A black background with a black square

Description automatically generated with medium confidence

**Faculté des Sciences**

**Département d’Informatique**

**IFT799 - Science des Données**

**RAPPORT - TP 3 :**

***Systèmes de recommandation***

**Enseignants :**

Shengrui Wang

Etienne G. Tajeuna

**Auteurs** :

Abdoul Madjid SANOUSSI LABO **(**CIP : SANA2909**)**

Abdou Rahime DAOUDA (CIP: DAOA2504)

Mohamed BOUBACAR BOUREIMA (CIP: BOUM3688)

Table des Matières

[Table des Matières 1](#_Toc154342892)

[**Présentation** 2](#_Toc154342893)

[**Exploration et statistiques descriptives de données** 2](#_Toc154342894)

[**Question 1 : Construction du diagramme en bâton illustrant le nombre de films que l’on a par genre** 4](#_Toc154342895)

[**Question 2 : Extraction des nouveaux jeux de données movies1.csv et ratings1.csv** 5](#_Toc154342896)

[**Question 3 : Construction de la Matrice binaire de contenu** 7](#_Toc154342897)

[**Question 4 : Construction de la matrice de profil des utilisateurs (P)** 7](#_Toc154342898)

[**Question 5 : Clustering spectral pour limiter la recherche d’items et d’utilisateurs** 9](#_Toc154342899)

[**Question 6 : Conception du model de système de recommandation** 14](#_Toc154342900)

[**a.** **Entrainement et évaluation du modèle de prédiction des votes des utilisateurs :** 15](#_Toc154342901)

[**b.** **Prédiction et évaluation du fichier ratings test.csv :** 23](#_Toc154342902)

[**Conclusion** 25](#_Toc154342903)

# **Présentation**

Le présent document a été rédigé dans le cadre de l’exploration du thème « Sciences de Données ». Ce document présente la mise en pratique de l’essentiel sur la compréhension, des systèmes de recommandations et les autres modules du thème. Notre rapport se base sur les matériaux présentés en cours et les documents qui ont été mis à notre disposition.

# **Exploration et statistiques descriptives de données**

L'exploration et les statistiques descriptives sont des étapes cruciales pour mieux comprendre nos données avant de prendre des décisions plus avancées concernant notre analyse. Elles nous permettent également de repérer d'éventuelles anomalies ou problèmes dans les données.

Nous disposons de deux jeux de données dont **ratings.csv,** contenant 24973468 de votes de 162541 utilisateurs sur 62423 films avec chacun d’utilisateur, ayant évaluée au moins 20 films et **movies.csv** contenant les informations sur les 62423 films et dont la principale variable nous intéressant, c’est l’ensemble des genres de chaque film. Quelques statistiques descriptive et informations sur les jeux de données, se présentent comme suit : (Voir notre fichier *Code – TP3 IFT799-Equipe 2.ipynb*)

* Pour **Movie.csv** :

A screenshot of a movie list

Description automatically generated

D’autres part, on s’est intéressé à la présence des valeurs manquantes, et on n’en trouve aucune présente sur le jeu de donnée. Ce qui est plutôt intéressant pour l’analyse qui suit.

A screen shot of a computer

Description automatically generatedA black and white text

Description automatically generated

* Pour **ratings.csv :**

A screenshot of a computer screen

Description automatically generated A table with numbers and symbols

Description automatically generated

Ratings a 4 variables, dont deux identificatrices (Utilisateur et Film), une catégorielle qui représente les votes donnés par chaque utilisateur sur chaque film et une date correspondante à celle ou le vote a été fait, en format timestamp. Quelques mesures descriptives sur les variables présentes, l’image en haut a droite, mais ne présentant aucun sens pour nous, car aucune n’est quantitative numérique.

On s’est également intéressé à la présence des valeurs manquantes, et on n’en trouve aucune présente sur le jeu de donnée Ratings.

A screenshot of a computer

Description automatically generated

A black text with black letters

Description automatically generated with medium confidence

Pour les variables timestamp et rating, nous cherchons à voir la distribution de celles-ci ainsi que la présence potentielle de valeur aberrantes.

A blue rectangular object with black lines

Description automatically generated A blue rectangular object with black lines

Description automatically generated

Distribution symétrique pour la variable, le timestamp (les dates des votes) et non symétrique pour les ratings, ce qui conclut la présence des données non balancées. Pour les valeurs aberrantes, nous émettons l’hypothèse, qu’aucune valeur aberrante n’est présente.

