People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research

IBN KHALDOUN UNIVERSITY OF TIARET

Dissertation

Presented to:

FACULTY OF MATHEMATICS AND COMPUTER SCIENCE
DEPARTMENT OF COMPUTER SCIENCE

in order to obtain the degree of:

MASTER

Specialty: software engineering

Presented by:

KHETTAF EL MENDIL TAYEB
KONTA IBRAHIM

On the theme:

Exploration of bipartite graph to improve recommendation task

Defended publicly on 12/06/2024 in Tiaret in front the jury composed of:

Mr. BOUDAA Boudjema	**MCA**	Tiaret University Ibn Khaldoun	Chairman
Mr. Gafour Yacine	**MCA**	Tiaret University Ibn Khaldoun	Examinator
Mr. KHARROUBI Sahraoui	**MCA**	Tiaret University Ibn Khaldoun	Supervisor

2023-2024

Thanks

We thank the great God for giving us health, courage and patience for the successful completion of this project.

Thanks will go to our **parents, brothers, sisters, friends, in a word, family,** for all the attention paid to us during the entire course.

We thank everyone **(professors, members of the administration and others)** of the computer science department without forgetting the **deanship, the student office (national and international) and others** who are responsible for pedagogy for the facilitation of our curriculum and active participation in improving our future.

Our thanks also go to our supervisor, the teacher “**Mr. KHARROUBI Sahraoui**”, for having agreed to direct this work, for having trusted us and given useful advice each time we proposed to orient ourselves differently during this memoir. We also thank him for his permanent presence and all his availability dedicated to supervising the dissertation. We also thank him for his encouragement during the hardest moments (failure to keep appointments, wasted time and others).

For his multiple advice and efforts to ensure quality work, his support, his skills and his foresight were invaluable.

We would like to sincerely thank the members of the jury for the interest they showed in our project by agreeing to examine the work and enrich it with their proposals.

Finally, we thank all those who directly or indirectly contributed to the accomplishment of this work.

This work is dedicated to everyone.

THANK YOU!!!!!!

Abstract

By using bipartite graphs in movie recommender systems, we can address the challenge of recommender system limitations (sparsity, scalability, cold start, etc.) to improve recommendation personalization. Traditionally, recommendation systems use methods such as collaborative filtering, content-based filtering, hybrid filtering and others to understand user preferences and suggest similar movies based on past ratings.

However, when a new user joins the system or when a new movie is added to the database, a problem arises. Bipartite graphs offer a promising solution to this problem. By representing users and movies as separate nodes in a bipartite graph, we can use similarity calculation algorithms (cosine similarity and others) to recommend movies to new users based on the preferences of similar users.

For example, by looking at past interactions between users and movies, we can identify communities of users who share similar tastes. We can then recommend movies to new users based on the preferences of similar users within those communities.

By integrating contextual information such as user ratings, movie genres, actors, directors, and release years, we can improve the relevance and personalization of movie recommendations. Using bipartite graphs, we can also efficiently manage new movies by linking them to users with similar preferences, even without direct ranking.

In conclusion, the use of bipartite graphs in movie recommend systems presents an innovative approach to overcoming the limitations of traditional recommend systems, improve the personalization of recommendations, and provide relevant suggestions even for new users and new movies.

Keywords: recommendation systems, bipartite graphs, cold start, personalization, user-movie interactions, collaborative filtering.

Tables of Contents

LISTES OF FIGURES

List of figures

LIST OF TABLES

GENERAL INTRODUCTION

General Introduction

In today's digital age, characterized by an abundance of information on various topics such as movies, books, music and news, the challenge of information overload is ever-present. The pressing need to quickly access relevant data collides with growing complexity which is particularly evident in organizations, selections and recommendations. In this context, contemporary research focuses on redefining access to information by integrating users into a global research model, aiming to provide information tailored to their specific needs, context and preferences.

Information retrieval (IR) thus explores the organization, storage and selection of information to meet user requirements. Recommendation systems, relating to information filtering, play a central role in selecting and presenting resources adapted to each user in a dynamic environment. This memory is particularly dedicated to the exploration of a bipartite graph in order to improve the recommendation task.

This dissertation explores an innovative approach in the field of recommendation systems, aiming to exploit the subtle relationships between users and elements to improve the precision and relevance of recommendations. For this, it is divided into four distinct chapters.

The first chapter presents the fundamental concepts of recommendation systems, providing a state of the art in the field. It explores the different techniques and approaches used to effectively recommend items to users.

The second chapter focuses on the modeling of recommendation systems using bipartite graphs. It examines how these graphs can represent relationships between users and items effectively, making it easier to recommend relevant items.

The third chapter details specific movie recommendation techniques. It explores how bipartite graphs can be used to recommend films similar to those enjoyed by a user, by exploiting relationships between films, directors, actors, genres, etc.

Finally, the last chapter is devoted to the practical implementation of a recommendation system based on similarity between neighbors. It presents how this innovative approach can be implemented in the context of recommendation systems, using concrete examples to illustrate the different facets of graph bipartitioning and its application in movie recommendation.

Chapter 1

State of the art in recommendation systems

1. Introduction

The evolution of recommendation systems has traveled a fascinating path from early attempts in the 1970s to today's sophistication of artificial intelligence. These systems play a central role in our digital lives, facilitating the discovery of relevant products, services and content in a context of information overload. The growing importance of automatic recommendations has prompted intensive research, exploring various approaches to meet user needs and improve personalization. In this chapter, we started with the journey or the first attempts of these systems through their history, context without forgetting the problems that we can encounter in this field, we will also focus on the different types of recommendation systems. Thus, we expose the basic architecture of each type of system while showing how it works. Finally, target the limits of each category of these systems to better address the problem and therefore seek appropriate solutions.

2. History

The era began with the Grundy Library System in 1979, an early attempt at classifying users for book recommendations. In the 1990s, the work of Breese and Heckerman laid the foundation for collaborative filtering, an approach that recommends content based on the preferences of similar users.

The Tapestry system in 1992 marks the use of collaborative filtering to recommend documents from newsgroups. GroupLens, founded in 1994, also uses this approach to identify articles likely to be of interest to users, marking the development of automatic recommendation systems.

Innovation in the 2000s saw the emergence of advanced techniques, including content-based filtering. Systems like Ringo and Bellcore in 1995 provide music and video recommendations, respectively. The work of Basu, Hirsh, Cohen and Burke contributes to diversifying approaches, introducing hybrid systems to improve the accuracy of recommendations.

In the late 2000s, with the emergence of Web 2.0, Koren and Bell explored the use of large data sets and their effective integration into online platforms. This approach marked a new era in the evolution of recommendation systems, enabling further exploitation of big data to improve the personalization and relevance of recommendations provided to users.

The machine learning era in the 2010s saw the introduction of this technology by researchers like Amatriain and Basilico, propelling significant advances in personalizing recommendations.

This decade is characterized by the intensive integration of machine learning into recommendation systems, improving their efficiency and accuracy.

In the 2020s, artificial intelligence has taken center stage, with increasing integration into recommendation systems. Recent work by Gebru, Buolamwini and others has shed light on the use of AI in recommendations, highlighting potential benefits while raising ethical questions, calling for careful consideration of transparency and bias management.

3. Background

Recommendation systems occupy a central place in our digital lives, simplifying our decision-making among a multitude of available options. Increasingly essential, these tools guide our choices by suggesting suitable content and products. Their importance is manifested in various fields such as video streaming, e-commerce, and social networks. For example, on Netflix, personalized recommendations use past preferences to suggest similar content (Gomez-Uribe & Hunt, 2016), while Amazon uses these systems to personalize the shopping experience by recommending products based on past purchases (Linden, Smith, & York, 2003). On social networks like Facebook, suggestions for friends, pages and groups are influenced by the user's past interactions (Guy, 2015). These concrete examples highlight the central role of recommendation systems in our online exploration, anticipating our needs and making our digital experience more personalized and enjoyable. In short, these tools become key facilitators, contributing significantly to enriching our digital daily lives.

How to rethink traditional recommendation systems to meet the changing needs of users and improve their recommendation experience in a more dynamic, personalized and diverse way, overcoming the limitations of cold start, lack of diversity and dependence on vast historical data sets?

4. Basic concepts

Paul Resnick and Hal R. Varian, said in their books “Recommender system” “IT IS OFTEN NECESSARY TO MAKE CHOICES WITHOUT SUFFICIENT PERSONAL EXPERIENCE OF THE ALTERNATIVES.

In everyday life, we rely on recommendations from other people, either through word of mouth, letters of recommendation, movie and book reviews published in newspapers, or general surveys such than Zagat restaurant guides. »

4.1 Definitions

The development of a recommendation system aims to reduce information overload by retrieving the most relevant information and services from a huge amount of data for personalized service provision. The most important characteristic of a recommender system is its ability to “guess” a user's preferences and interests by analyzing that user's behavior and/or that of other users to generate recommendations

A recommendation system is a software environment that provides each user with the elements (items) most likely to interest them [Ricci et al., 2015].

Recommendation systems have had many definitions over the years, one of the most famous being that of Mr. Robin Burke [Burke, 2002] “System capable of providing personalized recommendations or allowing the user to be guided towards resources interesting or useful within a large data space”

5. Classifications of recommender systems

Recommendation techniques can be categorized in various ways, and it is not uncommon to observe the use of interchangeable terms to describe similar methods. Robin Burke (2002) suggests considering three significant approaches:

- The recommendation based on demographic data,
- Knowledge-based recommendation,
- Utility-based recommendation.

It is worth noting that Burke emphasizes that these three methods are special cases of a more classical approach.

As mentioned by Adomavicius and Tuzhilin in 2005, the most commonly adopted classification is based on three fundamental types:

- Content-based filtering,
- Collaborative filtering,
- Hybrid filtering.

Thus, in scrutinizing recommendation techniques, it becomes evident that a diversity of approaches and classifications exist, each making its unique contribution to the complexity of the recommendation landscape. The variety of terms used and perspectives adopted highlight the richness and nuance inherent in this field, prompting in-depth exploration of different methodologies to better understand the subtleties and specific benefits of each approach Ricci, F., Rokach, L., & Shapira, B. (2015).

Recommender systems are software tools and techniques that suggest items (such as products, services, content) to users based on their preferences, past behavior, or other characteristics. These recommender systems can be classified into several categories depending on their approach, method and objective, so additional additions can be made (Herlocker et al., 2004; Ricci et al., 2011).

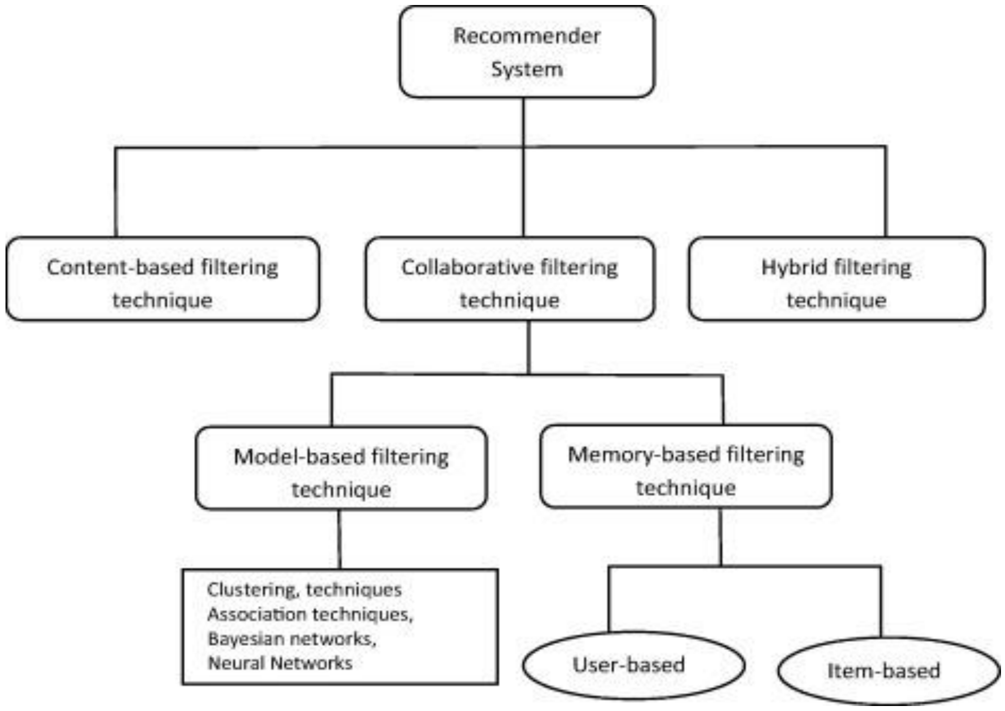

Recommender System
Content-based filtering technique
Collaborative filtering technique
Hybrid filtering technique
Model-based filtering technique
Memory-based filtering technique
Clustering, techniques
Association techniques,
Bayesian networks,
Neural Networks
User-based
Item-based

Figure 1- 1: Classification of recommendation systems [Isinkaye et al., 2015]

5.1 Content-Based-Filtering

The deployment of a content-based recommendation system requires the development of techniques to represent the elements and the user profile in a relevant manner, thus facilitating their comparison. This system stands out for its ability to suggest elements similar to those liked by the user, based on intrinsic characteristics such as genre, keywords or tags (Lops, Gemmis, & Semeraro, 2011). This approach aims to provide personalized recommendations by analyzing user preferences, thereby improving their experience (Pazzani & Billsus, 2007). The in-depth look at the architecture highlights the importance of collaboration between components to ensure the quality of recommendations.

Example of highlighting

Samanta, an avid movie lover, wants to discover new movies that match her cinematic preferences. A diverse catalog of films is available, including a variety of genres and directors. Based on her previous purchases and information on her cinematic preferences, the objective is to recommend to Samanta films that best match her tastes, or why not if Samanta likes for example Sonic 1 then recommend Sonic 2

Movie title	Gender	Director	Release year
Inception	Science Fiction	Christopher Nolan	2010
There The land	Romance	Damien Chazelle	2016
The Dark Knight	Action	Christopher Nolan	2008
pulp Fiction	Drama	Quentin Tarantino	1994
Jurassic Park	Adventure	Steven Spielberg	1993

Table 1- 1: Extract from the movie catalog

Favorite Genre	Favorite Director	Favorite Release Year
Science Fiction	Christopher Nolan	2010
Romance	Damien Chazelle	2016
Action	Christopher Nolan	2008

Table 1- 2: Extract from the User Profile

Movie title	Gender	Director	Release year	
Inception	Science Fiction	Christopher Nolan	2010	★
There The land	Romance	Damien Chazelle	2016	★
The Dark Knight	Action	Christopher Nolan	2008	★
pulp Fiction	Drama	Quentin Tarantino	1994	
Jurassic Park	Adventure	Steven Spielberg	1993	

Table 1- 3: Correlation between movie characteristics and star preferences

In this scenario, intuitively, the films "Inception" and "The Dark Knight" could be recommended to Mary, as their characteristics best match her preferences in terms of genre, director and year of release.

5.1.1 Operation of CBFs

Content feature extraction: Content features are identified and extracted. This can include attributes like keywords, genres, actors, directors, categories, etc., depending on the type of content (movies, books, music, etc.).

User profile construction: The system creates a user profile based on the user's past preferences by analyzing the content they have already consumed. The characteristics of the content liked by the user are used to build their profile.

Recommendation: Based on the user profile, the system recommends new items that share similar characteristics with those that the user previously liked. For example, if a user liked an action movie with a certain actor, the system could recommend other action movies starring the same actor.

Profile update: The system continually adjusts the user profile based on new interactions and preferences. This helps maintain accurate recommendations over time.

Content-based recommendation has both advantages and disadvantages.

5.1.2 Advantages

Personalization: Recommendations are personalized based on the user's specific preferences, which can improve the user experience by providing relevant content (Lops, Gemmis, & Semeraro, 2011).

Independence from user data: Unlike other methods like collaborative recommendation, content-based recommendation does not require data on the behaviors of other users. This can be advantageous if there is a lack of data on a large number of users.

Explanation of recommendations: It is generally easier to explain why a recommendation was made in the case of content-based recommendation. The content characteristics that led to the recommendation are often explicit (Aggarwal, 2016).

Adapting to new items: Content-based recommendation can work well even with new items for which there is no usage data yet, because it is based on the intrinsic characteristics of the content (Adomavicius & Tuzhilin, 2005).

5.1.3 Disadvantages

Limiting diversity: A major disadvantage is the risk of only recommending items similar to those that the user has already consumed, thus limiting the diversity of recommendations and the discovery of new content (Ziegler et al., 2005).

Need for relevant features: The quality of recommendations strongly depends on the quality of the features extracted from the content. If the features are not well chosen or do not represent the content well, the recommendations may be poor (Aggarwal, 2016).

Changing Preferences: If a user's preferences change over time, content-based recommendation may struggle to keep up, as it is often based on past preferences.

Lack of serendipity: Due to its content-driven nature, this type of recommendation may lack the aspect of serendipity, where the user discovers something new and unexpected (McNee, Riedl, & Konstan, 2006).

In summary, content-based recommendation is a powerful approach, but it has its limitations. It can be particularly effective when the quality of content features is high and when it is important to provide personalized recommendations without relying on other users' data. However, to ensure an optimal user experience, it can be combined with other recommendation approaches, such as collaborative recommendation, to best exploit the advantages of each method (Burke, 2002).

5.2 Collaborative filtering

Collaborative filtering is an online recommendation approach based on the sharing of opinions between users. Inspired by the concept of "word of mouth", it aims to predict a user's preferences for previously unreviewed items based on similar reviews from other users (Schafer, Konstan, & Riedl, 2001). There are two main subfamilies of collaborative filtering: memory-based methods (also called neighbor-based collaborative filtering NBCF) and model-based collaborative filtering methods. The former uses assessments stored in memory to make predictions, while the latter builds an offline model to make recommendations (Adomavicius & Tuzhilin, 2005). Collaborative filtering is widely used in a variety of online applications, from content streaming platforms to e-commerce sites, providing users with personalized and relevant recommendations.

5.2.1 Memory-based collaborative filtering (Memory-based filtering/Heuristics)

According to Adomavicius et al. (2005) or Desrosiers et al. (2011), collaborative methods, often based on neighbors, use user ratings stored in memory to predict preferences. These approaches rely on two main assumptions: first, similar users tend to give similar ratings, reflecting similar preferences, and second, similar items receive similar ratings. In summary, these methods seek to predict users' opinions based on their past preferences and similarity to other users, then use this information to make personalized recommendations.

Neighborhood-based approaches, whether they focus on users or items, generally follow two successive steps: first, identifying the relevant neighborhood, then predicting ratings from this identified neighborhood.

5.2.1.1 Neighborhood identification

Neighbor-Based Collaborative Filtration (NBCF) approaches use the R rating matrix to make recommendations based on the similarity between users or items. These approaches seek to identify similar users or items, thus forming a neighborhood, using the k-nearest neighbors (KNN) approach (Resnick et al., 1994). Users and items, represented by rating vectors in R, are compared using standardized metrics to measure their proximity. Two distinct approaches are used: one user-based, which uses the ratings of neighbors sharing similar preferences, and the other item-based, which determines a user's estimated preferences for a given item by based on user ratings for similar items (Sarwar et al., 2001).

5.2.1.2 User-based Collaborative Filtering

Uses reviews provided by similar users to make recommendations for a given user. To predict the rating that a user U would give to an item I, we calculate the weighted average of the ratings given by a group of k similar users (neighbors) to U, using the similarity between U and each of these neighbors to determine weight (Resnick et al., 1994).

$$P_{u,i} = \frac{\sum_{v \in N_i^k(u)} \mathrm{sim}\binom{u}{v}}{\sum_{v \in N_i^k(u)} \mathrm{sim}\binom{u}{v}} . P_{v,i} \qquad (1)$$

$P_{u,i}$: is the user's rating prediction for the item.

$N_i^k(u)$: is the set of the user's neighbors who also rated the item.

$P_{v,i}$: is the rating of the element by the neighbor.

$\mathrm{sim}\binom{u}{v}$: is the measure of similarity between users.

5.2.1.3 Item-based Collaborative Filtering

It first identifies a set of items similar to a given item. Then, the prediction is made based on the user's specific ratings for these similar items.

$$P_{u,i} = \frac{\sum_{j \in N_u^k(i)} \mathrm{sim}\binom{i}{j}}{\sum_{j \in N_u^k(i)} \mathrm{sim}\binom{i}{j}} . P_{u,j} \qquad (2)$$

$P_{u,i}$: is the user's rating prediction for the item.

$N_u^k(i)$: is the set of neighbors of similar elements that were also rated by the user.

$P_{u,j}$: is the user's rating for the item.

$\mathrm{sim}\binom{i}{j}$: is the measure of similarity between elements.

In summary, User-Based Collaborative Filtration predicts a user's ratings using its neighbors' ratings, while Item-Based Collaborative Filtration leverages a user's ratings for neighboring items. The approaches differ in how they solve problems, with frequent use of item neighborhoods to recommend items (Sarwar et al., 2001). Item-based methods provide clearer explanations of recommendations, using the neighborhood of items to justify results. Additionally, item-based methods can recommend similar items, while user-based methods can encourage diversity (Linden, Smith, & York, 2003). An important difference lies in the processing of ratings, with the need to center ratings on the user's average to avoid misinterpretations (Karypis, Han, & Kumar, 1999).

$$P_{u,i} = \mu_u + \frac{\sum_{v \in N_i^k(u)} \sin\binom{u}{v} . (P_{v,i} - \mu_u)}{\sum_{v \in N_i^k(u)} \sin\binom{u}{v}} \tag{3}$$

μ_u : est la moyenne des évaluations de l'utilisateur

In the specialized literature, various similarity measures have gained popularity in the recommendation field, such as cosine similarity and Pearson correlation, and we will enrich by adding other methods.

5.2.2 Model-based collaborative filtering

The first type of algorithms, as the name suggests, is based on models, supposed to reduce complexity. These models can be probabilistic and use the expectation of the evaluation to calculate the prediction. As they can be based on classifiers allowing classes to be created to reduce complexity.

