République Tunisienne

Ministère de l'Enseignement Supérieur

et de la Recherche Scientifique

Université de Tunis El Manar

Institut Supérieur d'Informatique d'El Manar

Rapport de Projet de stage d'été

Présenté en vue de l'obtention du

Diplôme National d'Ingénieur en Sciences Appliquées et Technologiques

Spécialité : Systèmes embarquées et objets connectés

Par

Balloumi Wahida

Implémentation et validation d'un système de recommendation

Encadrant professionnel : **Madame Karoui Wafa** Maitre Assistante

Réalisé au sein de ISI Ariana

Année Universitaire 2019/2020

République Tunisienne

Ministère de l'Enseignement Supérieur

et de la Recherche Scientifique

Université de Tunis El Manar

Institut Supérieur d'Informatique d'El Manar

Rapport de Projet de stage d'été

Présenté en vue de l'obtention du

Diplôme National d'Ingénieur en Sciences Appliquées et Technologiques

Spécialité : Systèmes embarquées et objets connectés

Par

Balloumi Wahida

Implémentation et validation d'un système de recommendation

Encadrant professionnel : **Madame Karoui Wafa** Maitre Assistante

Réalisé au sein de ISI Ariana

Année Universitaire 2019/2020

Dédicaces

Je dédie ce travail à tous ce que j'aime .

Balloumi Wahida

Remerciements

Je souhaiterais exprimer ma gratitude à Madame Karoui Wafa pour son encadrement, ses explications, sa disponibilité et surtout son assistance lors de l'élaboration du projet.

Je garde une place particulière à mes parents, mon frère et ma soeur qui sont toujours à mon coté.

Enfin, j'adresse mes plus sincères remerciements à tous mes proches, mes amis, et mes collègues, qui m'ont toujours soutenu et encouragé au cours de la réalisation de ce projet.

Table des matières

Table des figures

Liste des tableaux

Liste des abréviations

- **AI** = **A**artifial **I**ntellegent
- **FC** = **F**iltrage **C**ollaboratif
- **MF** = **M**atrix **F**actorisation
- **ML** = **M**achine **L**earning

Introduction générale

Face au volume croissant d'informations en ligne, les systèmes de recommandation constituent un moyen efficace pour surmonter le problème de surcharge d'informations. L'utilité des systèmes de recommandation ne peut être surestimée, compte tenu de son adoption généralisée dans de nombreuses applications Web, ainsi que de leur impact potentiel sur l'amélioration de nombreux problèmes liés à un choix excessif.Alors que l'internet continue à mûrir et devient plus accessible à l'utilisateur, la quantité d'informations disponibles augmente de manière exponentielle. En conséquence, il est de plus en plus difficile de trouver des informations utiles et pertinentes.En outre, une grande partie de l'information disponible est hautement subjective et est donc difficile à l'évaluer uniquement à l'aide d'algorithmes.Pour cela l'idée du filtrage collaboratif (FC) a été lancée, et elle fait l'objet de recherches approfondies.[1]
C'est dans ce cadre que s'intègre notre projet de stage d'été au sein de l'ISI Ariana. Le travail consiste à concevoir et réaliser un système de recommandation à base de réseaux du neurones et la factorisation matricielle.
Le présent rapport qui couronne le travail que nous a été confié, s'articule autour de quatre chapitres.

Le premier chapitre s'intéresse au cadre général de notre travail. Il débute par la présentation de l'organisme d'accueil suivi par une description du projet.

Dans le deuxième chapitre, nous présentons l'analyse et la spécification des besoins où nous décrivons les besoins fonctionnels et non fonctionnels.

L'objet de troisième chapitre est la conception de notre système et les choix technologiques.

Le quatrième chapitre présente la réalisation du travail.

Enfin, nous clôturons ce rapport par une conclusion générale dans laquelle nous donnons un résumé de notre travail.

Chapitre 1

Présentation de cadre du projet

Plan

Introduction

Dans ce chapitre, nous présenterons le cadre général du projet.Tout d'abord la présentation de l'entreprise d'accueil. Plus tard nous examinerons la problèmatique. Enfin nous exposerons le travail demandé.

1.1 Organisme d'accueil [2]

Vu les circonstances exceptionnelles suite à la pandémie de covid-19, les enseignantes de l'ISI nous proposent des sujets.La figure 1.1 présente le logo de ISI Ariana.

Figure 1.1: Logo ISI Ariana

L'institut supérieur d'informatique relevant de l'Université de Tunis El Manar a été crée par décret N° 1912 du 14 Aout 2001.Il offre les cycles de formation suivante :

- Cycle de formation en Licence en Informatique d'une durée de trois ans ;
- Cycle de formation d'Ingénieurs en Informatique d'une durée de trois ans ;
- Cycle de formation en Mastère Professionnel en Informatique "Sécurité des Systèmes Informatiques communicants et Embarqués" (SSICE) ;
- Cycle de formation en Mastère Professionnel en Informatique "Logiciels Libres" (MP2L) en collaboration avec l'Université Virtuel de Tunis ;
- Cycle de formation en Mastère de Recherche en Informatique avec deux parcours : SIIVA et GL.

1.2 Problématique

Dans cette partie nous allons présenté la problématique du projet :

1.2.1 Système de recommandation

Système de recommandation est une forme spécifique de filtrage de l'information.Il cherche à prédire les préférences de l'utilisateur et requiert généralement 3 étapes qui sont :

— Recueillir de l'information sur l'utillisateur.

— Batir une matrice contenant l'information recueillie.

— Extraire à partir de cette matrice une liste de recommandations.

Les systèmes de recommandation sont des téchniques fournissant des suggestions pour les articles qui peuvent être utiles à un utilisateur.Les suggestions concernent divers processus de décision tels que les articles à acheter, la musique à écouter ou les nouvelles en ligne à lire.
les systèmes de recommandation jouent un rôle important dans des sites internet aussi bien classés que Amazon, youtube et facebook.

1.2.2 Approches de système de recommandation[3]

Les systèmes de recommandation sont classés en fonction de l'approche utilisée, il y a deux approches qui sont :

— Filtrage à base de contenu

L'utilisateur se verra recommander des éléments semblables à ceux qu'il a préférés dans le passé. Les systèmes de recommandation basés sur le contenu s'appuient sur des évaluations effectuées par un utilisateur sur un ensemble de documents ou items. L'objectif est alors de comprendre les motivations l'ayant conduit à juger comme pertinent ou non un item donné. Le système peut alors proposer à l'utilisateur un choix parmi de nouveaux items jugés proches des items qu'il a précédemment appréciés.
Une telle approche ne requiert pas une grande communauté d'utilisateurs ou un gros historique d'utilisation du système.La figure 1.3 présente l'approche de filtrage à base de contenu.

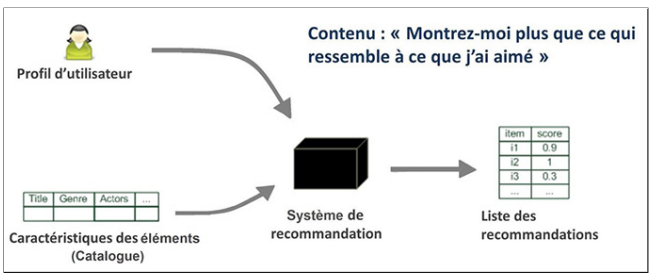


Figure 1.2: filtrage à base de contenu

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 .

— 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 ;

— 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.

— Filtrage collaboratif

Le filtrage collaboratif est parmi les technologies les plus populaires dans le domaine des systèmes de recommandation. La méthode consiste à faire des prévisions automatiques sur les intérêts d'un utilisateur en collectant des avis de nombreux utilisateurs. Il fonctionne en recommandant des Items en fonction du comportement passé des utilisateurs similaires, en effectuant une corrélation entre des utilisateurs ayant des préférences et intérêts similaires et cette tâche est faite en utilisant des méthodes qui collectent et analysent des données sur le comportement, les activités, les préférences des utilisateurs.
Il produit des recommandations en calculant la similarité entre les préférences d'un utilisateur et celles d'autres utilisateurs.La figure 1.4 présente l'approche de filtrage collaboratif.

