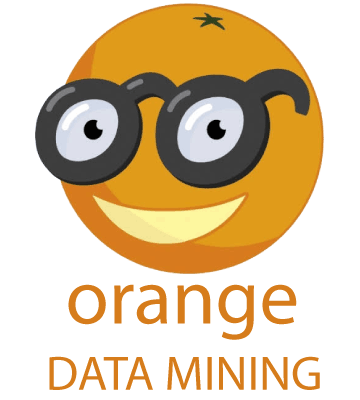


**Data mining et systèmes de recommandation**

Exemples avec les outils Orange et Python



Réalisé par : Encadré par : AMEKRANE Youssef Pr. SABIRI Mohamed

Table des matières

[**I. Introduction** 3](#_Toc154229300)

[**II.** **Qu’est-ce qu’un système de recommandation ?** 3](#_Toc154229301)

[**III.** **Quand utiliser un système de recommandation ?** 3](#_Toc154229302)

[**IV.** **Les approches des systèmes de recommandation** 4](#_Toc154229303)

[1) Les approches basées sur le filtrage collaboratif (Collaborative filetring) 4](#_Toc154229304)

[2) Les approches basées sur le contenu (Content based filetring) 7](#_Toc154229305)

[3) Les approches hybrides 9](#_Toc154229306)

[**V.** **Techniques de recommandation basées sur Data mining** 10](#_Toc154229307)

[1) Dataset utilisée 10](#_Toc154229308)

[2) Application sous Orange 11](#_Toc154229309)

[3) Application avec Python 12](#_Toc154229310)

[VI. **Conclusion** 14](#_Toc154229311)

# Introduction

Dans un monde où le nombre de choix est énorme, les systèmes de recommandation aident les utilisateurs à trouver et évaluer des articles d'intérêt. Ils mettent en relation les utilisateurs avec des éléments à "consommer" (acheter, visionner, écouter, etc.) en associant le contenu des articles recommandés ou les opinions d'autres individus aux actions ou opinions de l'utilisateur consommateur. De tels systèmes sont devenus des outils puissants dans des domaines allant du commerce électronique aux bibliothèques numériques et à la gestion des connaissances. Par exemple, un consommateur de presque n'importe quel grand détaillant en ligne qui manifeste un intérêt pour un article, que ce soit en consultant une description de produit ou en plaçant l'article dans son "panier d'achat", recevra probablement des recommandations pour d'autres produits. Ces produits peuvent être recommandés en se basant sur les meilleures ventes globales sur un site, sur les données démographiques du consommateur, ou sur une analyse du comportement d'achat passé du consommateur en tant que prédiction pour le comportement d'achat futur. Ce mini projet abordera la technologie utilisée pour générer des recommandations, en se concentrant sur l'application des techniques de fouille de données.

# **Qu’est-ce qu’un système de recommandation ?**

Un système de recommandation est une forme spécifique de filtrage de l’information qui a pour but de présenter à un utilisateur des éléments qui sont susceptibles de l’intéresser, et ce, en se basant sur ses préférences et son comportement. On tente donc de prédire votre appréciation face à un élément pour ainsi vous suggérer ce que vous serez le plus en mesure d’apprécier.

# **Quand utiliser un système de recommandation ?**

Globalement, un système de recommandation est un système de filtrage qu’on applique dans un contexte où il y a trop d’information pour l’utilisateur. Celui-ci passerait beaucoup de temps à retrouver l’information ou l’item qui l’intéresse ce qui pourrait causer une certaine frustration. C’est aussi un moyen de proposer des produits inconnus de l’utilisateur mais susceptibles de l’intéresser.

Le but est entre autres d’accroître la satisfaction de l’utilisateur en personnalisant son expérience sur le site ou en lui envoyant des messages ciblés via des canaux tels que les emails, les newsletters, l’interface web, ….

# **Les approches des systèmes de recommandation**

Nous abordons ici différentes approches pour proposer des recommandations à un utilisateur. Les trois approches les plus courantes sont celles basées sur le contenu comme dans Pandora, les approches collaboratives, celle d'Amazon.com par exemple, et les approches hybrides (qui sont une combinaison des deux précédentes) par exemple celle de Netflix.