Build offline a reduced image of the notes matrix (Koren et al., 2009; Salakhutdinov & Mnih, 2008) with the aim of reducing the complexity of calculations and/or dealing with the problem of missing notes. The model first goes through a learning stage, then it is used to make recommendations.
Several methods have been used for model-based recommendation algorithms. We can cite, among the most successful:

5.2.2.1 Decision trees

The history of model-based collaborative filtering, and more specifically the integration of decision trees, has its roots in the beginnings of recommender systems in the 1990s. The evolutionary approach was marked by the innovative work of Breese et al. In 1998, introducing cluster models and Bayesian networks. The increasing adoption of decision trees as a modeling method has materialized in extensive research, including that of Quoc-Cuong et al. In 2014, highlighting the ability of these trees to reconcile interpretability and precise modeling of user-item preferences.

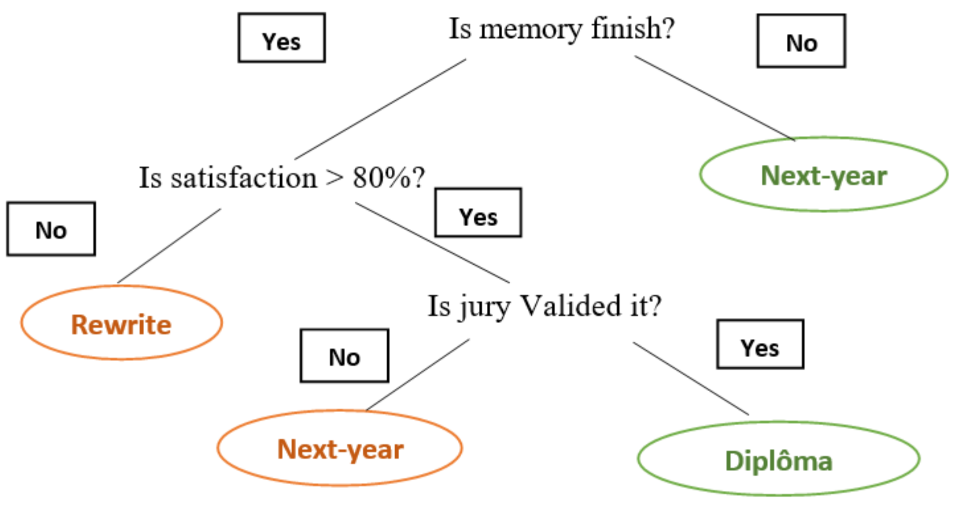


Figure 1- 2: Example of precision shaft operation

5.2.2.2 Singular value decomposition

Singular value decomposition (SVD) is a widely used mathematical technique for dimension reduction in the field of recommender systems (Koren et al., 2009). It allows a matrix to be factorized into three smaller matrices, thus revealing latent relationships between users and elements. This approach is particularly useful for dealing with large and sparse evaluation matrices frequently encountered in recommender systems (Salakhutdinov & Mnih, 2008).

How the SVD works

The SVD decomposes a matrix A into three matrices:

U: An orthogonal matrix containing the left singular vectors.

W: A diagonal matrix containing the singular values.

V: An orthogonal matrix containing the right singular vectors.

$$[A] = [U] \begin{matrix} W1 & 0 & 0 \\ 0 & . & 0 \\ 0 & 0 & Wn \end{matrix} [V]^T \tag{4}$$

SVD(A)= [U, W, A]

The SVD makes it possible to reduce the dimension of the matrix by considering only the k largest singular values. This results in the creation of a matrix Wk of reduced size, containing the k most important singular values. The associated U and V matrices are also reduced accordingly, leading to Uk and Vk matrices of smaller dimensions.

SVD is commonly used in recommender systems based on collaborative filtering. Indeed, it makes it possible to identify users sharing similar tastes and elements with common characteristics, thus contributing to the generation of more relevant and personalized recommendations (Koren et al., 2009).

In summary, singular value decomposition (SVD) has emerged as an effective and efficient dimension reduction technique for recommendation systems. Its ability to process large matrices, improve predictive accuracy, and reveal latent relationships makes it a valuable tool for building more efficient and personalized recommendation systems.

5.2.2.3 Probabilistic Model

Probabilistic models, notably based on clusters or Bayesian networks, were proposed by Breese et al. (1998). They developed approaches using probability models to predict user ratings.

The Bayesian network model, also proposed by Breese et al. (1998), represents users and items as nodes, with states corresponding to possible evaluations. Clustering methods make it possible to limit the number of individuals considered in calculating the prediction. The processing time will therefore be shorter and the results will potentially be more relevant since the observations will relate to a group closest to the active user. In other words, instead of consulting the entire population, we estimate the preference of a group of people with the same tastes as the user.

As part of model-based methods and to address the lack of data, clustering has been widely used. Among these works, clustering was applied either to users or to items or to both, in order to generate clusters of similar users or items in order to predict missing scores. Different clustering algorithms have been used for this purpose, notably: k-means [Ungar and Foster, 1998] and many other authors etc.

In the context of model-based approaches, the prediction can be made in two different ways:

- From the prediction provided by the model itself, for example by constructing a probabilistic model for the estimation of prediction values or directly from the model.
- Or, by grouping users\items using clustering methods and subsequently, memory-based methods (user-based or item-based) used to predict the ratings for the items.

We also have Association Rules Based Approaches which use association rules to identify patterns and relationships between user preferences.

5.2.2.4 Model-Based Approaches

These algorithms aim to construct a reduced representation of the score matrix offline Su, X., & Khoshgoftaar, T. M. (2009), which makes it possible to reduce the complexity of the calculations and to deal with problems linked to missing scores in the context of the recommendation. We have two essential stages for learning:

Learning (offline): The model is first trained on a set of data. This generally involves building a representation or model based on the characteristics of users, items and their interactions by Koren, Y., & Bell, R. (2015).

Recommendation (online): Once the model is trained, it is used to make recommendations in real time. This often involves predicting a user's preferences for items that they have not yet rated by Ricci, F., Rokach, L., & Shapira, B. (2015).

Model-based algorithms for recommendation use various techniques to model and predict user preferences, thereby providing personalized recommendations (Koren et al., 2009).

Neighborhood-Based Collaborative Filtering (NBCF) approaches, presented previously, were among the first collaborative filtering methods and have retained their popularity thanks to their simplicity. However, although they have advantages, they are not always the most suitable for predicting ratings accurately by Lops, P., De Gemmis, M., & Semeraro, G. (2011).

Unlike NBCF approaches, which are considered instance-specific methods, model-based approaches introduce a clear distinction between the training phase and the prediction phase (Koren et al., 2009). In the case of NBCF approaches, a model is not explicitly created in advance for prediction, except for pre-processing phases such as calculating similarities or predictions based on predefined equations. In contrast, model-based approaches involve the creation of a machine learning model, usually supervised, whose parameters are adjusted based on the ratings available in the dataset (Koren et al., 2009). This parameter adjustment phase is called model training.

Various machine learning models, such as decision trees, support vector machines (SVM), neural networks, and association rule-based models, have been widely used to solve practical problems, including rating prediction. These models can be adapted to deal with the problem of collaborative filtering, considering the prediction of ratings as a generalization of classification or regression problems. Among the models applied, we find decision trees, naive Bayesian models, models based on matrix factorization (or latent factor), and models based on neural networks (Aggarwal, 2016).

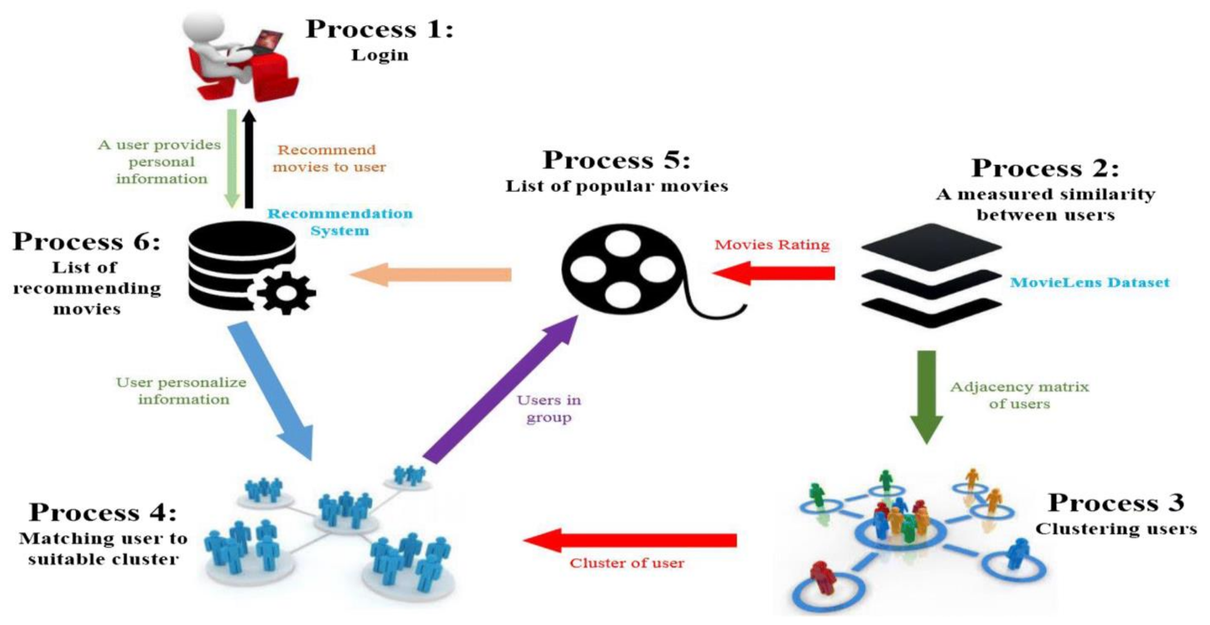


Figure 1- 3:Organization chart of the collaborative recommendation system

This image illustrates a movie recommendation process based on users' personal information. Here is a detailed expansion of each step:

- Connection

The user connects to the recommendation system (platform) using their identifiers. Upon login, the system has access to user profile information, including watched movie history, assigned ratings, preferred genres, etc.

- Similarity between Users

The system analyzes the similarity between user profiles based on criteria such as preferred movie genres, assigned ratings, or other relevant characteristics. Users with similar profiles are identified as similar peers.

- Grouping of Users

Similar users are grouped into clusters or groups based on their similarity. This grouping makes it possible to create segments of users sharing common cinematic preferences.

- Match Users to Groups

Each user is associated with the group that best represents their cinematic tastes. This group-user correspondence helps simplify the recommendation process by working with segments of users rather than isolated individuals.

- List of Popular Films

The system compiles a list of popular movies based on criteria such as overall ratings, number of views, etc. This list represents movies that are currently popular among all users of the system.

- List of Recommended Movies

Using information from the group the user is associated with, the system generates a personalized list of recommended movies. This recommendation process leverages collaboration between similar users to improve the accuracy and personalization of recommendations, providing a cinematic experience more tailored to each user's individual tastes.

This system recommends items based on the preferences of similar users.

5.2.3 Similarity calculation

5.2.3.1 Cosine Similarity

The cosine similarity method differs from other similarity measures by evaluating the similarity between users. Unlike other statistical approaches, it considers each user as a vector representing their evaluations. The cosine similarity measure calculates the distance between these two vectors. This measure is defined by the following formula:

$$\operatorname{sim}(a,b) = \cos\left(\overrightarrow{xa}, \overrightarrow{xb}\right) = \frac{\overrightarrow{xa}.\overrightarrow{xb}}{\left\|\overrightarrow{Xa}\right\|.\left\|\overrightarrow{Xb}\right\|} \qquad (5)$$

$\overrightarrow{Xa}$: User Rating Vectors for user a

$\overrightarrow{Xb}$: User Rating Vectors for user b

For users: Collaborative filtering involves the representation of users by vectors denoted xu, where each component xui corresponds to the notation assigned to item i by user u. To evaluate the similarity between two users, u and v, we calculate the cosine between their respective vectors xu and xv, limiting ourselves to a common set of items rated by the two users. This process makes it possible to measure the proximity in their preferences.

$$\operatorname{sim}(u,v) = \cos(\overrightarrow{xu}, \overrightarrow{xv}) = \frac{\sum_{i \in I_{u,v}} r_{u,i} \cdot r_{v,i}}{\sqrt{\sum_{i \in I_{u,v}} r_{u,i^2}} \cdot \sqrt{\sum_{i \in I_{u,v}} r_{v,i^2}}} \qquad (6)$$

The cosine can also be used to calculate the similarity between two items. To do this, simply substitute the users with the corresponding items in the equation, as illustrated in:

$$\operatorname{sim}(i,j) = \cos(\overrightarrow{xi}, \overrightarrow{xj}) = \frac{\sum_{u \in U_{i,j}} r_{u,i} \cdot r_{v,j}}{\sqrt{\sum_{u \in U_{i,j}} r_{u,i^2}} \cdot \sqrt{\sum_{u \in U_{i,j}} r_{v,j^2}}} \qquad (7)$$

5.2.3.2 Pearson correlation

Pearson correlation is an essential tool in Neighbor-Based Collaborative Filtration (NBCF) approaches. This statistical measure evaluates the linear relationship between two continuous variables, which is crucial in collaborative recommendation (Zhang, S., Yao, L., Sun, A. & Tay, Y.2020). It was developed by Karl Pearson in the early 20th century and is used to quantify the similarity between user or item preferences. In user-based NBCF, it measures the concordance of preference profiles, while in item-based NBCF, it assesses the similarity between the ratings assigned to an item. By integrating Pearson correlation, recommendation systems can identify similar users or items, thus improving the relevance of personalized recommendations. Pearson's correlation, known as the linear correlation coefficient, measures the strength and direction of a linear relationship between two continuous variables, denoted by "r" and varying from -1 to 1. An r of 1 indicates a correlation perfect positive, -1 a perfect negative correlation, and 0 suggests no linear correlation.
The following formula gives the Pearson similarity between two users.

$$\mathrm{sim}(u,v) = Pearson(u,v) = \frac{\sum_{i\in I_{u,v}}\left(r_{u,i}-r_u\right).\left(r_{v,i}-r_v\right)}{\sqrt{\sum_{i\in I_{u,v}}\left(r_{u,i}-r_u\right)^2}\ .\ \sqrt{\sum_{i\in I_{u,v}}\left(r_{v,i}-r_v\right)^2}} \quad (8)$$

The following formula gives the Pearson similarity between two items.

$$\mathrm{sim}(i,j) = Pearson(i,j) = \frac{\sum_{u\in U_{i,j}}\left(r_{u,i}-r_i\right).\left(r_{v,i}-r_j\right)}{\sqrt{\sum_{u\in U_{i,j}}\left(r_{u,i}-r_i\right)^2}\ .\ \sqrt{\sum_{u\in U_{i,j}}\left(r_{v,i}-r_j\right)^2}} \quad (9)$$

5.2.3.3 Other similarity methods

- ***Spearman Rank Correlation***

The Spearman correlation algorithm uses rating rankings (Zar, 2010) to evaluate the strength of the relationship between two variables. In case of non-normalized data distribution, Spearman can produce more accurate results than Pearson correlation. It measures the correlation between data ranks instead of rating values (Zar, 2010). For this, user ratings are converted into ranks, assigning the lowest rank to the highest rating, with an average for equal ratings. Calculating Spearman's correlation is similar to Pearson's, but uses ranks instead of ratings.

The calculation of the Spearman correlation is carried out as follows:

$$\mathrm{sim(u,u")^{SRCC}} = 1 - \frac{6\ \sum_{\mathrm{i}\in \mathrm{I}} \mathrm{rank}\left(\mathrm{r_{u,i}}\right)^2 - \mathrm{rank}\left(\mathrm{r_{u",i}}\right)^2}{|\mathrm{I}|.\left(|\mathrm{I}|^2-1\right)} \quad (10)$$

"Where |I| is the cardinality of the co-evaluated elements."

- ***Mean squared displacement***

The metric consists of taking the sum of the squares of the differences between the ratings given by user u and user v for the elements they have in common. Then, this sum is divided by the number of items shared between the two users. This measure aims to assess the similarity of tastes between users based on the ratings of their common elements. The mean square distance diagram is defined as follows:

In some applications, particularly when comparing distances, the quadratic Euclidean distance is preferred by including the square root in the calculation. In the context of Neighbor-Based Collaborative Filtering (NBCF), users (or items) are considered as points in a Euclidean space. However, the quadratic Euclidean distance may be biased, favoring users rating more items (Aggarwal, 2016).

$$\mathrm{d}\left(\vec{a},\vec{b}\right) = (a_1 - b_1)^2 + (a_2 - b_2)^2 + (a_n - b_n)^2 \qquad (11)$$

To correct this, Mean Squared Distances (MSD) is used, providing a balanced measure of similarity between users or items, taking into account the density of ratings.

For users:

$$MSD(u_1,u_2) = \frac{\sum_{i\in I_{u_1,u_2}}\left(r_{u_1,i} - r_{u_2,i}\right)^2}{\mathrm{I}_{\mathrm{u}_1,\mathrm{u}_2}} \qquad (12)$$

For user similarity we go the opposite way:

$$sim_{MSD(\mathrm{u}_1,\mathrm{u}_2)} = \frac{1}{MSD(u_1,u_2) + 1} \qquad (13)$$

For items:

$$MSD(i,j) = \frac{\sum_{u\in U_{i,j}}\left(r_{u,i} - r_{u,j}\right)^2}{\mathrm{U}_{\mathrm{i,j}}} \qquad (14)$$

For the similarity of items, we go the opposite way:

$$sim_{MSD(i,j)} = \frac{1}{MSD(i,j) + 1} \qquad (15)$$

- ***The adjusted cosine***

When Pearson similarity is used to calculate the similarity between two items, the ratings of the same user are centered in relation to the average of their ratings. However, the variation in rating for the same user is not as important as the variation between different users. This is why it is more interesting, when calculating the similarity between two items, to adjust the ratings in relation to the average of the users' ratings rather than in relation to the average of the item ratings. This is the role of the adjusted Cosine, which was introduced by Sarwar et al. [Sarwar et al., 2001]. According to [Schafer et al., 2007], the adjusted cosine is considered one of the most effective and popular ways to calculate the similarity between two items for collaborative filtering algorithms.

$$\mathrm{sim}(i,j) = Adjusted_{cosine(i,j)} = \frac{\sum_{u\in U_{i,j}}\left(r_{u,i} - r_u\right).\left(r_{u,j} - r_u\right)}{\sqrt{\sum_{u\in U_{i,j}}\left(r_{u,i} - r_u\right)^2} \ . \ \sqrt{\sum_{u\in U_{i,j}}\left(r_{u,j} - r_u\right)^2}} \quad (16)$$

Notation: We will use the following notation in the formulas which will be presented in the rest of this section:

Settings	Explanation
u,v	Users
i, j	Article
Ui	All users who liked an article i
Iu	Set of articles liked by a user u
Rui	User preference u for item i
R	Normalized preference value
R	Predicted preference value
Naked)	Users similar to user u
W	Weighting value

Table 1- 4: Notation and parameters used

5.3 Hybrid Filtering

Hybrid filtering systems integrate the collaborative filtering system with other recommendation techniques, often by combining content-based systems, to generate predictions or recommendations. This approach aims to overcome the limitations and drawbacks inherent in content-based and collaborative systems (Rao & Talwar, 2008).

Generally, hybrid filtering adopts various methods, such as weighting, cascading, switching, etc., to merge recommendation sets and generate final suggestions for users (Burke, 2002).

5.3.1 Methods used for hybrid filtering

- ***Weighted approach***

As described by Adomavicius and Tuzhilin (2005), is a hybrid method that merges the results of various approaches using appropriate weighting of the scores of each recommendation technique. This technique involves linearly combining the scores generated by different recommendation systems, thus providing a final estimate of the rating that the user would assign to an item. For example, a model suggested by Miranda et al. (1999) merges the predictions of a content-based filtering (CBF) system and a collaborative filtering (CF) system, adjusting the weights assigned to each system based on user feedback.

- ***Switching approach***

Described by Billsus et al. (2000), consists of choosing among several recommendation models according to specific criteria. The system must define these criteria, thereby determining when to use one technique over another, taking into consideration the strengths and weaknesses of each method. In switched hybridization, the system alternates between different recommendation techniques based on certain parameters. For example, Billsus et al. (2000) propose a method where a content-based system is used when data is limited, while a collaborative system is employed when more comprehensive data is available. A similar approach is presented by Benouaret (2017) for cultural site recommendation, alternating between a demographic, content-based and collaborative approach depending on the level of user feedback.

- ***Mixed approach***

As explained by Smyth and Cotter (2000), the recommender system does not directly merge data, but rather enriches the description of datasets by taking into account user estimates and item characteristics. This avoids problems encountered with the cold start of collaborative filtering. Unlike classic hybrid methods, where recommender systems are merged, in the mixed approach, recommendations from different techniques are presented to the user simultaneously. In other words, this method offers a list of items that contains recommendations of each approach used, thus providing a diverse overview of the available suggestions.

- ***The waterfall method***

Described by Lampropoulos et al. (2012) is a sequential recommendation approach where different techniques are used step by step to refine the suggestions. It begins by generating an initial list of recommended items, then a second technique intervenes to improve this list by taking into account the user's specific preferences. For example, in music recommendation, one could first use content-based filtering to find similar songs, then collaborative filtering to refine the selection based on other users' preferences. This approach makes it possible to combine the strengths of each method to offer more relevant and personalized recommendations, thus improving the user experience and overall satisfaction.

- ***Features combination***

In a hybrid based on the combination of features, data from collaborative techniques are treated as a feature, while a content-based approach is used on these data (Adomavicius and Tuzhilin, 2005).

- ***Feature augmentation***

This method, similar to waterfall, uses the results of the first technique as an added feature for the second technique (Adomavicius and Tuzhilin, 2005).

- ***Meta-level approach***

In a meta-level hybrid, a first technique is used to produce a model, and in the second step, the entire model serves as input for the second technique (Adomavicius and Tuzhilin, 2005).

