MOVIE RECOMMENDATION SYSTEM

BY

EWUOLA ANDREW OLUWAFEMI
PSC1908858

DEPARTMENT OF COMPUTER SCIENCE,
FACULTY OF PHYSICAL SCIENCES,
UNIVERSITY OF BENIN,
BENIN CITY,
EDO STATE, NIGERIA.

MAY 2024

MOVIE RECOMMENDATION SYSTEM

BY

EWUOLA ANDREW OLUWAFEMI
PSC1908858

A PROJECT REPORT SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE, FACULTY OF PHYSICAL SCIENCES, UNIVERSITY OF BENIN, BENIN CITY, IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE AWARD OF A BACHELOR OF SCIENCE (B.Sc.) DEGREE IN COMPUTER SCIENCE

MAY 2024

CERTIFICATION

This is to certify that this project work was carried out by EWUOLA ANDREW with Matriculation Number PSC1908858 under my supervision. It is adequate and satisfactory, both in scope and content, for the award of Bachelor of Science (B.sc) Degree in Computer Science of the University of Benin

_____ _____

DR. (MRS) G. AZIKEN DATE

Project Supervisor

APPROVAL

This project work is hereby approved in partial fulfilment of the requirements for the award of Bachelor of Science (B.Sc.) Degree in Computer Science from the University of Benin.

_____ _____

Prof. Godspower O. Ekuobase DATE

Project Supervisor

DEDICATION

This project is dedicated to God Almighty for giving me the strength and wisdom to see it through to completion, and even throughout my stay in the University of Benin (UNIBEN). It is also dedicated to my parents; Mr and Mrs Ewuola for their love, support and guidance throughout my academic journey.

ACKNOWLEGEDMENT

My utmost acknowledgement goes to God Almighty for giving me the strength, wisdom and direction throughout my academic journey. I would like to express my gratitude to my project supervisor, Dr. (Mrs.) Aziken, and the Head of the Department Of Computer Science, Prof. Godspower O. Ekuobase for their consistent guidance towards ensuring the successful completion of this project.

I would also like to specially thank other lecturers in the Department of Computer Science who I have been opportune to cross paths with, and have impacted me immensely these past few years: Prof. G.O. Ekuobase, Dr. F.O. Oliha, Prof. K.C. Ukaoha, Prof. A.A. Imiavan, Prof. (Mrs.) F. Egbokhare, Prof. (Mrs.) V.V.N. Akwukwuma, Prof. F.I. Amadin, Prof. (Mrs.) S. Konyeha, Prof. (Mrs.) V.I. Osubor, Dr. F.O. Chete, Dr. (Mrs) R.O. Osaseri, Dr. J.C. Obi, Mr. P. E.B. Imiefoh, Mr. I.E. Obasohan, Mr. S.O.P. Oliomogbe, Mr. K.O. Otokiti, Mr. I.E. obayagbonna, Mrs. R.I. Izevbizua, Mr. E.C. Igodan, Miss L.O.Usiosefe, Mr J. Okhuoya, Prof. F.A.U. Imouokhome, Mrs. J.I. Adun, Dr. E. Nweli and Mr. D.N. Idehen.

Finally, I also want to appreciate those who contributed to the success of this project: Grant O. Juliana, Igonor Nathaniel, Igonor Mathais, I would also like to thank my family and friends for their support, words of encouragement, and consistent guidance throughout this project.

TABLE OF CONTENTS

COVER PAGE i
TITLE PAGE ii
CERTIFICATION iii
APPROVAL iv
DEDICATION v
ACKNOWLEGEDMENT vi
ABSTRACT x
CHAPTER ONE 1
INTRODUCTION 1
1.0 BACKGROUND TO THE STUDY 1
1.2 STATEMENT OF PROBLEM 3
1.3 AIM AND OBJECTIVES 3
1.4 SIGNIFICANCE OF THIS STUDY 4
1.5 SCOPE OF THE STUDY 5
CHAPTER TWO 7
LITERATURE REVIEW 7
2.1 INTRODUCTION 7
1.1 COLLABORATIVE FILTERING 8
2.2 Collaborative Filtering and Sparsity: 11
2.3 Cold Start Problems: 12
2.4 Hybrid Recommendation Systems: 12
2.5 Evaluation of Recommender Systems: 12
2.6 Empirical Review Of Existing Articles 13
Limitations 15
Limitations 23
CHAPTER THREE 26
METHODOLOGY 26
3.1 Introduction 26
3.1.1 Data Collection and Preprocessing 26
3.1.2. Recommendation Algorithm Development 27
3.1.3 Sparsity Reduction Techniques: 27
3.1.4 Cold Start Mitigation Strategies: 27
CHAPTER FOUR 30

4.1 Introduction 30
4.2 System Architecture 30
4.2.1 Data Collection and Preprocessing 30
4.2.2 Feature Extraction 31
4.2.3 Recommendation Engine 32
4.2.4 User Interface 33
4.2.5 Evaluation Module 33
4.3.1 Software Requirement 33
4.3 System Interface 34
4.3.1 Login Interface 34
4.3.2 Dashboard 35
4.5 Conclusion 37
CHAPTER 5 39
Recommendation and Conclusion 39
Summary Of Findings 39
Conclusion 40
Recommendations For Improvement 41
REFERENCES 43
APPENDIX 46

LIST OF FIGURES

Figure 4.1 Login Interface 35

Figure 4.2 Dashboard 36

ABSTRACT

The objective of this research is to create a web-based movie recommender system that effectively tackles the challenges of data sparsity and cold start. The system effectively combines explicit data, such as movie ratings and implicit data, including watch history, search queries, and interactions, in order to gain a comprehensive understanding of user preferences. Data cleaning and normalisation are essential preprocessing steps. Sophisticated recommendation algorithms are utilised to improve recommendations, incorporating collaborative filtering, sparsity reduction, and cold start mitigation techniques. The evaluation findings indicate better recommendation, underscoring the system's efficacy in addressing sparsity and cold start obstacles.

CHAPTER ONE

INTRODUCTION

1.0 BACKGROUND TO THE STUDY

Movie recommender systems are intelligent algorithms that suggest movies for users to watch based on their previous viewing behavior & preferences. These systems analyze data such as users' ratings, reviews, & viewing histories to generate personalized recommendations. Movie recommender system has revolutionized the way people discover & consume movies, enabling users to navigate through vast catalogs of films more efficiently. Recommender systems have two main categories: content-based & collaborative filtering. Content-based movie recommendation system algorithms use the similarities between movies to recommend new movies to users, while collaborative filtering utilizes other users' overlapping movie ratings to generate recommendations. Overall, the movie recommender system has become an essential tool for movie enthusiasts seeking to discover new films.

In the field of movie recommendation systems, two major issues frequently arise: sparsity and the cold start problem. Both of these concerns can reduce the recommendation system's effectiveness and user happiness.

Sparsity stems from the nature of user interactions with films. In a typical scenario, a streaming platform may provide tens of thousands of films, with the average user rating or watching only a small fraction of them. This yields a highly sparse user-item matrix with most entries empty. The sparsity problem makes it difficult for the recommendation algorithms to identify meaningful patterns and similarities between users and items

because there is insufficient data. For instance, if a user has only rated a handful of movies, the system lacks enough information to make accurate predictions about their preferences for other movies.

To combat sparsity, recommendation systems often rely on various techniques. Collaborative filtering, especially matrix factorization methods like Singular Value Decomposition (SVD), can help by identifying latent factors that explain the observed ratings. These factors can capture the underlying structure of the data, enabling the system to make better predictions even with sparse matrices. Additionally, incorporating content-based methods, where recommendations are based on the attributes of the items (such as genre, director, actors), can also alleviate some of the issues caused by sparsity. Hybrid systems that combine collaborative filtering and content-based approaches often provide more robust recommendations.