Il serait également important de connaitre la distribution des classes présente, tout autant que leur nombre. Le camembert ci-dessous donne un aperçu sur cette connaissance des étiquettes.

Nous verrons dans les prochaines étapes, les prétraitements requis pour la suite de notre modélisation du système de recommandation et aussi les statistiques descriptives qu’on aurait manqué dans cette section.

# **Question 1 : Construction du diagramme en bâton illustrant le nombre de films que l’on a par genre**

Pour répondre à cette question, il suffisait de créer un dictionnaire dont la clé c’est le genre et la valeur, c’est le nombre de fois qu’on rencontre le genre, lors du parcours de l’ensemble du jeu de donnée film.

Nous soulignons ici que les films au genre non listés: ‘ (no genres listed) ’ seront ignoré lors du comptage. Nous obtenons ainsi le diagramme demandé comme suit :

A graph with blue bars

Description automatically generated

Sur cette image, nous constatons la présence de 19 genres distincts globalement et que les genres majoritaires sont le **Drama** et la **Comédie** et ceux minoritaire sont **IMAX** et **Film-Noir**.

Autre remarque, il serait possible de séparer le goût des utilisateurs en deux grandes catégories. Ceux positives (Ramance, Comedy,…) et ceux négatives (Horror, Crime, War).

# **Question 2 : Extraction des nouveaux jeux de données movies1.csv et ratings1.csv**

Il est demandé ici de créer deux jeux de données, issus d’un prétraitement sur les jeux de données initiaux.

A screenshot of a movie list

Description automatically generated

* Movies1.csv : En supprimant, de movies.csv, toutes lignes dont l’identifiant du film est non listé. En sortie, on a un nouveau jeu avec 57361 films, décrit par le tableau suivant :
* A table with numbers and a text

  Description automatically generatedRatings1.csv : En changeant les ratings de valeurs 5.5, 4.5, 3.5, 2.5, 1.5, 0.5 par les valeurs respectives de 5, 4, 3, 2, 1, 1 et également, tous les votes des utilisateurs dont le film est sans genre, seront supprimés. On a en sortie le nouveau jeu avec 24973468 votes au total.

A pie chart with numbers and a number of percentages

Description automatically generatedPour mieux voir la nouvelle distribution de la variable rating, nous décidons de tracer la proportion des votes, via ce diagramme de camembert :

Les votes sont asymétriquement distribués toujours. Un jeu de données note balancé avec le rating 4 est majoritairement présent dans notre jeu de donnée (35,4%), et inversement Rate 1, qui est minoritaire (6.3%). Les données ne sont pas balancées, une classe pourrait être discriminée, lors de sa prédiction, basé sur un modèle conçu sur cette base de données.

Le code ayant servi à ce prétraitement, donnera plus de détails sur le procède. Les fichiers résultants sont enregistrés et seront envoyés, avec ce rapport, tel que demandé par la consigne.

# **Question 3 : Construction de la Matrice binaire de contenu**

A base des 19 genres associés à chaque film, nous coconstruisons la matrice binaire de contenu C caractérisant chaque film. Une matrice a 57361 lignes de film et 19 colonnes, une pour chaque caractéristique, dont les valeurs sont 1 si le film présente ce genre et 0 sinon. Une colonne additionnelle pour représenter l’identifiant du film en question (movieId). Ci-dessous un aperçu de la matrice résultante :

A screenshot of a table

Description automatically generatedA screenshot of a number

Description automatically generated

Pour voir le détail de la façon d’extraire ce contenu, veiller consulter le fichier code de notre TP.

# **Question 4 : Construction de la matrice de profil des utilisateurs (P)**

Notre système de recommandation est un système mixte, basé le contenu et le collaboratif. Pour pouvoir exploité les deux procédures, on devrait avoir en entrer du modèle de recommandation, des données qui prennent en compte le contenu d’un côté, ici la matrice binaire des genres, et les votes des utilisateurs de l’autre côté. La matrice de profil, fait effectivement le compromis entre ces deux utilisateurs, en procurant un certain score d’intérêt à un utilisateur donné, pour les groupes de caractéristique donnés.