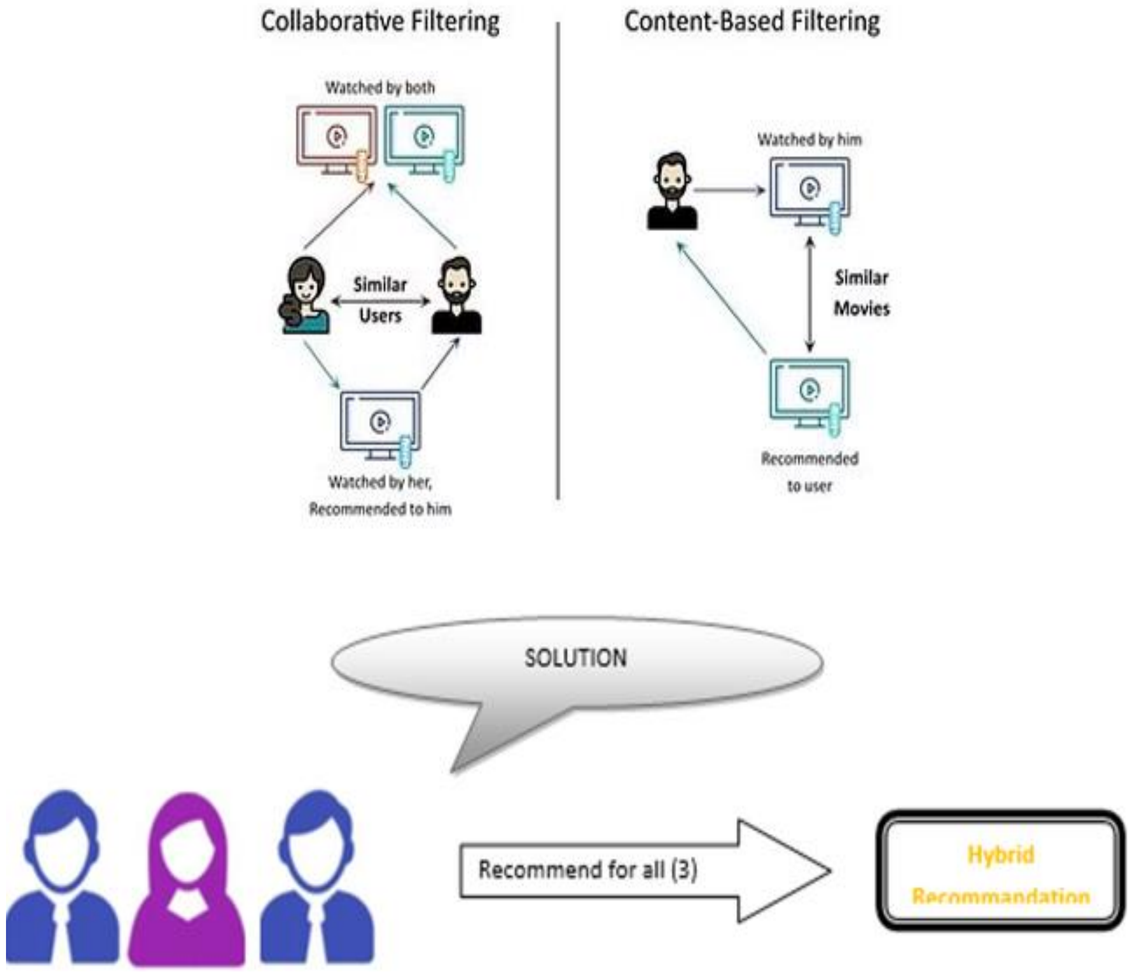


Figure 1- 4:Highlighting hybrid filtering

5.4 Other forms of filtering

5.4.1 Demographic Filtering

Relies on the use of demographic information such as users' age, gender and geographic location to make personalized recommendations. This approach aims to understand user preferences by taking into account their demographic characteristics.

5.4.2 Knowledge-Based Filtering

Recommends items based on expert-defined rules or an explicit knowledge base. This method relies on a deep understanding of user needs and preferences.

5.4.3 Temporal Filtering

Takes the time dimension into consideration to formulate recommendations. It adjusts its suggestions based on changing trends or seasonal user preferences, improving the relevance of recommendations over time.

5.4.4 Trust-Based Filtering

Uses trust information between users to refine recommendations. Based on the trust relationships established between users, this approach seeks to strengthen the reliability of suggestions.

5.4.5 Critical Filtering (Critical-Based Filtering)

Leverages user reviews and ratings to generate recommendations. By taking into account user feedback and opinions, this method aims to personalize suggestions based on individual preferences.

5.4.6 Utility-Based Filtering

Explicitly evaluates the usefulness of items to a user, taking into account various criteria such as past satisfaction, relevance, and other factors.

5.4.7 Community-Based Filtering

Involves the use of collective information from a community of users to make recommendations. By leveraging emerging trends and preferences within the community, this approach helps improve the quality of suggestions. It is important to note that these classifications are not mutually exclusive, and many recommender systems can incorporate a combination of these approaches to optimize the relevance of recommendations.

6. Limitations of recommendation systems

With the advent of the Internet and the wealth of information it offers, recommender systems are proving to be crucial tools in solving the problem of information overload and helping users find relevant information. However, despite their many advantages, these systems also have significant limitations. Among these limitations, two major problems are often highlighted.

6.1 Cold start

Constituting a major challenge for recommendation systems, appearing when introducing new elements such as users or items. This specific situation requires special attention, because the lack of initial information limits the system's ability to make relevant recommendations, as highlighted by Rashid et al. (2002).

6.2 Parsimony

Characterized by the scarcity of data. This challenge arises from the unavailability of a large number of scored items for each active user, thus contributing to a very sparse score matrix, sometimes reaching up to 95% missing values, as reported by Papagelis et al. (2005). This sparsity complicates community formation and may result in less accurate recommendation results, based on discussions by Adomavicius & Tuzhilin (2005) and Linden et al. (2003).

6.3 Serendipity

Since the content-based recommendation system only recommends items that match the user's profile, i.e. items related to these interactions, users will only receive recommendations similar to those that they have already met. He will have no chance of receiving unexpected suggestions. This can cause users to become bored with it.

6.4 The gray sheep problem

In the field of recommender systems, the "gray sheep" problem refers to a situation where a user or item has very similar characteristics or preferences to other users or items, thus making it difficult to recommend items. relevant or the differentiation between them (Baltrunas & Ricci, 2011).

7. Conclusion

In this chapter, we discussed the foundations of recommender systems. These tools are proving effective in identifying potential projects of interest, providing a valuable solution to the information overload we face. SRs are able to predict user preferences based on the user's history of interactions with the system. These predictions can be based on the specific preferences of targeted users (CBF method) or on those of similar users (CF method). A hybrid approach is also presented, aiming to leverage the advantages of each method while overcoming their possible shortcomings. By examining current limitations, such as cold starting and sparsity, this chapter highlights persistent challenges in the field, thereby justifying the need for new approaches and innovations in recommender systems.

Chapter 2

Modeling of recommendation systems using bipartite graphs

1. Introduction

Bipartite graphs are useful in modeling a multitude of relationships observed in the real world, affecting various fields such as medicine, social networks, and marketing. These relationships embrace a wide variety of concrete cases, such as interactions between drugs and their side effects, links between genes and different diseases, collaborations between researchers and the publications they co-author, as well as relationships between actors and the films in which they appear (Newman, 2010).

In medicine, bipartite graphs are used to represent associations between drugs and adverse reactions, helping to identify potential risks and improve the safety of treatments (Atzori et al., 2014). In the field of social networks, these graphs are used to map relationships between users, their interactions and their preferences, thus facilitating the recommendation of new contacts or relevant content (Wasserman & Faust, 1994). In marketing, bipartite graphs can model the relationships between customers and the products they buy or like, helping companies target their offers and personalize their sales strategies (Borgatti & Halgin, 2011).

Other examples include the use of bipartite graphs to represent species interactions in ecosystems, relationships between users and documents in search engines, and connections between skills and job postings in recruitment systems (Proulx, Promislow, & Phillips, 2005). In summary, bipartite graphs provide a powerful framework for analyzing and understanding the complex relationships that underlie many real-world phenomena. These graphs provide a powerful tool for visualizing and analyzing the relationships between two distinct sets of objects. In this section, we will dive into introducing bipartite graphs, exploring their definition, properties, and applications (Easley & Kleinberg, 2010).

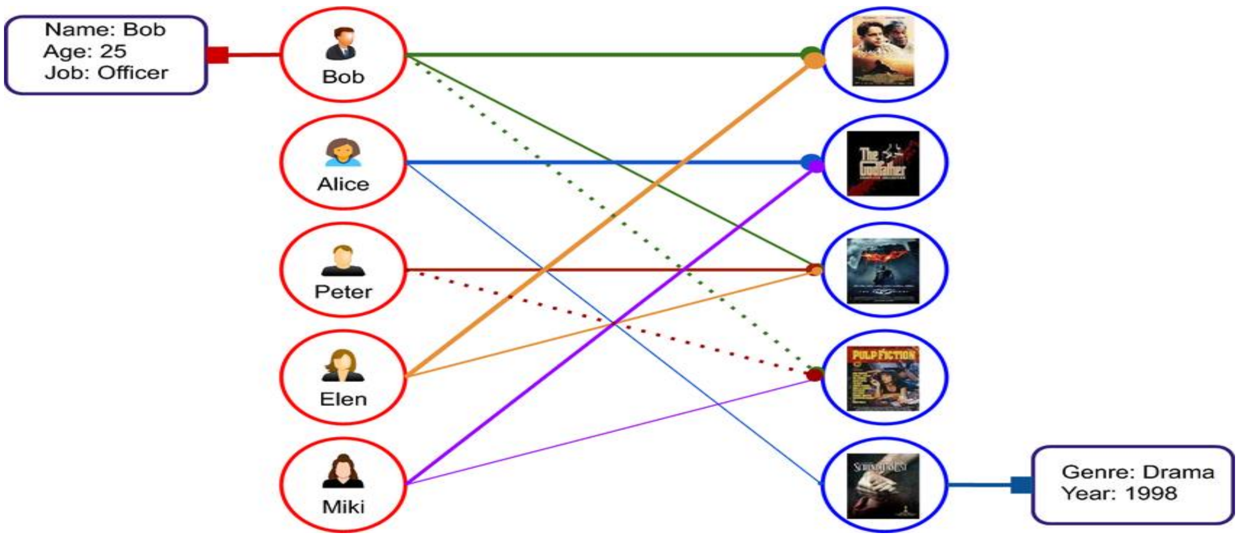


Figure 2- 1:Example of bipartite graph

2. History of graphs

The history of graphs dates back to the 18th century, with the innovative work of Swiss mathematician Leonhard Euler. In 1736, Euler solved the famous Königsberg bridge problem, where he introduced the notion of graphs to represent the connections between the city's land and bridges. Euler formalized the concepts of nodes (or vertices) and edges (or connections) in his work on graphs. His approach laid the foundation for modern graph theory, which has since undergone considerable development. When it comes to graph types, they have arisen over time in response to various analysis and modeling needs in different domains. This is how bipartite graphs came into existence and thanks to the significant contributions to various areas of mathematics, including set theory, logic and graph theory by Augustus De Morgan, a 19th century British mathematician century. Although he was not the first to introduce the concept of a bipartite graph, he played an important role in its popularization and understanding. In his work "On the Syllogism, and Other Logical Writings" published in 1860, De Morgan discussed ideas related to the bipartition of sets and the representation of relationships between them. He examined structures where elements of a set were divided into two distinct sets, and how these divisions could be used to model specific relationships. It was in this context that he discussed bipartite graphs. De Morgan was one of the first to recognize the importance of bipartite graphs in representing various relationships in the real world, such as relationships between sets of different objects. Thus, although De Morgan was not the creator of the concept of bipartite graph, he contributed to its understanding and recognition in the field of mathematics.

3. Practical case

An interesting example of the use of bipartite graphs is their application in the study of social networks. A 2001 article by Lada A. Adamic and Eytan Adar, titled "Friends and neighbors on the web", presents an analysis of online social networks using bipartite graphs.

In this paper, the authors examined connection patterns between users and the web pages they visited, constructing a bipartite graph connecting users to pages. They found that users tended to connect to pages similar to those visited by their online friends, indicating a correlation between browsing choices and social network structure.

This study contributed to the understanding of online social interactions and to the development of predictive models based on bipartite graphs. Additionally, it paved the way for future research into online social networks and their influence on user behavior.

Our bipartite graph approach will focus on ways of using it in various domains.

3.1 Knowledge graphs

3.1.1 General overview

In 1972, during discussions about building course modules, the term "knowledge graph" was introduced. In the late 1980s, the universities of Groningen and Twente launched a joint project called Knowledge Graphs, aiming to establish a semantic network structure. Initially, knowledge graphs are mainly used in specific domains. For example, in 1985, WordNet created a knowledge graph focused on the relationships between words and their meanings, while in 2005, GeoNames presented a graph in the field of geography. In 2007, DBpedia and Freebase emerged as two general knowledge graphs, although the term was not yet widely used. In 2012, Google launched the Google Knowledge Graph (GKG), which integrated data from DBpedia and Freebase, as described in the work of Bollacker et al. (2008) for Freebase and de Auer et al. (2007) for DBpedia. Subsequently, the GKG enriches its data by integrating formats such as RDFa, microdata and JSON-LD, from sources such as the CIA World Factbook, Wikidata and Wikipedia, as documented in the publications of Bizer et al. (2009). The GKG thus helps popularize the notion of knowledge graph, marking the start of a trend adopted by other technology giants such as Airbnb (Chang, 2018), Amazon (Krishnan, 2018), eBay (Pittman and al., 2017), Facebook (Noy et al., 2019), IBM (Devarajan, 2017), LinkedIn (He et al., 2016), Microsoft (Shrsivastava, 2017), Uber (Hamad et al., 2018), and even more. This growing adoption by industry has sparked intense interest in academia, leading to a proliferation of academic work on the topic. Works like those of Pan et al. (2017), Qi et al. (2021), Fensel et al. (2020), Kejriwal et al. (2021), as well as articles defining key concepts such as those by Ehrlinger and Wöß (2016), have contributed to enriching the scientific literature. Furthermore, extensive studies on various aspects of knowledge graphs, such as those by Paulheim (2017) and Wang et al. (2017), have expanded our understanding of the field.

In this section, we will take a different perspective by examining the contributions of knowledge graphs to recommendation systems, emphasizing the quantitative and data-driven aspects. More specifically, we will explore advances in Linked Data and recent techniques associated with large-scale knowledge graphs, such as DBpedia. We will focus in particular on the use of techniques such as embeddings, which make it possible to represent entities in a space defined by the knowledge graph.

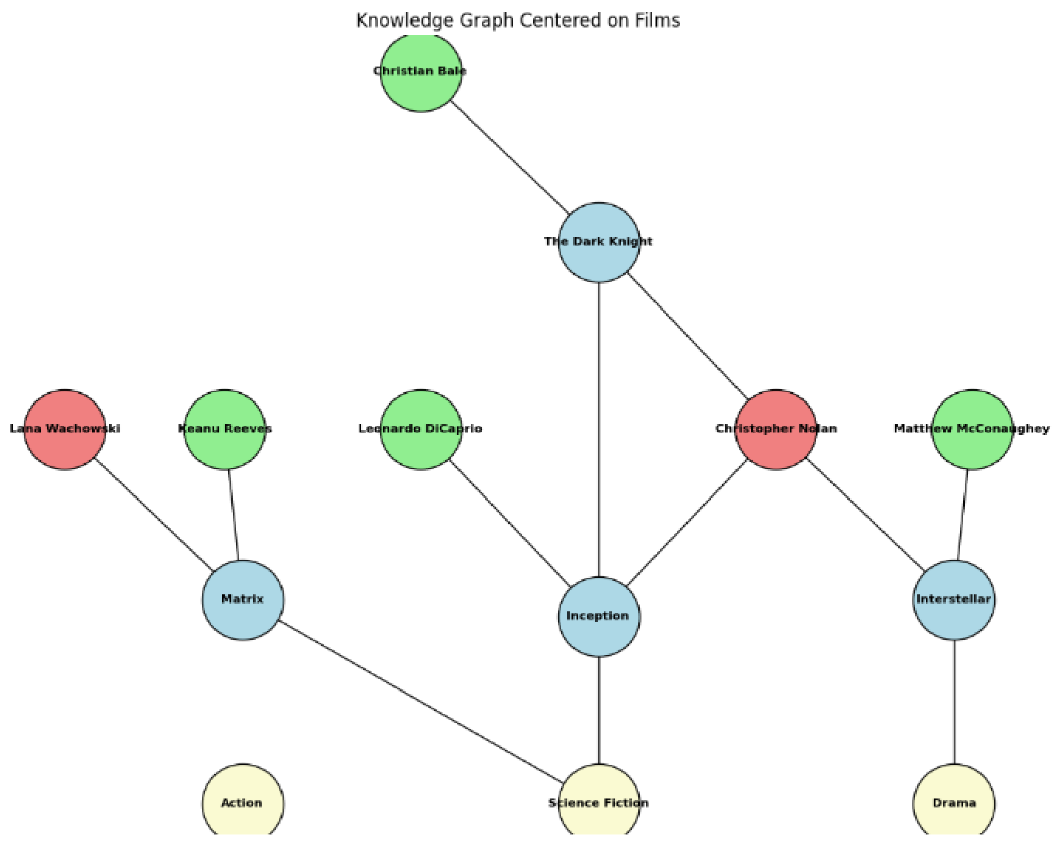


Figure 2- 2:Illustration of a knowledge graph

3.1.2 Highlighting process

The essence of knowledge graphs is found in triples, elementary units that represent factual relationships between entities. Each triplet consists of three elements:

Subject (head): The entity about which the relationship is established.
Predicate (relation): The type of relationship that exists between the subject and the object
Object (tail): The entity to which the subject is linked by the relationship.

We can note a triple as follows: (head, relation, tail)

Let's take the example of the sentence "Tayeb and Konta study at Ibn Khaldoun". In this case:
Subject: Tayeb and Konta
Predicate: work to
Subject: Ibn Khaldun

The set of relationships and types of admissible entities in a knowledge graph defines its ontology. This ontology acts as a frame of reference, similar to the way our brain organizes and structures its knowledge, allowing us to connect information and build a more comprehensive understanding. In this context, knowledge graphs mimic the human cognitive process by identifying and relating facts about people, places, and other entities. This approach helps generate more accurate and relevant results for users.

Instead of drowning in a multitude of web pages, links and irrelevant details, knowledge graphs extract the most relevant, popular or searched facts related to a given topic. This allows users to quickly and efficiently access the information that interests them.

In summary, triples form the backbone of knowledge graphs, capturing factual relationships and allowing information to be structured and organized in a way similar to human reasoning.

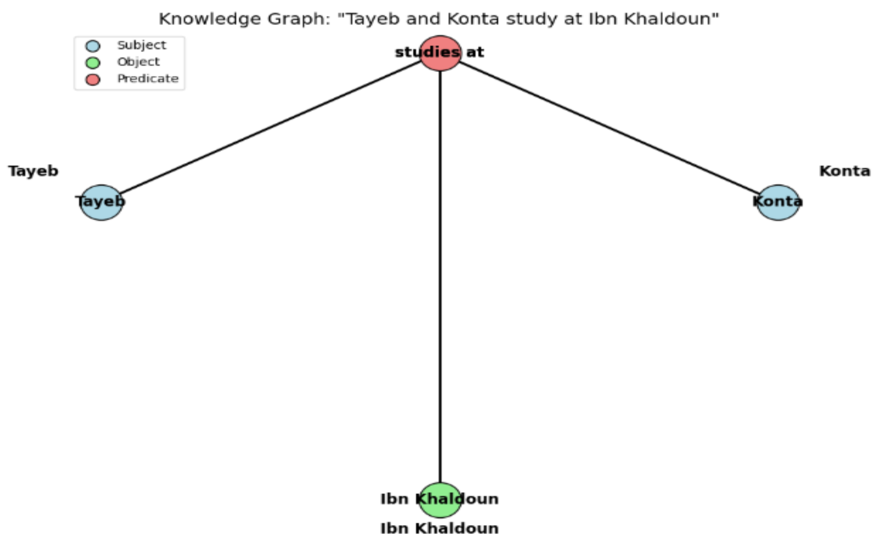


Figure 2- 3: Triplet knowledge graph (subject, predicate, object)

3.1.3 Recommendation models based on Linked Data

The concept of "Linked data", introduced by Tim Berners-Lee, director of the W3C (World Wide Web Consortium), aims to transform the way in which data is published on the Web. Rather than being isolated in silos, this data is interconnected to form a global network of structured information.

This initiative, known as the Web of Data, is based on Web standards such as HTTP and URI. Unlike their traditional use for human navigation, the Web of Data extends these standards to enable information sharing between machines. This means that data can be accessed automatically, regardless of where it is stored, without duplication. In summary, the Web of Data opens new perspectives in data management and sharing on the Internet, promoting more efficient interconnection and knowledge exploration by Heath, T., & Bizer, C. (2011).

3.1.3.1 The LOD (Linked Open Data) project

Linked Open Data (LOD) has revolutionized the way we manipulate and analyze information Berners-Lee, T. (2006). Building on key principles like URIs (to designate entities being manipulated), RDF (for a common machine-readable representation of data), and SPARQL (a common way to query this data), LODs enable to create a vast knowledge network where data from diverse sources can be interconnected and understood by machines.

This concept, born from the Semantic Web and popularized by Tim Berners-Lee, has experienced tremendous growth since its beginnings in 2004. Major projects such as

DBpedia (Auer et al., 2007) and Wikidata (which we will discuss with the next titles) have emerged, offering comprehensive of rich and varied LOD data. Today, LODs are being adopted in many areas, from research and libraries to government and business.

Rather than blocking data in siled aggregation systems (like relational databases), LOD allows you to find information where it already exists. The data is structured and standardized, then placed on the web in conjunction with other data on the same subject. One of the main advantages of LODs is their interoperability.

