transactions $\langle T, U\rangle$, une séquence de préférence $\langle I_C I_P I_N\rangle$ doit contenir au moins 2 itemsets de préférence et un maximum de 3 itemsets de préférence. Plus précisément, selon la structure d'une séquence de préférence générée à partir d'une base de transactions, il y a 7 cas sur les motifs séquentiels extraits et tous ces cas à l'exception de 1 permettent de construire une règle de préférence. L'exemple suivant illsutre cette propriété.

Exemple 3. *Considère des itemsets de préférence qui contiennent deux items* $\{\mathtt{C}{:}i_1, \mathtt{C}{:}i_2\}$, $\{\mathtt{P}{:}i_3, \mathtt{P}{:}i_4\}$ *et* $\{\mathtt{N}{:}i_5, \mathtt{N}{:}i_6\}$*, on a les correspondances suivantes entre les motifs séquentiels et les règles de préférence :*

1. $\langle\{\mathtt{C}{:}i_1, \mathtt{C}{:}i_2\}\{\mathtt{P}{:}i_3, \mathtt{P}{:}i_4\}\{\mathtt{N}{:}i_5, \mathtt{N}{:}i_6\}\rangle \iff \{\mathtt{C}{:}i_1, \mathtt{C}{:}i_2\} \rightarrow \{\mathtt{P}{:}i_3, \mathtt{P}{:}i_4\} \succ \{\mathtt{N}{:}i_5, \mathtt{N}{:}i_6\}$;
2. $\langle\{\mathtt{C}{:}i_1, \mathtt{C}{:}i_2\}\{\mathtt{P}{:}i_3, \mathtt{P}{:}i_4\}\rangle \iff \{\mathtt{C}{:}i_1, \mathtt{C}{:}i_2\} \rightarrow \{\mathtt{P}{:}i_3, \mathtt{P}{:}i_4\} \succ \{\}$;
3. $\langle\{\mathtt{C}{:}i_1, \mathtt{C}{:}i_2\}\{\mathtt{N}{:}i_5, \mathtt{N}{:}i_6\}\rangle \iff \{\mathtt{C}{:}i_1, \mathtt{C}{:}i_2\} \rightarrow \{\} \succ \{\mathtt{N}{:}i_5, \mathtt{N}{:}i_6\}$;

4. $\langle\{\texttt{P}{:}i_3, \texttt{P}{:}i_4\}\{\texttt{N}{:}i_5, \texttt{N}{:}i_6\}\rangle \iff \{\} \rightarrow \{\texttt{P}{:}i_3, \texttt{P}{:}i_4\} \succ \{\texttt{N}{:}i_5, \texttt{N}{:}i_6\}$;
5. $\langle\{\texttt{P}{:}i_3, \texttt{P}{:}i_4\}\rangle \iff \{\} \rightarrow \{\texttt{P}{:}i_3, \texttt{P}{:}i_4\} \succ \{\}$;
6. $\langle\{\texttt{N}{:}i_5, \texttt{N}{:}i_6\}\rangle \iff \{\} \rightarrow \{\} \succ \{\texttt{N}{:}i_5, \texttt{N}{:}i_6\}$;
7. *par contre, le motif séquentiel* $\langle\{\texttt{C}{:}i_1, \texttt{C}{:}i_2\}\rangle$ *sera systématiquement ignoré car il ne sert à aucune règle pertinente.*

□

Pour cette raison, l'extraction des séquences de préférence fréquentes est plus simple que l'extraction de tous les motifs séquentiels (Agrawal et Srikant, 1995; Pei et al., 2001). De plus, comme un modèle de préférence est constitué uniquement de règles de préférences minimales, le problème de construction se réduit même à l'extraction des séquences génératrices fréquentes introduites par Lo et al. (2008).