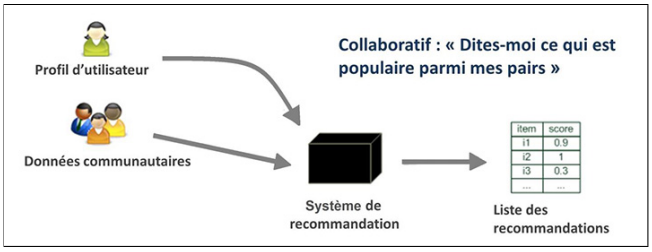


Figure 1.3: filtrage collaboratif

Les systèmes de recommandation collaboratifs ont comme avantages :

— utiliser les scores d'autres utilisateurs pour évaluer l'utilité des éléments ;

— trouver des utilisateurs ou groupes d'utilisateurs dont les intérêts correspondent à l'utilisateur courant ;

— il y a d'utilisateurs plus il y a de scores : meilleurs sont alors les résultats.

Cette approche peuvent reposer sur diverses méthodes, Il existe deux types de recommandation de filtrage collaboratif présenté dans la figure 1.5.

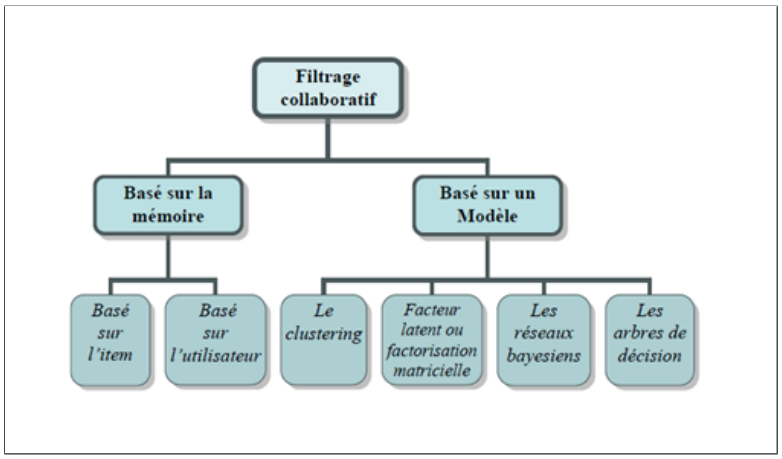


Figure 1.4: les méthodes filtrage collaboratif

— Memory Based

Les modèles Memory based se base sur la similarité calculé par les différentes métriques d'évaluation.Il existe deux types de Memory Model :
User-Item based : cet algorithme va prendre un utilisateur, trouve les utilisateurs les plus similaires à lui, Puis recommande les items aimé par ces utilisateurs (ça prend un user et retourne des items). En général on l'utilise pour dire "Les utilisateurs qui sont similaires à vous, ont aimé aussi"
Item-Based : prend un item, cherche les utilisateurs qui ont aimé cet item et recommande les items aimés par ces utilisateurs (ça prend un item et retourne une liste des items). En général on l'utilise pour dire "Les utilisateurs qui ont aimé ça, ont aimé aussi"

— Model Based :

Les méthodes basées sur un modèle ont été intégrées aux systèmes de recommandation pour améliorer et remédier aux problèmes des méthodes basées sur la mémoire. Les algorithmes basés sur le modèle se basent aussi sur les évaluations précédentes (les profils) des usagers, mais cette méthode ne calcule pas directement les prédictions, elle classifie les usagers suivant des groupes ou d'apprendre les modèles à partir de leurs données Pour la construction du modèle plusieurs méthodes sont utilisées.les méthodes basées sur le modèle telles que le clustering, la factorisation matricielle, les réseaux bayesiens, les arbres de décision.
Dans notre travail, nous allons nous focaliser essentiellement sur la factorisation matricielle aussi dite décomposition matricielle. Elle se base sur la multiplication de matrices ou Matrix Factorisation (MF) : Méthode d'apprentissage non supervisé de décomposition et de réduction de dimensionnalité en utilisant les caractères latents (k : espace des caractères cachés).

Le but de ces algorithmes est de trouver les préférences latentes des utilisateurs (sous forme d'une matrice P) et des items (sous forme d'une matrice Q) à partir des notes attribués. Puis, la multiplication de ces deux matrices de structure de faible rang donne la matrice de prédiction.

1.3 Travail demandé

Il s'agit d'implémenter un système de recommandation qui peut reposer sur diverses méthodes et être évalué sur diverses collections. La plateforme doit fournir à l'utilisateur la possibilité de choisir une méthode parmi quatre choix à appliquer sur une collection sélectionnée parmi trois choix offerts par la plateforme Epinion, MovieLens, et Netflix.

Conclusion

Tout au long de ce chapitre, nous expliquons les concepts de base qui présentent l'objectif de notre projet, nous mentionnons également l'énoncé du problèmatique et le travail demandé. Dans le prochain chapitre, nous présenterons les besoins.

Chapitre 2

Analyse et spécification des besoins

Plan

Introduction

L'analyse et la spécification des besoins représentent une étape importante. Dans ce chapitre, nous allons détailler en premier lieu les besoins fonctionnels et en deuxième lieu les besoins non fonctionnels.

2.1 Besoins fonctionnels

En général, les besoins fonctionnels consistent à satisfaire l'utilisateur.

2.1.1 Problématique

Les systèmes de recommandation cherchent à prédire l'« avis » de l'utilisateur par des approches, nous parlerons de l'approche basée sur le contenu et de l'approche de filtrage collaboratif. Ces approches peuvent reposer sur diverses méthodes.

2.1.2 Formulation du problème

Il s'agit d'implémenter un système de recommandation qui peut reposer sur diverses méthodes et être évalué sur diverses collections. La plateforme doit fournir à l'utilisateur la possibilité de choisir une méthode parmi quatre choix à appliquer sur une collection sélectionnée parmi trois choix offerts par la plateforme Epinion, MovieLens, et Netflix.
Les méthodes sont déjà implémentées et disponibles dans les bibliothèques Java et Python.
Une fois les collections et les méthodes sont intégrées, les résultats sont évalués selon les protocoles d'évaluation de chaque collection. Chaque collection a son système d'évaluation.
Les scripts d'évaluation existent en ligne. L'output correspond à des fichiers structurés.Une étape de visualisation graphique vient illustrer les résultats dans les fichiers structurés.

— Etape 1 : Interface et Collections.

— Etape 2 : Méthodes.

— Etape 3 : Evaluation et étude comparative.

2.2 Besoins non fonctionnels

Les besoins non fonctionnels représentent l'ensemble des contraintes liées au bon fonctionnement du système en termes de qualité et de performance.

Nous devons prendre en compte de ces besoins dans notre conception pour aboutir à une compréhension globale et complète de toutes les fonctionnalités du système. En effet, le système doit assurer les besoins non fonctionnels suivants :

- Fiabilité : l'application doit réaliser les fonctionnalités définies avec un degré de précision.
- Disponibilité : le système doit accomplir les fonctionnalités définies dans des conditions données à un instant donné.
- Performance : le temps de réponse et la rapidité d'accès aux différentes données doivent être pris en considération.
- Contraintes de programmation : l'implémentation de notre solution ne doit pas influer sur les autres fonctionnalités.
- Ergonomie et souplesse : le système doit offrir une interface pratique pour l'utilisateur avec un maximum de confort et d'efficacité.
- Exactitude : l'application doit satisfaire les spécifications déjà établies et les attentes des utilisateurs.

Conclusion

Après avoir précisé les différents besoins fonctionnels et non fonctionnels du projet.Nous présenterons les détails conceptuels et les choix technologiques dans le chapitre suivant.

