Université Mohammed-V de Rabat
École Nationale Supérieure d'Informatique
et d'Analyse des Systèmes
ENSIAS

Projet Data Driven Decision Making

Filière : Génie Logiciel

Conception et Évaluation d'un Système de Recommandation Intelligent pour la Personnalisation des Suggestions Produits

Élèves ingénieurs :

Mtejjal Aya

H'mida Asma

Encadrant :

Pr.Youness TABII

Année Académique 2024/2025

Table de matières

Chapitre 1

Introduction Générale et Objectif du Projet

1.1 Introduction

Dans un environnement numérique où la quantité de produits disponibles ne cesse de croître, les utilisateurs sont confrontés à un choix de plus en plus vaste, rendant difficile l'identification des articles correspondant réellement à leurs préférences. Les systèmes de recommandation sont devenus des outils essentiels pour résoudre ce problème, en proposant aux utilisateurs des produits susceptibles de les intéresser, basés sur leurs comportements antérieurs et leurs préférences explicites ou implicites.

1.2 Objectif du projet

L'objectif de ce projet est de concevoir un système de recommandation performant en combinant plusieurs approches complémentaires. Il s'appuie sur le filtrage collaboratif, qui utilise des algorithmes comme SVD et KNNBasic pour recommander des produits en fonction des préférences similaires entre utilisateurs, et sur le filtrage basé sur le contenu, exploitant les caractéristiques des produits via des techniques telles que TF-IDF pour la vectorisation des descriptions et les embeddings BERT pour capturer les relations sémantiques avancées.

Chapitre 2

Description du Dataset et Prétraitement des Données

2.1 Jeu de Données

Les données exploitées dans ce projet proviennent de Kaggle et contiennent des informations sur les produits ainsi que sur les interactions des utilisateurs. Le jeu de données comporte 5000 lignes, et les données utilisées sont les suivantes :

- **Uniq Id** : Identifiant unique d'utilisateur.
- **Product Id** : Identifiant unique du produit.
- **Product Name** : Nom du produit.
- **Product Rating** : Note attribuée au produit par les utilisateurs (échelle de 1 à 5).
- **Clicked** : Indique si l'utilisateur a cliqué sur le produit (1 = oui, 0 = non).
- **View_Time_Sec** : Temps passé à consulter le produit (en secondes).
- **Product Description** : Description détaillée du produit.
- **Product Tags** : Mots-clés associés au produit.
- **Product Category** : Catégorie du produit.
- **Product Brand** : Marque du produit.
- **Product Reviews Count** : Nombre d'avis sur le produit.

Un aperçu des premières lignes du dataset

Uniq Id	Product Id	Product Name	Product Rating	Clicked	View_Time_Sec	Product Description	Product Tags	Product Category	Product Brand	Product Reviews Count
792d82aa2f2d3caf1c07c53f4	2e17bf4acecdece67fc00f07ad62c910	OPI Infinite Shine, Nail Lacquer Nail Polish, ...	NaN	0	0	NaN	OPI Infinite Shine, Nail Lacquer Nail Polish, ...	Premium Beauty > Premium Makeup > Premium Nail...	OPI	NaN
:6f4810fcfc7ff244fd06784f11	076e5854a62dd283c253d6bae415af1f	Nice n Easy Permanent Color, 111 Natural Mediu...	NaN	1	147	Pack of 3 Pack of 3 for the UPC: 381519000201 ...	Nice 'n Easy Permanent Color, 111 Natural Medi...	Beauty > Hair Care > Hair Color > Auburn Hair ...	Nice'n Easy	NaN
78d3ed181b15a4102b287f2	8a4fe5d9c7a6ed26cc44d785a454b124	Clairol Nice N Easy Permanent Color 7/106A Nat...	4.5	0	0	This Clairol Nice N Easy Permanent Color gives...	Clairol Nice 'N Easy Permanent Color 7/106A Na...	Beauty > Hair Care > Hair Color > Permanent Ha...	Clairol	29221.0

FIGURE 2.1 – Aperçu des premières lignes du jeu de données des produits.

