INF6083 - Systèmes de recommandation

Projet P1 : Manipulation de données et concepts fondamentaux

Hiver 2026

Enseignant	Courriel	Local
Etienne Tajeuna	etiennegael.tajeuna@uqo.ca	A2222

Département d'Informatique et d'Ingénierie (DII)

30 janvier 2026

Sommaire

Dans le cadre de ce projet (P1), les étudiants travailleront avec un jeu de données réel de grande envergure provenant d'Amazon Reviews 2023. Ce projet vise à explorer les concepts fondamentaux nécessaires à la conception de systèmes de recommandation à l'échelle, incluant les mesures de similarité, la représentation sous forme de graphe, les techniques de regroupement, ainsi que les méthodes de prédiction et d'évaluation hors ligne. L'objectif est de permettre aux étudiants de maîtriser les techniques de base tout en relevant les défis liés à la volumétrie des données.

Table des matières

1 Contexte et jeu de données

1.1 Contexte

Les systèmes de recommandation modernes doivent traiter des volumes massifs de données utilisateur-item. Des plateformes comme Amazon, Netflix ou Spotify gèrent des millions d'utilisateurs et des millions de produits, générant des milliards d'interactions. Cette volumétrie pose des défis importants en termes de traitement, de stockage et d'optimisation algorithmique.

La conception d'un système de recommandation efficace à grande échelle nécessite non seulement une compréhension approfondie des algorithmes, mais également une maîtrise des techniques de manipulation de données volumineuses, d'échantillonnage intelligent et d'optimisation computationnelle. Ce projet vise à explorer ces concepts à travers une série de tâches pratiques sur un jeu de données réel de haute volumétrie.

1.2 Jeu de données : Amazon Reviews 2023

Vous travaillerez avec le jeu de données **Amazon Reviews 2023**, collecté par le McAuley Lab de l'Université de Californie à San Diego. Ce dataset est l'un des plus grands corpus publics d'évaluations de produits.

Catégorie recommandée : `Books` (Livres)

Source : `https://amazon-reviews-2023.github.io/`

Lien direct :

— **Reviews** : `https://mcauleylab.ucsd.edu/public_datasets/data/amazon_2023/raw/review_categories/Books.jsonl.gz`
— **Metadata** : `https://mcauleylab.ucsd.edu/public_datasets/data/amazon_2023/raw/meta_categories/meta_Books.jsonl.gz`

1.2.1 Caractéristiques du jeu de données Books

Métrique	**Valeur**
Nombre d'utilisateurs	10.3M
Nombre de livres	4.4M
Nombre d'évaluations	29.5M
Période temporelle	Mai 1996 - Sep 2023
Tokens de reviews	2.9 milliards
Tokens de metadata	3.7 milliards
Échelle de notation	1.0 à 5.0

Table 1 – Statistiques du dataset Amazon Books 2023

1.2.2 Structure des données

Fichier de reviews (format JSONL compressé) :

— `user_id` : Identifiant unique de l'utilisateur
— `parent_asin` : Identifiant unique du livre (clé principale)
— `rating` : Évaluation (1.0 à 5.0)
— `timestamp` : Horodatage Unix (secondes)
— `title` : Titre de la review
— `text` : Texte de la review
— `helpful_vote` : Votes d'utilité
— `verified_purchase` : Achat vérifié (bool)

Fichier de metadata (optionnel pour ce projet) :

— `parent_asin` : Identifiant du livre
— `title` : Titre du livre
— `average_rating` : Note moyenne affichée
— `price` : Prix en USD
— `categories` : Catégories hiérarchiques
— `description` : Description du livre

Important : Utilisez `parent_asin` comme identifiant de livre (et non `asin`).

1.3 Défis de volumétrie

Ce projet vous confrontera aux défis suivants :

1. **Chargement efficace** : Lecture de fichiers JSONL compressés de plusieurs GB
2. **Échantillonnage** : Sélection de sous-ensembles représentatifs
3. **Mémoire** : Utilisation de structures de données optimales (sparse matrices)
4. **Temps de calcul** : Optimisation des algorithmes pour traiter des millions d'interactions
5. **Stockage** : Sauvegarde efficace des résultats intermédiaires

2 Objectifs

Ce projet vise à :

1. Comprendre et implémenter différentes mesures de similarité à grande échelle
2. Représenter et analyser les données sous forme de graphe
3. Appliquer des techniques de regroupement pour identifier des profils d'utilisateurs
4. Implémenter et évaluer des méthodes simples de prédiction
5. Développer des compétences en optimisation algorithmique et gestion de la mémoire
6. Développer une capacité d'analyse critique des résultats obtenus