Cette matrice est donnée par la formule suivante :

A black and white symbols

Description automatically generated

* Uu est l’utilisateur considéré et Iu l’ensemble des films qu’il a regardé.
* ru,i , Le rating donné par Uu sur Iu
* Ci, le vecteur binaire caractérisant le film Ii. (La ligne i de votre matrice C).

Est en détails, l’implémentation de ladite matrice, dans le fichier code du TP.

Un aperçu de la matrice de profil :

A table with numbers and letters

Description automatically generatedA screenshot of a graph

Description automatically generated

Nous avons également accompagné la matrice par userId, pour pouvoir identifié l’utilisateur cherché, pour des raisons futures.

Une interprétation plausible, pour le **profil de l’utilisateur 1**, est qu’il s’intéresse plus aux films de genre **Mystery** (avec un score de 196) et absolument pas aux films de genre **Film-Noir** (0 comme score). Mieux, on pourrait avouer qu’il n’en a jamais regardé, car aucun vote sur celui-ci).

Un **pairplot** des caractéristiques, révèle l’existence de certains genres, corrélé entres eux. C’est le cas par exemple de **Horror** et **War** ou encore **Musical et Fantasy**.

A grid of blue dots

Description automatically generated

# **Question 5 : Clustering spectral pour limiter la recherche d’items et d’utilisateurs**

Ici on voudrait trouver les groupes d’utilisateurs et d’items présentant les mêmes caractéristiques. En utilisant la matrice de profil P. Testons plusieurs valeurs de K = {2, 3, 4, 5} pour performer un clustering spectral, à k clusters et retenons celle dont le score de silhouette est le plus élevé.

Pour ce faire, nous avons effectué deux méthodes différentes, la seconde pour remédier à la problématique des ressources qu’on a rencontré lors de la première.

* **En utilisant la fonction SpectralClustering de sk-learn :**

Nous testons le modèle sur les 4 valeurs de k (2,3,4,5). Pour chacune, nous évaluons le score silhouette, pour comparaison. Toutefois, nos ressource étant limitées, il nous a été impossible d’avoir un résultat en utilisant tout d’un coup la matrice de profil. L’algorithme mettant du temps à exécuter. Nous décidons alors d’utiliser une matrice réduite des 10000 premiers utilisateurs pour espérer obtenir les clusters optimaux.

Ci-dessous les scatterplots obtenues sur une projection UMAP, pour les 4 clusters coloriés, ainsi que la courbe associée au score silhouette pour les 4 valeurs de k.

A graph with a blue line

Description automatically generated

Pour k = 2 clusters, on obtient le meilleur score, quoi que la division ne soit pas équitable. Toutefois, on pourrait être en possession des données non balancées. Et rappelons-nous l’hypothèse selon laquelle on a deux catégories de caractéristique (positive et négative). Drama et Comedy sont les plus majoritaires et peuvent être vu comme étant des genres positives. S’ajoute le fait que si les utilisateurs préfèrent ces genres majoritaires.

A group of images of a map

Description automatically generated

Ainsi notre clustering semble logique, toutefois serait-il possible de généraliser sur les 150000 autres utilisateurs ? Sans aucune idée sur leur distribution? D’où l’utilité de la prochaine méthode que nous implémentons.

* **Procédure ‘Traditionnelle’ ACP + K-means :**

Ici, nous performons une **ACP** sur l’ensembles des données initiales, pour choisir un nombre de composante principale limité, qui représenterons le maximum possible de la variance de nos données. Puis faire un K-means sur les 4 composantes principales.

Le choix du nombre de composante est fait selon le critère de coude sur la variance expliquée.

A graph with a line

Description automatically generated A table with numbers and symbols

Description automatically generated

Les 4 premières composantes, représentent à elles seules 97,4% d'information. Nous n'en garderons qu'elle seule pour effectuer notre clustering K-Means. Les 4 k résultants accompagnés par le score silhouette sont comme suit :

A graph with a line

Description automatically generated

A group of images of a horseshoe

Description automatically generated

L'une ou l'autre des méthodes, k = 2 clusters, présente le plus grand score silhouette quoi que les données ne soient pas bien balancées. Nous allons nous en servir pour effectuer la classification des utilisateurs. La seconde méthode serait utilisée pour cette classification vu qu’elle prend en compte les utilisateurs dans leur entièreté.

