Compte Rendu de Projet

SAE821 – Gérer un projet

Rapport de présenation de la modélisation d'un système de recommandation respectant la vie privée

---

Pegliasco Matteo
Berge Enzo
Audouard Florian
Hermelin Lois
Master Informatique et Mathématiques
Université de Toulon, La Garde, Var, France

Table des matières

Introduction

Ce rapport présente les travaux réalisés dans le cadre de la SAE821, dont l'objectif est la conception et la modélisation d'un système de recommandation de films intégrant des mécanismes de protection de la vie privée. Les systèmes de recommandation sont aujourd'hui omniprésents : ils jouent un rôle central dans l'orientation des utilisateurs vers des contenus adaptés à leurs préférences, que ce soit sur les plateformes de commerce en ligne, les réseaux sociaux ou les services de streaming. Cependant, leur efficacité repose souvent sur l'exploitation massive de données personnelles, ce qui soulève d'importantes problématiques liées à la confidentialité, la sécurité et l'intégrité des informations collectées. Face à ces enjeux, il devient essentiel de concevoir des systèmes capables de proposer des recommandations pertinentes tout en garantissant un haut niveau de protection des données utilisateurs. Dans cette optique, le présent projet vise à modéliser un système de recommandation respectueux de la vie privée. Le rapport expose dans un premier temps les fondements théoriques de ce type de système, avant de détailler les choix techniques effectués et les approches mises en œuvre pour concilier personnalisation et respect de la confidentialité.

1 Recherches théoriques

1.1 Systèmes de recommandation

1.1.1 Histoire et évolution des systèmes de recommandation

Comme présenté par Dabbas (2018) [7], les systèmes de recommandation ont connu une évolution progressive au fil des décennies. Les premières approches, développées dans les années 1990, reposaient sur des techniques relativement simples comme le filtrage collaboratif. Cette méthode suppose que les utilisateurs ayant manifesté des préférences similaires dans le passé auront des goûts comparables à l'avenir. Deux variantes principales existent : l'approche centrée utilisateur, qui identifie des individus aux comportements proches, et l'approche centrée objet, qui cherche des éléments similaires à ceux déjà appréciés par un utilisateur donné. En parallèle, le filtrage basé sur le contenu se concentre sur les caractéristiques intrinsèques des éléments consultés, sans prendre en compte les autres utilisateurs.
Au début des années 2000, les recherches ont convergé vers des modèles hybrides combinant plusieurs approches. Cette hybridation permet de pallier les faiblesses de chaque méthode, notamment en cas de démarrage à froid. Par ailleurs, l'introduction du deep learning a marqué une avancée majeure en permettant aux systèmes de découvrir automatiquement des représentations complexes des utilisateurs et des objets, grâce à des réseaux de neurones entraînés sur de larges jeux de données.
Les années 2010 ont vu l'apparition de systèmes intégrant des informations contextuelles telles que le moment de la consultation, la localisation géographique, le type de terminal ou encore l'état émotionnel de l'utilisateur. D'autres systèmes se sont appuyés sur des bases de connaissances structurées, exploitant des règles logiques et des ontologies pour enrichir les profils utilisateurs, notamment dans des cas où les données sont limitées.
Plus récemment, les approches par apprentissage par renforcement ont introduit une dimension interactive dans la recommandation. Ces systèmes ajustent leurs suggestions en fonction des réactions successives des utilisateurs, optimisant les résultats à long terme. En parallèle, la notion d'explicabilité s'est développée : il ne s'agit plus seulement de recommander, mais aussi d'expliquer pourquoi une suggestion est faite, ce qui renforce la confiance dans le système.

Catégories de systèmes de recommandation :

Il existe plusieurs catégories de systèmes de recommandation selon leur architecture et leurs objectifs. Certains sont basés sur la mémoire (comme le filtrage collaboratif simple), d'autres reposent sur des modèles mathématiques ou statistiques plus élaborés, notamment la factorisation de matrices ou l'utilisation de réseaux neuronaux. Les systèmes basés sur le contenu utilisent quant à eux les attributs des objets pour établir des correspondances. Les modèles hybrides cherchent à combiner les forces de ces différentes approches. Les systèmes contextuels intègrent des dimensions supplémentaires pour améliorer la pertinence. Les systèmes basés sur la connaissance s'appuient sur des règles explicites et des bases de faits. D'autres intègrent l'apprentissage par renforcement, ou encore la capacité d'expliquer les recommandations. Enfin, certains modèles plus récents adoptent une approche multimodale, en combinant

plusieurs types de données (texte, image, son), ou incluent directement des mécanismes de protection de la vie privée.
Parmi les approches avancées utilisées pour construire des systèmes performants, la factorisation de matrice occupe une place centrale. Une des méthodes les plus répandues est celle des moindres carrés alternés (ALS), qui vise à approximer une matrice de notes par le produit de deux matrices de dimension réduite représentant respectivement les utilisateurs et les objets. Cette méthode permet de retrouver les préférences implicites à partir de données incomplètes tout en limitant le surapprentissage grâce à un terme de régularisation.

- La décomposition en valeurs singulières (SVD) constitue une autre technique importante. Elle consiste à décomposer une matrice utilisateur-objet en trois matrices contenant les vecteurs de préférences des utilisateurs, les caractéristiques des objets, et les valeurs singulières associées. Le produit scalaire entre ces vecteurs permet d'estimer la note qu'un utilisateur donnerait à un objet donné. L'algorithme ajuste ces vecteurs de façon à minimiser l'erreur entre les notes prédites et les notes réelles connues.
- Les réseaux de neurones récurrents (RNN) offrent une solution efficace pour traiter des données séquentielles, comme l'historique de navigation ou les séries de clics. Grâce à leur mémoire interne, ils sont capables de modéliser les dépendances temporelles entre les actions des utilisateurs. À chaque instant, le réseau reçoit une nouvelle entrée et met à jour son état en fonction des informations précédentes, ce qui lui permet de prédire l'action suivante ou de recommander un contenu en cohérence avec le comportement récent.
- Les réseaux de neurones convolutifs (CNN) sont généralement utilisés pour traiter des données visuelles ou audio, mais ils peuvent également être appliqués à la recommandation si l'on cherche à extraire des motifs locaux à partir de signaux structurés. Ces réseaux utilisent des couches de convolution pour capter des régularités locales, des opérations de pooling pour réduire la dimensionnalité, puis des couches denses pour prendre des décisions finales.
- Enfin, l'approche dite du "multi-armed bandit" constitue un cas particulier d'apprentissage par renforcement, où le système cherche à maximiser la récompense (ex. : satisfaction de l'utilisateur) tout en explorant de nouvelles options. Il s'agit d'un équilibre complexe entre exploitation des recommandations déjà validées et exploration de contenus moins connus mais potentiellement intéressants.