## Les approches basées sur le filtrage collaboratif (Collaborative filetring)

La technique de recommandation collaborative utilise la connaissance des préférences de l’utilisateur actif (l’utilisateur à qui on veut faire une recommandation) c.à.d. les items qu’il a le plus consultés dans le passé ainsi que les préférences d’une large communauté d’utilisateurs. Elle s’appuie sur deux paradigmes :

* Des utilisateurs similaires expriment des préférences similaires et donnent des notes similaires (approche user-based). On prédit la préférence d’un utilisateur A pour un item X sur base des ratings donnés dans le passé par des utilisateurs ayant les mêmes intérêts qu’A à l’item X.
* Des items similaires reçoivent des ratings similaires. Pour prédire le rating que donnerait l’utilisateur A à l’item X, on se base sur les ratings donnés par cet utilisateur à des items similaires à X (approche item-based)

La figure 1 illustre ce processus :

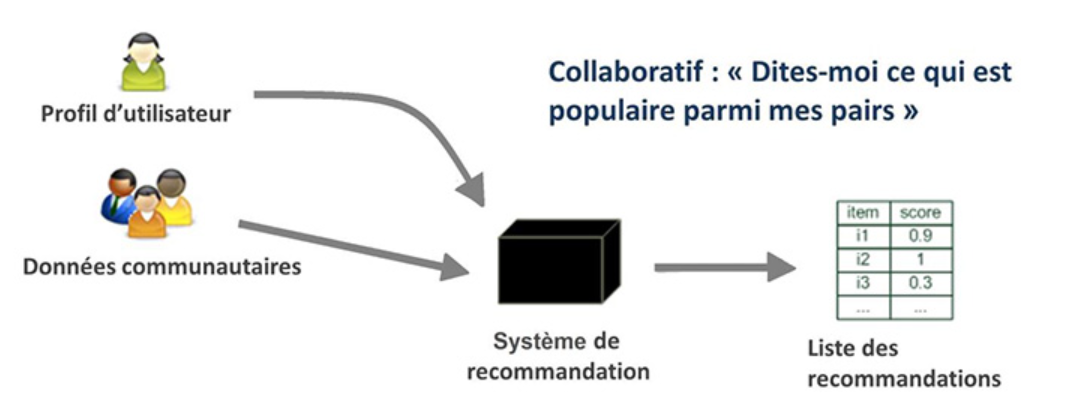


Figure1 : Un système de recommandation collaboratif recommandation

* + 1. Les types de données :

Les données nécessaires pour l’implémentation d’un tel système sont les interactions entre utilisateurs et items (produits, contenus, …). Ces données se présentent sous forme de matrice utilisateurs-items dont chaque élément ui donne une indication de la préférence de l’utilisateur u pour l’item i. Cette matrice est dite creuse car il n’y a pas de valeurs pour beaucoup de ses éléments. Le but de l’algorithme est donc de prédire les cellules vides, afin de connaître le degré d’intérêt d’un utilisateur pour un item. Les valeurs composant la matrice sont typiquement des ratings (notes) et celles-ci peuvent être de plusieurs types :

* Les ratings explicites. L’utilisateur est invité à donner une note à un item (une notation allant de 1 à 5 étoiles est couramment utilisée). Ces ratings explicites sont particulièrement difficiles à obtenir.
* Les ratings implicites. L’utilisateur ne donne pas de note aux items mais celle-ci est calculée à partir de certaines observations telles que le nombre de cliques, le temps de consultation de la page, etc. Dans ce cas, la note calculée est une mesure approximative de l’intérêt de l’utilisateur pour un item.
* Les ratings binaires. La préférence positive ou négative est clairement exprimée. Ex : j’aime/je n’aime pas.
* Les ratings unaires. Seule la préférence positive est exprimée. Ex : achat d’un livre {Oui, Non}.
  + 1. La collection des données

Deux cas de figure peuvent se présenter :

* L’utilisateur est connecté avec son compte. Il est alors aisé de lier les observations collectées lors de l’utilisation du site internet à son compte et de créer un profil utilisateur.
* L’utilisateur n’est pas connecté avec son compte. Les cookies peuvent être utilisés pour indirectement identifier l’utilisateur et le rattacher à son compte ou pour créer un profil anonyme qui suit l’utilisateur lors de ces interactions suivantes. Cependant, ces données ne sont pas tout à fait fiables : l’appareil utilisé peut-être commun à plusieurs personnes, l’utilisateur se connecte avec plusieurs appareils, l’utilisateur se connecte avec un VPN, sans parler des problèmes de respect du RGPD que de telles pratiques peuvent soulever.
  + 1. La recommandation