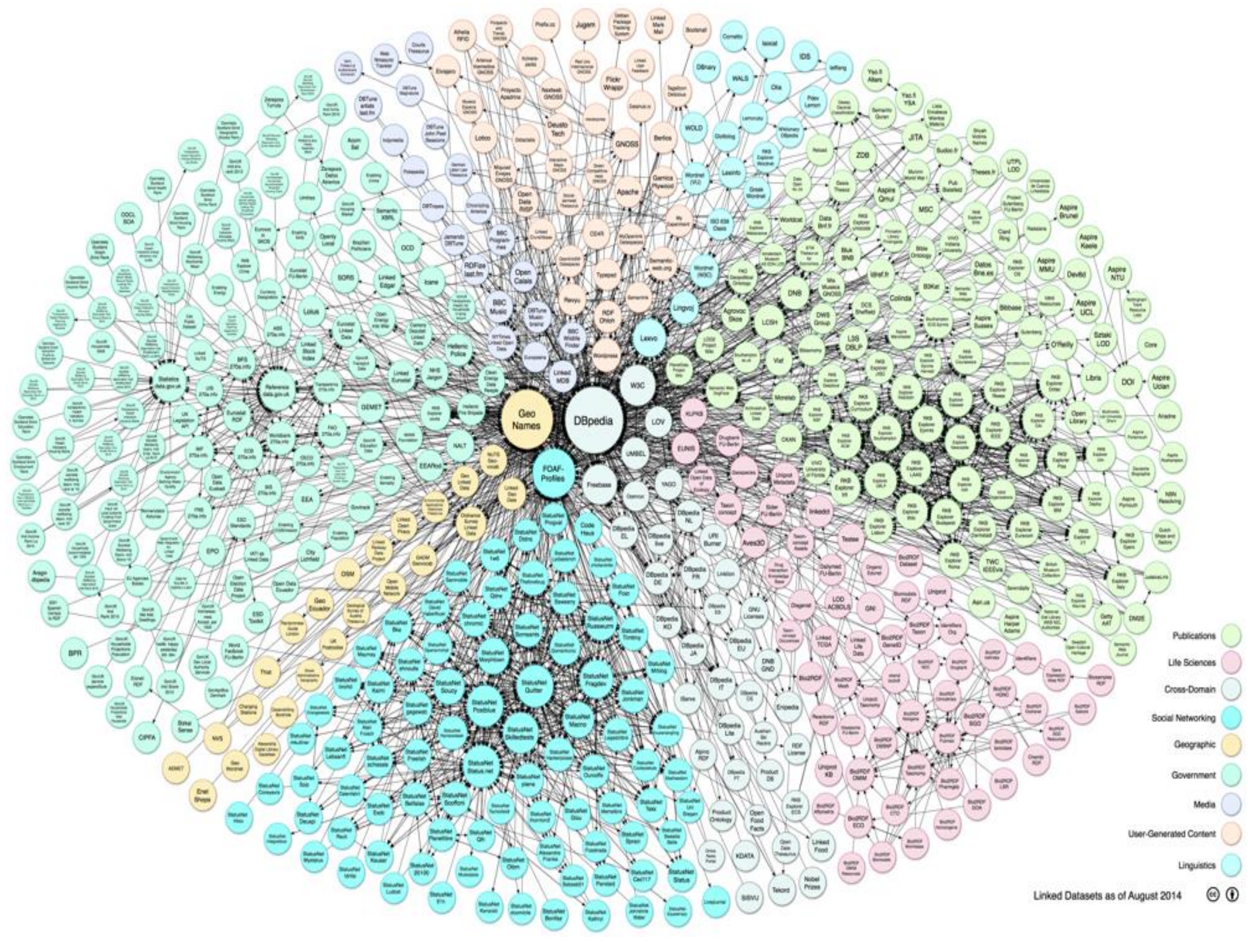


Figure 2- 4: Linked open data (LOD) cloud.

- **Application of LOD**

Europeana: European digital library which aggregates millions of digitized works from libraries, museums and archives across Europe.

They use LOD to connect these collections and allow users to discover cultural resources in a more integrated way.

GeoNames: Is a geographic database that provides information on place names, geographic coordinates and other associated data.

They also use LOD to link this information to other geospatial data sources.

Linked Data for Libraries (LD4L): Is a collaborative project between several American university libraries.

They work to create links between bibliographic data, authors, subjects and other information relevant to academic research.

American Art Collaborative (AAC): A group of fourteen American museums working to establish a critical mass of open and linked data on American art. The project aims to improve understanding and appreciation of art by making data more accessible. With a grant from the Andrew W. Mellon Foundation, the American Art Museum of the Smithsonian Institution launched this project in fall 2014. Participating museums convert their data to LOD and develop research and education applications from it. those data.

By leveraging distance-based similarity measures in LOD graphs, recommender systems can identify movies, music, books, or other content similar to those the user has already enjoyed. Taking into account contextual relationships and entity attributes makes it possible to offer more original and less predictable suggestions, taking into account the user's tastes and context.

Let's imagine a user who enjoyed the films "The Godfather" and "Goodfellas", two classic gangster films directed by Francis Ford Coppola. The recommender system should identify similar movies that might appeal to that user, drawing on LOD data from sources like Wikidata, DBpedia, and The Movie Database.

- **Construction of the LOD graph**

The LOD graph is constructed from data from the sources mentioned above. Nodes represent films, directors, actors, genres, etc., and links represent relationships between these entities.

For example, the node "The Godfather" is linked to the node "Francis Ford Coppola" by a director relationship, and to the node "Martin Scorsese" by an actor relationship.

Representation of entities (Nodes):

Movies, Directors, Actors, Genres, Reviews.

Representation of relationships (Links):

Film directed by director

Film with actor

Film belongs to genre

Film receives a review

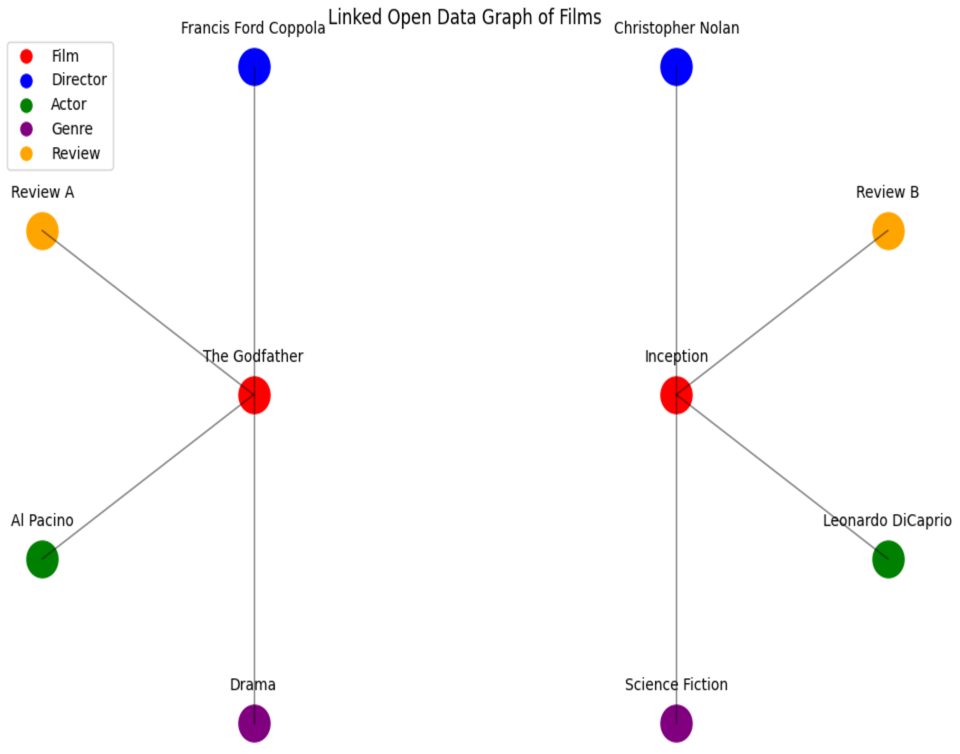


Figure 2- 5:Linked open data graph for films

- **Path distance calculation**

The system calculates the path distance between "The Godfather" and the movies in the database. For example, the path distance between "The Godfather" and "Goodfellas" is 1, because they are made by the same director. The path distance between "The Godfather" and "Taxi Driver", another Scorsese film, is 2, because there is an indirect connection via the director.

Depth-first search (DFS) or breadth-first search (BFS) algorithm to find the shortest path between movies.

- **Neighborhood similarity**

For each film to recommend, the system identifies its neighbors in the LOD graph. A movie's neighbors are the other movies that are linked to it by a defined number of links (for example, 1 or 2 links).

For example, "The Godfather" 's neighbors might include "Goodfellas," "Taxi Driver," "Casino," "The Good, the Bad and the Ugly" and other gangster or Scorsese films.

The system calculates the similarity between "The Godfather" and the films to recommend based on the similarity of their neighbors. A film with neighbors more similar to "The Godfather" is considered more similar.

The system retrieves movie attributes such as genre, release year, actors, director, reviews, etc.

For example, attributes for "The Godfather" might include "gangster", "1972", "Al Pacino", "Francis Ford Coppola", "positive reviews", etc.

The system calculates the similarity between "The Godfather" and recommended movies based on the similarity of their attributes. A film sharing more common attributes with "The Godfather" is considered more similar.

- **Nearest movie recommendation**

Select movies with the shortest path distance from "The Godfather". Consider neighborhood similarity and attribute-based similarity to refine recommendations.

Offer the user a list of personalized films, highlighting films with multiple paths or paths passing through valued entities (for example, an actor or a director).

3.1.4 Advanced Networks

There are a number of popular, consumer-facing knowledge graphs that define user expectations for enterprise search systems. Here are some examples of these knowledge graphs:

3.1.4.1 DBpedia

In 2007, Sören Auer, Jens Lehmann of the University of Leipzig and Christian Bizer of the University of Mannheim (formerly FU Berlin), with the support of OpenLink Software, initiated a project to extract structured information from Wikipedia, called infoboxes, to represent them in RDF (Resource Description Framework) or even in LOD form. This project, called DBpedia, is a collaboration between academia and the online community, and represents a major initiative in automated data extraction. Its main objective is to provide a structured and standardized version of Wikipedia content in the semantic web format. Additionally, DBpedia aims to establish links between Wikipedia and other open data sets on the Web of Data. Designed as one of the pillars of open data and the Web of data, DBpedia aspires to become an essential gateway to this new digital ecosystem.

Figure 2- 6: logo of the three entities

The new DBpedia knowledge base describes more than 3.4 million entities, of which 1.47 million are classified into a coherent ontology, including 312,000 people, 413,000 places, 94,000 music albums, 49,000 movies, 15,000 video games, 140,000 organizations, 146,000 species and 4,600 diseases. (...) The complete DBpedia knowledge base contains more than a billion pieces of information (triple RDF). Its version 3.5 of DBpedia was launched on April 10, 2010, highlighting its achievements.

3.1.4.2 Wikidata

Launched in October 2012 primarily to support Wikipedia and other Wikimedia projects, is a huge collaborative database managed by the Wikimedia Foundation, which hosts Wikipedia (Vrandečić & Krötzsch, 2014). This platform allows contributors from around the world to add factual information on millions of knowledge objects in diverse fields, ranging from general knowledge to biology to physics.

Unlike Wikipedia, Wikidata stores structured data and metadata in the form of triples (subject, predicate, object), making it easier to create links between different entities and query them automatically (Vrandečić & Krötzsch, 2014). Used by several Wikimedia projects, notably Wikipedia, Wikidata enriches articles by providing them with precise and complete information and accessible in many languages.

Unlike other databases, Wikidata goes beyond simply collecting data and its relationships. It also stores identifiers linking to external data sources and references supporting specific claims. This data structure, accessible to humans and machines, allows Wikidata to be used by various computing tools and websites (Vrandečić & Krötzsch, 2014).

Wikidata continues to grow thanks to daily contributions from its users around the world, as of April 2022 it has more than 97 million data elements, almost half of which are linked to scientific articles. A study conducted by Cobb in 2020 identified that of the more than 84 million items present at the time, 38.7 million were intended for scientific articles. As contributors around the world add content, the number of scientific work entries, their authors, and associated linked data elements increases daily.

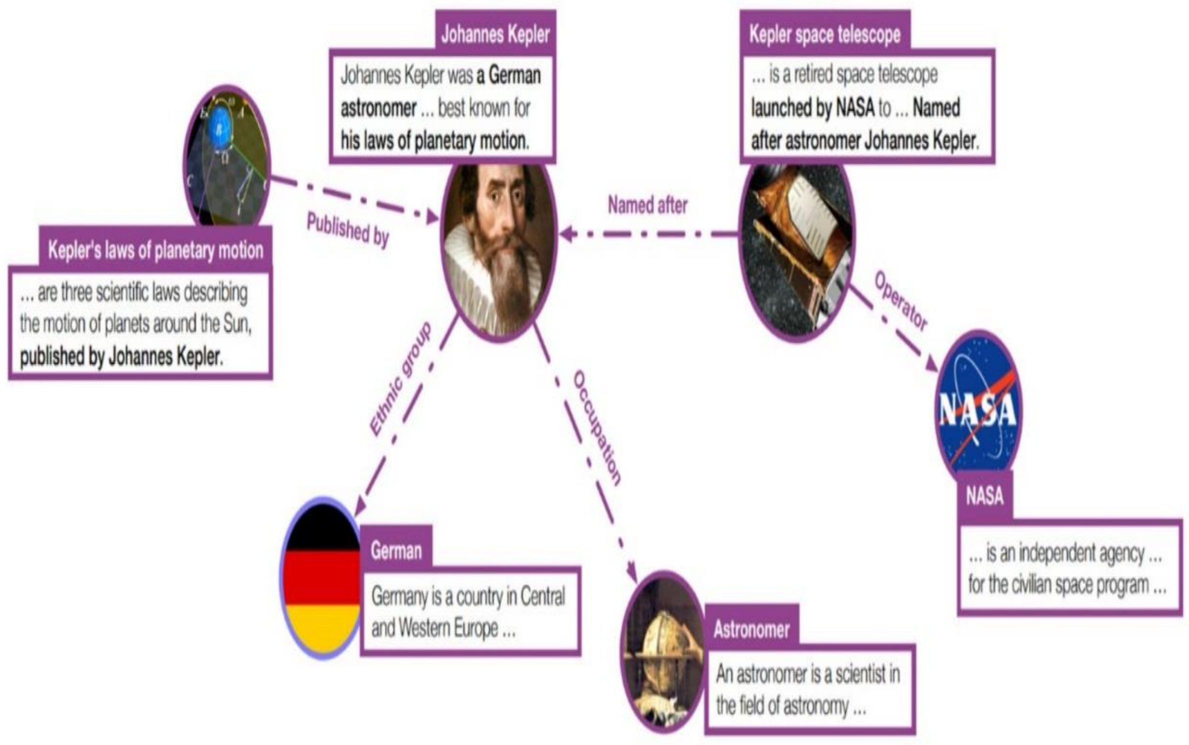

Johannes Kepler
Johannes Kepler was a German astronomer ... best known for his laws of planetary motion.
Kepler space telescope
... is a retired space telescope launched by NASA to ... Named after astronomer Johannes Kepler.
Published by
Named after
Kepler's laws of planetary motion
... are three scientific laws describing the motion of planets around the Sun, published by Johannes Kepler.
Operator
Ethnic group
Occupation
NASA
NASA
... is an independent agency ... for the civilian space program ...
German
Germany is a country in Central and Western Europe ...
Astronomer
An astronomer is a scientist in the field of astronomy ...

Figure 2- 7: Linked data visualization example

By comparison between DBpedia and Wikidata, we will say that they are two different knowledge graphs used for Wikipedia.org data. DBpedia is made up of data from Wikipedia's infoboxes, while Wikidata focuses on secondary and tertiary objects. Both are usually published in an RDF format. DBpedia and Wikidata are examples of large linked open data sets.

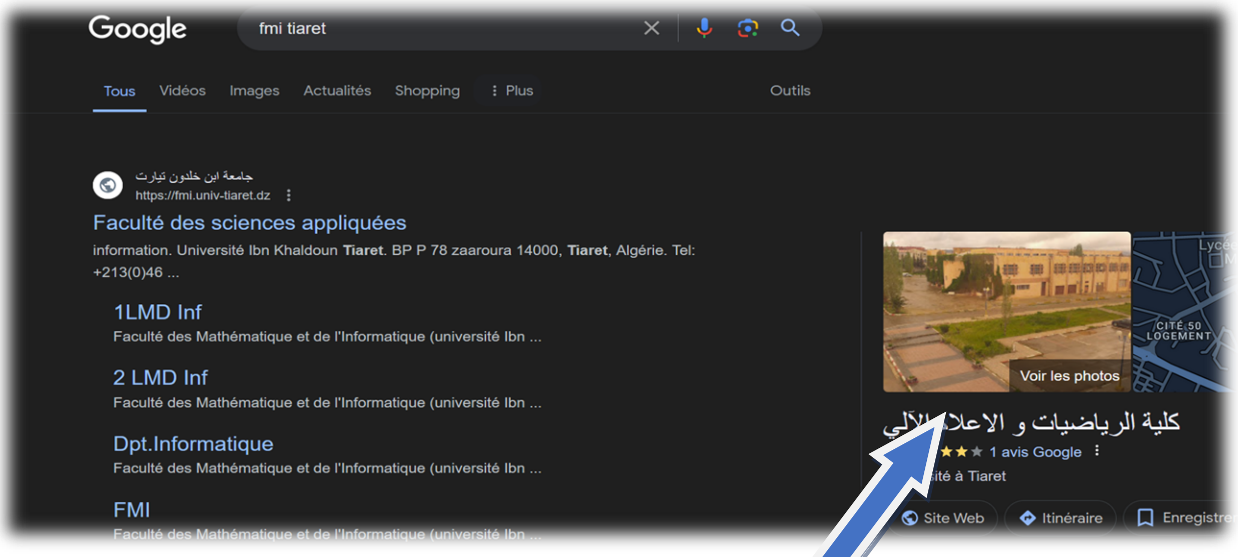
3.1.4.3 Google Knowledge Graph

Google launched its Knowledge Graph on May 16, 2012 with the aim of improving the quality of search results and the project is powered in part by Freebase (Singhal, 2012).

In August 2014, the Knowledge Vault project was launched, aiming for the automatic collection of information from across the Internet to answer direct questions. It was reported that the Vault had collected over 1.6 billion facts, of which 271 million were considered “trust facts.” (Dong et al., 2014).

Unlike the Knowledge Graph, the Knowledge Vault automatically gathered information instead of relying on crowdsourced data. Google extracts relevant information and presents it as an infobox alongside its search results. This feature allows users to quickly access instant responses.

Knowledge Graph data is automatically generated from a multitude of sources, covering a diverse range of topics such as locations, people and companies. Since its launch, the size of information contained in Google's Knowledge Graph has grown significantly, tripling in just seven months to include 570 million entities and 18 billion facts. In 2016, Google claimed to have 70 billion facts and answer about a third of the 100 billion monthly searches performed. By May 2020, this figure had increased to 500 billion facts covering 5 billion entities (Singhal, 2012).
Although there is no official documentation on how the Google Knowledge Graph is implemented, Google states that its information comes from a variety of sources, including the CIA World Factbook, Freebase, and Wikipedia. This knowledge base is used to answer questions asked directly to Google Assistant and voice queries from Google Home.



جامعة ابن خلدون تيارت
https://fmi.univ-tiaret.dz

Faculté des sciences appliquées

information. Université Ibn Khaldoun **Tiaret**. BP P 78 zaaroura 14000, **Tiaret**, Algérie. Tel: +213(0)46 ...

1LMD Inf
Faculté des Mathématique et de l'Informatique (université Ibn ...

2 LMD Inf
Faculté des Mathématique et de l'Informatique (université Ibn ...

Dpt.Informatique
Faculté des Mathématique et de l'Informatique (université Ibn ...

FMI
Faculté des Mathématique et de l'Informatique (université Ibn ...

كلية الرياضيات و الاعلام الآلي
★★★ 1 avis Google
ité à Tiaret

Figure 2- 8: Google Infobox

3.2 Extension of graphs (knowledge graph embedding)

This passage highlights the evolution of methods for manipulating knowledge graphs (KG) on a large scale. According to Wang et al. (2017), although RDF triples are effective for structuring knowledge, they often make KGs difficult to manage, especially when they are of considerable size. To address this, knowledge graph embedding (KGE) models have emerged. These models aim to project the entities and relationships of a knowledge graph into vector spaces, thus facilitating their manipulation while preserving their semantics.

In essence, as highlighted by Nickel et al. (2012), KGEs seek to ensure that similar elements in the knowledge graph are close in vector space. These models, often based on machine learning techniques and associated with specific objective functions, generate vector representations of entities and relationships. These representations can then be used for various tasks, such as KG completion, relationship extraction, and entity classification (Socher et al., 2013).

Link prediction and knowledge graph completion are perhaps the best-known uses of KG embeddings. Although KGs store vast amounts of data, they are often incomplete.

For example, given the KG in Figure 13, which is an extract from DBpedia, it will not be possible to answer the following questions

Q1: Where is Berkshire located?
Q2: What is the nationality of Daniel Craig?

Answering Q1 requires predicting the missing entity in the triple

< dbr: Berkshire, dbo: locatedIn?>.

Likewise, for Q2, the nationality of Daniel Craig would have to be deduced from the information available in the KG. The efficiency of question-answering applications based on KGs can therefore be improved by using embeddings to predict missing links in a KG, which we call KG completion (Nickel et al., 2016). Other applications of KG embeddings include similarity search, entity classification, recommender systems, semantic search, and question-answering (Wang et al., 2017).

Additionally, embedding converts symbolic knowledge into numerical representations, allowing structured knowledge to be incorporated into machine learning and AI models, enabling reasoning through KGs (Bordes et al., 2013). Although promising KG embedding models are widely used in various applications, there is potential to learn improved embeddings covering an even wider range of input information and opening new opportunities. For example, one can consider additional signals in the KG beyond structural information, such as multimodal and hierarchical information, as well as external textual data, or information related to a certain domain or context (Ji et al., 2021).

Some models struggle to adequately represent rare or long-tail entities, while others are unable to cope with little or no training data. Additionally, there is potential to design models that better account for dynamic and temporal information in the KG (Kazemi et al., 2020). Likewise, KGs are often multilingual, which can allow for improved representations. Most models also lack explicit interpretability or explainability (Chen et al., 2020).