Chapitre 3

Conception et choix technologiques

Plan

Introduction

Nous allons réserver ce chapitre pour la phase de conception qui représente une étape déterminante dans le cycle de développement d'un projet.Nous allons exposer en premier temps l'architecture générale de réseau de neurones puis nous allons expliquer la factorisation matricielle.En suite nous allons détailler l'architecture d'un système de recommandation basé sur le réseau de neurones et la factorisation matricielle.Pour finir, nous allons présenter les choix technologiques.

3.1 Conception

3.1.1 Réseaux de neurones[4]

Un réseau de neurones artificiels ou Neural Network est un système informatique s'inspirant du fonctionnement du cerveau humain pour apprendre.La figure 3.1 présente le neurone biologique

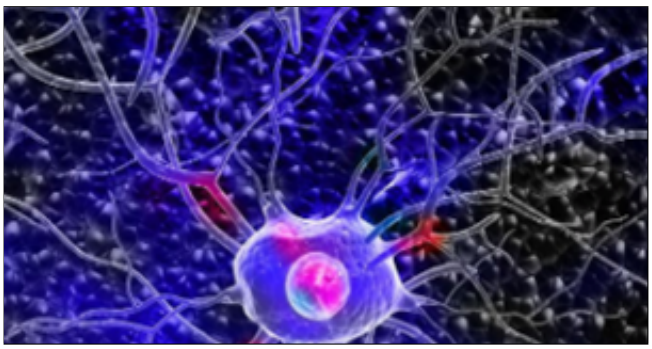


Figure 3.1: le neurone biologique

Cette technologie se développe à grande vitesse, et ses cas d'usage se multiplient dans tous les secteurs. Selon les experts, l'IA est amenée à bouleverser tous les aspects de notre société dans les années à venir. En règle générale, un réseau de neurones repose sur un grand nombre de processeurs opérant en parallèle et organisés en tiers. Le premier tiers (input layer) reçoit les entrées d'informations brutes, un peu comme les nerfs optiques de l'être humain lorsqu'il traite des signaux visuels. Par la suite, chaque tiers (hidden layer) reçoit les sorties d'informations du tiers précédent. On retrouve le même processus chez l'homme, lorsque les neurones reçoivent des signaux en provenance des neurones proches du nerf optique. Le dernier tiers (output layer), quant à lui, produit les résultats du système. La figure 3.2 présente l'architecture du réseaux de neurones.

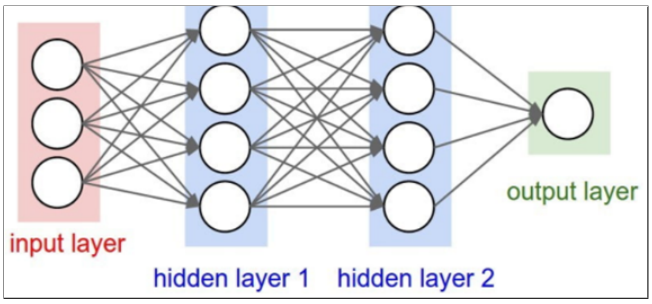


Figure 3.2: réseaux de neurones

— L'apprentissage : la propagation des informations entre les différents tiers de neurones peut varier. Dans la variante la plus simple, celle du réseau de neurones dit « feed-forward », les informations passent directement de l'entrée aux noeuds de traitement puis aux sorties.

— Loss function : Utiliser pour mesurer la distance entre la prédiction et le résultat réel. Les métriques utilisées pour valider la précision de la prédiction sont les suivantes : l'erreur moyenne absolue (MAE) et l'erreur quadratique moyenne (RMSE).

— Optimizer : Utiliser pour la mise à jour de poids de model. Les optimizer utilisées sont adam, sgd...

La figure 3.3 présente l'architecture de fonctionnement de réseaux du neurones.

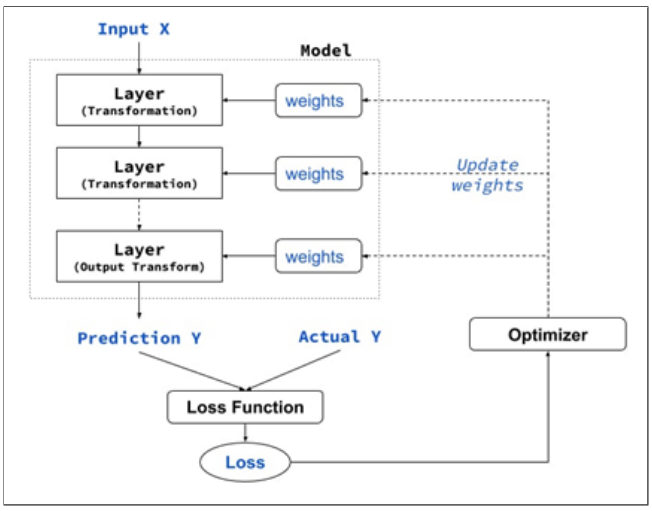


Figure 3.3: l'architecture de fonctionnement de réseaux des neurones

3.1.2 Factorisation matricielle

Un problème de factorisation matricielle est un problème d'optimisation où nous supposons que le modèle est le produit de matrices de facteurs latents. nous considèrons que le nombre de variables latentes est très inférieure à la taille des données. Le but de la factorisation matricielle est d'obtenir une reconstruction qui jouera le rôle de fonction score. Le principal intérêt de la

factorisation, particulièrement dans la recommandation, est de fournir une représentation de faible rang d'un modèle de grande dimension.

3.1.3 Création d'un système de recommandation basé sur les réseaux de neurones et la factorisation matricielle

Pour la réalisation de notre système, nous sommes basé sur la méthode de factorisation matricielle de l'approche filtrage collaboratif. Tout d'abord, nous avons un ensemble d'utilisateurs (ou spectateurs) U et un ensemble d'articles (ou films) I. Soit X la matrice de taille (m x n) qui contient toutes les notes que les utilisateurs ont attribuées aux articles (rating matrix).
La figure 3.4 présente la factorisation matricielle de système de recommandation.

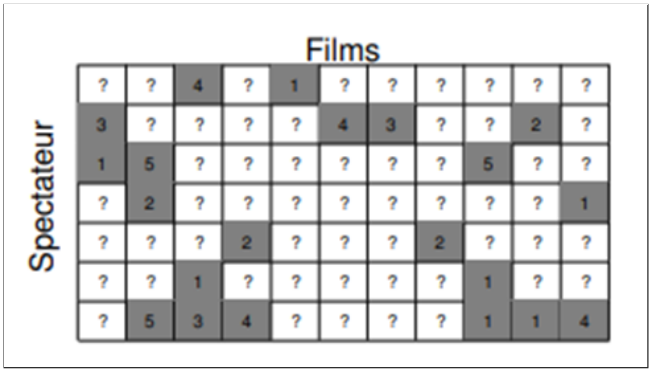


Figure 3.4: factorisation matricielle de système de recommandation

Pour le réseau de neurone :

— Input : Utilisateur et Article

— Output : Rating

La figure 3.5 présente l'architecture de réseau du neurone de système de recommandation.

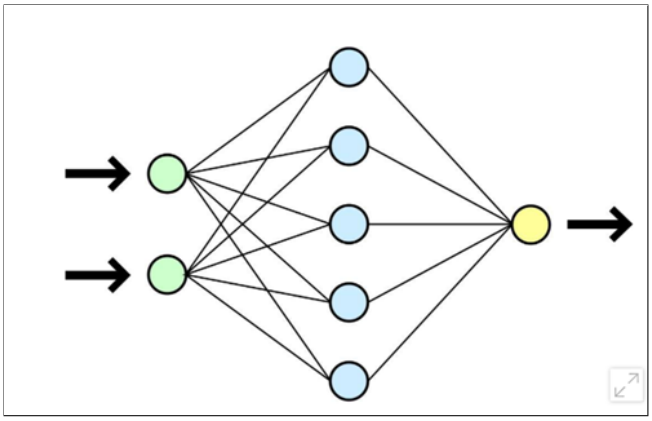


Figure 3.5: l'architecture de réseau de neurone de système de recommandation

Les systèmes de factorisation de matrice peuvent être considérés implicitement comme des systèmes d'apprentissage profond. Ils peuvent être vus comme une approche simpliste des réseaux de neurones. 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 signifiant 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.
La figure 3.6 présente l'architecture de réseau de neurone de système de recommandation [5].