La méthode Sprex-Build utilise ensuite l'ensemble des motifs séquentiels extraits afin de générer un modèle (ligne 10) via une fonction de modélisation. Avant d'appliquer une fonction de modélisation, nous conservons pour chaque paire de transactions la *meilleure* règle qui la supporte. Cette notion de « meilleure » est modélisée par un ordre sur les règles déjà utilisé par Liu et al. (1998) pour la construction de classifieur. Ainsi, une règle est d'autant meilleure que sa confiance est élevée ; en cas d'égalité, on choisit alors celle dont le support est le plus grand. La fonction de modélisation effectue ensuite deux opérations : 1) sélection de règles de préférence contextuelle en respectant des contraintes définies par l'utilisateur (p. ex. le filtrage basé sur la confiance, le support, la composition, la structure ou la taille des règles) ; et 2) nouveau tri des règles sélectionnées en respectant un ordre défini par l'utilisateur en cas de besoin. Il s'agit donc de conserver des règles sûres et générales pour que le modèle final soit à la fois précis et concis.

4.3 Phase de prédiction : Sprex-Predict

L'utilisation par Sprex-Predict du modèle construit lors de la phase précédente pour effectuer une prédiction entre deux préférences est détaillée en dessous.

Étant donné un modèle $\mathcal{M}$, une fonction de préférence ρ retourne un score c entre 0 et 1 qui prédit entre les transactions T et U, celle qui est préférée par l'utilisateur en se référant au modèle $\mathcal{M}$. Si $c > 0.5$, cela signifie que l'utilisateur préfère la transaction T à la transaction U et ainsi Sprex-Predict retourne la préférence $T \succ_{\mathcal{M}} U$; si $c < 0.5$, Sprex-Predict retourne la préférence $U \succ_{\mathcal{M}} T$ car l'utilisateur préfère la transaction U à la transaction T ; sinon, Sprex-Predict retourne $T \sim_{\mathcal{M}} U$, qui signifie l'indécision de Sprex-Predict quant à la préférence de l'utilisateur entre T et U.

La fonction de préférence utilise les règles du modèle $\mathcal{M}$ afin de déterminer quelle est la transaction préférée entre T et U. Une fonction simple de préférence est d'utiliser la *meilleure* règle du profil r_{best} qui permet de préférer T à U ou vice-et-versa. Si T est préférée à U selon r_{best}, la fonction de préférence ρ_{best} retournera $conf_{\mathcal{P}}(r_{best})$ (une valeur > 0.5). À l'inverse, si U est préférée à T selon r_{best}, la fonction de préférence ρ_{best} retournera $1 - conf_{\mathcal{P}}(r_b)$ (une valeur < 0.5). Si aucune règle du profil ne s'applique, la fonction ρ_{best} retournera 0.5. À nouveau, nous utilisons un ordre défini sur la meilleure confiance, puis sur le meilleur support comme lors de la construction du modèle afin de choisir la meilleure règle.

Malheureusement, la fonction de préférence ρ_{best} est souvent indécise car il est peu fréquent que deux transactions soient directement comparables par une règle du modèle, selon l'étude expérimentale menée dans de Amo et al. (2012). Au lieu de comparer directement les deux transactions avec ρ_{best}, nous organisons un vote par valeur pour prendre la décision « qui est le meilleur candidat ? ». Notre vote par valeur est un système de vote pour une élection à un siège dans lequel les électeurs $\mathcal{E}$ évalue les deux candidats en leur attribuant une valeur entre 0 et 1 en utilisant ρ_{best}. Les valeurs de chaque candidat sont additionnées, et celui ayant le plus haut score est le gagnant. Formellement, nous avons :

$$\rho_{vote} = \begin{cases} 1, & \text{si } \sum_{V\in\mathcal{E}} \rho_{best}(\mathcal{M}, \langle T, V\rangle) > \sum_{V\in\mathcal{E}} \rho_{best}(\mathcal{M}, \langle U, V\rangle) \\ 0, & \text{si } \sum_{V\in\mathcal{E}} \rho_{best}(\mathcal{M}, \langle T, V\rangle) < \sum_{V\in\mathcal{E}} \rho_{best}(\mathcal{M}, \langle U, V\rangle) \\ 0.5, & \text{si } \sum_{V\in\mathcal{E}} \rho_{best}(\mathcal{M}, \langle T, V\rangle) = \sum_{V\in\mathcal{E}} \rho_{best}(\mathcal{M}, \langle U, V\rangle) \end{cases} . \quad (3)$$

Dans les expérimentations, nous utiliserons exclusivement ρ_{vote} dont le rappel est toujours supérieur à ρ_{best} pour une précision comparable.

