الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي والبحث العلمي
République Algérienne Démocratique et Populaire
Ministères de l'Enseignement Supérieure et de la Recherche

Université Badji Mokhtar -Annaba
Faculté des Sciences de l'ingénieur
Département de l'informatique

جامعة باجي مختار -عنابة
كلية علوم المهندس
قسم الاعلام الآلي

Mémoire

Présenté en vue de l'obtention du Diplôme de Master

Intitulé

Principe de la factorisation matricielle dans un système de recommandation basé sur le filtrage collaboratif

Domaine : Mathématiques-Informatique

Filière : Informatique

Spécialité : Gestion et Analyse des données massive

Par : Melle/ Rouabhia sumia

Jury d'évaluation

Baki Halima	Pr	UBMA	President
Mohamed Ben Ali	Pr	UBMA	Encadrant
Sari Toufik	Dr	UBMA	Examinateur

Année Universitaire 2020/2021

Remerciement

Nous tenons tout d'abord à remercier Dieu le tout puissant et miséricordieux, qui nous a donné la force et la patience d'accomplir ce travail

Je souhaitais adresser mes remerciements les plus sincères à mon encadreur de recherche, Madame Mohamed Ben Ali Yamina pour les conseils, les encouragements et toute l'aide qu'il m'a apporté durant ce projet de recherche.

Nos vifs remerciements vont également aux membres du jury qui ont accepté de juger notre travail.

Je tiens à remercier aussi toutes les personnes rencontrées lors des recherches effectuées et qui ont accepté de répondre à mes questions avec gentillesse.

J'exprime ma gratitude à mes parents pour leur contribution, leur soutien et leur patience.

Merci à tous et à tout

Dédicace

Je dédie ce mémoire à mes très chers parents.

A l'âme de mon petit frère Noureddine

A tout personnes rencontrées lors des recherches effectuées et qui ont accepté de répondre à mes questions avec gentillesse.

RÉSUMÉ

Le travail présenté dans ce mémoire se situe dans le domaine des systèmes de recommandation qui est devenu une méthodologie dominante dans la majorité des applications Web, y compris les sites web de commerce comme le Netflix,

Les systèmes de recommandation sont utilisés avec succès pour fournir des items exemple (les films, la musique, les livres, les nouvelles, les images) adaptées aux préférences des utilisateurs. Parmi les approches proposées, nous utilisons l'approche de filtrage collaboratif qui consiste à trouver l'information qui satisfait l'utilisateur en utilisant les évaluations des autres utilisateurs, l'avantage de cette approche est le fait qu'elle ne nécessite pas un nouvel algorithme pour le calcul des prédictions. Nous allons appliquer l'algorithme de factorisation matricielle qui se base sur la méthode de réduction dimensionnelle et plus précisément SVD (décomposition de la valeur singulière).

Mots-clés : Système de recommandation, Filtrage collaboratif, Factorisation matricielle, Variable latente, Information implicite, Information Explicite, ACP, NNM, BFM, SVD.

ABSTRACT

The work presented in this dissertation is in the area of recommendation systems which has become a dominant methodology in the majority of web applications, including e-commerce websites like Netflix,

Recommendation systems are used successfully to deliver items (ex: movies, music, books, news, pictures) tailored to user preferences. Among the proposed approaches, we use the collaborative filtering approach which consists in finding the information that satisfies the user by using the evaluations of other users, the advantage of this approach is the fact that it does not require a new algorithm for calculating predictions. We will apply the matrix factorization algorithm which is based on the dimensional reduction method and more precisely SVD (singular value decomposition).

Keywords: Recommendation system, Collaborative filtering, Matrix factorization, Latent variable, Implicit information, Explicit information, PCA, NNM, PFM, SVD.

الملخص

العمل المقدم في هذه الرسالة هو في مجال أنظمة التوصية او نظام الترشيحات كما يطلق عليها في بعض المراجع المترجمة إلى اللغة العربية، حيث أصبحت هذه الانظمة منهجية سائدة في غالبية تطبيقات الويب (WEB)، بما في ذلك مواقع التجارة مثل Netflix وتُستخدم أنظمة التوصيات بنجاح لتوصيل او اقتراح العناصر على المتصفح للمواقع غبر الانترنيت (مثل الأفلام والموسيقى والكتب والأخبار والصور) والتي تصمم حسب تفضيلات المستخدم.

سنطرق في رسالتنا هذه إلى نظام التصفية التعاوني بالخصوص وسنستند إليه في الجزء التطبيقي كونه أبرز نهج تستخدمه انظمة التصفية والذي يتمثل في العثور على المعلومات التي ترضي المستخدم باستخدام تقييمات المستخدمين الآخرين، وتتمثل ميزة هذا النهج في حقيقة أنه لا يتطلب خوارزمية جديدة لحساب التنبؤات.

سنقوم بتطبيق خوارزمية عامل المصفوفة على مجوعة البيانات التي اخرناها للدراسة، وتعتمد هذه الخوارزمية على طريقة تقليل الأبعاد وبشكل أكثر دقة سنطبق طريقة SVD (تحلل القيم الفردية).

الكلمات المفتاحية: نظام التوصية، التصفية التعاونية، عامل المصفوفة، المتغير الكامن، المعلومات الضمنية، المعلومات الصريحة،

Table des matières

Table des Figures

Table des Formules

Abréviation :

Pour des raisons de lisibilité, la signification d'un acronyme ou d'une abréviation n'est en général rappelée que lors de sa première utilisation dans le texte d'un chapitre. Par ailleurs, nous employons le terme français ou le terme anglais suivant l'usage le plus répandu.

SR Systèmes de Recommandation

FC Filtrage Collaboratif

FBC Filtrage Basé Contenu

FM Factorisation Matricielle

NMF Factorisation en Matrice Non Négative

SVD Singular Value Decomposition

PMF Probabilistic Matrix Factorization

MAE Mean Absolute Error

RMSE Root Mean Squared Error

1- Introduction générale

1. Contexte

Avec l'avènement du web et les évolutions technologiques, les services de connectivité, d'information et de communication sont de plus en plus consommés et la recherche d'une information consiste à trouver les documents pertinents adaptés au besoin de l'utilisateur donc la volonté de pallier le problème de surcharge d'information conduit à la naissance des systèmes de recommandation, l'engouement des chercheurs au cours de ces vingt dernières années a fait grandement avancer les choses, d'où viens les systèmes développés pour cette raison, qui suggèrent des éléments explorant les préférences de l'utilisateur, les aidant à résoudre ce problème de surcharge.

Le succès des outils de recommandation tels que Netflix, qui nous suggère des films, ou encore Amazon, qui nous recommande des livres, repose sur trois approches principales (filtrage collaboratif, filtrage sur le contenu et le filtrage hybride qui combine les deux précédentes).

La plupart des études sur les systèmes de recommandation utilisent les notes des utilisateurs pour trouver la similarité entre les items et ne considère pas le contexte dans lequel se trouve l'utilisateur au moment de noter.

L'utilisation des systèmes de recommandation par ces trois approches (système basé sur le contenu, filtrage collaboratif et système hybride) est devenue une nécessité urgente vu qu'ils permettent de fournir l'information pertinente avec moins d'effort et dans un délai de réponse satisfaisant.

De nombreux travaux sont intéressés aux systèmes de recommandations mais certains défis restent à lever encore aujourd'hui comme le démarrage à froid qui désigne un manque d'information lors de l'ajout d'un nouvel utilisateur ou d'un nouvel item au système.

Le travail présenté dans ce mémoire s'articule autour des systèmes de recommandation et plus particulièrement la factorisation matricielle dans le filtrage collaboratif nous souhaitons dans le chapitre implémentation créer un système de recommandation simple pour une base de données de films qui sera capable de prédire les variables latentes.

L'objectif principal des méthodes de factorisation de grandes matrices est d'extraire des variables latentes qui permettent d'expliquer les données dans un espace de dimension inférieure.

2. Problématique :

Notre problématique c'est de prédire aux utilisateurs des items qui représentent des films dans notre mémoire, à base de l'historique d'autres utilisateurs qui ont des préférences similaires dans le cadre de filtrage collaboratif.

3. Objectif :

Notre objectif est de construire un système de recommandation basé sur le filtrage collaboratif capable de prédire des items (films) pour l'utilisateur à partir des préférences d'autres utilisateurs similaires à ses préférences, nous utilisons pour réaliser cette implémentation le langage python.

4. Organisation du mémoire

Le mémoire est structuré en trois chapitres, d'abord une introduction générale sur les systèmes de recommandation puis nous avons trois chapitres comme suit :

Chapitre 01 : consiste à présenter quelques concepts de base des systèmes de recommandation.

Chapitre 02 : est dédié au problème de la factorisation matricielle son fonctionnement et les méthodes de réduction dimensionnelle.

Chapitre 03 : consiste à la conception et implémentation de notre algorithme de factorisation en utilisant le SVD.

Et on conclut par une conclusion générale.

CHAPITRE : 01

Les systèmes de recommandation

I- Introduction

Les systèmes de recommandation sont développés en parallèle avec le web, se sont une classe étendue d'applications Web qui implique la prédiction des réponses des utilisateurs aux opinions, donc ce sont des outils et techniques logiciels fournissant des suggestions d'items à un utilisateur. Le système de recommandation doit connaître les préférences de chaque utilisateur. Il tente, alors, d'acquérir les informations nécessaires pour construire des profils d'utilisateurs. En particulier, il exploite les traces laissées par les utilisateurs eux-mêmes. Il collecte les traces laissées explicitement ou implicitement. [01]

En d'autres termes les systèmes de recommandation aident à surmonter la surcharge d'information. En fournissant des suggestions personnalisées basées sur un historique de goûts et dégoûts d'un utilisateur. Il y a trois approches des systèmes de recommandation (Filtrage collaboratif, système basé sur le contenu et système hybride) [02]

- **Système Basée sur le contenu** : Le système recommande des items qui sont similaires à ceux que l'utilisateur a aimés dans le passé.
- **Système Basé sur le Filtrage collaboratif** : L'implémentation la plus simple et originale de cette approche est de recommander à un utilisateur actif les items que d'autres utilisateurs avec des goûts similaires ont aimés dans le passé.
- **Système hybride** : combine les deux filtrage collaboratif et basé sur le contenu.[03]

II- Historique des systèmes de recommandation

Les préludes des systèmes de recommandation découlent de recherches menées dans la construction de modèles représentant les choix d'utilisateurs. Ces recherches sont issues de domaines distincts tels que la recherche documentaire, les sciences de gestion et marketing, les sciences cognitives et les théories d'approximation. La recommandation peut être comparée à un dialogue entre une personne experte d'un domaine et l'autre d′désireuse d'acquérir des informations dans ce domaine. Plus concrètement, un bibliothécaire va pouvoir, en fonction des gouts d'un de ses clients, proposer une liste d'ouvrages à ce dernier qui ne sera autre qu'une recommandation au sens des systèmes de recommandation.[04]

La problématique des systèmes de recommandation a émergé comme un domaine de recherche propre dans les années 90. L'année 1992 voit l'apparition du système de recommandation de documents Tapestry et la création du laboratoire de recherche GroupLens qui travail explicitement sur le problème de la recommandation automatique dans le cadre des forums de news de Usenet. Tapestry avait pour but de recommander à des groupes d'utilisateurs des documents sur les

newsgroups susceptibles de les intéresser. L'approche choisie est de type « plus proche voisin » à partir de l'historique de l'utilisateur. On parle alors de filtrage collaboratif, comme une réponse au besoin d'outils pour le filtrage de l'information énoncé à la même époque. Ce filtrage collaboratif n'avait pas le sens qu'il a maintenant, il s'agissait en fait d'une action collaborative des utilisateurs qui recommandaient aux autres utilisateurs des documents en attribuant des notes à ces documents selon certains critères. [05]

En 1995 apparaissent successivement Ringo, un système de recommandation de musique, basé sur les appréciations des utilisateurs et 'Bellcore' un système de recommandation de vidéos. La même année, 'GroupLens' crée la société 'Net Perceptions 'dont un des premiers clients a été Amazon. C'est en 1996 que le premier workshop dédié aux systèmes de recommandation s'est avéré qui est ensuite devenue la conférence ACM RecSys, référence dans le domaine. [06]

En 2001, la notion du filtrage collaboratif basé item a été introduite par Sarwar et *al*. Elle a étendu le champ de popularité des systèmes de recommandations du secteur académique vers le secteur commercial.[07]

III- Domaine d'application des systèmes de recommandation

Ces dernières années, les systèmes de recommandation ont fait l'objet de nombreuses recherches, ils ont pour but de recommander aux utilisateurs des plateformes avec des contenus qui facilitent la navigation des utilisateurs parmi de larges catalogues de produits, ces systèmes sont devenus un enjeu majeur pour ces plateformes. Parmi les services en ligne qui reposent sur la recommandation c'est le Netflix. [08]

Netflix : est un service de streaming sur abonnement qui permet à nos utilisateurs de regarder des séries TV et des films sans publicité sur un appareil connecté à Internet.