2.2 Environnements Utilisés

2.2.1 Environnement

Au début, on a tenté d'implémenter les deux techniques de système de recommandation (le filtrage basé sur le contenu et le filtrage collaboratif) dans un même notebook sous Jupyter. Cependant, nous avons rencontré des problèmes de dépendances. Nous avons donc décidé de créer des environnements virtuels distincts dans Anaconda Navigator, à l'aide de Conda, afin de séparer les notebooks et résoudre les conflits de dépendances.

FIGURE 2.2 – Outil en ligne de commande intégré à Anaconda Navigator

2.2.2 Bibliothèques principales :

— **NumPy** et **Pandas** : Pour le traitement et l'analyse des données.

— **Matplotlib** et **Seaborn** : Pour la visualisation des données et des résultats des modèles.

— **Pickle** : Pour la sauvegarde et le chargement des modèles entraînés et des datasets nettoyés (sérialisation/désérialisation).

— **Surprise** : Bibliothèque dédiée aux systèmes de recommandation, utilisée pour implémenter des algorithmes tels que *SVD* (Singular Value Decomposition) et

KNN (K-Nearest Neighbors).

— **surprise.model_selection** : Pour la validation croisée des modèles et la division des données.

— **surprise.accuracy** : Pour le calcul des métriques d'évaluation telles que *RMSE* et *MAE*, qui mesurent la précision des prédictions.

2.3 Vérification des Données Brutes

2.3.1 Dimensions et Structure :

Le dataset utilisé dans ce projet de système de recommandation contient **5000** interactions utilisateur-produit, avec **34** colonnes initiales, réduites à **11** colonnes pertinentes sélectionnées en fonction de leur importance pour le système de recommandation.

2.3.2 Cohérence des Types de Données :

-Les identifiants utilisateur et produit (Uniq Id, Product Id) ont été convertis en chaînes de caractères, afin d'assurer la compatibilité avec les bibliothèques de recommandation comme *Surprise*, qui requièrent ce format.
-On a converti la colonne `Product Rating` en format numérique, en remplaçant les entrées non valides par `NaN`, ce qui est essentiel pour permettre les calculs statistiques.

2.3.3 Analyse des valeurs manquantes

Une stratégie rigoureuse de traitement des valeurs manquantes a été appliquée afin d'assurer la qualité des recommandations. L'analyse initiale a révélé des taux élevés de valeurs manquantes, notamment pour `Product Rating` (56,12 %) et `Product Reviews Count` (33,08 %).

Les notes ont été imputées par la moyenne des produits de la même catégorie, ou par la moyenne globale en l'absence de référence. Les descriptions manquantes ont été remplacées par des chaînes vides, tandis que les marques et catégories absentes ont été

étiquetées `"Unknown"`. Les valeurs manquantes du nombre d'avis ont été remplacées par la moyenne globale après conversion numérique.

2.3.4 Normalisation du temps de visionnage

Le temps de visionnage est une valeur brute très variable (allant de 10 à 500 secondes). S'il est utilisé tel quel, il risque de dominer les autres variables dans les calculs, notamment lors de la construction du score hybride combinant `Product Rating`, `Clicked` et `View_Time_Sec`, ce qui fausserait la représentation du comportement réel des utilisateurs.

2.3.5 Statistiques descriptives

Statistiques descriptives après nettoyage:

	Product Rating	Clicked	View_Time_Sec	Product Reviews Count	Norm_View_Time	Rating
count	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000
mean	4.311178	0.694800	64.052600	571.035565	0.355848	3.553637
std	0.587811	0.460538	60.194755	2033.317010	0.334415	1.003667
min	1.000000	0.000000	0.000000	1.000000	0.000000	0.500000
25%	4.106667	0.000000	0.000000	7.000000	0.000000	2.354545
50%	4.400000	1.000000	52.000000	167.000000	0.288889	3.957435
75%	4.600000	1.000000	118.000000	571.035565	0.655556	4.344444
max	5.000000	1.000000	180.000000	29242.000000	1.000000	5.000000

FIGURE 2.3 – Résumé des statistiques descriptives du jeu de données

La note hybride (`Rating`) offre une meilleure répartition des scores (moyenne : **3,55**, écart-type : **1,00**), améliorant ainsi la capacité de discrimination du système. Le taux de clics élevé (**69,5 %**) témoigne d'un bon engagement utilisateur, tandis que la note moyenne des produits (**4,31**/5) traduit une tendance générale à l'évaluation positive.