Pour faire une recommandation d’items à un utilisateur actif (à qui on veut faire la recommandation), il faut au préalable prédire son degré d’intérêt pour les items pour lesquels il n’a pas donné de ratings. Les items obtenant les scores les plus élevés sont proposés à l’utilisateur. Il y a pour cela deux façons de procéder :

* L’approche basée sur les voisins. On trouve des utilisateurs ayant des préférences similaires à celle de l’utilisateur actif c.-à-d. partageant un historique commun. On calcule ensuite le rating manquant en faisant une moyenne pondérée des ratings attribués par les k utilisateurs les plus similaires (fonction de préférence). Cette méthode pose des problèmes de performances quand la matrice utilisateur-item est très large.
* L’approche basée sur l’apprentissage d’un modèle. Elle vise à pallier le problème de performance observé avec la technique des voisins. La méthode la plus couramment utilisée est la factorisation matricielle (SVD, facteurs latents) qui permet de représenter la matrice originale par deux matrices de plus petites tailles à partir desquelles on peut prédire les scores manquants
  + 1. Les avantages et les inconvénients :

Les systèmes de recommandation collaboratifs ont comme avantages :

* D’utiliser les scores d’autres utilisateurs pour évaluer l’utilité des éléments ;
* De trouver des utilisateurs ou groupes d’utilisateurs dont les intérêts correspondent à l’utilisateur courant ;
* Et plus il y a d’utilisateurs plus il y a de scores : meilleurs sont alors les résultats.

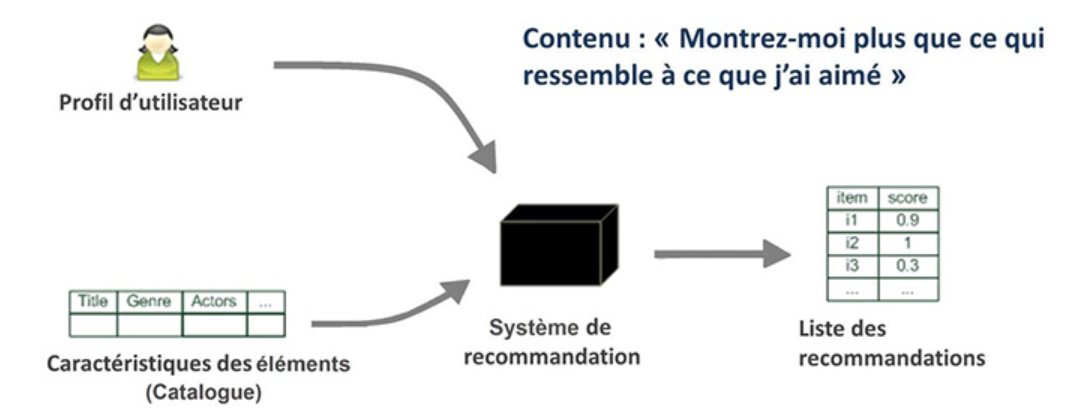
Cependant, de tels systèmes ont aussi des inconvénients :

* Trouver des utilisateurs ou groupes d’utilisateurs similaires est difficile ;
* Le système de recommandation se heurte à la faible densité de la matrice Utilisateurs X Éléments ;
* De plus, il existe aussi le problème du démarrage à froid : lorsqu’un nouvel utilisateur utilise le système, ses préférences ne sont pas connues et lorsqu’un nouvel élément est ajouté au catalogue, personne ne lui a attribué de score ;
* Dans les systèmes avec un grand nombre d’éléments et d’utilisateurs, le calcul croît linéairement ; des algorithmes appropriés sont donc nécessaires ;
* La « non-diversité » : il n’est pas utile de recommander tous les films avec l’acteur Antonio Banderas à un utilisateur qui a aimé l’un d’eux dans le passé.

## Les approches basées sur le contenu (Content based filetring)

Le principe des systèmes de recommandation basé sur le contenu diffère du système collaboratif dans le fait qu’on utilise uniquement les préférences de l’utilisateur actif pour faire les recommandations. L’objectif est de retrouver des items similaires à ceux pour lesquels l’utilisateur a exprimé une préférence dans le passé. Chaque item possède des attributs (content) qui sont exploités pour faire la recommandation. Ces attributs constituent un descriptif de l’item qui se présente sous forme nominale (ex. : genre et auteur d’un livre) ou sous forme de texte libre.