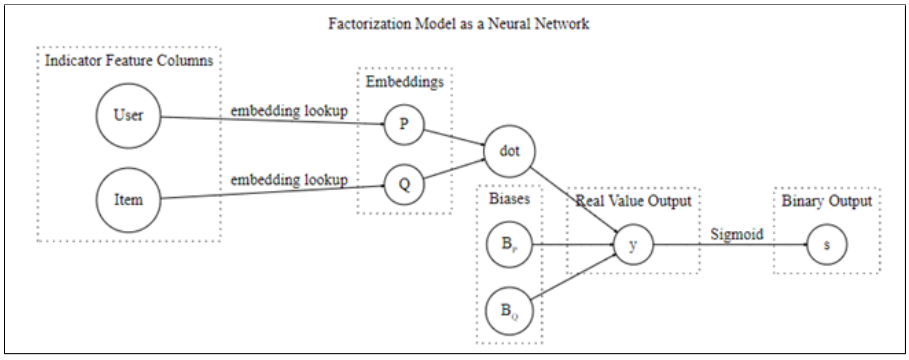


Figure 3.6: l'architecture de réseau de neurone de système de recommandation

3.2 Choix technologiques

3.2.1 Langage Python[6]

Python est un langage de programmation interprété, de haut niveau et polyvalent. La figure 3.7 présente le logo de python.

Figure 3.7: Python

Nous avons chosit la longage de programmation Python grâce à ses bibliothèques tels que :

— Pandas : Pandas est un package Python open-source qui fournit des structures de données et des outils d'analyse des données de haute performance et faciles à utiliser pour les données étiquetées dans le langage Python. Pandas signifie Python Data Analysis Library. Pandas est conçu pour la manipulation rapide et facile des données : La lecture, l'agrégation et la visualisation. La figure 3.8 présente la bibliothèque pandas.

Figure 3.8: Pandas

— NumPy : C'est l'un des paquets les plus fondamentaux en Python. Numpy est un paquet de traitement de tableaux à usage général. Il fournit des objets de tableaux multidimensionnels de haute performance et des outils pour travailler avec ces tableaux. Numpy est un conteneur efficace de données multidimensionnelles génériques. La figure 3.9 présente la bibliothèque numpy.

Figure 3.9: NumPy

— Matplotlib : C'est une autre bibliothèque de la pile Scipy. Matplotlib dessine des figures en 2D. La figure 3.10 présente la bibliothèque matplotlib.

Figure 3.10: Matplotlib

— Seaborn : Lorsque vous lisez la documentation officielle sur Seaborn, il est défini comme la bibliothèque de visualisation de données basée sur Matplotlib qui fournit une interface de haut niveau pour dessiner des graphiques statistiques attrayants et informatifs. Pour le dire simplement, seaborn est une extension de Matplotlib avec des fonctionnalités avancées.

— Scikit Learn : Scikit Learn est une robuste bibliothèque d'apprentissage automatique pour Python. Il dispose d'algorithmes de ML comme Svms, forêts aléatoires, k-moyens clustering, le clustering spectral, le décalage moyen, la validation croisée ... etc.
La figure 3.11 présente la bibliothèque scikit learn.

Figure 3.11: Scikit Learn

— TensorFlow : Tensorflow est une bibliothèque d'intelligence artificielle qui aide les développeurs à créer des réseaux neuronaux à grande échelle avec de nombreuses couches en utilisant des graphiques de flux de données. Tensorflow facilite également la création de modèles Deep Learning, pousse l'état de l'art en ML/AI et permet un déploiement facile des applications alimentées en ML. La figure 3.12 présente la bibliothèque tenserflow.

Figure 3.12: Tensorflow

— Keras : Keras est l'API de haut niveau de Tensorflow pour le développement et la formation du code Deep Neural Network. Il s'agit d'une bibliothèque réseau neuronal open-source en Python. Avec Keras, la modélisation statistique, le travail avec les images et le texte est beaucoup plus facile surtout avec le codage simplifié pour l'apprentissage en profondeur. La figure 3.13 présente la bibliothéque keras.

Figure 3.13: Keras

3.2.2 Editeur de code Python « Pycharm »

Nous avons choisit l'éditeur de code PyCharm car il est un environnement de développement intégré utilisé spécifiquement pour le langage Python. nous avons le utilisée pour la création de l'interface d'acceuil. La figure 3.14 présente l'éditeur pycharm.

Figure 3.14: Pycharm

3.2.3 Notebook « Google Colab »

Nous avons choisit Google Colab parceque il nous permet d'entraîner des modèles directement dans le cloud sans d'avoir besoin d'installer quoi que ce soit sur notre ordinateur. La figure 3.15 présente le logo de google colab.

Figure 3.15: Google Colab

Conclusion

Au cours de ce chapitre nous avons détaillé la partie conception de projet, nous avons vu son architecture globale et détaillée. Dans le prochain chapitre, nous allons s'intéresser à la phase de la réalisation.

Chapitre 4

Réalisation

Plan

Introduction

Dans ce chapitre, nous allons présenter le coté applicatif de notre travail.Nous allons exposer l'interface d'accueil et les différentes étapes d'implémentation.Aussi nous allons comparer les résultats des évaluations.

4.1 Interface d'accueil

D'abord, nous avons créé une interface d'accueil qui fournir à l'utilisateur la possibilité de choisir une méthode parmi quatre choix à appliquer sur une collection sélectionnée parmi trois choix offerts par la plateforme Epinion, MovieLens, et Netflix. La figure 4.1 présente l'interface d'accueil.

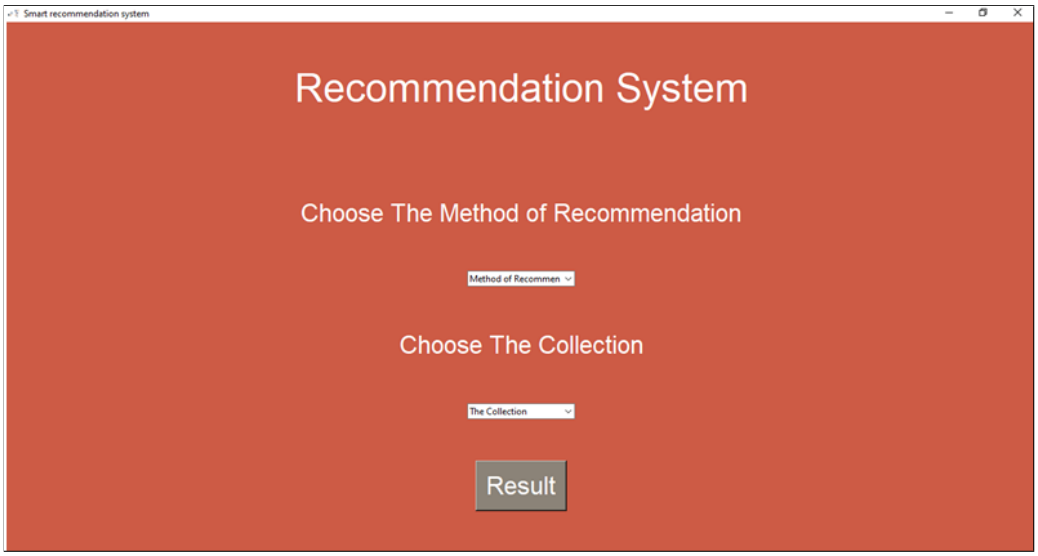


Figure 4.1: interface d'accueil

4.2 Etapes d'implémentation du système de recommandation

Nous allons présenter les differents phases d'implémentation pour réaliser un système de recommandation.