1.1.2 Filtrage collaboratif

Le filtrage collaboratif est une méthode de recommandation qui s'appuie sur les interactions passées entre les utilisateurs et les objets (tels que des films, produits ou musiques). Contrairement aux approches basées sur le contenu, cette technique ne nécessite aucune information sur les caractéristiques des objets à recommander. Elle repose sur l'idée que des utilisateurs ayant eu des comportements similaires dans le passé auront vraisemblablement des goûts semblables à l'avenir.
Son fonctionnement suit plusieurs étapes clés. Tout d'abord, les données sont collectées à partir des interactions utilisateur-objet. Ces données peuvent être explicites, comme les notes attribuées ou les avis écrits, ou implicites, comme les clics, la durée de consultation ou le nombre de consultations. Une fois cette phase de collecte terminée, on construit une matrice utilisateur-objet, dans laquelle chaque ligne représente un utilisateur, chaque colonne un objet, et chaque cellule contient l'information d'interaction correspondante.
À partir de cette matrice, des mesures de similarité sont calculées. Cela peut concerner les utilisateurs entre eux (approche dite user-based) ou les objets entre eux (approche item-based). Ces similarités sont mesurées à l'aide d'outils mathématiques tels que la similarité cosinus ou le coefficient de corrélation de Pearson. Une fois les similarités identifiées, le système peut prédire la préférence d'un utilisateur pour un objet qu'il n'a pas encore évalué, en se basant sur les évaluations d'utilisateurs ou objets similaires. Ces prédictions permettent ensuite de générer des recommandations personnalisées.
Prenons l'exemple suivant pour illustrer le principe. Soit la matrice de notation suivante :

	Film A	Film B	Film C	Film D
Utilisateur 1	5	?	4	2
Utilisateur 2	4	3	5	1
Utilisateur 3	2	4	1	5

Dans ce tableau, l'utilisateur 1 n'a pas encore noté le film B. Le système peut alors estimer cette note en analysant les comportements des utilisateurs 2 et 3, si ceux-ci présentent un profil similaire à celui de l'utilisateur 1.
Le filtrage collaboratif présente plusieurs avantages. Il ne dépend pas des métadonnées des objets, ce qui le rend applicable même lorsque ces informations sont manquantes ou incomplètes. Il permet également de générer des recommandations souvent personnalisées et pertinentes, y compris inattendues, en se basant sur les préférences collectives. Le système évolue naturellement avec les comportements des utilisateurs, ce qui le rend dynamique et adaptable.
Cependant, cette méthode comporte aussi des limitations. Elle souffre notamment du problème du démarrage à froid : en l'absence d'historique, les recommandations pour un nouvel utilisateur ou un nouvel objet sont peu fiables. De plus, les matrices de notation sont souvent très clairsemées, ce qui complique le calcul de similarités précises. Enfin, en se basant uniquement sur les préférences de groupes d'utilisateurs, le système peut parfois produire des recommandations peu pertinentes pour un individu spécifique.
Le filtrage collaboratif est aujourd'hui utilisé dans de nombreux domaines. Il est largement exploité dans le divertissement (Netflix, Spotify), le commerce en ligne (Amazon), l'éducation (recommandation de cours ou de contenus pédagogiques) et les réseaux sociaux (suggestions de contacts ou de groupes).

1.1.3 Factorisation de matrice

La factorisation de matrice [9] est une technique de recommandation basée sur les utilisateurs qui approxime les préférences des utilisateurs par le produit de deux matrices de plus faible dimension. Ces matrices représentent des caractéristiques latentes des utilisateurs et des items — ici, les films. Comme évoqué précédemment, cette approximation peut être obtenue à l'aide de l'algorithme des moindres carrés alternés ou de la descente de gradient stochastique. Cette méthode s'applique efficacement à de grands ensembles de données, peut gérer la sparsité des notations, et peut être étendue pour intégrer des informations supplémentaires (comme des métadonnées ou du contenu).

Par exemple :

Considérons une matrice de notes R représentant les notes données par 2 utilisateurs à 4 films :

$$R = \begin{bmatrix} 5 & 3 & \_ & 1 \\ 4 & \_ & \_ & 1 \end{bmatrix}$$

où R_{ij} est la note donnée par l'utilisateur i au film j, et « _ » représente une note manquante et sera remplacé par 0 lors de l'approximation de la factorisation.
L'objectif de la factorisation est donc d'approcher R par le produit de deux matrices : $R \approx U \cdot V$. En partant de deux matrices de tailles arbitraires et en corrigeant les erreurs, on peut obtenir une approximation de la matrice de notes R comme suit :

$$R \approx \begin{bmatrix} 1.5 & 0.5 \\ 1.2 & 0.4 \end{bmatrix} \cdot \begin{bmatrix} 3.0 & 1.5 & 0.5 & 0.2 \\ 1.0 & 1.0 & 0.2 & 0.5 \end{bmatrix} = \begin{bmatrix} 5.00 & 2.75 & 0.85 & 0.55 \\ 4.00 & 2.20 & 0.68 & 0.44 \end{bmatrix}$$

Avec cette factorisation, la méthode prédit alors que l'utilisateur 1 donnerait une note proche de 1 au troisième film et l'utilisateur 2 donnerait une note proche de 2 pour le deuxième films et une notre 0.5 pour le troisième film. Toutefois, cette approche présente certaines limites.
Tout d'abord, elle nécessite une étude profonde de ses hyperparamètres pour obtenir des résultats pertinents, notamment le nombre de facteurs latents à utiliser. De plus, elle est sensible au problème du démarrage à froid pour les nouveaux utilisateurs ou items — films dans le cas présent — et le coût de calcul peut devenir important selon la taille du jeu de données.

1.2 Système de recommandation à froid

Au fil du temps, plusieurs approches ont été développées pour répondre aux problèmes de recommandation à froid, c'est-à-dire pour suggérer de nouveaux films ou proposer des recommandations à de nouveaux utilisateurs. Parmi celles-ci, nous pouvons retrouver les approches suivantes :

— **Les méthodes basées sur le contenu (content-based)**, présentées par Schein, A. I., Popescul, A., Ungar, L. H., et Pennock, D. M. (2002) [2], exploitent les caractéristiques des items — ici des films — telles que la description, le genre ou les acteurs. En calculant la similarité entre les caractéristiques des films (par exemple avec la similarité cosinus), on recommande ceux qui sont proches de ceux déjà appréciés par un utilisateur de référence.
— **Les approches hybrides** combinent filtrage collaboratif et filtrage basé sur le contenu afin de bénéficier des avantages des deux. Par exemple, comme présenté par Xiaoyuan Su et Taghi M. Khoshgoftaar (2010) [4], plusieurs implémentations sont possibles : on peut d'abord appliquer séparément une méthode collaborative et une méthode basée sur le contenu, puis combiner les résultats pour générer des recommandations plus pertinentes, ou encore développer un modèle unifié intégrant à la fois les caractéristiques du contenu et celles du filtrage collaboratif. L'ajout progressif d'interactions utilisateur permet ainsi d'améliorer la précision des recommandations.
— **Les méthodes basées sur le clustering** cherchent à regrouper les utilisateurs ou les items selon leurs caractéristiques communes pour générer des recommandations. Une approche non supervisée, présentée par Snider (2019) [8], applique des techniques de traitement automatique du langage pour former des clusters d'items sans interaction préalable avec l'utilisateur, ce qui la rend adaptée au démarrage à froid.
— **Les méthodes de créations de profils actifs** certaines approches, comme celle utilisée par Nguyen et al. (2020) [15], amène l'utilisateur à évaluer un ensemble d'items représentatifs pour construire son profil.