Nous pouvons également télécharger des séries TV et des films sur notre appareil iOS, Android ou Windows 10 pour les regarder hors ligne.[08]

Netflix, comme plusieurs plateformes nous demande de passer par une étape de sélection de nos intérêts avant de pouvoir utiliser l'application. Cet effort actif demandé à l'utilisateur permet de résoudre en partie le problème du démarrage à froid. [09]

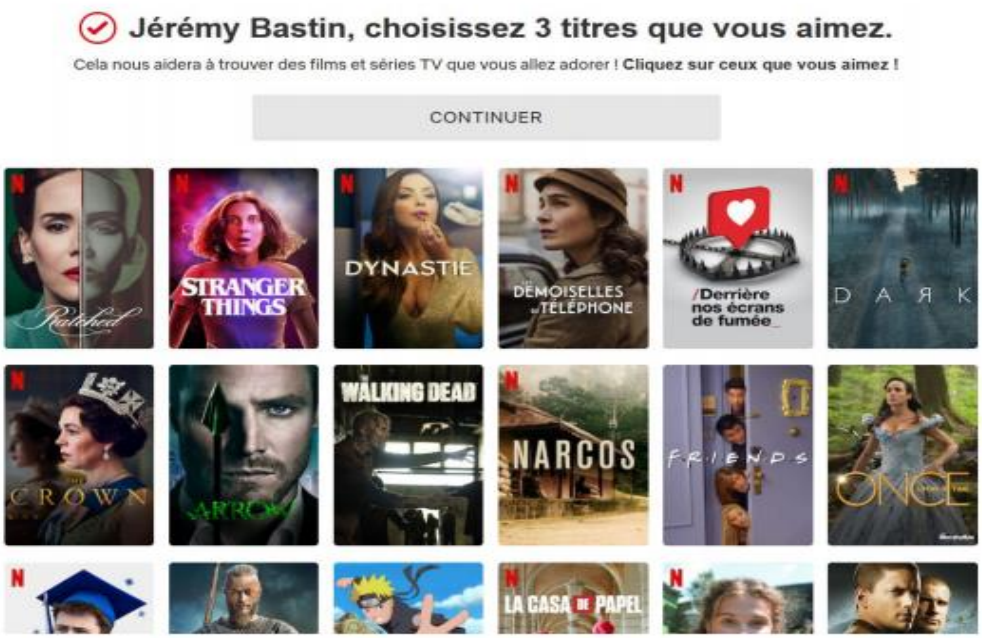


Figure 01 : Sélection de préférences de l'utilisateur lors de la création d'un profil sur la plateforme Netflix [09]

La recommandation est utilisée dans des différents secteurs industriels a fin de :

- Amélioration de l'expérience utilisateur.
- Augmentation continue des performances clés (durée de visionnement, temps de lecture, panier moyen, raccourcissement des délais de recherche de contenus/produits, etc.)
- Gestion d'un volume croissant de données impossible à traiter manuellement
- Analyse pointue des données pour des recommandations personnalisées pertinentes
- Automatisation du filtrage des données. [10]

La figure suivante représente les différents secteurs utilisant la recommandation.

Figure 02 : Système de recommandation dans les différents secteurs industriels [10]

IV- Le fonctionnement des systèmes de recommandation

Le fonctionnement des systèmes de recommandation a été identifié en 3 étapes :

La première consiste à collecter des informations sur les utilisateurs, celles-ci peuvent être implicites ou explicites. En général, les utilisateurs offrent une quantité importante d'informations les concernant en échange des services proposés. Les données explicites sont celles transmises consciemment et volontairement par l'utilisateur elles correspondent les réponses à des enquêtes et remplir le questionnaire etc.

Tandis que les données implicites sont collectées sans que l'utilisateur n'en soit réellement Conscient. Cela correspond aux interactions des utilisateurs, il s'agit plus souvent à des historiques de navigation de la durée passée sur un site.

La deuxieme étape représente le réel système de recommandation. Il s'agit d'analyser et de transformer les données de manière à pouvoir les exploiter et comme résultat nous obtenons une base de données dans laquelle on retrouve généralement un ensemble de notes attribuées par des utilisateurs à des items(éléments). Ce que nous appelons ici « items » correspond à des produits ou services que l'utilisateur va pouvoir évaluer tels qu'un film, une musique, un bien qu'il a pu acheter, etc. Cette base de données peut alors être divisée en deux sous-groupes : un test set et un training set. L'ensemble des éléments repris dans le training set va permettre d'établir le modèle et ceux appartenant au test set serviront à l'évaluer.

La troisième étape permet de fournir des recommandations aux utilisateurs ainsi que de faire des prédictions concernant leurs évaluations à propos de différents items. Le but étant de fournir aux utilisateurs l'expérience la plus intéressante [11]

Ces données sont désignées par une matrice d'utilité, où chaque valeur indique la préférence d'un utilisateur pour un article couvrant toutes les combinaisons possibles d'élément utilisateur. Les valeurs sont constituées à partir d'un ensemble ordonné, par exemple, 1-10 représentant le niveau d'évaluation que l'utilisateur a donné pour un élément spécifique. La préférence pour l'élément peut ne pas être connue. [12]

En d'autres termes :

Soit **C** l'ensemble de tous les utilisateurs, **I** l'ensemble de tous les éléments possibles qui peuvent être recommandés (comme des livres, des films ou des restaurants) et soit $\boldsymbol{u}$ une fonction qui mesure l'utilité d'un élément $\boldsymbol{i}$ à l'utilisateur $\boldsymbol{c}$, c'est-à-dire $\boldsymbol{u} : \mathbf{C} \times \mathbf{I} \rightarrow \mathbb{R}$

Dans les systèmes de recommandation, l'utilité d'un élément est habituellement représentée par un score qui indique comment un utilisateur spécifique a aimé un élément particulier.

Exemple :

- On considère le tableau suivant (où les lignes représentent les 'user' et les colonnes représentent les 'items' film.), les utilisateurs Mohamed, Ali, Sarah et Linda sont susceptibles d'avoir notés. Par exemple, l'utilisateur Mohamed à donner au film « la bataille d'Alger » le score de 3 (sur 10) Nous obtenons la matrice C × I : [13]

user \ item	El-Rissala	La bataille d'Alger	Le Clandestin	Studio Sport	Notre Santé
Mohamed	5	3	?	2	9
Ali	?	1	2	3	6
Sarah	3	5	6	?	7
Linda	5	3	9	4	?

Figure 03 : Exemple d'une matrice de notes

Dans les systèmes de recommandation, l'utilité d'un élément est généralement représentée par un score qui indique comment un utilisateur particulier a aimé un élément particulier.

Le problème principal à résoudre est l'estimation de scores pour des éléments qui n'ont pas encore été évalués par un utilisateur. Le nombre d'éléments ainsi que le nombre d'utilisateurs du système peuvent être très importants ; il est, de ce fait, difficile que chaque utilisateur puisse voir tous les éléments ou que chaque élément soit évalué par tous les utilisateurs. Lorsqu'il est possible d'estimer des scores pour les éléments non encore évalués, les éléments ayant les scores estimés les plus élevés peuvent être recommandés à l'utilisateur.[13]

V- Défis des systèmes de recommandation

3- Le démarrage à froid (cold-start problem)

Lors de rencontre d'un nouvel élément qui n'a pas été précédemment évalué, que ce soit un produit ou un utilisateur, il doit être gérer comme étant un cas spécial, le problème du démarrage à froid survient lorsqu'il n'y a pas assez d'information pour être en mesure d'émettre des recommandations par le système. Par exemple, lorsqu'un nouvel utilisateur s'ajoute au système, il n'a pas encore d'historique. Évidemment, il est impensable d'offrir des recommandations personnalisées à un utilisateur qui n'est pas, ou très peu, connu du système.

Afin de surmonter ce problème, le système peut décider de ne pas émettre de recommandations à un utilisateur dont le système n'a pas atteint un certain nombre d'informations requises, car il n'est pas capable de cibler ses préférences avec le peu d'information disponible.

Pour éviter ce problème une solution plus bénéfique à la fois pour l'utilisateur et pour le système de recommandations est d'utiliser d'autres sources d'informations. En ayant ainsi un système hybride, par exemple un système utilisant à la fois le filtrage collaboratif et le filtrage par contenu, il est possible d'aider l'utilisateur à faire un choix judicieux en lui offrant des recommandations, peu importe s'il est nouveau ou non.

Un autre problème similaire au démarrage à froid pour les produits est le « **long tail problèm** ». La problématique s'explique comme étant une file où les produits les plus populaires se retrouvent à la tête de la file et que les produits les moins populaires, ou encore les nouveaux produits dans le système, se retrouvent dans la queue.

La solution proposée afin de surmonter ce problème est d'utiliser la segmentation des produits de la queue afin d'estimer les cotes manquantes. Après avoir trouvé le point de séparation entre la tête et la queue, le système effectue une segmentation des produits en utilisant un algorithme expectation-minimisation et crée des modèles de prédiction pour chacun des segments avec des séparateurs à vaste marge. [14]

4- Données dispersées

Ce défi, similaire au démarrage à froid, se distingue du fait que le problème n'est pas tant le manque de données causé par un nouvel utilisateur ou un nouveau produit, que la diversité des produits et des préférences des utilisateurs. Ce problème suit le même principe que le problème des grandes dimensionnalités, la malédiction de la dimensionnalité.

Par exemple dans le domaine de la recommandation de films, si un utilisateur a des préférences particulières, il ne sera pas évident pour un système utilisant un filtrage collaboratif de trouver d'autres utilisateurs avec des préférences similaires. Aussi, du côté des produits, si très peu d'utilisateurs cotent certains films, même avec de très bonnes cotes, ceux-ci ne seront recommandés que très rarement.

Comme dans le cas du démarrage à froid, des recherches ont montré qu'en combinant d'autres sources d'informations, par exemple des données démographiques, il était possible d'obtenir une meilleure mesure de similarité entre les individus.

La résolution de ce problème est d'utiliser la décomposition en valeurs singulières, afin de réduire le nombre de dimensions et ainsi, être en mesure d'offrir une mesure de similarité qui rejoint un plus grand nombre d'utilisateurs.[14]

VI- Les approches des systèmes de recommandation

Les systèmes de recommandation ont été étudiés dans des domaines divers et variés comme le web, le e-commerce et bien d'autres. 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. La figure suivante représente les deux types de recommandation. [15]

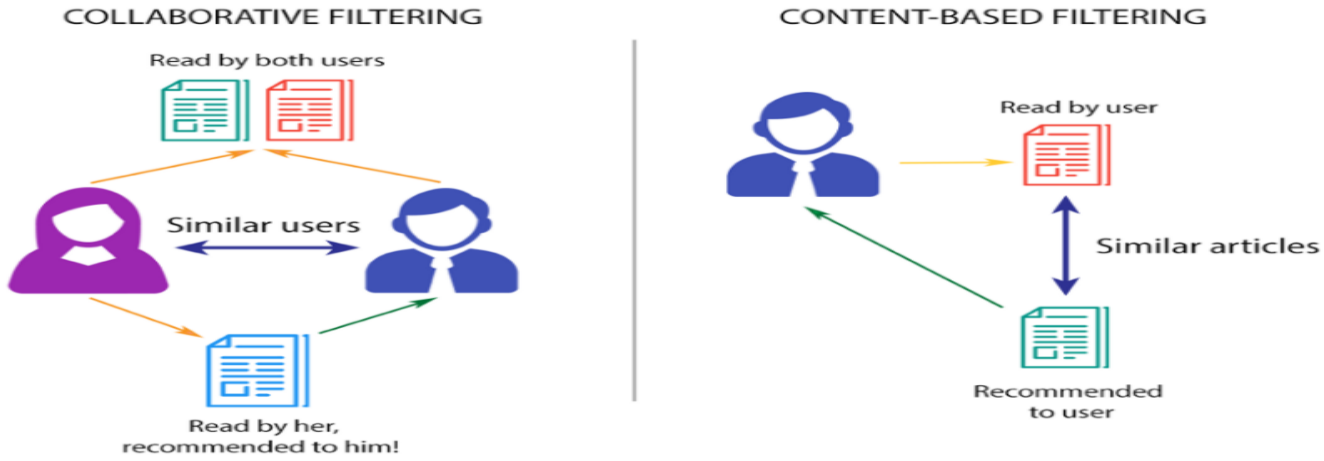


Figure 04 : les deux approches des systèmes de recommandation.[15]

5) Système de recommandation basé sur le contenu

Les recommandations fondées sur le contenu (Content Based) sont construites à partir des données ou informations disponibles sur les articles. La première information c'est les détails de l'article et la deuxième est le résumé des choix de l'utilisateur, les méthodes basées sur le contenu émettent des recommandations en analysant la description des articles qui ont été évalués par l'utilisateur. Sur la base de préférences de l'utilisateur, les mots-clés sont associés aux éléments.