Ensuite, nous effectuerons une assignation de la matrice de profil initiale, sur tous les utilisateurs, selon les clusters résultants du K-means sur la matrice de profil.

A red and blue dotted circle

Description automatically generated

A coffee bean with a brown liquid

Description automatically generated with medium confidence

A close up of numbers

Description automatically generatedNous obtenons, le scaterplot résultant des clusters selon les 3 projections, ACP, TSNE et UMAP. La forme de la projection t-SNE, nous révèle une forme, suffisamment séparée (graine qui pousse), pour valider le clustering avec k = 2.

# **Question 6 : Conception du model de système de recommandation**

On voudrait à présent modéliser notre système de recommandation. Pour cela nous préparons les données qui permettront de valider notre futur système de recommandation. Sachant que les votes varient de 1 à 5, on voudrait s’assurer que tous les cas de figures soient pris en compte lors de l’apprentissage. Pour ce faire, suivant le fichier ratings1.csv, on va subdiviser en trois fichiers distincts, ratings train.csv, ratings evaluation.csv et ratings test.csv. Le fichier ratings train.csv contient 60% des données de ratings1.csv tandis que ratings evaluation.csv et ratings test.csv doivent contenir 20% des données de ratings1.csv chacun. La subdivision se fait de manière aléatoire. Un petit aperçu des données issues de cette étape, qui est une forme d’évaluation hors ligne du modèle.

A screen shot of a computer

Description automatically generated

A white screen with black text

Description automatically generated

## **Entrainement et évaluation du modèle de prédiction des votes des utilisateurs :**

Entrainement d'un modèle d'arbre de décision, pour la prédiction du rating d'un utilisateur u selon le vote de ses 5 plus proches voisins en exploitant les clusters pour retrouver ces plus proches voisins.

**NB :** On testera plusieurs hyperparamètres pour notre arbre de décision et retenir celui qui nous donnerait les meilleurs résultats suivant le critère F1 score sur les données d’évaluation.

Les différentes étapes pour la conception du prédicteur, sont comme suit :

* **1. Conception du DataFrame contenant les ratings des 5 voisins :**

Premièrement, nous établissons une fonction qui identifie les 5 utilisateurs les plus proches d'un utilisateur donné (identifié par user\_id). Ces 5 utilisateurs sont cherchés dans la matrice de profil, en calculant la distance de l’utilisateur, avec le reste des utilisateurs. Cette fonction construit ensuite un DataFrame contenant les données de ces 5 utilisateurs les plus proches à partir de la matrice des évaluations. Cette matrice peut être extraite des données d'entraînement si l'objectif est d’entraîner le modèle, ou des données d'évaluation si l'on souhaite évaluer le modèle.

Lors du calcul des distances, nous prenons en compte le fait qu'un utilisateur n'aurait pas regardé un film. Cette information est déjà intégrée dans la matrice de profil, provenant de la matrice des contenus. Ainsi, il n'est pas nécessaire de filtrer les utilisateurs en fonction de leurs interactions avec des films spécifiques. Pour plus de détails sur la mise en œuvre de cette fonction, veuillez consulter le fichier de code associé. Un exemple de la fonction, appliqué sur l’utilisateur 1 et 3, appartenant au cluster 0 et 1 respectivement.

A close-up of a screen

Description automatically generated

A screenshot of a cell phone

Description automatically generated Pour l’utilisateur 1 A screenshot of a cell phone

Description automatically generated

* **2. Entraînement du modèle d'arbre de décision :**

Le modèle est ensuite entraîné en utilisant la méthode de validation croisée (cross-validation avec 5 groupes) pour évaluer ses performances sur différentes subdivisions des données. Une recherche de grille (grid search) est effectuée pour trouver les meilleurs hyperparamètres du modèle, optimisant ainsi sa capacité à prédire les votes. La grille est composée de :

A screenshot of a computer code

Description automatically generated

A close-up of a text

Description automatically generatedOn obtient pour U1 :

Les meilleurs hyperparamètres suivant :

* **3. Évaluation des meilleurs hyperparamètres sur les données d'évaluation :**

Les meilleurs hyperparamètres identifiés précédemment sont utilisés pour prédire les évaluations des 5 utilisateurs les plus proches dans le jeu de données d'évaluation. Le score F1 est calculé pour évaluer la précision des prédictions par rapport aux évaluations réelles. Dans cette étape, on ne s’intéresse qu’à la prédiction, des votes des films déjà regardés, pour l’utilisateur en entré. Le vote des nouveaux films se fera ultérieurement. Obtient comme sortie :

