Pour une meilleure exploitation de la classification croisée dans les systèmes de filtrage collaboratif

Aghiles Salah, Nicoleta Rogovschi, François Role, Mohamed Nadif

LIPADE, Université Paris Descartes, France
45, rue des Saints Pères,
75006, Paris
Prénom.Nom@parisdescartes.fr

Résumé. Pour la prédiction automatique des items préférés par des utilisateurs sur le Web, différents systèmes de filtrage collaboratif ont été proposés. La plupart d'entre eux sont basés sur la factorisation matricielle et les approches de type k plus proches voisins. Malheureusement ces deux approches requièrent un temps de calcul important. Une partie de ces problèmes a pu être surmontée par la classification croisée ou *co-clustering* qui s'avère pertinente du fait qu'elle permet par nature une gestion simultanée des ensembles correspondant aux utilisateurs et aux items. Cependant, des travaux doivent encore être menés pour une meilleure prise en compte des données manquantes. Dans ce travail, nous proposons donc une gestion efficace des données non observées permettant une meilleure exploitation du potentiel de la classification croisée dans le domaine des systèmes de recommandation. Nous montrons de plus qu'elle permet d'obtenir des représentations à base de graphes bipartis facilitant l'interprétation interactive des affinités entre des groupes d'utilisateurs et des groupe d'items.

1 Introduction

L'objectif des systèmes de recommandation est de prédire les choix et les préférences individuelles en fonction des comportements et des préférences observées. Le filtrage collaboratif est la technique la plus utilisée par les systèmes de recommandation. Il consiste à comparer les données d'un utilisateur avec des données similaires d'autres utilisateurs, basée sur les habitudes d'achat et de navigation (Goldberg et al., 1992). Il permet aux commerçants de fournir des recommendations aux clients pour de futurs achats. Dans la suite, les données sont représentées par une matrice $\mathbf{U}$ de taille $(n \times p)$ où chaque ligne représente un utilisateur, les colonnes représentent des items, et chaque cellule (u_{ij}) de $\mathbf{U}$ est la note attribuée par un utilisateur i pour un item j. Les notes (u_{ij}) peuvent être binaires, ou réelles et dans ce cas $\mathbf{U}$ est appelée matrice réelle de notations. La matrice $\mathbf{U}$ peut être obtenue de manière explicite (en gardant les évaluations fournies par les utilisateurs pour des articles donnés) ou de manière implicite (en considérant qu'un utilisateur préfère implicitement acheter ou pas les éléments présentés sur des pages Web visitées).

Dans le filtrage collaboratif (désormais désigné par FC), plusieurs approches sont utilisées. Les techniques de FC actuelles telles que celles basées sur la corrélation entre utilisateurs

(Bobadilla et al., 2013) ou sur la factorisation matricielle (Koren, 2009; Sarwar et al., 2000; Delporte et al., 2014) sont couramment utilisées, mais nécessitent un temps de calcul très coûteux et ne peuvent être déployées en ligne. Dans ce contexte, la classification croisée ou *co-clustering*, qui consiste à regrouper simultanément les utilisateurs et les items, est une bonne solution. Elle est particulièrement appropriée dans les systèmes de de recommendation. Il est ainsi, par exemple, intéressant de disposer de groupes d'utilisateurs appréciant un groupe de films. Dans (George et Merugu, 2005), les auteurs ont proposé une approche de FC basé sur un algorithme de classification croisée pondérée (COCLUST) qui implique le regroupement simultané des utilisateurs et des articles. Malheureusement, dans cette approche la prise en compte des données manquantes n'est pas appropriée, conduisant ainsi à une faible qualité de recommandation. Nous proposons donc de faire un meilleur usage de cet algorithme par une prise en compte plus efficace des données manquantes. D'autre part, en exploitant le potentiel des résultats de la classification croisée, nous développons un outil interactif de visualisation et d'interprétation simultanée des groupes d'utilisateurs et des groupes d'items.

Le reste du papier est organisé comme suit. La section 2 présente le système de FC basé sur la classification croisée (COCLUST). Les sections 3 et 4 fournissent des détails sur nos approches de gestion des notes manquantes et de visualisation. La section 5 démontre l'efficacité des approches proposées sur des données réelles. Enfin, la section 6 conclut et présente les directions pour des recherches futures.

Notation. Soit $\mathbf{U}$ la matrice des notes, une classification croisée en $K \times L$ *co-clusters* (blocs ou sous-matrices résultant d'une classification croisée) par COCLUST conduit à une partition de l'ensemble des utilisateurs en K classes et une partition de l'ensemble des items en L classes. Notons $\mathbf{Z} = (z_{ik})$ la matrice de classification binaire de taille $(n \times K)$ définie par $z_{ik} = 1$ si l'utilisateur i appartient à la $k^{ième}$ classe et 0 sinon. De la même manière notons $\mathbf{W} = (w_{j\ell})$ la matrice de classification binaire de taille $(p \times L)$ définie par $w_{j\ell} = 1$ si l'item j appartient à la $\ell^{ième}$ classe et 0 sinon. Par commodité, nous utiliserons également $\mathbf{z} = (z_1, \ldots, z_n)$ avec $z_i \in \{1, \ldots, K\}$ (respectivement $\mathbf{w} = (w_1, \ldots, w_p)$ avec $w_j \in \{1, \ldots, L\}$; le vecteur des étiquettes des items). Enfin, nous utiliserons les indices i, j, k et ℓ pour désigner implicitement les lignes (utilisateurs), les colonnes (items), les classes en ligne (classes d'utilisateurs) et les classes en colonnes (classes des items) respectivement.