3 Tâches du projet

3.1 Tâche 0 : Chargement et échantillonnage des données

3.1.1 Échantillonnage stratégique

Étant donné la taille du dataset (29.5M reviews), vous devez créer un échantillon gérable. Implémentez les stratégies suivantes :

1. **Échantillonnage par utilisateur actif** :
   — Identifiez les utilisateurs ayant au moins 20 évaluations
   — Sélectionnez aléatoirement 50,000 utilisateurs parmi eux
   — Conservez toutes leurs évaluations
2. **Échantillonnage temporel** (alternative ou complément) :
   — Sélectionnez les reviews d'une période spécifique (ex : 2020-2023)
   — Ou échantillonnez uniformément dans le temps
3. **Justification** : Documentez votre stratégie d'échantillonnage et justifiez vos choix en termes de :
   — Représentativité
   — Volumétrie cible (viser 500K - 2M reviews)
   — Préservation de la structure des données

3.1.2 Analyse exploratoire

Sur votre échantillon, calculez les statistiques descriptives suivantes :

1. **Statistiques de base** :
   — Nombre total d'utilisateurs, de livres et d'évaluations
   — Distribution des évaluations (histogramme)
   — Nombre moyen d'évaluations par utilisateur et par livre
   — Identification des 10 utilisateurs les plus actifs et des 10 livres les plus populaires
2. **Taux de sparsité** ρ de la matrice utilisateur-item :

$$\rho = 1 - \frac{|R|}{|U| \times |I|} \tag{1}$$

   où $|R|$ est le nombre d'évaluations, $|U|$ le nombre d'utilisateurs et $|I|$ le nombre de livres.
3. **Analyse de la distribution** :
   — Distribution de la popularité des livres (phénomène de « longue traîne »)
   — Distribution temporelle des évaluations
   — Distribution des votes d'utilité (`helpful_vote`)
   — Proportion d'achats vérifiés (`verified_purchase`)
4. **Visualisations** : Créez au moins 4 graphiques pertinents pour comprendre les données.

3.1.3 Prétraitement

1. **Nettoyage** :
   — Supprimez les reviews sans rating
   — Gérez les timestamps invalides
   — Convertissez les ratings en float
2. **Filtrage** : Appliquez les seuils suivants (ou justifiez vos propres choix) :
   — Nombre minimum d'évaluations par utilisateur : 10-20
   — Nombre minimum d'évaluations par livre : 5-10
   Recalculez le taux de sparsité après filtrage.
3. **Matrice utilisateur-item** : Construisez la matrice $\mathbf{R} \in R^{|U|\times|I|}$ où $r_{u,i}$ représente l'évaluation de l'utilisateur u pour le livre i.
   Impératif : Utilisez une représentation `scipy.sparse.csr_matrix` pour optimiser la mémoire.
4. **Division train/test** : Divisez les données en :
   — Ensemble d'entraînement : 80% des évaluations
   — Ensemble de test : 20% des évaluations
   Stratifiez par utilisateur : chaque utilisateur doit avoir au moins une évaluation dans chaque ensemble.

3.2 Tâche 1 : Mesures de similarité

3.2.1 Implémentation des mesures

Implémentez les trois mesures de similarité suivantes pour calculer la similarité entre utilisateurs :

1. **Similarité cosinus** :

$$\text{sim}_{\cos}(u,v) = \frac{\mathbf{r}_u \cdot \mathbf{r}_v}{\|\mathbf{r}_u\|\|\mathbf{r}_v\|} = \frac{\sum_{i\in I_{u,v}} r_{u,i}\cdot r_{v,i}}{\sqrt{\sum_{i\in I_u} r_{u,i}^2}\sqrt{\sum_{i\in I_v} r_{v,i}^2}} \tag{2}$$

   où $I_{u,v}$ est l'ensemble des livres évalués à la fois par u et v.
2. **Corrélation de Pearson** :

$$\text{sim}_{\text{pear}}(u,v) = \frac{\sum_{i\in I_{u,v}}(r_{u,i}-\bar{r}_u)(r_{v,i}-\bar{r}_v)}{\sqrt{\sum_{i\in I_{u,v}}(r_{u,i}-\bar{r}_u)^2}\sqrt{\sum_{i\in I_{u,v}}(r_{v,i}-\bar{r}_v)^2}} \tag{3}$$

   où $\bar{r}_u$ est l'évaluation moyenne de l'utilisateur u.
3. **Similarité de Jaccard** :

$$\text{sim}_{\text{jac}}(u,v) = \frac{|I_u \cap I_v|}{|I_u \cup I_v|} \tag{4}$$

   où I_u et I_v sont les ensembles de livres évalués par u et v respectivement.

