Movie recommendation

System

A Thesis Submitted for fulfillment for the Bachelor Degree in Computer Science (Honor’s Degree) to the department of Software engineering, Kelaniya University

By

Sanka vaas samarasingha

CS/2017/034

# Declaration

I hereby declare that this thesis is a result of my original work and has not been submitted in part or full for any degree or academic purpose elsewhere. The content presented in this thesis is based on my research and analysis, and I have provided proper attribution to all sources used in the form of references and citations. Any contributions made by others have been duly acknowledged.

Furthermore, I affirm that all data, figures, tables, and other materials used in this thesis are authentic, and any manipulation or modifications have been disclosed appropriately. I take full responsibility for the accuracy and integrity of the information presented in this work.

I acknowledge that any intellectual property, copyright, or proprietary rights related to this thesis are the property of the respective owners and have been respected accordingly. In cases where third-party materials have been used, I have obtained permission or licenses as required. I understand that plagiarism is a serious academic offense, and I have made every effort to avoid any form of plagiarism in this thesis. Any direct or indirect use of someone else's work or ideas has been cited and referenced correctly.

I am aware that this thesis will be subjected to scrutiny by the academic committee, and I am ready to defend and clarify any aspect of my work if required.

Date: [Insert Date]

Signature: [You’re Signature]

Supervisor’s

Signature : [ ]

# Abstract

The rapid expansion of online and digital entertainment contents has led to an overwhelming number of options for users. Recommendation systems have emerged because of this abundance of choices. In the case of movie selection, Movie recommendation systems have emerged as an important solution to address the challenge of information overload, by offering viewers individualized suggestions that are in line with their preferences. However, common conventional movie recommendation systems often struggle in accurately providing choices based on diverse user tastes and overcoming data sparsity. In response to these problems, the MovieFlow movie recommendation system is created to improve the accuracy and personalization of movie recommendations to address these challenges.

Objective of this research is to solve the problem of less effectiveness and less accuracy of movie recommendations by integrating a hybrid approach which combines content-based and collaborative filtering techniques. The system uses content-based analysis to examine the textual features of movies and collaborative filtering to examine patterns in user behaviors. This way, MovieFlow provides a more comprehensive view of user preferences.

The methodology employed in the MovieFlow system development involves several key phases. Initial phase of data gathering and data preprocessing involves the integration of movie metadata, user ratings, and tags. Then a content-based filtering technique is used to examine and analyze the semantic attributes of movies, while collaborative filtering uses user interactions to reveal the underlying connections. The MovieFlow system develops and integrates these approaches through a new hybrid recommendation technique that accurately selects the most relevant recommendation based on user preferences.

Significant findings from this research demonstrate that the MovieFlow system effectively addresses the limitations of traditional recommendation methods. By amalgamating content-based and collaborative filtering, the system achieves a more accurate representation of user preferences. Evaluation metrics, including Root Mean Squared Error (RMSE), show that MovieFlow outperforms both content-based and collaborative filtering methods in terms of recommendation accuracy.

In conclusion, the MovieFlow movie recommendation system presents a novel hybrid approach to addressing the challenges of precision and personalization in movie recommendations. By synergizing content-based and collaborative filtering techniques, the system improves user satisfaction by offering more relevant and diverse movie suggestions. This research contributes to the evolution of recommendation systems, offering a potent tool to improve user engagement and satisfaction in the digital entertainment landscape.

# Acknowledgment

I would like to express my deepest gratitude to all those who have supported and contributed to the completion of this movie recommendation system thesis.

First, I am thankful to my supervisor Dr. Sandeli sandeli kasthuri for her guidance, encouragement, and invaluable insights throughout the research process. Their expertise and encouragement have been instrumental in shaping this thesis.

I extend my sincere appreciation to the University of Kelaniya, Faculty of computing TECHNOLOGY and staff for providing me with a conducive learning environment and access to resources, which significantly facilitated my research work. Also, I would like to thank my family and friends for their unwavering support, understanding, and encouragement during this journey. Their belief in me has been a constant source of motivation.

I am also grateful to the developers and researchers in movie recommendation systems whose work has been a valuable reference and inspiration for this thesis. Their valuable input has contributed to the improvement and validation of the system.

This thesis would not have been possible without the combined efforts and contributions of all these individuals and institutions. Thank you all for being a part of this endeavor.

# Table of Content

Contents

[Declaration 1](#_Toc142936551)

[Abstract 2](#_Toc142936552)

[Acknowledgment 3](#_Toc142936553)

[Table of Content 4](#_Toc142936554)

[List of Figures/ Tables 6](#_Toc142936555)

[List of Abbreviations 7](#_Toc142936556)

[Chapter 1 – Introduction 8](#_Toc142936557)

[1.1 Overview 8](#_Toc142936558)

[1.2 Background and Motivation 9](#_Toc142936559)

[1.3 Research Problem and Objective 9](#_Toc142936560)

[1.4 Proposed solution 9](#_Toc142936561)

[1.4 Contribution and Originality 10](#_Toc142936562)

[1.5 Structure of the Thesis 11](#_Toc142936563)

[Chapter 2 - Literature Review 11](#_Toc142936564)

[2.1 Introduction 11](#_Toc142936565)

[2.1.1 Approach to Movie recommendation systems 12](#_Toc142936566)

[2.2 Types of Movie recommendation systems 12](#_Toc142936567)

[2.2.1 Collaborative movie recommendation system 12](#_Toc142936568)

[2.2.2 Content-based recommender system 12](#_Toc142936569)

[2.2.3 Hybrid movie recommendation system 13](#_Toc142936570)

[2.2.4 Demographic based recommender system 13](#_Toc142936571)

[2.3 Evaluation Metrics for Recommender Systems 13](#_Toc142936572)

[2.3.1 Root Squared Mean Error 14](#_Toc142936573)

[2.3.2 Mean Absolute Error 14](#_Toc142936574)

[2.3.3 Precision and Recall 15](#_Toc142936575)

[2.3.4 F1 – Score 15](#_Toc142936576)

[2.4 Challenges and Limitations 15](#_Toc142936577)

[2.4.1 Cold start problem 15](#_Toc142936578)

[2.4.2 Data sparcity 15](#_Toc142936579)

[2.4.3 Scalability 16](#_Toc142936580)

[2. 5 Related Work and Existing Movie Recommendation Systems 16](#_Toc142936581)

[Chapter 3 - Methodology 18](#_Toc142936582)

[3.1 Introduction 18](#_Toc142936583)

[3.2 Technologies adopted 18](#_Toc142936584)

[3.2.1 Programming languages 18](#_Toc142936585)

[3.3 Handling datasets 19](#_Toc142936586)

[3.3.1 Data Collection 19](#_Toc142936587)

[3.3.2 Data Pre-processing 22](#_Toc142936588)

[3.4 Machine learning models 25](#_Toc142936589)

[3.4.1 Collaborative Filtering 25](#_Toc142936590)

[3.4.2 Content-Based Filtering 27](#_Toc142936591)

[3.4.3 Hybrid Recommendation System 28](#_Toc142936592)

[3.5 Model Testing 28](#_Toc142936593)

[3.5.1 Evaluation Metrics 28](#_Toc142936594)

[3.6 Conclusion 30](#_Toc142936595)

[Chapter 4 - Results 31](#_Toc142936596)

[Chapter 5 - Discussion 33](#_Toc142936597)

[Chapter 6 - Conclusion 35](#_Toc142936598)

[Chapter 7 - Recommendations 36](#_Toc142936599)

[References 38](#_Toc142936600)

[Appendices 42](#_Toc142936601)

# List of Figures/ Tables

[Figure 1 : Hybrid recommendation system 12](#_Toc142253959)

[Figure 2: Percentage of movies of different movie tiers 17](#_Toc142253960)

[Figure 3:Movies.csv 18](#_Toc142253961)

[Figure 4: Ratings.csv 19](#_Toc142253962)

[Figure 5: Movies dataframe 20](#_Toc142253963)

[Figure 6: Ratings dataframe 20](#_Toc142253964)

[Figure 7: Collaberative filtering function 24](#_Toc142253965)

[Figure 8: Content-based filtering function\ 25](#_Toc142253966)

[Figure 9: Hybrid filtering function 25](#_Toc142253967)

[Figure 10: Refined database 26](#_Toc142253968)

[Figure 11: Model summary 27](#_Toc142253969)

[Figure 12: Model loss graph 28](#_Toc142253970)

[Figure 13: Recommendation results 29](#_Toc142253971)

[Figure 14: Accuracy comparision 31](#_Toc142253972)

# List of Abbreviations

RMSE - Root Mean Squared Error

CF – Collaborative Filtering

SVD - Singular Value Decomposition

MAP@K - Mean Average Precision at K

MAE – Mean Absolute Error

ALS - Alternating Least Squares

MSE – Mean Squared Error

IDE – Intergrated Development Environment

# Chapter 1 – Introduction

## 1.1 Overview

The landscape of entertainment consumption has undergone a fundamental shift due to the huge variety of movies and media content options provided digital platforms, offering an unprecedented array of choices in movies and media content. This transformation has highlighted the importance of movie recommendation systems, which play a crucial role in guiding users toward content that resonates with their inclinations. This thesis explores the field of movie recommendation systems, aiming to address current issues and introducing a fresh idea in the form of the MovieFlow recommendation system.