2 Classification croisée par COCLUST

Partant de l'algorithme *weighted Bregman co-clustering* (Banerjee et al., 2004), les auteurs dans (George et Merugu, 2005) ont proposé de s'attaquer au problème de recommandation moyennant une reconstitution des données observées suivant une classification croisée donnée. Plus précisément, à partir de la réorganisation en co-clusters obtenus par l'algorithme COCLUST, les auteurs proposent une matrice d'approximation de $\mathbf{U}$ sparse par une matrice $\hat{\mathbf{U}} = (\hat{u}_{ij})$ non sparse où chaque cellule est définie de la manière suivante :

$$\hat{u}_{ij} = \overline{u}_{k\ell} + (\overline{u}_i - \overline{u}_{k.}) + (\overline{u}_j - \overline{u}_{.\ell}) \tag{1}$$

avec $\overline{u}_{k\ell}$, $\overline{u}_{k.}$, $\overline{u}_{.\ell}$, $\overline{u}_i$, $\overline{u}_j$ sont respectivement les moyennes calculées sur l'ensemble des valeurs observées dans le co-cluster (k, ℓ), dans la classe des utilisateurs k, dans la classe des items, ℓ pour chaque utilisateur et pour chaque item. Notons donc que dans cette formulation $\hat{u}_{ij}$ dépend de i, j, k et ℓ.

Sachant que les partitions $\mathbf{Z}$ and $\mathbf{W}$ sont inconnues, le critère à minimiser par COCLUST est le suivant :

$$\arg \min_{(\mathbf{Z},\mathbf{W})} \sum_{i=1}^{n} \sum_{j=1}^{p} m_{ij}(u_{ij} - \hat{u}_{ij})^2 \tag{2}$$

où $\mathbf{M} = (m_{ij})$ est une matrice binaire de taille $(n \times p)$ où $m_{ij} = 1$ si u_{ij} est observé et $m_{ij} = 0$ si u_{ij} est manquant. Une solution (optimum local) de ce problème peut être obtenue par une minimisation alternée ; sachant $\mathbf{Z}$ puis sachant $\mathbf{W}$ (Banerjee et al., 2004) jusqu'à la convergence (Algorithm 1). A la convergence la prédiction est obtenue en utilisant (1).

Algorithm 1 Training based on Co-clustering.

Input : $\mathbf{U}$, K, L
Ouput : (Z, W), $\hat{\mathbf{U}}$
Initialisation of $\mathbf{Z}$ and $\mathbf{W}$;
repeat
1. Compute : $\overline{u}_{k\ell}$, $\overline{u}_{k.}$, $\overline{u}_{.\ell}$, $\overline{u}_i$ and $\overline{u}_j$; ($\forall$ k, ℓ, i and j)
2. Update $\mathbf{Z}$:
for $i = 1$ **to** n **do**
$z_i = \arg\min_{1 \leq k \leq K} \sum_{j=1}^{p} m_{ij}(u_{ij} - \overline{u}_{kw_j} - \overline{u}_i + \overline{u}_{k.} - \overline{u}_j + \overline{u}_{.w_j})^2$
end for
3. Update $\mathbf{W}$:
for $j = 1$ **to** p **do**
$w_j = \arg\min_{1 \leq l \leq L} \sum_{i=1}^{n} m_{ij}(u_{ij} - \overline{u}_{z_i\ell} - \overline{u}_i + \overline{u}_{z_i.} - \overline{u}_j + \overline{u}_{.\ell})^2$
end for
until Convergence

Comme indiqué précedemment les estimations au cours des itérations de COCLUST sont basées uniquement sur les données observées. Malheureusement et étant donné que le taux des données manquantes est très élevé (la *sparsité* de certaines matrices peut être de l'ordre de 99 %), les prédictions sont biaisées impliquant des qualités de recommandation discutables. D'autre part, George et Merugu (2005) ont proposé de remplacer la moyenne d'un co-cluster vide (qui ne contient aucune note observée) par la moyenne globale. Cette stratégie peut fortement perturber la qualité de la classification croisée, et de plus elle ne garantit pas la convergence de COCLUST, comme le montre figure 1. Pour plus d'explications, nous avons rapporté