Chapitre 3

Théorie des Algorithmes Utilisés

3.1 Introduction

Ce projet repose sur deux techniques de recommandation : le **filtrage collaboratif**, qui prédit les préférences d'un utilisateur à partir des interactions d'utilisateurs similaires, et le **filtrage basé sur le contenu**, qui se base sur les caractéristiques des produits.

3.2 SVD (Décomposition en valeurs singulières)

Le modèle **SVD (Décomposition en Valeurs Singulières)** est une méthode de factorisation matricielle utilisée dans les systèmes de recommandation collaboratifs. Il consiste à décomposer la matrice utilisateur-produit R en trois matrices plus simples :

$$R \approx U \cdot \Sigma \cdot V^T$$

où :

— U : matrice représentant les utilisateurs dans un espace latent,

— Σ : matrice diagonale contenant les valeurs singulières (pondérations des facteurs latents),

— V^T : transposée d'une matrice représentant les produits dans le même espace.

Le modèle **SVD** représente chaque utilisateur et chaque produit par un vecteur de fac-

teurs latents, permettant de prédire une note via le produit scalaire entre ces vecteurs. À partir d'une matrice utilisateur-produit partiellement remplie, il complète les valeurs manquantes en exploitant les similarités dans un espace latent de faible dimension.
Dans notre projet, **SVD** a été implémenté à l'aide de la bibliothèque `Surprise`, et entraîné sur la matrice contenant les interactions explicites (notes) entre utilisateurs et produits.

3.3 KNNBasic (K-Nearest Neighbors Basic)

Le modèle `KNNBasic`, issu de la bibliothèque `Surprise`, applique l'algorithme des *k plus proches voisins* au filtrage collaboratif. En mode *(basé sur les utilisateurs)* , il recommande des produits à un utilisateur en se basant sur les évaluations d'utilisateurs similaires, identifiés à l'aide de la similarité cosinus. Celle-ci est calculée selon la formule :

$$\text{similarité}(u, v) = \frac{\sum(u_i \times v_i)}{\sqrt{\sum u_i^2} \times \sqrt{\sum v_i^2}}$$

où u_i et v_i sont les notes données par les utilisateurs u et v aux mêmes produits. La note prédite est ensuite une moyenne pondérée des notes des voisins, pondérée par leur degré de similarité.

3.4 TF-IDF (Fréquence du Terme – Fréquence Inverse du Document) + KNN (k plus proches voisins)

Dans le cadre du filtrage basé sur le contenu, nous avons utilisé l'approche **TF-IDF** combinée à **KNN** pour analyser les caractéristiques textuelles des produits et générer des recommandations basées sur leurs similarités.

Vectorisation TF-IDF : Les descriptions des produits sont converties en vecteurs numériques, où chaque terme est pondéré pour refléter son importance spécifique.

Formule :

$$\text{TF-IDF}(t, d) = \text{TF}(t, d) \times \text{IDF}(t)$$

où :

— **TF(t, d)** mesure la fréquence d'un terme t dans un document d,

— **IDF(t)** réduit l'impact des termes communs dans l'ensemble des documents.

Application de KNN : L'algorithme des *k plus proches voisins* mesure la similarité entre les vecteurs TF-IDF (via la similarité cosinus) afin d'identifier les produits les plus proches en contenu.

3.5 BERT + KNN (k plus proches voisins)

Pour la technique de filtrage basée sur le contenu, nous avons utilisé le modèle préentraîné **paraphrase-MiniLM-L6-v2**, fourni par la bibliothèque `sentence-transformers`. Ce modèle, dérivé de BERT et spécialement optimisé pour des tâches de similarité et de paraphrase, transforme les descriptions textuelles des produits en vecteurs numériques appelés *embeddings*. Ces embeddings, générés par l'encodeur du modèle, capturent le sens global des descriptions tout en tenant compte du contexte sémantique.