La figure 2 illustre ce processus.



**Figure2 : Un système de recommandation basé sur le contenu**

* + 1. Les types de données :

Les ratings de l’utilisateur actif et les descriptifs des items sont utilisés. Les interactions historiques entre l’utilisateur et les items sont utilisés pour construire un profil.

* + 1. La recommandation :

Pour chaque utilisateur on définit un profil qui exprime les préférences de l’utilisateur. La méthode la plus simple est de construire le profil à partir d’attributs semblables à ceux des items. Le profil utilisateur est alors comparé aux attributs des items et les items les plus similaires sont recommandés à l’utilisateur. Pour faire le match entre le profil utilisateur et les items, on utilise des mesures de similarité telles que la similarité cosinus.

Une méthode plus élaborée consiste à transformer le problème de recommandation en un problème de classification. Pour chaque utilisateur, on entraine un modèle de recommandation (profil) qui prend en entrée les attributs d’un item et qui prédit l’intérêt de l’utilisateur pour cet item

* + 1. Les avantages et les inconvénients

Les systèmes de recommandation basés sur le contenu présentent les avantages suivants :

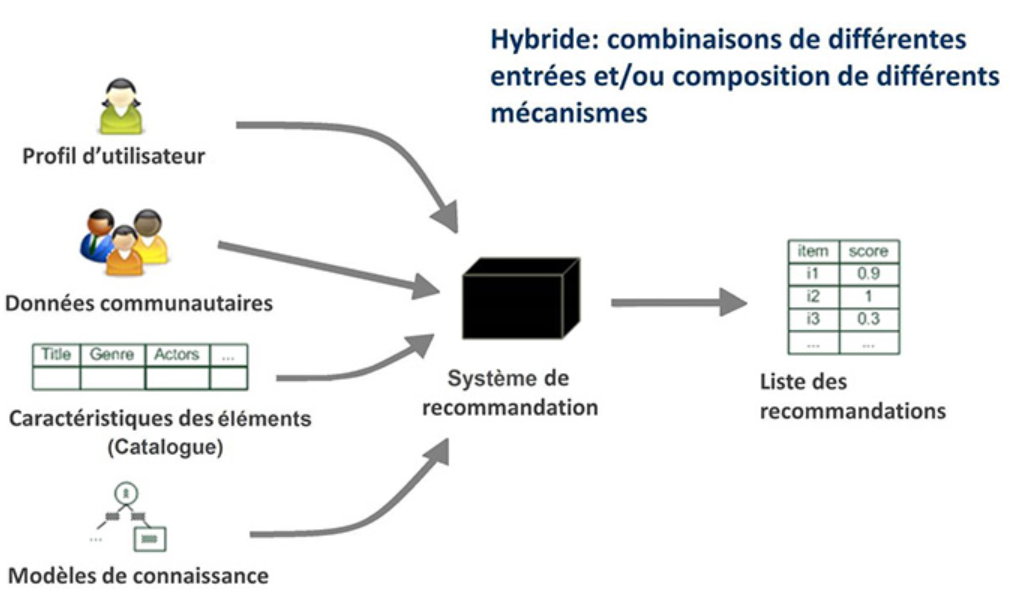
* Ils recommandent des éléments similaires à ceux que les utilisateurs ont aimés dans le passé ;
* Ils prennent en compte le profil des utilisateurs qui est la clé pour avoir les recommandations les plus pertinentes pour chacun ;
* Faire coïncider les préférences de l’utilisateur et les caractéristiques des éléments fonctionne pour de nombreux types de données (textuelles, numériques, etc.) puisqu’on utilise généralement des listes de mots-clés ;
* Les données relatives aux autres utilisateurs sont inutiles ;
* Il n’y a pas de problème de démarrage à froid lorsqu’un nouvel élément est ajouté au catalogue ou de faible densité puisqu’il s’agit de faire coïncider les préférences de l’utilisateur et les caractéristiques des éléments ;
* Il est possible de faire des recommandations à des utilisateurs avec des goûts « uniques » ;
* Il est possible de recommander de nouveaux éléments ou même des éléments qui ne sont pas populaires.