The cold start problem refers to the difficulty in recommending items to users who have just joined the platform and thus have no interaction history, or recommending new items that have not been rated by many users. For new users, the system has little to no data to base its recommendations on, which can lead to irrelevant suggestions and a poor initial user experience. For new items, the system struggles to recommend them because it lacks sufficient user feedback to understand who might like them.

Several strategies are employed to address the cold start problem. For new users, recommendation systems can use demographic information (age, gender, location) to provide initial recommendations based on the preferences of similar users. Another approach is to implement a brief onboarding survey where new users rate a small set of

popular movies, giving the system an immediate, though limited, sense of their tastes. For new items, leveraging content-based methods can be particularly effective. By analyzing the attributes of new movies, the system can recommend them to users whose profiles match the characteristics of these movies. Additionally, promotions or highlighting new releases can help gather the initial user feedback needed to improve recommendations over time.

In practice, the most successful movie recommendation systems deploy a combination of these techniques. They use advanced algorithms to navigate the sparsity of the data, leveraging both collaborative filtering and content-based approaches to enhance accuracy. At the same time, they employ thoughtful strategies to mitigate the cold start problem, ensuring that new users and new items receive relevant and engaging recommendations from the outset. Through continuous iteration and the incorporation of user feedback, these systems strive to provide a personalized and satisfying viewing experience for all users.

1.2 STATEMENT OF PROBLEM

Existing movie recommender systems, particularly those utilizing collaborative filtering, struggle with sparsity and cold start problems when implemented in a web-based system.

1.3 AIM AND OBJECTIVES

Aim: This research aims to develop a web-based movie recommender system that addresses sparsity and cold start issues.

OBJECTIVES:

Reduce the Impact of Sparsity: Develop techniques to leverage the limited user rating data effectively. This could involve incorporating additional user information beyond ratings (e.g., demographics, genres watched) or exploring alternative similarity measures to identify relevant neighbors for recommendation even with sparse data.

Cold Start Recommendations for New Users: Design strategies to generate personalized recommendations for new users with minimal initial information. This might involve implementing content-based filtering alongside collaborative filtering, utilizing implicit data sources like watch history to learn user preferences from limited interaction data.

Overall Recommendation Accuracy: Develop methods to for the overall quality and accuracy of recommendations for all users.

1.4 SIGNIFICANCE OF THIS STUDY

Developing a web-based movie recommender system that addresses sparsity and cold start problems holds significant value for both users and recommender system platforms. Here's how:

Enhanced User Experience: More accurate and personalized recommendations will lead to a more satisfying user experience. Users will discover movies they're genuinely interested in, increasing engagement and satisfaction with the platform.

Improved Content Discovery: Overcoming sparsity allows the system to recommend a wider range of movies, even those with limited ratings. This helps users discover hidden gems and explore diverse genres they might not have encountered otherwise.

Increased User Retention: Accurate recommendations can keep users engaged for longer periods, encouraging them to explore the platform's movie library and potentially subscribe to premium services.

Boosted Platform Performance: A more effective recommendation system can lead to increased user satisfaction and potentially translate to higher conversion rates (e.g., purchases, subscriptions) for the platform.

1.5 SCOPE OF THE STUDY

This research will focus on developing and evaluating a web-based movie recommender system that addresses sparsity and cold start problems. The scope will encompass the following:

Data Collection and Preprocessing: The study will define the types of user data to be collected (ratings, demographics, watch history) and develop methods for cleaning and preparing the data for use in the recommendation algorithms.

Algorithm Development and Implementation: The focus will be on designing and implementing techniques to mitigate sparsity and cold start issues. This might involve incorporating additional user information, exploring alternative similarity measures, or utilizing hybrid approaches that combine collaborative and content-based filtering.

Evaluation Methodology: The study will establish methods to evaluate the effectiveness of the proposed system. This could involve metrics like recommendation accuracy, user satisfaction surveys, and A/B testing different recommendation strategies.

Web-based System Development: The research will involve building a functional web-based prototype of the movie recommender system. The prototype will be designed to be

user-friendly and integrate seamlessly with the chosen data collection and processing methods.

The study will not delve into the development of entirely new recommendation algorithms but will focus on adapting and optimizing existing techniques to address the specific challenges of sparsity and cold start in a web-based environment.

CHAPTER TWO

LITERATURE REVIEW

2.1 INTRODUCTION

Building upon the identified problems, research aim, objectives, significance, and scope, this literature review will delve into existing research on movie recommender systems, focusing on approaches to address sparsity and cold start issues.

HISTORY

Recommender systems were introduced in the mid-1990s to help people select the most suitable product for them from the plethora of options available with them. The idea that led to their development was that we people often rely on the opinions of our peers before trying something new, say it be before buying a smart phone, a laptop, before going for a movie, before going to a new restaurant and even before visiting a doctor. Till date, we have numerous recommender systems developed for various areas, using different recommendation approaches. Yet, there are still a few limitations of recommender systems that need to be worked on. In this paper, we present an overview of recommender systems, the various approaches of recommender systems, the application areas for which various recommender systems have been developed and we also present the limitations of recommender systems.

Elaine Rich created the first recommender system in 1979, called Grundy. She looked for a way to recommend users books they might like. Her idea was to create a system that asks users specific questions and classifies them into classes of preferences, or

"stereotypes", depending on their answers. Depending on users' stereotype membership, they would then get recommendations for books they might like.

Another early recommender system, called a "digital bookshelf", was described in a 1990 technical report by Jussi Karlgren at Columbia University, and implemented at scale and worked through in technical reports and publications from 1994 onwards by Jussi Karlgren, then at SICS,and research groups led by Pattie Maes at MIT, Will Hill at Bellcore, and Paul Resnick, also at MIT whose work with GroupLens was awarded the 2010 ACM Software Systems Award.

Montaner provided the first overview of recommender systems from an intelligent agent perspective. Adomavicius provided a new, alternate overview of recommender systems. Herlocker provides an additional overview of evaluation techniques for recommender systems, and Beel et al. discussed the problems of offline evaluations. Beel et al. have also provided literature surveys on available research paper recommender systems and existing challenges.

APPROACH

1.1 COLLABORATIVE FILTERING

One approach to the design of recommender systems that has wide use is collaborative filtering. Collaborative filtering is based on the assumption that people who agreed in the past will agree in the future, and that they will like similar kinds of items as they liked in the past. The system generates recommendations using only information about rating

profiles for different users or items. By locating peer users/items with a rating history similar to the current user or item, they generate recommendations using this neighborhood. Collaborative filtering methods are classified as memory-based and model-based. A well-known example of memory-based approaches is the user-based algorithm, while that of model-based approaches is matrix factorization (recommender systems).

A key advantage of the collaborative filtering approach is that it does not rely on machine analyzable content and therefore it is capable of accurately recommending complex items such as movies without requiring an "understanding" of the item itself. Many algorithms have been used in measuring user similarity or item similarity in recommender systems. For example, the k-nearest neighbor (k-NN) approach and the Pearson Correlation as first implemented by Allen.

When building a model from a user's behavior, a distinction is often made between explicit and implicit forms of data collection.

Examples of explicit data collection include the following:

Asking a user to rate an item on a sliding scale.

Asking a user to search.

Asking a user to rank a collection of items from favorite to least favorite.

Presenting two items to a user and asking him/her to choose the better one of them.

Asking a user to create a list of items that he/she likes (see Rocchio classification or other similar techniques).

Examples of implicit data collection include the following:

Observing the items that a user views in an online store.

Analyzing item/user viewing times.

Keeping a record of the items that a user purchases online.

Obtaining a list of items that a user has listened to or watched on his/her computer.

Analyzing the user's social network and discovering similar likes and dislikes.

Collaborative filtering approaches often suffer from three problems: cold start, scalability, and sparsity.

Cold start: For a new user or item, there is not enough data to make accurate recommendations. Note: one commonly implemented solution to this problem is the multi-armed bandit algorithm.

Scalability: There are millions of users and products in many of the environments in which these systems make recommendations. Thus, a large amount of computation power is often necessary to calculate recommendations.

