

Cahier thématique

Recommandation de Films Basé sur le filtrage Collaboratif - Approche du User-Based

**Manipulation 1 :** Préparation des données d’apprentissage et de test

**Manipulation 2 :** Construire le modèle de recommandation User-based

**Manipulation 3 :** Application du modèle de recommender sur l'ensemble de test

**Préparé par :** Nesrine Zemirli  
  
© Nesrine Zemirli et Hafed Benteftifa 2015-2016

Ce document ne peut être utilisé dans le cadre d’une formation, publication papier, site internet ou tout support sans mon accord express. Aucune reproduction, même partielle, ne peut être faite de ce document et de l'ensemble de son contenu : textes, images, etc. sans mon autorisation express. Pour toutes informations, communiquer avec moi sur info@degenio.com.

|  |  |  |
| --- | --- | --- |
| **Date** | **Version** | **Changement** |
| 20 Juin 2016 | 1.0 | Version initiale |
|  |  |  |
|  |  |  |

# Mise en contexte et présentation du projet

# Format : individuel

# Jeux de données : on a à notre disposition une source de données disponible sur RStudio

## Préliminaire

* R-studio est disponible.
* Utilisation du package « recommenderlab » Contenant lesfonction pour Développer et Tester les algorithmes de recommandation.

## Objectif

Construire un système de recommandation basé sur le filtrage collaboratif pour suggérer des films aux utilisateurs. Pour cela, nous allons appliquer l’approche du filtrage collaborative basé sur les utilisateurs.

* **User-based Collabirative filtering**: le filtrage collaboratif basé sur l'utilisateur recommande à un utilisateur les items qui sont les plus préférés par les utilisateurs qui lui sont similaires.

# Description : MovieLens

Les données du TP sont issues de la collection des films de MovieLens, projet de l’université du Minnesota, représentent les avis des utilisateurs sur des films. Chaque film a un certain nombre de commentaires. Les avis (critique) sont des notes de 1 à 5 : 1 pour pas intéressant et 5 pour très intéressant.

Les données sont sur le site de MovieLens : <http://grouplens.org/datasets/movielens/100k/>

Cet ensemble contient les notes que les utilisateurs ont donnés aux films. Chaque ligne de MovieLense correspond à un utilisateur, et chaque colonne correspond à un film. Il y a : 943 utilisateurs pour 1664 évaluation.

Il y a donc plus de 1.500.000 combinaisons entre un utilisateur et un film. Par conséquent, le stockage de la matrice complète nécessiterait plus de 1.500.000 cellules. Cependant, les utilisateur n’ont pas évaluée tous les films (environ 20 films par utilisateur). Par conséquent, il y a moins de 100.000 notes, et la matrice est creuse.

## Démarche

Dans cet TP 2, nous allons utiliser l'approche du user-based. Premièrement, étant donné un nouvel utilisateur, nous allons identifier ses utilisateurs similaires. Ensuite, nous recommanderons les articles meilleurs achetés par les utilisateurs similaires.

Pour chaque nouvel utilisateur, les étapes sont :

1. Mesurer la similarité de nouveau utilisateur avec l’ensemble des utilisateurs. Les mesures de similarité populaires sont corrélation et cosinus.
2. Identifier les utilisateurs les plus semblables. Les options sont :

* Prendre en compte les *k* plus proches utilisateurs (k-nearest\_neighbors)
* Prendre en compte les utilisateurs dont la similarité est supérieure à un seuil défini

1. Obtenir le ratings des items évaluées par les utilisateurs les plus similaire. La valeur du rating est une moyenne parmi le rating des utilisateurs similaires :
   * Moyenne des ratings
   * Moyenne pondérée des ratings, en utilisant les similarité s comme poids
2. Choisissez les meilleurs items

## Manipulation 1 : Préparation des données d’apprentissage et de test

**Objectif**: construire le modèle en utilisant une partie de l'ensemble de données MovieLense comme ensemble d’apprentissage et l'appliquer sur l'autre partie (le jeu de test).

**Démarche**

Les deux ensembles sont les suivantes :

• Jeu d’apprentissage : Cet ensemble comprend les utilisateurs à partir de laquelle le modèle apprend

• Jeu de test : Cet ensemble comprend les utilisateurs auxquels nous recommandons des films

L'algorithme normalise automatiquement les données, afin que nous puissions utiliser ratings\_movies qui contient les utilisateurs et les films de MovieLense pertinents. Nous avons défini ratings\_movies dans la section précédente qui contient les données pertinentes à prendre en considération : le sous-ensemble d'utilisateurs MovieLense qui ont classé au moins 50 films et des films qui ont été notés au moins 100 fois.