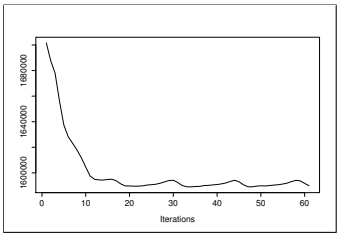
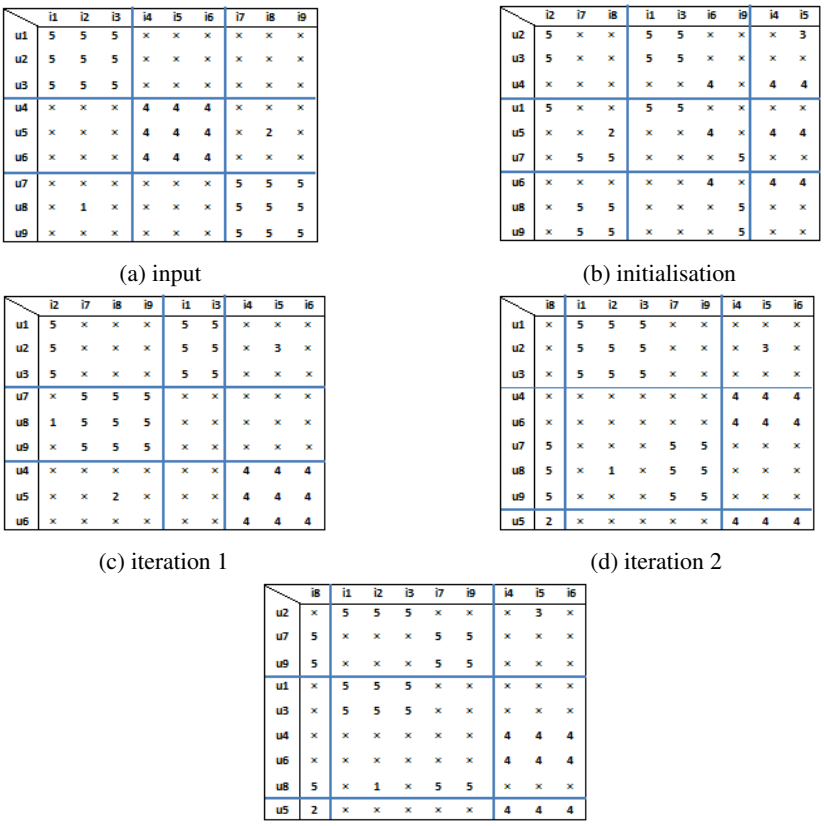


FIG. 1: Illustration de la divergence COCLUST sur les données sparse *MovieLens*.



(a) input

(b) initialisation

(c) iteration 1

(d) iteration 2

(e) iteration 3

FIG. 2: Classification croisée en utilisant COCLUST (le symbole $\times$ représente une note manquante) .

dans figure 2a une matrice de données, avec trois classes lignes et trois classes colonnes bien séparées. Les figures de 2b à 2e présentent les différentes itérations de COCLUST. Le résultat final figure 2e montre que COCLUST n'est pas en mesure de trouver les bonnes partitions des lignes et des colonnes, même si les classes sont bien séparées. En effet, nous notons que le résultat obtenu à l'itération 1 figure 2c est meilleur que le résultat final en terme d'homogénéité des co-clusters. Cela est dû au fait que la moyenne des co-clusters vides est remplacée par la moyenne globale. Dans la section suivante, nous proposons une nouvelle manière d'imputer les données manquantes, afin de pouvoir appliquer COCLUST et pallier les problèmes cités précedemment.

	i1	i2	i3	i4
u1	5	NA	3	NA
u2	3	5	2	NA
u3	4	NA	1	4
u4	4	NA	1	5
u5	NA	NA	NA	NA

(a) Matrice utilisateur-item M0

	i1	i2	i3	i4
u1	5	5	3	4.5
u2	3	5	2	4.5
u3	4	5	1	4
u4	4	5	1	5
u5	4	5	1.75	4.5

(b) M0 après imputation

FIG. 3: Imputation par les moyennes des items

3 Gestion des notes manquantes dans le FC

Pour surmonter le problème des données manquantes dans le FC, deux approches sont principalement utilisées. La première consiste à travailler uniquement sur les valeurs observées, et la deuxième consiste à utiliser les procédures d'imputation. L'imputation par la moyenne est la plus couramment utilisée, elle consiste à remplacer les notes manquantes d'un item/utilisateur par la moyenne de ses notes observées.