4.2.1 Importer les données

Nous utilisons 3 collections du plateformes MovieLens , Epinion et Netflix :

— MovieLens est un système de recommandation basé sur le Web et une communauté virtuelle qui recommande des films à ses utilisateurs à regarder, en fonction de leurs préférences de films en utilisant un filtrage collaboratif des classements de films et des critiques de films des membres[7].

La collection de données comporte 2 fichiers csv : Items.csv et Ratings.csv

La figure 4.2 présente la platforme movieslens.

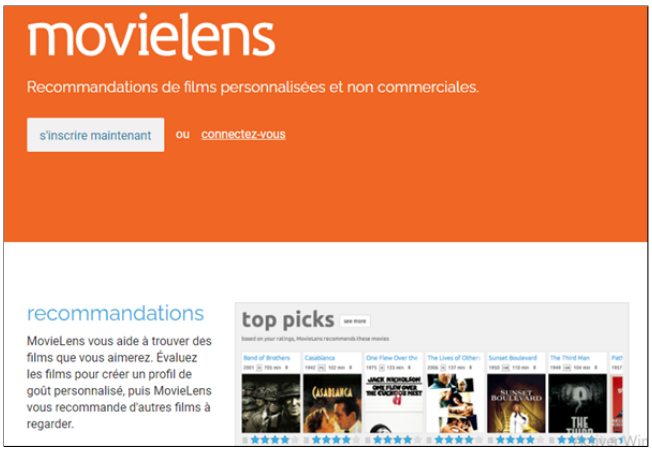


Figure 4.2: MovieLens[8]

— Epinions.com était un site général d'avis de consommateurs créé en 1999. Epinions a été acquis en 2003 par DealTime, plus tard Shopping.com, qui a été acquis par eBay en 2005. Les utilisateurs d'Epinions pouvaient accéder à des avis sur une variété d'articles. La collection de données contient fichier texte : datarating.txt.

La figure 4.3 présente la platforme Epinion.

Figure 4.3: Epinion[9]

— Netflix est un fournisseur américain de services technologiques et multimédias et une société de production dont le siège est à Los Gatos, en Californie. Netflix a été fondé en 1997 par Reed Hastings et Marc Randolph à Scotts Valley, en Californie. L'activité principale de la société est son service de diffusion en continu par abonnement qui offre la diffusion en ligne d'une bibliothèque de films et de séries télévisées, y compris ceux produits en interne.[10]. La collection de données comporte 4 fichiers texte.

La figure 4.4 présente la plateforme Netflix.

Figure 4.4: netflix[11]

la collection de plateforme MovieLens a la taille la plus faible taille de donnée puis la collection de plateforme Epinion ensuite la collection de plateforme Netflix qui a la plus grande taille de données.

Importer les bibliothèques nécessaires et les données :

La figure 4.5 présente les bibliothèques.

```
[ ]  import numpy as np
     import pandas as pd
     import matplotlib.pyplot as plt
     import seaborn as sns

     from sklearn.preprocessing import LabelEncoder

[ ]  from keras.models import Model
     from keras.layers import Input, Embedding, Flatten, Dot, Add, Lambda, Activation, Reshape, Concatenate, Dense
     from keras.regularizers import l2
     from keras.constraints import non_neg
     from keras.utils import plot_model
     from keras.utils.vis_utils import model_to_dot
```

Figure 4.5: import bibliothèque

La figure 4.6 présente commment importer la données.

```
[ ]  df = pd.read_csv('/content/combined_data_4.txt', header = None, names = ['Cust_Id', 'Rating'], usecols = [0,1])
```

Figure 4.6: import les données

4.2.2 Nettoyer les données

Encodage : par la méthode « LabelEncoder » de la classe preprocessing de la bibliothèque sklearn qui normaliser les étiquettes de sortie. La figure 4.7 présente comment nettoyer les données.

```
[ ] # Data Encoding
    DATA, user_encoder, item_encoder = encode_user_item(df_ratings, "user_id", "movie_id", "rating", "unix_timestamp")
```

Figure 4.7: Nettoyage de données

4.2.3 Diviser les données : données d'apprentissage et données de teste

— Données d'apprentissage : pour construire l'algorithme de prédiction et ajuster les poids sur le réseau neuronal (80 de taille de données).

— Données de teste : pour tester la solution finale à fin de confirmer la puissance prédictive réelle de réseau(20 de taille de données).

La figure 4.8 présente la division de données.

```
train, test = random_split(DATA, [0.8, 0.2])
```

Figure 4.8: diviser les données

4.2.4 Implémentation des méthodes

— Factorisation matricelle : La figure 4.9 présente l'architecture de la méthode de factorisation matricelle.

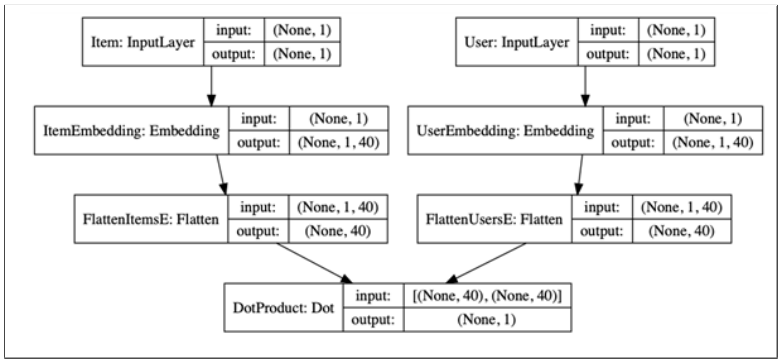


Figure 4.9: Méthode de factorisation matricelle

La figure 4.10 présente l'implèmentaion de la méthode de factorisation matricelle.

```
[ ]  def ExplicitMF (n_users, n_items, n_factors):

        # Item Layer
        item_input = Input(shape=[1], name='Item')
        item_embedding = Embedding(n_items, n_factors,
                                   embeddings_regularizer=l2(1e-6),
                                   name='ItemEmbedding')(item_input)
        item_vec = Flatten(name='FlattenItemsE')(item_embedding)

        # User Layer
        user_input = Input(shape=[1], name='User')
        user_embedding = Embedding(n_users, n_factors,
                                   embeddings_regularizer=l2(1e-6),
                                   name='UserEmbedding')(user_input)
        user_vec = Flatten(name='FlattenUsersE')(user_embedding)

        # Dot Product of Item and User
        rating = Dot(axes=1, name='DotProduct')([item_vec, user_vec])

        # Model Creation
        model = Model([user_input, item_input], rating)

        # Compile Model
        model.compile(loss='mean_squared_error', optimizer='Adam')

        return model
```

Figure 4.10: L'implèmentation du méthode de factorisation matricelle

— Factorisation matricelle avec biais : La figure 4.11 présente l'architecture de la méthode de factorisation matricelle avec biais.