1.3 Confidentialité

1.3.1 Chiffrement homomorphe

Le chiffrement homomorphe est une technique de cryptographie qui permet d'effectuer des calculs sur des données chiffrées sans avoir besoin de les déchiffrer au préalable, tout en garantissant que le résultat déchiffré correspond aux opérations effectuées sur les données claires. Pour cela, plusieurs catégories de chiffrement homomorphe existent :
— **Chiffrement partiellement homomorphe** : Il permet de réaliser des opérations sur des données chiffrées selon une seule loi, par exemple l'addition ou la multiplication.
— **Chiffrement quasi-entièrement homomorphe** : Il permet de réaliser un nombre d'opérations limité sur des données chiffrées, pour plusieurs lois, par exemple l'addition et la multiplication, mais la capacité est limitée à cause de l'accumulation de bruit.
— **Chiffrement entièrement homomorphe** : Il permet d'effectuer un nombre illimité d'opérations sur des données chiffrées, pour toutes les lois possibles, bien qu'il soit limité par la taille du bruit accumulé lors des opérations.

Dans le cadre de ce projet, nous nous sommes particulièrement intéressés au chiffrement entièrement homomorphe pour sa flexibilité d'utilisation dans différents systèmes de recommandation. Plusieurs approches existent pour la mise en place du chiffrement homomorphe :
— **Chiffrement DGHV avec bootstrapping [12]** : Le chiffrement DGHV [10] est un système homomorphe opérant sur des entiers, basé sur le problème de la somme de sous-ensembles clairsemée et le problème du PGCD approché.
— **Chiffrement GSW avec bootstrapping [12]** : Le chiffrement GSW [11] est un système homomorphe opérant sur des bits, basé sur le problème de la factorisation des entiers et le problème du logarithme discret.
— **Chiffrement TFHE** : Le chiffrement TFHE [14] est un système homomorphe opérant sur des bits, basé sur la version torique du problème Learning With Errors.

Cependant, même si le chiffrement homomorphe permet de préserver la confidentialité des données, son utilisation présente plusieurs contraintes :
— Les données chiffrées sont généralement beaucoup plus volumineuses que les données claires, ce qui peut poser des problèmes de stockage et de transmission. Par exemple, dans l'article de I. Chillotti, N. Gama, M. Georgieva, et M. Izabachène (2016) [6], il est indiqué que la taille du texte chiffré peut être jusqu'à 400 000 fois supérieure à celle des données d'origine.
— L'utilisation du "bootstrapping" [12], qui sert à limiter l'accumulation de bruit, augmente le temps de calcul lorsque de nombreuses opérations sont réalisées. En effet, en agissant comme un contrôle

sur la taille du bruit, les opérations de bootstrapping allongent la durée des calculs sur les textes chiffrés.
— La mise en place sans ressources externes est complexe, et les ressources disponibles — sans même considérer leur fiabilité — sont principalement fournies via des bibliothèques en C++ ou Rust, telles que Microsoft SEAL, HElib ou PALISADE, ce qui complique l'interopérabilité avec d'autres langages de programmation comme Java.

1.3.2 Differential privacy

La confidentialité différentielle est une approche qui vise à garantir la protection de la vie privée des individus dans les ensembles de données en limitant la corrélation entre les données de sortie et les données personnelles d'un utilisateur. Pour cela, plusieurs approches existent :
— **Mécanisme exponentiel** : Le mécanisme exponentiel [13] permet de sélectionner aléatoirement un élément parmi un ensemble de données, en favorisant les éléments ayant le plus de poids selon une fonction d'utilité donnée.
— **Réponse aléatoire** : Le mécanisme de réponse aléatoire, telle qu'utilisée par Begin G. et Savard F. [1], vise à brouiller les réponses en remplaçant aléatoirement certaines réponses par des valeurs aléatoires. En connaissant la probabilité de remplacement, il est possible d'estimer la proportion de données réelles et leur variance.
— ***K-anonymisation* et *L-diversité*** : La *K-anonymisation* [3] et la *L-diversité* consistent à masquer les éléments de quasi-identification d'un individu en le regroupant avec $K-1$ autres individus dans une représentation plus large, tout en conservant la visibilité des L informations spécifiques à chaque utilisateur destinées à être partagées.
— **Mécanisme de Laplace ou Gaussien** : Le mécanisme de Laplace ou Gaussien [5] consiste à altérer les données des utilisateurs en ajoutant un bruit aléatoire calibré, afin de masquer les données réelles tout en conservant les caractéristiques statistiques globales.

2 Démarche choisie

2.1 Motivations du choix

2.1.1 Confidentialité des données

Le cadre de ce projet est de mettre en place un système de recommandation de films respectant la vie privée des utilisateurs. Nous avons considéré que la base de données des utilisateurs ainsi que les films proposés pouvaient évoluer au fil du temps. Pour la protection de la vie privée des utilisateurs, nous avions considéré l'utilisation du chiffrement homomorphe TFHE, mais la complexité de son implémentation et de son utilisation, avec notamment l'interaction entre différents langages de programmation (Java, C++, Rust), ainsi que la complexité de l'utilisation de l'apprentissage machine avec une bibliothèque spécialisée, nous a poussés à nous intéresser davantage à la confidentialité différentielle.
Plus précisément, nous nous sommes particulièrement intéressés à la mise en œuvre du mécanisme de bruitage et du mécanisme exponentiel dans le cadre de la confidentialité différentielle, respectivement pour les recommandations destinées aux nouveaux utilisateurs et pour celles personnalisées pour les utilisateurs ayant déjà évalué plusieurs films. En ce qui concerne le mécanisme de génération du bruit, nous avons choisi d'utiliser le mécanisme de Laplace pour masquer les données des utilisateurs. Ce choix est motivé par le contexte de ce projet. En effet, les utilisateurs peuvent modifier leur evaluation d'un film. Dans cet optique, et contrairement au mécanisme gaussien, nous avons favorisé le mécanisme de Laplace, car il garantit une confidentialité différentielle ϵ, c'est-à-dire que toute modification d'une entrée dans la base de données engendre une modification bornée de la distribution des probabilités De plus, étant donné que la base de données initiale possède relativement peu d'évaluations utilisateurs pour chaque film, le mécanisme de Laplace est privilégié puisque la génération de bruit à forte valeur est moins probable que pour le mécanisme gaussien, dû à une distribution plus centrée en 0 pour la courbe Laplacienne, ce qui permet de garantir une meilleure précision dans l'étude de la moyenne des évaluations des films.
Enfin, dans le cadre d'une protection des données privée des utilisateurs, nous avons eu la volonté de limiter l'utilisation de ressources externes pour la génération des données bruitées afin de reposer au minimum sur la fiabilité de ces ressources et avoir un maximum de contrôle sur la génération de bruit

(c.f 2.1.2 Analyse du bruit Laplacien). Ce faisant, l'implémentation du mécanisme gaussien impliquerait l'implémentation complexe, avec le temps disponible, de l'approximation de l'inverse de la fonction :

$$F(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}} dt \tag{1}$$

afin de générer du bruit. Ainsi, nous avons privilégié l'implémentation plus simple du mécanisme de Laplace qui s'adapte mieux aux traitements ponctuels sur des bases de données de taille variable, correspondant aux besoins de notre système d'entraînement pour nos recommandations. En complément, et comme présenté précédemment, nous avons considéré l'utilisation du mécanisme exponentiel pour une sélection aléatoire biaisée d'un film recommandé. Cette approche permet à la fois de favoriser certains films - ici ceux qui ont une note moyenne la plus éloignée de 2.5 pour recommander des films qui n'ont jamais été notés ou des films avec une bonne notation - grâce à une fonction d'utilité définie dans le système, tout en limitant le risque que l'utilisateur puisse extraire des informations individuelles sur les autres utilisateurs ayant permis de générer les recommandations avec une bonne calibration.