Cependant, de tels systèmes ont aussi des inconvénients :

* Tous les contenus ne peuvent pas être représentés avec des mots-clés (par exemple, les images) ;
* Des éléments représentés par le même ensemble de mots-clés ne peuvent pas être distingués ;
* Les utilisateurs ayant visualisé un très grand nombre d’éléments posent un problème (trop d’informations dans le profil de l’utilisateur à faire coïncider avec les caractéristiques des éléments) ;
* Lorsqu’un nouvel utilisateur commence à utiliser le système, il n’existe pas d’historique ;
* Un risque de « sur-spécialisation » apparaît, c’est-à-dire que l’on se limite aux éléments similaires et que les réponses sont trop homogènes ;
* Les profils des utilisateurs restent difficiles à élaborer et, qui plus est, il faut prendre en compte l’évolution des intérêts de l’utilisateur ;
* Pour que le système produise des recommandations précises, l’utilisateur doit fournir un feedback sur les suggestions retournées mais cela est chronophage pour lui ;
* Finalement, ces systèmes sont entièrement basés sur les scores d’éléments et les scores d’intérêt : moins il y a de scores, plus l’ensemble de recommandations possibles est limité.

## Les approches hybrides

Un système de recommandation hybride utilise des composants de différents types d’approches de recommandation ou s’appuie sur leur logique. Par exemple, un tel système peut utiliser à la fois des connaissances extérieures et les caractéristiques des éléments, combinant ainsi des approches collaboratives et basées sur le contenu.



**Figure3 : Un système de recommandation hybride**

 Les moteurs de recommandation peuvent être combinés de plusieurs façons en fonction de problème :

* En mode monolithique : on implémente un moteur unique de recommandation qui combine des éléments des différentes techniques. Ex. : la similarité des items est calculée en utilisant les contenus, et la prédiction de la préférence de l’utilisateur par rapport à ces items se fait sur base des données des autres utilisateurs ;
* En mode parallèle (ensembles) : plusieurs moteurs de recommandation tournent en parallèle et produisent chacun des résultats qui sont ensuite combinés et pondérés pour fournir la recommandation finale ;
* En mode séquentiel : les résultats d’un moteur de recommandation sont utilisés en entrée du moteur de recommandation suivant.

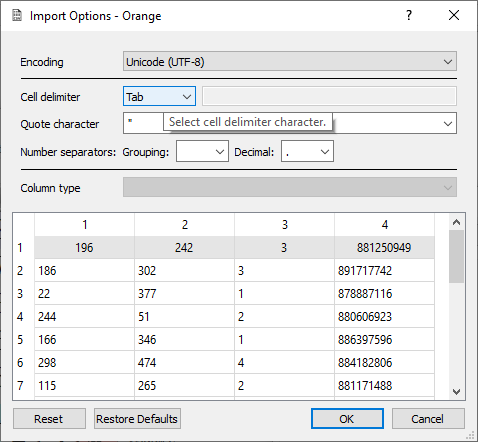
# **Techniques de recommandation basées sur Data mining**

## Dataset utilisée

Orange3-Recommendation peut lire des fichiers au format natif délimité par des tabulations ou charger des données à partir de l'un des principaux types de fichiers de feuilles de calcul standard, tels que CSV et Excel. Le format natif commence par une ligne d'en-tête avec les noms des caractéristiques (colonnes). La deuxième ligne d'en-tête donne le type d'attribut, qui peut être continu, discret, chaîne de caractères ou temporel. La troisième ligne d'en-tête contient des informations métadonnées pour identifier les caractéristiques dépendantes (classe), les caractéristiques non pertinentes (ignorer) ou les caractéristiques métadonnées (méta).

Dans ce mini projet le dataset MovieLens 100k Dataset sera utilisé pour démontrer les approches présentées préalablement.

Voici un aperçu des données collectés dans ce Dataset :

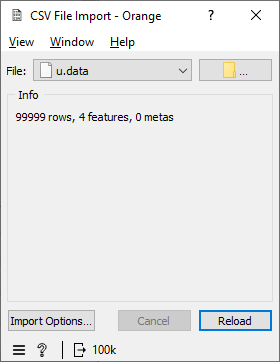


La première colonne représente l’identifiant de l’utilisateur ;

La deuxième colonne représente l’identifiant des films ;