Ces approches peuvent être efficaces, si peu de valeurs sont manquantes, et que le mécanismes des données manquantes est *Missing completely at random* ou *Missing at random* (Little et Rubin, 2002). Malheureusement le taux de notes manquantes dans le filtrage collaboratif est très élevé, ce qui rend ces approches inefficaces dans ce contexte. En effet, elles peuvent conduire à des estimations fortement biaisées, ce qui impacte négativement la qualité des recommandations. Pour illustrer ce propos, nous avons rapporté dans figure 3 un exemple d'une matrice utilisateur-item, avant (figure 3a) et après l'imputation par les moyennes des items (figure 3b). Si nous voulons ordonner les items en fonction des préférences des utilisateurs, l'ordre le plus fiable serait : $i1$, $i4$, $i2$, $i3$ (tels que $i1$ est l'élément le plus apprécié). En revanche si nous utilisons la matrice après imputation pour trier ces éléments de la même manière, nous obtiendrons l'ordre suivant : $i2$, $i4$, $i1$, $i3$ qui est absurde, puisque $i1$ arrive seulement en troisième position et $i2$ arrive en première position. Cela est dû aux estimations fortement biaisées des moyennes des utilisateurs $i2$ et $i4$. Dans ce qui suit, nous allons présenter une nouvelle méthode d'imputation basée sur la version en ligne de l'algorithme $k$means sphérique (OSPK-means) (Zhong, 2005). Notre approche repose sur les deux étapes principales, 1) Partitionner l'ensemble des utilisateurs en k classes, en utilisant l'algorithme OSPK-means et en tenant compte des valeurs manquantes, 2) Estimer les notes manquantes, en se basant sur les résultats de la classification. Et enfin remplacer celles-ci dans la matrice $\mathbf{U}$. Ci-dessous, nous décrivons de manière détaillée les différentes étapes de notre approche :

3.1 Etape de Classification

Dans le but de partitionner l'ensemble des utilisateurs en k groupes, nous proposons les procédures suivantes :

Initialisation : Dans la version initiale de OSPK-means, l'initialisation se fait par un tirage aléatoire de K centres initiaux parmi l'ensemble des utilisateurs. Cependant cette stratégie n'est pas efficace dans notre cas. En effet la probabilité de choisir un utilisateur avec très peu de notes observées, comme un centre de gravité initial est élevée. D'autre part, sélectionner

les centres initiaux uniquement parmi l'ensemble des utilisateurs ayant noté beaucoup d'items, permettrait seulement la détection de certains groupes. Afin de surmonter ces difficultés, nous proposons la procédure d'initialisation suivante :

1. Générer une partition aléatoire des utilisateurs en k classes.
2. Estimer les centres initiaux comme suit : soit μ_{kj} la $j^{ième}$ composante du centre k alors :

$$\mu_{kj} = \begin{cases} \frac{\sum_i z_{ik} m_{ij} u_{ij}}{\sum_i z_{ik} m_{ij}} & \text{; si } \sum_i z_{ik} m_{ij} \geq S \\ \frac{\sum_i z_{ik} m_{ij} u_{ij}}{S} & \text{; sinon} \end{cases}$$

où S est un seuil proportionnel à la taille de la classe k, et peut être défini par l'utilisateur. Intuitivement, cette stratégie permet d'estimer la $j^{ième}$ composante du centre de la $k^{ième}$ classe à partir des données disponibles, mais seulement s'il y a suffisamment de notes observées pour dans cette classe. En revanche, quand peu de valeurs sont observées pour une composante j l'estimation de celle-ci est pénalisée, en divisant par le seuil S.

Etape de mise à jour : Lorsque l'utilisateur choisi dans l'étape d'affectation ne dispose pas de suffisamment de notes-observées, l'assignation de celui-ci n'est pas fiable. Par conséquent le centre correspondant ne doit pas être déplacé dans le sens de cet utilisateur. Pour résoudre ce problème, nous introduisons une fonction binaire ($h(\mathbf{u}) \in \{0, 1\}$) qui annule la mise à jour dans ce cas. L'Algorithme 2 fournit plus de détails sur cette étape de classification.

Algorithm 2 Classification.

Input : n normalized users $\mathbf{u}_i$ ($\|\mathbf{u}_i\| = 1$) in $\mathbb{R}^p$, K : number of user clusters, η : learning rate, B : number of batch iterations ;
Output : K Centroids $\boldsymbol{\mu}_k$ in $\mathbb{R}^p$, and $\mathbf{z} = (z_1, \ldots, z_n)$;
Steps :
1. Random initialization of the partition $\mathbf{z}$;
2. Estimation of initial centroids : $\mu_{kj} = \begin{cases} \frac{\sum_i z_{ik} m_{ij} u_{ij}}{\sum_i z_{ik} m_{ij}} & \text{if } \sum_i z_{ik} m_{ij} \geq S \\ \frac{\sum_i z_{ik} m_{ij} u_{ij}}{S} & \text{; otherwize.} \end{cases}$

for $b = 1$ **to** B **do**
  for $i = 1$ **to** n **do**
    3. Assignment : for each user $\mathbf{u}_i$, compute z_i : $z_i = \arg\min_k (1 - \mathbf{u}_i^T \boldsymbol{\mu}_k)$.
    4. Update the winner centroid : $\hat{\boldsymbol{\mu}}_{z_i} = \frac{\boldsymbol{\mu}_{z_i} + \eta h(\mathbf{u}_i) \mathbf{u}_i}{\|\boldsymbol{\mu}_{z_i} + \eta h(\mathbf{u}_i) \mathbf{u}_i\|}$;
    $t = t + 1$;
  end for
end for