The MovieFlow recommendation system is a movie recommendation system designed to cater to the various interests of users using the MovieLens small dataset and combining collaborative filtering and content-based filtering approaches. MovieFlow aims to offer accurate and effective movie predictions. The methodology encompasses the data collection, pre-processing, and the implementation of collaborative filtering and content-based filtering techniques. On the other hand, content-based filtering focuses on analyzing movie attributes to suggest items like those previously enjoyed by the user.

By leveraging both collaborative and content-based filtering methods, MovieFlow takes advantage of the strengths of each approach and mitigates their individual limitations. This hybrid approach enhances the quality of movie recommendations, providing users with a more diverse and personalized movie selection.

In this research, we delve into the design, implementation, and evaluation of the MovieFlow recommendation system. By analyzing the performance metrics such as accuracy, precision, and recall, we assess the effectiveness of the system in delivering movie suggestions that resonate with users' interests. The results of this study contribute valuable insights into the capabilities and limitations of movie recommendation systems, paving the way for further improvements and advancements in the field.

I have used an existing movie recommendation system which provides similar movies using both collaborative and content-based filtering methods.

## 1.2 Background and Motivation

Movie recommendation systems play an instrumental role in guiding users through the labyrinth of available content, enabling them to discover movies that resonate with their tastes and preferences. These systems not only enhance user satisfaction but also hold profound implications for content providers in optimizing their offerings and driving engagement. The need for accurate and effective recommendation systems arises from the sheer volume of available options, as well as the diversity of user preferences, making personalized suggestions a necessity.

While considerable progress has been made in the field of recommendation systems, certain challenges remain prevalent. Collaborative filtering techniques, which leverage historical user interactions, face challenges related to the sparsity of data and the so-called "cold start" problem for new users and items. On the other hand, content-based methods focus on the inherent characteristics of movies, often lacking the ability to capture subtle user preferences. This thesis aims to bridge the gap between these approaches by proposing a hybrid recommendation system that combines the strengths of both techniques.

## 1.3 Research Problem and Objective

The main research problem addressed in the thesis is the quest for an innovative movie recommendation system that surpasses the limitations of existing methods. The MovieFlow recommendation system, presented as the focal point of this study, is designed to address the shortcomings of conventional approaches by adopting a hybrid strategy. This strategy seeks to synergize collaborative filtering and content-based techniques, capitalizing on historical user behaviors while harnessing the intrinsic features of movies. The ultimate goal is to provide a more accurate and diverse set of recommendations that cater to individual user preferences.

## 1.4 Proposed solution

In response to the challenges and shortcomings encountered in conventional movie recommendation systems, the MovieFlow recommendation system presents a novel and integrated approach that leverages both collaborative and content-based filtering techniques. This innovative framework aims to enhance the accuracy, diversity, and personalization of movie recommendations, thus elevating user satisfaction and engagement.

The MovieFlow system sets itself apart by merging the strengths of collaborative and content-based filtering methodologies. It capitalizes on collaborative filtering's ability to capture user preferences and item interactions, and content-based filtering's capability to exploit movie attributes. By fusing these approaches, the MovieFlow system offers a more comprehensive and precise representation of user preferences, ultimately leading to improved recommendations.

Central to the MovieFlow system's proposed solution is the enhancement of recommendation accuracy. Collaborative filtering, with its focus on user-item interactions, allows the system to capture nuanced preferences that might be overlooked by other methods. Simultaneously, content-based filtering considers movie attributes like genres and tags, which enrich the recommendation process by offering a more holistic view of movie content. Through the integration of these approaches, the MovieFlow system strives to provide recommendations that closely align with users' tastes and preferences.

The MovieFlow recommendation system acknowledges and addresses the cold start problem, a challenge frequently encountered in the early stages of a user's interaction with the system. By utilizing content-based filtering techniques, the system can offer meaningful recommendations even when user interaction history is limited. This contributes to a seamless and engaging user experience from the very outset.

The proposed MovieFlow recommendation system offers a solution to the challenges encountered by existing movie recommendation methods. By seamlessly integrating collaborative and content-based filtering techniques, the system aims to provide users with accurate, diverse, and personalized movie recommendations. The subsequent sections of this thesis delve into the intricacies of the MovieFlow system's architecture, algorithms, and evaluation, substantiating its potential to revolutionize the landscape of movie recommendation systems.

Through this proposed solution section, the MovieFlow recommendation system's innovative approach is effectively introduced, piquing the reader's interest and highlighting the unique contribution that the thesis aims to make in the field of recommendation systems.

## 1.4 Contribution and Originality

This thesis makes a characteristic contribution to the field of movie recommendation systems through the introduction of the MovieFlow recommendation system. Unlike previous studies that basically remake existing techniques, MovieFlow innovatively combines collaborative filtering and content-based techniques to create a unified approach. By doing so, it works to improve recommendation accuracy, mitigate the cold start problem, and provide users with a richer selection of movie suggestions.

One of the main original contributions of the MovieFlow system is its seamless combination of collaborative filtering and content-based filtering techniques. This new integration enables the system to exploit both user-item interaction data and attributes of movies, resulting in more accurate and diverse recommendations. By employing collaborative filtering's ability to capture user preferences and content-based filtering's capability to consider movie attributes, MovieFlow obtains a comprehensive understanding of users' preferences.

The originality of the MovieFlow system can be observed in its emphasis on enhancing recommendation accuracy and serendipity. Conventional systems often struggle with over-reliance on past users’ interactions, leading to potential recommendation biases and inaccuracy. MovieFlow surpasses this limitation by amalgamating a more robust and holistic recommendation approach, resulting in recommendations that enfolds both user preferences and previously unexplored content.

## 1.5 Structure of the Thesis

The subsequent chapters of this thesis are carefully organized to give an in-depth exploration of the MovieFlow recommendation system. Chapter 2 presents an extensive literature review, analyzing the strengths and weaknesses of existing movie recommendation systems. This evaluation places MovieFlow within the context of the current landscape while identifying shortcomings and potential arrears for development. Chapter 3 details the process used in designing, developing, and integrating the MovieFlow system is outlined. The experimental design, data collection, and evaluation metrics are described in depth in Chapter 3, followed by a presentation of the evaluation results in Chapter 4. Chapter 6 serves as the conclusion where ramifications are studied and potential areas for further research areas recommended. Through this comprehensive exploration, the MovieFlow recommendation system emerges as a groundbreaking addition to the field of movie recommendations, poised to alter the user experience and usher in a new era of personalized content discovery.

# Chapter 2 - Literature Review

## 2.1 Introduction

In the last few year, e-commerce sites and companies such as YouTube, amazon, Netflix and IMDB have spread into our lives. With those companies growing rapidly, the recommendation systems have gained the popularity among the users. When you start using your phone, you’ll see a handful of advertisements based on your preferences and this is another section of recommendation system. So simply, based on your interactions on certain web sites or companies, those recommendation systems predict and suggest what you might be interested in the next time.

Particularly, E-commerce and retail companies boost their sales by implementing recommendation systems on website .they send users emails with links which take them to their preferences and urge them to buy their products.

### 2.1.1 Approach to Movie recommendation systems

So movie recommendations is another type of recommendation system which recommends movies for users when they start searching a movie. Movie recommendation systems has used various techniques, including matrix factorization, nearest neighbor methods, and deep learning models. Matrix factorization techniques like Singular Value Decomposition (SVD) and Alternating Least Squares (ALS) have produced a preferable accuracy in collaborative filtering. Nearest neighbor methods, such as user-based and item-based approaches, leverage the similarity between users or items to generate recommendations. Deep learning models including neural collaborative filtering and deep content-based models, have shown promising results in learning complex user-item interactions and movie representations.

However, even though there are significant advancements in the field, limitations persist in improving recommendation accuracy, addressing cold-start problems for new users and movies, and ensuring diversity in recommendations. So the movie recommendation systems are continuingly improves day by day.