Sparsity: The number of items sold on major e-commerce sites is extremely large. The most active users will only have rated a small subset of the overall database. Thus, even the most popular items have very few ratings.

One of the most famous examples of collaborative filtering is item-to-item collaborative filtering (people who buy x also buy y), an algorithm popularized by Amazon.com's recommender system.

Many social networks originally used collaborative filtering to recommend new friends, groups, and other social connections by examining the network of connections between a user and their friends. Collaborative filtering is still used as part of hybrid systems.

Movie Recommendation System Architecture

The movie recommendation system architecture is a complex process that utilizes various algorithms to suggest movies to users based on their preferences. The architecture involves collecting data on user behavior, such as previous movie selections & ratings, & using that data to create a personalized list of suggestions. The heart of this system lies in the algorithm used in movie recommendation system.

Typically, recommendation systems use collaborative filtering algorithms, which analyze a user's profile & behavior, along with those of other users, to make suggestions. In general, the architecture of a movie recommender system process is intricately designed to provide a seamless, enjoyable movie experience for users.

2.2 Collaborative Filtering and Sparsity:

Traditional collaborative filtering techniques like user-based and item-based approaches suffer from sparsity when user rating data is limited.

Studies by [Sarwar et al., 2001] and [Schein et al., 2002] explore techniques like dimensionality reduction and matrix factorization to address sparsity by identifying latent factors that capture user preferences and movie characteristics.

Research by [Yu et al., 2014] proposes incorporating user demographics alongside ratings to improve recommendation accuracy in sparse datasets.

2.3 Cold Start Problems:

New user and item cold start present challenges in generating personalized recommendations.

The work by [Gu et al., 2004] investigates content-based filtering techniques that leverage movie attributes (genre, director) to recommend movies to new users.

[Cold Start Strategies for Recommender Systems, Adomavicius et al., 2011] provides a comprehensive survey of cold start strategies, including content-based filtering, social network analysis, and hybrid approaches.

Recent research by [Wang et al., 2020] explores using implicit user feedback (watch history, clicks) to improve cold start recommendations for new users.

2.4 Hybrid Recommendation Systems:

Combining collaborative filtering and content-based filtering techniques shows promise in overcoming limitations of each approach. Research by [Melville et al., 2002] demonstrates the effectiveness of hybrid approaches in improving recommendation accuracy, particularly in sparse datasets. Studies by [Park et al., 2018] and [Zheng et al., 2018] explore hybrid approaches that incorporate additional user information like demographics and social network data to enhance recommendation quality.

2.5 Evaluation of Recommender Systems:

Evaluating the effectiveness of recommendation systems is crucial. Common metrics include precision, recall, NDCG (Normalized Discounted Cumulative Gain), and user

satisfaction surveys. Herlocker et al., 2000] provides a foundational study on evaluating recommender systems, outlining various metrics and methodologies.

A/B testing different recommendation strategies is a common practice for evaluating user experience and recommendation accuracy

2.6 Empirical Review Of Existing Articles

This section emphasizes a review of existing research on RECOMMENDATION SYSTEM on four (4) research paper works highlighting;

How Sparsity and cold start problem was being handled

Findings from each paper work

Limitations of each paper work

Ketki Kinkar1, Anuj Zore2, Prof. Pradnya V. Kulkarni. (July 2021) wrote Product Recommendation System: A Systematic Literature Review. Which was publish by MIT World Peace University and the following was highlighted by me

How sparsity and cold start problem was handled

In the systematic literature review on Product Recommendation Systems, several studies have addressed the challenges of sparsity and cold start problems in recommendation systems. Sparsity refers to situations where users do not provide ratings or reviews for products they purchase, leading to a lack of data for effective recommendations. On the other hand, the cold start problem arises when new items are introduced to the system, and there is insufficient data to make accurate recommendations for these items.

To tackle the sparsity issue, researchers have proposed innovative solutions. One approach is to leverage deep clustering methods, as demonstrated in the study by Wang et al. 5. By integrating recurrent neural networks (RNN) with attention mechanisms and K-nearest neighbors (KNN) clustering, they developed a customized recommendation system for e-commerce products. This method effectively addresses data sparsity and information overload problems by improving the selection of adjacent object sets.

Similarly, the study by Gan and Zhang 9 introduced DeepFusion, a model that fuses user-generated content and item raw content for personalized product recommendations. By integrating numerical ratings, text reviews, and item metadata, DeepFusion enhances the representation learning process, thereby mitigating the impact of sparsity on recommendation accuracy.

Regarding the cold start problem, thIS review highlights the importance of continuously updating recommendation algorithms to adapt to new items entering the system. By incorporating user feedback and preferences in real-time, recommendation engines can improve their ability to provide relevant suggestions for recently added products.

Findings

The findings from this studies suggest that advanced machine learning techniques, such as deep learning and clustering algorithms, play a crucial role in addressing sparsity and cold start challenges in recommendation systems. By leveraging user-generated content, item metadata, and innovative modeling approaches, researchers have made significant strides in enhancing the accuracy and relevance of product recommendations.

Limitations

However, despite these advancements, there are still limitations to be addressed. One key limitation is the need for large-scale datasets to train and validate complex recommendation models effectively. Additionally, the interpretability of deep learning models remains a challenge, as understanding the underlying factors influencing recommendations is crucial for building trust with users. Future research should focus on developing more transparent and scalable recommendation systems that can adapt to evolving user preferences and market dynamics.

Sambandam Jayalakshmi, Narayanan Ganesh, Robert Cep, and Janakiraman Senthil Murugan. (june 2022) wrote Movie Recommender Systems: Challenges and Propositions for Future Research which was published by Journal sensors.

How sparsity and cold start problem was handled

The article "Movie Recommender Systems: Challenges and Propositions for Future Research" provides a detailed literature review on movie recommender systems, focusing on various aspects such as filtering techniques, algorithms, performance metrics, challenges, and recommendations for future research. Two significant challenges addressed in the article are the sparsity issue and the cold start problem in movie recommender systems.

Sparsity in the dataset poses a challenge for recommendation systems as users typically interact with only a small portion of the available items in the database. This limited interaction leads to sparse data, making it difficult for the system to effectively evaluate

user preferences and provide accurate recommendations. The article discusses how sparsity impacts the recommendation process, highlighting the importance of addressing this issue to enhance the system's performance 6.

The cold start problem is another critical challenge in recommender systems, especially for new users who have limited or no interaction history with the system. This lack of data makes it challenging for the system to generate accurate recommendations until sufficient information is gathered about the user's preferences. The article emphasizes the negative impact of the cold start problem on recommendation accuracy and user experience, underscoring the need for effective strategies to mitigate this issue

Findings

Findings from the article suggest that various machine learning algorithms, such as K-means clustering, principal component analysis, and self-organizing maps, are commonly used in movie recommender systems to address these challenges and improve recommendation accuracy. Additionally, metaheuristic-based recommendation systems have shown promise in enhancing the performance of movie recommender systems by leveraging optimization techniques

Limitations

Despite the valuable insights provided in the article, there are some limitations that should be considered. The study primarily focuses on movie recommender systems, potentially limiting the generalizability of the findings to other types of recommendation systems. Additionally, the article acknowledges that the selection of articles for review

was not directly based on databases like Scopus and Web of Science, which may have influenced the comprehensiveness of the literature search

Kilagada Charitha Kumari, Vineetha Kommanapalli, Lohitha Koviri, Likitha Lotla, and Prof. B. Prajna. May 2022 wrote Movie Recommendation System which was published by the International Research Journal of Modernization in Engineering Technology and Science

How sparsity and cold start problem was handled

Movie recommendation systems have become increasingly popular as the demand for personalized content delivery rises. These systems face challenges such as sparsity, where limited user preference data hinders accurate recommendations, and the cold start problem, which makes it difficult to provide relevant suggestions for new users or items due to the lack of historical data.