2.1.2 Analyse du bruit Laplacien

Comme présenté plus tôt, la mise en place manuelle du mécanisme de Laplace pour la génération de bruit permet un meilleur contrôle sur la génération du bruit. Cette section vise à présenter les implications du choix des paramètres du mécanisme de Laplace pour la génération de bruit.
La génération du bruit de Laplace est basée sur l'inversion de la fonction de répartition de Laplace. Plus précisément, et comme mise en place dans le système en posant b le budget de confidentialité, la génération du bruit de Laplace peut s'écrire sous la forme :

$$Laplace(b) = b \cdot \text{sign}(x) \cdot \ln(1 - 2|x|) \tag{2}$$

, où x est une variable aléatoire suivant la loi uniforme continue sur l'intervalle $]-0.5, 0.5[$.
À partir de cette écriture, il est alors possible de définir la variance du bruit généré par le mécanisme de Laplace ainsi que de choisir le budget de confidentialité b de façon à ce que la probabilité que le bruit généré dépasse une certaine limite L soit inférieure à un seuil p.

Propriété 1 :
Soit $b > 0$.
La variance fonctionnelle du bruit généré par le mécanisme de Laplace est donnée par : $2b^2$.

Démonstration :
Plaçons nous dans les conditions de l'énoncé.
La variance fontionnelle de la fonction $\phi : x \mapsto Laplace(b)$, noté $V(\phi)$, est donnée par : $\int_{-0.5}^{0.5}(Laplace(b) - \overline{Laplace})^2 dx$, où $\overline{Laplace}$ est la moyenne de la fonction ϕ sur l'intervalle $]-0.5, 0.5[$.
La fonction ϕ étant antisymétrique, on a alors que $V(\phi) = \int_{-0.5}^{0.5} Laplace(b)^2 dx = \int_{-0.5}^{0.5} b^2 \ln(1 - 2|x|)^2$.

Par symétrie de la fonction $x \mapsto \ln(1 - 2|x|)$, on a : $V(\phi) = 2b^2 \int_{-0.5}^{0} \ln(1 + 2x)^2 dx$.
En posant $X = 1 + 2x$, on a alors : $V(\phi) = 2b^2 \int_0^1 \frac{\ln(X)^2}{2} dX$.
En utilisant l'intégration par parties, on peut alors écrire : $V(\phi) = b^2([X \ln(X)]_0^1 - 2[X(\ln(X) - 1)]_0^1)$.
Par croissance comparée , on obtient le résultat : $V(\phi) = 2b^2$.

Propriété 2 :
Soient $p \in]0, 1[$, $L > 0$ et $b > 0$.
$P(|Laplace(b)| > L) \leq p$ si et seulement si $b \leq \frac{-L}{\ln(p)}$.

Démonstration :
Plaçons nous dans les condition de l'énoncé.
Sachant que l'évènement $(|Laplace(b)| > L)$ est équivalent à $(Laplace(b) < -L) \cup (Laplace(b) > L)$, on a alors que : $P(|Laplace(b)| > L) = P(Laplace(b) < -L) + P(Laplace(b) > L)$ car les évènements sont disjoints.

Comme $b > 0$ par hypothèse et que $x \mapsto \ln(1 - 2|x|)$ est négative sur l'intervalle $]-0.5, 0.5[$, on a

alors que : $(Laplace(b) > L) \Leftrightarrow (-b\ln(1+2x) > L)$ et $(Laplace(b) < -L) \Leftrightarrow (b\ln(1-2x) < -L)$.
Ainsi : $(Laplace(b) > L) \Leftrightarrow (-b\ln(1+2x) > L) \Leftrightarrow (x < \frac{e^{-L/b}-1}{2})$ avec $x \in]-0.5, 0[$
Et : $(Laplace(b) < -L) \Leftrightarrow (b\ln(1-2x) < -L) \Leftrightarrow (x > \frac{1-e^{-L/b}}{2})$ avec $x \in]0, 0.5[$.

Par hypothèse, comme x suit une une loi uniforme sur l'intervalle $]-0.5, 0.5[$, on a alors que :

$$P(Laplace(b) > L) = \frac{e^{-L/b}-1}{2} - (-0.5) = \frac{e^{-L/b}}{2}$$
$$P(Laplace(b) < -L) = 0.5 - \frac{1-e^{-L/b}}{2} = \frac{e^{-L/b}}{2}$$

Et donc : $P(|Laplace(b)| > L) = e^{\frac{-L}{b}}$.
Par croissance des fonctions $x \mapsto e^x$ et $x \mapsto \ln(x)$ et comme $\ln(x) < 0$ sur $]0, 1[$, on a alors le résultat suivant : $P(|Laplace(b)| > L) \leq p \Leftrightarrow e^{\frac{-L}{b}} \leq p \Leftrightarrow b \leq \frac{-L}{\ln(p)}$.

Ainsi, avec une implémentation manuelle du mécanisme de Laplace, il est alors possible de manipuler le budget de confidentialité b pour contrôler la génération de bruit tout en maximisant la variance du bruit généré pour augmenter la confidentialité des données. Toutefois, le choix de ce budget de confidentialité necessite une analyse approfondie des données à protéger, car un budget trop faible peut conduire à une fuite de donnée tandis qu'un budget trop élevé empêchera l'analyse statistique des données.

Pour cela, nous avons fixé notre budget de confidentialité à $b = 0.5$ afin de s'adapter aux données fournies par MovieLens, où les films possèdent au plus 329 évaluations, nécessitant ainsi une génération faible de bruit pour permettre l'étude statistique.
Grâce à la propriété 2, nous savons alors que le bruit généré par le mécanisme de Laplace est tel que la probabilité que le bruit dépasse 1 est d'environ 0.135 (car $0.5 = \frac{-1}{\ln(e^{-2})} \approx \frac{-1}{\ln(0.135)}$) et que la probabilité que le bruit dépasse 0.5 est d'environ 0.368 (car $0.5 = \frac{-0.5}{\ln(e^{-1})} \approx \frac{-0.5}{\ln(0.368)}$)

2.1.3 Modèles de recommandations

Pour le système de recommandation, nous avons choisi de mettre en place deux approches pour la recommandation de films : un système se basant sur le SVD pour un utilisateurs connus du système - individu ayant déjà noté un certain nombre de films - et un autre système de recommandation basé sur la clustering non supervisé pour la recommandation à froid d'un nouvel utilisateur.

— **Pour le SVD :**

Le système de recommandation à chaud mis en place repose sur une approche collaborative fondée sur la décomposition en valeurs singulières (SVD). Inspirée des travaux classiques en filtrage collaboratif, cette méthode permet de prédire les préférences d'un utilisateur connu du système, c'est-à-dire ayant déjà noté un certain nombre de films.