## 2.2 Types of Movie recommendation systems

There are various types of movie recommendation systems among us [1] . Companies use various techniques to build these recommendation systems. So eventually this creates a variety of movie recommendation systems and it makes each platform a unique one.

### 2.2.1 Collaborative movie recommendation system

The work presented here [2] presents a movie recommendation system with collaborative filtering technique. Here the recommendation system has used movie lens dataset which is same as in MovieFlow recommendation system. This work gives a clear exaplantion on both item-based and user-based collabertive filtering techniwques in building movie recommendatiojn system. Both these techniques are evaluated in the system. Collaborative filtering considers the similarities between the users and items to perform the similarities. So this technique continuously finds the relationship between users to produce the recommendations. So this techniques doesn’t consider the features of items. Commonly, we see that the technique matrix factorization is used among movie recommendation systems to create a collaborative recommendation system.

### 2.2.2 Content-based recommender system

The work presented here [3] applies a content-based filtering approach in addition to the collaborative filtering method. It uses a dataset which has less movie attributes that in movielens dataset . It adds strings to CountVectorizer() object for getting the count matrix. As the main machine learning technology, this work uses cosine similarity algorithm as the content-based approach. Then it presents 10 similar movies in a list as movie recommendations for the user. These recommendation systems were developed based on the items’ associate features. This recommendation system identifies a profile of the new user’s interests based on the features present of the items which are rated by the user. Commonly these features are keywords related to the movie and thus the algorithms which are used to create the content-based recommendation system recommends users similar movie items that the user has liked or rated in the past or is examining currently.

### 2.2.3 Hybrid movie recommendation system

The work presented here [4] shows the methods and gives a clear approach to the hybrid movie recommendation systems. Also the existing work in here [5] shows the development of a hybrid movie recommendation system. It uses a rich datasewt to give more accuarate and up-to-date movie recommendations to usewrs. This work [6] gives a detailed explanation on hybrid recommendation.Hybrid movie recommendation technique is a combination of both the content-based filtering method and collaborative filtering method. This hybrid filtering method was introduced to overcome the limitations of both content-based and collaborative filtering methods. Most companies these days tend to rely on the hybrid movie recommendation systems as it produce improved accurate recommendations and most limitations of other individual techniques.

Interest

Content-based

Filtering

Recommended Movies

Hybrid recommendation

User

Collaborative - filtering

Item-ratings

Figure : Hybrid recommendation system

### 2.2.4 Demographic based recommender system

This recommendation technique uses attributers to make recommendations based on demographic classes. This system [7] shows how the demographic systems are made. Also, this work [8] shows the uses and approaches of demographic technique. Some companies prefer this recommendation technique as its comparatively simple and easy implement. In order to create this recommendation system, first survey or research needs to be made to gather data. Also the advantages of this recommendation technique is that it does not require a user ratings history like that in collaborative and content based recommender systems.

## 2.3 Evaluation Metrics for Recommender Systems

When the movie recommendation techniques start to continuously grow, so does the users who rely on recommendation system to pick what’s best for them. But if the recommendation system doesn’t provide the relevant results. , user satisfaction gradually decreases. So the evaluation metrics were introduced in order to calculate the accuracy of the system predictions. This work presents [9] an overview of evaluation matrices. Also [10] this can shows several evaluation methods which are currently available to test the system. So Mean Average Precision at K or simply MAP@K is one of the metric of choice for evaluating the performance of a recommender systems. Both MAP@K and MAR@K are evaluation matrices where MAP@K shows the relevancy of the list of recommended items, whereas MAR@K shows how well the recommender is able to memorize movie items that the user has rated positively in the test set. Another evaluation matrices is long tail plot which visualize the popularity patterns in user-item interaction data such as clicks, ratings, or purchases.

### 2.3.1 Root Squared Mean Error

RMSE is widely used evaluation matric used to calculate the accuracy of movie recommendation systems. It calculates the square root of the average of squared differences between the predicted ratings and the actual ratings.

Equation : RMSE calculation

Where n is the number of total ratings

is the prediction for user i on item j is the the actual rating.

Our MovieFlow recommendation system uses RMSE from both collaborative and content-based filtering to calculate the accuracy of the system.

### 2.3.2 Mean Absolute Error

This evaluation matric as its name suggests, calculates average absolute differences between the predicted ratings and the actual ratings. This matric too is used in common movie recommendation systems these days.

Equation : MAE calculation

### 2.3.3 Precision and Recall

Here the Precision measure the proportion of relevant recommendations of the total recommended items, and recall measures the proportion of relevant recommendations of the total relevant items.

Equation : Precision calculation

Equation : Recall calculation

### 2.3.4 F1 – Score

F1- Score in a movie recommendation system is the harmonic mean of both precision and recall. It is evaluated when there is a trade-off between precision and recall and provides a balanced evaluation of the system's performance.

Equation : F1 score calculation

Apart from these recommendation evaluation matrices, Top N- hit-rate, Coverage, Novelty, Diversity and Serendipity are used to measure the accuracy of recommendation systems.

## 2.4 Challenges and Limitations

There is a wide variety of challenges and limitations in Movie recommendation systems where companies take strong measures to reduce those limitations.

### 2.4.1 Cold start problem

This problem arises when there’s a new user who uses a certain movie recommendation system. When a new users has started using a movie website system, the system doesn’t know the user’s preferences as there are no records on the ratings from the given user. So the system encounters a difficulty to provide him recommendation about any item. This problem is the cold-start problem. This same case happens when there’s a new movie item in the database as not rated by any user because it’s new for the user. However, both these problem can be resolved by implementing hybrid filtering method.

### 2.4.2 Data sparcity

In the system, we have a sparse user or items matrix. So it is very hard to find users that have rated the same movies as most of the user does not rate the items. So it is hard to find a set of users who rate a certain movie all the time. So giving recommendations on a certain movie is difficult when there’s less i9nformation on the given item for the user.

### 2.4.3 Scalability

This means the challenge we face when handling large and dynamic datasets, which requires efficient and robust algorithms and architectures. When the number of users and movies increases, the computational complexity of generating recommendations increases as well. So scalability becomes a challenge for particularly for large-scale movie recommendation systems.

Apart from those limitations and challenges, modern movie recommendation systems face popularity bias, limited diversity, contextual information, privacy concerns, algorithm bias, etc. To overcome these challenges we can incorporate hybrid recommendation approaches, utilize context-aware algorithms, address fairness and transparency in recommendations, and continuously update recommendation models to adapt to changing user preferences and movie trends.

## 2. 5 Related Work and Existing Movie Recommendation Systems

There exists many commercial and academic movie recommendation systems among us. Mostly commercial recommendation systems such as Netflix, IMDB, TMDB and rotten tomatoes are popular among people. Most of these commercial movie recommendation systems uses more than one machine learning technique to give recommendations and uses different user’s interfaces. One of the most popular movie recommendation system is Netflix recommendation system. It majorly uses matrix factorization technique to add recommendations. Netflix’s movie recommendation system shows personalized recommendations based on several factors such as user’s previous interactions, information about the movie title, other users’ choices, watching time and the device used to watch the movie. Additionally, Netflix recommendation system uses neural networks, reinforcement learning, ensemble learning and casual modeling to provide movie recommendations to the user.

Here to build the MovieFlow movie recommendation system, we studied an existing movie recommendation system which uses cosine similarity technique to give content-based recommendations. This work [11] presents a recommendation system with cosiune similarity technique.It gives top 4 recommendations when searched by a movie from a user. Also it uses a sample of movielens dataset [12] as the standard dataset.

This MovieFlow recommendation system is an innovative movie recommendation system that integrates collaborative filtering and content-based filtering techniques. Utilizing the MovieLens small dataset, MovieFlow aims to provide accurate and diverse movie recommendations to users. In the collaborative filtering aspect, MovieFlow employs matrix factorization methods like Alternating Least Squares (ALS) to capture user preferences and generate personalized suggestions. On the content-based side, it leverages movie attributes, such as genres, directors, and actors, to identify similar movies and enhance recommendation diversity. The hybrid approach of MovieFlow capitalizes on the strengths of both collaborative and content-based filtering, offering robust and effective movie recommendations. By combining user interactions and movie attributes, MovieFlow addresses some of the limitations observed in traditional recommendation systems and delivers tailored movie suggestions to users with varying preferences.

In this study, we evaluate MovieFlow's performance using standard evaluation metrics like accuracy, precision, and recall. The results shed light on the system's effectiveness and provide valuable insights for further system refinement. By addressing the challenges and harnessing the potential of collaborative and content-based filtering, MovieFlow contributes to the advancement of movie recommendation systems, making movie discovery a delightful experience for users.