Ces techniques suggèrent des éléments qui sont similaires à ceux que l'utilisateur a aimés dans le passé cette similarité des items est calculée en se basant sur les caractéristiques associées aux items comparés. Par exemple, si l'utilisateur a noté positivement un film qui appartient au genre « comédie », donc le système peut fournir des recommandations de films de ce genre. [16]
Un système de filtrage basé sur le contenu sélectionne des éléments en fonction de la corrélation entre le contenu des éléments et les préférences de l'utilisateur par contre dans le filtrage

collaboratif, le système choisit des éléments en fonction de la corrélation entre des personnes ayant des préférences similaires.

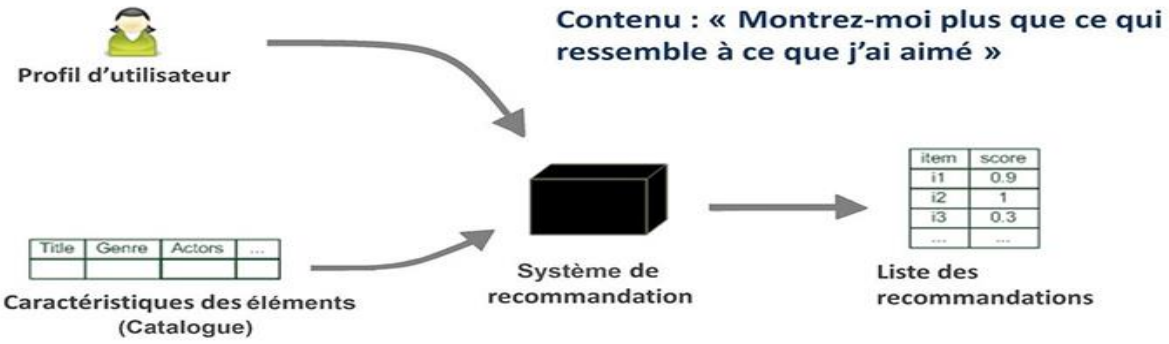


Figure 05 : Un système de recommandation basé sur le contenu [12]

6) Systèmes de filtrage collaboratif

Le filtrage collaboratif (Collaborative Filtering) est l'un parmi les technologies les plus populaires dans le domaine des systèmes de recommandation, il est fondé uniquement sur l'historique de consommation des utilisateurs.

On note ***U*** le nombre d'utilisateurs et ***I*** le nombre d'articles disponibles dans le catalogue. Les interactions passées entre les utilisateurs et les articles sont stockés dans une matrice **Y** de taille ***U*** × ***I.*** Chaque coefficient $\mathbf{y_{ui}}$ de la matrice correspond au retour d'un utilisateur $u \in \{1, \ldots, U\}$ sur un article (*item* en anglais) $i \in \{1, \ldots, I\}$. Généralement, la matrice **Y** est de très grande dimension (des millions d'utilisateurs qui interagissent avec des millions d'articles). Cependant, cette matrice est généralement très creuse puisque les utilisateurs n'interagissent qu'avec un nombre limité d'articles du catalogue. [17]

Il existe deux approches pour faire des recommandations en filtrage collaboratif.

a) **Memory-based :** cette approche est fondée sur la « mémoire » du système. Nous utilisons alors des heuristiques fondées sur **Y** qui permettent de faire des prédictions. Dans cette catégorie, on retrouve notamment les méthodes dites des plus proches voisins. Grossièrement, ces approches consistent à comparer les utilisateurs entre eux à l'aide de mesures de corrélation (exemple la corrélation de Pearson). Ainsi, pour chaque utilisateur, on pourra se baser sur l'historique de consommation de ses voisins pour lui proposer de nouveaux contenus. [17]

b) **Model-based :** c'est une approche fondée sur l'apprentissage d'un modèle, Cette catégorie se décompose en deux étapes.

- ✓ D'abord, on infère les paramètres du modèle à partir des observations passées (les retours des utilisateurs). Apres nous utilisons ces paramètres pour faire des prédictions sur les prochains retours d'utilisateurs.
- ✓ Dans cette thèse on traitera des méthodes dites à facteurs latents. Elles regroupent : les méthodes de factorisation de matrices et les réseaux de neurones.

Le filtrage collaboratif souffre du problème de démarrage à froid. [17]

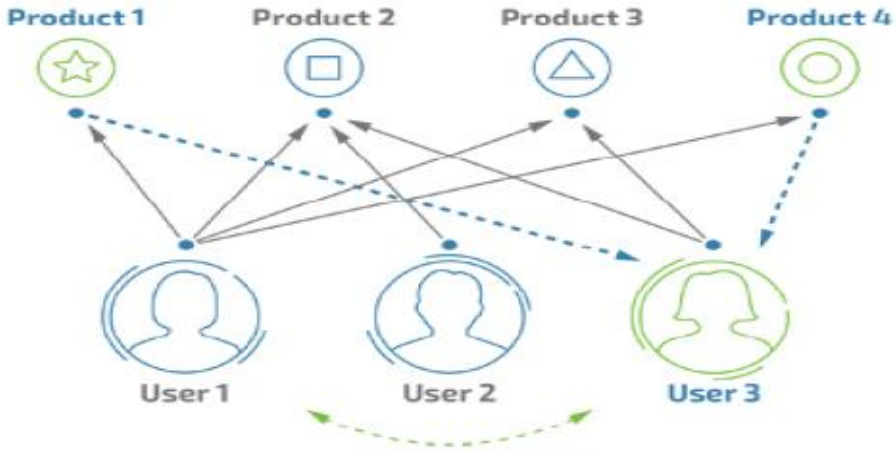


Figure 06 : filtrage collaboratif [18]

7) L'algorithme de recommandation de filtrage collaboratif

Des algorithmes tentent de prédire ce que l'utilisateur aimera en cherchant d'autres utilisateurs qui ont les mêmes comportements que l'utilisateur à qui nous souhaitons faire des recommandations. L'idée de base est donc de dire que si des utilisateurs ont partagé des mêmes intérêts dans le passé, il y a de fortes chances qu'ils partagent aussi les mêmes goûts dans le futur.

Les étapes de base de l'algorithme de recommandation de filtrage collaboratif

Etape 1 : Établissez la matrice d'évaluation des éléments utilisateur R **(**m utilisateurs et n éléments)

Etape 2 : Calculez la similitude de la matrice de notation des éléments utilisateur

R utilisant l'algorithme de similarité utilisateur pour obtenir un utilisateur matrice de similarité.

Etape 3 : triez la matrice de similarité utilisateur de grande à petite et sélectionnez certains utilisateurs comme voisins les plus proches de l'utilisateur cible.

Etape 4 : Utilisez l'algorithme de prédiction de score pour calculer le résultat de l'étape 3, prédire l'évaluation de l'utilisateur cible d'élément inconnu et le recommander. [18]

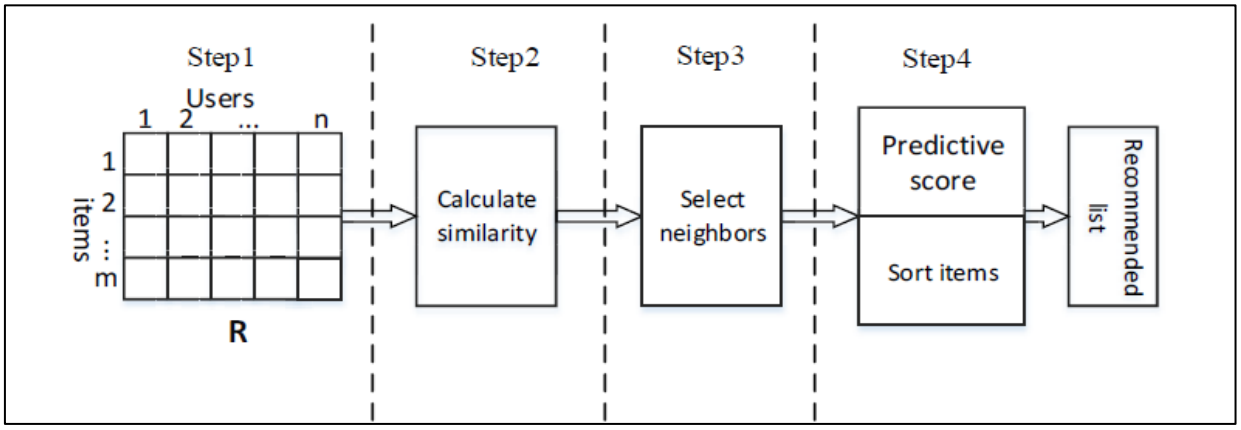


Figure 07 : Etapes de base de l'algorithme de recommandation de filtrage collaboratif.[18]

L'hypothèse principale derrière le filtrage collaboratif est que si un utilisateur apprécie un produit, alors les utilisateurs ayant des goûts similaires à cet utilisateur apprécieront aussi ce produit. [19]

Le filtrage collaboratif utilise pour la recommandation trois (03) techniques :

- ✓ **La première Technique est basée sur la mémoire** (plus proche voisins) qui permet de déterminer quels sont les voisins les plus pertinents à sélectionner et générer des recommandations fiables, pour cela nous utilisons généralement l'algorithme du *k* meilleurs voisins ayant la plus haute valeur de corrélation. Une autre approche (corrélation**)** qui sélectionne seulement les voisins possédant une corrélation plus grande qu'un certain seuil. Nous pouvons distinguer deux méthodes de filtrage collaboratif basé sur la mémoire : la méthode basée sur la mémoire centrée sur l'item et la méthode basée sur la mémoire centrée sur l'utilisateur
- ✓ **La deuxième Technique est basée sur le model,** en général, les méthodes basées sur le modèle utilisent les techniques d'apprentissage automatiques, telles que le clustering, la factorisation matricielle (méthode des facteurs latente), les réseaux bayesiens, les arbres de décision, etc.
- ✓ **La troisième c'est une Technique Hybride** qui combine plusieurs techniques.[16]

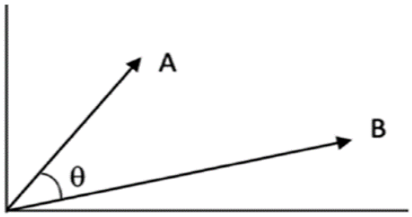
8) Les Métriques de similarité :

a) Cosine similarité

La similarité cosinus mesure la similarité entre deux vecteurs d'un espace produit interne. Il est mesuré par le cosinus de l'angle entre deux vecteurs et détermine si deux vecteurs

pointent à peu près dans la même direction. Il est souvent utilisé pour mesurer la similarité de documents dans l'analyse de texte.