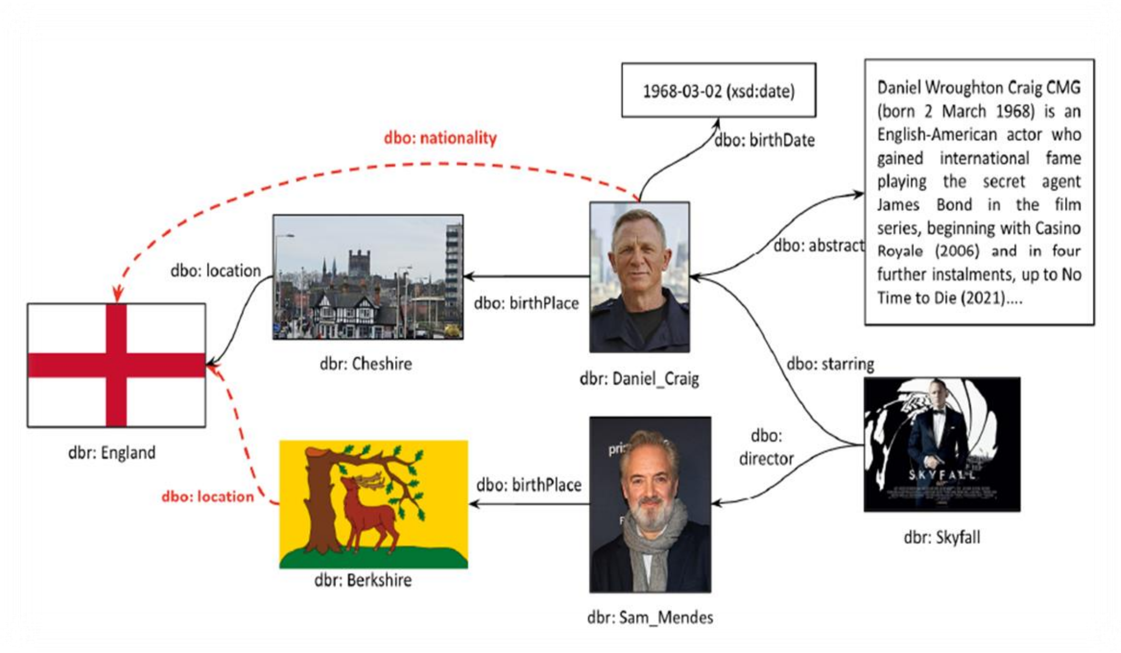

1968-03-02 (xsd:date)
dbo: birthDate
dbo: nationality
Daniel Wroughton Craig CMG (born 2 March 1968) is an English-American actor who gained international fame playing the secret agent James Bond in the film series, beginning with Casino Royale (2006) and in four further instalments, up to No Time to Die (2021)....
dbo: abstract
dbo: location
dbo: birthPlace
dbr: Cheshire
dbr: Daniel_Craig
dbr: England
dbo: starring
dbo: director
dbo: location
dbo: birthPlace
dbr: Berkshire
dbr: Sam_Mendes
dbr: Skyfall

Figure 2- 9: Deduction from graph extension information

3.3 Model based on the completion of a knowledge graph

Knowledge graphs represent structured information about entities and their rich relationships. Although a typical knowledge graph may contain millions of entities and billions of relational facts, it is usually far from complete. Knowledge graph completion aims to predict missing relationships between entities present in the graph. This process allows the discovery of new relational facts, which constitutes an important complement to the extraction of relations from simple texts.

3.3.1 Challenges of completing knowledge graphs

Knowledge graph completion is similar to link prediction in social network analysis, but it presents additional challenges

3.3.1.1 Heterogeneity of entities

Entities in a knowledge graph can have different types and attributes, which implies more complex representations (Wang et al., 2019).

3.3.1.2 Diversity of relationships

The relationships in a knowledge graph can be of different types, ranging from simple 1-to-1 relationships to more complex N-to-N relationships (Wang et al., 2019).

3.3.1.3 Relationship Type Prediction

In addition to determining the existence of a relationship, knowledge graph completion also aims to predict the exact type of relationship between entities (Wang et al., 2019).

3.3.2 Vector embedding approaches

Recently, embedding knowledge graphs in continuous vector spaces has shown promise for knowledge graph completion. This approach makes it possible to capture the semantic relationships between entities and relationships and project them into a d-dimensional vector space.

3.3.2.1 TransE and TransH

Trance (Bordes et al., 2013) whose fundamental idea is that the relationship between two entities corresponds to a translation between the embeddings of the entities, that is to say h + r ≃ t when (h, r, t) is true and TransH (Wang et al., 2014) with each triplet (h, r, t) of entities in the space is first projected into the relation space r in the form hr and tr with the Mr operation, then hr + r ≃ tr.

These are simple and efficient vector models for completing knowledge graphs. They learn vector representations for entities and relationships, assuming that the relationship between two entities corresponds to a translation between their vector representations.

Modeling entities and relationships in distinct spaces TransR (Lin et al., 2015) proposes to model entities and relationships in distinct spaces: the entity space and relationship-specific entity spaces. This approach helps to better capture the nuances of relationships and model the different aspects of the entities involved in different relationships.

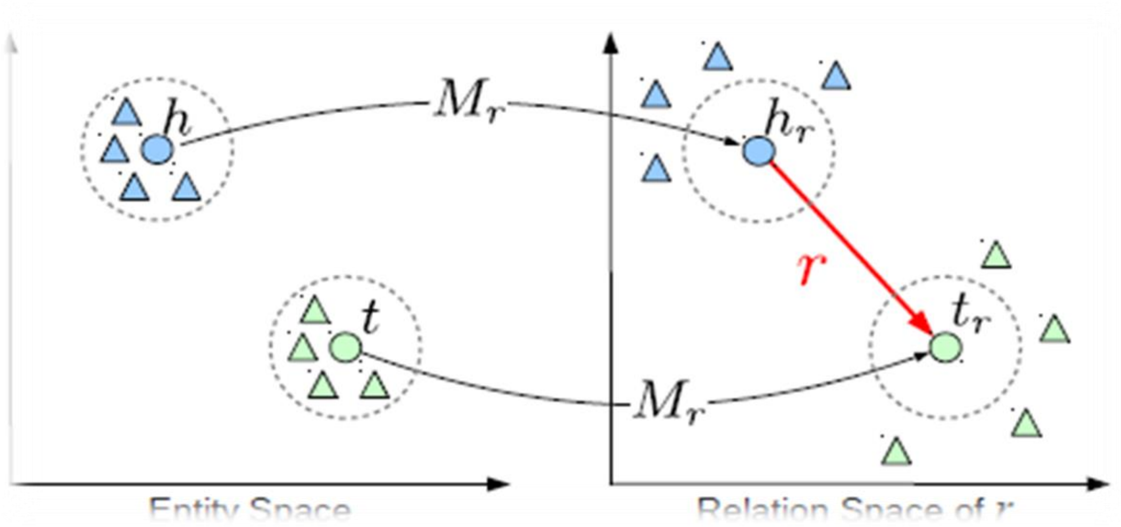


Figure 2- 10: simple illustration of transR

3.3.3 Application of knowledge graphs in some areas:

However, knowledge graphs also have applications in many sectors including:

3.3.3.1 Distribution

Knowledge graphs have been used for upselling and cross-selling strategies, recommending products based on individual purchasing behavior and popular purchasing trends across demographic groups.

3.3.3.2 Leisure sector

Knowledge graphs are also used by artificial intelligence (AI)-based recommendation engines for content platforms, such as Netflix, social media. Based on clicks and other online engagement behaviors, these providers recommend new content for users to read or watch.

3.3.3.3 Finance

This technology has also been used for know-your-customer (KYC) and anti-money laundering initiatives in the finance sector. Knowledge graphs contribute to the investigation and prevention of financial crime, allowing banking institutions to understand the flow of their customer funds and identify non-compliant customers.

3.3.3.4 Health care

Knowledge graphs also benefit the healthcare industry by organizing and classifying relationships in medical research. This information helps providers validate diagnoses and identify therapies based on individual needs.

4. Conclusion

This chapter introduces the use of bipartite graphs to model recommendation systems, highlighting their advantages for capturing complex relationships between users and items (Newman, 2003). Through the exploration of knowledge graphs and embedding techniques, it demonstrates how these approaches can improve the precision and relevance of recommendations (Nickel et al., 2016). Challenges, such as the heterogeneity of entities and the diversity of relationships, are also addressed, highlighting the complexity of this approach but also its significant potential (Wang et al., 2017). Future research could focus on improving the models' ability to capture more complex relationships and manage large-scale knowledge graphs (Ji et al., 2021).

Chapter 3

Collaborative Prediction Based on graphs

1. Introduction

In this chapter, we present the general architecture of the system composed of several phases, then we highlight the conceptual side of our application which constitutes a fundamental step which precedes implementation, by detailing the different scenarios to be implemented in the following phase. This will allow us to better understand our system. This presentation will explore an innovative movie recommendation system, based on similarity between neighbors (collaborative filtering). We will then examine the key principles of this system through a method centered on the use of bipartite graphs and how recommendation can offer personalized and relevant movie suggestions to users.

2. Context and challenges of film recommendation systems

The cinema entertainment industry is experiencing remarkable expansion, with increasing production of films and diversity in genres and styles. With this abundance of choice, viewers may have difficulty finding films that suit their tastes and interests. It is in this context that movie recommendation systems play a crucial role in helping users discover new movies they will enjoy without making huge efforts through endless searches for movies to watch later (Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. 1994).

Movie recommendation systems are computer tools that analyze movie and user data to provide personalized suggestions. The data is collected in different ways either by an approach based on the characteristics of films appreciated, rated, viewed or others, which fall within the framework of filtering by content, or by similarity which differentiates itself with its way of targeted collaboration or even by hybridization which take care of the combination of several filters for a better recommendation with precision and of course the outcome of all these approaches is for the satisfaction of moviegoers.

Movie recommendation systems provide many benefits to users, streaming platforms and movie studios. For users, they allow them to discover new films and improve their viewing experience. For streaming platforms, they increase user engagement and reduce the churn rate. For film studios, they allow them to promote their films and increase their revenues.

3. Problem & Objectives

In an ever-expanding cinematic landscape, movie fans are faced with a challenge of choice, searching for films that match their preferences from a plethora of offerings. Movie recommendation systems aim to address this need, but they are hampered by limitations such as data bias, cold start issues, and low recommendation diversity.

How to design a movie recommendation system that improves user convenience, while solving the cold start problem through categorization filtering and user engagement to provide a more personalized, diverse, and satisfying movie experience for users?

How to facilitate user interactions with items (movies) using bipartite graphs for recommendation?

The main objective of our work is to help cinema lovers feel good or comfortable when watching films by allowing them to select films from collective top-ratings which brings together all the ratings made by everyone. users, or personal rating panels based on similar users who themselves are involved in the development of their own rating panels which revolve around similar users only, to allow the resolution of the cold start problem by filtering by categorization of films or the user goes through the top ratings, which also takes into account the problem of lack of data, integrating bipartite graphs to model the relationships between users and films (items).

4. Overall system architecture

The movie recommendation system that we are going to propose is based on a robust and scalable architecture. It mainly includes three essential parts:

- Collection and processing of user data,
- Calculating similarities between users,
- And a recommendation engine optimized to offer the most relevant films.

We are interested here in states where the user is either a novice or one already confirmed by the system, in addition to filtering by similarity and filtering based on content (category or characteristics) to ensure the best quality of the recommendation.

This architecture allows for seamless integration with other data sources (API call), such as movie information, to constantly improve recommendations and user experience.

We present the general architecture of this system in the diagram below.

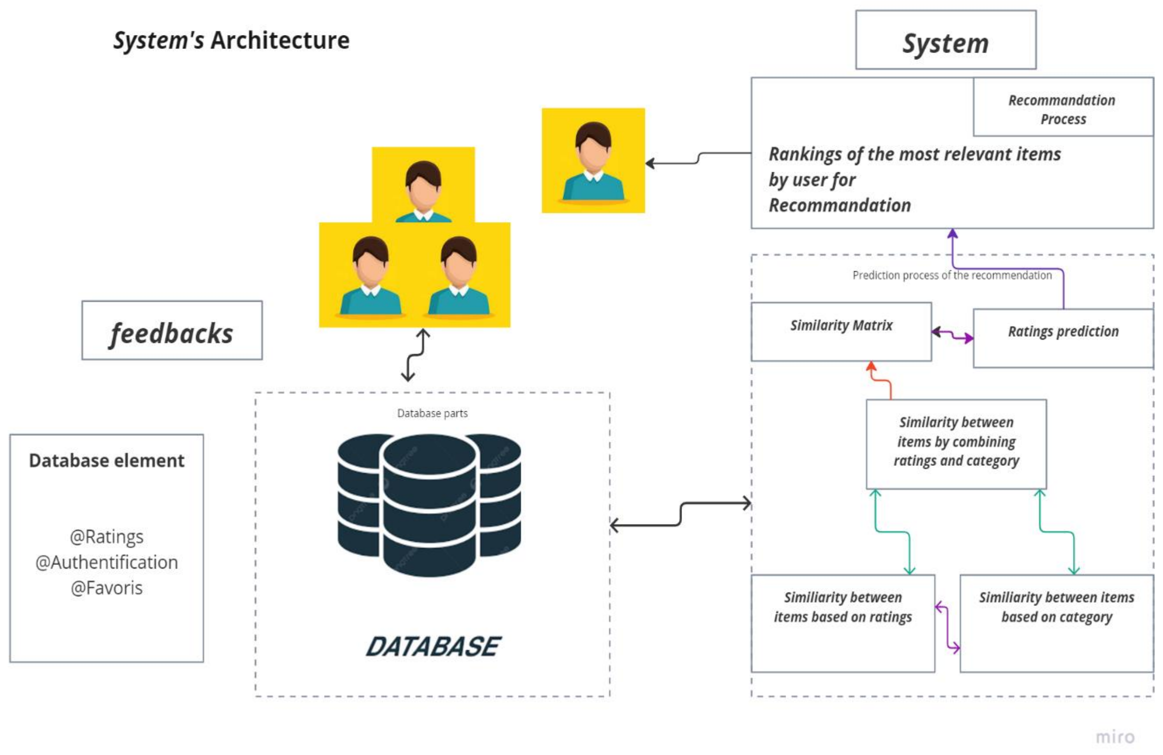

System's Architecture
System
Recommandation Process
Rankings of the most relevant items by user for Recommandation
Prediction process of the recommendation
Similarity Matrix
Ratings prediction
Similarity between items by combining ratings and category
Similiarity between items based on ratings
Similiarity between items based on category
feedbacks
Database element
@Ratings
@Authentification
@Favoris
Database parts
DATABASE
miro

Figure 3- 1: Overall architecture of our system

4.1 System operation

4.1.1 Cold start case

Lack of information about user expectations through recommender systems, especially when recorded, can reduce the quality of recommendations and their suitability to user preferences. This will quickly discourage this new user and make him abandon the system as was highlighted by Prof. **OUARED ABDELKADER** (professor at Ibn Khaldoun University of Tiaret) during a practical session on user experience in mobile applications.

So, we took precautions in particular by allowing the user to filter the information by alphabet letter for example if a user only knows a film by name and he forgets a good part of the name then he filters directly from the letter that begins the name of the film.

To have more precision there is also another part which concerns filtering by category which allows the user to sort the film by category (action, documentary, etc.)

In addition to these alternatives for a novice of the system, we have thought of top-ratings to allow the user to start well from films rated by other users even if the recommendation is not yet personalized.

4.1.2 Case of data scarcity

We have a part that will allow the new user to first start watching before giving a rating which will then facilitate the recommendation.

4.1.3 Case of a user already having ratings

Using other users' rating data, we can try to predict the ratings a user would give to a movie they haven't yet seen. This involves identifying its "neighbors" with similar tastes, the data is then organized and formatted so that it can be used effectively by the filtering algorithm and the calculation of the similarity between their respective profiles is carried out. We will thus be able to estimate the ratings that the user would give to other films and recommend them accordingly.

As presented in a previous section, the recommender systems we consider in this study include:

- The approach based on the popularity of items (Top-*ratings*),
- The collaborative filtering approach based on the similarity between items and users,
- The hybrid approach which combines the previous two.

In our movie recommendation system, every user interaction is meticulously tracked to deliver a personalized and rewarding viewing experience. The user starts by watching a video and then rates the movie based on their satisfaction level, providing a direct indication of their preferences. After watching a movie, the user can add it to their favorites list or viewing history, which allows us to better understand their tastes and interests. If the rating given by the user is satisfactory, our system automatically offers recommendations of similar films, taking into account their preferences. However, if the rating is below a predefined threshold, the system does not issue a recommendation unless the supported rating margin is reached, ensuring that only films meeting the user's expectations are recommended. At the same time, we analyze the different ways in which users select films, whether by exploring categories or consulting top ratings, in order to better understand their preferences in terms of genres, styles and popularity of films. Also using collaborative filtering techniques, we recommend movies based on similarity to other users with similar tastes, expanding the scope of recommendations to provide optimal variety and relevance. Through this comprehensive approach, our system aims to offer each user a personalized and enriching cinematic experience.

4.2 Calculation of Similarities between Users

Initial data:

	Item	**i 1**	**i 2**	**i 3**	**i 4**	**i 5**
User						
u 1		**5**	🟩	**0.5**	**1**	**5**
u 2		**2**	**5**	🟩	**4**	🟩
u 3		**4**	**3**	**1**	**4**	**5**
u 4		**4**	**1**	**5**	🟩	🟩
u 5		**5**	**1**	**4**	🟩	**1**

Table 3- 1:An example of the evaluation matrix table

Let's recap: Formally, note history is defined as follows:
Each user $u \in U = \{u_1, u_2 \ldots \ldots, u_n\}$ rated a set I_u *inclus* $I = \{i_1, i_2 \ldots \ldots, i_n\}$ of item(s). Each of the notes $r_{u,i} \in I_u$ concerning an item can be presented in binary form (like/dislike) or real. Our example is similar to the GroupLens corpus where the ratings are real and lie between 1 (bad) and 5 (good).

4.2.1 Context of the bipartite graph

The evaluation matrix model proposed above will be based on the use of graphs. The latter have recently proven their effectiveness, in this context, in the sense that they can teach us more latent information about the nodes and the relationships between them. A bipartite graph will be constructed based on the evaluation matrix M: $U \times I$ with two types of nodes: users and items as mentioned in the table.
We will begin by listing the elements to be addressed on the BDD evaluation matrix. Indeed, this matrix contains the users, the items, the notes (ratings) that the users have given to the items (Goldberg, D., Nichols, D., Oki, B. M., & Terry, D. 1992).
To do this, we extracted the identifier of each user, the identifier of each item as well as the ratings given to the items by these users. This results in an adjacency matrix of a bipartite graph linking each user to the list of items he has rated.

4.2.1.1 Construction of the bipartite graph

From the matrix M: $U \times I$ resulting in the previous step we will generate a bipartite graph connecting each user to the set of items that he noted according to the following rule:

An edge is generated between a user node and an item node if this user has evaluated this item through a rating.

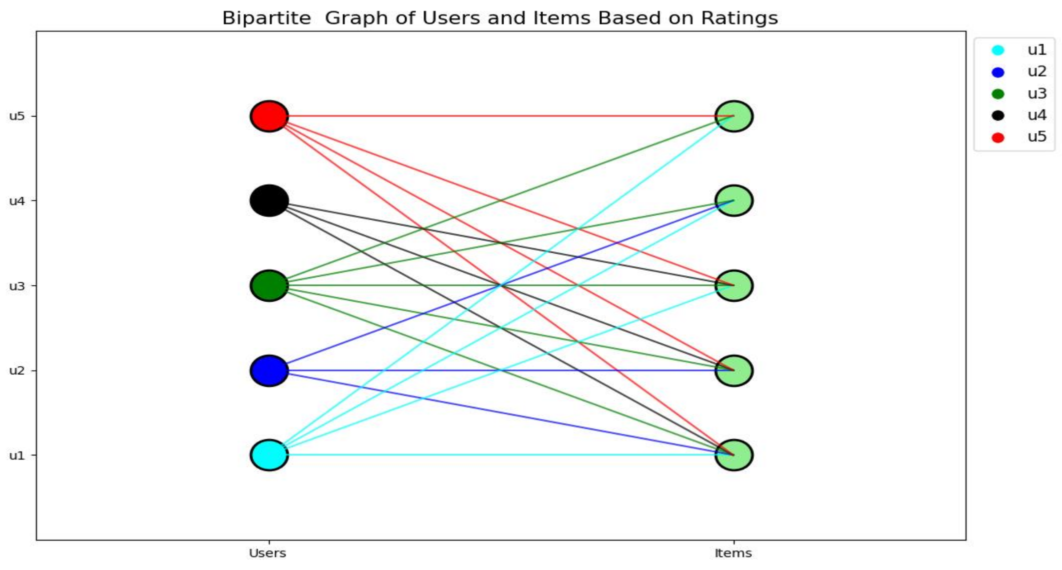


Figure 3- 2: An example of constructing a bipartite graph from an evaluation matrix

4.2.2 Similarity calculation

Data preprocessing

Once the data has been collected, it must be pre-processed before it can be used for a recommendation system. Data preprocessing can include the following steps:

- **Cleaning**: Data cleaning involves identifying and correcting errors, such as missing values, duplicates, and inconsistencies.
- **Standardization**: Data standardization involves putting all data on a common scale. This can be useful for some similarity measures.
- **Handling missing values**: Missing values can be imputed (estimated) or removed.
- **Data transformation**: Data can be transformed into a format more suitable for the recommendation system. For example, dates can be transformed into timestamps and texts can be transformed into numeric vectors.

For all the methods used for calculating similarity we will use the following:

- **Cosine similarity**

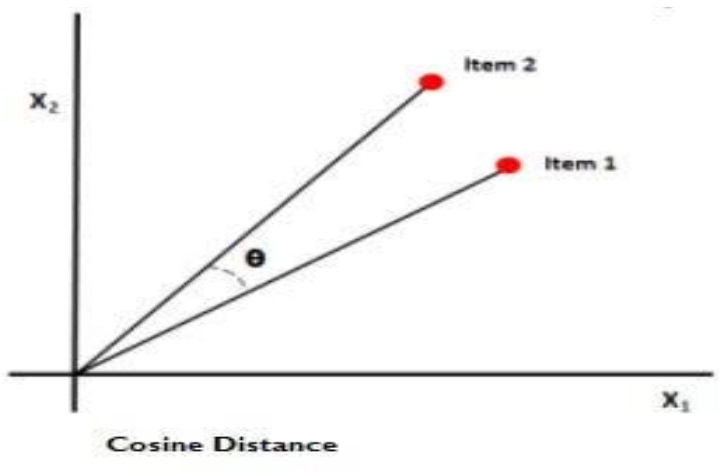


Figure 3- 3: Distance to get the cosine from table 5

	Item	**u 1**	**u 2**	**u 3**	**u 4**	**u 5**
User						
u 1		**1**	**0.344**	**0.845**	**0.622**	**0.724**
u 2		**0.344**	**1**	**0.528**	**0.938**	**0.591**
u 3		**0.845**	**0.528**	**1**	**0.783**	**0.820**
u 4		**0.622**	**0.938**	**|0.783**	**1**	**0.778**
u 5		**0.724**	**0.591**	**0.820**	**0.778**	**1**

Table 3- 2: cosine similarity

The number of neighbors for each user can vary depending on the chosen similarity threshold.