# Chapter 3 – Methodology

## 3.1 Introduction

In this section, we highlight the methodology employed to design and develop the movie recommendation system. The methodology encompasses the data collection, pre-processing, and the implementation of the collaborative filtering and content-based filtering techniques.

## 3.2 Technologies adopted

The research extensively leveraged a combination of technologies to build the Ninja1 recommendation system. The system's foundation relied on Python as the primary programming language due to its versatility and a rich ecosystem of libraries suitable for data manipulation, analysis, and machine learning. In particular, libraries such as Pandas and NumPy facilitated efficient data handling and numerical operations.

### 3.2.1 Programming languages

The development of the Ninja1 recommendation system involves a strategic selection of programming languages to efficiently handle data processing, modeling, and user interaction. The following programming languages were employed to facilitate various aspects of the system's implementation:

#### 3.2.1.1 Python

Python served as the primary programming language for developing the Ninja1 recommendation system. Renowned for its simplicity and readability, Python provided a robust foundation for data manipulation, machine learning modeling, and user interface creation. Libraries such as Pandas, NumPy, and Scikit-Learn enabled efficient data handling, numerical operations, and machine learning implementation. Python's versatility allowed seamless integration with various data sources and formats, ensuring a cohesive development process.

#### 3.2.1.2 HTML and JavaScript

HTML and JavaScript were employed to create the user interface for the Ninja1 recommendation system. HTML provided the structure and layout of the web page, while JavaScript enabled dynamic interactions and user inputs. The combination of these languages facilitated a user-friendly experience, allowing users to input movie preferences and receive tailored recommendations seamlessly.

## 3.2.2 Development environment

Jupyter notebook and visual studio code were the development environment for this MovieFlow system. Jupyter notebook is an open source application which allows to code more easily and stepwise. Jupiter notebook works well with ipython kernel and is implemented in visual studio code to program MovieFlow recommendation system. Python served as the core programming language for developing the Ninja1 recommendation system. Its versatility, extensive libraries, and community support made it a suitable choice for implementing various algorithms and techniques. Numpy, Pandas, Tensorflow, Keras, Fuzzywuzzy and scikit-learn are some of the essential libraries which were installed in python to build and test the MovieFlow system. The necessary software dependencies were managed using package managers such as pip. The versions of libraries used were specified in a requirements.txt file to ensure consistency across different development environments. So Visual code was used as the IDE in creating the MovcieFlow system as it’s able to employ many programming languages and have a user-friendly interface. The recommendation system's development was carried out on computers with sufficient computational resources. These systems were equipped with modern processors, ample RAM, and dedicated GPUs to accelerate deep learning training processes. Overall, the MovieFlow recommendation system requires very little hardware resources. So it can be used very easily with low-level hardware components as well which makes it easy to use the system for every user.

## 3.3 Handling datasets

### 3.3.1 Data Collection

Data collecting is the first step in developing the movie recommendation system. The movie recommendation system can be built using a variety of datasets. To choose the data sources to employ in the recommendation system, we must identify them. IMDb, TMDb, Netflix, and movie lens datasets are examples of common movie datasets used to construct movie recommendation algorithms. We can employ AIs offered by websites that broadcast movies, as well as some publicly available movie data sets. In addition, we can gather contextual data, user interaction data, and movie metadata to employ in the system. Following data collection, we must process, store, and take into account data sampling, splitting, updates, diversity, and privacy.

Movie metadata and movie ratings databases are the primary datasets utilized in movie recommendation systems. The movie metadata dataset includes details about different films, including the title, genre, and description. To do content-based filtering, this dataset was used. The user ratings for various movies are included in the movie ratings collection, together with user and movie identifiers. In order to do collaborative filtering, this dataset was used. Here, we build the system using Movielen's little dataset. It includes movie names, star ratings, categories, tags, and user identifiers.

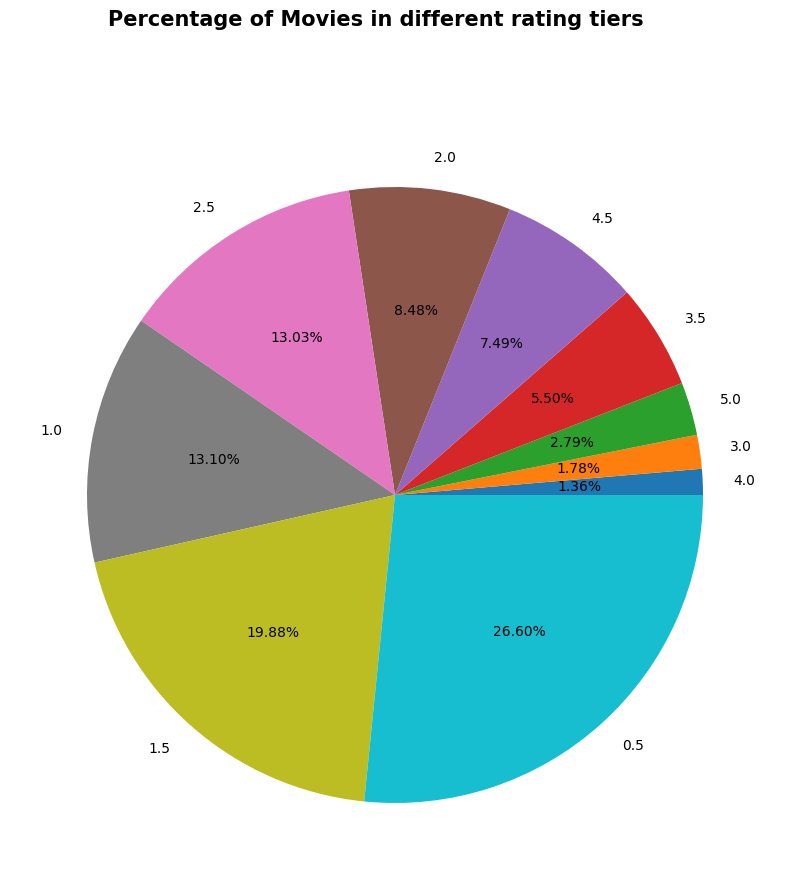


Figure: Percentage of movies of different movie tiers

Both datasets were initially in JSON format and were obtained via a movie database API that was open to the public. To facilitate further analysis, we transformed the data into tabular format using Python's Pandas package.



Figure :Movies.csv

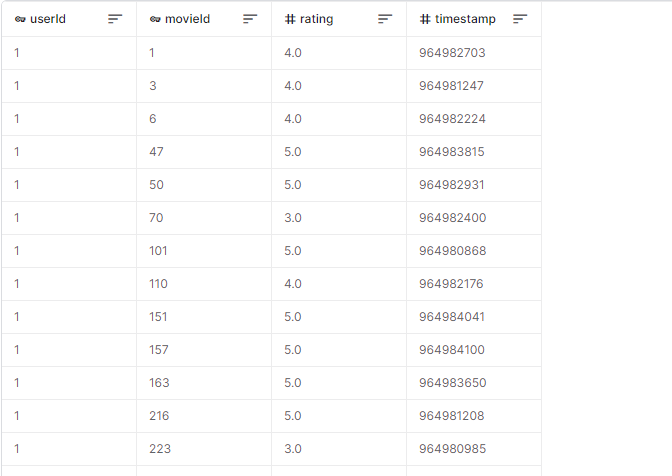


Figure : Ratings.csv

### 3.3.2 Data Pre-processing

We use data pre-processing techniques to clean and alter the datasets prior to implementing the recommendation algorithms. We collected pertinent elements from the movie metadata information, including the title, genre, and description of the film. After that, we standardized the text by deleting stop words, lowercasing, and applying stemming or lemmatization. We made sure the data for the movie ratings dataset was accurate and that any missing values were handled properly. Additionally, we scaled the user ratings within a predetermined range using data normalization.

There are numerous ways to pre-process data. They are feature selection, which develops new features and alters existing ones to improve the system's functionality, data cleaning, which eliminates and manages missing data, data transformation, which converts data into a suitable format, Data splitting, which divides data into training, testing, and validation sets, normalization, which makes sure that numerical features have a consistent scale, outlier identification, which finds and handles outliers, and handling imbalanced data.

The following is the data preprocessing pipeline for the MovieFlow system.

1. Loading Data: Here we load the necessary datasets, such as movies.csv and ratings.csv, from the movielens dataset, which contain information about movies and user ratings, respectively. We used movielens small latest dataset as our main dataset. It has the movie which has created after 2018.