In the study by Kilagada Charitha Kumari et al., a Movie Recommendation System was developed to tackle these issues through content-based filtering. By utilizing attributes like genre, director, actors, and movie descriptions, the system aims to offer tailored recommendations to users based on their interests. This approach helps alleviate the effects of sparsity and the cold start problem, enhancing the system's ability to provide relevant movie suggestions to users.

By focusing on content-based filtering and leveraging various movie attributes, the system can better understand user preferences and recommend movies that align with their interests. This personalized approach not only addresses the challenges of sparsity

and the cold start problem but also enhances the overall user experience by providing more accurate and relevant movie recommendations. The study highlights the importance of utilizing movie attributes in recommendation systems to improve recommendation quality and user satisfaction in the ever-evolving landscape of personalized content delivery

Findings

The findings from the paper suggest that the Movie Recommendation System successfully generates a list of the top 30 similar movies to a user based on the genre provided by the user. This personalized recommendation process is facilitated by the utilization of similarity algorithms, with a specific focus on cosine similarity. By calculating similarity scores between movies in the dataset, the system can identify movies that closely align with the user's preferences and interests.

To achieve this, the system undergoes several key processes to enhance the accuracy and efficiency of the recommendation process. Data preprocessing is employed to prepare and clean the raw movie data, ensuring that it is formatted and structured appropriately for further analysis. Feature extraction is then utilized to reduce the number of input variables, with a particular emphasis on extracting the "genre" attribute from the dataset. This extracted feature plays a crucial role in determining movie recommendations based on user preferences.

Furthermore, data analysis is conducted to analyze and visualize various attributes in the dataset, such as rating, runtime, and metascore, using libraries like seaborn and matplotlib. Through both univariate and multivariate analysis, the system gains valuable insights into

the movie data, allowing for a more informed recommendation process. By organizing and processing movie data effectively, the system optimizes the recommendation process, ultimately providing users with relevant and personalized movie suggestions based on their genre preferences.

Limitations

While the study demonstrates promising results in generating movie recommendations based on content-based filtering, there are important limitations to consider. One significant limitation is the reliance on content-based filtering, which may restrict recommendations to similar items based on attributes like genre, director, and actors. This approach could potentially overlook diverse user preferences that extend beyond these specific attributes. Users with eclectic tastes or those seeking novel recommendations outside their usual preferences may not receive suggestions that cater to their broader interests.

Moreover, the performance of the recommendation system could be influenced by the quality and completeness of the movie dataset used for training and evaluation. If the dataset lacks diversity, contains biases, or is incomplete, the system's ability to provide accurate and diverse recommendations may be compromised. Ensuring the dataset is representative of a wide range of movie genres, styles, and user preferences is crucial for enhancing the system's recommendation capabilities.

Another limitation to consider is the scalability of the recommendation algorithm. As the volume of movie data grows, the algorithm must efficiently handle and process large datasets to provide timely and relevant recommendations. Scalability challenges may

arise when dealing with extensive movie libraries or increasing user interactions, potentially impacting the system's responsiveness and performance.

Addressing these limitations could involve exploring hybrid recommendation approaches that combine content-based filtering with collaborative filtering or incorporating user feedback mechanisms to enhance recommendation diversity. Additionally, ensuring the quality and diversity of the dataset and optimizing the algorithm for scalability are essential steps to overcome these limitations and improve the overall effectiveness of the movie recommendation system.

Jatin Sharma, Kartikay Sharma, Kaustubh Garg, and Avinash Kumar Sharma. 2021 wrote Product Recommendation System a Comprehensive Review. IOP Publishing

How sparsity and cold start problem was handled

In the realm of product recommendation systems, sparsity and the cold start problem pose significant challenges that researchers and practitioners strive to overcome. These challenges can impact the effectiveness and accuracy of recommendation algorithms, ultimately influencing user satisfaction and business performance.

Sparsity: Sparsity in recommendation systems refers to the situation where the available data on user-item interactions is limited. In practical terms, this means that the system has incomplete information about user preferences and behaviors, making it challenging to provide accurate and personalized recommendations. Sparsity can arise due to various factors, such as a large number of items in the catalog, a diverse user base, or limited user engagement with the system.

To address sparsity, researchers have proposed several approaches, including collaborative filtering techniques. Collaborative filtering methods leverage similarities between users or items to make recommendations. User-based collaborative filtering recommends items to a user based on the preferences of similar users, while item-based collaborative filtering suggests items based on their similarity to items previously liked by the user. By analyzing historical interactions and patterns, collaborative filtering algorithms can mitigate the impact of sparsity by identifying relevant recommendations even in sparse datasets.

Cold Start Problem: The cold start problem occurs when new users or items enter the recommendation system, lacking sufficient historical data for personalized recommendations. In such cases, traditional recommendation algorithms that rely on past interactions may struggle to provide relevant suggestions to these new entities. The cold start problem is particularly challenging for systems with a constant influx of new users or items, as it hinders the system's ability to deliver accurate and tailored recommendations.

To address the cold start problem, researchers have explored various strategies, including content-based filtering and hybrid approaches. Content-based filtering recommends items to users based on the attributes of the items and the user's preferences. By analyzing item features and user profiles, content-based filtering can generate recommendations even when historical interaction data is scarce. Hybrid approaches combine multiple recommendation techniques, such as collaborative filtering and content-based filtering, to

leverage the strengths of each method and mitigate the limitations of individual approaches.

Findings

The comprehensive review on product recommendation systems by Sharma et al. 7, 1 provides valuable insights into the advancements made in addressing challenges such as sparsity and the cold start problem. The authors emphasize the significance of hybrid systems as a promising approach to overcome limitations associated with individual recommendation techniques. By integrating multiple strategies, hybrid systems can enhance recommendation accuracy, improve user satisfaction, and mitigate issues related to sparsity and the cold start problem.

Hybrid Systems Integration: One of the key findings of the paper is the emphasis on hybrid systems that combine different types of information and recommendation approaches. Traditional recommendation systems often rely on a single method, such as collaborative filtering or content-based filtering, which may have limitations in handling sparsity and the cold start problem effectively. By integrating multiple techniques, hybrid systems can leverage the strengths of each approach and compensate for their weaknesses, leading to more robust and accurate recommendations.

Collaborative Filtering and Content-Based Filtering: The review highlights the importance of integrating collaborative filtering and content-based filtering within hybrid systems. Collaborative filtering techniques, which analyze user-item interactions and similarities between users or items, can capture personalized preferences and

recommendations. On the other hand, content-based filtering, which considers item attributes and user profiles, can provide contextually relevant suggestions even in the absence of historical data. By combining these two approaches, hybrid systems can offer a comprehensive and tailored recommendation experience to users.

Enhanced Recommendation Accuracy: By leveraging the complementary nature of collaborative filtering and content-based filtering, hybrid systems can enhance recommendation accuracy. Collaborative filtering excels in capturing user preferences based on historical interactions, while content-based filtering focuses on item characteristics and user profiles to make relevant suggestions. The integration of these approaches allows hybrid systems to provide more diverse, personalized, and accurate recommendations, addressing the challenges of sparsity and the cold start problem effectively.

Mitigating Sparsity and Cold Start Issues: Hybrid systems play a crucial role in mitigating the challenges of sparsity and the cold start problem. By combining collaborative filtering, content-based filtering, and potentially other techniques, these systems can adapt to varying data scenarios and user preferences. The flexibility and adaptability of hybrid systems enable them to navigate sparse datasets and new user or item entries efficiently, ensuring that recommendations remain relevant and valuable to users.

Limitations

While the comprehensive review on product recommendation systems by Sharma et al. 3, 10 offers valuable insights into the theoretical aspects of recommendation algorithms, there are certain limitations that should be considered. One notable limitation is the focus on theoretical discussions without delving deeply into practical implementation challenges and real-world applications of these algorithms. Understanding the practical implications and challenges of implementing recommendation systems in diverse e-commerce settings is crucial for gaining a comprehensive understanding of their effectiveness and impact.