1. Construction de la matrice d'interactions : une matrice utilisateur–film est construite à partir des notes données par les utilisateurs. Chaque ligne représente un utilisateur et chaque colonne un film, les entrées correspondant aux notes attribuées.
2. Factorisation de la matrice : cette matrice est ensuite factorisée en trois sous-matrices :

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

— U représente les facteurs latents des utilisateurs,
— V les facteurs latents des films,
— Σ une matrice diagonale contenant les valeurs singulières.
Cette décomposition permet d'identifier des facteurs implicites (préférences, styles, affinités thématiques) non directement observables dans les données brutes.

3. Prédiction des notes : une fois les vecteurs latents appris, le système peut estimer la note qu'un utilisateur donnerait à un film non encore vu, en effectuant le produit scalaire entre les

vecteurs latents correspondants. Les films obtenant les meilleures prédictions peuvent alors être recommandés.

L'objectif de ce système de recommandation est donc, à la fois, de proposer des suggestions personnalisées aux utilisateurs connus du système, en exploitant les préférences implicites issues des notations passées, de maximiser la pertinence des recommandations en limitant la redondance, et de tirer parti des corrélations latentes entre utilisateurs et films pour améliorer la qualité globale des prédictions.

— **Pour le système de clustering :**

Le système de recommandation à froid mis en place se base sur une recommandation item-based via un système de clustering, à l'image de Snider 2019 [8] et de la sélection aléatoire de films. Plus précisément, l'algotithme de recommandation se compose en trois étapes principales :

1. Sélection des films pouvant être recommandés selon certain critères. Dans notre implémentation, les films recommandables sont les films ayant une note moyenne supérieure ou égale à un seuil (ici, 3) ou ayant un nombre d'évaluations inférieur ou égal à un seuil (ici, 0). Cette approche permet alors de recommander des films étant globalement appréciés par les utilisateurs l'ayant noté et donc de recommander des films qui pourront plaire à l'utilisateur mais aussi de recommander des films peu notés pour obtenir plus d'informations statistiques sur ces films tout en favorisant la diversité des films recommandés.
2. Clustering des films : Une fois nos films sélectionnés, des vecteurs de caractéristiques sont générés à partir de leurs informations descriptives. Ici, nous avons utilisé les genres et le synopsis des films ainsi que leurs dates de sortie (année, jour et mois) pour générer un vecteur de caractéristiques. Pour générer le vecteur de caractéristiques des genres, respectivement du synopsis, dans le serveur Python, nous avons utiliser la fonction "MultiLabelBinarizer" de la bibliothèque "sklearn" et la fonction "SentenceTransformer" de la bibliothèque Python "sentence_transformers".
   Après la normalisation de ces vecteurs, une réduction de dimension est appliquée (via l'algorithme UMAP) suivie d'un clustering non supervisé (DBSCAN) comme présenté par les tables en annexe 1 2 . L'utilisation de l'algorithme DBSCAN permet ainsi de regrouper les films selon leur similarité - ici, la similarité cosinus - tout en offrant une meilleure adapation à l'évolution des films dans la base de données que des clusters supervisés.
3. Sélection aléatoire de films : Afin de favoriser la diversité des films recommandés, le système sélectionne aléatoirement un cluster, puis, de manière biaisée pour favoriser les films avec la moyenne la plus faible (représentant les films sans évaluation), un film au sein de celui-ci. Pour ce faire, nous avons utilisé une fonction d'utilité définie dans le système pour favoriser les films ayant une note moyenne la plus proche de 0, qui après le tri de l'étape 1, représente les films les moins notés.

L'objectif de ce système de recommandation vise donc, à la fois, de représenter la diversité des films dans la base de données, limiter la redondance des recommandations avec les choix aléatoires de clusters et de films, mais aussi de recommander des films ayant un certain intêret, ici les films appréciés en moyenne par les utilisateurs ou les films peu notés (car possiblement récent ou peu vu).

2.2 Implémentation dans le système

Cette partie sert à présenter l'implémentation de la démarche choisie dans le système de recommandation de films tout en résumant l'approche adoptée pour la mise en place de la confidentialité des données.
Ainsi pour implémenter ces différents éléments, nous avons structuré notre projet de la manière suivante :

— Un serveur de base de données PostgreSQL, implémenté en Java via le framework Quarkus, dédié au stockage des informations des films ainsi que les informations relatives aux utilisateurs. Dans un optique de performance dans le cadre d'une utilisation en production, ce serveur est relié à une

couche de recherche rapide via une base de données NoSQL, elle aussi implémentée en Java, qui permet d'augmenter la rapidité de réponse du système lors des recherches.
— Un serveur de recommandation, implémenté en Python, dédié aux recommandations à chaud et à froid. Il est relié au serveur de base de données pour récupérer les informations nécessaires à la génération des recommandations via un transfert de fichiers CSV.

Plus précisément, notre implémentation nécessite l'initialisation de la base de données via une requête au serveur Python. À partir de cette requête, le serveur Python va alors récupérer les films ainsi que les évaluations stockées dans la base de données afin d'initialiser et d'entraîner les modèles de recommandation. Lors de cette étape et afin de garantir la confidentialité des données, les notes utilisateurs sont alors chiffrées avec le mécanisme de Laplace.
Par la suite, lors de la demande de recommandations pour un utilisateur, la base de données est interrogée pour obtenir le nombre d'évaluations données par celui-ci. Si l'utilisateur a un nombre de notes supérieur à un seuil (ici, 0), le système de recommandation à chaud est utilisé pour générer des recommandations personnalisées. Dans le cas contraire, le système de recommandation à froid est utilisé pour générer le double des recommandations demandées. Cette augmentation du nombre de recommandations vise, comme présenté précédemment, à garantir la confidentialité des données en sélectionnant aléatoirement la moitié des recommandations en privilégiant les films avec les notes moyennes les plus élevés ou les plus faiblesles films avec les notes moyennes les plus élevées ou les plus faibles (pour lequelle la note moyenne est fixée à 0).

3 Résultats expérimentaux

3.1 Précision des recommandations et performance computationnelles

3.1.1 Système de recommandation à chaud

L'évaluation de notre système de recommandation à chaud met en évidence des résultats satisfaisants, tant sur le plan de la qualité des suggestions que sur celui des performances techniques. Sur le plan de la précision des recommandations , le modèle atteint un score de **72,2%**, ce qui indique qu'une large majorité des éléments recommandés sont pertinents pour les utilisateurs. Le **rappel**, quant à lui, s'élève à **69,1%**, traduisant la capacité du système à retrouver une part importante des éléments pertinents parmi l'ensemble de ceux disponibles. Ces résultats démontrent un bon équilibre entre exactitude et couverture, ce qui est crucial dans le contexte d'un système à chaud, où les préférences peuvent évoluer rapidement. Concernant les performances computationnelles, les mesures effectuées montrent une **initialisation des données très rapide (0,14 seconde)**, suivie d'une **entraînement du modèle en 0,72 seconde**, ce qui reste tout à fait raisonnable dans un usage en ligne ou en quasi temps réel. Le **temps de réponse moyen aux requêtes** est particulièrement bas, avec une valeur de **0,03 seconde**, garantissant une expérience fluide et réactive pour l'utilisateur. Ces résultats confirment l'efficacité du modèle SVD employé dans un cadre de recommandations à chaud, capable de fournir des suggestions pertinentes tout en maintenant une latence très faible.