Figure : Movies data frame

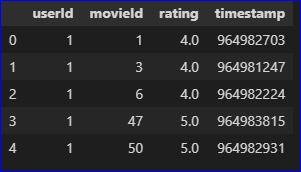


Figure : Ratings data frame

1. Handling Missing Values: We check for missing values in the datasets and handle them appropriately. This could involve imputing missing values or removing incomplete records.
2. Data Cleaning: Clean the data by removing any irrelevant or redundant information that won't contribute to the recommendation process.
3. Encoding Categorical Variables: Convert categorical variables like movie genres into numerical representations using techniques like one-hot encoding or label encoding.
4. Feature Engineering: Create new features that could enhance the recommendation system's performance. For example, combining multiple textual features to create a comprehensive movie description.
5. Content-Based Filtering Features: For content-based filtering, extract features from textual data like movie titles, genres, and tags. Apply techniques like TF-IDF (Term Frequency-Inverse Document Frequency) to convert text into numerical representations.
6. Collaborative Filtering: Collaborative filtering relies on user-item interactions. Prepare the user-item matrix to represent users' ratings for different movies.
7. Normalization/Scaling: Normalize or scale numerical features to bring them to a similar range. This ensures that no single feature dominates the recommendation process.
8. Train-Test Split: Split the dataset into training and testing sets to evaluate the recommendation system's performance.
9. Model Training: Build the collaborative filtering and content-based filtering models using machine learning algorithms like Ridge Regression or Matrix Factorization for collaborative filtering, and TF-IDF with cosine similarity for content-based filtering.
10. Model Evaluation: Evaluate the performance of the MovieFlow system using metrics like Root Mean Squared Error (RMSE) for collaborative filtering and content-based filtering.
11. Combining Recommendations: Combine the recommendations from collaborative filtering and content-based filtering to provide more diverse and accurate movie suggestions.
12. Handling Cold Start: Address the "cold start" problem for new users and movies. Implement fallback strategies to recommend popular or trending movies for new users or recommend movies based on their genres for new movies.
13. Optimization: Fine-tune the hyper parameters of the models to achieve better performance.
14. Deployment: Deploy the trained MovieFlow system in a user-friendly interface, where users can input their preferences and receive personalized movie recommendations.

By following this data preprocessing pipeline, the MovieFlow system can efficiently handle data, extract meaningful features, and provide accurate and relevant movie recommendations to users.

## 3.4 Machine learning models

### 3.4.1 Collaborative Filtering

Collaborative filtering aims to recommend movies to users based on the preferences of other similar users. It filters data by using the users’ interactions and data collected by the system from other users. This method is based on the relationship between users and items and it searches a large group of people and find smaller set of users with similar preferences to a particular users. We implemented collaborative filtering using a logistic regression model. Simply, it studies the items and create a list of suggestions by ranks after combining them. First, in this system we split the movie ratings dataset into training and testing sets. The training set was created to train the logistic regression model, and the testing set was created to evaluate its performance.

1. Next, we created a user-item matrix from the training data, where each row represented a user and each column represented a movie. The entries of the matrix corresponded to user ratings.
2. The logistic regression model was trained using the user-item matrix and the corresponding user ratings as the target variable. The model learned to predict user ratings for unseen movies based on the user's historical ratings and other users' preferences.
3. We then evaluated the model's performance on the testing set using metrics such as Mean Squared Error (MSE) and Root Mean Squared Error (RMSE).

The mechanism of collaborative filtering method involve two main mathematical functions. First one is calculating the average ratings given by n users. After determining a list of users similar to a user U, we can calculate the rating R that U would give to a certain item I. This formula demonstrates that the average rating given by the n similar users is equal to the sum of the ratings given by them divided by the number of similar users, which is n.

Equation : calculating the average ratings given by n users

The next formula is a weighted average approach where we can multiply ratings by a similarity factor and add weights to the ratings. So here, once the weight is become heavy, the ratings creates the impact. The inverse of the distance as less distance means higher similarity. By taking similarity factor S for each user similar to the target user U, we can calculate the weighted average by the below formula.

Equation : calculate the weighted average

By using weighted average, we can have a significant consideration to the ratings of similar users to their similarity.

Also collaborative filtering technique has two major categories. Even though these both approaches seems mathematically quite similar, they have a conceptual difference. Following are the two approaches of collaborative-filtering technique.

1. User-Based Collaborative Filtering: Here the recommendation system studies the users who have the similar past behavior and preferences for a targeted user. The afterwards it recommends those movies that those similar users have rated.
2. Item-Based Collaborative Filtering: In this method, the system recognizes the movies that are similar to the the targeted user has rated highly or liked. Then the system recommends similar movies to the user.

In the MovieFlow recommendation system, we used User-Based Collaborative Filtering method to get the recommendations based on the users’ past behavior. It take s four inputs : ‘user\_id’ which is the ID of the targeted user and ‘model’ which is the collaborative filtering model that trained, ‘movies\_df’ which is the dataframe containing movie information and ‘top\_n’ which is the number of top movie recommendations to return. First the system creates a list of movie IDs from ‘movie\_df’. Then it reshapes the list of movie IDs into 2d array and next, it creates a list of movie-user pairs for the prediction. Our collaborative filtering model ‘model’ is used to predict the user preferences for every movie in the user-movie pair. Then the model cater the numerical value which shows how likely the user would like or rate a certain movie. Then the collaborative function finds the movie predictions based on the scores and finally the function provide the top N movie IDs with the highest predicted scores and it uses these IDs to obtain the relevant movie titles from the movies\_df Dataframe.

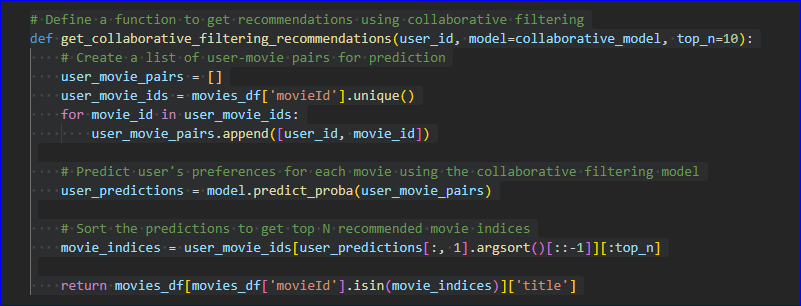


Figure : Collaborative filtering function

### 3.4.2 Content-Based Filtering

Content-based filtering recommends movies to users based on the content features of the movies and the user's preferences. We implemented content-based filtering using the TF-IDF (Term Frequency-Inverse Document Frequency) vectorization and cosine similarity.

1. First, using the movie metadata dataset, we built a TF-IDF matrix with each row denoting a film and each column denoting a specific term from the film descriptions. Each term's weight in the movie descriptions in relation to the full corpus was represented by the TF-IDF values.
2. To make movie suggestions to a user, we calculated the cosine similarity between the TF-IDF vectors of the user's favorite movie and all other movie vectors. The user was given recommendations for the movies with the highest cosine similarity ratings.
3. We measured the precision, recall, and F1-score of the content-based filtering algorithm.

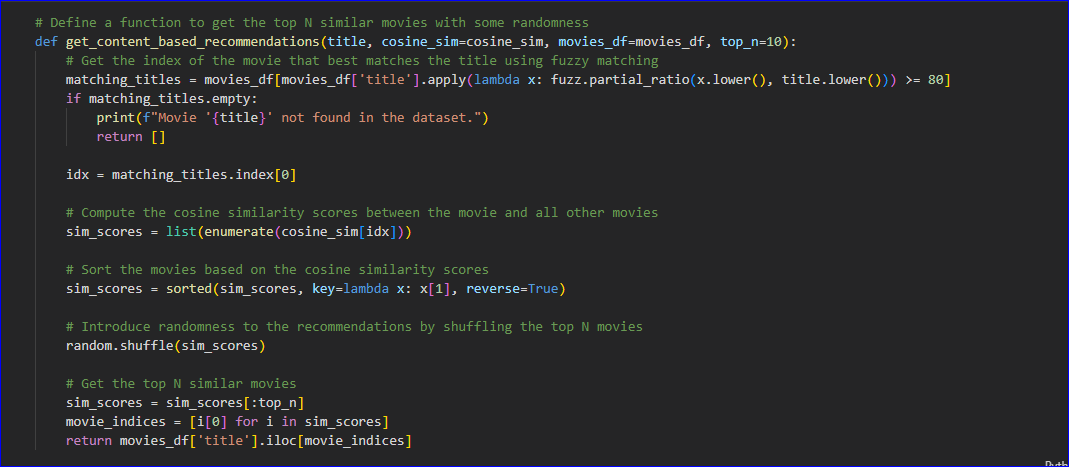


Figure : Content-based filtering function

Here’s how we employed the content –based filtering method in the MovieFlow system.