La Cosine similarité est calculer comme suit : [20]



$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

Formule 01: Cosine similarité [01]

b) Corrélation de Pearson

Coefficient de corrélation de Pearson : ce coefficient a été utilisé notamment par les auteurs du système GroupLens, pour calculer la similarité entre deux utilisateurs *u* et *v*. Le coefficient de corrélation de Pearson mesure le rapport entre la covariance et le produit de l'écart-type des notes données par les deux utilisateurs. Il permet ainsi de mesurer la similarité en utilisant les items notés à la fois par *u* et *v*. Plus les deux utilisateurs auront tendance à noter les mêmes items de façon équivalente, plus ils seront similaires. [20]

$$sim(u,v) = Pearson(u,v) = \frac{\sum_{i \in I_{uv}} (r_{u,i} - \bar{r_u}).(r_{v,i} - \bar{r_v})}{\sqrt{\sum_{i \in I_{uv}} (r_{u,i} - \bar{r_u})^2}.\sqrt{\sum_{i \in I_{uv}} (r_{v,i} - \bar{r_v})^2}}$$

Formule 02 : Corrélation de Pearson entre deux utilisateurs [02]

Le coefficient de corrélation de Pearson peut également être utilisé pour mesurer la corrélation entre deux items *i* et *j*.

$$sim(i,j) = Pearson(i,j) = \frac{\sum_{u \in U_{ij}} (r_{u,i} - \bar{r_i}).(r_{u,j} - \bar{r_j})}{\sqrt{\sum_{u \in U_{ij}} (r_{u,i} - \bar{r_i})^2}.\sqrt{\sum_{u \in U_{ij}} (r_{u,j} - \bar{r_j})^2}}$$

Formule 03 : Corrélation de Pearson entre deux items [03]

Le calcul des recommandations dans filtrage collaboratif se base sur diverses manières :

- ✓ En se basant sur le profil des utilisateurs (User-based).

1- User-Based : son principe de fonctionnement est comme suit :

- Utiliser la matrice de notation des éléments par l'utilisateur (user-item rating matrix)
- Calcule la corrélation entre utilisateurs (users)
- Trouver des utilisateurs (users) hautement corrélé
- Recommander des articles préférés par ces utilisateurs

Calcul de similarité entre deux utilisateurs reprenant notre précèdent exemple

user \ item	El-Rissala	La bataille d'Alger	Le Clandestin	Studio Sport	Notre Santé
Mohamed	5	3	?	2	9
Ali	?	1	2	3	6
Sarah	3	5	6	?	7
Linda	5	3	9	4	?

a) Similarité corrélation entre deux individus (x,y)

$$cor(x,y) = \frac{\sum_{i \in I_{xy}} (r_{x,i} - \bar{r}_x)(r_{y,i} - \bar{r}_y)}{\sqrt{\sum_{i \in I_{xy}} (r_{x,i} - \bar{r}_x)^2 \times \sum_{i \in I_{xy}} (r_{y,i} - \bar{r}_y)^2}}$$

I_{xy} représente l'ensemble des items où on dispose des valeurs à la fois pour x et y

Formule 04 :Similarité corrélation [20]

$r_{u,i}$ Note (rating) de l'utilisateur u pour l'item i

$\bar{r}_{u,i}$ Moyenne des notes attribuées par l'utilisateur u (ex. $\bar{r}_{Sarah} = ¼(3+5+6+7) = 5.25$

2- Item-Based

- Utiliser la matrice d'évaluation des éléments par l'utilisateur (user-item rating matrix).
- Créer une corrélation élément à élément.
- Trouvez des éléments hautement corrélés.
- Recommander des éléments avec la corrélation la plus élevée.

Nous allons utiliser une régression simple pour éviter la dimensionnalité et volumétrie Régression multiple qui rendent l'approche impossible. [20]

$Y = x + b$ Un seul paramètre à estimer c'est la constante b

user \ item	El-Rissala	La bataille d'Alger	Le Clandestin	Studio Sport	Notre Santé
Mohamed	5	3	?	2	9
Ali	?	1	2	3	6
Sarah	3	5	6	?	7
Linda	5	3	9	4	?

Prédire item 5 pour Linda

A partir de item1

$$b_1 = \frac{(9-5)+(7-3)}{2} = 4$$

A partir de item2

$$b_2 = \frac{(9-3)+(6-1)+(7-5)}{3} = 4.33$$

A partir de item3

$$b_3 = \frac{(6-2)+(7-6)}{2} = 2.5$$

A partir de item4

$$b_4 = \frac{(9-2)+(6-3)}{2} = 5$$

$\hat{r}^1$ **Linda**= valeur de item1 + b_1

$\hat{r}^1 = 5+4 = 9$

$\hat{r}^2 = 3+4.33 = 7.33$

$\hat{r}^3 = 9+2.5 = 11.5$

$\hat{r}^4 = 4+5 = 9$

$$\hat{r}^1 \text{ Linda,5} = (4 \times 9) + (4.33 \times 7.33) + (2.5 \times 11.5) + (5 \times 9) / (4 + 4.33 + 2.5 + 5) = 8.93$$

- ✓ Ou encore en faisant de la factorisation de matrice ce que nous allons le voir dans le deuxième chapitre [20]

4) Filtrage hybride

Pour remédier aux inconvénients et profiter des avantages de l'approche basé sur le contenu et l'approche collaboratif, l'approche hybride a vu le jour (c'est une combinaison des deux approches).

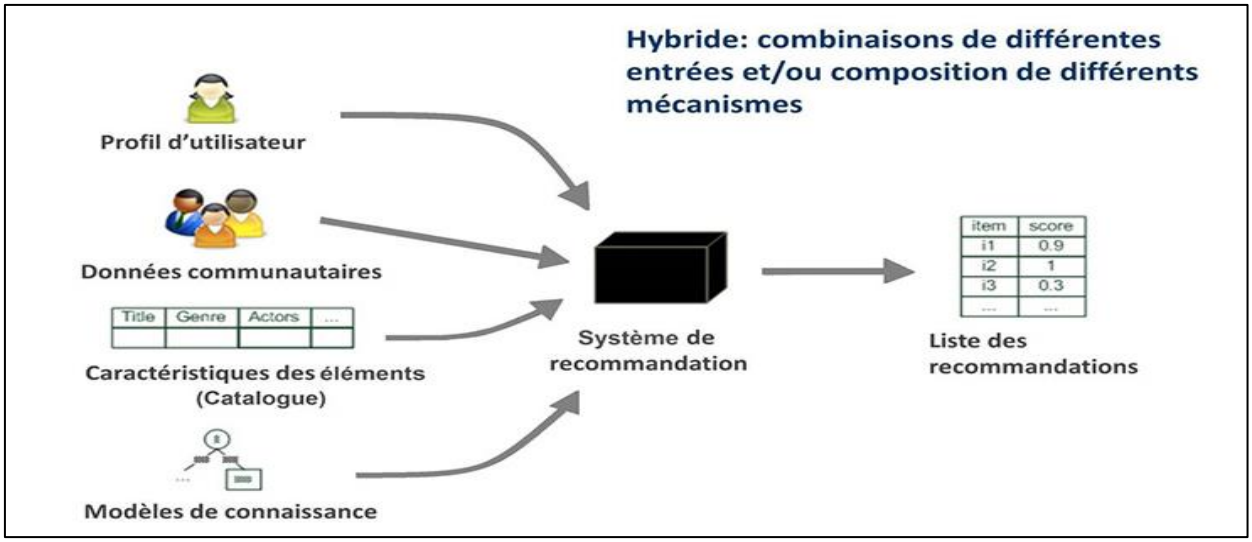


Figure 08 : Le système de recommandation hybride [12].

C'est à noter que le terme « hybride » est un artefact de l'évolution historique des systèmes de recommandation où certaines sources de connaissances ont été exploitées en premier lieu, conduisant à des techniques bien établies qui sont ensuite combinées. L'objectif est alors de s'appuyer sur des sources de connaissances multiples, en choisissant les plus appropriées à une tâche donnée afin de les utiliser le plus efficacement possible.

Il existe trois grandes catégories de combinaisons de systèmes de recommandation pour concevoir un système de recommandation hybride : la combinaison monolithique (monolithic hybridization design), la combinaison parallèle (parallelized hybridization design) et la combinaison tubulaire (pipelined hybridization design).

« Monolithique » décrit une conception d'hybridation qui intègre les aspects de différentes stratégies de recommandation en un seul algorithme. Comme illustré sur la figure 09, différents systèmes de recommandation y contribuent puisque l'approche hybride utilise des données d'entrée additionnelles qui sont spécifiques à un autre algorithme de recommandation, ou bien les données d'entrée sont complétées par une technique et exploitées par une autre. Par exemple, un système de recommandation basé sur le contenu qui exploite également des données communautaires pour déterminer des similarités entre éléments relève de cette catégorie.[12]

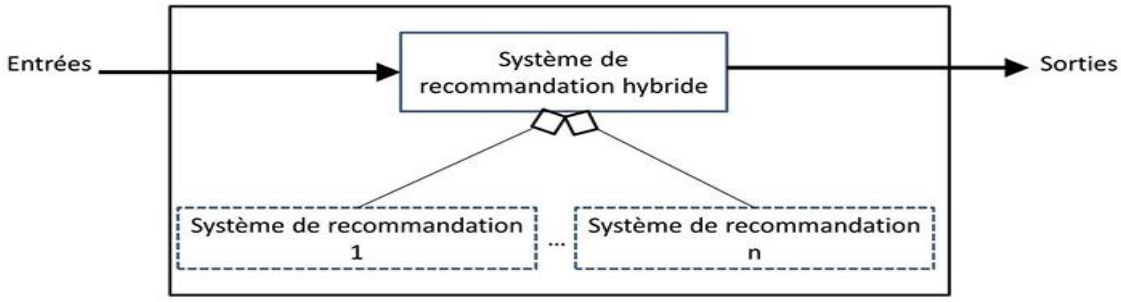


Figure 09 : Conception d'hybridation monolithique [12]

Les deux autres approches hybrides nécessitent au moins deux mises en œuvre de recommandations séparées qui sont combinées en conséquence. Sur la base de leurs données d'entrée, les systèmes hybrides de recommandation parallèles fonctionnent indépendamment l'un de l'autre et produisent des listes de recommandations distinctes, comme illustré sur la figure 10. Dans une étape ultérieure d'hybridation, leurs sorties sont combinées en un ensemble final de recommandations.

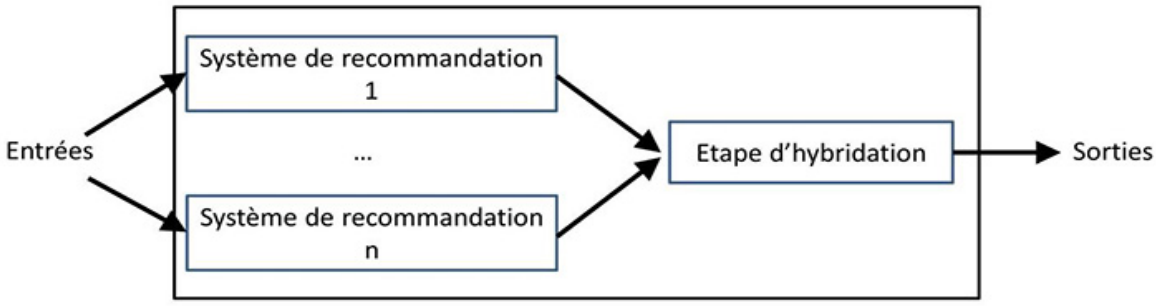


Figure 10 : Conception d'hybridation parallèle [12]

Lorsque plusieurs systèmes de recommandation sont joints dans une architecture tubulaire, comme illustré par la figure 11, la sortie de l'un des systèmes de recommandation devient une partie des données d'entrée du système suivant

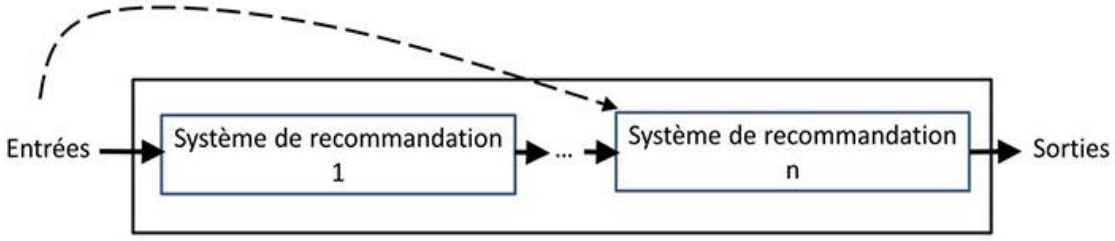


Figure 11 : Conception d'hybridation tubulaire [12]

VII-Conclusion :

Dans ce chapitre nous avons présenté les systèmes de recommandation d'une façon générale commençant par une introduction et une brève historique après nous avons parlé de domaine d'application des systèmes de recommandation, leur fonctionnement et leurs défis notamment le problème de démarrage à froid, qui se pose lors d'un nouvel utilisateur.