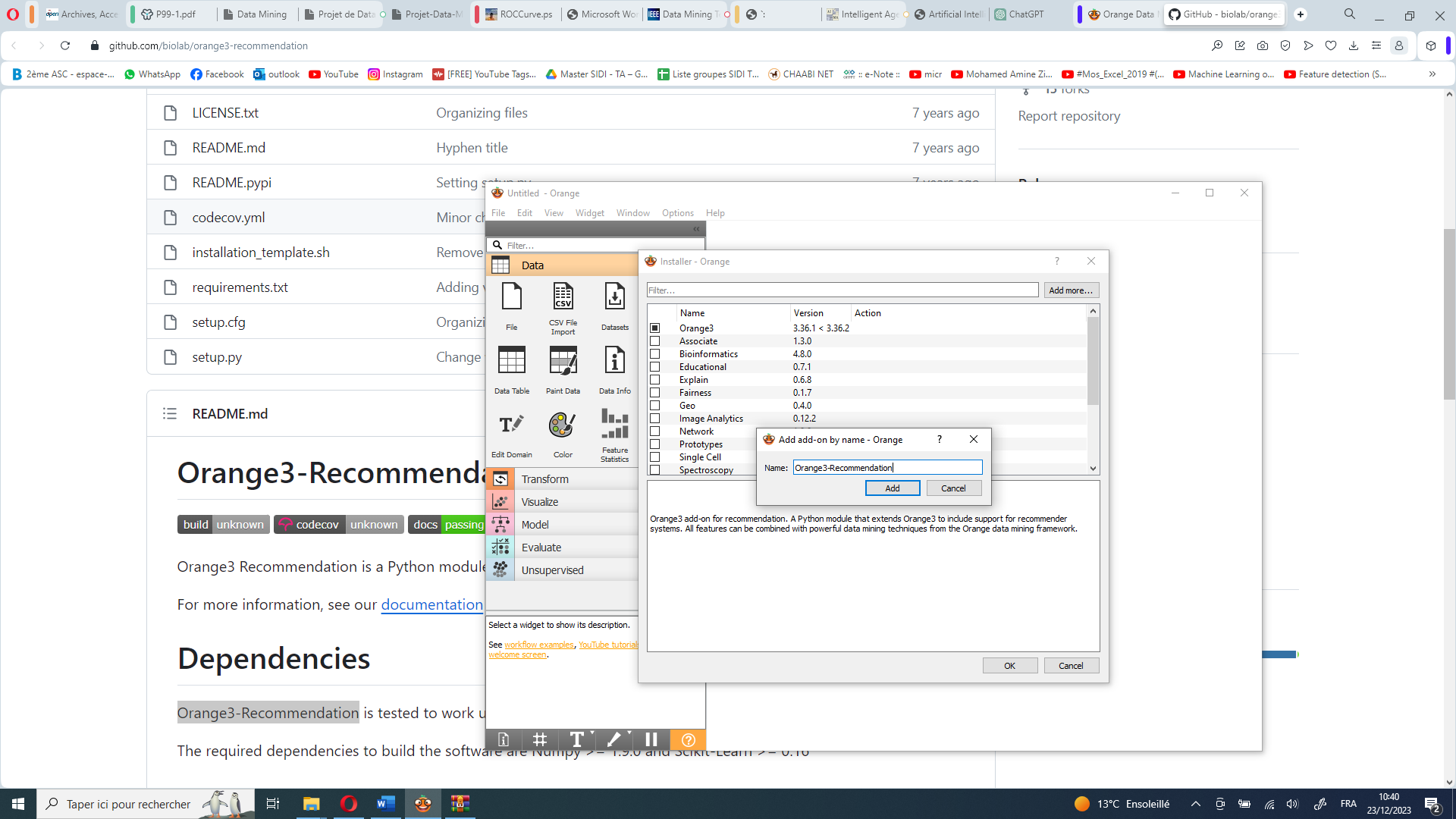
La troisième colonne l’évaluation donnée au film ;