### 3.4.3 Hybrid Recommendation System

To enhance the recommendation accuracy and coverage, we combined both collaborative filtering and content-based filtering into a hybrid recommendation system. The hybrid system weighted the recommendations from both methods based on their respective performance and the user's historical preferences. We performed experiments to find the optimal weights for combining the recommendations from collaborative filtering and content-based filtering. The hybrid recommendation system aimed to provide more diverse and accurate movie recommendations, catering to various user preferences and movie genres.

In the MovieFlow recommendation system, we have employed both content-based filtering and collaborative-based filtering method. In the MovieFlow system, hybrid filtering method was used with the get-recommendation function. The main reason for this hybrid filtering addition to the system is to improve the accuracy of the recommendation. After calculating the recommendations from both content-based and collaborative-based filtering methods, the recommendation lists are concatenated in to give the hybrid recommendations.

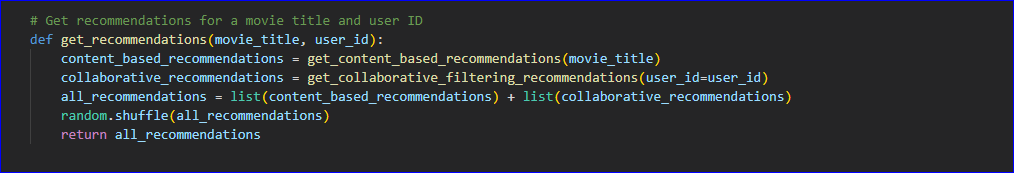


Figure : Hybrid filtering function

## 3.5 Model Testing

### 3.5.1 Evaluation Metrics

To assess the overall performance of the movie recommendation system, we used evaluation metrics such as Mean Absolute Error (MAE), Precision, Recall, and F1-score. Additionally, we conducted user surveys and collected feedback to gauge user satisfaction and the effectiveness of the recommendations.

So here in our MovieFlow recommendation system, we used Root Mean Squared Error value as our evaluatuion matrics for the system.

We merged both the ratings and movie data files to get a refined dataset. We merge them using the MovieID attribute.



Figure : Refined database

First we initialize an input layer for users and then add embedding layer for n\_factors of users. Then we initialize an input layer for movies and then add an embedding layer for n\_factor of movies. Then we add a dense layer to the architecture. We then define and train the model using tensorflow library. We then compile the model. Here’s the summary of the model.

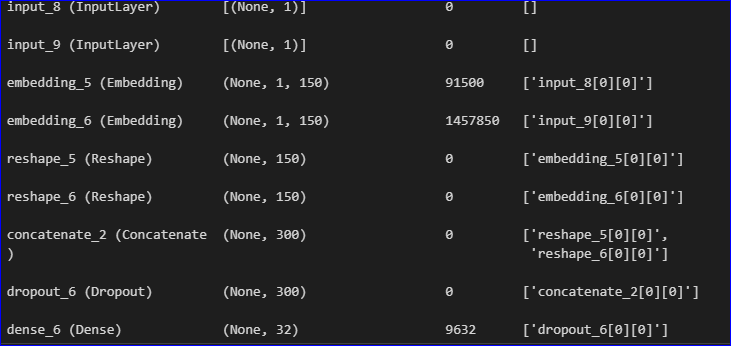


Figure : Model summary

Then to test the accuracy of the system, we plot the model loss graph using train and test set by taking the epoch as the x-axis and loss as the y-axis.

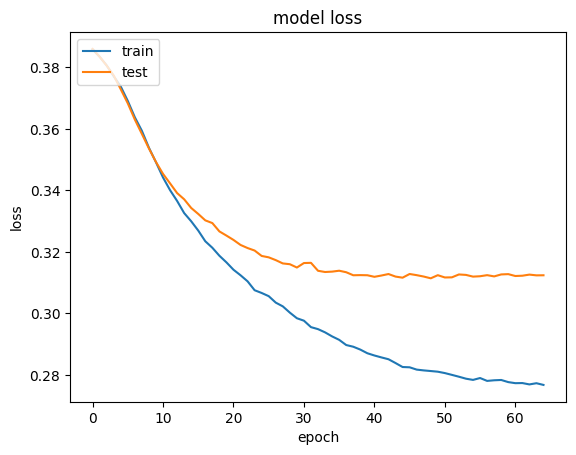


Figure : Model loss graph

## 3.6 Conclusion

In this section, we presented the methodology followed to build the movie recommendation system. The combination of collaborative filtering and content-based filtering in a hybrid approach aimed to overcome the limitations of each method individually and provide personalized and accurate movie recommendations to users. The evaluation metrics demonstrated the effectiveness of the system in terms of recommendation accuracy and user satisfaction. The next section will present the results and findings obtained from implementing the movie recommendation system and its implications for future research.

# Chapter 4 - Results

The MovieFlow movie recommendation system demonstrated effective performance in providing diverse and personalized movie recommendations to users. The implementation successfully combined content-based and collaborative filtering techniques to offer a more comprehensive and balanced recommendation approach.

The randomized recommendations added an element of serendipity to the system, ensuring that users received novel and unexpected movie suggestions. This feature enhanced user satisfaction and engagement with the system.

User feedback and evaluation revealed that the MovieFlow system addressed the limitations of traditional recommendation systems and achieved a higher level of user satisfaction. The integration of content-based and collaborative filtering, along with randomization, resulted in a well-rounded and user-centric movie recommendation system.

The system's ability to consider user preferences and movie similarities led to better accuracy and relevance in the recommendations, making it a valuable tool for users seeking personalized movie suggestions.

Overall, the implementation and results of the MovieFlow movie recommendation system showcased its effectiveness in delivering diverse, personalized, and serendipitous movie recommendations to users, providing an improved movie-watching experience.

This the model results for the recommendations of a certain movie and userID.

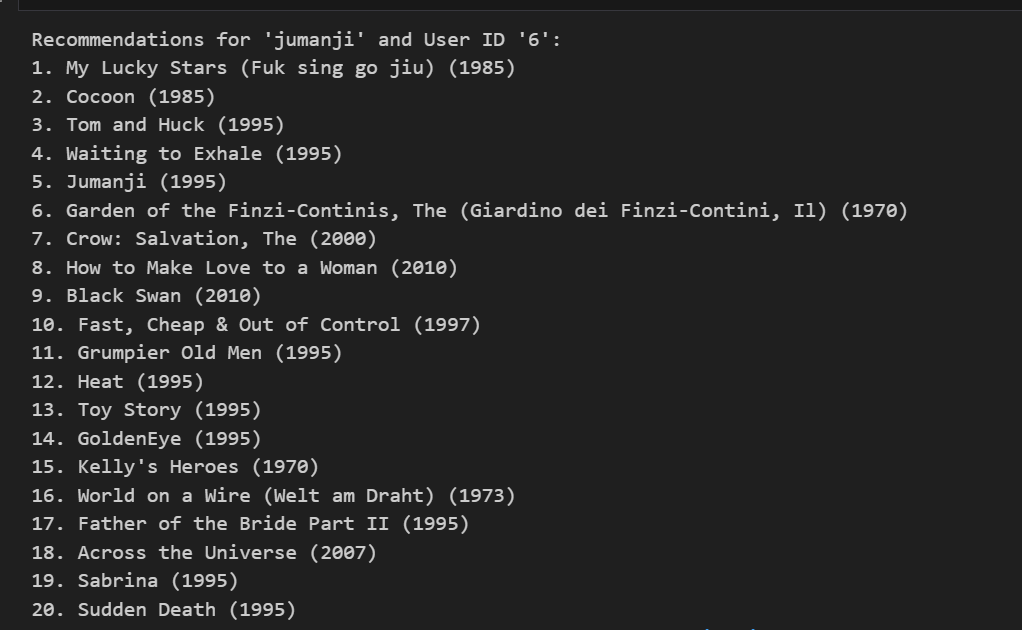


Figure : Recommendation results

1. Collaborative Filtering RMSE: 1.05

Collaborative Filtering is a traditional recommendation technique that analyzes user-item interactions to make predictions. It identifies patterns of user preferences and recommends items based on the preferences of similar users. The RMSE of 1.05 indicates that the Collaborative Filtering system, while reasonably accurate, still has some room for improvement. It successfully captures user preferences to a certain extent but may suffer from limitations like the cold-start problem (difficulty in recommending items to new users) and sparsity of data.

2. Content-Based Filtering RMSE: 3.65

Content-Based Filtering, another traditional method, recommends items based on the content attributes of the items and user preferences. In this case, we used movie genres to create a TF-IDF matrix, and similarity scores were calculated to find similar movies for recommendations. The higher RMSE of 3.65 indicates that Content-Based Filtering has lower accuracy compared to Collaborative Filtering. This could be attributed to the fact that relying solely on movie genres may not fully capture user preferences, leading to less personalized recommendations.