Chaque produit est ainsi représenté dans un *espace vectoriel dense*, où la proximité entre deux vecteurs reflète leur similarité sémantique. Pour identifier les produits les plus similaires à un produit donné, nous avons utilisé l'algorithme **KNN (k plus proches voisins)**, avec la mesure de similarité cosinus appliquée aux vecteurs. Cette approche permet d'exploiter efficacement les représentations sémantiques pour proposer des recommandations basées sur le contenu.

Chapitre 4

Résultats des Algorithmes et Comparaison de leurs Performances

4.1 Résultats de recommandation des produits :

4.1.1 Recommandation de produits en se basant sur le filtrage collaboratif par SVD et KNNBasic

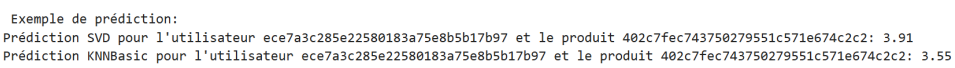

FIGURE 4.1

Un utilisateur et un produit ont été sélectionnés de manière aléatoire afin d'illustrer le fonctionnement du filtrage collaboratif. Les prédictions obtenues sont les suivantes :

— **SVD** : 3.91 / 5

— **KNNBasic** : 3.55 / 5

Ces résultats, relativement élevés, montrent que les deux modèles estiment que le produit correspond aux préférences de l'utilisateur. Cela illustre la capacité du système à générer des recommandations personnalisées.

4.1.2 Recommandation de produits en se basant sur le filtrage basé sur le contenu avec TF-IDF et KNN

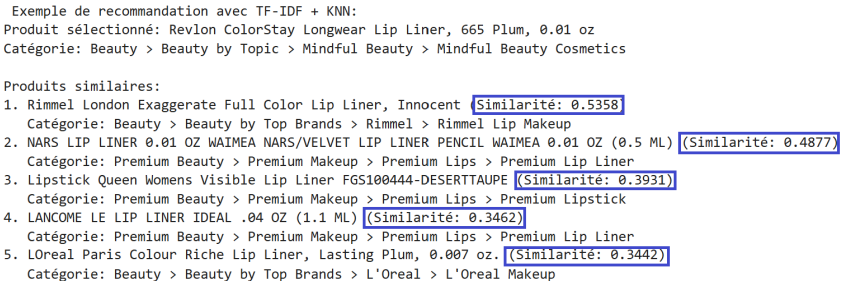

FIGURE 4.2

Un exemple de recommandation basée sur le contenu (TF-IDF + KNN) a été réalisé à partir du produit *Revlon ColorStay Lip Liner*. Les produits suggérés sont tous issus de la même catégorie fonctionnelle (crayons ou rouges à lèvres), avec des scores de similarité compris entre 0.34 et 0.53. Ces résultats confirment que le modèle est capable d'identifier des produits sémantiquement proches à partir de leur description, même si les marques et sous-catégories diffèrent.

4.1.3 Recommandation de produits en se basant sur le filtrage basé sur le contenu avec BERT et KNN

Un exemple de recommandation avec l'approche **BERT + KNN** montre que le modèle identifie efficacement des produits capillaires sémantiquement proches, malgré des formulations différentes. Les similarités élevées entre les produits suggérés confirment la capacité de BERT à capturer le sens global des descriptions pour proposer des recommandations cohérentes.

FIGURE 4.3

4.2 Comparaison entre les modèles

Validation croisée des modèles collaboratifs

Une validation croisée à 5 plis a été utilisée pour évaluer la robustesse des modèles. À chaque itération, le modèle est entraîné sur 80 % des données et testé sur les 20 % restantes. La moyenne des cinq itérations fournit la performance finale.