3.2 Estimation des notes manquantes

Dans cette étape les notes manquantes sont estimées, en se basant sur les résultats de la classification. Cependant, une pondération des notes s'avère encore nécessaire. Nous avons

choisi d'accorder plus d'importance aux utilisateurs representant le mieux leur classe d'appartenance en pondérant par $cos(\mathbf{u}_i, \boldsymbol{\mu}_k)$ tout en atténuant l'effet des utilisateurs qui ne sont pas en accord avec la préférence globale pour un item au sein de leur classe d'appartenance à l'aide de $p(u_{ij})$. Soit $\mathbf{u}_a$ un utilisateur actif, $k = z_a$, la note pour pour un item j prend la forme suivante :

$$u_{aj} = \frac{\sum_i^n z_{ik} \times cos(\mathbf{u}_i, \boldsymbol{\mu}_k) \times p(u_{ij}) \times u_{ij}}{\sum_i^n z_{ik} \times cos(\mathbf{u}_i, \boldsymbol{\mu}_k) \times p(u_{ij})} ; p(u_{ij}) = \begin{cases} p(u_{ij} > r_{med}) & ; \text{si } u_{ij} > r_{med} \\ p(u_{ij} \leq r_{med}) & ; \text{sinon} \end{cases} \tag{3}$$

où r_{med} est la note médiane ($r_{med} = 3$ if $u_{ij} \in \{1, 2, 3, 4, 5\}$, $p(u_{ij} \geq r_{med})$ est la probabilité qu'un item j soit apprécié au sein d'un groupe k, tel que :

$$p(u_{ij} > r_{med}) = \begin{cases} \frac{\sum_{i, u_{ij} \geq r_{med}} z_{ik} m_{ij}}{\sum_i z_{ik} m_{ij}} & ; \text{si } \sum_i z_{ik} m_{ij} \geq S \\ 0.5 & ; \text{sinon} \end{cases}$$

Dans la section suivante, nous proposons d'exploiter les résultats de classification croisée, dans le but de fournir aux utilisateurs une représentation interactive basée sur des graphes bipartis. Cette dernière permet non seulement de faciliter l'interprétation des résultats, mais aussi de donner un sens aux préférences des utilisateurs dans le contexte du filtrage collaboratif.

4 Visualisation des résultats de la classification croisée

Il y a très peu de travaux qui se sont intéressés à l'aspect visualisation dans le contexte des systèmes de recommandation. En effet ces systèmes sont souvent évalués pour leur capacité à faire de bonnes recommandations, mais leur fonctionnement reste abstrait pour les utilisateurs. Parmi les quelques travaux de visualisation on peut citer la méthode de visualisation des données du FC (Mei et Shelton, 2006) qui consiste à représenter les utilisateurs à côté des items qu'ils aiment, sur le même espace euclidien. On peut aussi citer *PeerChooser* (Smyth et al., 2008) qui est un système de FC interactif qui permet de visualiser sous forme de graphe les interactions entre un utilisateur actif et son voisinage, tout en offrant la possibilité de modifier ce dérnier. Il existe aussi des travaux qui proposent de visualiser sur un plan à deux dimensions la liste d'items à recommander pour un utilisateur actif, en utilisant les techniques classiques telles que l'ACP, MDS, SOM.

Dans ce travail nous proposons une nouvelle approche de visualisation dans le contexte des systèmes de recommandations. Contrairement aux autres méthodes citées ci-dessus, notre approche est globale, c'est à dire qu'elle ne se focalise pas uniquement sur l'utilisateur actif. Elle exploite la dualité inhérente de la classification croisée pour mieux mettre en évidence les affinités entre certains types de groupes d'utilisateurs et certains types de produits. Plus précisément, nous proposons de représenter les relations de préférences entre des groupes d'utilisateurs et groupes d'items, au moyen des graphes bipartis. Notre approche peut être décrite comme suit : 1) Classifier la matrice utilisateur-item en K classes d'utilisateurs et L classes d'items. Dans nos expérimentations nous avons utilisé COCLUST, après la gestion des données manquantes, présentée dans la section précédente, 2) Construire une matrice résumant les résultats de la classification croisée, dans laquelle chaque groupe de de lignes et chaque groupe

de colonnes est représenté par les utilisateurs et les items les plus populaires (qui ont le plus de votes) respectivement, 3) Calculer la relation de préférence entre chaque groupe d'utilisateurs et chaque groupe d'items, à l'aide de la formule (4), 4) Construire le graphe biparti, étape décrite en détail dans la partie expérimentale.