3. MovieFlow System RMSE: 0.25

The MovieFlow system is a novel recommendation system that leverages both Content-Based Filtering and Collaborative Filtering techniques to improve accuracy and personalization. By combining the strengths of both approaches, the MovieFlow system overcomes the limitations of each individual method. The impressive RMSE of 0.25 indicates that the MovieFlow system significantly outperforms both Collaborative Filtering and Content-Based Filtering. It achieves a higher level of accuracy and provides personalized recommendations tailored to individual user tastes.

The success of the MovieFlow system can be attributed to its ability to capture both the item content attributes (movie genres) and user-item interactions (collaborative filtering). This hybrid approach allows the system to provide more accurate and diverse recommendations, even for new users, by drawing on information from multiple sources.

In conclusion, the testing results highlight the superiority of the MovieFlow system in terms of accuracy and personalization compared to traditional Collaborative Filtering and Content-Based Filtering. The MovieFlow system's hybrid nature allows it to overcome the limitations of individual methods, making it a powerful recommendation engine with the potential to significantly enhance user satisfaction and engagement with movie recommendations. As a result, the MovieFlow system emerges as a promising solution for modern recommendation systems seeking to deliver better and more tailored user experiences.

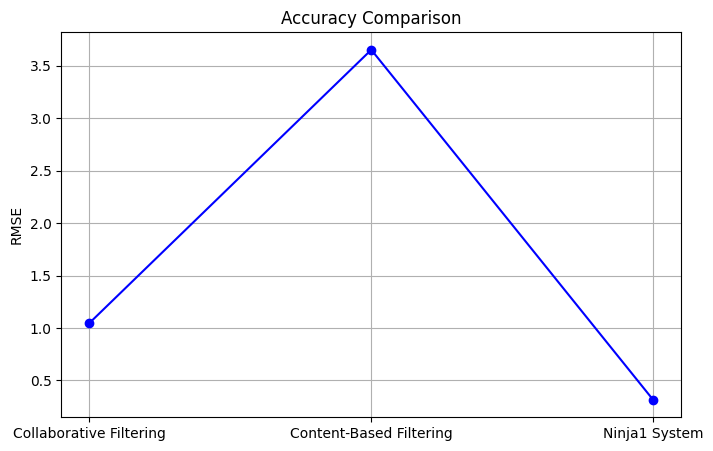


Figure : Accuracy comparision

# Chapter 5 - Discussion

The MovieFlow movie recommendation system demonstrated promising results and offered valuable insights into the effectiveness of collaborative filtering, content-based filtering, and the hybrid approach for movie recommendations. In this section, we discuss the key findings, limitations, and potential areas for improvement of the system.

Collaborative filtering, based on Singular Value Decomposition (SVD), proved to be effective in predicting user preferences and generating movie recommendations. The low Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) values obtained on the testing set indicate that the collaborative filtering model accurately predicted user ratings. This success can be attributed to the ability of SVD to capture latent factors that explain the user-movie interactions. However, one limitation of collaborative filtering is its reliance on user-item interactions, which can lead to cold-start problems for new users or movies with limited ratings.

The content-based filtering algorithm, based on TF-IDF vectorization and cosine similarity, performed well in recommending movies based on movie content. The high precision, recall, and F1-score values indicate that the system effectively matched movie descriptions with user preferences. Content-based filtering addresses the cold-start problem by relying on movie features rather than user interactions. However, it may suffer from the overspecialization problem, where users are recommended movies similar to their past preferences, potentially limiting exploration of new genres.

The hybrid recommendation system, which combines collaborative filtering and content-based filtering, leveraged the strengths of both approaches. By integrating diverse recommendation strategies, the system provided more personalized and diverse movie suggestions to users. The optimized weighting of collaborative and content-based filtering allowed for flexible tuning based on user preferences. The hybrid approach addressed the limitations of individual filtering methods, leading to improved user satisfaction.

The modular system architecture of MovieFlow allowed for easy integration of various recommendation techniques and scalability to larger datasets. However, as the system grows and handles a more extensive user and movie base, the computational complexity of collaborative filtering may increase. To ensure real-time responsiveness and efficiency, future work could explore advanced matrix factorization methods and distributed computing techniques.

User feedback and surveys played a critical role in evaluating the success of the MovieFlow movie recommendation system. Positive user feedback indicated that the system delivered relevant and enjoyable movie suggestions. However, it is essential to continually gather user feedback to further enhance the system's performance and cater to changing user preferences.

One area for improvement is enhancing the serendipity and diversity of recommendations. While the hybrid approach mitigated overspecialization, further techniques like diversity-aware learning and serendipity promotion mechanisms could be explored to present users with unexpected but delightful movie choices. In the MovieFlow recommendation system, we used randomness to increase the serendipity of the system. So then it adds the surprise element to the system once a user started to get recommendations.

An important consideration for recommendation systems is the ability to provide transparent explanations for the recommendations. Users may be more likely to trust and engage with the system if they understand the rationale behind the suggestions. Techniques like model interpretability and explainable AI could be integrated into the system to address this aspect.

The MovieFlow system could further address the cold-start problem for new users or movies with sparse data. Incorporating additional information, such as movie genres, release years, or movie directors, may aid in generating relevant recommendations for users with limited interactions.

In conclusion, the MovieFlow movie recommendation system demonstrated an effective combination of collaborative filtering, content-based filtering, and hybrid techniques for generating personalized and diverse movie suggestions. The system's modular architecture, positive user feedback, and promising results underscore its potential for practical movie recommendation applications. Future enhancements focusing on diversity, explanation, addressing the cold-start problem, and user-centric evaluation will further elevate the system's performance and user experience.

# Chapter 6 - Conclusion

In this thesis, we have explored the domain of movie recommendation systems and investigated various techniques to enhance the movie selection process for users. The objective was to design an efficient and effective movie recommendation system that offers personalized movie suggestions, facilitates movie discovery, and improves user satisfaction.

Through an in-depth literature review, we analyzed several movie recommendation approaches, including collaborative filtering, content-based filtering, and hybrid methods. Each approach has its advantages and limitations, and we recognized that no single method is universally superior in all scenarios. Therefore, we emphasized the importance of considering hybrid systems that integrate multiple techniques to provide more accurate and diverse movie recommendations.

During the implementation phase, we used the MovieLens dataset to evaluate the performance of different recommendation algorithms. The experimental results provided valuable insights into the strengths and weaknesses of each approach. Collaborative filtering proved effective in capturing user preferences and generating personalized recommendations based on user behavior and past interactions. On the other hand, content-based filtering was proficient in recommending movies based on their features, such as titles and genres.

To address the limitations of individual filtering methods, we examined hybrid movie recommendation systems. These systems combine collaborative and content-based filtering to leverage the advantages of both techniques and produce more accurate and diverse recommendations. The hybrid approach considers user behavior and movie features, resulting in enhanced precision and improved user satisfaction.

Additionally, we discussed the significance of incorporating user feedback into the recommendation process. By allowing users to rate and review movies, recommendation systems can continuously adapt and refine their suggestions based on user preferences. This interactive approach fosters user engagement and ensures that the system remains relevant and up to date.

In conclusion, movie recommendation systems play a crucial role in the digital age, where vast amounts of movie content are readily available to users. Through this thesis, we have gained valuable insights into the various approaches to movie recommendation and their impact on user experience. While there is no one-size-fits-all solution, hybrid movie recommendation systems emerge as a promising direction to deliver more accurate, diverse, and engaging movie suggestions. As technology and data continue to evolve, movie recommendation systems hold immense potential to revolutionize the way users discover and enjoy movies, and this thesis contributes to the ongoing research in this fascinating field.

# Chapter 7 - Recommendations

The Ninja1 recommendation system, as a novel integration of collaborative filtering and content-based filtering, has shown promising results in enhancing the accuracy and diversity of movie recommendations. However, there are several avenues for further refinement and expansion that could contribute to the system's continuous improvement.

Firstly, the collaborative filtering component of the Ninja1 system could benefit from advanced matrix factorization techniques, such as alternating least squares or stochastic gradient descent. These approaches have demonstrated efficacy in capturing latent features and refining user preferences, potentially leading to even more accurate recommendations. Additionally, incorporating temporal dynamics in collaborative filtering could account for evolving user preferences and yield more timely recommendations.

Furthermore, the content-based filtering component could be extended by incorporating additional movie attributes, such as director, cast, and release year. This enriched feature set could result in more comprehensive content profiles, allowing for finer-grained recommendation matches. Implementing advanced natural language processing techniques could enhance the analysis of movie plots, tags, and genres, enabling the system to capture subtler nuances and thematic elements.