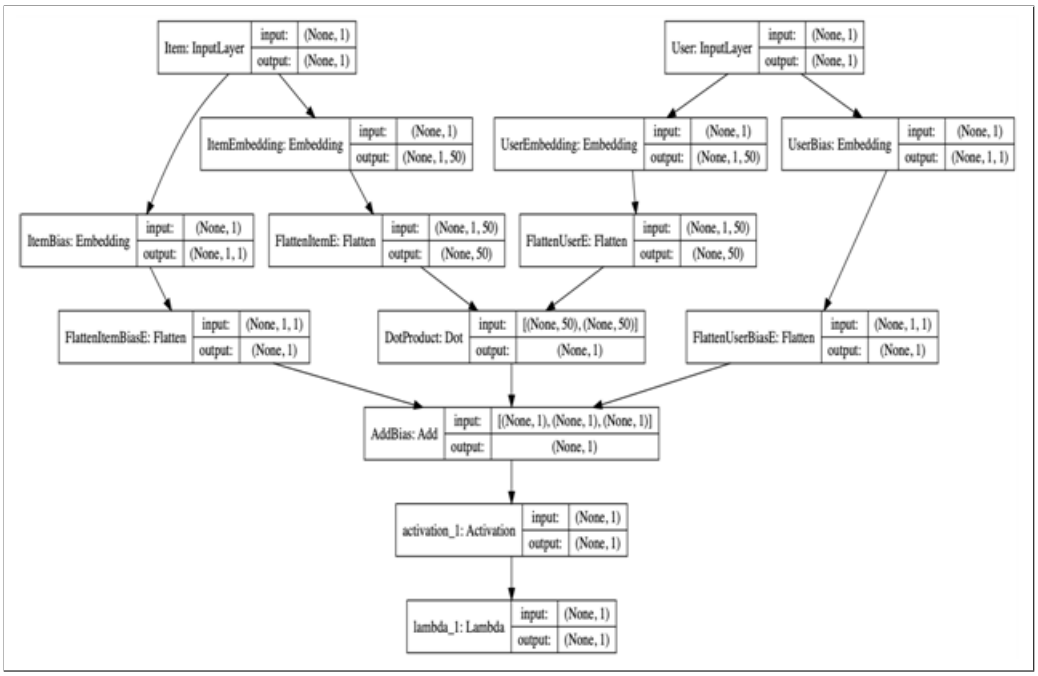


Figure 4.11: Méthode de factorisation matricelle avec biais

La figure 4.12 présente l'implèmentaion de la méthode de factorisation matricelle avec biais.

```
[ ]  def Explicit_MF_Bias(n_users, n_items, n_factors):

        # Item Layer
        item_input = Input(shape=[1], name='Item')
        item_embedding = Embedding(n_items, n_factors, embeddings_regularizer=l2(1e-6), name='ItemEmbedding')(item_input)
        item_vec = Flatten(name='FlattenItemE')(item_embedding)

        # Item Bias
        item_bias = Embedding(n_items, 1, embeddings_regularizer=l2(1e-6), name='ItemBias')(item_input)
        item_bias_vec = Flatten(name='FlattenItemBiasE')(item_bias)

        # User Layer
        user_input = Input(shape=[1], name='User')
        user_embedding = Embedding(n_users, n_factors, embeddings_regularizer=l2(1e-6), name='UserEmbedding')(user_input)
        user_vec = Flatten(name='FlattenUserE')(user_embedding)

        # User Bias
        user_bias = Embedding(n_users, 1, embeddings_regularizer=l2(1e-6), name='UserBias')(user_input)
        user_bias_vec = Flatten(name='FlattenUserBiasE')(user_bias)

        # Dot Product of Item and User & then Add Bias
        DotProduct = Dot(axes=1, name='DotProduct')([item_vec, user_vec])
        AddBias = Add(name="AddBias")([DotProduct, item_bias_vec, user_bias_vec])

    # Scaling for each user
    y = Activation('sigmoid')(AddBias)
    rating_output = Lambda(lambda x: x * (max_rating - min_rating) + min_rating)(y)

    # Model Creation
    model = Model([user_input, item_input], rating_output)

    # Compile Model
    model.compile(loss='mean_squared_error', optimizer=Adam(lr=0.001))

    return model
```

Figure 4.12: L'implèmentation du méthode de factorisation matricelle avec biais

— Deep Matrix Factorization La figure 4.13 présente l'architecture de la méthode de factorisation matricelle profonde.

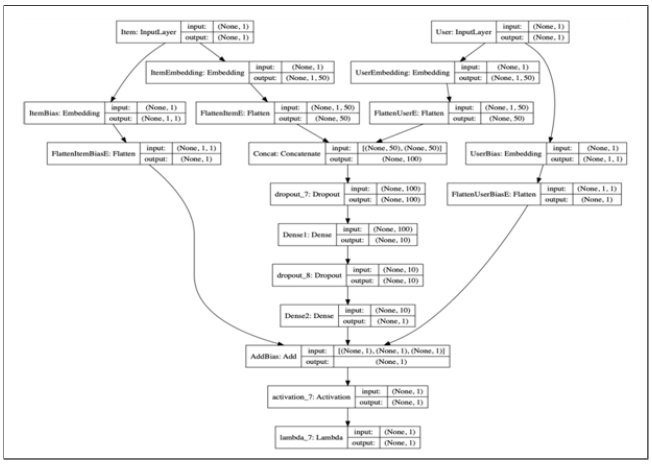


Figure 4.13: Méthode de factorisation matricelle profonde

La figure 4.14 présente l'implèmentaion de la méthode de factorisation matricelle profonde.

```python
def Deep_MF(n_users, n_items, n_factors):

    # Item Layer
    item_input = Input(shape=[1], name='Item')
    item_embedding = Embedding(n_items, n_factors, embeddings_regularizer=l2(1e-6),
                               embeddings_initializer='glorot_normal',
                               name='ItemEmbedding')(item_input)
    item_vec = Flatten(name='FlattenItemE')(item_embedding)

    # Item Bias
    item_bias = Embedding(n_items, 1, embeddings_regularizer=l2(1e-6),
                          embeddings_initializer='glorot_normal',
                          name='ItemBias')(item_input)
    item_bias_vec = Flatten(name='FlattenItemBiasE')(item_bias)

    # User Layer
    user_input = Input(shape=[1], name='User')
    user_embedding = Embedding(n_users, n_factors, embeddings_regularizer=l2(1e-6),
                               embeddings_initializer='glorot_normal',
                               name='UserEmbedding')(user_input)
    user_vec = Flatten(name='FlattenUserE')(user_embedding)

    # User Bias
    user_bias = Embedding(n_users, 1, embeddings_regularizer=l2(1e-6),
                          embeddings_initializer='glorot_normal',
                          name='UserBias')(user_input)
user_bias_vec = Flatten(name='FlattenUserBiasE')(user_bias)

# Dot Product of Item and User & then Add Bias
Concat = Concatenate(name='Concat')([item_vec, user_vec])
ConcatDrop = Dropout(0.5)(Concat)

kernel_initializer='he_normal'

# Use Dense to learn non-linear dense representation
Dense_1 = Dense(10, kernel_initializer='glorot_normal', name="Dense1")(ConcatDrop)
Dense_1_Drop = Dropout(0.5)(Dense_1)
Dense_2 = Dense(1, kernel_initializer='glorot_normal', name="Dense2")(Dense_1_Drop)


AddBias = Add(name="AddBias")([Dense_2, item_bias_vec, user_bias_vec])


# Scaling for each user
y = Activation('relu')(AddBias)
rating_output = Lambda(lambda x: x * (max_rating - min_rating) + min_rating)(y)

# Model Creation
model = Model([user_input, item_input], rating_output)

# Compile Model
model.compile(loss='mean_squared_error', optimizer=Adam(lr=0.001))

return model
```

Figure 4.14: L'implèmentation du méthode de factorisation matricelle profonde

— Deep Neural Collaborative filtering a figure 4.15 présente l'architecture de la méthode de factorisation matricelle plus profonde.