Cette méthodologie a été appliquée aux deux modèles suivants :

— **SVD**

— **KNNBasic**

Métrique de performance : RMSE

Après l'entraînement, la performance des modèles a été évaluée à l'aide du **RMSE** (Root Mean Square Error), une métrique couramment utilisée dans les systèmes de recommandation. Elle mesure l'écart moyen entre les notes réelles et les notes prédites, en pénalisant davantage les grandes erreurs.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_{\text{réel},i} - y_{\text{prédit},i})^2}$$

Métrique de performance : MAE

En complément du RMSE, nous avons également évalué les performances de nos modèles de recommandation à l'aide de l'**erreur absolue moyenne (Mean Absolute Error, MAE)**. Cette métrique est couramment utilisée dans les systèmes de recommandation, car elle mesure l'écart absolu moyen entre les notes réelles attribuées par les utilisateurs ($y_{\text{réel}}$) et les notes prédites par les modèles ($y_{\text{prédit}}$).

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_{\text{réel},i} - y_{\text{prédit},i}|$$

Comparaison des modèles : Tableau récapitulatif

```
 Comparaison des modèles de filtrage collaboratif:
     Modèle      RMSE       MAE
0       SVD  1.003136  0.884288
1  KNNBasic  1.003499  0.885256
```

FIGURE 4.4

Les deux modèles de filtrage collaboratif présentent des performances très proches. Le modèle **SVD** obtient un **RMSE de 1,0031** et un **MAE de 0,8843**, légèrement inférieurs à ceux du modèle **KNNBasic** (**RMSE : 1,0035**, **MAE : 0,8853**). Ces résultats indiquent que **SVD offre une précision légèrement meilleure**.

Précision@k

mesure la proportion d'éléments pertinents parmi les k premiers éléments recommandés. Elle reflète la capacité du système à proposer des résultats corrects.

$$\text{Précision@k} = \frac{|\text{Recommandés} \cap \text{Pertinents}|}{k}$$

Rappel@k (Recall@k)

mesure la proportion d'éléments pertinents retrouvés parmi les k recommandations, par rapport au nombre total d'éléments pertinents.

$$\text{Rappel@k} = \frac{|\text{Recommandés} \cap \text{Pertinents}|}{|\text{Pertinents}|}$$

F1-score

est la moyenne harmonique de la précision et du rappel, offrant un équilibre entre ces deux métriques. Il est calculé selon la formule suivante :

$$F1 = 2 \times \frac{\text{Précision} \times \text{Rappel}}{\text{Précision} + \text{Rappel}}$$

Nouveauté

mesure la capacité du système à recommander des éléments peu populaires, donc moins susceptibles d'être découverts spontanément. Elle est calculée comme la moyenne de l'inverse logarithmique de la popularité des produits recommandés :

$$\text{Nouveauté} = \frac{1}{k} \sum_{i=1}^{k} -\log_2(\text{popularité}(i))$$

où la popularité d'un produit est définie comme le ratio entre le nombre d'interactions avec ce produit et le nombre total d'interactions dans le jeu de données.

Comparaison des modèles : Tableau récapitulatif

Évaluation avancée des modèles de filtrage basé sur le contenu:

	Modèle	Précision@5	Rappel@5	F1@5	Nouveauté
0	TF-IDF + KNN	0.373034	0.131241	0.139653	12.262092
1	BERT + KNN	0.411236	0.141233	0.156299	12.268420

FIGURE 4.5

Les résultats montrent que le modèle **BERT + KNN** surpasse systématiquement **TF-IDF + KNN** sur l'ensemble des métriques. Il atteint une **Précision@5 de 41,1 %** et un **Rappel@5 de 14,1 %**, contre respectivement **37,3 %** et **13,1 %** pour le modèle TF-IDF. Le **F1-score@5**, qui reflète le compromis entre précision et rappel, est également supérieur avec BERT (**0,156** contre **0,140**). Enfin, la **valeur de nouveauté**, légèrement plus élevée, indique que BERT recommande des produits un peu moins populaires, favorisant ainsi la diversité et la découverte de nouveaux articles.

Conclusion :

Ce projet a développé un système de recommandation complet qui combine efficacement plusieurs approches (filtrage collaboratif avec SVD et KNN, filtrage basé sur le contenu avec TF-IDF et BERT) pour suggérer des produits pertinents aux utilisateurs.