Optimisation : Pour un dataset de cette taille, ne calculez pas la matrice de similarité complète. Utilisez plutôt :

— `sklearn.metrics.pairwise.cosine_similarity` avec des matrices sparse
— Calcul par batch pour la corrélation de Pearson
— Stockage sparse des similarités (> seuil minimal)

3.2.2 Analyse comparative

1. **Échantillon d'utilisateurs** : Sélectionnez 5 utilisateurs avec différents profils :
   — 1 utilisateur très actif (> 100 reviews)
   — 2 utilisateurs moyennement actifs (30-50 reviews)
   — 2 utilisateurs peu actifs (10-20 reviews)
2. **Identification des voisins** : Pour chaque utilisateur, identifiez leurs 10 plus proches voisins selon chaque mesure.
3. **Comparaison** :
   — Créez un tableau comparant les voisins identifiés par chaque mesure
   — Calculez le coefficient de Jaccard entre les ensembles de voisins
   — Analysez les différences : pourquoi certains voisins diffèrent ?
4. **Distribution des similarités** :
   — Pour chaque mesure, visualisez la distribution des similarités (histogramme)
   — Calculez moyenne, médiane, écart-type
   — Identifiez les valeurs aberrantes
5. **Visualisation** : Créez une heatmap des similarités pour 30 utilisateurs sélectionnés aléatoirement.
6. **Discussion** : Analysez les forces et faiblesses de chaque mesure :
   — Impact de la sparsité
   — Sensibilité aux biais utilisateurs
   — Temps de calcul
   — Pertinence pour les recommandations de livres

3.3 Tâche 2 : Représentation en graphe

3.3.1 Construction du graphe biparti

Construisez un graphe biparti $G = (U \cup I, E)$ où :

— U est l'ensemble des nœuds utilisateurs
— I est l'ensemble des nœuds livres
— E est l'ensemble des arêtes (u, i) avec poids $w_{u,i} = r_{u,i}$

1. **Graphe complet** : Créez le graphe représentant toutes les interactions de votre échantillon.
   Note : Pour un échantillon de 1M+ reviews, le graphe sera très grand. Utilisez `networkx.Graph()` avec stockage efficace.
2. **Sous-graphe réduit** : Pour la visualisation, créez un sous-graphe contenant :
   — Les 30 utilisateurs les plus actifs
   — Les 50 livres les plus populaires
   — Les arêtes correspondantes

3. **Visualisation** : Visualisez ce sous-graphe avec :
   — Une disposition bipartite claire
   — Taille des nœuds proportionnelle à leur degré
   — Couleurs différentes pour utilisateurs et livres

3.3.2 Analyse du graphe

Calculez les métriques suivantes :

1. **Métriques globales** (sur le graphe complet) :
   — Nombre de nœuds : $|V| = |U| + |I|$
   — Nombre d'arêtes : $|E|$
   — Degré moyen : $\bar{d} = \frac{2|E|}{|V|}$
   — Densité : $\delta = \frac{|E|}{|U| \times |I|}$
   — Distribution des degrés (échelle log-log pour identifier une loi de puissance)
2. **Centralité des livres** (sur le sous-graphe) :

$$C_d(i) = \frac{\deg(i)}{|U|} \tag{5}$$

   — Identifiez les 20 livres avec la centralité la plus élevée
   — Comparez avec les livres les plus populaires (nombre de reviews)
   — Y a-t-il des différences ? Pourquoi ?
3. **Coefficient de clustering** :
   — Calculez le coefficient de clustering moyen du sous-graphe
   — Interprétez : que signifie-t-il dans le contexte des recommandations ?
4. **Composantes connexes** :
   — Identifiez le nombre de composantes connexes
   — Quelle est la taille de la plus grande composante ?
5. **Analyse et interprétation** :
   — Que révèlent ces métriques sur le comportement des lecteurs ?
   — Comment ces informations pourraient-elles améliorer les recommandations ?
   — Comparez avec les caractéristiques attendues d'un réseau social

3.4 Tâche 3 : Regroupement des utilisateurs

3.4.1 Préparation et détermination du nombre de clusters

1. **Préparation des données** :
   — Sélectionnez un échantillon de 10,000 utilisateurs aléatoires
   — Normalisez les vecteurs utilisateur avec `StandardScaler`
   — Justifiez le choix de la taille d'échantillon
2. **K-Means pour différents K** : Appliquez K-Means pour $K \in \{3, 4, 5, 6, 7, 8\}$. Pour chaque K, calculez le **score de Silhouette** (`https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html`) et affichez le diagramme en barre.
3. Faites le choix du meilleur K et justifiez votre choix suivant le contexte en étude.