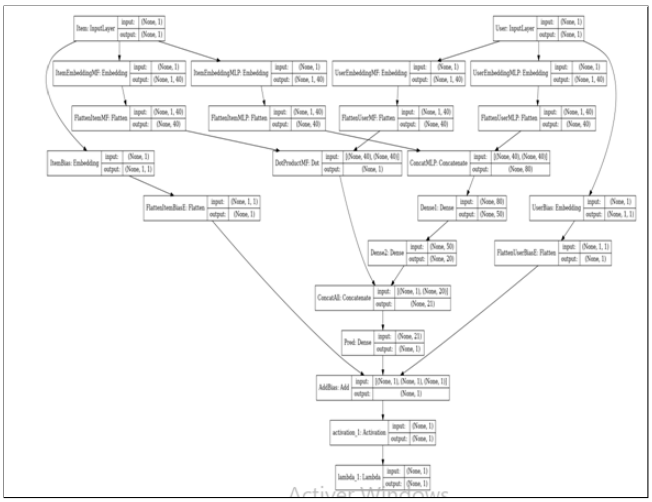


Figure 4.15: Méthode de factorisation matricelle plus profonde

La figure 4.16 présente l'implèmentaion de la méthode de factorisation matricelle plus profonde.

```
def Neural_CF(n_users, n_items, n_factors):

    # Item Layer
    item_input = Input(shape=[1], name='Item')

    # Item Embedding MF
    item_embedding_mf = Embedding(n_items, n_factors, embeddings_regularizer=l2(1e-6),
                                  embeddings_initializer='he_normal',
                                  name='ItemEmbeddingMF')(item_input)
    item_vec_mf = Flatten(name='FlattenItemMF')(item_embedding_mf)


    # Item embedding MLP
    item_embedding_mlp = Embedding(n_items, n_factors, embeddings_regularizer=l2(1e-6),
                                   embeddings_initializer='he_normal',
                                   name='ItemEmbeddingMLP')(item_input)
    item_vec_mlp = Flatten(name='FlattenItemMLP')(item_embedding_mlp)


    # User Layer
    user_input = Input(shape=[1], name='User')

    # User Embedding MF
    user_embedding_mf = Embedding(n_users, n_factors, embeddings_regularizer=l2(1e-6),
                                  embeddings_initializer='he_normal',
                                  name='UserEmbeddingMF')(user_input)
    user_vec_mf = Flatten(name='FlattenUserMF')(user_embedding_mf)

    # User Embedding MF
    user_embedding_mlp = Embedding(n_users, n_factors, embeddings_regularizer=l2(1e-6),
                                   embeddings_initializer='he_normal',
                                   name='UserEmbeddingMLP')(user_input)
    user_vec_mlp = Flatten(name='FlattenUserMLP')(user_embedding_mlp)

    # Multiply MF paths
    DotProductMF = Dot(axes=1, name='DotProductMF')([item_vec_mf, user_vec_mf])

    # Concat MLP paths
    ConcatMLP = Concatenate(name='ConcatMLP')([item_vec_mlp, user_vec_mlp])

    # Use Dense to learn non-linear dense representation
    Dense_1 = Dense(50, name="Dense1")(ConcatMLP)
    Dense_2 = Dense(20, name="Dense2")(Dense_1)

    # Concatenate MF and MLP paths
    Concat = Concatenate(name="ConcatAll")([DotProductMF, Dense_2])

    # Use Dense to learn non-linear dense representation
    Pred = Dense(1, name="Pred")(Concat)

    # Item Bias
    item_bias = Embedding(n_items, 1, embeddings_regularizer=l2(1e-5), name='ItemBias')(item_input)
    item_bias_vec = Flatten(name='FlattenItemBiasE')(item_bias)

    # User Bias
    user_bias = Embedding(n_users, 1, embeddings_regularizer=l2(1e-5), name='UserBias')(user_input)
    user_bias_vec = Flatten(name='FlattenUserBiasE')(user_bias)

    # Pred with bias added
    PredAddBias = Add(name="AddBias")([Pred, item_bias_vec, user_bias_vec])


    # Scaling for each user
    y = Activation('sigmoid')(PredAddBias)
    rating_output = Lambda(lambda x: x * (max_rating - min_rating) + min_rating)(y)

    # Model Creation
    model = Model([user_input, item_input], rating_output)

    # Compile Model
    model.compile(loss='mean_squared_error', optimizer="adam")

    return model
```

Figure 4.16: L'implèmentation du méthode de factorisation matricelle plus profonde

4.2.5 Entrainement de modèle

La figure 4.17 présente comment apprendre un réseau de neurones.

```
[ ] %%time
    output = model.fit([train.USER, train.ITEM], train.RATING,batch_size=128,
                                        epochs=2, verbose=1,validation_data= ([test.USER, test.ITEM], test.RATING))

    Epoch 1/2
    5096/5096 [==============================] - 966s 189ms/step - loss: 0.9594 - val_loss: 1.1521
    Epoch 2/2
    5096/5096 [==============================] - 972s 191ms/step - loss: 0.4054 - val_loss: 1.2259
```

Figure 4.17: phase d'apprentissage

4.2.6 Prediction et recommandation

— Collection de plateforme MovieLens : La figure 4.18 présente la recommandation de la collection de plateforme MovieLens .

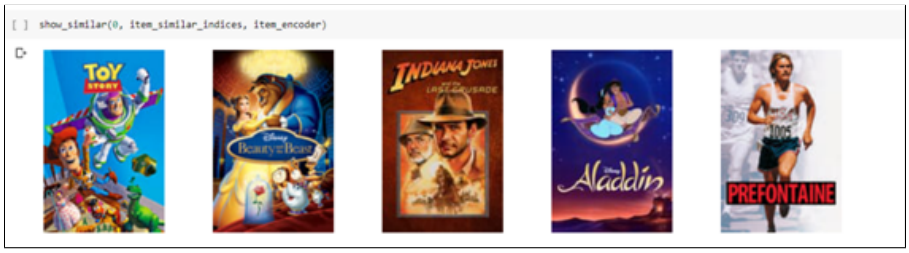


Figure 4.18: la recommandation de platforme MovieLens

— Collection de plateforme Epinion : La figure 4.19 présente la recommandation de la collection de plateforme Epinion.

Figure 4.19: la recommandation de plateforme Epinion

— Collection de plateforme Nettflix : La figure 4.20 présente la recommandation de la collection de plateforme Netflix.

Figure 4.20: la recommdation de plateforme Netflix

4.3 Evaluation des méthodes avec différents collections

Tableau 4.1: Les erreurs des collections MovieLens, Epinion et Netflix avec les différentes méthodes implèmentès

	Matrix Factorization	**Matrix Factorization with bias**	**Deep Matrix Factorization**	**Deep Neural Collaborative Filtering**
"Mean Squared error" de collection MovieLens	**0.93**	**0.84**	**0.88**	**0.97**
"Mean Squared error" de collection Epinion	**10.92**	**1.50**	**1.17**	**1.30**
"Mean Squared error" de collection Netflix	**13.26**	**1.35**	**1.25**	**1.22**

A partir de tableau nous pouvons remarquer que la meilleure méthode de recommandation pour la la collection de données :

MovieLens est Matrix Factorization with bias.

Epinion est Deep Matrix Factorization.

Netflix est Deep Neural Collaborative Filtering.

Nous pouvons conclure que les modèles du réseaux de neurones diffèrent selon la taille du collection de données. Plus de données, plus nous ajoutons de couches cachées dans le réseau.

Conclusion

Dans ce dernier chapitre, nous avons exposé les différentes phases de réalisation de notre système de recommandation, qui consiste à recommander une liste pertinente de films et des produits à un utilisateur donné, en utilisant l'approche de filtrage collaboratif et les réreaux de neurones.

Conclusion générale

Dans le cadre de notre projet de stage d'été, nous avons mis en place un système de recommandation. Le présent rapport présente toutes les étapes par lesquelles nous sommes passés pour arriver aux résultats attendus.

Notre point de départ était une récolte des informations nécessaires pour décrire le cadre général du projet qui a permis de comprendre les principaux concepts autour desquels tourne notre projet et de présenter un aperçu de la problématique.

Nous avons enchainé par l'analyse et la spécification des besoins où nous avons détaillé les besoins fonctionnels et non fonctionnels.

La partie suivante était consacrée pour la conception et les choix technologiques.

La dernière partie de notre projet était la partie la plus importante, dans laquelle nous avons traduit notre modélisation conceptuelle en une implémentation.Nous pouvons conclure que les réseaux du neurones et la factorisation matricielle ont joué un rôle important dans les systèmes de recommandation.

Chaque étape de notre projet a nécessité un effort et une recherche approfondie en vue de satisfaire les besoins. Notre projet est donc une source d'enrichissement technique, culturel, personnel et humain.Par ailleurs, ce projet a été pour nous une expérience très bénéfique et enrichissante qui nous a approchés du monde professionnel.