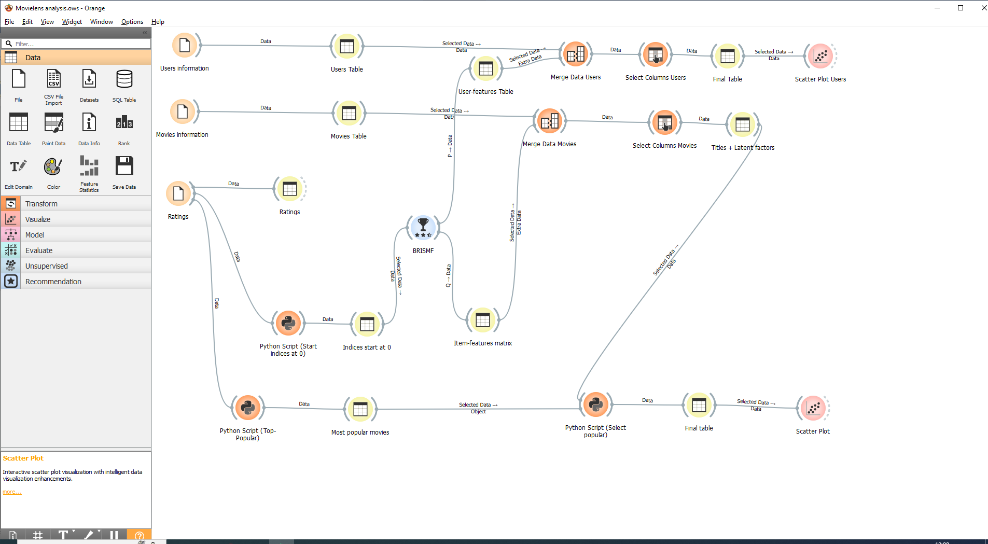
La quatrième colonne représente la date (timestamp) ;



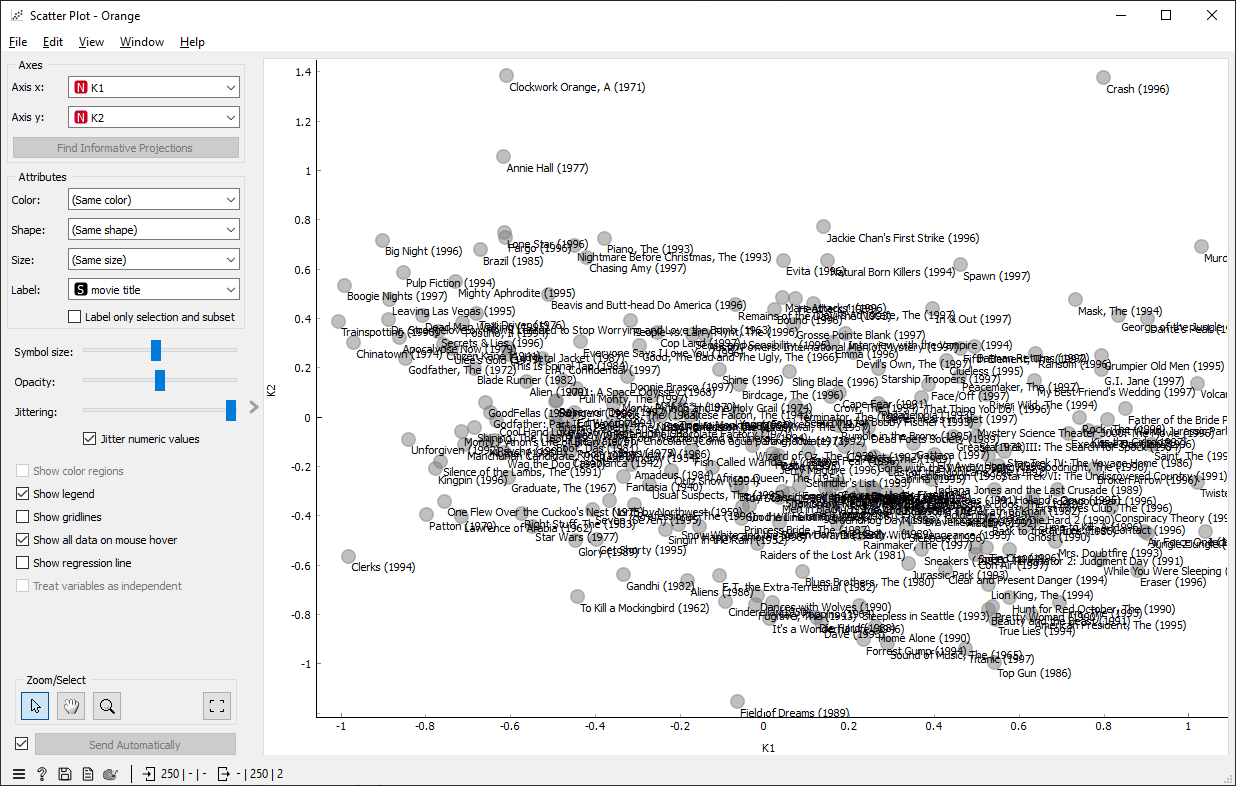
Le nombre d’enregistrement est 10000.