Soit $\mathbf{U}' = (u'_{ij})$ la matrice résumée de l'étape 2, avec n utilisateurs et p items. Et soit $\mathbf{E} = (e_{ij})$ une matrice binaire de $n \times p$, tel que $e_{ij} = 1$ si l'utilisateur i aime l'item j et $e_{ij} = 0$ sinon. Alors la corrélation entre la $k^{ième}$ classe d'utilisateurs et le $\ell^{ième}$ classe d'items est calculée comme suit :

$$c_{k\ell} = \frac{\sum_i \sum_j z_{ik} \times w_{jl} \times e_{ij} \times u_{ij}}{\sum_i z_{ik} \times \sum_j w_{j\ell}}; \tag{4}$$

Intuitivement la corrélation (de préférence) 4, entre un groupe d'utilisateurs k et un certain groupe d'items ℓ, représente la proportion des items populaires dans la $\ell^{ième}$ classe ayant été appréciée par les utilisateurs les plus populaires de la classe k. La section suivante présente les résultats expérimentaux démontrant l'efficacité des approches proposées.

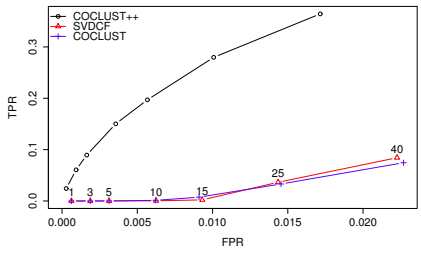
Algorithm 3 Bipartite procedure.

Input : $\mathbf{U}$, K and L ;
Output : $\mathbf{C}$: correlation matrix between clusters ;
Steps :
1. Compute $(\mathbf{Z}, \mathbf{W})$ into K row clusters and L column clusters ;
2. Compute $\mathbf{U}'$ with the relevant users and items.
for $k = 1$ **to** K **do**
  for $l = 1$ **to** L **do**
    4. Compute $\mathbf{C} = (c_{k\ell})$ the correlation matrix between clusters, by using (4)
  end for
end for
5. Build the bipartite graph

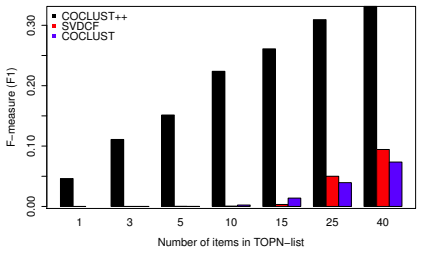
5 Résultats expérimentaux

Dans nos expériences, nous avons choisi les deux jeux de données de MovieLens [1] (ML-100K et ML-1M) qui sont beaucoup utilisés dans le domaine. L'échantillon ML-1M est constitué de 6040 utilisateurs, 3952 films, et de 1 million de notes observées. L'ensemble ML-100K contient 100,000 notes fournies par 943 utilisateurs pour 1664 films. La proportion des notes observées dans ce dernier est seulement de $6,4\%$. Les évaluations des utilisateurs (u_{ij}) appartiennent à l'intervalle : $[1; 5]$, et les notes manquantes sont codées par : NA. Les données MovieLens fournissent également certaines informations démographiques sur les utilisateurs, telles que : le sexe, l'âge, la profession, code postal ; et des informations de base sur les films tels que : le titre, le genre, la date de sortie, etc. A noter qu'un film peut être de plusieurs genres à la fois. Nous proposons dans la suite de réaliser la comparaison des courbes ROC et de la

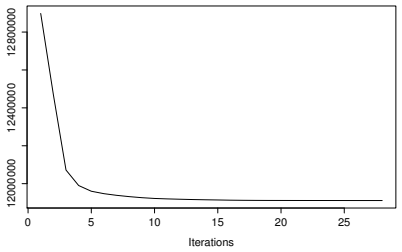
1. http ://grouplens.org/datasets/movielens/



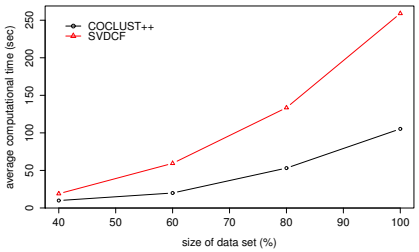
(a) Comparaison des courbes ROC sur l'ensemble ML-100k



(b) Comparaison de la F-measure (F1) sur l'ensemble ML-100k



(c) Convergence de COCLUST++ sur les données ML-1M



(d) Comparaison du temps de calcule, sur les données ML-1M

FIG. 4: Evaluation de plusieurs systèmes de FC sur les données MovieLens.

F-measure, des systèmes de FC suivants : COCLUST, le FC incrémental basé sur la décomposition en valeurs singulières SVDCF (Sarwar et al., 2002), et COCLUST++ (COCLUST après la gestion des valeurs manquantes). Ces comparaisons sont réalisées sous *recommenderlab* (Hahsler, 2011), que nous avons combiné avec le langage C pour implémenter les différentes méthodes ci-dessus. Les courbes de figure. 4a sont construites en faisant varier le nombre d'items à recommander de 1 à 40. Les deux figures 4a et 4b montrent une amélioration significative des performances de COCLUST, grâce à la gestion des données manquantes que nous proposons. On remarque aussi, une faible qualité des recommandations pour SVDCF, qui est due à une gestion des données manquantes inappropriée. En effet dans cette dernière approche (Sarwar et al., 2002), les notes manquantes sont remplacées par les moyennes des items dont les estimations sont fortement biaisées. En d'autres termes cette imputation favorise les items avec très peu de notes observées, comme illustré dans la section 3 (figure 3). La figure 4d montre que même avec l'étape d'imputation COCLUST++ reste plus rapide que SVDCF.