3.1.2 Système de recommandation à froid

Pour les systèmes de recommandation à froid, les notions de précision et de rappel ne sont pas adaptées, car le système ne dispose pas d'informations sur les préférences des utilisateurs. Il ne peut donc pas prédire quels films seraient pertinents pour eux. Notre objectif avec ce type de système se centre principalement sur la représentation de la diversité des films présents dans la base de données, tout en limitant la redondance des recommandations et en mettant en avant des films peu notés ou bien évalués.
Ainsi, nous avons choisi d'évaluer la qualité des recommandations à l'aide des métriques suivantes :

— Le nombre de clusters différents présents dans les recommandations pour un utilisateur.
— Le nombre de films distincts recommandés par utilisateur.
— Le pourcentage de films sans évaluation (i.e., n'ayant aucune note) recommandés.
— La diversité des genres des films recommandés.

Par ailleurs, afin de comparer les performances des systèmes de recommandation à froid selon la méthode

de réduction de dimension utilisée, nous avons également analysé les temps d'initialisation, d'entraînement et de réponse du système. Pour effectuer ces mesures, visibles dans les tableaux « Résultats expérimentaux du système de recommandation à froid sans bruit » (Tableau 1) et « Résultats expérimentaux du système de recommandation à froid avec bruit » (Tableau 2), nous avons utilisé une même graine aléatoire pour la génération des clusters et réalisé 609 tests pour chaque configuration de réduction de dimension avec une demande recommandation de 20 films. Ces deux tableaux présentent les métriques suivantes :

- La dimension des vecteurs après réduction de dimension : **k**
- Le nombre de clusters différents présents dans les recommandations pour un utilisateur : **Clusters**
- Le ratio des genres recommandés par rapport à ceux présents dans la base de données : **Genre Div.**
- Le ratio des genres recommandés par rapport à ceux effectivement recommandables : **Inner Div.**
- Le ratio de films peu notés (n'ayant aucune note dans notre implémentation) parmi les films recommandés : **0 Cov.**
- Le taux de couverture des recommandations sur les films éligibles : **Cov.**
- Le nombre moyen de films distincts recommandés par utilisateur : **Mov./User**
- Le nombre moyen de clusters distincts recommandés par utilisateur : **Clust/User**
- Le nombre moyen de films recommandés ayant peu de notes : **0 Mov./Reco**
- Le temps de réponse moyen du système de recommandation : **Resp. Time**
- Le temps moyen d'initialisation du système : **Init.**
- Le temps moyen d'entraînement du système : **Train.**

A partir des résultats obtenus, nous avons pu observer que les approches de réduction de dimension permettent une meilleure séparation des films, avec une augmentation significative du nombre de clusters, passant de 2 (sans réduction) à environ 10 en moyenne (avec réduction), une meilleure représentation de la diversité des genres recommandés, de la diversité des films peu noté recommandés ainsi qu'une augmentation du nombre de films peu noté recommandé, d'environ respectivement 28% , 352% et de 58% pour les données bruité avec réduction par rapport à celles sans réduction.
. En revanche, l'approche sans réduction semble permettre une meilleure diversité de recommandations des films, avec une représentation d'environ 73% en moyenne des films recommandables contre environ 22% avec la réduction de dimension en moyenne (colonne « Cov. »). Cette augmentation de la couverture des films impacte aussi la redondance des recommandations, avec un nombre moyen de films distinctsrecommandés par utilisateur passant de 8 (sans réduction) à environ 2 (avec réduction), visible au travers de la colonne « Mov./User ».
De plus, le temps d'entraînement, bien que variable, augmente avec la réduction de dimension (environ 147 secondes en moyenne avec bruit, contre 101 secondes sans réduction). Les temps d'initialisation et de réponse restent très faibles dans tous les cas. Ainsi, afin de favoriser au maximum la représentation de la diversité de la base de données, nous nous sommes concentrés sur les approches avec une réduction de dimensions, car elles semblent offrir un meilleur compromis entre la diversité des recommandations et la performance du système.

Nous constatons alors que les performances du système de recommandation à froid sont relativement stables entre les données bruitées, avec un budget de confidentialité fixé à $b = 0.5$. Dans la suite, nous nous concentrons sur les résultats obtenus avec bruit, car ce sont les données qui sont utilisées dans le système pour les recommandations.
Le tableau « Résultats expérimentaux du système de recommandation à froid avec bruit » (Tableau 2) confirme la stabilité des performances du système, comme en témoignent les faibles variances des métriques analysées.

Globalement, le système montre une certaine redondance dans ses recommandations, avec un nombre moyen de films distincts recommandés d'environ 2 par utilisateur pour 20 films. En revanche, il parvient à une diversité efficace de genres, avec une couverture moyenne d'environ 60% des genres présents dans la base de données. La représentation des clusters est également satisfaisante, avec en moyenne 9 à 11 clusters différents représentés dans les recommandations, soit une couverture proche des 10 clusters moyen extraits lors du clustering.
Enfin, environ 23% des films recommandés n'ont reçu aucune évaluation, ce qui équivaut à environ 4 films peu notés par recommandation, ce qui montre donc la mise en avant de contenus peu explorés.

3.2 Limites et perspectives

3.2.1 Système de recommandation à chaud

Malgré ses performances encourageantes, le système de recommandation à chaud présente plusieurs limites inhérentes à sa nature et à son fonctionnement. Tout d'abord, la dépendance aux données récentes peut entraîner une instabilité des recommandations lorsque les préférences des utilisateurs évoluent rapidement ou de manière erratique. En effet, le système s'appuie essentiellement sur les interactions les plus récentes, ce qui peut conduire à une surexposition d'éléments populaires temporairement, au détriment de la diversité et de la découverte. Ensuite, la méthode SVD utilisée dans ce contexte, bien que performante, repose sur des hypothèses linéaires et peut rencontrer des difficultés à modéliser des interactions complexes ou non linéaires entre utilisateurs et items. Cela limite potentiellement la capacité du système à capturer certaines nuances des préférences des utilisateurs. De plus, les systèmes à chaud exigent une mise à jour rapide et fréquente des modèles pour rester pertinents, ce qui peut engendrer une charge computationnelle non négligeable dans des environnements à très grande échelle ou en présence de flux de données massifs. Enfin, la qualité des recommandations reste fortement dépendante de la quantité et de la qualité des données disponibles. En cas de démarrage avec peu de données (problème du démarrage à froid) ou en présence d'utilisateurs peu actifs, les performances du système peuvent fortement diminuer.

Ces limites invitent à envisager des améliorations possibles, comme l'intégration de modèles hybrides, l'incorporation de filtres basés sur le contenu, ou encore l'optimisation des stratégies de mise à jour afin d'équilibrer précision, diversité et performance.

3.2.2 Système de recommandation à froid

Le système de recommandations à froid mis en place dans ce projet présente plusieurs limites. Tout d'abord, il repose sur la sélection aléatoire de films au sein des clusters. Ainsi, lorsque le nombre de films à recommander augmente, la probabilité que certain clusters ne soient pas sélectionnés diminue.
Pour répondre aux demandes de recommandations en petit nombre tout en assurant une certaine diversité, il serait pertinent de sélectionner un film par cluster lors des premières suggestions, en suivant des règles spécifiques. Par exemple, on pourrait privilégier les clusters les plus importants. Cela permettrait de garantir une représentation minimale de l'ensemble de la base de données tout en assurant la diversité des recommandations.