A user can be a neighbor of several other users.

Exploitation of similarities and neighbors for recommendation

Cosine similarities and identified neighbors can be used for different recommendation tasks, such as:

- Rating prediction for items not rated by a user

By calculating the weighted average of similar item scores for the user, using similarities as weights.

- Recommendation of popular items among a user's neighbors

By identifying the items best rated by the user's neighbors.

- Recommendation of new or little-known items

By exploiting the knowledge of the user's neighbors to discover relevant items

User	**Neighbors (Cosine similarity ≥ 0.5)**
u 1	**u 3 (0.8452)**
u 2	**u 4 (0.9381)**
u 3	**u 1 (0.8452), u4 (0.7834), u5 (0.8204)**
u 4	**u2 (0.9381), u3 (0.7834), u5 (0.7780)**
u 5	**u1 (0.7237), u3 (0.8204), u4 (0.7780)**

Table 3- 3: Neighborhood demonstration between users

4.2.3 Community detection

4.2.3.1 Analysis of interactions

According to S. Fortunato: A community is a set of entities having a lot of interactions between them and little interaction with the outside world. Girvan and Newman define a community as a set of entities that have internal relationships more than external relationships. Radicchi et al. Improve this definition by the constraint that each individual in a community has more neighbors inside their community than outside. They call these structures strong communities.

A community is generally a social group whose members share a common living space or interests. This means that it is a collection of individuals with many relationships with each other within the group and fewer relationships with those outside.

Identifying communities within the user network is essential to understanding interaction patterns and affinities between individuals.

We chose to use community detection techniques (Fortunato and Hric 2016) which consist of a process of discovering cohesive groups of nodes in a graph, called community.

The set of nodes belonging to a community are strongly connected inside the community as well as outside.

4.2.3.2 Visualization of graphic results

The graphical representation of the discovered communities facilitates the analysis and understanding of social links within the recommendation system.

The communities that are formed are as follows:

- **Item 4 (i4) community**

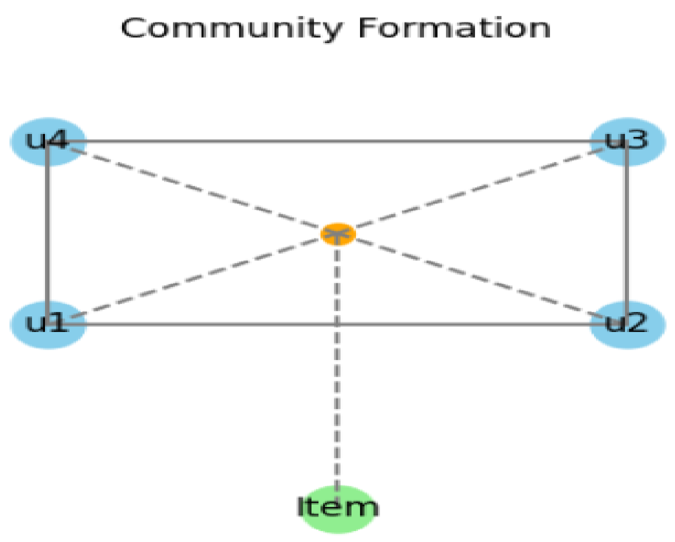


Figure 3- 4: Representation of the community formation of Figure 15

- **General idea of all communities**

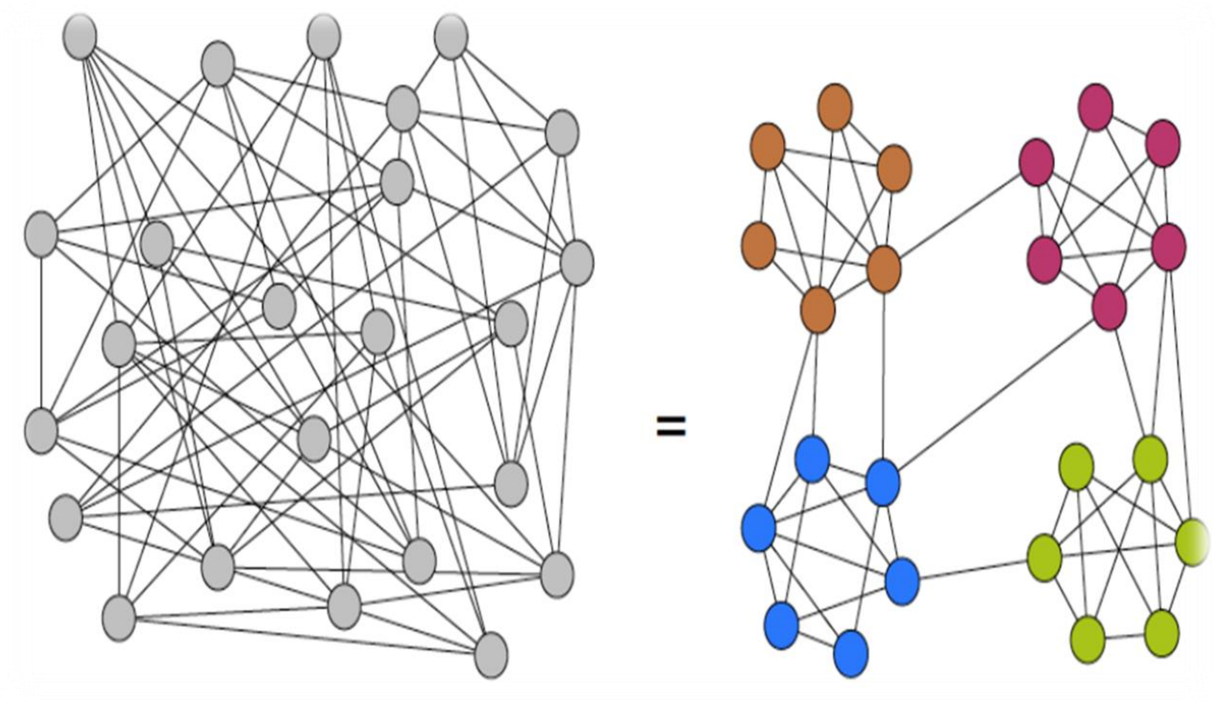


Figure 3- 5: Representation of the formation of all communities

4.2.4 Calculation of predictions

This phase, just like the previous ones, is of crucial importance since the objective of any collaborative filtering system is the calculation of predictions to generate relevant recommendations to an active user. The most used method for calculating these predictions is the "weighted sum" [Herlocker et al., 1999]. This method considers the nearest neighbors U_a(in correlation with the active user) having already rated the item to calculate the prediction of the $i_k$over $i_k$rated $Pred(u_a, i_k)$ratingu_a $Sim(u_a, u_b)$denotes the similarity value between $u_a$and a neighboru_b $(u_b|\in| U_a)$and perhaps instantiated by the similarities calculated from the Pearson coefficient $\left(CorrP\left((u_a, u_b)\right)\right)$or from the measure based on cosine $\left(Cos\left((u_a, u_b)\right)\right)$.

$$Pred(u_a, i_k) = \overline{v(u_a)} + \frac{\sum_{u_b \in U_a} Sim(u_a, u_v) * \left(v(u_b, i_k) - \overline{v(u_b)}\right)}{\sum_{u_b \in U_a} Sim(u_a, u_v)} \quad (17)$$

There are several strategies for choosing the most relevant neighbors in recommender systems:

Using a similarity threshold: This method consists of selecting the closest neighbors that have a strong correlation with the active user, using a predefined similarity threshold 【 Breese et al., 1998; Shardanand and Maes, 1995 】.

Selection of optimal neighborhood size: This approach allows choosing a fixed number of nearest neighbors, such as the 20, 50 or 100 best neighbors 【Herlocker et al., 1999】

Establishing a threshold for co-evaluated items: Here, the strategy consists of filtering the nearest neighbors based on the number of items they have rated in common with the active user 【Viappiani et al., 2006】.

Note: Once the predictions have been calculated, the system recommends to the active user the items with the highest prediction values.

According to the basic principle of collaborative filtering, users must provide their ratings on documents so that the system forms communities. Evaluating a recommendation can be done explicitly or implicitly by F. Maxwell Harper and Joseph A. Konstan. 2015, as follows.

Explicit: The user gives a numerical value on a given scale (1 to 5 for example) or a qualitative satisfaction value (for example: bad, average, good and excellent.)
Implicit: The system induces user satisfaction through its actions.
For example, the system will consider that a deleted recommendation corresponds to a very bad evaluation, while a recommendation viewed (several times) or added to the panel may be interpreted as a good evaluation.
It should also be noted that the recommendations that a user evaluates can be generated by the system and/or chosen by the user himself [MovieLens].

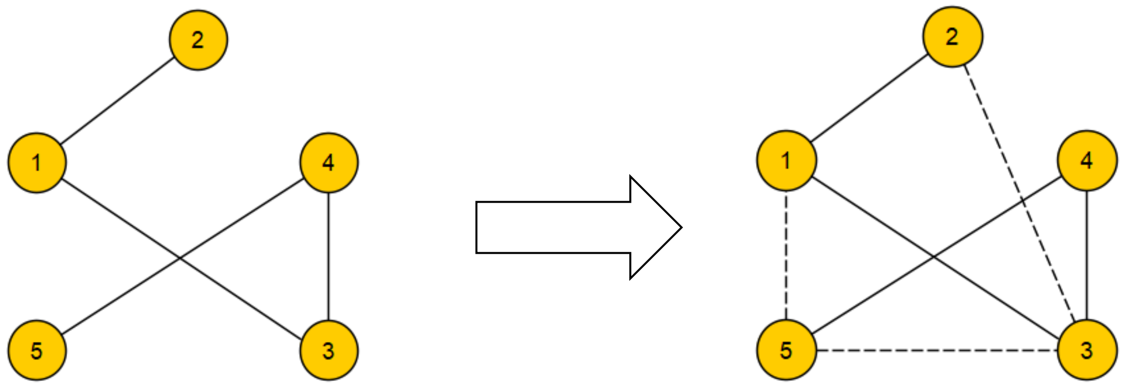


Figure 3- 6: Predicting links in a graph.[23]

4.2.5 Updating algorithms in a scalable manner

- Refine and update recommendation algorithms based on new data and user feedback.
- Implement an iterative process of testing and continuous improvement to ensure that the system meets user expectations.

To improve the accuracy and relevance of our recommendation system, we integrate additional data (as indicated in the functional but secondary needs section) from other sources such as movie reviews, trailers, information on actors and directors, as well as sociodemographic data of users.

This integration allows us to refine our recommendation algorithms and offer more personalized recommendations adapted to the tastes of each user.

5. Conclusion

This chapter details the architecture and methodologies for collaborative graph-based predictions, with a particular focus on movie recommendations. It presents techniques for calculating user similarities and community detection, demonstrating how these methods can be used to improve recommendation systems. By analyzing experimental and graphical results, this chapter shows the effectiveness of bipartite graphs in overcoming challenges like cold start and data sparsity, and concludes with recommendations for future improvements and research.

Chapter 4

Implementation and Experimentation

1. Introduction

Theory without practice can be called blind. This chapter relates to putting into practice the theories we have studied previously. Its purpose is to present practical illustrations of the film recommendation system that we will create at the end of our dissertation. We present the set of tools and environments, diagrams for the different actions performed and we end with some interfaces which present our system.

2. Analysis and design

Our movie recommendation system focuses on creating an intuitive and functional user interface, efficient administrative management, and a robust and intelligent recommendation engine to deliver a personalized and satisfying experience to all users.

2.1 Specification of system requirements

2.1.1 Functional requirements

Main features

- Recommend movies to users based on their rating and viewing history.
- Allow users to rate movies.
- Explore movies similar to a given movie.
- Recommend popular movies in top-ratings.
- Manage user preferences.

Secondary features

- Provide detailed information about movies, such as synopsis and reviews.
- Recommend movies to groups of users (e.g. friends, family).
- Integrate the system with other platforms (e.g. social networks).

2.1.2 Non-functional requirements

Non-functional needs are important because they act indirectly on the result and on the user's performance, which means that they must not be neglected, for this the following requirements must be met:

Reliability: The system must operate consistently without errors and must be satisfactory.

Errors: Ambiguities must be indicated by well-organized error messages to properly guide the user and familiarize them with the system.

Ergonomics and good interface: the site must be adapted to the user without requiring any effort (clear and easy to use).

Security: Our solution must above all respect the confidentiality of the personal data of customers who connect to the system

2.2 Unified Modeling Language UML:

UML, or Unified Modeling Language, is a visual modeling language widely used in the field of software engineering. It allows you to design, specify, visualize and document software systems using a variety of diagrams to represent different aspects of the system.

Standardized by the Object Management Group (OMG), UML offers a set of standardized symbols and notations that are widely accepted and used in the industry. As a communication tool, UML facilitates collaboration within software development teams and with stakeholders by providing a clear and intuitive view of the system, thus promoting understanding and discussion of different aspects of the project.

Some of the most commonly used diagrams include:

2.2.1 Class diagrams

A UML class diagram is a visual representation of the classes, attributes, methods, and relationships between classes in a system. It is an essential tool for modeling the static structure of a software system.

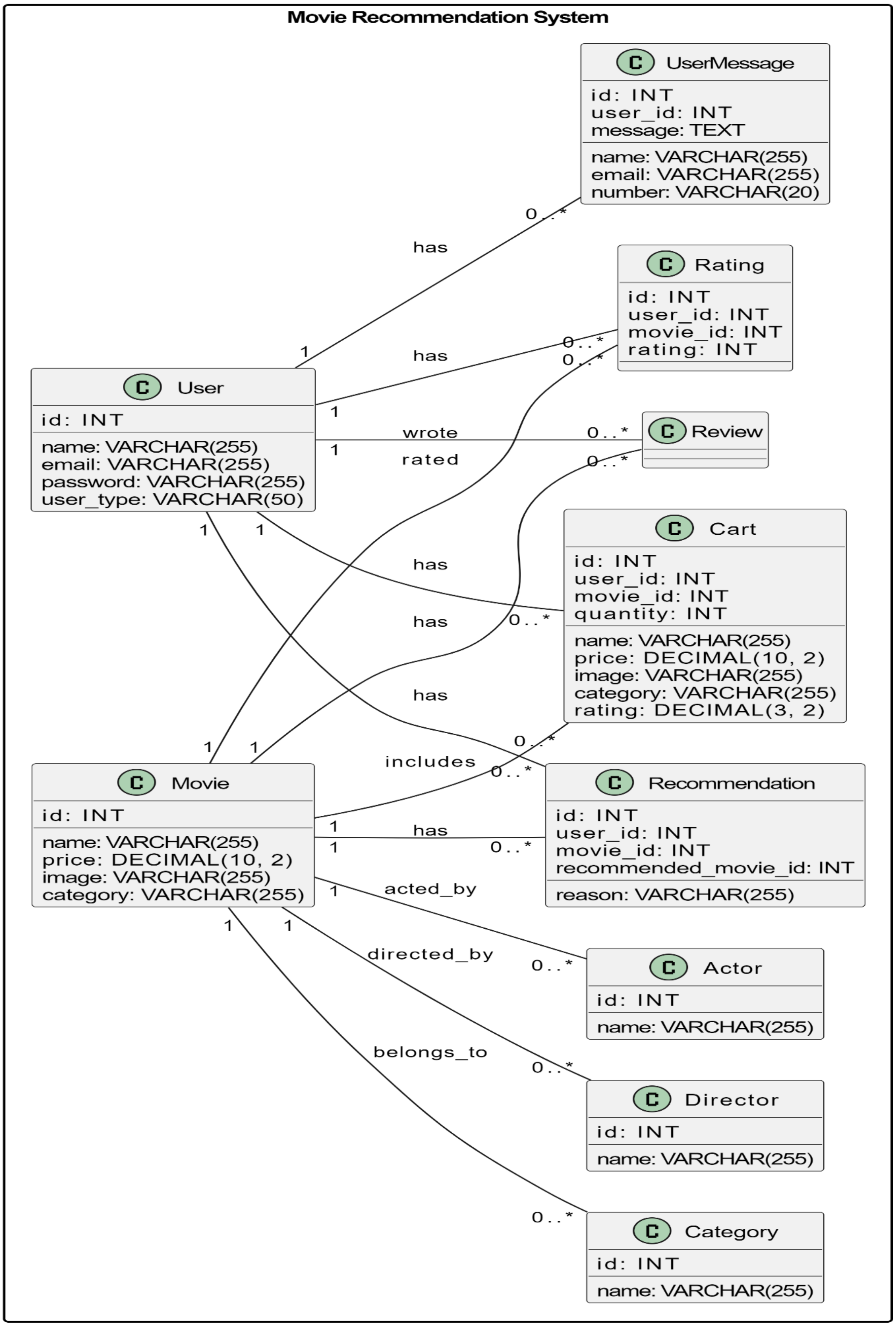


Figure 4- 1: class diagram

2.2.2 Use case diagrams

Use case diagrams are a type of Unified Modeling Language (UML) diagram used to represent interactions between users (or actors) and a system. They help to visualize the different functionalities offered by the system, as well as the relationships between these functionalities and the users.

2.2.2.1 Identification of actors

The actors of a system are the entities external to this system which interact with it. In our system, the actors who interact with the system are the system users and the administrator.

2.2.2.2 Identification of use cases

A use case is used to define the behavior of a system or the semantics of any other entity without revealing its internal structure. Each use case specifies a sequence of actions, including variations, that the entity performs, interacting with the entity's actors. The responsibility of a use case is to specify a set of instances, where a use case instance represents. A sequence of actions that the system carries out and which provides a result observable by the actor.

Here are the use cases of our system

- For the user

Authentication: The system verifies that the user is who they say they are and then gives them authorization to access its interface.

View Recommendation: User can see the list of recommended movies if it exists.

Choose a film: it selects a film from the list of films added by the admin or from the list of top-ratings.

View the film presentation: The system displays information about the selected film (category, year, description, etc.).

Watch Movie: Allows you to watch the movie.

Choose a movie by category: Allows the user to choose the category of movie they want to watch.

Choose a movie by: Allows the user to choose the letter of movie they want to watch.

Evaluate a film: The user can evaluate the film by giving it a rating in the range of 1 to 5.

Watch movies by their similarities: The user can see the list of movies based on their similarity in their own recommendation panel.

- For the administrator

Authentication: The system verifies that the user is who they say they are and then gives them authorization to access its interface.

Account management: delete users

Film management: it can control the content published on the platform (add and delete films).

- For the system

Recommend: Recommend based on similarity and top-ratings.