Bibliographie

[1] Z. Chouaib. Mémoire de fin d'études master. Université de 8 Mai 1945 – Guelma -. Juillet 2019. [En ligne]
Disponible sur : http ://dspace.univ-guelma.dz :8080/xmlui/bitstream/handle/123456789/4333/PFE2019 [Accès le 22-Novembre-2020].

[2] I. Ariana. Présentation de l'institut supèrieur d'informatique. ISI Ariana. Jan. 2002. [En ligne]
Disponible sur : http ://www.isi.rnu.tn/ [Accès le 22-Novembre-2020].

[3] E. Negre. Les systèmes de recommandation : une catégorisation. Interstices info. 20/09/2018. [En ligne]
Disponible sur : https ://interstices.info/les-systemes-de-recommandation-categorisation/ [Accès le 10-Aout-2020].

[4] B. L. Réseau de neurones artificiels : qu'est-ce que c'est et à quoi ça sert ? Big Data. 5 avril 2019. [En ligne]
Disponible sur : https ://www.lebigdata.fr/reseau-de-neurones-artificiels-definition [Accès le 16 Aout-2020].

[5] K. Chung. Factorisation matricielle pour les systèmes de recommandatio. github. 21 nov.2019 Dernière mise à jour (24 juil.2019 Première mise en ligne). [En ligne]
Disponible sur : https ://everdark.github.io/k9/notebooks/ml/matrix$_f$actorization/matrix$_f$actorizat
2020].

[6] P. Merieme. Top 10 des bibliothèques python pour la data science. datascientist. 8 janvier 2020. [En ligne]
Disponible sur : https ://le-datascientist.fr/top-10-des-bibliotheques-python-pour-la-data-science [Accès le 28 Aout-2020].

[7] U. article de Wikipédia. Movielens. l'encyclopédie libre. [En ligne]
Disponible sur : https ://fr.qaz.wiki/wiki/MovieLens [Accès le 28 Aout-2020].

[8] movielens. Movielens recommandations de films personnalisées et non commerciales. MovieLens. [En ligne]
Disponible sur : https ://movielens.org/ [Accès le 29 Aout-2020].

[9] shopping.com. Recherchez plus d'un million de produits sur shopping.com. shopping.com. [En ligne]
Disponible sur : https ://shopping.com/ [Accès le 29 Aout-2020].

[10] U. article de Wikipédia. Netflix. l'encyclopédie libre. [En ligne]
Disponible sur : https ://fr.qaz.wiki/wiki/Netflix [Accès le 29 Aout-2020].

[11] netflix.com. Films, séries tv et bien plus en illimité. netflix. [En ligne]
Disponible sur : https ://www.netflix.com/tn-fr/ [Accès le 29 Aout-2020].

Annexes

Annexe1 :l'interface d'accuiel

Figure 4.21: implèmentation de l'interface d'accuiel

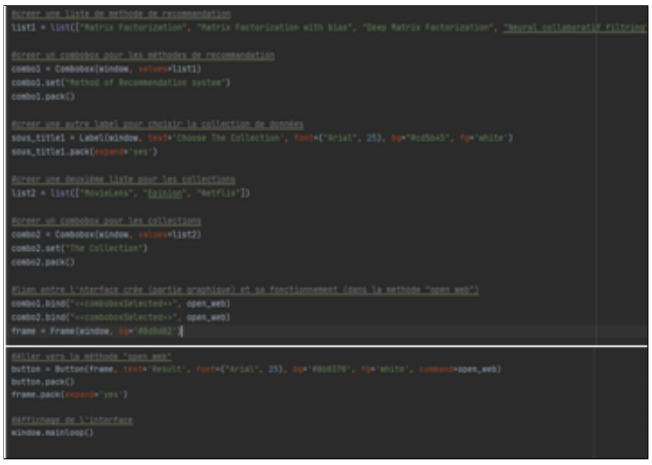
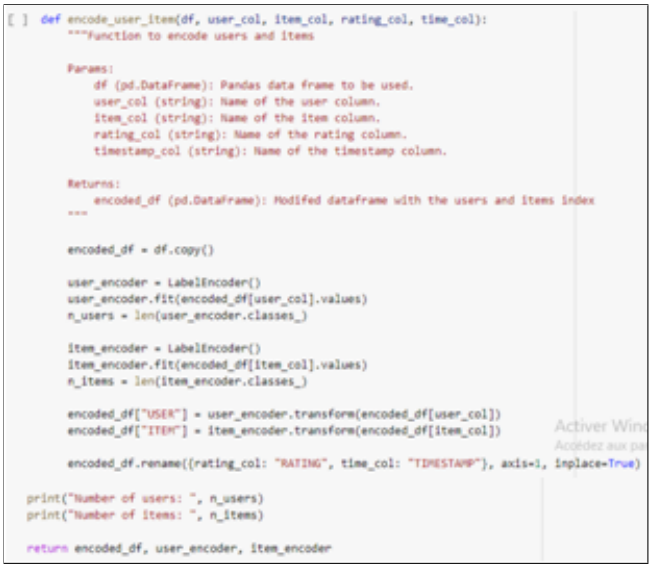


Figure 4.22: implèmentation de l'interface d'accuiel

Annexe2 :nettoyage de données



```
[ ]  def encode_user_item(df, user_col, item_col, rating_col, time_col):
         """Function to encode users and items

         Params:
             df (pd.DataFrame): Pandas data frame to be used.
             user_col (string): Name of the user column.
             item_col (string): Name of the item column.
             rating_col (string): Name of the rating column.
             timestamp_col (string): Name of the timestamp column.

         Returns:
             encoded_df (pd.DataFrame): Modifed dataframe with the users and items index
         """

         encoded_df = df.copy()

         user_encoder = LabelEncoder()
         user_encoder.fit(encoded_df[user_col].values)
         n_users = len(user_encoder.classes_)

         item_encoder = LabelEncoder()
         item_encoder.fit(encoded_df[item_col].values)
         n_items = len(item_encoder.classes_)

         encoded_df["USER"] = user_encoder.transform(encoded_df[user_col])
         encoded_df["ITEM"] = item_encoder.transform(encoded_df[item_col])

         encoded_df.rename({rating_col: "RATING", time_col: "TIMESTAMP"}, axis=1, inplace=True)

     print("Number of users: ", n_users)
     print("Number of items: ", n_items)

     return encoded_df, user_encoder, item_encoder
```

Figure 4.23: implèmentation de l'interface d'accuiel

ملخّص

يتكون مشروع التدريب الصيفي الخاص بنا الذي يتم تنفيذه في المعهد العالي لعلوم الكمبيوتر من تنفيذ نظام التوصية والتحقق من صحته. للوصول إلى نهاية هذا العمل ، اعتمدنا على الشبكات العصبية وعوامل المصفوفة. استخدمنا بايثون كلغة برمجة.

كلمات مفاتيح : الشبكات العصبية ، عامل المصفوفة ، صيتهن ، لـ رلَب

Résumé

Notre projet de stage d'été réalisée au sein de l'institut supérieur d'informatique d'ariana consiste à implémenter et valider un système de recommandation. Pour venir à bout de ce travail, nous avons basé sur les réseaux de neurones et la factorisation matricielle et nous avons utilisé Python comme une langage de programmation.

Mots clés : : Réseaux de neurones, factorisation matricielle, Python, Google Colab.

Abstract

Our work was carried out in the context of summer project at the higher institute of informatics of ariana consists of implementing and validating a recommendation system. To achieve this, we based on neural networks and matrix factorization and we used Python as a programming language.

Keywords : Neural networks, matrix factorization, Python, Google Colab.

Vous pouvez ajouter l'intitulé de l'entreprise ici.

2 نهج أبو الريحان الباروني 2080 أريانة الهاتف: 164 706 71 الفاكس: 698 706 71 البريد الالكتروني: isi@isim.rnu.tn

2 Abou Raihane Bayrouni 2080 l'Ariana **Tél**: 71 706 164 **Fax**: 71 706 698 **Email**: isi@isim.rnu.tn