En ce qui concerne les possibilités de visualisation exploitant la classification croisée, la figure 5 montre un exemple de graphe biparti, qui est construit comme suit 1) Classification de l'ensemble ML-100k, en 6 classes utilisateurs et 8 classes d'items, en utilisant COCLUST++, 2) Calculer les corrélations entre les groupes d'utilisateurs et d'items, via la formule (4), 3) Construire le graphe biparti où les rectangles de gauche représentent des groupes d'utilisateurs, tandis que ceux de droite des groupes d'items. Seuls les liens qui correspondent à de fortes corrélations sont représentés. Pour chaque groupe d'utilisateurs, les deux professions les plus populaires sont présentées, de même les deux genres les plus populaires dans chaque classe

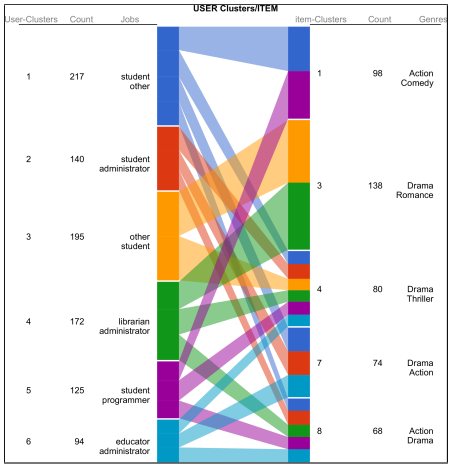


FIG. 5: Graphe biparti représentant les données ML-100k partitionnées en 6 et 7 classes d'utilisateurs et d'items, respectivement .

de films sont représentés. Toutes les représentations des graphes bipartis, qui suivent, ont été réalisées à l'aide de la bibliothèque *D3.js* [2].

La figure 5 montre que les groupes d'utilisateurs et de films ont des proportions différentes. Ces dernières correspondent à la hauteur des rectangles. Par exemple les classes d'utilisateurs 1 et 3, qui correspondent aux rectangles bleu et jaune, respectivement, représentent une plus grande proportion d'utilisateurs par rapport aux autres groupes. Nous notons également une

2. http ://d3js.org/

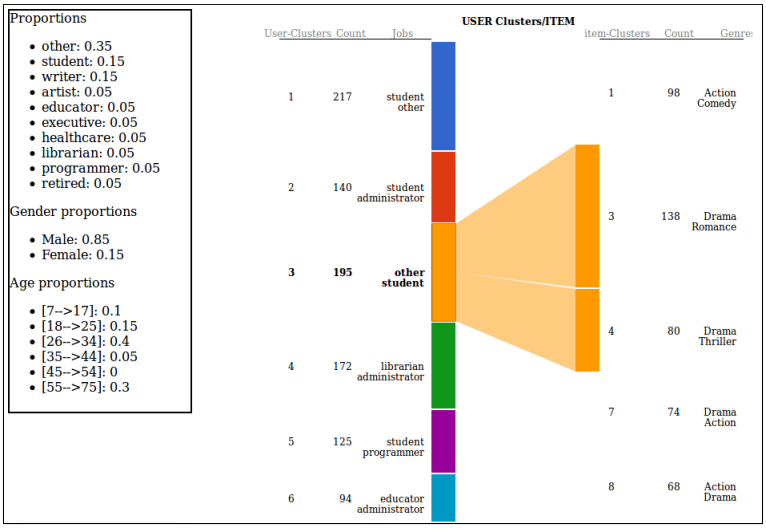


FIG. 6: Graphe biparti de figure 5 au survol par la souris de la classe utilisateurs 3.

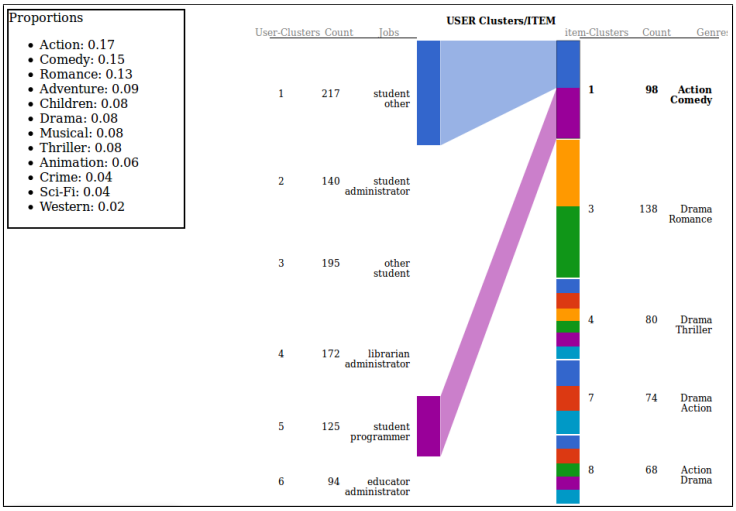


FIG. 7: Graphe biparti de figure 5 au survol par la souris de la classe item 1.

corrélation intéressante, entre le groupe d'utilisateurs 5 (violet) et les groupes d'items 1, 4 et 8, ce qui peut s'interpréter comme une préférence particulière des *étudiants* et *programmeurs* (les professions les plus populaires dans la classe 5) pour des films *d'Action, Thriller* et *Drame*.