Et nous avons terminés par les trois approches qui constituent les systèmes de recommandation (système basé sur le contenu, filtrage collaboratif et filtrage hybride) et les métriques de similarité (la similarité de Cosine et la corrélation de Pearson).

Par conséquent, les systèmes de recommandation peuvent aider les utilisateurs par des bonnes prédictions qui peuvent réduire son espace de recherche.

Chapitre 02 :
La factorisation matricielle dans le filtrage collaboratif

I- Introduction

La Factorisation de matrices dans les systèmes de recommandation est l'une des méthodes utilisées par les services en ligne comme Netflix afin d'accélérer la recherche de recommandations de contenu pour les utilisateurs.

Les approches à base de factorisation matricielle pour le filtrage collaboratif reposent sur la décomposition de matrices comportant des "trous", c'est à dire que les valeurs de certaines entrées sont inconnues. Ces trous correspondent par exemple aux notes manquantes que l'on veut prédire. [21]

La factorisation aide à représenter approximativement la même relation entre la cible et les prédicteurs en utilisant une matrice de dimension inférieure.

L'idée principale de factorisation de matrice pour le système de recommandation est l'existence de caractéristiques latentes représentant la relation entre les éléments et les utilisateurs. [22]

II- Les variables latentes :

1. Définition

Une variable latente est une variable qui n'est pas directement observable, c'est-à-dire une variable cachée, dont les valeurs peuvent être estimées à partir des données observables. A titre d'exemple, on peut citer les facteurs de l'analyse factorielle.

Les variables latentes ou facteurs latents, jouent un rôle de plus en plus important dans la modélisation statistique des problèmes de toute natures. Sur un plan théorique, les variables latentes permettent de prendre en compte plus finement les différentes composantes d'un phénomène complexe sans se soucier nécessairement de leur observabilité du modèle, c'est-à-dire la possibilité de remonter aux paramètres inconnus à partir des seuls éléments observables. [06]

2. L'élaboration d'un modèle latent

L'élaboration d'un modèle latent dans le domaine des systèmes de recommandations emploie souvent la technique de factorisation de matrice. Cette technique permet d'associer les utilisateurs et les items dans un nouvel espace où les liens entre les utilisateurs et les items peuvent être définis par l'utilisation du produit scalaire.

La matrice de l'espace initiale est généralement formée d'utilisateurs pour les lignes et de items pour les colonnes. Ainsi, pour calculer une cote estimée pour un item, il faut prendre le vecteur de l'utilisateur et celui de l'item dans l'espace latent et ensuite, calculer le produit scalaire entre ces deux vecteurs. Toutefois, comme les ensembles de données pour les systèmes de

recommandations sont souvent dispersés, il est important de faire attention pour ne pas ajouter trop de bruit dans le nouvel espace.

Grâce aux préférences déterminées avec le rétrocontrôle, on obtient les vecteurs de caractéristiques des utilisateurs.

Chaque item possède un vecteur de caractéristiques qui lui est propre, plus la correspondance entre les caractéristiques d'un utilisateur avec celles d'un item est grande, plus la recommandation sera considérée comme pertinente.

Parmi les méthodes à facteurs latents, la factorisation de matrices (Matrix Factorisation) est devenue très populaire en filtrage collaboratif à la suite du concours Netflix Prize (le prix de Netflix). [06]

III- La factorisation matricielle

4. Un problème de factorisation matricielle c'est quoi ?

Un problème de factorisation matricielle est un problème d'optimisation où l'on suppose que le modèle *M* est le produit de matrices de facteurs latents, comme par exemple :

$M_{i\,j} = \langle U^*_i , V^*_j \rangle$, où U^*_i est le vecteur (de taille *d*) des variables latentes explicatives de l'utilisateur *i* et V^*_j est le vecteur des variables latentes explicatives de l'article *j* . Généralement, on considère que le nombre de variables latentes *d* est très inférieure à la taille des données.

Le but de la factorisation matricielle est d'estimer ces matrices U* et V* afin d'obtenir une reconstruction qui jouera le rôle de fonction score $\hat{M}i\,j = \langle U_i , V_j \rangle$. Le principal intérêt de la factorisation (particulièrement dans la recommandation) est de fournir une représentation de faible rang d'un modèle de grande dimension. [05]

5. Comment fonctionne la factorisation matricielle ?

La factorisation matricielle fonctionne sur une matrice, où l'axe des *x* et l'axe des *y* représentent des notations. Dans l'ensemble de données que nous allons utiliser, l'axe des *x* représente l'ID du film et l'axe des *y* représente l'ID du client.

Chaque valeur de cellule représente la note d'un client sur un film, de 1 à 5. Il est assez évident qu'il s'agit d'une matrice très clairsemée, comme c'est le cas avec les données du monde réel. Il s'agit d'un système de filtrage collaboratif, c'est-à-dire qu'il ne repose que sur les notes des autres et pas du tout sur les attributs intrinsèques des films.

La factorisation matricielle suit les étapes suivantes :

1) Initialisez deux matrices aléatoires *a* et *b* de dimensions *m* par *j* et *j* par *n* telles que, multipliées, leur dimension corresponde à la matrice d'origine *z* (qui a des dimensions *m* par *n*).
2) Multipliez *a* par *b* pour obtenir une estimation de *z*.
3) Soustrayez *z* de *y* pour les valeurs connues de *z*, ou une autre fonction de perte, pour évaluer à quel point l'estimation est éloignée de la matrice réelle.
4) Utilisez des formules de descente de gradient pour ajuster chacune des valeurs de *a* et *b* dans la bonne direction.
5) Répétez les étapes 2 à 4 à plusieurs reprises jusqu'à ce que l'erreur atteigne une valeur raisonnable.
6) En multipliant *a* par *b*, nous avons maintenant une estimation de *z* qui non seulement correspond étroitement aux valeurs connues de *z*, mais fournit également une estimation des valeurs inconnues. [22]

6. Avantages et inconvénients

Parmi les avantages et les inconvénients de la factorisation matricielle en site les suivants :

1) Le modèle peut aider les utilisateurs à découvrir de nouveaux intérêts.
2) Le système n'a besoin que d'une matrice de rétroaction pour démarrer, donc la collecte des données n'est pas un problème.
3) Comme le montre l'exemple du monde réel, la factorisation matricielle est impossible à grande échelle où le résultat est recherché très rapidement. Cependant, il peut être efficace dans les recommandations où la vitesse n'est pas requise.
4) Le problème du démarrage à froid - lorsque les données sont rares au début (démarrage à froid), la supposition du modèle est à peu près aussi bonne qu'une estimation aléatoire.
5) La factorisation matricielle désigne un modèle à mémoire efficace. [22]

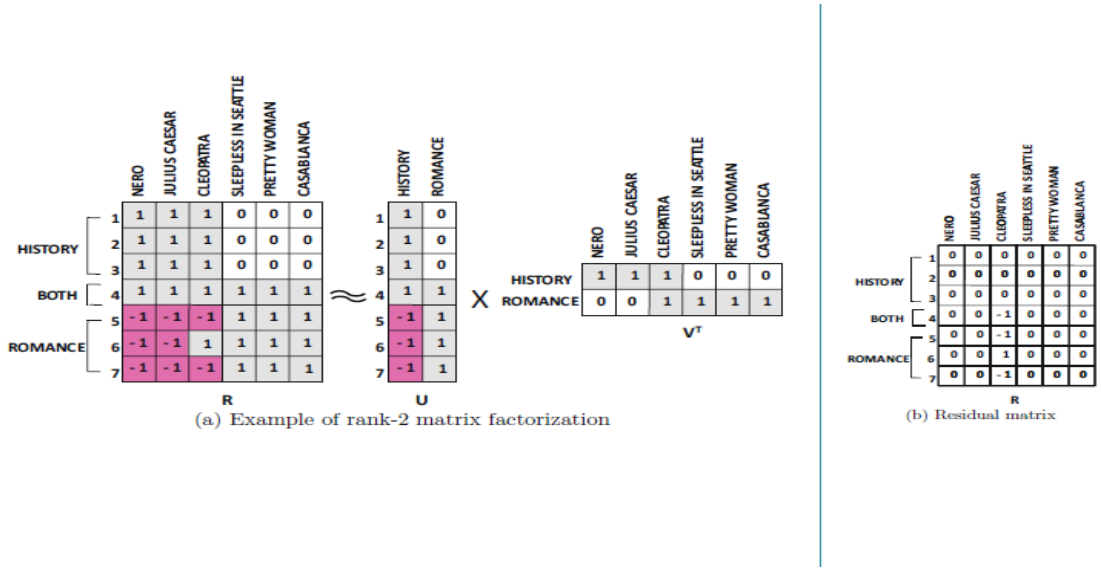


Figure 12 : problème de factorisation matricielle pour le Netflix [29]

IV- Les méthodes de factorisation matricielle

Les algorithmes filtrage collaboratif sont principalement basés sur des modèles d'intégration euclidienne et de factorisation matricielle, Les méthodes de cette dernière sont utilisées pour l'extraction des variables latentes telles que l'Analyse en Composante Principale (ACP), la Décomposition en Valeur Singulière (SVD), La factorisation probabiliste matricielle (PMF), La factorisation matricielle non négative (NMF). Elles ont été essentiellement utilisées, soit pour réduire la dimension de la matrice des notes, soit pour réduire la dimension de la matrice de similarité. [24]

5. L'analyse en composantes principales ACP

L'analyse en composantes principale est un outil extrêmement puissant de compression et de synthèse de l'information, très utile lorsque l'on est en présence d'une somme importante de données quantitatives à traiter et interpréter. L'apparition au cours des dernières années des logiciels chaque fois plus performants et faciles à utilisées.

L'ACP est une analyse factorielle, en ce sens qu'elle produit des facteurs (ou axes principaux) qui sont des *combinaisons linéaires* des *variables* initiales, hiérarchisées et indépendantes les unes des autres. On appelle parfois ces facteurs des « dimensions latentes », du fait qu'ils sont l'expression de processus généraux dirigeant la répartition de plusieurs phénomènes qui se retrouvent ainsi corrélés entre eux.

L'ACP se réalise sur une *population* donnée. Dans l'exemple choisi ici, les *individus statistiques* constituant cette *population.* [25]

1.2 Quelles sont les principales étapes de l'analyse en composantes principales ?

Il existe 4 principales étapes lors d'une analyse en composantes principales :

1. Définir les objectifs de l'analyse et l'approche (exploratoire ou confirmatoire) adaptée au type de problème, selon l'existence ou non d'a priori théoriques.
2. Préparer l'analyse en déterminant le nombre de variables conservées, le type de variables (continues ou dichotomiques), et la taille de l'échantillon.
3. S'assurer de l'existence de corrélations minimales entre les variables analysées, en recourant à une matrice de corrélation, puis mesurer l'adéquation de l'échantillonnage et réaliser un test de sphéricité dit de Bartlett.
4. Choisir le nombre de facteurs à extraire grâce à l'ACP en se fiant à deux critères distincts. [25]

6. La méthode SVD (Décomposition des valeurs singulières)

La décomposition des valeurs singulières, est basée sur un critère de moindres carrés, elle factorise la matrice de notation $M_{m\times n}$ avec un rang de k, à trois (03) matrices :

$$U_{m\times k},\ \Sigma_{k\times k} \text{ and } I^T_{n\times k}:$$

$$M = U\Sigma_k I^T$$

$$M \approx U\Sigma_{k'} I^T$$

U c'est la matrice des préférences de l'utilisateur, I représente les préférences des items et Σ c'est la matrice des valeurs singulières. La beauté de La décomposition des valeurs singulières réside dans cette notion simple qu'au lieu d'un espace vectoriel complet k, nous pouvons approximer M sur un espace latent k′ beaucoup plus petit. Cette approximation réduira non seulement les dimensions de la matrice d'évaluation, mais elle ne prend également en compte que les valeurs singulières les plus importantes et laisse de côté les valeurs singulières plus petites qui pourraient autrement entraîner du bruit.