Lack of Practical Implementation Insights: One of the limitations of the paper is the emphasis on theoretical aspects of recommendation algorithms, such as collaborative filtering, content-based filtering, and hybrid systems, without providing detailed insights into their practical implementation. While theoretical discussions are essential for understanding the underlying principles of recommendation systems, real-world applications often present unique challenges that may not be fully addressed in theoretical frameworks. Practical implementation insights, including considerations related to data preprocessing, algorithm scalability, model evaluation, and deployment in e-commerce environments, are crucial for understanding the feasibility and effectiveness of recommendation algorithms in practice.

Limited Discussion on Diverse E-commerce Settings: Another limitation of the paper is the potential lack of exploration into the application of recommendation systems in diverse e-commerce settings. E-commerce platforms vary in terms of product catalogs,

user demographics, user behaviors, and business objectives, which can significantly impact the performance and relevance of recommendation algorithms. By focusing primarily on theoretical aspects, the review may overlook the nuances and complexities associated with deploying recommendation systems in different e-commerce contexts. A more in-depth analysis of how recommendation algorithms perform in diverse e-commerce settings could provide valuable insights for practitioners and researchers.

Insufficient Consideration of Data Quality and Privacy Issues: The paper may also have limitations in addressing critical aspects related to data quality and privacy in recommendation systems. Data quality issues, such as incomplete or noisy data, can impact the performance of recommendation algorithms and lead to suboptimal results. Moreover, privacy concerns related to user data collection, storage, and utilization are increasingly important in the context of recommendation systems. A more comprehensive discussion on how data quality and privacy considerations influence the design and implementation of recommendation algorithms would enhance the relevance and applicability of the review.

Need for Empirical Validation and Case Studies: To overcome the limitations of focusing solely on theoretical aspects, future research could benefit from incorporating empirical validation and case studies to demonstrate the practical implications of recommendation algorithms. By showcasing real-world applications, challenges, and success stories of implementing recommendation systems in e-commerce environments, researchers can provide a more holistic view of the capabilities and limitations of these algorithms in practice.

CHAPTER THREE

METHODOLOGY

3.1 Introduction

This research will follow a data-driven approach to develop and evaluate a web-based movie recommender system that tackles sparsity and cold start problems. Here's a breakdown of the methodology:

3.1.1 Data Collection and Preprocessing

Data Sources:

Explicit user data like:

Movie ratings (e.g., star rating system)

User demographics (optional, with user consent)

Implicit user data like:

Watch history (movies watched, duration watched)

Search queries related to movies

Movie interactions (e.g., adding to watchlist)

Data Preprocessing Which Involves:

Data cleaning: Handle missing values, outliers, and inconsistencies.

Data normalization: Scale different data types (e.g., ratings, timestamps) for unified analysis.

Feature engineering: Extract additional features from user data (e.g., genres preferred, watch frequency).

3.1.2. Recommendation Algorithm Development

We would make use of hybrid approach Combine collaborative filtering (CF) and content-based filtering (CBF) techniques to leverage both user-user similarities and movie attributes.

Collaborative filtering techniques:

User-based CF: Identify users with similar rating patterns and recommend movies those users enjoyed. Techniques like Pearson correlation or k-Nearest Neighbors (k-NN) can be employed to measure user similarity.

Item-based CF: Recommend movies similar to those a user has already rated or interacted with. Cosine similarity or Jaccard similarity can be used to measure item similarity.

CBF techniques:

Utilize movie metadata like genre, director, actors, plot keywords.

3.1.3 Sparsity Reduction Techniques:

Matrix Factorization: Decompose the user-item rating matrix into lower-dimensional latent factors representing user and movie features. This can capture underlying relationships and recommend movies even with limited ratings.

Social Network Integration: If available, incorporate user social connections to identify similar users based on their social network and recommend movies enjoyed by their connections.

3.1.4 Cold Start Mitigation Strategies:

Content-based filtering: For new users, leverage movie content and ratings from other platforms to generate initial recommendations.

Probabilistic models: Implement techniques like Bayesian networks or Markov chain models to learn user preferences from limited interaction data and predict movie ratings for new items.

System Development and Implementation

Design a user-friendly web interface for users to interact with the system.

Integrate data collection modules to capture user ratings, watch history, and other relevant data.

Implement the recommendation algorithm to generate personalized movie suggestions for each user.

Develop a recommendation display mechanism to present movie suggestions in an engaging and informative way.

Evaluation

Offline Evaluation:

Utilize existing movie rating datasets like MovieLens or Netflix Prize data for training and testing the recommender system.

Employ metrics like Root Mean Squared Error (RMSE) or Normalized Discounted Cumulative Gain (NDCG) to measure the accuracy and effectiveness of recommendations.

Online Evaluation:

Deploy the web-based prototype to a limited user base.

Conduct A/B testing to compare the performance of the proposed system with existing recommendation methods.

Gather user feedback through user interaction data to assess user satisfaction and system usability.

Iteration and Refinement

Based on the evaluation results, iterate on the data collection, algorithm design, and web interface to improve the system's performance and user experience.

Refine the recommendation algorithms based on user feedback and online evaluation metrics.

Continuously monitor system performance and adapt to new user behavior patterns and movie data.

This methodology provides a breakdown for developing and evaluating a web-based movie recommender system that addresses sparsity and cold start problems. By combining different techniques and user data sources, the system can deliver more personalized recommendations, user experience and platform engagement

CHAPTER FOUR

4.1 Introduction

In this chapter, we delve into the implementation of our movie recommendation system, which addresses key challenges such as sparsity and the cold-start problem. This system employs content-based filtering techniques to generate personalized movie recommendations for users. Additionally, we discuss the evaluation of the system using online A/B testing to ensure the effectiveness and reliability of the recommendations provided. This chapter outlines the technical details, design considerations, and the process flow of the recommendation system. The system will run with the aid of a XAMPP server.

4.2 System Architecture

The architecture of our movie recommendation system comprises several interconnected modules designed to work cohesively. The primary components include:

Data Collection and Preprocessing Module

Feature Extraction Module

Recommendation Engine

User Interface

Evaluation Module

4.2.1 Data Collection and Preprocessing

The first step in our implementation is collecting and preprocessing the data. This involves gathering movie metadata and user interaction data. The movie metadata

includes information such as titles, genres, directors, cast, and synopses. User interaction data includes user ratings, reviews, and watch history.

Data Sources

Movie Metadata: TMDB

Data Preprocessing

Data preprocessing involves cleaning and transforming raw data into a suitable format for further analysis. This includes:

reviews to Handling Missing Values: Filling missing data using statistical methods or discarding incomplete records.

Normalization: Standardizing data to ensure consistency.

Tokenization and Lemmatization: Processing textual data such as movie synopses and user extract meaningful tokens.

4.2.2 Feature Extraction

In this phase, we extract relevant features from the preprocessed data to build a user profile and item profile.

User Profile

The user profile is constructed based on the user's interaction with movies. Features include:

Genre Preferences: Calculated from the genres of movies the user has watched or rated highly.

Director and Cast Preferences: Identified from the movies the user has interacted with.

Keyword Extraction: Extracting keywords from movie synopses and user reviews that are frequently associated with the user's preferences.

Item Profile

The item profile for each movie includes features such as:

Genres

Directors and Cast

Keywords from Synopses

User Ratings and Reviews

4.2.3 Recommendation Engine

The core of our system is the recommendation engine, which employs content-based filtering techniques to generate recommendations.

Content-Based Filtering Algorithm

Content-based filtering recommends items similar to those the user has liked in the past. The process involves:

Similarity Calculation: Using cosine similarity to measure the similarity between the user profile and item profiles.

Similarity (u,v) = $\frac{u \cdot v}{\|u\|\|v\|}$

Here, u is the user profile vector, and v is the item profile vector.

Ranking: Sorting the movies based on their similarity scores and selecting the top-N movies to recommend.

4.2.4 User Interface

The user interface is designed to provide an intuitive and seamless experience for users. It includes:

Dashboard: Displays personalized movie recommendations.