## Charger les données

library(pander)

library("recommenderlab")

library("ggplot2")

library("knitr")

set.seed(1)

data(MovieLense)

# sélection des prédicteurs pertinent

ratings\_movies <- MovieLense[rowCounts(MovieLense) > 50,

colCounts(MovieLense) > 100]

# Normalisation des valeurs

ratings\_movies\_norm <- normalize(ratings\_movies)

## Donnée d’apprentissage et de test

Premièrement, nous définissons au hasard le vecteur which\_train qui est vrai pour les utilisateurs dans l'ensemble d’apprentissage et FALSE pour les autres. Nous allons définir la probabilité de la formation définie comme 80 pour cent :

which\_train <- sample(x = c(TRUE, FALSE), size = nrow(ratings\_movies),

replace = TRUE, prob = c(0.8, 0.2))

head(which\_train)

## [1] TRUE TRUE TRUE FALSE TRUE FALSE

Puis nous construisons les jeux de données respectivement

#Let's define the training and the test sets:

recc\_data\_train <- ratings\_movies[which\_train, ]

recc\_data\_test <- ratings\_movies[!which\_train, ]

recc\_data\_train

451 x 332 rating matrix of class ‘realRatingMatrix’ with 43846 ratings.

recc\_data\_test

109 x 332 rating matrix of class ‘realRatingMatrix’ with 11452 ratings.

## Manipulation 2 : Construire le modèle de recommandation User -Based Collaboratif Filtering

Objectif : construire la fonction qui implémente l’algorithme de recommandation modèles, ses entrées sont les suivantes :

* Données: Ceci est l'ensemble de données d’apprentissage
* Méthode: Ceci est le nom de la technique
* Paramètres: Ce sont des paramètres optionnels de la technique

## Fonction du user-based collaboratif

La fonction que l’on va utiliser est la suivante :

#Instancier les modèles du recommender

recommender\_models <- recommenderRegistry$get\_entries(dataType = "realRatingMatrix")

Sélectionner le UBCF et voir ces paramètres

df\_parameters <- data.frame(

parameter = names(recommender\_models$UBCF\_realRatingMatrix$parameters),

default = unlist(recommender\_models$UBCF\_realRatingMatrix$parameters)

)

rownames(df\_parameters) <- NULL

pander(head(df\_parameters))

Résultat :

---------------------

parameter default

----------- ---------

method cosine

nn 25

sample FALSE

normalize center

---------------------

## Construction du modèle de recommandation

Certains paramètres importants sont les suivants :

* method : Cela montre comment calculer la similarité entre les utilisateurs, par défaut : la similarité du cosinus
* nn : Cela montre le nombre d'utilisateurs similaires

Construire le modèle

recc\_model <- Recommender(data = recc\_data\_train,

method = "UBCF")

recc\_model

Résultat sera :

Recommender of type ‘UBCF’ for ‘realRatingMatrix’

learned using 451 users.

## Manipulation 3 : Application du modèle de recommender sur l'ensemble de test

**Objectif**: Recommander des films aux aux utilisateurs dans l'ensemble de test.

**Démarche**: Nous allons définir n\_recommended qui spécifie le nombre d'éléments de recommander à chaque utilisateur.

De la même manière que le IBCF, nous pouvons déterminer les six premières recommandations pour chaque nouvel utilisateur.

## Appliquer le modèle de recommandation sur le jeu de test

Dans notre ça nous souhaitons recommander à chaque utilisateur 6 films.

# set variable value

n\_recommended <- 6

recc\_predicted <- predict(object = recc\_model,

newdata = recc\_data\_test,

n = n\_recommended)

recc\_predicted

Résultat sera :

Recommendations as ‘topNList’ with n = 6 for 109 users.

## Inspecter les résultats

L’objet recc\_predicted contient les recommandations. Inspecter sa structure :

class(recc\_predicted)

slotNames(recc\_predicted)

## [1] "topNList"

## [1] "items" "ratings" "itemLabels" "n"

Les slotNames sont:

* items : Ceci est la liste avec les indices des items recommandés pour chaque utilisateur
* ratings : Ceci est la valeur du rating
* itemLabels: Ceci est le nom des items
* n: Ceci est le nombre de recommandations

Par exemple, ce sont les recommandations pour le premier utilisateur :

recc\_predicted@items[[1]]

8 30 165 69 119 4

Nous pouvons extraire les noms des films recommandés avec recc\_predicted@item

recc\_user\_1 <- recc\_predicted@items[[1]]

movies\_user\_1 <- recc\_predicted@itemLabels[recc\_user\_1]

# visualiser

table\_user\_1 <- data.frame(index = recc\_user\_1,

movie = movies\_user\_1)

pander(table\_user\_1)