5 Évaluations expérimentales

Dans cette section, nous rapportons les évaluations expérimentales de notre approche Sprex sur les jeux de données réels [1] construits à partir des données disponibles sur MovieLens [2] et IMDB [3]. Nous comparons Sprex à l'approche SVMRank développée par Joachims (2006) car il s'agit d'une des méthodes les plus performantes et des plus reconnues. De plus, la méthode proposée par de Amo et al. (2012) utilise la même référence.

Toutes nos expérimentations ont été effectuées sur un serveur de 16-Core 2.40GHz Intel Xeon avec 32 giga-octets de mémoire vive et avec le noyau Linux 2.6.32. L'extraction de motifs séquentiels est réalisée en utilisant le principe PatternGrowth proposé par Pei et al. (2001). Toutes les méthodes sont implémentées en C++ Template et compilées par LLVM-Clang++.

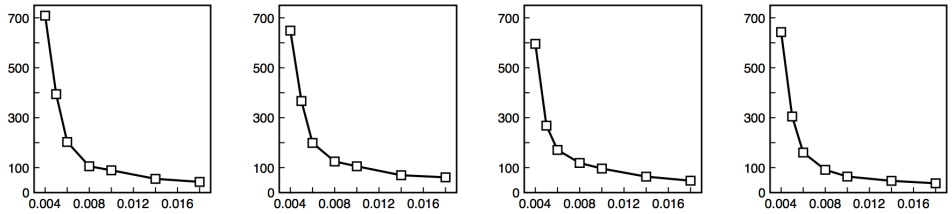
Le jeu de données concerné dans nos expérimentations est constitué des 800 156 notes attribuées par 6 040 utilisateurs sur 3 881 films. Chaque film constitue un ensemble d'attributs incluant un identifiant unique, l'année de sortie, les genres (multi-valeurs), les réalisateurs (multi-valeurs) et les principaux acteurs/actrices (multi-valeurs). Chaque utilisateur a évalué un certain nombre de films (de quelques uns à quelques milliers) avec des notes comprises entre 1 et 5 [4]. Tous les films votés par un même utilisateur composent un jeu de données indépendant. Alors, soit $\mathcal{D}$ un jeu de films votés par un utilisateur, pour toute paire de films $(m_1, m_2) \in \mathcal{D}$, on a $m_1.rating > m2.rating$ impose $m_1 \succ m_2$. Ainsi, chaque jeu de films correspond à une base de préférences de l'utilisateur. La table 1 liste les 4 jeux de données testés dans nos expérimentations où chaque jeu est identifié par l'identifiant de l'utilisateur et contient plus de 500 films votés afin d'assurer une validation croisée en 5 blocs telle que chaque bloc de données contient du moins 100 films votés.

Dans une première étape, les motifs séquentiels sont extraits avec le support minimal variant entre 0,4% et 1,8% pour créer des *profils bruts* qui contiennent les règles de préférence

1. Disponible sur `http://www.info.univ-tours.fr/~li/data/pref.html`
2. `http://www.movielens.org`
3. `http://www.imdb.com`
4. Nous avons délibérément choisi un jeu de données avec des feedbacks explicites plutôt que implicites pour pouvoir nous comparer avec SVMRank.