3.4.2 Analyse des clusters

1. **Profils des clusters** : Pour chaque cluster C_k identifié, calculez :
   — Taille du cluster : $|C_k|$
   — Centre du cluster : $\boldsymbol{\mu}_k = \frac{1}{|C_k|} \sum_{u \in C_k} \mathbf{r}_u$
   — Évaluation moyenne des utilisateurs du cluster
   — Écart-type des évaluations
   — Les 10 livres préférés du cluster (moyennes les plus élevées)
   — Les genres de livres dominants (si metadata disponible)
2. **Caractérisation qualitative** : Pour chaque cluster, proposez une interprétation qualitative :
   — Exemples : « Lecteurs de fiction contemporaine », « Critiques sévères », « Amateurs de science-fiction », « Lecteurs occasionnels », etc.
   — Basez-vous sur les livres préférés, les moyennes d'évaluation, les patterns observés
3. **Visualisation 2D** :
   — Utilisez PCA pour réduire à 2 dimensions
   — Ou utilisez t-SNE (plus long mais potentiellement plus révélateur)
   — Créez un scatter plot avec couleurs par cluster
   — Ajoutez les centres des clusters

3.5 Tâche 4 : Prédiction des évaluations

3.5.1 Modèles de référence (Baselines)

Implémentez deux modèles de référence simples :

1. **Moyenne globale** :
$$\hat{r}_{u,i} = \bar{r} \tag{6}$$
où $\bar{r}$ est la moyenne globale de toutes les évaluations de l'ensemble d'entraînement.
2. **Moyenne par livre** :
$$\hat{r}_{u,i} = \bar{r}_i \tag{7}$$
où $\bar{r}_i$ est la moyenne des évaluations du livre i.
Si le livre n'est pas dans l'ensemble d'entraînement, utilisez $\bar{r}$.

Évaluez ces modèles sur l'ensemble de test avec :

— RMSE : $\text{RMSE} = \sqrt{\frac{1}{|T|} \sum_{(u,i) \in T} (r_{u,i} - \hat{r}_{u,i})^2}$
— MAE : $\text{MAE} = \frac{1}{|T|} \sum_{(u,i) \in T} |r_{u,i} - \hat{r}_{u,i}|$

où T est l'ensemble de test.

3.5.2 k-NN collaboratif basé utilisateur

Implémentez l'algorithme de prédiction k-NN :

$$\hat{r}_{u,i} = \bar{r}_u + \frac{\sum_{v \in N_k(u,i)} \text{sim}(u,v) \cdot (r_{v,i} - \bar{r}_v)}{\sum_{v \in N_k(u,i)} |\text{sim}(u,v)|} \tag{8}$$

où $N_k(u,i)$ est l'ensemble des k utilisateurs les plus similaires à u ayant évalué le livre i.

1. **Optimisation des hyperparamètres** :
   Testez les configurations suivantes :
   — Valeurs de k : $\{10, 20, 30, 50, 100\}$
   — Mesures de similarité : cosinus, Pearson, Jaccard
   Conseil : Pour accélérer, testez sur un sous-échantillon du test set (10-20%).
2. **Gestion des cas limites** :
   — Si aucun voisin n'a évalué i : retourner $\bar{r}_u$ ou $\bar{r}_i$
   — Si moins de k voisins ont évalué i : utiliser tous les voisins disponibles
3. **Optimisation computationnelle** :
   — Pré-calculez et stockez les k voisins les plus similaires pour chaque utilisateur
   — Utilisez des structures de données efficaces (dictionnaires, matrices sparse)
   — Mesurez le temps d'exécution
4. **Identification de la meilleure configuration** : Identifiez (k^*, sim^*) qui minimise le RMSE.

3.5.3 Analyse des résultats

1. **Tableau comparatif** : Créez un tableau comparant tous vos modèles :

Modèle	**RMSE**	**MAE**	**Temps (s)**
Baseline - Moyenne globale	...	...	...
Baseline - Moyenne par livre	...	...	...
k-NN (k=10, cosinus)	...	...	...
k-NN (k=20, Pearson)	...	...	...
k-NN (meilleure config)	...	...	...