Pour approximer M, nous aimerions trouver les matrices U et I dans l'espace k′ en utilisant tous les taux connus, ce qui signifierait que nous résoudrons un problème d'optimisation, chaque entrée de

$$r_{ui} = p_u \cdot q_i$$

taux dans M, r_{ui} peut être écrite comme un produit scalaire de p_u et q_i :

Où p_u constitue les lignes de U et q_i les colonnes de I^T [19]

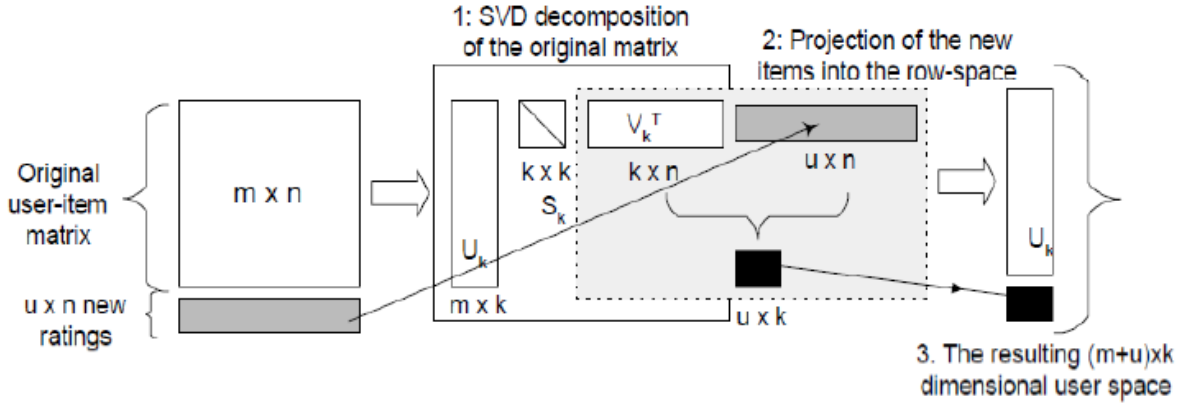


Figure13 : Décomposition de la valeur singulière SVD. [19]

7. La factorisation probabiliste matricielle (PMF) :

De nombreux modèles probabilistes ont été proposés pour le filtrage collaboratif. Les plus aboutis modélisent les comportements des utilisateurs à l'aide d'une variable latente pouvant prendre K valeurs, chaque valeur correspondant à un comportement type. Dans la suite, Z est une variable latente dans {1, ..., K}, U et u sont une variable et un indice d'utilisateurs, Y et y sont une variable et un indice d'articles, R et r sont une variable note et une note dans {1, ..., V }.

Avec ces notations, chaque utilisateur u est représenté par une distribution p (Z|U = u), modélisant à quel point chaque comportement type intervient dans le comportement de u. A chaque comportement type z et à chaque item y est associée une distribution multinomiale p (R|Z = z, Y = y) qui modélise les goûts du comportement type z pour l'article y. D'un point de vue génératif, la note attribuée par l'utilisateur u à l'article y est générée de la façon suivante : un comportement type z est choisi suivant la distribution p (Z|U = u), puis la note r est choisie suivant la distribution p (R|Z = z, Y = y). Lorsque les distributions sont connues, la prédiction de notes pour l'utilisateur u et l'article y consiste à calculer les p (R = r|U = u, Y = y) pour toutes les notes r, et à prédire la note médiane rˆ telle que rˆ = {r|p(R = r|U = u, Y = y) ≤ 1/2, p(R = r|U = u, Y = y) ≥ 1/2}. [26]

8. La factorisation matricielle non négative (NMF) :

La factorisation en matrices non négatives (FMN) est une méthode de décomposition matricielle, introduite par. Elle permet d'approximer toute matrice X de taille ($n \times p$) et dont les éléments sont tous positifs, grâce à une décomposition de la forme X ≈ WH, où W et H sont des matrices (n×k) et (k ×m).

La factorisation non-négative de la matrice X est la recherche de deux matrices $W_{n\times r}$ et $H_{r\times p}$ ne contenant que des valeurs positives ou nulles et dont le produit approche X.

Le choix du rang de factorisation **r << *min(n, p)*** assure une réduction drastique de dimension et donc des représentations parcimonieuses. Évidemment, la qualité d'approximation dépend de la parcimonie de la matrice initiale.

L'originalité de la factorisation en matrices non négatives réside dans les contraintes de non-négativité qu'elle impose à W et H ; c'est à dire que leurs éléments doivent être tous positifs. Ces contraintes font que les vecteurs de base comportent beaucoup de 0 et que leurs parties non nulles se chevauchent rarement. La représentation d'un objet (décrit par un vecteur de réels positifs) comme une somme de ces vecteurs de base, correspond alors à l'intuition d'une décomposition par parties. [27]

La factorisation est résolue par la recherche d'un optimum local du problème d'optimisation : $\min_{\mathbf{W},\mathbf{H}\geq 0} [L(\mathbf{X}, \mathbf{WH}) + P(\mathbf{W}, \mathbf{H})]$. [27]

V- Complétion de matrice

Les premiers travaux de filtrage collaboratif se sont focalisés sur le traitement de données explicites, lorsque, les données sont des notes d'appréciation, la valeur "0" signifie en principe une valeur manquante

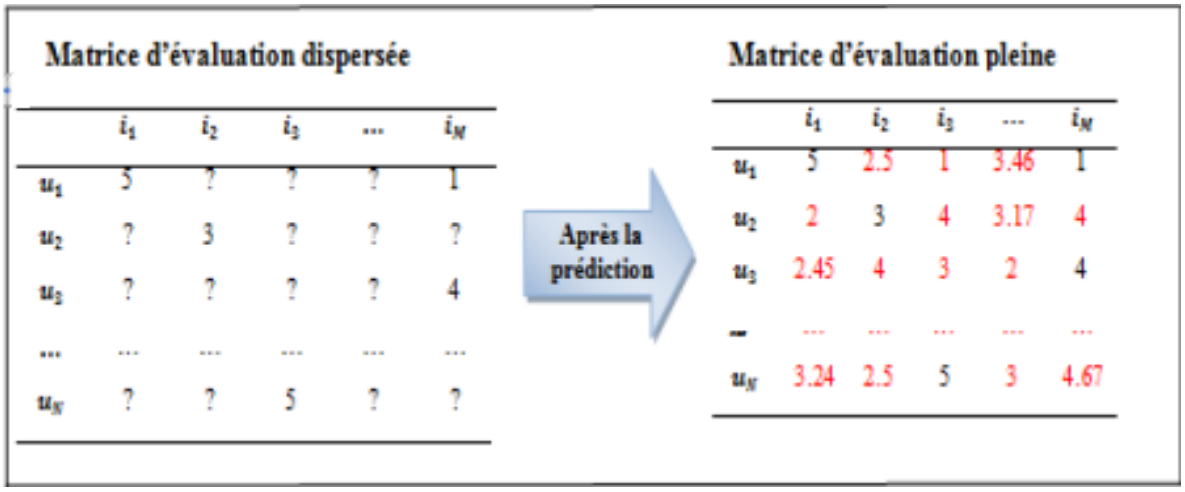
Dans ce cas, le but du système de recommandation est de compléter les valeurs manquantes de la matrice **Y.** Les prédictions de font alors à l'aide de l'approximation de rang faible $\hat{\mathbf{Y}}$. Lorsque **Y** est une matrice pleine (dont tous les coefficients sont observés), Dans ce manuscrite nous allons utiliser la décomposition en valeurs singulières (SVD) car permet d'obtenir la meilleure approximation de rang faible de **Y.** Pour cela, il suffit de sélectionner les ***K*** plus grandes valeurs singulières de la décomposition. Cependant, dans le cadre du filtrage collaboratif appliqué à des données explicites, la matrice **Y** n'est que partiellement observée. On ne peut donc plus appliquer cette méthode pour obtenir l'approximation.[27]

VI- La prédiction des évaluations et métriques d'exactitude :

La prédiction des évaluations est un problème qui a beaucoup été abordé dans les systèmes de recommandation Ce problème peut être posé de la manière suivante : les interactions entre les utilisateurs et les articles sont stockées dans une matrice R d'évaluation et sont souvent encodées sous la forme d'entiers allant de 1 à 5 étoiles avec 1 la valeur minimale signifiant que l'utilisateur n'a pas apprécié l'article et 5 signifiants que l'utilisateur a particulièrement apprécié l'article. Le but du système de recommandation S est de prédire les futures évaluations des utilisateurs. Le processus est le suivant : le jeu de données est divisé en deux, un ensemble d'entrainement (Train)

sur lequel l'algorithme de recommandation apprend à prédire la note, et un second ensemble (Test) sur lequel l'algorithme teste ces prédictions.

Pour le calcul de la prédiction, la méthode la plus simple est de calculer la moyenne des notes de tous les voisins de l'utilisateur courant u comme l'illustre l'équation.



	i_1	i_2	i_3	...	i_N
u_1	5	?	?	?	1
u_2	?	3	?	?	?
u_3	?	?	?	?	4
...	...	...	...	...	...
u_N	?	?	5	?	?

	i_1	i_2	i_3	...	i_N
u_1	5	2.5	1	3.46	1
u_2	2	3	4	3.17	4
u_3	2.45	4	3	2	4
...	...	...	...	...	...
u_N	3.24	2.5	5	3	4.67

Figure 14 : Un exemple illustratif de la prédiction des évaluations manquantes [07]

Les métriques utilisées pour valider la précision de la prédiction afin d'évaluer la performance de systèmes sont les suivantes :

L'erreur moyenne absolue (MAE) et l'erreur quadratique moyenne (RMSE).

$$\mathrm{MAE} = \frac{1}{N}\sum_{u,i}\left|p_{u,i} - r_{u,i}\right| \quad \mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{u,i}\left(p_{u,i} - r_{u,i}\right)^2}$$

[28]

Où

$P_{u,i}$ est la note prédite pour l'utilisateur *u* sur l'élément *i*.

$r_{u,i}$ est la note réelle et N est le nombre total de notes sur l'ensemble d'articles.

L'optimisation de ce système est de minimiser l'erreur de prédiction, c'est-à-dire arriver à des valeurs de MAE et RMSE minimales. Les recherches sur les prédictions sont corrélées à la nature des jeux de données disponibles.

Les jeux de MovieLens sont composés des appréciations explicites que les utilisateurs ont données aux films. Initialement, la majorité des systèmes de recommandation sont des systèmes prédictifs, c'est-à-dire qu'ils prédisent une opinion ou la probabilité d'un achat. L'hypothèse est que les recommandations produites par un système ayant ces valeurs MAE ou RMSE minimales, sera le système qui apportera les meilleures recommandations aux utilisateurs. [28]

VII- Conclusion :

Dans ce chapitre nous avons présenté des méthodes qui se reposant sur des problèmes de factorisation matricielle notamment les modèles à facteurs latin nous avons définis d'abord la valeur latente et l'élaboration d'un model latent.

Ensuite nous avons déterminé que signifie un problème de factorisation matricielle et comment il fonctionne en citant les avantages et les inconvénients de ce problème.

Après nous avons parlé des méthodes de factorisation d'une façon générale notamment dis l'analyse des composantes principale ACP, la décomposition en valeurs singulière SVD, la factorisation probabiliste matricielle et la non négative matrice.

La complétion de matrice et nous avons terminé par la prédiction et évaluation de métrique tel que (RMSE) Root Mean Square Error et (MAE) l'Erreur Moyenne Absolue.

Chapitre 03 :

Implémentation

I- Introduction

En informatique l'implémentation désigne mettre en œuvre une analyse informatique (programmer) ou un programme informatique (paramétrer). Effectuer l'ensemble des opérations qui permettent de définir un projet et de le réaliser, de l'analyse du besoin à l'installation et la mise en service du système ou du produit.

Dans ce chapitre du mémoire, nous mettons en avant notre implémentation et interprétation, de l'importation des bibliothèques nécessaires et importante, jusqu'à l'obtention des résultats.
Avant ça, nous allons définir brièvement l'environnement de développement et les outils pour réaliser ce travail.
Nous avons implémenté notre projet sur un ordinateur portable HP avec un processeur Intel® Celeron® CPU N3060 @ 1.60GHz 1.60 GHz, Mémoire RAM installé 4 Go, le système d'exploitation est Windows 10 sur 64 bits.