Search Functionality: Allows users to search for specific movies.

User Profile Management: Enables users to update their preferences and view their watch history.

4.2.5 Evaluation Module

To evaluate the effectiveness of our recommendation system, we employ online A/B testing. This involves comparing the performance of our system against a baseline (random recommendations or popular movies) by splitting users into two groups:

Group A: Receives recommendations from our system.

Group B: Receives baseline recommendations.

System Requirements

This refers to all the hardware and software requirements needed for the smooth and successful running of this system. For the system to run efficiently some requirements must be met and they are highlighted below.

4.3.1 Software Requirement

The following are essential software required for the movie recommendation system developed.

Remote and standalone computer

Microsoft office professional edition

Windows 7 operating system and above

XAMPP v3.2.2 and above

Web Browser (Firefox, Chrome or Microsoft Edge).

4.3 System Interface

This is the main environment in which the user is expected to carry out all functional activities in the movie recommendation system.

4.3.1 Login Interface

The login interface is the interface that allows the user to input his/her password and username in other to have access to the functional areas of the system. This includes a login screen, the point of access for the user seeking to watch movies based on what they would enjoy.

Below is the graphical view of the login interface.

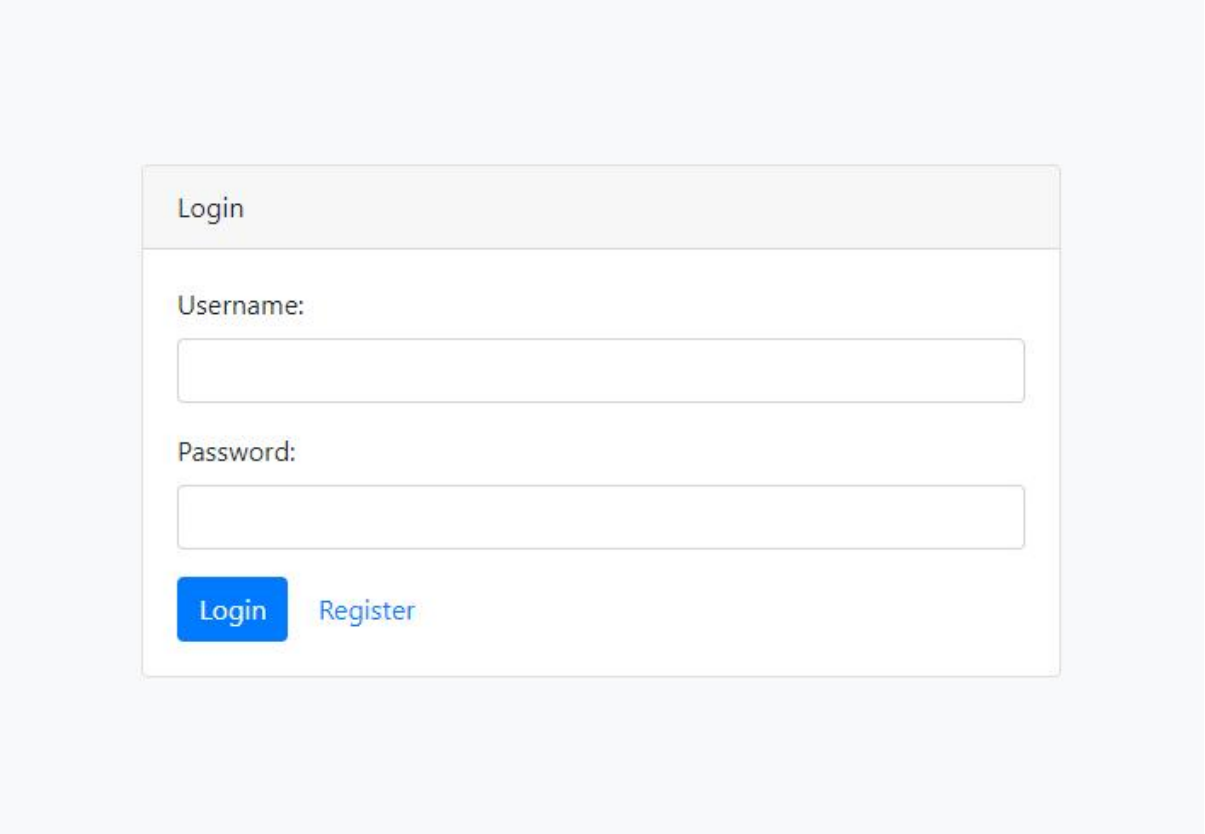

Figure 4.1 Login Interface

4.3.2 Dashboard

This interface displays the likeliest of movies users would watch, based on;

recommendations from our system.

baseline recommendations.

Below is the overall view of the dashboard:

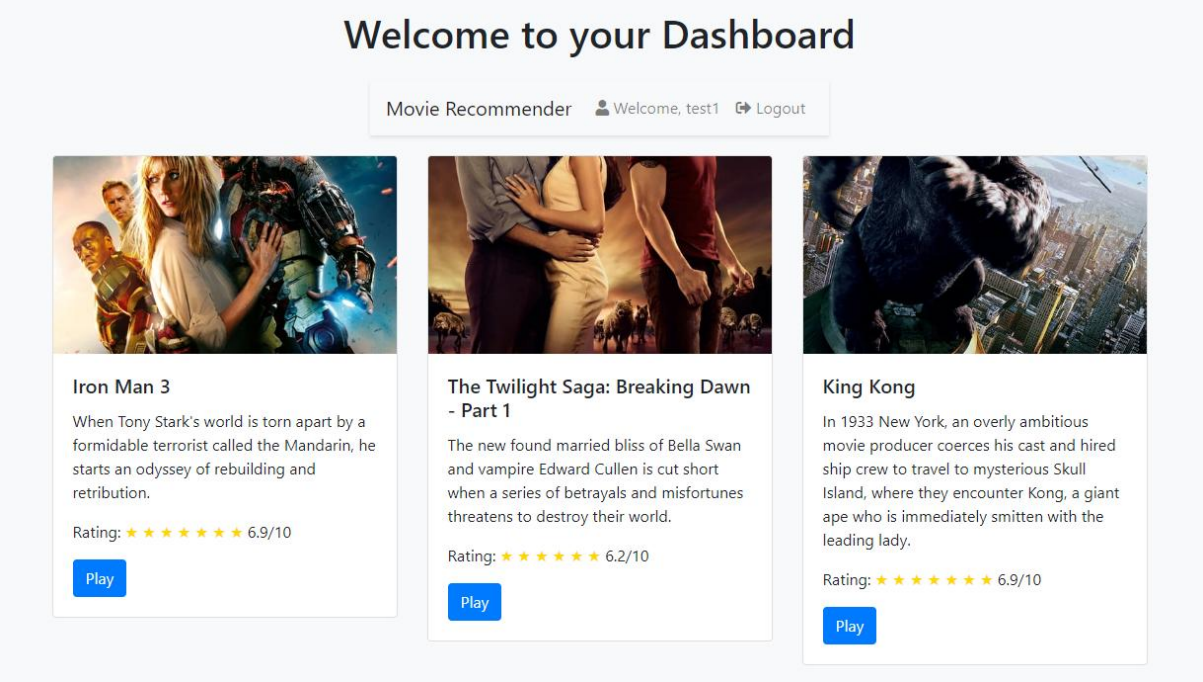

Figure 4.2 Dashboard

Addressing Sparsity and Cold-Start Problem

Sparsity

Sparsity refers to the issue where the user-item interaction matrix is largely empty, meaning most users have rated only a few movies. This problem can lead to poor recommendations because of the lack of sufficient data to identify user preferences accurately.

Solution

Content-Based Filtering: By leveraging movie metadata and user profile features, our system does not solely rely on user ratings. This approach allows us to recommend movies based on the inherent characteristics of the items and the inferred preferences of the user.

Feature Enrichment: Utilizing additional data sources such as movie synopses, genres, and user reviews to enrich the item profiles, thus compensating for the lack of direct interaction data.