* Un F1-score **de 0.281.**
* Et une prédiction des films regards comme suit :

A black numbers on a white background

Description automatically generated A number with numbers and letters

Description automatically generated with medium confidence A number with numbers on it

Description automatically generated with medium confidence

* **4. Retour du meilleur modèle et Répétition des étapes pour tous les autres utilisateurs :**

Le modèle d'arbre de décision optimal, ayant démontré les meilleures performances pour l’utilisateur 1, sur les données d'entraînement et d'évaluation, est retourné. Les étapes 1,2,3 et 4 sont répétées pour chaque utilisateur, en identifiant les 5 utilisateurs les plus proches spécifiques à chaque utilisateur et optimisant les hyperparamètres

**A close-up of a number

Description automatically generated**

.

.

.

.

* **5. Calcul de la moyenne du F1-score sur l'ensemble des utilisateurs :**

Les scores F1 obtenus pour chaque utilisateur sont moyennés pour obtenir une mesure agrégée de la performance du modèle sur l'ensemble des utilisateurs.

Le modèle final, avec les hyperparamètres les plus couramment utilisés, est retourné comme le modèle optimal pour cette tâche particulière. Et celui-ci pourrait être utilisé pour prédire les votes d’un nouvel utilisateur. Pour ce faire, il conviendrait de passer l’utilisateur en question, avec l’ensemble de ses votes donnés aux films qu’il a regardé. Et la procédure de la prédiction, est identique que pour la conception du modèle.

**Problématique :**

Cependant, la même problématique de ressource que précédemment, survient à cette étape. Car reproduire pour chaque utilisateur les 5 étapes précédentes, ne serait-ce que le calcul de distance, prend du temps à plus forte raison l’entrainement de l’arbre de décision, sur le data des 5 utilisateurs.

**Remarque :** De plus, la fonction de classification utilisée, semble inappropriée dans notre cas, surtout vu l’inexistence des features intéressants, dans la matrice en entrée de l’arbre de décision. Juste l’identifiant du film et le timestamp qui n’est que la date ou le vote est fait. L’arbre de décision aurait du mal à classifier les votes, et même s’il arrive, ces résultats n’auront pas une très grande signification.

La consigne du TP, suggérer que les données en entrée seront les ratings des 5 plus proche voisins, mais l’on ne peut pas quand même utiliser la variable rating uniquement pour faire la prédiction du rating. Non plus l’inclure dans les feature du modèle d’entrainement, car celle-ci discriminera le modèle, et fera ressortir un F1-score de 1. Notre objectif étant de prédire les votes de l’utilisateur pour un film nouveau, on ne pourrait disposer de son vote, sur le film non regardé.

**Solution proposée :**

Pour réduire la complexité de l'algorithme, on s'en servira du clustering pour prendre uniquement un nombre restreint N dans chaque cluster, puis calculer la moyenne de leurs prédictions (la meilleure technique qu’on aurait due utilisée pour la fonction de classifieur, sur les 5 plus proches voisins), ensuite généraliser sur le cluster. Car plus les utilisateurs auront le même profil (même gout), plus ils auront tendance à faire le même vote sur le même film. Ceci n’est que dans l’objectif d’obtenir le meilleur modèle déjà entraîné et évalué, pour s’en servir de celui-ci plus tard. Notre solution, implique également la prédiction des film non regardés en plus de ceux regardés.

La concaténation pourrait se faire en par colonne également, mais l’algo sera plus complexe. Nous essayons d’éviter celle-ci, mais potentiellement, elle donnera un meilleur résultat sur l’arbre de décision.

A table of numbers and letters

Description automatically generated

Ci-dessous la procédure suivie, pour implémenter cette solution et quelques sorties de la fonction de prédiction.

* Recherche des N utilisateur dans les deux clusters distincts et prédiction de leur vote individuellement, sur les films regardés et non regardés.

On a en sortie ici, les meilleurs hyperparamètres pour chaque individu et les F1-scores :

**Cluster 0 :**

**A table with numbers and text

Description automatically generated**A graph with blue lines and numbers

Description automatically generated

A table with numbers and letters

Description automatically generated

Une valeur NaN, implique l’impossibilité de prédire le vote, car même les 5 plus proches utilisateur n’ont pas regardé le film en question.