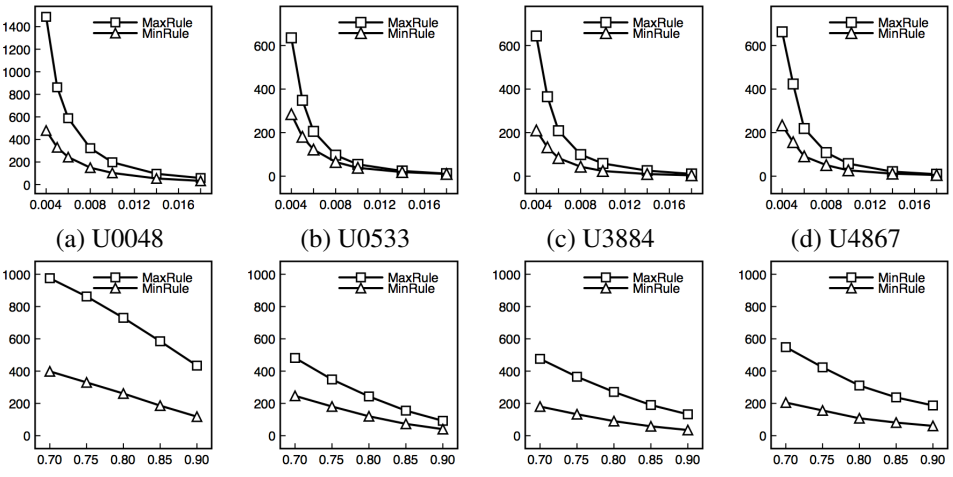
Base	Films votés	Séquences de préférence	Items distincts
U0048	541	93365	2514
U0533	565	122757	2602
U3884	571	124974	2797
U4867	552	97860	2674

TAB. 1 – *Jeux de données réels sur les films votés.*



(a) U0048 (b) U0533 (c) U3884 (d) U4867

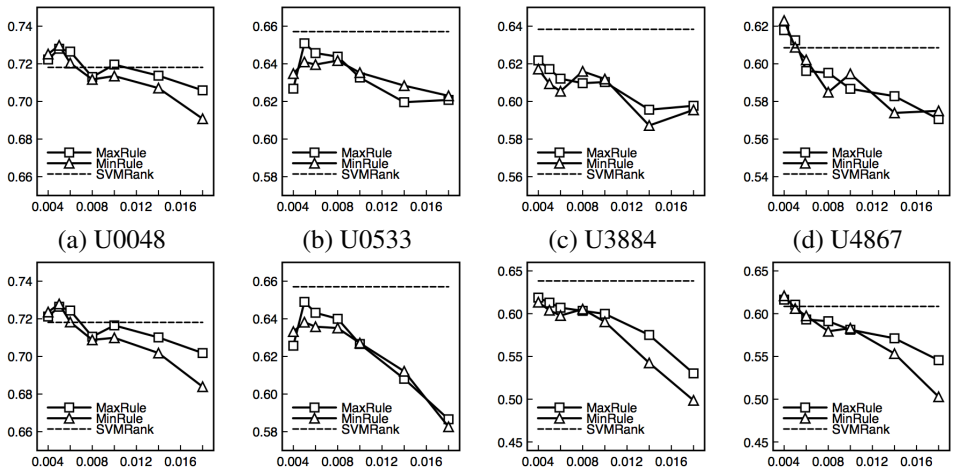
FIG. 1 – *Les temps moyens (en seconde) de construction de profils bruts par rapport à la variation du support minimal.*