Le résultat sera :

index movie

------- --------------------------

8 Usual Suspects, The (1995)

30 Shawshank Redemption, The

(1994)

165 Contact (1997)

69 Godfather, The (1972)

119 Nikita (La Femme Nikita)

(1990)

4 Twelve Monkeys (1995)

----------------------------------

Nous pouvons définir une matrice avec les recommandations pour chaque utilisateur :

recc\_matrix <- sapply(recc\_predicted@items, function(x){

colnames(ratings\_movies)[x]

})

dim(recc\_matrix)

# 6 109

Puis, visualiser les recommandations pour les 4 premiers utilisateurs :

pander(recc\_matrix[, 1:4])

------------------------------------------------------------------------------

5 7 8

-------------------------- -------------------------- -------------------------

Usual Suspects, The (1995) Lone Star (1996) Silence of the Lambs, The

(1991)

Shawshank Redemption, The This Is Spinal Tap (1984) Shawshank Redemption, The

(1994) (1994)

Contact (1997) Wrong Trousers, The (1993) Jaws (1975)

Godfather, The (1972) Hoop Dreams (1994) Schindler's List (1993)

Nikita (La Femme Nikita) Mighty Aphrodite (1995) Fargo (1996)

(1990)

Twelve Monkeys (1995) Big Night (1996) Boot, Das (1981)

-------------------------------------------------------------------------------

## Les films les plus recommandés

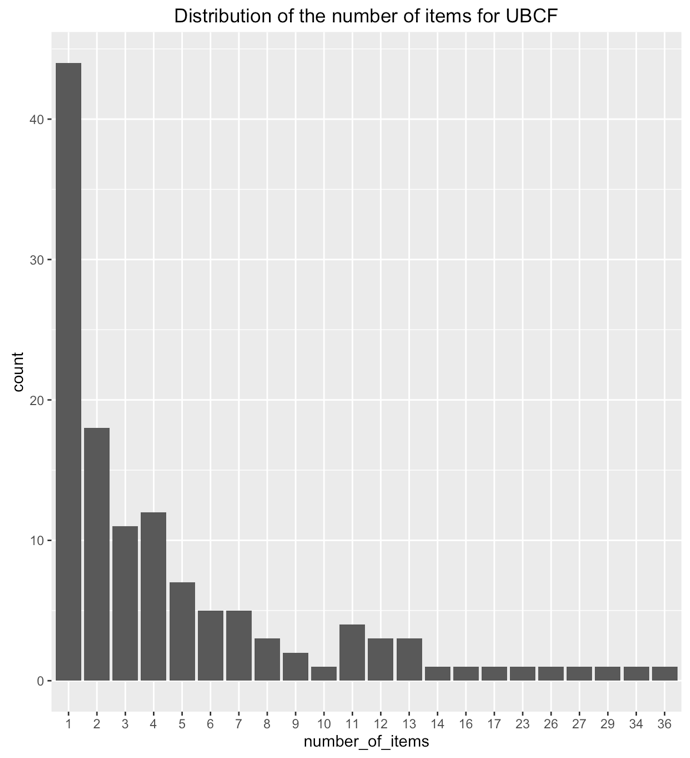
Maintenant, nous pouvons identifier les films les plus recommandés. A cet effet, nous allons définir un vecteur avec toutes les recommandations, et nous allons construire un tracé de fréquence:

number\_of\_items <- factor(table(recc\_matrix))

chart\_title <- "Distribution of the number of items for UBCF"

Visualiser

qplot(number\_of\_items) + ggtitle(chart\_title)



**Film les plus populaires**

La plupart des films ont été recommandés que quelques fois, et quelques films ont été recommandées à plusieurs reprises. Voyons quels sont les films les plus populaires :

number\_of\_items\_sorted <- sort(number\_of\_items, decreasing = TRUE)

number\_of\_items\_top <- head(number\_of\_items\_sorted, n = 4)

table\_top <- data.frame(

names(number\_of\_items\_top),

number\_of\_items\_top)

pander(table\_top)

Le résultat sera :

--------------------------------------------------------------------------------

&nbsp; names.number\_of\_items\_top. number\_of\_items\_top

----------------------------- ---------------------------- ---------------------

\*\*Schindler's List (1993)\*\* Schindler's List (1993) 36

\*\*Shawshank Redemption, The Shawshank Redemption, The 34

(1994)\*\* (1994)

\*\*Silence of the Lambs, The Silence of the Lambs, The 29

(1991)\*\* (1991)

\*\*Godfather, The (1972)\*\* Godfather, The (1972) 27

--------------------------------------------------------------------------------