Un autre résultat important pouvant être déduit à partir de la figure 5 est que certains groupes peuvent être fusionnés. Par exemple les groupes d'items 4 et 8 peuvent être fusionnés. En effet, ces deux groupes sont représentés principalement par des films de *Drama*, et ils sont appréciés par presque tous les groupes d'utilisateurs.

D'autres interactions sont possibles, comme l'affichage d'informations supplémentaires sur un groupe, au survol de celui-ci par la souris (figure 6). Il est ainsi possible de constater que les membres de la classe d'utilisateurs 3 ont une forte préférence pour les films des groupes 3 et 4. La préférence pour les films de *Drame* et *Romance*, peut être expliquée par la présence *d'écrivains, de bibliothécaires* et *d'artistes* dans le groupe d'utilisateur 3, et également par la forte proportion des personnes âgées entre 55 et 75 ans, dans ce dernier. La Figure 7 montre une autre interaction au survol de la classe d'items 1.

6 Conclusion

Dans ce papier nous avons proposé une meilleure exploitation du potentiel de la classification croisée dans les systèmes de FC. Pour ce faire, nous avons développé une nouvelle stratégie pour une gestion efficace des données manquantes. Nous avons ensuite proposé une nouvelle approche interactive basée sur des graphes bipartis, permettant d'interpréter et de comprendre les résultats de la classification croisée dans le contexte du FC. Les résultats expérimentaux montrent une amélioration importante des performances de la classification croisée dans le FC, grâce à une meilleure gestion des notes manquantes. Nous avons aussi montré, comment les représentations interactives basées sur des graphes bipartis peuvent aider les uti-

lisateurs à donner un sens aux résultats obtenus, à la fois en détectant les co-clusters les plus intéressants et en analysant le contenu de ces derniers.

Références

Banerjee, A., I. Dhillon, J. Ghosh, S. Merugu, et D. S. Modha (2004). A generalized maximum entropy approach to bregman co-clustering and matrix approximation. In *KDD*.

Bobadilla, J., F. Ortega, A. Hernando, et A. Gutiérrez (2013). Recommender systems survey. *Knowledge-Based Systems*, 109 – 132.

Delporte, J., A. Karatzoglou, et S. Canu (2014). Apprentissage et factorisation pour la recommandation. *Revue des Nouvelles Technologies de l'Information (RNTI) RNTI-A-6*.

George, T. et S. Merugu (2005). A scalable collaborative filtering framework based on co-clustering. In *Fifth IEEE International Conference on Data Mining*, pp. 625–628.

Goldberg, D., D. Nichols, B. Oki, et D. Terry (1992). Using collaborative filtering to weave an information tapestry. *Communications of the ACM 35*(12), 70.

Hahsler, M. (2011). recommenderlab : A framework for developing and testing recommendation algorithms .

Koren, Y. (2009). The bellkor solution to the netflix grand prize.

Little, R. J. A. et D. B. Rubin (2002). *Statistical analysis with missing data (second edition)*.

Mei, G. et C. R. Shelton (2006). Visualization of collaborative data. In *UAI*, pp. 341–348.

Sarwar, B., G. Karypis, J. Konstan, et J. Riedl (2002). Incremental singular value decomposition algorithms for highly scalable recommender systems. In *Fifth ICIS*, pp. 27–28.

Sarwar, B. M., G. Karypis, J. A. Konstan, et J. T. Riedl (2000). Application of dimensionality reduction in recommender system – a case study. In *ACM WEBKDD WORKSHOP*.

Smyth, B., B. Gretarsson, S. Bost, et T. Höllerer (2008). Peerchooser : visual interactive recommendation. In *In CHI '08*, pp. 1085–1088. ACM.

Zhong, S. (2005). Efficient online spherical k-means clustering. In *In Proc. IEEE Int. Joint Conf. Neural Networks*, pp. 3180–3185.

Summary

Collaborative filtering systems (CFs) aim to provide relevant items for users on the web. Most of existing CFs are based on matrix factorization and k nearest neighbors methods. Unfortunately both approaches are expensive in terms of computational time, and do not treat missing data in the user-item rating matrix. The computational time flaw, can be addressed by using co-clustering methods, which involve the user and item spaces simultaneously. However, the latter approaches still need an efficient strategy for handling missing values. In this work, we propose an effective method for handling unobserved ratings, allowing a better use of co-clustering approaches in CF. Furthermore we propose an interactive representation of co-clustering results. Based on bipartite graphs, this representation allows an easy interpretation and sense making of the preferences between user and item clusters.