Ensuite, comme mentionné précédemment, le système de recommandations à froid repose sur les caractéristiques des films représentés sous forme de vecteurs. Toutefois, afin de la spécialiser à une base de données spécifique d'utilisateurs, il serait intéressant d'y intégrer des statistiques globales telles que le nombre d'évaluations, une approximation de la note moyenne et la variance des notes pour chaque film. Cette approche pourrait enrichir la représentation des films en tenant compte de leur réception par les utilisateurs, et ainsi permettre une recommandation plus pertinente.

Enfin, approfondir l'analyse des performances du système de recommandations à froid en augmentant le budget de confidentialité pourrait permettre de mieux évaluer l'impact du bruit sur la diversité des recommandations ainsi que sur la précision des recommandations.

3.2.3 Mécanisme de Laplace

Comme nous l'avons vu dans la section 2.1.2, il est possible de contrôler la génération du bruit dans le mécanisme de Laplace. Toutefois, celle-ci est indépendante du nombre de données à protéger pour un film donné. Un bruit important tend à avantager les films ayant un grand nombre d'évaluations, tout en défavorisant ceux qui en ont peu, rendant alors difficile l'extraction d'informations statistiques pertinentes. Par exemple, en observant l'écart moyen du bruit généré par l'algorithme 1 (présenté en annexe), avec un budget de confidentialité $b = \frac{-1}{\ln(0.9)}$, on peut obtenir environ 6,5 points d'écart par rapport à zéro pour 10 évaluations, contre seulement -0,1 pour 10 000 évaluations. Au contraire, en prenant $b = \frac{-0.5}{\ln(0.2)}$, on peut obtenir environ 0.5 point d'écart par rapport à zéro pour 10 évaluations, contre seulement -0.0005 pour 10 000 évaluations

Dans ce projet, notre approche repose sur l'utilisation d'un budget de confidentialité faible, afin de garantir une étude statistique fidèle aux données réelles, quel que soit le nombre d'évaluations d'un film. Cette stratégie est adaptée à une base de données avec peu d'évaluations, comme MovieLens, où le nombre maximal de notes pour un film est de 329. En revanche, dans une base comportant un volume beaucoup plus important, un bruit trop faible pourrait permettre une extraction probabiliste des données sensibles.

Dans cette optique, il serait intéressant d'envisager la mise en place d'un budget de confidentialité variable, ajusté en fonction du nombre d'évaluations par film et d'étudier la compatibilité d'une telle approche avec les systèmes de recommandation.

3.2.4 Mécanisme exponentiel

Dans ce projet, le mécanisme exponentiel a été mis en place afin d'ajouter un niveau de confidentialité côté interface utilisateur, limitant ainsi le risque que les habitudes de consommation d'autres utilisateurs puissent être déduites. Toutefois, l'implémentation de ce mécanisme augmente le temps de génération des recommandations, car il est nécessaire de recalculer les probabilités de sélection pour chaque film à chaque itération.

Ainsi, il serait intéressant d'étudier des optimisations du mécanisme exponentiel permettant de réduire ce temps de calcul tout en conservant les garanties de confidentialité. Une autre piste serait l'implémentation d'un système de préchargement des recommandations, afin de limiter le temps de réponse côté interface utilisateur.

3.3 Résumé

Dans ce projet, nous avons tenté de mettre en place un système de recommandation de films tout en respectant la confidentialité des données des utilisateurs. Pour la recommandation de films, nous avons opté pour deux approches complémentaires : un système de recommandation pour les utilisateurs connus du système, basé sur la décomposition en valeurs singulières (SVD), et un système de recommandation pour les utilisateurs inconnus du système, basé sur le clustering non supervisé.
Pour la confidentialité des données, nous avons choisi la mise en place manuelle du mécanisme de Laplace, permettant un meilleur contrôle de l'ajout du bruit, ainsi que l'utilisation d'un mécanisme exponentiel pour la sélection aléatoire de films recommandés pour les recommandations à froid. L'objectif du mécanisme exponentiel est de garantir la confidentialité des données des utilisateurs en limitant le risque que les habitudes de consommation d'autres utilisateurs puissent être déduites.

L'ensemble de données sur lequel nous avons travaillé est un jeu de données MovieLens, qui contient des informations sur les films ainsi que les évaluations des utilisateurs, soit environ 10 000 films et 100 000 évaluations. Toutefois, le nombre d'évaluations par film est relativement faible, avec un maximum de 329 évaluations pour un film. Ce faisant, nous avons choisi de mettre en place un budget de confidentialité faible pour garantir une étude statistique fidèle aux données réelles. Ici, nous avons choisi un budget de confidentialité de 0.5, signifiant, d'après la propriété 2 de la section 2.1.2 , que la probabilité que le bruit généré dépasse 0.5 est d'environ 0.368.

Avec ce budget de confidentialité, nous avons pu voir que le système à chaud atteint une précision de 72,2% et un rappel de 69,1%, avec un temps de réponse moyen de 0.03 seconde. Le système de recommandation à froid, quant à lui, permet de représenter en moyenne 9 à 11 clusters différents, avec une représentation des genres de la base de données d'environ 60% tout en recommandant en moyenne 22.6% de films sans évaluation. De plus, le système de recommandation à froid permet d'atténuer la redondance des films en recommandant en moyenne 2 films distincts par utilisateur.

Toutefois, certaines améliorations peuvent être envisagées comme la spécialisation de la représentation des films dans une base de données en intégrant des statistiques globales relatives à celle-ci, l'optimisation du mécanisme exponentiel pour réduire le temps de réponse ou encore l'implémentation d'un budget de confidentialité variable pour le mécanisme de Laplace.

Références

[1] J.-P. Rolland, « La technique de la réponse aléatoire : un moyen de contrôler la désirabilité sociale dans la mesure de l'estime de soi », *Bulletin de Psychologie*, vol. 33, no. 343, 1979. [En ligne]. Disponible : https://www.persee.fr/doc/bupsy_0007-4403_1979_num_33_343_1128 [Consulté le 28 mai 2025].

[2] A. I. Schein, A. Popescul, L. H. Ungar, et D. M. Pennock, « Methods and Metrics for Cold-Start recommandations », in *Proc. 25th Annu. Int. ACM SIGIR Conf. Res. Dev. Inf. Retr. (SIGIR '02)*, 2002, pp. 253–260. [En ligne]. Disponible : https://www.researchgate.net/publication/221300490_Methods_and_Metrics_for_Cold-Start_recommandations [Consulté le 28 mai 2025].

[3] Wikipedia, « K-anonymisation », 2006. [En ligne]. Disponible : https://fr.wikipedia.org/wiki/K-anonymisation [Consulté le 28 mai 2025].

[4] X. Su et T. M. Khoshgoftaar, « A Survey of Collaborative Filtering Techniques », *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 1, pp. 107–117, 2010. [En ligne]. Disponible : https://pages.stern.nyu.edu/~atuzhili/pdf/TKDE-Paper-as-Printed.pdf [Consulté le 28 mai 2025].