II- Environnement de développement

On a utilisé plusieurs outils et environnement pour atteindre notre objectif, nous présenterons dans les paragraphes qui se suivent.

4. **Anaconda :**

   ✓ **Anaconda** est une distribution de gestion de paquets en Open source des langages de programmation Python et R appliqué au développement d'applications dédiées à la science des données et à l'apprentissage automatique (traitement de données à grande échelle, analyse prédictive, calcul scientifique), qui vise à simplifier la gestion des paquets et de déploiement .

   ✓ **Navigateur Anaconda** est une interface graphique (GUI) incluse dans la distribution Anaconda, et qui permet aux utilisateurs de lancer des applications, mais aussi de gérer les librairies conda, les environnements et les canaux sans utiliser la moindre ligne de commande. Le Navigateur peut également accéder à des librairies présentes sur le Cloud Anaconda ou dans un Repository Anaconda local, afin de les installer dans un environnement, les exécuter et les mettre à jour. Il est disponible pour Windows, MacOs et Linux.[29]

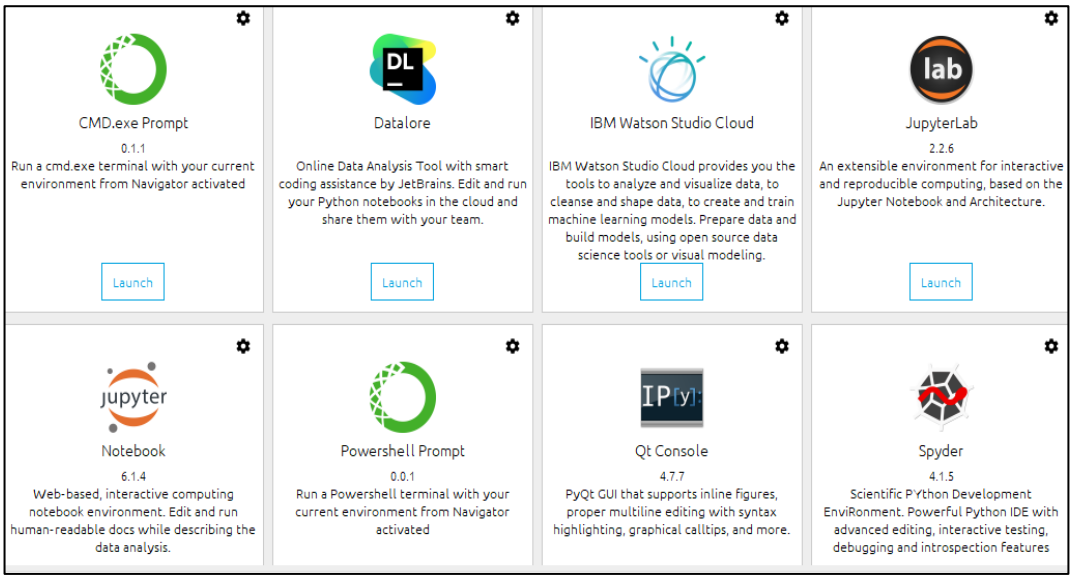


Figure15 : Interface anaconda [29]

5. **Jupyter Notebook :** est une application Web open source qui permet de créer et de partager des documents contenant du code en direct, des équations, des visualisations et du texte explicatif. Les utilisations incluent : le nettoyage et la transformation des données, la simulation numérique, la modélisation statistique, l'apprentissage automatique et bien plus encore.[30]

6. **Python** : est un langage de programmation open source et interprété, il est conçu pour optimiser la productivité des programmeurs en offrant des outils de haut niveau et une syntaxe simple à utiliser.

   Python est le langage de programmation le plus employé par les informaticiens. Ce langage s'est propulsé en tête de la gestion d'infrastructure, d'analyse de données ou dans le domaine du développement de logiciels.

   C'est le langage le plus utilise hors web au monde il fonctionne sur la plupart des plates-formes informatiques, des smartphones et peut aussi être traduit en Java ou .NET. [31]

Figure 16 : logo Python [31]

III- Etapes de l'implémentation :

- Importation des bibliothèques
- Importation du dataset.
- Prétraitement des données.
- Phase d'apprentissage.
- Phase de test
- Evaluation et interprétation.

2. Importation des bibliothèques nécessaires :

1.1. Pandas :

Une des bibliothèques Python Open source, les plus utilisées pour la Data Science, Développé en 2008 par Wes McKinney, implémenté à partir de C – d'où sa rapidité qui sert a introduit les objets Data Frame et Séries.[32]

1.2. NumPy :

est une bibliothèque Python utilisée pour travailler avec des tableaux, elle permet d'effectuer des calculs numériques avec Python. Elle introduit une gestion facilitée des tableaux de nombres. [32]

1.3. Scikit-learn :

est la principale bibliothèque d'outils dédiés au machine Learning et à la data-science dans l'univers Python. [32]

1.4. SciPy :

Le module SciPy contient de nombreux algorithmes d'algèbre linéaire, intégration numérique. On peut voir ce module comme une extension de Numpy car il contient toutes ses fonctions.[32]

```
import pandas as pd
import numpy as np
from scipy.sparse import csr_matrix
from sklearn.decomposition import TruncatedSVD
```

Figure 17 : Importation des bibliothèques

2. Importation des données et lecture de data set

La factorisation matricielle s'applique à plusieurs types de données de différents domaines pour les services en ligne (Netflix, Amazon, PANDORA, réseaux sociaux.)

Dans ce mémoire nous allons utiliser la base de données **MovieLens 1M** il s'agit d'un ensemble de données pour les notes des films donner par les utilisateurs (1 million d'avis de 6000 utilisateurs sur 4000 films).

Ces fichiers contiennent 1 000 209 évaluations anonymes d'environ 3 900 films réalisé par 6 040 utilisateurs de MovieLens.

```
movies_df = pd.read_csv('D:/ml-1m/ml-1m/movies.dat', sep="::", header=None)
movies_df.columns = ['MovieID', 'Title', 'Genres']

users_df = pd.read_csv('D:/ml-1m/ml-1m/users.dat', sep="::", header=None)
users_df.columns = ['UserID', 'Sexe', 'Âge', 'Occupation','MovieID']

ratings_df = pd.read_csv('D:/ml-1m/ml-1m/ratings.dat', sep="::", header=None)
ratings_df.columns = ['UserID', 'MovieID', 'Rating', 'Timestamp']
```

Figure 18 : Lecture des fichiers

- **Description du fichier de notation**

Toutes les notes sont contenues dans le fichier "ratings.dat" et sont dans le format suivant:

UserID -MovieID -Rating - Timestamp

Les ID utilisateur varient entre 1 et 6040

Les MovieIDs sont compris entre 1 et 3952

Nous avons affiché des cinq premières lignes de la matrice note ' ratings' par défaut.

```
: ratings_df.head()
```

	UserID	MovieID	Rating	Timestamp
0	1	1193	5	978300760
1	1	661	3	978302109
2	1	914	3	978301968
3	1	3408	4	978300275
4	1	2355	5	978824291

Figure 19 : Affichage de la matrice de note (cinq premières lignes par défaut)

- **Description du fichier Film**

Les informations sur le film se trouvent dans le fichier "movies.dat" et se trouvent dans le suivant format :

ID du film - Titre – Genres.

Nous avons affiché des cinq premières lignes de la matrice films par défaut.

```
]: movies_df.head()
```

	MovieID	Title	Genres
0	1	Toy Story (1995)	Animation|Children's|Comedy
1	2	Jumanji (1995)	Adventure|Children's|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama
4	5	Father of the Bride Part II (1995)	Comedy

Figure 20 : Affichage de la matrice film (cinq premières lignes par défaut)

- **Description du fichier utilisateur**

Les informations sur l'utilisateur se trouvent dans le fichier "users.dat" et se trouvent dans le suivant format :

ID utilisateur – Sexe – Âge -Occupation

Nous avons affiché des cinq premières lignes de la matrice utilisateurs par défaut.

```
5]: users_df.head()
```

	UserID	Sexe	Âge	Occupation	MovieID
0	1	F	1	10	48067
1	2	M	56	16	70072
2	3	M	25	15	55117
3	4	M	45	7	02460
4	5	M	25	20	55455

Figure 21 : Affichage de la matrice utilisateur (cinq premières lignes par défaut)

3. Prétraitement des données

Le prétraitement des données est une étape très importante qui consiste à nettoyer, éliminer, identifier et corriger les informations inexactes.

Nous avons transformé le format de la matrice d'évaluation R (utilisateur par ligne et film par colonne). Et nous avons pivoter la matrice des notes pour obtenir la nouvelle variable `R`.

```
]: R_df = ratings_df.pivot(index = 'UserID', columns ='MovieID', values = 'Rating').fillna(0)
   R_df.head()
```

MovieID	1	2	3	4	5	6	7	8	9	10	...	3943	3944	3945	3946	3947	3948	3949	3950	3951	3952
UserID																					
1	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
5	0.0	0.0	0.0	0.0	0.0	2.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

5 rows × 3706 columns

Figure 22 : La nouvelle matrice R

Après nous avons calculer la moyenne de notation 'ratings' par un utilisateur et en à afficher la dimension de la nouvelle matrice R.

```
: R = R_df.to_numpy()
  user_ratings_mean = np.mean(R, axis = 1)
  R_demeaned = R - user_ratings_mean.reshape(-1, 1)

: R.shape

(6040, 3706)
```

Figure 23 : la dimension de la matrice R

Nous avons utilisé la fonction Scipy svds car elle me permet de choisir le nombre de facteurs latents que je souhaite utiliser pour approximer la matrice d'évaluation d'origine (U, sigma et Vt) avec un rang K=50 et par la suite nous avons affiché la matrice Vt.

```
: from scipy.sparse.linalg import svds
  U, sigma, Vt = svds(R_demeaned, k = 50)

: Vt.shape

(50, 3706)
```

Figure 24 : la dimension de la matrice vt

4. La recommandation

Nous avons passé par trois étapes :

- La première est Obtenir et trier les prédictions de l'utilisateur.

```
def recommend_movies(predictions_df, userID, movies_df, original_ratings_df, num_recommendations=5):

    # Obtenir et trier les prédictions de l'utilisateur
    user_row_number = userID - 1 # UserID starts at 1, not 0
    sorted_user_predictions = predictions_df.iloc[user_row_number].sort_values(ascending=False)
```

Figure 25 : trier les prédictions utilisateur

- La deuxième est d'obtenir les données de l'utilisateur et fusionnez dans les informations du film et afficher

```
    # Obtenez les données de l'utilisateur et fusionnez dans les informations du film
    user_data = original_ratings_df[original_ratings_df.UserID == (userID)]
    user_full = (user_data.merge(movies_df, how = 'left', left_on = 'MovieID', right_on = 'MovieID').
                 sort_values(['Rating'], ascending=False)
                )

    print ('User {0} has already rated {1} movies.'.format(userID, user_full.shape[0]))
    print ('Recommending the highest {0} predicted ratings movies not already rated.'.format(num_recommendations))
```

Figure 26: Fusionnement des données utilisateur et données film

- La troisième est de recommander les films les mieux notés que l'utilisateur n'a pas encore vus.

```
# Recommandez les films les mieux notés que l'utilisateur n'a pas encore vus.
recommendations = (movies_df[~movies_df['MovieID'].isin(user_full['MovieID'])].
     merge(pd.DataFrame(sorted_user_predictions).reset_index(), how = 'left',
           left_on = 'MovieID',
           right_on = 'MovieID').
     rename(columns = {user_row_number: 'Predictions'}).
     sort_values('Predictions', ascending = False).
                  iloc[:num_recommendations, :-1]
                 )

return user_full, recommendations
```

Figure 27 : Recommandation des 10 films les mieux notés

Et nous avons afficher la matrice suivante

```
]: already_rated.head(10)
```

	UserID	MovieID	Rating	Timestamp	Title	Genres
36	837	858	5	975360036	Godfather, The (1972)	Action|Crime|Drama
35	837	1387	5	975360036	Jaws (1975)	Action|Horror
65	837	2028	5	975360089	Saving Private Ryan (1998)	Action|Drama|War
63	837	1221	5	975360036	Godfather: Part II, The (1974)	Action|Crime|Drama
11	837	913	5	975359921	Maltese Falcon, The (1941)	Film-Noir|Mystery
20	837	3417	5	975360893	Crimson Pirate, The (1952)	Adventure|Comedy|Sci-Fi
34	837	2186	4	975359955	Strangers on a Train (1951)	Film-Noir|Thriller
55	837	2791	4	975360893	Airplane! (1980)	Comedy
31	837	1188	4	975360920	Strictly Ballroom (1992)	Comedy|Romance
28	837	1304	4	975360058	Butch Cassidy and the Sundance Kid (1969)	Action|Comedy|Western

Figure 28: Matrice des 10 films les mieux notés.