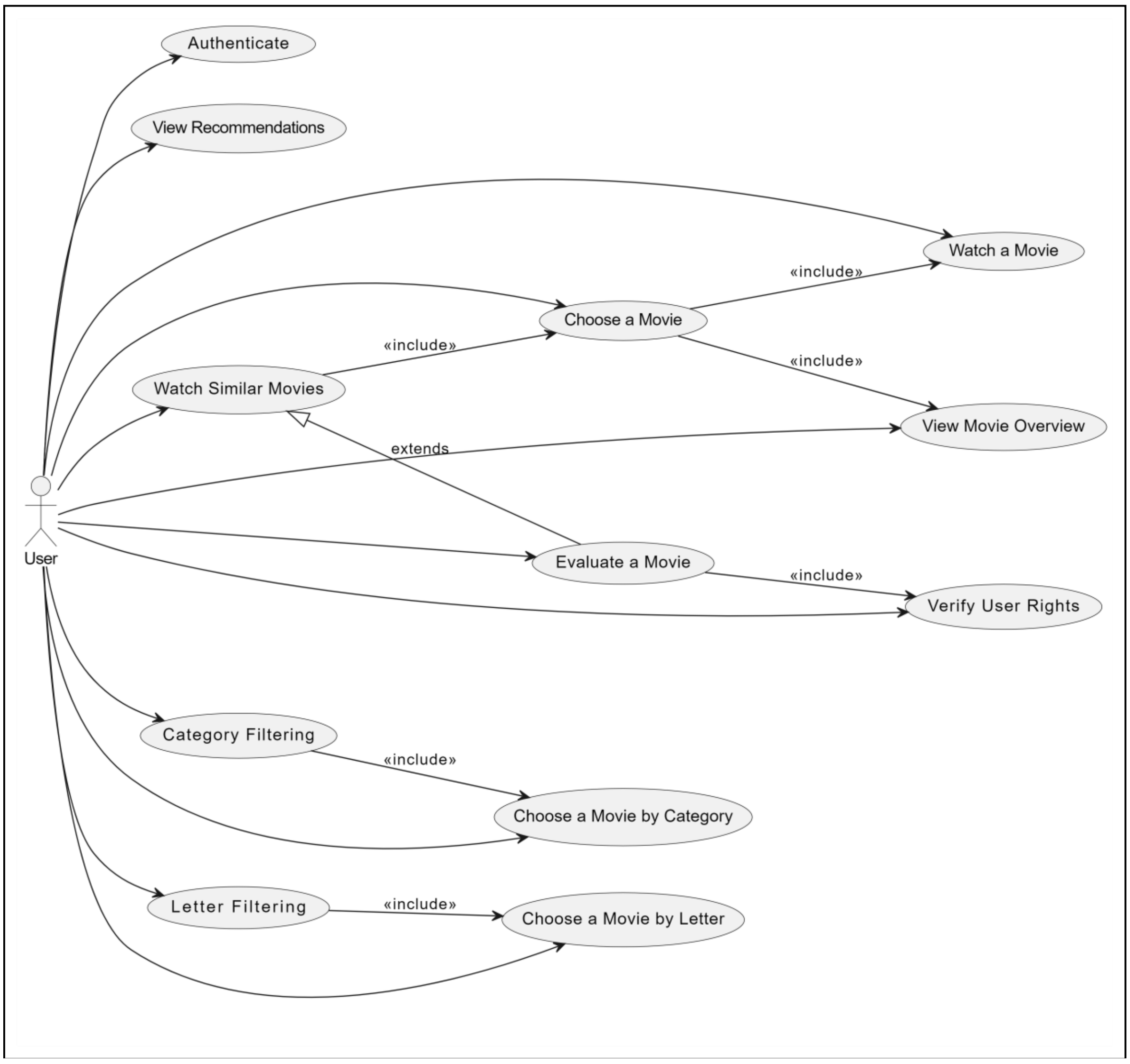


Figure 4- 2:actor use case

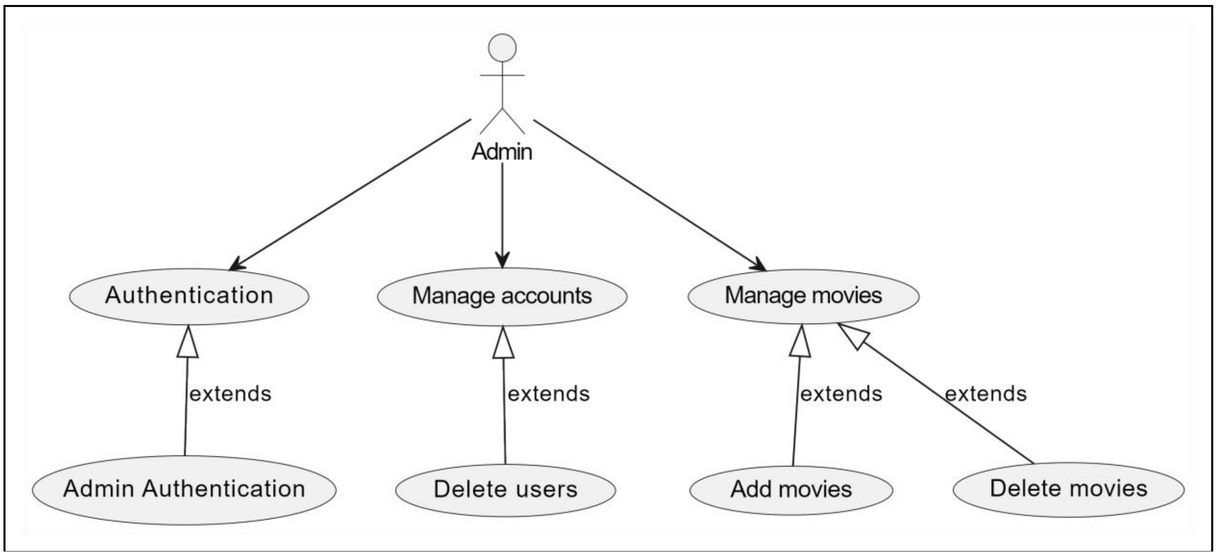


Figure 4- 3:Admin use case

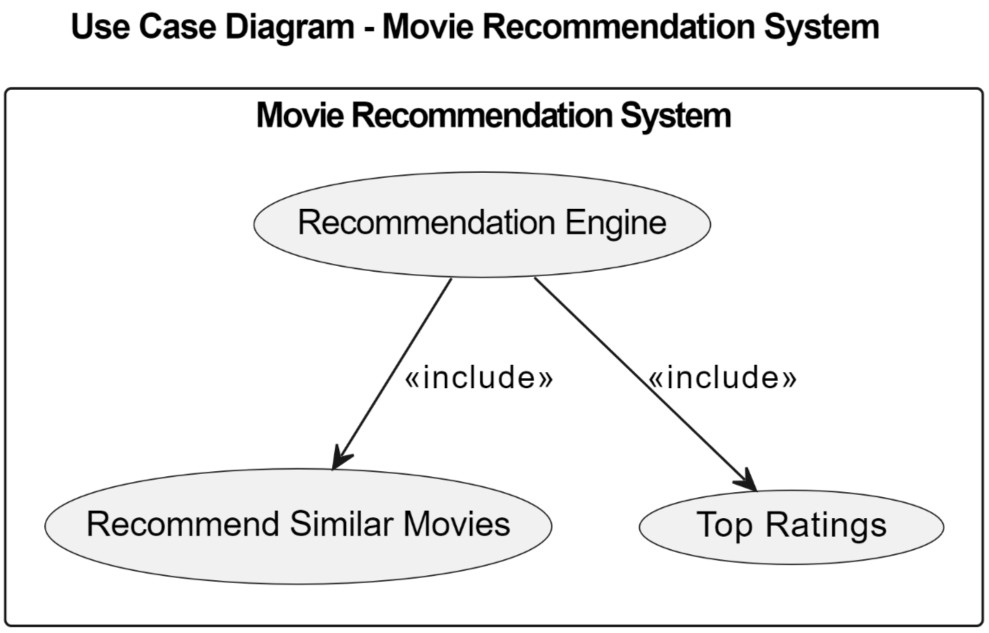


Figure 4- 4: Use case diagram

2.2.3 Sequence diagram

A sequence diagram in UML illustrates how objects interact in a system over time by showing the order of messages exchanged between them. It includes actors, lifelines, messages, and activations, read from top to bottom to represent the flow of time and actions. This helps visualize and design the dynamic behavior of a system.

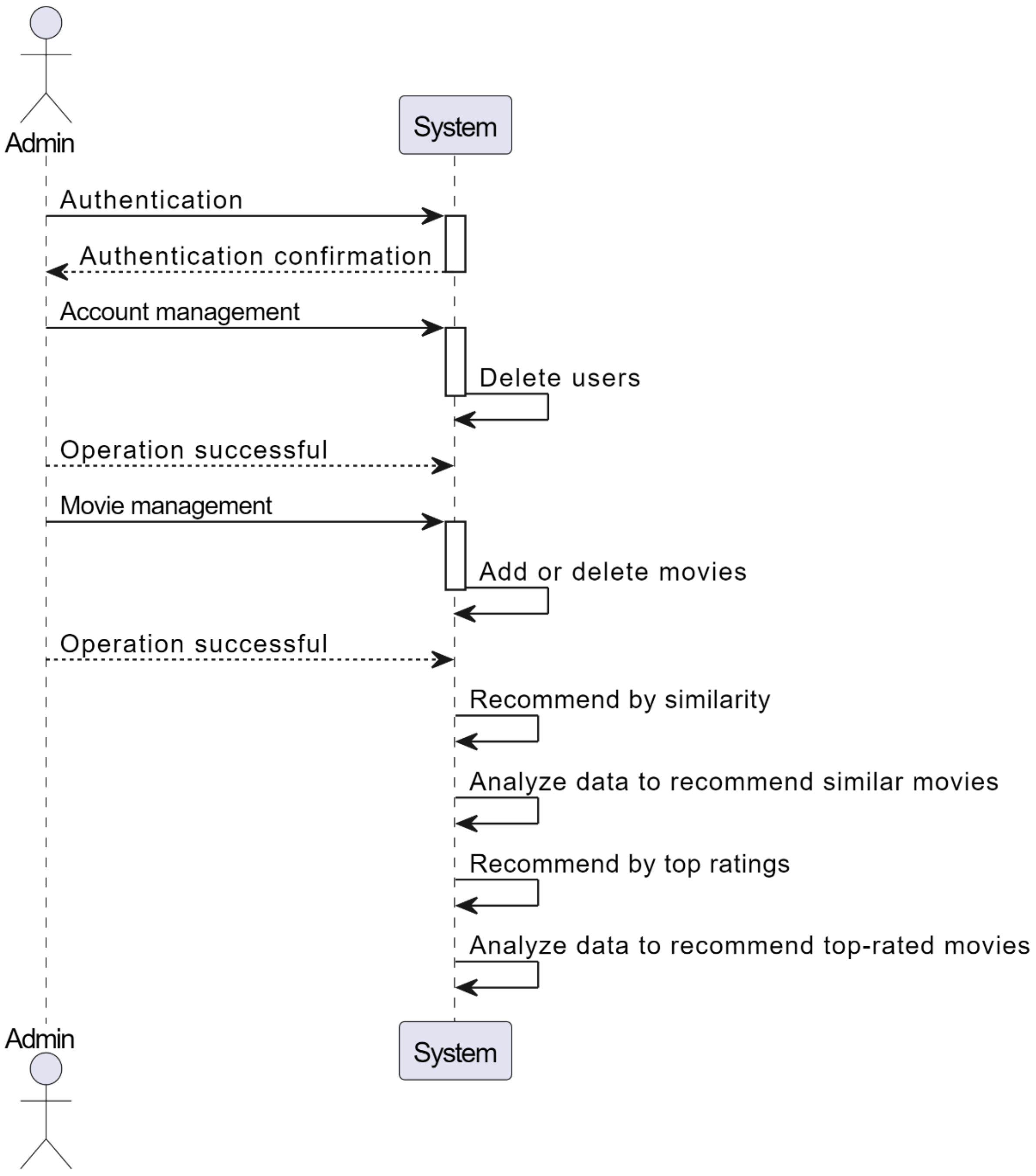


Figure 4- 5: admin interactions

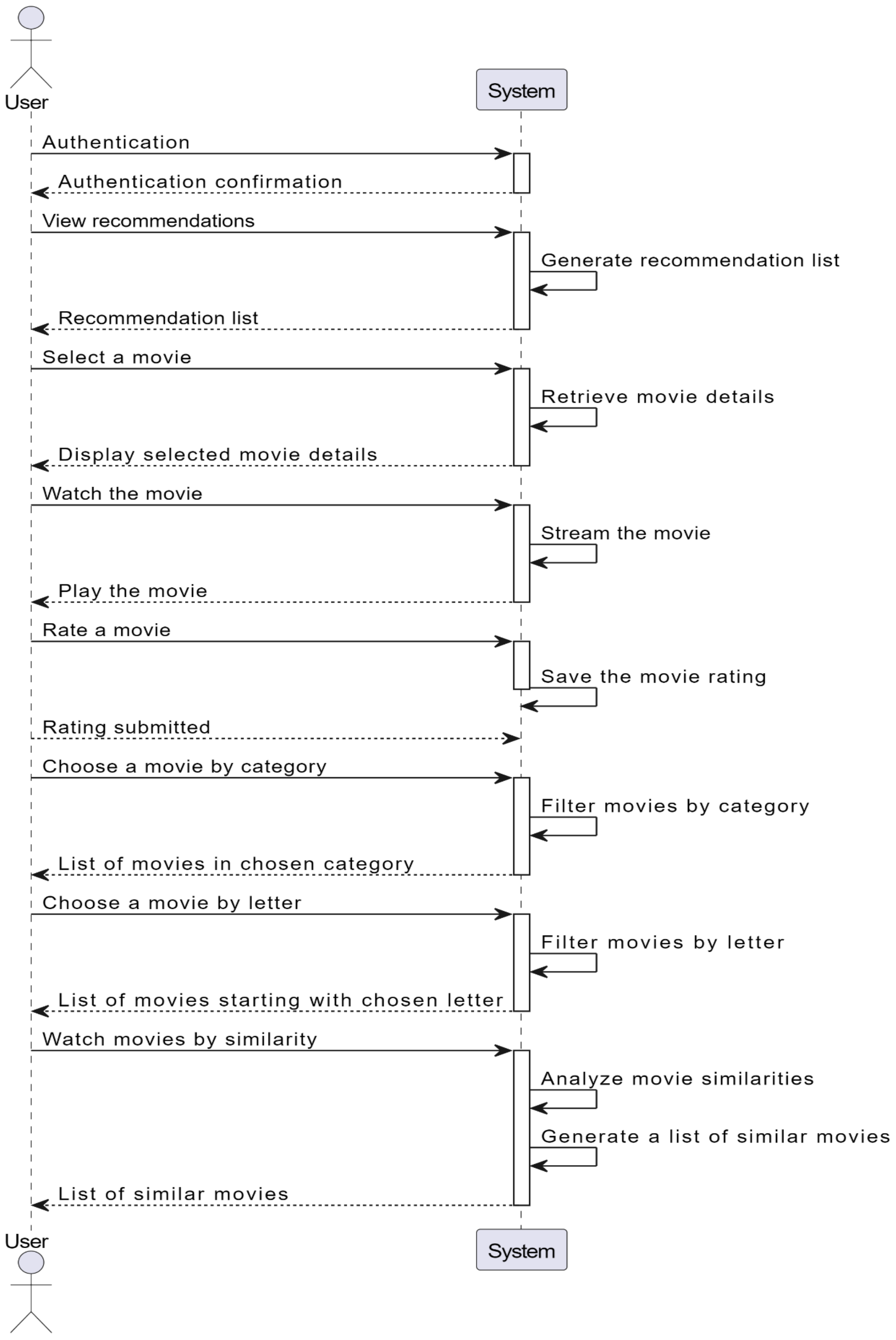

User
System
Authentication
Authentication confirmation
View recommendations
Generate recommendation list
Recommendation list
Select a movie
Retrieve movie details
Display selected movie details
Watch the movie
Stream the movie
Play the movie
Rate a movie
Save the movie rating
Rating submitted
Choose a movie by category
Filter movies by category
List of movies in chosen category
Choose a movie by letter
Filter movies by letter
List of movies starting with chosen letter
Watch movies by similarity
Analyze movie similarities
Generate a list of similar movies
List of similar movies
User
System

Figure 4- 6: user interactions

Figure 4- 5: Sequence diagram

2.3 Development environment

2.3.1 Development tools

Vs Code: Visual Studio is a suite of powerful development tools that provides a complete environment for the entire development cycle. This integrated development environment (IDE) allows developers to write, modify, debug, and generate code, as well as deploy their applications. In addition to code editing and debugging features, Visual Studio offers compilers, code completion tools, source control, extensions, and many other features to improve every step of the software development process.

Jupyter Notebook: Represents the next generation of web-based interactive development environments designed specifically for notebooks, code, and data. With a flexible interface, it allows users to configure and organize their workflows in various fields such as data science, scientific computing, computational journalism, and machine learning. Its modular architecture encourages extensions to enrich and extend its functionality, providing a flexible and scalable platform for developers and researchers.

Zotero : Zotero is a free, open source and cross-platform bibliographic reference management software that is part of the Web 2.0 philosophy. It allows you to manage bibliographic data and research documents.

2.2.2 Programming languages

HTML or Hypertext Markup Language: It is a markup language used to create and structure web pages. Its main role is to describe the content and structure of hypertext documents intended to be displayed on web browsers. As a client-side language, it works to provide a complete user experience. HTML is standardized and developed by the World Wide Web Consortium (W3C). Its origin dates back to the early days of the web in the 1989s, where it was designed to present documents online and create hyperlinks between them.

Python: Open-source language most used by computer scientists. This language has propelled itself to the forefront in infrastructure management, data analysis or in the field of software development offering a variety of libraries. Indeed, among its qualities, Python allows developers to focus on what they do rather than how they do it. It freed developers from the form constraints that occupied their time with older languages.

CSS : Cascading Style Sheets. It is a style language whose syntax is extremely simple but its performance is remarkable. Indeed, CSS is concerned with the formatting of content integrated with HTML.

PHP: Known as a recursive acronym for "Hypertext Preprocessor" but originally called " Personal Home Page", is a general-purpose, open-source scripting language. It is mainly used for web application development and can be easily integrated with HTML. PHP allows you to create dynamic web pages by generating HTML content from scripts executed on the server side. With its simple syntax and great flexibility, PHP is widely adopted in the world of web development for creating interactive sites and content management systems (CMS) such as WordPress, Joomla and Drupal.

JavaScript: Often abbreviated to "JS", is a scripting programming language used primarily for client-side web development. It allows you to add interactive features to web pages, such as animations and real-time updates. JavaScript is interpreted by web browsers and has a flexible and expressive syntax. It supports multiple programming paradigms and is also used for server-side development, mobile applications, and games. In summary, JavaScript is a general-purpose language widely used for creating dynamic and interactive web applications, as well as other types of software on various real-time content platforms.

Miro: Miro, formerly known as RealtimeBoard, is an online collaborative whiteboard solution designed to facilitate remote and distributed team communication and project management. As an online workspace for innovation, it allows employees to share ideas and meets various needs: meetings, workshops, brainstorming, agile workflows, UX design, mind mapping, and much more.

Draw.io : Proficiency in diagramming tools such as diagrams.net (formerly draw.io) is part of the experience creating flowcharts, wireframes, UML diagrams, flowcharts and network diagrams.

NB: Some python libraries are also used

2.2.3 User interfaces

Home page:

In this part we can see the main page for all users, user can logout, do some research and found his recommend panel

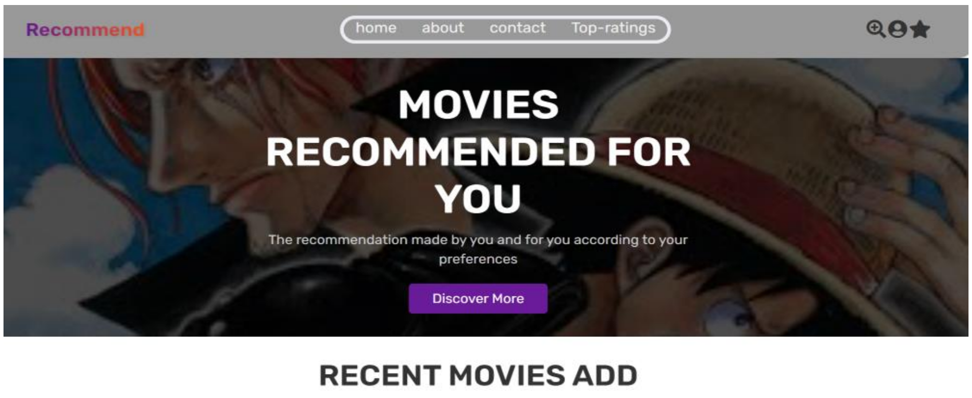

Recommend
home
about
contact
Top-ratings
MOVIES RECOMMENDED FOR YOU
The recommendation made by you and for you according to your preferences
Discover More
RECENT MOVIES ADD

Figure 4- 7: Home page

New movies add:

In this part admin can publish movie with name and category's information in the main page where users can do some interactions with them, can filter by letter .

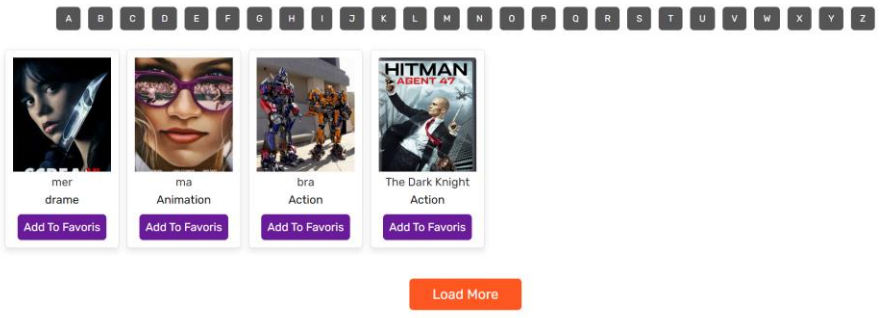


Figure 4- 8: New movies add

Movies from a dataset with the image API

In this part, a dataset with movies information like release date, category, rating is using and the images is coming from the movie database API

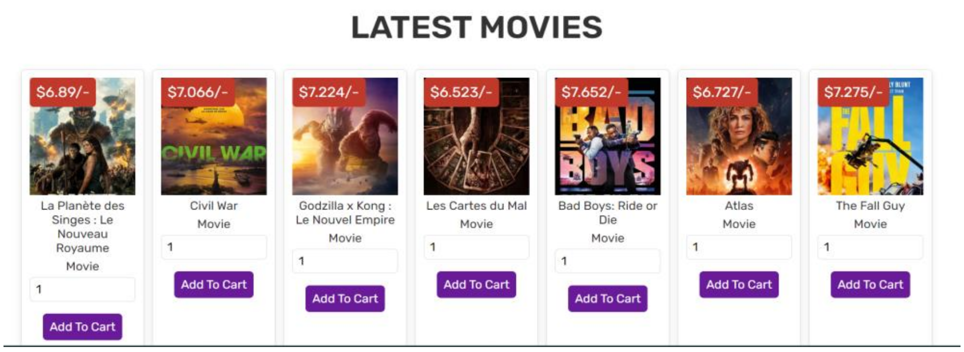


Figure 4-9: Movies from a dataset with the image API

Messaging system

User can send messages to admin if something wrong with the system, post some suggestions in admin panel for consultation, help to make a good system.

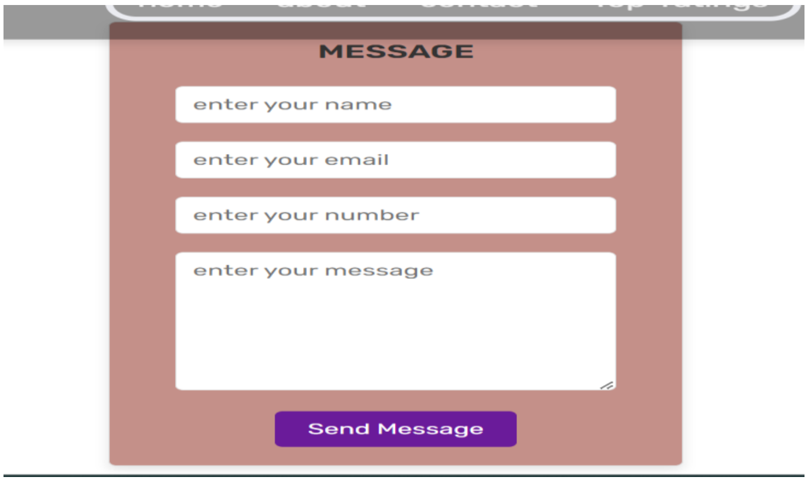


Figure 4- 10: Messaging system

Authentication page

This page is for authentication of admin/user, they will put mail address and password and the difference between them if it is admin or user and the system take the correspondent panel’s main page.