(a) U0048 (b) U0533 (c) U3884 (d) U4867

(e) U0048 (f) U0533 (g) U3884 (h) U4867

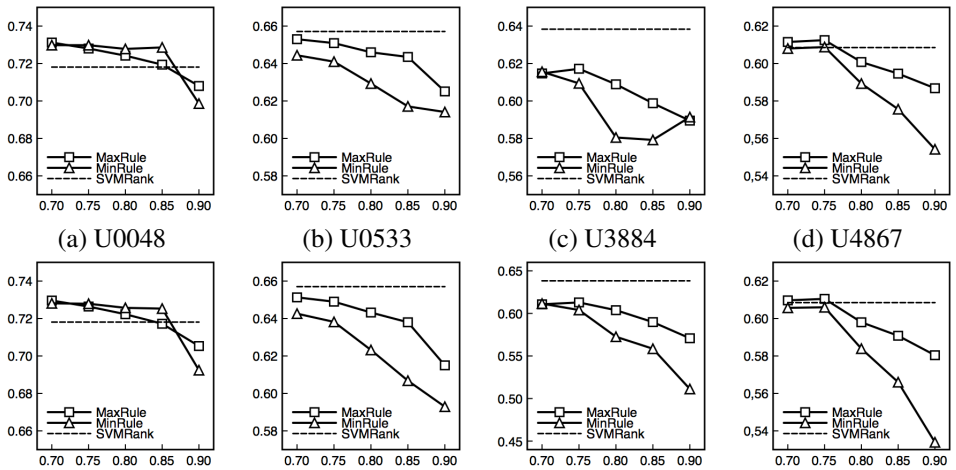
FIG. 2 – *Les tailles moyennes de profils par rapport à la variation du support minimal avec la confiance minimale fixée à 75% (a–d) et à la variation de la confiance minimale avec le support minimal fixé à 0,5% (e–h).*



(a) U0048 (b) U0533 (c) U3884 (d) U4867

(e) U0048 (f) U0533 (g) U3884 (h) U4867

FIG. 3 – *Les précisions moyennes (a–d) et les rappels moyens (e–h) de profils par rapport à la variation du support minimal avec la confiance minimale fixée à 75%.*



(a) U0048 (b) U0533 (c) U3884 (d) U4867

(e) U0048 (f) U0533 (g) U3884 (h) U4867

FIG. 4 – *Les précisions moyennes (a–d) et les rappels moyens (e–h) de profils par rapport à la variation de la confiance minimale avec le support minimal fixé à 0,5%.*

contextuelle minimales avec la confiance minimale fixée à 50% afin de faciliter les tests suivants. Ensuite, pour chaque profil brut, deux fonctions de modélisation, MaxRule et MinRule, sont appliquées pour construire des profils finaux. La fonction MaxRule permet de construire des profils en sélectionnant les règles les plus spécifiques parmi toutes les règles minimales générées précédemment, par contre la fonction MaxRule construit un profil en utilisant les règles moins expressives comme introduites dans les approches proposées par Agrawal et al. (2006) et par de Amo et al. (2012). Dans cette étape, la confiance minimale a varié entre 70% et 90%. Enfin, les résultats obtenus par SVMRank sont utilisés comme référence.

La figure 1 montre les temps moyens de construction de profils bruts à partir des 4 jeux de films. En effet, avec le support minimal 0,4%, les constructions se terminent en 700 secondes dont les nombres moyens de règles par profil sont respectivement 20 934, 13 208, 12 020 et 17 699. Après avoir appliqué la fonction de modélisation, la taille des profils est significativement réduite comme le montre la figure 2.

Nous avons testé la qualité de prédiction de chaque profil construit en mesurant la précision et le rappel de la prédiction. Pour cet objectif, deux séries de tests ont été effectuées en variant le support minimal (la figure 3) et en variant la confiance minimale (la figure 4). Les courbes montrent que la qualité de la prédiction augmente avec l'expressivité des règles qui composent les profils. Par ailleurs, les courbes montrent également qu'il y a une corrélation forte entre la qualité de prédiction et la diminution des seuils de support et de confiance. La table 2 présente un exemple des règles de préférence contextuelle extraites à partir du jeu de films U0048.