**Cluster 1 :**

**A table with numbers and text

Description automatically generated**A graph with blue dots and numbers

Description automatically generated

* Ensuite on moyenne les F1-scores et on cherche les hyperparamètres populaires :

**Cluster 0 : Cluster 1 :**

**A white background with black text

Description automatically generated A white background with black text

Description automatically generated**

****

* Puis on moyenne également les votes résultants, pour ces N utilisateurs. Il faudrait ici souligner que le moyennage, permet d’obtenir les votes d’un utilisateur, même s’il n’a pas regardé le même film, et aussi ses plus proches utilisateurs. Il acquiert ce vote, de la part des autres membres de son cluster.

**Cluster 0 : Cluster 1 :**

**A screenshot of a ballot

Description automatically generated** **A screenshot of a cellphone

Description automatically generated**

* Enfin on généralise le vote sur l’ensemble des utilisateurs des 2 clusters dans les données initiales.

**A table with numbers and numbers

Description automatically generated**

La sortie finale de notre entrainement-validation, c’est le meilleur modèle accompagné de ses meilleurs hyperparamètres et du meilleur F1-score résultant. F1-score de 0.22, laisse croire que le modèle n’est pas suffisamment performant, du certainement à 2 raisons :

* + - Arbre de décision, inapproprié pour la prédiction.
    - La généralisation par moyennage appliquée, cause de manque de ressource.

On va toutefois s’en servir pour le tester sur les données tests et calculer le F1-score.

## **Prédiction et évaluation du fichier ratings test.csv :**

A partir du prédicteur conçu précédemment, D() nous prédisons les votes des utilisateurs qui se trouvent dans le fichier ratings test.csv, tout en évaluant la performance du modèle suivant le critère de F1-score. Ci-dessous un aperçu de la prédiction :

A screenshot of a table

Description automatically generated

Il faut noter ici, qu’on ne s’intéressera qu’au vote des films regardes, dans l’intention de pouvoir évaluer notre modèle. Ci-dessous la matrice de confusion associée ainsi que d’autre critères d’évaluation.

Le dataframe serait en accompagnement à ce rapport, ainsi que le fichier code.

A graph with numbers and a number

Description automatically generated with medium confidence

A screenshot of a graph

Description automatically generated



Nous terminons avec un f1-score faible de **0.32** sur cet ensemble test. Le modèle semble ne pas bien performer sur nos données test, tout comme sur les données d’entrainement, vu l’abstention de l’utilisation de certaines données qu’on a faite, en les remplaçant par une sorte de généralisation.

Il pourrait à présent être déployer en ligne, pour mieux évaluer ce dernier ci, en utilisant de nouvelles données des utilisateurs, et donc on pourrait actualiser le modèle de façon à améliorer son f1-score, en permanence.

# **Conclusion**

Ce rapport de TP sur les systèmes de recommandation a permis une exploration approfondie des données, ainsi que la mise en œuvre de diverses techniques et modèles pour la construction d'un système de recommandation. Nous avons réalisé des statistiques descriptives, extrait de nouveaux jeux de données, construit des matrices de contenu et de profil d'utilisateurs, et utilisé le clustering spectral pour limiter la recherche d'items et d'utilisateurs. L'entraînement et l'évaluation du modèle de prédiction des votes des utilisateurs ont également été abordés, avec une attention particulière portée à l'évaluation des performances à l'aide du F1-score.

Malgré les défis rencontrés, tels que la présence de données non balancées et l'identification de modèles sous-performant, notre analyse a permis de mettre en lumière des pistes d'amélioration pour le déploiement futur du système de recommandation. En effet, la prédiction et l'évaluation sur le fichier ratings test.csv ont révélé un F1-score de 0.32, indiquant une performance faible du modèle sur cet ensemble de test. Cela souligne la nécessité d'actualiser le modèle en permanence en utilisant de nouvelles données d'utilisateurs pour améliorer sa précision et son efficacité (évaluation en ligne).

En conclusion, ce rapport a été une opportunité d'appliquer les concepts et les techniques appris dans le cadre du cours de Sciences des Données, et a mis en lumière l'importance de l'exploration approfondie des données, de la modélisation précise et de l'évaluation rigoureuse des performances pour la construction de systèmes de recommandation efficaces.