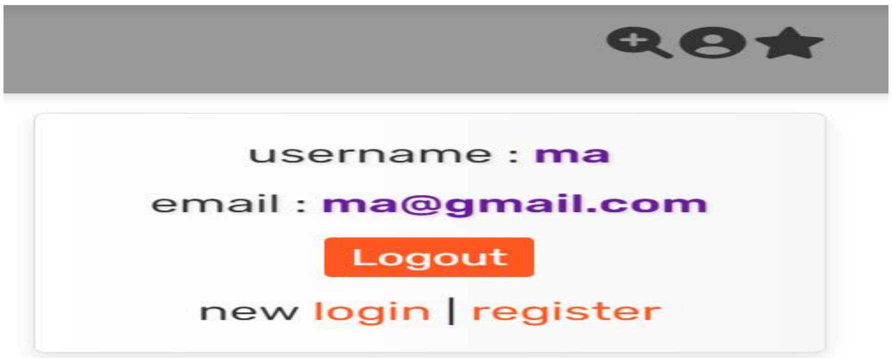


Figure 4- 11: Authentication admin

List of best notes

All users movie's notes are in this part, if user rate some movies and the system will do the selection of the most graduate element

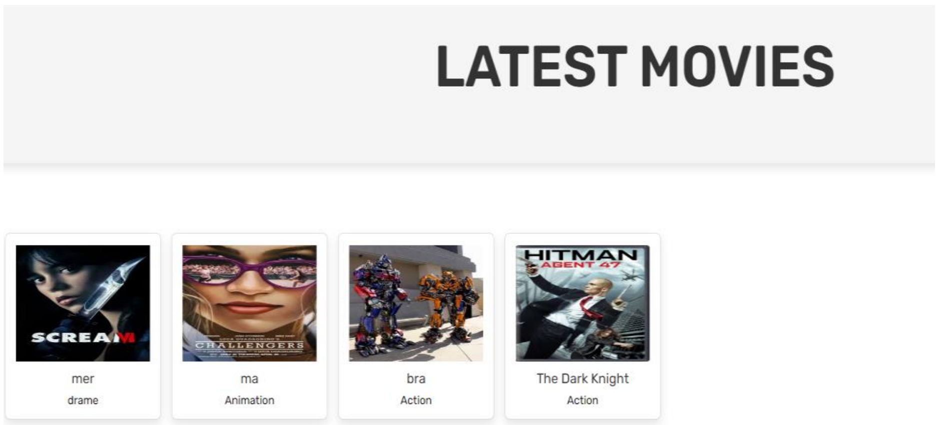


Figure 4- 12: list of best rating

Search bar

The search bar allows users to find movies by title, genre, or actors. Advanced algorithms deliver relevant results quickly, with filters for refining searches based on user preferences.

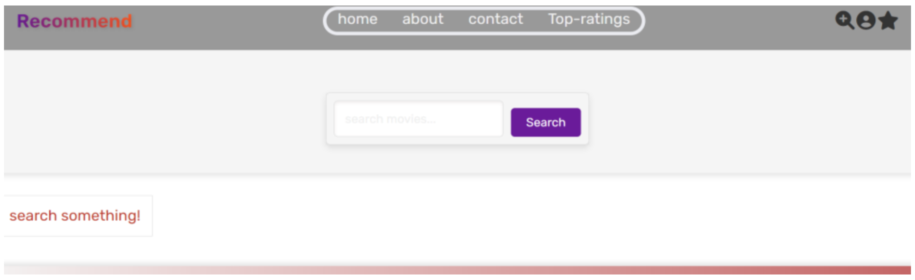


Figure 4- 13: Search bar

Movies with trailer and rating system

Each movie page includes trailers for preview and a rating system for user feedback. Ratings influence movie visibility and user decisions, enhancing interaction and informed viewing choices.

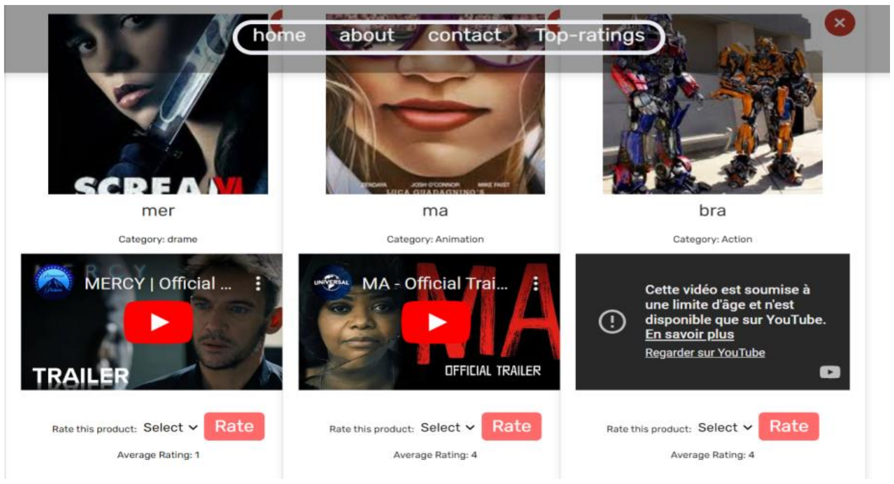


Figure 4- 14: Movies with trailer and rating

Similar movies

Using algorithms, the platform suggests similar movies based on genre, actors, and user preferences. These recommendations enhance user engagement and facilitate content discovery.

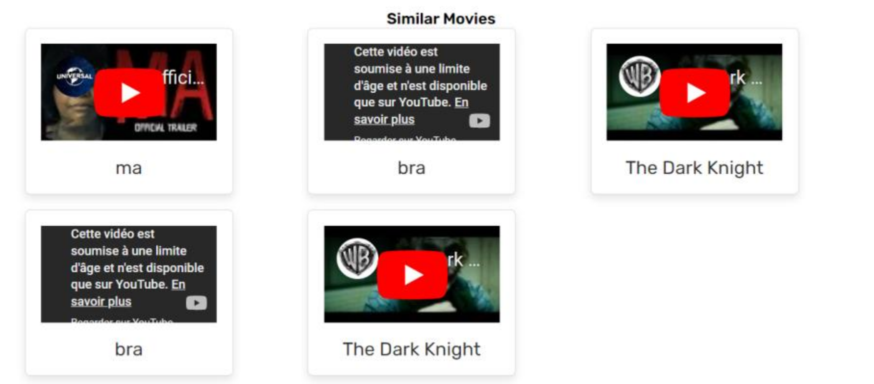

Similar Movies
ma
bra
The Dark Knight
bra
The Dark Knight
Cette vidéo est soumise à une limite d'âge et n'est disponible que sur YouTube. En savoir plus

Figure 4- 15: Similar movies

Admin panel for control

The admin panel centralizes user management, content moderation, system configuration, and analytics. It empowers administrators with tools for overseeing platform operations and optimizing user experience.

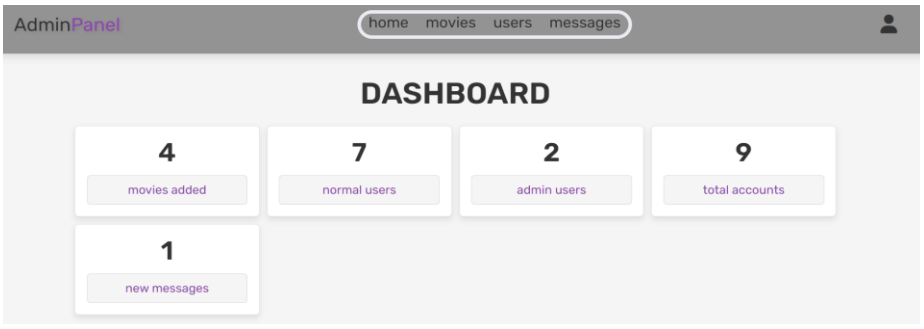


Figure 4- 16: Admin panel for control

Add movies

Users input movie details (title, genre, etc.) and upload media (poster, trailer). Submissions undergo validation and approval by administrators, ensuring content quality and alignment with platform standards.

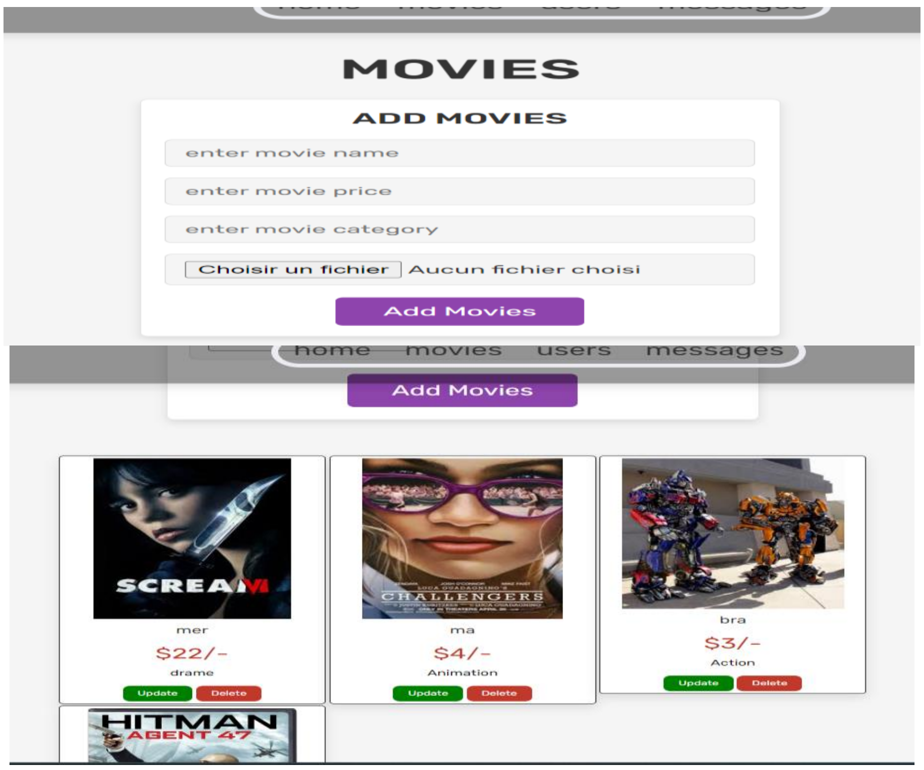
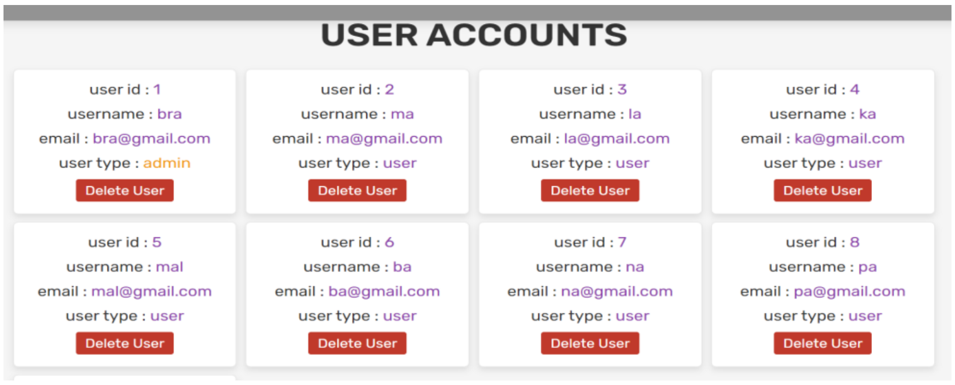


Figure 4- 17: Add movies

User's accounts

User accounts store essential details like usernames, hashed passwords for security, emails, and optional profile information. These accounts enable personalized experiences, secure authentication, and controlled access to platform features.



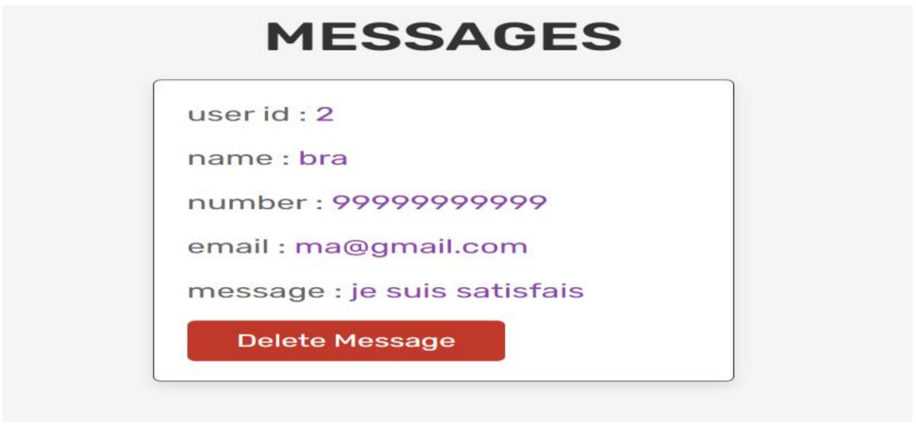
USER ACCOUNTS

user id : 1 username : bra email : bra@gmail.com user type : admin Delete User	user id : 2 username : ma email : ma@gmail.com user type : user Delete User	user id : 3 username : la email : la@gmail.com user type : user Delete User	user id : 4 username : ka email : ka@gmail.com user type : user Delete User
user id : 5 username : mal email : mal@gmail.com user type : user Delete User	user id : 6 username : ba email : ba@gmail.com user type : user Delete User	user id : 7 username : na email : na@gmail.com user type : user Delete User	user id : 8 username : pa email : pa@gmail.com user type : user Delete User

Figure 4- 18: All users

User's message

Facilitating user-admin and user-user communication, the messaging system supports inquiries and feedback. It integrates with user accounts and the admin panel for efficient issue resolution and community engagement.



MESSAGES

user id : 2

name : bra

number : 99999999999

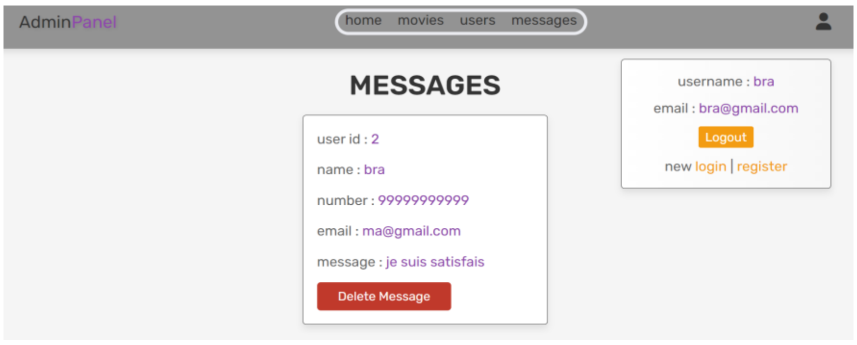
email : ma@gmail.com

message : je suis satisfais

Figure 4- 19: All users messages

Admin authentication

Admin authentication" typically refers to the process of verifying the identity of administrators who are accessing or managing a system, network, or application. It is a critical security measure implemented to ensure that only authorized personnel can perform administrative tasks and access sensitive information



MESSAGES

user id : 2

name : bra

number : 99999999999

email : ma@gmail.com

message : je suis satisfais

username : bra

email : bra@gmail.com

Figure 4-20: Admin authentication

3. Conclusion

Experimentation and implementation constituted an essential step in the validation of the proposed methods for recommendation systems. Through analysis and design, the system requirements were specified in detail, enabling efficient implementation. UML modeling provided a structured framework for system design, while the development environment provided the necessary tools and languages for creating functional prototypes.

GENERAL CONCLUSION

General conclusion

This dissertation sets out to provide an in-depth analysis and practical implementation of movie recommendation systems using bipartite graphs. We started with a detailed state of the art on recommendation systems, highlighting the different techniques used and the associated challenges. Next, we explored modeling using bipartite graphs in movie recommendation systems providing a robust and efficient approach to improve the quality of recommendations.

This method not only helps manage new users and new movies efficiently, but also offers more relevant and personalized suggestions. The conceptual study, followed by the practical implementation of our system, illustrates the tangible advantages of this approach. By integrating various contextual data sources, we were able to overcome the limitations of traditional methods and provide a robust and efficient system.

The results obtained show that the use of bipartite graphs makes it possible to improve the precision and relevance of the recommendations. However, our work also opens new questions and possibilities for future research, such as integrating additional contextual data and optimizing recommendation algorithms for increased performance while providing promising avenues for further research.

Our proposal is a concrete solution in the form of a movie recommendation system, supported by a rigorous conceptual study. The implementation of this solution was carried out using appropriate development tools and programming languages, and we modeled our system using UML diagrams for better understanding and communication of concepts.

1. Idrissi, N. & Ahmed, Z. A systematic literature review of sparsity issues in recommender systems . Soc. Netw . Anal. Min. 10 , (2020).

2. Dong, Z., Wang, Z., Xu, J., Tang, R. & Wen, J. A Brief History of Recommender Systems. Preprint at http://arxiv.org/abs/2209.01860 (2022).

3. Crété-Roffet , F. Estimating, measuring and correcting compression artifacts for digital television. (Joseph-Fourier University - Grenoble I, 2007).

4. Gasmi , W. Content-based filtering for course recommendation (FCRC).

5. Wang, S. et al. A Survey on Session-based Recommender Systems. Preprint at https://doi.org/10.48550/arXiv.1902.04864 (2021).

6. Jena, KK et al. E-Learning Course Recommender System Using Collaborative Filtering Models. Electronics 12 , 157 (2023).

7. Sahoo, AK, Pradhan, C., Barik , RK & Dubey, H. DeepReco : Deep Learning Based Health Recommender System Using Collaborative Filtering. Computation 7 , 25 (2019).

8. Naak , A. Papyres : a research article management and recommendation system. (2009).

9. Zhang, S., Yao, L., Sun, A. & Tay, Y. Deep Learning based Recommender System: A Survey and New Perspectives. ACM Comput. Surv . 52 , 1–38 (2020).

10. Esslimani , I. Towards a behavioral recommendation approach: contribution of usage analysis to a personalization process. (Nancy II University, 2010).

11. Hammoudi, M. & Kechra , RA Automatic Vehicle Identification Systems. (Ibn Khaldoun University -Tiaret-, 2022).

12. Souilah, S. New Cold Start Method for Recommender Systems. http://dspace.univ-guelma.dz/jspui/handle/123456789/4346 (2019).

13. Gauthier, L.-A. Inference of signed links in social networks, by learning from user interactions. (Pierre and Marie Curie University - Paris VI, 2015).

14. Du, Y. From data to knowledge: towards more relevant, diversified and transparent recommendations. (IMT - MINES ALES - IMT - Mines Alès Ecole Mines - Télécom, 2021).

15. Category: Doctor from Joseph-Fourier University - Grenoble 1. Wikipedia (2023).

16. Towards Distributed Collaborative Filtering: the RSB model - Inria - National Institute for Research in Digital Sciences and Technologies. https://inria.hal.science/inria-00000509.

17. Batmaz , Z., Yurekli , A., Bilge , A. & Kaleli , C. A review on depth learning for recommender systems : challenges and remedies . Artif . Intellect . Rev. 52, 1–37 (2019).

18. Bouteldja , A. & Moussaoui, H. Factorization techniques in an SFC system. (Ibn Khaldoun University, 2023).

19. Recommend Systems Handbook SpringerLink . https://link.springer.com/book/10.1007/978-0-387-85820-3.

20. Donald, T. Design and implementation of an ecommerce application with machine learning based recommendation system using machine learning techniques. Tedom Noutchogouin Donald (2023).

21. Movie Recommendation System using Machine Learning - Shiksha Online. https://www.shiksha.com/online-courses/articles/movie-recommendation-system-using-machine-learning/.

22. B. M., Konstan, J. A., & Riedl, J. Sarwar, "Distributed Recommender Systems for Internet Commerce," 2005.

23. Lherisson , P.-R. Fair recommendation system for digital works . In search of diversity.

24. Dyhia , D. & Celia, M. Definition of a user profile for a recommendation system in information retrieval. (Moloud Mammeri University, 2019).

25. G. Pavlopoulos, P. Kontou, A. Pavlopoulou, C. Bouyioukos, E. Markou, and P. Bagos, “Bipartite graphs in systems biology and medicine: A survey of methods and applications,” GigaScience, vol. 7, pp. 1-31, 2018.

26. M. Balabanović and Y. Shoham, “Fab: Content-Based, collaborative recommendation,” Communications of the ACM, vol. 40, no. 3, pp. 66-72, 1997.

27. Biswas, R. et al. Knowledge Graph Embeddings: Open Challenges and Opportunities. Trans. Graph Data Knowl. TGDK 1, 4:1-4:32 (2023).

28. Fournier-S’niehotta, R., Mouza, C. D. & Viard, T. M´etriques pour l’analyse de grands graphes bipartis.

29. Lin, Y., Liu, Z., Sun, M., Liu, Y. & Zhu, X. Learning Entity and Relation Embeddings for Knowledge Graph Completion. Proc. AAAI Conf. Artif. Intell. 29, (2015).

30. Crampes, M. & Plantié, M. Détection de communautés dans les graphes bipartis. in IC 2012 125 (PARIS, France, 2012).

31. Beauguitte, L. L’analyse des graphes bipartis.
32. Sawant, S. Collaborative Filtering using Weighted BiPartite Graph Projection A Recommendation System for Yelp.

33. Dahimene, M. R. Filtrage et Recommandation sur les Réseaux Sociaux. (Conservatoire national des arts et metiers - CNAM, 2014).
34. Benouaret, I. Un système de recommandation contextuel et composite pour la visite personnalisée de sites culturels.
35. Attal, J.-P. Nouveaux algorithmes pour la détection de communautés disjointes et chevauchantes basés sur la propagation de labels et adaptés aux grands graphes.
36. Creusefond, J. Caractériser et détecter les communautés dans les réseaux sociaux.
37. Yao, L. Study On Bipartite Network In Collaborative Filtering Recommender System. (Pennsylvania State University, 2015).
38. Garrouch, K. Modèles de Recherche d'information basés sur les Réseaux Bayésiens et les Réseaux Possibilistes. (2017).
39. Minel, J.-L. Filtrage semantique de textesproblemes, conception et realisation d'une plate-forme informatique.
40. Laitos - Comparing ranking-based collaborative filtering al.pdf.
41. Zaïer, Z. Modèle multi-agents pour le filtrage collaboratif de l'information.