[5] C. Dwork et A. Roth, *The Algorithmic Foundations of Differential Privacy*, Found. Trends Theor. Comput. Sci., vol. 9, no. 3–4, pp. 211–407, 2014. [En ligne]. Disponible : https://www.cis.upenn.edu/~aaroth/Papers/privacybook.pdf [Consulté le 28 mai 2025].

[6] I. Chillotti, N. Gama, M. Georgieva, et M. Izabachène, « Faster Fully Homomorphic Encryption : Bootstrapping in Less Than 0.1 Seconds », *Cryptology ePrint Archive*, Rapport 2016/870, 2016. [En ligne]. Disponible : https://eprint.iacr.org/2016/870 [Consulté le 28 mai 2025].

[7] E. Dabbas, « Delving Deeper into Recommender Systems : From Basics to State-of-the-Art », *Medium*, 2018. [En ligne]. Disponible : https://medium.com/@eliasah/delving-deeper-into-recommander-systems-from-basics-to-state-of-the-art-d92ee8e277f2 [Consulté le 4 juin 2025].

[8] G. Snider, « Unsupervised Product Clustering : Exploring the Cold Start Problem », *SSENSE Tech Blog*, Medium, 2019. [En ligne]. Disponible : https://medium.com/ssense-tech/unsupervised-product-clustering-exploring-the-cold-start-problem-8053ef04bac9 [Consulté le 28 mai 2025].

[9] E. Azar, « Deep Dive into Matrix Factorization for recommander Systems : From Basics to Implementation », Medium, 2020. [En ligne]. Disponible : https://medium.com/@eliasah/deep-dive-into-matrix-factorization-for-recommander-systems-from-basics-to-implementation-79e4f [Consulté le 28 mai 2025].

[10] E. Goblé et R. Gatelier, « Le chiffrement DGHV », Université de Technologie de Compiègne. [En ligne]. Disponible : https://www.utc.fr/~wschon/sr06/tx_chiffrement_homomorphe/pages/dghv.html [Consulté le 28 mai 2025].

[11] E. Goblé et R. Gatelier, « GSW », Université de Technologie de Compiègne. [En ligne]. Disponible : https://www.utc.fr/~wschon/sr06/tx_chiffrement_homomorphe/pages/vecteurs_propres_approx.html [Consulté le 28 mai 2025].

[12] E. Goblé et R. Gatelier, « Le Bootstrap », Université de Technologie de Compiègne. [En ligne]. Disponible : https://www.utc.fr/~wschon/sr06/tx_chiffrement_homomorphe/pages/bootstrapping.html [Consulté le 28 mai 2025].

[13] *Programming Differential Privacy*, « Chapter 9 — Fully Homomorphic Encryption (FHE) ». [En ligne]. Disponible : https://programming-dp.com/ch9.html [Consulté le 28 mai 2025].

[14] B. Chandran, A. Ghoshal, S. Halevi, S. Ranellucci, et N. P. Smart, « Faster Homomorphic Comparison Operations for BGV and BFV », *Cryptology ePrint Archive*, Rapport 2023/958, 2023. [En ligne]. Disponible : https://eprint.iacr.org/2023/958 [Consulté le 28 mai 2025].

[15] Q. Nguyen, M. Naghiaei, et Y. Zhang, « Cold-start recommandation by Personalized Embedding Region Elicitation », *arXiv preprint* arXiv :2406.00973, 2024. [En ligne]. Disponible : https://arxiv.org/abs/2406.00973 [Consulté le 28 mai 2025].

Annexes

Listing 1 – Fonction de génération de bruit Laplacien

```
def sign(x):
    if x >= 0:
        return 1
    elif x < 0:
        return -1

def laplace(b):
    x = random.uniform(-0.5, 0.5)
    return b * sign(x) * math.log(1 - 2 * abs(x))

def average_noise(b, n_test):
    total_noise = 0
    for _ in range(n_test):
        total_noise += laplace(b)
    return total_noise / n_test
```

TABLE 1 – Résultats expérimentaux du système de recommandation à froid **sans bruit** ($b = 0.5$, 20 films recommandés, 609 tests par configuration)

k	Clusters	Genre Div.	Inner Div.	0 Cov.	Cov.	Mov./User	Clust/User	0 Mov./Reco	Resp. Time (s)	Init (s)	Train (s)
sans reduction	2	0.456	0.456	0.0025	0.703	8.057	2.0	0.044	0.0267	0.032	94.74
50	11	0.602	0.602	0.0104	0.206	2.363	9.422	0.187	0.0065	0.030	157.94
100	10	0.602	0.602	0.0117	0.215	2.471	8.754	0.210	0.0071	0.016	157.40
150	10	0.609	0.609	0.0130	0.230	2.634	8.874	0.235	0.0069	0.027	180.02
200	12	0.575	0.575	0.0115	0.213	2.445	9.903	0.207	0.0075	0.018	189.72
250	11	0.607	0.607	0.0120	0.230	2.634	9.368	0.217	0.0056	0.016	187.13
300	11	0.565	0.565	0.0109	0.221	2.534	9.386	0.197	0.0054	0.015	201.89
350	11	0.589	0.589	0.0074	0.189	2.163	9.455	0.133	0.0053	0.016	204.99
400	10	0.578	0.578	0.0133	0.209	2.402	8.824	0.240	0.0071	0.018	223.61
Moyenne	10.75	0.591	0.591	0.0113	0.214	2.456	9.248	0.190	0.0064	0.019	187.83
Variance	0.73	0.002	0.002	0.00004	0.0003	0.036	0.38	0.0011	0.0000007	0.00003	569.77

TABLE 2 – Résultats expérimentaux du système de recommandation à froid **avec bruit** ($b = 0.5$, 20 films recommandés, 609 tests par configuration)

k	Clusters	Genre Div.	Inner Div.	0 Cov.	Cov.	Mov./User	Clust/User	0 Mov./Reco	Resp. Time (s)	Init (s)	Train (s)
sans reduction	2	0.466	0.466	0.0018	0.762	7.936	2.0	0.033	0.0325	0.031	100.69
50	10	0.603	0.603	0.0140	0.245	2.550	8.811	0.251	0.0042	0.014	111.17
100	10	0.613	0.613	0.0119	0.248	2.583	8.824	0.213	0.0051	0.019	124.10
150	10	0.604	0.604	0.0146	0.247	2.570	8.810	0.263	0.0050	0.014	125.55
200	9	0.585	0.585	0.0136	0.256	2.667	8.209	0.245	0.0052	0.014	132.46
250	10	0.598	0.598	0.0109	0.237	2.468	8.796	0.195	0.0055	0.013	170.78
300	9	0.585	0.585	0.0135	0.245	2.552	8.141	0.243	0.0056	0.014	160.96
350	12	0.607	0.607	0.0100	0.222	2.314	9.888	0.181	0.0042	0.013	166.62
400	11	0.593	0.593	0.0121	0.218	2.271	9.450	0.218	0.0049	0.013	179.57
Moyenne	10.13	0.598	0.598	0.0128	0.240	2.497	8.854	0.226	0.0050	0.014	146.90
Variance	0.99	0.001	0.001	0.000002	0.0002	0.021	0.38	0.0008	0.0000002	0.000004	682.13