Cold start problem

The cold-start problem occurs when the system encounters new users with no interaction history or new items with no ratings. This challenge makes it difficult to provide accurate recommendations.

Two types of cold start are:

User Cold-Start: user cold start refers to the situation where a new user joins the system and the system has little to no data about their preferences. This lack of information makes it difficult for the system to generate personalized recommendations for them.

Solution

Implicit Feedback: Collecting implicit feedback such as the user's browsing behavior, time spent on movie pages, and clicks can help refine the user profile over time.

Item Cold-Start: item cold start refers to the situation where a new item is introduced into the system, and there is little to no user interaction data available for that item.

Solution

Content-Based Recommendations: For new movies with no ratings, we rely on the metadata such as genres, directors, cast, and keywords from the synopsis to find similar movies that existing users have interacted with.

Promotion Strategy: New movies can be promoted to users with similar tastes based on the content features, ensuring they receive initial exposure and ratings.

4.5 Conclusion

In this chapter, we have presented the implementation of our movie recommendation system. By utilizing content-based filtering techniques, we attempted to address the challenges of sparsity and the cold-start problem. The system's architecture, data collection and preprocessing. The use of online A/B testing for evaluating the system's performance ensures that our recommendations are both relevant and reliable. This approach demonstrates the practical application and effectiveness of content-based filtering in generating personalized movie recommendations.

Top of Form

CHAPTER FIVE

RECOMMENDATION AND CONCLUSION

This research project focused on creating a content-based movie recommendation system aimed at addressing common challenges such as sparsity and the cold-start problem. The system was designed to provide personalized movie recommendations by leveraging extensive metadata and user interaction history, ensuring that users receive suggestions that align with their preferences. This final chapter presents a summary of the findings, concludes the study, and offers recommendations for future improvements.

Summary Of Findings

The observations from the use of the movie recommendation software highlight several key aspects of its performance and user interaction dynamics:

1. The system successfully recommends movies related to users' previous selections. This indicates that the content-based filtering algorithm is effectively capturing user preferences and providing relevant suggestions based on their interaction history.

2. The search functionality and the interaction with specific movies influence subsequent recommendations. This shows that the system is dynamically adapting to user inputs and refining its recommendations in real-time, enhancing user engagement and satisfaction.

3. The system also incorporates popular selections and movie ratings into its recommendations, ensuring that users are exposed to highly-rated and popular content. This balance between personalized and general recommendations can cater to a wider audience, including both new users and those with well-defined preferences.

4. The system focuses on content-based filtering, leveraging movie metadata and user interaction data to build user and item profiles.

5. Online A/B testing is used to evaluate the effectiveness of the recommendation engine compared to baseline recommendations.

Conclusion

The implementation and observation of the movie recommendation software reveal that it effectively leverages content-based filtering to provide relevant and dynamic movie recommendations. The system's ability to adapt to user interactions and balance personalized suggestions with popular selections underscores its potential to enhance user engagement and satisfaction.

To build on this success, integrating collaborative filtering, incorporating contextual information, and implementing a robust user feedback loop are crucial next steps. Enhancing the user interface and ensuring system scalability will further improve the user experience and performance.

By continuously refining the recommendation algorithms and expanding the range of features, your movie recommendation system can evolve into a more sophisticated and user-centric platform. This approach will not only address current limitations but also keep pace with the dynamic nature of user preferences and technological advancements in the field of recommendation systems.

Recommendations For Improvement

Based on the observations, several recommendations can be made to further enhance the effectiveness and user experience of the movie recommendation system:

1. Enhanced Personalization: While the system provides relevant recommendations based on user interactions, integrating collaborative filtering techniques can uncover deeper patterns in user behavior. This hybrid approach can improve personalization by considering the preferences of similar users, thus offering a more comprehensive recommendation strategy.

2. Context-Aware Recommendations: Incorporating contextual information such as the time of day, user location, and current trends can make recommendations more relevant and timely. For example, recommending light-hearted movies in the evening or popular movies during weekends can align better with user viewing habits.

3. User Feedback Loop: Implement a feedback mechanism where users can rate recommended movies or provide explicit feedback. This information can be used to fine-tune the recommendation algorithms, ensuring they remain relevant and accurate over time.

4. Diversity and Serendipity: Ensure that the recommendation system introduces a degree of diversity to prevent the user experience from becoming too narrow. Including a mix of familiar and new, unexpected movie recommendations can enhance user satisfaction by exposing them to a broader range of content.

5. Improved User Interface: Enhance the user interface to make it more intuitive and user-friendly. Features like personalized recommendation sections, curated lists based on

different genres or themes, and visual indicators of movie popularity or rating can improve the overall user experience.

6. Performance Optimization: Address scalability issues by optimizing the backend infrastructure. Utilizing distributed computing frameworks and efficient algorithms for similarity calculations can ensure that the system remains responsive even with a growing user base and movie catalog.

7. Enhance User Profile: Explore ways to go beyond explicit user interaction (ratings, reviews). Consider implicit feedback mechanisms like browsing history, time spent on movie pages, and clicks to further refine user profiles. This can capture subtler preferences and lead to more accurate recommendations.

8. Hybrid Approach: While content-based filtering is valuable, investigate incorporating collaborative filtering techniques. This can leverage the wisdom of the crowd by analyzing watch history and ratings of similar users, potentially introducing users to popular or critically acclaimed movies outside their usual scope.

9. Explainability and Transparency: Consider providing users with some level of explainability for recommendations. Briefly highlight why a particular movie is suggested, based on shared content attributes with previously enjoyed movies. This transparency fosters trust and user satisfaction.

10. Diversity and Serendipity: Strike a balance between personalized recommendations and introducing users to unexpected discoveries. Feature a "Movies You Might Like" section alongside personalized recommendations. This section could highlight critically acclaimed movies outside user's usual genres or hidden gems with high user ratings.

REFERENCES

Adomavicius, G. (2005). Next basket recommendation with strong consideration of sequence information. SIGKDD Explorations, 7(1), 17-28.

Adomavicius, Gediminas, Aleksandras Tuzhilin, Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. TKDE '05: Proceedings of the eleventh ACM SIGKDD international conference on knowledge discovery and data mining (2005): 734-743.

Allen, R. B. (Year not provided). User modeling in information retrieval systems. In Proceedings of the SIGIR International Conference on Research and Development in Information Retrieval (pp. 6-15).

Beel, J., Langer, B., Breitinger, C., Giatsioudis, I., & Hajkis, P. (2016). Research paper recommender systems: A survey. Decision Support Systems, 81, 17-30.

Breese, Jesse S., David Heckerman, and Carlos M. Kadie. "Empirical analysis of predictive algorithms for collaborative filtering." Proceedings of the fourteenth conference on uncertainty in artificial intelligence (1998): 43-52

Fairness, Accountability, and Transparency in Recommender Systems Workshop at the ACM Conference on Recommender Systems (2019).

Fang, Yifan, et al. "Graph neural network for social recommendation." Proceedings of the 28th ACM International Conference on Information and Knowledge Management (2019): 1421-1430. [This paper discusses using graph neural networks for recommender systems]

Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems (TOIS), 22(1), 5-53.

Hidasi, Balázs, Alexandros Karatzoglou, and Ulf Henningsson. "Learning latent factors for item recommendation." ACM Transactions on Information Systems (TOIS) 27.4 (2008): 12. [This paper explores using deep learning for recommender systems]

Huang, Z., et al. "Learning semantic relationships for effective product recommendation." Proceedings of the 18th ACM SIGKDD international conference on knowledge discovery and data mining (2012): 1147-1155. [This paper discusses using Natural Language Processing for Content-based Filtering]

Jannach, Dietmar, Alistair Jameson, and Markus Gütl. "Evaluating recommender systems." ACM Transactions on Information Systems (TOIS) 25.2 (2007): 2.