## Application sous Orange

Avant de commencer, il faut installer l’Add-on ‘Orange3-Recommendation’

Après vous devez redémarrez orange pour appliquer les changements,  
Prenons le flux de travail suivant pour effectuer une analyse de base sur les résultats obtenus en factorisant les matrices de caractéristiques utilisateur et d'éléments avec BRISMF sur l'ensemble de données MovieLens 100k.

L'une des analyses à effectuer consiste à représenter graphiquement les films les plus populaires à travers les deux premiers vecteurs de la décomposition de la matrice. Ensuite, essayer de trouver des clusters, les ajuster un peu et identifier des relations croisées (par exemple, homme/femme par rapport à action/drame).



## Application avec Python

Dans cette partie, on s’est concentré sur la méthode SVD

1. La décomposition en valeurs singulières :

La décomposition en valeurs singulières (SVD) est une technique mathématique fondamentale utilisée dans l'analyse de matrices. Elle permet de décomposer une matrice en trois matrices plus simples, ce qui facilite la compréhension des structures sous-jacentes des données. En contexte d'analyse de recommandation de films, la SVD peut être appliquée à une matrice utilisateur-film, où les utilisateurs et les films sont représentés par des vecteurs dans un espace latent. En utilisant les deux premiers vecteurs résultants de la SVD, on peut visualiser les relations entre les films et les utilisateurs dans un espace bidimensionnel. Cette représentation offre des possibilités d'analyse, telles que l'identification de groupes de films similaires, la suggestion de recommandations basées sur des préférences similaires, et l'exploration des relations entre différents attributs comme le genre des films et les préférences des utilisateurs.

Avant de procéder il faut installer la bibliothèque surprise :

pip install surprise

Après importer les bibliothèques suivantes :

from surprise import SVD  
from surprise import Dataset  
from surprise import accuracy  
from surprise.model\_selection import train\_test\_split  
from collections import defaultdict

charger le dataset :

data = Dataset.load\_builtin('ml-100k')

Diviser les données en training et test :

trainset, testset = train\_test\_split(data, test\_size=.25)

Instancier un objet SVD et commencer l’apprentissage :

algo = SVD()   
algo.fit(trainset)   
predictions = algo.test(testset)   
accuracy.rmse(predictions)

Après le test, l’erreur RMSE affiché et :

RMSE: 0.9390

Ensuite en affiche les films avec leurs recommandations :

def get\_top\_n(predictions, n=10):  
 top\_n = defaultdict(list)  
 for uid, iid, true\_r, est, \_ in predictions:  
 top\_n[uid].append((iid, est))  
 for uid, user\_ratings in top\_n.items():  
 user\_ratings.sort(key=lambda x: x[1], reverse=True)  
 top\_n[uid] = user\_ratings[:n]  
 return top\_n  
  
top\_n = get\_top\_n(predictions, n=10)  
for uid, user\_ratings in top\_n.items():  
 print(uid, [iid for (iid, \_) in user\_ratings])

On se limite aux 4 premières lignes pour ne pas trop charger se document :

757 ['651', '7', '183', '100', '196', '265', '202', '175', '228', '742']

184 ['488', '9', '480', '134', '192', '170', '478', '52', '213', '1137']

764 ['50', '173', '633', '176', '117', '531', '216', '275', '31', '15']

897 ['210', '705', '435', '646', '203', '496', '659', '378', '136', '182']

On obtient les 10 films susceptibles d’intéresser chaque utilisateur.

# **Conclusion**

L’utilisation du data mining, et notamment de techniques telles que la décomposition en valeurs singulières, dans les systèmes de recommandation apporte une valeur significative en matière d'analyse et de personnalisation. Ces méthodes permettent de découvrir des motifs complexes au sein des données utilisateur-film, facilitant ainsi la prédiction des préférences individuelles. En exploitant des algorithmes de data mining, les systèmes de recommandation peuvent offrir des suggestions plus précises et personnalisées, améliorant ainsi l'expérience utilisateur. De plus, ces techniques permettent de découvrir des relations subtiles entre différents attributs, comme les genres de films et les préférences des utilisateurs, ouvrant la voie à une compréhension plus approfondie des comportements et des goûts. Toutefois, il est essentiel de considérer les défis liés à la confidentialité des données et à la transparence des algorithmes, afin d'assurer une utilisation éthique et responsable du data mining dans le domaine des recommandations.