2. **Analyse de performance** :
   — Le k-NN améliore-t-il significativement les baselines ?
   — Quel est l'impact de k sur les performances ? Tracez RMSE vs k.
   — Quelle mesure de similarité performe le mieux ? Pourquoi ?
   — Y a-t-il des différences selon le type d'utilisateur (actif vs occasionnel) ?
3. **Compromis précision/temps** :
   — Comparez le temps de calcul des différentes approches
   — Est-ce que la meilleure précision vaut le coût computationnel ?
   — Proposez des stratégies pour accélérer le k-NN
4. **Analyse d'erreurs** :
   — Pour quels types de livres les erreurs sont-elles les plus grandes ?
   — Les livres populaires sont-ils mieux prédits que les livres de niche ?
   — Analysez quelques exemples de prédictions très erronées

3.6 Tâche 5 : Discussion et analyse critique

1. **Synthèse des résultats** :
   - Résumez les principaux résultats obtenus dans chaque tâche
   - Identifiez les insights les plus importants
   - Reliez les résultats entre les différentes tâches
2. **Limitations identifiées** :
   - **Sparsité** : Quel est l'impact de la sparsité extrême (>99% après échantillonnage) ?
   - **Scalabilité** : Vos approches passeraient-elles à l'échelle du dataset complet (29.5M reviews) ?
   - **Biais** : Avez-vous identifié des biais (temporels, de popularité, etc.) ?
3. **Défis de volumétrie** :
   - Quels ont été les principaux défis rencontrés avec ce dataset ?
   - Comment avez-vous optimisé l'utilisation de la mémoire ?
   - Quelles techniques d'optimisation ont été les plus efficaces ?
   - Qu'auriez-vous fait différemment avec des ressources computationnelles illimitées ?

4 Livrables et critères d'évaluation

Le projet doit être réalisé par groupe de minimum 3 étudiants et maximum 4 étudiants.

Date de remise du projet, 20 février 2026 au plutard à 23h59.

Suivre le lien suivant pour créer votre groupe :

`https://docs.google.com/spreadsheets/d/1YSmV8M-DazOPPtGeMv-xAHTGIPbRGSE2Wx564OtKEGo/edit?usp=sharing`

4.1 Format de remise

Les étudiants doivent soumettre un fichier compressé (ZIP) nommé `INF6083-P1-EquipeN.zip` contenant :

1. **Rapport (PDF)** : `INF6083-P1-EquipeN-Rapport.pdf`
   - Maximum 20 pages (excluant les annexes)
   - Structure claire suivant les 5 tâches du projet
   - Figures et tableaux numérotés et légendés
   - Décrire votre stratégie d'échantillonnage

   - Analyse et interprétation des résultats
   - **NE PAS** inclure de code source ni de capture d'écran de votre code source dans votre rapport.
2. **Code source** : `INF6083-P1-EquipeN-Code.ipynb` ou fichiers `.py`
   - Code bien commenté et structuré
   - Instructions d'exécution claires
   - Fichier `requirements.txt` avec les dépendances
   - Scripts de chargement et d'échantillonnage
   - Gestion efficace de la mémoire documentée
3. **Données** :
   - **NE PAS** inclure les fichiers Amazon originaux (trop volumineux)
   - Inclure les résultats intermédiaires essentiels (matrices de similarité, clusters, etc.) en format compact
4. **README** : `README.md`
   - Noms et matricules des membres de l'équipe
   - Instructions de téléchargement du dataset Amazon
   - Instructions d'installation et d'exécution
   - Organisation du code

4.2 Critères d'évaluation

Le projet sera évalué selon les critères suivants :

Critère	Description
Tâche 0	Chargement efficace, échantillonnage justifié, exploration complète, prétraitement approprié
Tâche 1	Implémentation correcte des similarités, optimisation pour la volumétrie, analyse comparative approfondie
Tâche 2	Construction et analyse du graphe, métriques pertinentes, visualisations claires, interprétations
Tâche 3	K-Means optimisé, justification du K, caractérisation des clusters, visualisations 2D
Tâche 4	Baselines correctes, k-NN optimisé, analyse comparative, gestion des cas limites
Tâche 5	Synthèse pertinente, identification des limitations, propositions d'amélioration, réflexion sur la volumétrie
Qualité du code	Exécutable, optimisé pour la mémoire, bien structuré, commenté, reproductible
Qualité du rapport	Structure claire, figures professionnelles, interprétations approfondies, français correct, références