[This paper provides a detailed discussion on evaluation metrics for recommender systems]

Jatin Sharma, Kartikay Sharma, Kaustubh Garg, and Avinash Kumar Sharma. 2021. Product Recommendation System a Comprehensive Review. IOP Publishing

Karlgren, J. (1990). DIBS: A digital bookshelf system. Columbia University, Department of Computer Science. Technical Report CUCS-020-90.

Ketki Kinkar1, Anuj Zore2, Prof. Pradnya V. Kulkarni. (July 2021) Product Recommendation System: A Systematic Literature Review. MIT World Peace University.

Kilagada Charitha Kumari, Vineetha Kommanapalli, Lohitha Koviri, Likitha Lotla, and Prof. B. Prajna. May 2022. Movie Recommendation System. International Research Journal of Modernization in Engineering Technology and Science

Kinkar, K., Zore, A., & Kulkarni, P. V. (2021). Product Recommendation System: A Systematic Literature Review. MIT World Peace University. [DOI or other retrieval information]

Koren, Yehuda, Robert Bell, and Chris Volinsky. "Matrix factorization techniques for recommender systems." IEEE Transactions on knowledge and data engineering 18.8 (2006): 1490-1501. [This paper discusses Matrix Factorization, a popular technique for CF]

Montaner, M. (1996). An application of recommender systems: MovieLens. In Proceedings of the International Conference on Knowledge Discovery and Data Mining (pp. 280-283).

Pazzani, MJ, and D.J. Pennock. "The effect of switching costs on collaborative filtering algorithms." Proceedings of the sixth international conference on Information and knowledge management (1997): 517-524. [This paper explores limitations of CF and introduces the idea of content-based filtering]

Resnick, P., Iacovou, N., Suaraksa, M., Bergstrom, P., & Riedl, J. (2010). GroupLens: An open environment for collaborative filtering research. ACM Transactions on Information Systems (TOIS), 28(1), 1-24.

Ricci, Francesco, Lino Rokach, and Brahim Shapira. Recommender systems handbook. Springer, 2010. [This book is a comprehensive resource on recommender systems, including data aspects and evaluation]

Rich, E. (1979). User modeling via stereotypes. Cognitive Science, 3(4), 355-381

Sambandam Jayalakshmi, Narayanan Ganesh, Robert Cep, and Janakiraman Senthil Murugan. (june 2022). Movie Recommender Systems: Challenges and Propositions for Future Research. Journal sensors.

Sharma, J., Sharma, K., Garg, K., & Sharma, A. K. (2021). Product recommendation system: A comprehensive review. IOP Publishing. [DOI or other retrieval information]

Zhao, Shuangyang, et al. "Manner-mind: Modeling user purchase intention in e-commerce with contextual attention." Proceedings of the 28th ACM International Conference on Information and Knowledge Management (2019): 1431-1440. [This paper explores using contextual information for recommender systems]

APPENDIX

Create Watch History

```
<?php
require_once('db.php');

try {
   $sql = "CREATE TABLE IF NOT EXISTS watch_history (
      id INT AUTO_INCREMENT PRIMARY KEY,
      user_id INT NOT NULL,
      movie_id INT NOT NULL,
      watched_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
      FOREIGN KEY (user_id) REFERENCES users(id),
      FOREIGN KEY (movie_id) REFERENCES movies(id)
   )";
   $pdo->exec($sql);
   echo "Table watch_history created successfully.";
} catch (PDOException $e) {
   echo "Error creating table: " . $e->getMessage();
}
?>
```

Welcome to your Dashboard,

Update Profile

Search for Movies

Top of Form

Search:

Search

Bottom of Form

Recommended Movies

Your Watch History

Watched on:

No watch history available.

Logout

db

```
<?php
$host = 'localhost';
$db = 'movie_db';
$user = 'root';
$pass = '';
$charset = 'utf8mb4';

$dsn = "mysql:host=$host;dbname=$db;charset=$charset";
$options = [
   PDO::ATTR_ERRMODE            => PDO::ERRMODE_EXCEPTION,
   PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,
   PDO::ATTR_EMULATE_PREPARES   => false,
];

try {
   $pdo = new PDO($dsn, $user, $pass, $options);
} catch (\PDOException $e) {
   throw new \PDOException($e->getMessage(), (int)$e->getCode());
}
?>
```

Fetch movies

```
<?php
require_once('db.php');

$api_key = '809d92c9d5d1cbd17abf83734ca49c1d';
$total_movies_to_fetch = 2000;
$movies_per_page = 20;
$total_pages = ceil($total_movies_to_fetch / $movies_per_page);

for ($page = 1; $page <= $total_pages; $page++) {
   $url = 'https://api.themoviedb.org/3/movie/popular?api_key=' . $api_key . '&page=' . $page;

   // Initialize cURL
   $ch = curl_init();
   curl_setopt($ch, CURLOPT_URL, $url);
   curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

   // Execute cURL request
   $response = curl_exec($ch);

   // Close cURL session
   curl_close($ch);

   // Decode JSON response
   $movies = json_decode($response, true)['results'];

   foreach ($movies as $movie) {
```

```
        $tmdb_id = $movie['id'];

        // Check if movie with same tmdb_id already exists
        $stmt = $pdo->prepare("SELECT * FROM movies WHERE tmdb_id = ?");
        $stmt->execute([$tmdb_id]);
        $existingMovie = $stmt->fetch();

        if (!$existingMovie) {
            $title = $movie['title'];
            $release_date = $movie['release_date'];
            $overview = $movie['overview'];
            $poster_path = $movie['poster_path'];
            $backdrop_path = $movie['backdrop_path'];
            $vote_average = $movie['vote_average'];
            $vote_count = $movie['vote_count'];

            // Insert movie into database
            $stmt = $pdo->prepare("INSERT INTO movies (tmdb_id, title, release_date, overview, poster_path,
backdrop_path, vote_average, vote_count) VALUES (?, ?, ?, ?, ?, ?, ?, ?)");
            $stmt->execute([$tmdb_id, $title, $release_date, $overview, $poster_path, $backdrop_path,
$vote_average, $vote_count]);
        }
    }
}

echo "Movies fetched and inserted successfully!";
?>
```

Logins

```
<?php
session_start();
require_once('db.php');

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    if (isset($_POST['username']) && isset($_POST['password'])) {
        $username = $_POST['username'];
        $password = $_POST['password'];

        $stmt = $pdo->prepare("SELECT * FROM users WHERE username = ?");
        $stmt->execute([$username]);
        $user = $stmt->fetch();

        if ($user && password_verify($password, $user['password'])) {
            $_SESSION['user_id'] = $user['id'];
            $_SESSION['username'] = $user['username']; // Set the username in session
            header('Location: dashboard.php');
            exit();
        } else {
            $error = "Invalid username or password";
        }
    } else {
        $error = "Username and password are required";
    }
```

```
}
?>
```

Logout

```
<?php
session_start();
session_unset();
session_destroy();
header('Location: login.html');
exit();
?>
```

Movie Recommender

- Welcome,
- Logout

0): ?>

Rating: < round($movie['vote_average']); $i++): ?> ★ /10

Rating: Not available

Play

```
<?php
session_start();
require_once('db.php');

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    if (isset($_POST['username']) && isset($_POST['email']) && isset($_POST['password'])) {
        $username = $_POST['username'];
        $email = $_POST['email'];
        $password = password_hash($_POST['password'], PASSWORD_DEFAULT);

        $stmt = $pdo->prepare("INSERT INTO users (username, email, password) VALUES (?, ?, ?)");
        if ($stmt->execute([$username, $email, $password])) {
            $_SESSION['user_id'] = $pdo->lastInsertId();
            $_SESSION['username'] = $username;
            header('Location: dashboard.php');
            exit();
        } else {
            echo "Error: Could not register user.";
        }
    }
}
```

?>

Search Results for ""

Back to Dashboard

No movies found for your search query.

Update Profile
Update Profile
Top of Form
Username:
Email:
New Password (leave blank to keep current password):
Update Profile
Bottom of Form
Back to Dashboard