5. La prédiction

Nous avons défini une fonction pour la prédiction qui regroupe les étapes suivantes :

1. La première partie concerne model.fit

```
def prediction(ratings_df):

    R_df = ratings_df.pivot(index = 'UserID', columns ='MovieID', values = 'Rating').fillna(0)
    R = R_df.to_numpy()
    user_ratings_mean = np.mean(R, axis = 1)
    R_demeaned = R - user_ratings_mean.reshape(-1, 1)
    U, sigma, Vt = svds(R_demeaned, k = 50)
    sigma = np.diag(sigma)
    all_user_predicted_ratings = np.dot(np.dot(U, sigma), Vt) + user_ratings_mean.reshape(-1, 1)
    return pd.DataFrame(all_user_predicted_ratings, columns = R_df.columns)

def recommend_movies(predictions_df, userID, movies_df, original_ratings_df, num_recommendations=5):
```

2. Avoir un tri de prédiction des utilisateurs

```
# Get and sort the user's predictions
user_row_number = userID - 1 # UserID starts at 1, not 0
sorted_user_predictions = predictions_df.iloc[user_row_number].sort_values(ascending=False)
```

3. Fusionner la matrice utilisateur avec la matrice film

```
# Get the user's data and merge in the movie information.
user_data = original_ratings_df[original_ratings_df.UserID == (userID)]
user_full = (user_data.merge(movies_df, how = 'left', left_on = 'MovieID', right_on = 'MovieID').
             sort_values(['Rating'], ascending=False)
            )

print ('User {0} has already rated {1} movies.'.format(userID, user_full.shape[0]))
print ('Recommending the highest {0} predicted ratings movies not already rated.'.format(num_recommer
```

4. Recommander les notations les plus élever des films que l'utilisateur n'a pas encore vu.

```
# Recommend the highest predicted rating movies that the user hasn't seen yet.
recommendations = (movies_df[~movies_df['MovieID'].isin(user_full['MovieID'])].
     merge(pd.DataFrame(sorted_user_predictions).reset_index(), how = 'left',
           left_on = 'MovieID',
           right_on = 'MovieID').
     rename(columns = {user_row_number: 'Predictions'}).
     sort_values('Predictions', ascending = False).
                 iloc[:num_recommendations, :-1]
                )

return user_full, recommendations
```

5. Et la fin nous avons obtenu la matrice des prédictions suivante :

```
]: preds_df = prediction(ratings_df)
   already_rated, predictions = recommend_movies(preds_df, 837, movies_df, ratings_df, 10)
   already_rated.head(10)
```

```
User 837 has already rated 69 movies.
Recommending the highest 10 predicted ratings movies not already rated.
```

	UserID	MovieID	Rating	Timestamp	Title	Genres
36	837	858	5	975360036	Godfather, The (1972)	Action|Crime|Drama
35	837	1387	5	975360036	Jaws (1975)	Action|Horror
65	837	2028	5	975360089	Saving Private Ryan (1998)	Action|Drama|War
63	837	1221	5	975360036	Godfather: Part II, The (1974)	Action|Crime|Drama
11	837	913	5	975359921	Maltese Falcon, The (1941)	Film-Noir|Mystery
20	837	3417	5	975360893	Crimson Pirate, The (1952)	Adventure|Comedy|Sci-Fi
34	837	2186	4	975359955	Strangers on a Train (1951)	Film-Noir|Thriller
55	837	2791	4	975360893	Airplane! (1980)	Comedy
31	837	1188	4	975360920	Strictly Ballroom (1992)	Comedy|Romance
28	837	1304	4	975360058	Butch Cassidy and the Sundance Kid (1969)	Action|Comedy|Western

IV-Conclusion :

Dans ce chapitre du mémoire nous avons présenté les étapes de notre implémentation, les outils utilisés et les différentes parties de notre phase d'implémentation. Ainsi nous avons présenté quelques bibliothèques qui ont été indispensables, enfin nous avons interprété les différents résultats obtenus.

Conclusion
et
Perspectives

Conclusion et Perspectives

Les systèmes de recommandation sont des plateformes pour une interaction sociale, ils proposent aux utilisateurs des produits qui sont susceptibles de l'intéresser, les systèmes de recommandation aident aux utilisateurs d'estimer leurs besoins à partir des certaines données.

Les démarches que nous avons eu à mener à savoir de la définition des systèmes de recommandation en générale et le filtrage collaboratif notamment les méthodes de la factorisation matricielle où les matrices d'interaction utilisateur-élément répertorient généralement les utilisateurs et les éléments dans des lignes et des colonnes, respectivement. Ensuite, les enregistrements d'interaction entre eux sont représentés comme des éléments correspondants dans la matrice.

L'objectif principal de ce mémoire est de construire un système de recommandation basé sur le filtrage collaboratif capable de prédire des items (films) pour l'utilisateur à partir des préférences d'autres utilisateurs similaires à ses préférences.

L'implémentation a été faite avec le langage de programmation python et on a utilisé des bibliothèques pour faciliter la tâche de création de notre modèle et pour l'accélération du training.

Notre travail n`est que dans sa version initiale, nous pouvons dire que ce travail reste ouvert pour des travaux de comparaison avec d'autres méthodes de factorisation matricielle.

Comme perspectives de recherches futures, nous envisageons de :

1. D'appliquer d'autre méthodes de factorisation sur notre data set tel que factorisation matricielle non négative.

2. De faire une comparaison entre les résultats de la prédiction par deux méthodes de factorisation par exemple (SVD et ACP) .

Bibliographie

1. **Haydar, Charif Alchiekh.** *Les systèmes de recommandation à base de confiance.* France : Université de Lorraine, 2018.

2. **Prem Melville, Raymond J. Mooney , Ramadass Nagarajan.** *Content-Boosted Collaborative Filtering for Improved Recommendations.* Canada : University of Texas,Department of Computer Sciences, 2002.

3. **Picot-Clémente, Romain.** *Une architecture générique de Systèmes de recommandation de combinaison d'items : application au domaine du tourisme.* France : 'Université de Bourgogne, 2011.

4. **Bechet, Nicolas.** *Etat de l'art sur les Systemes de Recommandation.* 2012.

5. **Delporte, Julien.** *Factorisation matricielle, application à la recommandation personnalisée de préférences.* s.l. : LITIS - Laboratoire d'Informatique, 2014.

6. **Benouaret, Idir.** *Un système de recommandation contextuel et composite pour la visite personnalisée de sites culturels.* France : Université de technologie de Compiègne, 2020.

7. **Maatallah-Majda.** *Une Technique Hybride pour les Systèmes de Recommandation- .* Annaba : Université Badji Mokhtar,Département d'informatique, 2016.

8. Factorisation de matrices pour les systmes de recommandation. *Wikipédia.* [En ligne]

9. **Jérémy, Bastin.** *Etude des systèmes de recommandations et mise en pratique des algorithmes.* Belgique : Université de Liege ,Ecole de Gestion, 2020.

10. **dgdevlab_admin.** How Recommendation Systems Work On Amazon & Netflix (Simplilearn Webinar). *DINUS GAME DEV LAB.* [En ligne] 04 septembre 2019. [Citation : 22 Mai 2021.] http://dgdevlab.intsys-research.org/2019/09/04/how-recommendation-systems-work-on-amazon-netflix-simplilearn-webinar/.

11. **Vancompernolle Vromman, Flore.** *Les systèmes de recommandation enferment-ils les utilisateurs, au cours du temps, dans leurs préférences dominantes ?* Belgique : Faculté Louvain School of Management, 2020.

12. **Gupta, Koyel Datta.** *A survey on recommender system.* India : Maharaja Surajmal Institute of Technology, 2019.

13. **Negre, Elsa.** Les systèmes de recommandation : une catégorisation. *Interstices.info.* [En ligne] 20 Septembre 2018. [Citation : 29 Mail 2021.] https://interstices.info/les-systemes-de-recommandation-categorisation/.

14. **Renaud-Deputter, Simon.** *Système de recommandations utilisant une combinaison de filtrage collaboratif et de segmentation pour des données implicites.* Canada : FACULTÉ DES SCIENCES, Département d'informatique , 2013.

15. **Rohit Sharma, avineet Kumar,.** *Improving Movie Recommendation System .* s.l. : Université of south carolina, 2018.

16. **GOUVERT, M. OLIVIER.** *Factorisation bayésienne de matrices pour le filtrage collaboratif.* Toulouse : Institut de Recherche en Informatique, 2019.

17. **Amine, Naak.** *Papyres : Un système de gestion et de recommandation.* Montréal : Faculté des arts et des sciences Département d'informatique, 2009.

18. **Bahmed, Hassane Boumriga.** *La protection de la vie privée dans un système de gestion d'identité.* Telemcen : Université Abou Bakr Belkaid- Département d'informatique, 2012.

19. Collaborative filtering algorithm based on rating difference and user interest. *ieeexplore.* [En ligne] 20 Avril 2018. [Citation : 30 Mai 2021.] https://ieeexplore.ieee.org/abstract/document/8386462.

20. **John S, Breese David , Heckerman Carl Kadie.** *Empirical Analysis of Predictive Algorithms for Collaborative Filtering.* 2013.

21. **Jean-François Pessiot, Vinh Truong, Nicolas Usunier,Massih-Reza Amini, Patrick Gallinari.** *Filtrage Collaboratif avec un Algorithme.* Paris : Laboratoire d'Informatique , 2007.

22. **Delporte, Julien.** *Factorisation matricielle, application à la recommandation personnalisée de préférences.* 2014.

23. **Gormley, Matt.** *Introduction to Machine Learning- Matrix Factorization and Collaborative Filtering.* USA : School of Computer Science, Carnegie Mellon University, 2017.

24. *Nouveaux défis du pluralisme juridique.* **María Teresa Sierra, Rebecca Lemos Igreja.** Brasil : Open Edition Journals, 2019.

25. Analyse en composantes principales (ACP) : définition et cas d'usage. *JDN.* [En ligne] 19 Mai 2021. [Citation : 05 juin 2021.]

26. **Jean-François Pessiot, Tuong Vinh Truong.** *Filtrage Collaboratif avec un Algorithme d'Ordonnancement.* 2021.

27. **Jean-François Pessiot, Vinh Truong, Nicolas UsunierMassih-Reza Amini et Patrick Gallinari.** *Factorisation en matrices non négatives pour.* Paris : Laboratoire d'informatique, 2006.

28. **Lhérisson, Pierre-René.** *Système de recommandation équitable d'œuvres numériques.* s.l. : Laboratoire Hubert Curien, 2018.

29. Anaconda. *Intelligence artificielle.* [En ligne] [Citation : 05 Juin 2021.] https://intelligence-artificielle.agency/anaconda/.

30. Google_colab. *Le dataScientist.* [En ligne] [Citation : 06 Juin 2021.] https://ledatascientist.com/google-colab-le-guide-ultime/.

31. Python. *le bigdata.* [En ligne] [Citation : 07 Juin 2021.] https://www.lebigdata.fr/python-langage-definition.

32. **Mendjel, Amani.** *La prédiction des familles de protéines en utilisant le réseau .* Annaba : Université BADJI MOKHTAR , 2020.

Webographie

29. Anaconda. *Intelligence artificielle.* [En ligne] [Citation : 05 Juin 2021.] https://intelligence-artificielle.agency/anaconda/.

30. Google_colab. *Le dataScientist.* [En ligne] [Citation : 06 Juin 2021.] https://ledatascientist.com/google-colab-le-guide-ultime/.

31. Python. *le bigdata.* [En ligne] [Citation : 07 Juin 2021.] https://www.lebigdata.fr/python-langage-definition.