Contexte	**Préféré**	**Non-préféré**	**Confiance**	**Support**
{Action}	{Fantasy, Harrison_Ford}	{1990s}	0.987578	11222
{Comedy}	{Romance}	{Family, Sport}	0.968254	1487
{1990s}	{Thriller, Drama}	{Comedy, Adventure}	0.955490	1620
{1990s}	{Thriller, Tommy_Lee_Jones}	{Comedy}	0.922667	1639

TAB. 2 – *Un exemple des règles de préférence contextuelle.*

6 Conclusions

Dans cet article, nous avons présenté Sprex, une approche basée sur l'extraction de séquences fréquentes pour la construction de modèles de préférences. L'idée essentielle est de représenter les données d'apprentissage et les règles de préférence utilisateur sous la forme de triplets formant des séquences de préférence. De cette manière, nous avons pu tirer profits des travaux issus de l'extraction de motifs séquentiels fréquents et mettre en place une approche efficace. Par ailleurs, cette proposition a également l'avantage d'étendre l'expressivité des préférences extraites. Nous avons évalué notre approche sur une base de données filmographique du monde réel, et les résultats expérimentaux ont montré que les modèles de préférence construits sont relativement stables par rapport à la variation du seuil minimum de support.

Nos recherches futures vont s'orienter vers le développement de fonctions de modélisation et des fonctions de préférence afin d'améliorer la précision et le rappel de la prédiction des préférences de l'utilisateur. Enfin, nous pensons exploiter le potentiel des séquences au sein de l'approche Sprex pour introduire la notion de temps dans les règles de préférences.

Références

Agrawal, R., R. Rantzau, et E. Terzi (2006). Context-sensitive ranking. In *SIGMOD*, pp. 383–394.

Agrawal, R. et R. Srikant (1995). Mining sequential patterns. In *ICDE*, pp. 3–14.

Agrawal, R. et E. L. Wimmers (2000). A framework for expressing and combining preferences. In *SIGMOD*, pp. 297–306.

Boutilier, C., R. I. Brafman, C. Domshlak, H. H. Hoos, et D. Poole (2004). Cp-nets : A tool for representing and reasoning with conditional ceteris paribus preference statements. *Journal of Artificial Intelligence Research 21*, 135–191.

de Amo, S., M. S. Diallo, C. T. Diop, A. Giacometti, H. D. Li, et A. Soulet (2012). Mining contextual preference rules for building user profiles. In *DaWaK*, pp. 229–242.

Freund, Y., R. D. Iyer, R. E. Schapire, et Y. Singer (2003). An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research 4*, 933–969.

Holland, S., M. Ester, et W. Kießling (2003). Preference mining : A novel approach on mining user preferences for personalized applications. In *PKDD*, pp. 204–216.

Jiang, B., J. Pei, X. Lin, D. W. Cheung, et J. Han (2008). Mining preferences from superior and inferior examples. In *KDD*, pp. 390–398.

Joachims, T. (2006). Training linear SVMs in linear time. In *KDD*, pp. 217–226.

Kießling, W. (2002). Foundations of preferences in database systems. In *VLDB*, pp. 311–322.

Liu, B., W. Hsu, et Y. Ma (1998). Integrating classification and association rule mining. In *KDD*, pp. 80–86.

Lo, D., S.-C. Khoo, et J. Li (2008). Mining and ranking generators of sequential patterns. In *SDM*, pp. 553–564.

Pei, J., J. Han, B. Mortazavi-Asl, et H. Pinto (2001). PrefixSpan : Mining sequential patterns efficiently by prefix-projected pattern growth. In *ICDE*, pp. 215–224.

Stefanidis, K., E. Pitoura, et P. Vassiliadis (2007). Adding context to preferences. In *ICDE*, pp. 846–855.

Xu, J. et H. Li (2007). Adarank : A boosting algorithm for information retrieval. In *SIGIR*, pp. 391–398.

Summary

The use of preferences brings an increasing interest for personalizing the answers of queries and for performing recommendation, where the essential step is to automatically build user preference profiles from data. In this paper, we present a sequential pattern mining based method for generating contextual preference rules in order to construct a profile that models user preferences. Indeed, our approach allows to use any frequent sequence mining algorithm to to generate preference rules that contain much rich information than existing methods. The experimental results on real-world datasets show the performance and effectiveness of our approach.