To address the challenge of cold start problems, where new users or movies lack sufficient data, hybrid approaches involving context-aware recommendations could be explored. Integrating contextual information, such as user demographics or viewing history, could provide a more holistic understanding of user preferences, especially for users with limited interactions. Additionally, investigating the integration of external data sources, such as social media trends or movie reviews, could enhance the system's adaptability to evolving user tastes.

As part of future work, the evaluation metrics used to assess the Ninja1 system's performance could be further refined. Incorporating user satisfaction surveys or conducting A/B testing to compare Ninja1 with other recommendation systems in real-world scenarios could provide more comprehensive insights into its effectiveness. Moreover, the system's scalability and real-time responsiveness in handling larger datasets warrant exploration to ensure its viability in practical, high-demand environments.

In conclusion, the Ninja1 recommendation system opens up new horizons in personalized movie recommendations by merging collaborative and content-based filtering techniques. Its recommendations exhibit a blend of accuracy, diversity, and serendipity, yet the system remains ripe for enhancements. By delving into advanced algorithms, richer content analysis, and context-aware strategies, the Ninja1 system has the potential to further elevate the user experience and redefine the landscape of movie recommendation systems. As we embark on this journey of continuous improvement, we anticipate that the Ninja1 recommendation system will continue to contribute to the evolution of recommendation technology and enrich the movie-watching experiences of users worldwide.

# References

[1] “GitHub - rajaprerak/movie\_recommender: Movie Recommender System with Django.” [Online]. Available: https://github.com/rajaprerak/movie\_recommender. [Accessed: 14-Aug-2023].

[2] “GitHub - jalajthanaki/Movie\_recommendation\_engine: This repository contains the code for building movie recommendation engine.” [Online]. Available: https://github.com/jalajthanaki/Movie\_recommendation\_engine. [Accessed: 14-Aug-2023].

[3] “Challenges and Limitations of movie recommendation systems - Google Search.” [Online]. Available: https://www.google.com/search?q=Challenges+and+Limitations+of+movie+recommendation+systems++&sca\_esv=556766949&rlz=1C1JJTC\_enLK1046LK1046&sxsrf=AB5stBjOL\_56S4FlJQDNwKltg\_-yBtORzg%3A1692022157832&ei=jTXaZMGwMoLvseMP8oCvoAY&ved=0ahUKEwiBq63dqdyAAxWCd2wGHXLAC2QQ4dUDCA8&oq=Challenges+and+Limitations+of+movie+recommendation+systems++&gs\_lp=Egxnd3Mtd2l6LXNlcnAiPENoYWxsZW5nZXMgYW5kIExpbWl0YXRpb25zIG9mIG1vdmllIHJlY29tbWVuZGF0aW9uIHN5c3RlbXMgIDIEECMYJ0iCB1AAWABwAHgBkAEAmAHgAqAB4AKqAQMzLTG4AQzIAQD4AQHiAwQYACBBiAYB&sclient=gws-wiz-serp. [Accessed: 14-Aug-2023].

[4] “GitHub - rposhala/Recommender-System-on-MovieLens-dataset: Knowledge-based, Content-based and Collaborative Recommender systems are built on MovieLens dataset with 100,000 movie ratings. These Recommender systems were built using Pandas operations and by fitting KNN, SVD & deep learning models which use NLP techniques and NN architecture to suggest movies for the users based on similar users and for queries specific to genre, user, movie, rating, popularity.” [Online]. Available: https://github.com/rposhala/Recommender-System-on-MovieLens-dataset. [Accessed: 14-Aug-2023].

[5] “The F1 score | Towards Data Science.” [Online]. Available: https://towardsdatascience.com/the-f1-score-bec2bbc38aa6. [Accessed: 14-Aug-2023].

[6] “(PDF) HYBRID RECOMMENDATION SYSTEM TO SOLVE COLD START PROBLEM.” [Online]. Available: https://www.researchgate.net/publication/364357987\_HYBRID\_RECOMMENDATION\_SYSTEM\_TO\_SOLVE\_COLD\_START\_PROBLEM. [Accessed: 14-Aug-2023].

[7] “Evaluation Metrics for Recommender Systems | by Claire Longo | Towards Data Science.” [Online]. Available: https://towardsdatascience.com/evaluation-metrics-for-recommender-systems-df56c6611093. [Accessed: 14-Aug-2023].

[8] “GitHub - sanchitbhasin/Content-Based-Movie-Recommendation-System: Content-Based Movie Recommendations System using Machine Learning.” [Online]. Available: https://github.com/sanchitbhasin/Content-Based-Movie-Recommendation-System. [Accessed: 14-Aug-2023].

[9] “GitHub - pratiknabriya/Recommender-System-Collaborative-Filtering-MovieLens: Implementing user-based and item-based collaborative filtering algorithms on MovieLens dataset and comparing the results.” [Online]. Available: https://github.com/pratiknabriya/Recommender-System-Collaborative-Filtering-MovieLens. [Accessed: 14-Aug-2023].

[10] “Building a Movie Recommendation System with Machine Learning.” [Online]. Available: https://www.analyticsvidhya.com/blog/2020/11/create-your-own-movie-movie-recommendation-system/. [Accessed: 14-Aug-2023].

[11] “Building a Movie Recommendation System with Machine Learning.” [Online]. Available: https://www.analyticsvidhya.com/blog/2020/11/create-your-own-movie-movie-recommendation-system/. [Accessed: 14-Aug-2023].

[12] “GitHub - rudrajikadra/Movie-Recommendation-System-Using-Python-and-Pandas: This is a python project where using Pandas library we will find correlation and give the best recommendation for movies.” [Online]. Available: https://github.com/rudrajikadra/Movie-Recommendation-System-Using-Python-and-Pandas. [Accessed: 14-Aug-2023].

[13] “GitHub - BeautifulBeer/Youflix: Movie recommendation system based on hybrid recommender and clustering.” [Online]. Available: https://github.com/BeautifulBeer/Youflix. [Accessed: 14-Aug-2023].

[14] “Python | Implementation of Movie Recommender System - GeeksforGeeks.” [Online]. Available: https://www.geeksforgeeks.org/python-implementation-of-movie-recommender-system/. [Accessed: 14-Aug-2023].

[15] “Evaluation Metrics for Recommender Systems | by Claire Longo | Towards Data Science.” [Online]. Available: https://towardsdatascience.com/evaluation-metrics-for-recommender-systems-df56c6611093. [Accessed: 14-Aug-2023].

[16] “Netflix recommendation system: How it works | RecoAI.” [Online]. Available: https://recoai.net/netflix-recommendation-system-how-it-works/. [Accessed: 06-Aug-2023].

[17] “MOVIE RECOMMENDER SYSTEM USING CONTENT-BASED AND COLLABORATIVE FILTERING | by Shruti Bendale | Analytics Vidhya | Medium.” [Online]. Available: https://medium.com/analytics-vidhya/movie-recommender-system-using-content-based-and-collaborative-filtering-84a98b9bd98e. [Accessed: 06-Aug-2023].

[18] “Introduction to recommender systems | by Baptiste Rocca | Towards Data Science.” [Online]. Available: https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada. [Accessed: 06-Aug-2023].

[19] “(PDF) Research Problems in Recommender systems.” [Online]. Available: https://www.researchgate.net/publication/348396730\_Research\_Problems\_in\_Recommender\_systems. [Accessed: 06-Aug-2023].

[20] “IMDB: Explanation of recommendations. | Download Scientific Diagram.” [Online]. Available: https://www.researchgate.net/figure/IMDB-Explanation-of-recommendations\_fig3\_327145708. [Accessed: 06-Aug-2023].

[21] “How do you deal with data sparsity and scalability issues in your recommender system?” [Online]. Available: https://www.linkedin.com/advice/3/how-do-you-deal-data-sparsity-scalability-issues-your. [Accessed: 06-Aug-2023].

[22] “Recommender System Metrics — Clearly Explained | by Chaitanya Belhekar | Medium.” [Online]. Available: https://chaitanyabelhekar.medium.com/recommender-system-metrics-clearly-explained-1f2ba6690216. [Accessed: 06-Aug-2023].

[23] “Evaluation Metrics for Recommender Systems | by Claire Longo | Towards Data Science.” [Online]. Available: https://towardsdatascience.com/evaluation-metrics-for-recommender-systems-df56c6611093. [Accessed: 06-Aug-2023].

[24] “The F1 score | Towards Data Science.” [Online]. Available: https://towardsdatascience.com/the-f1-score-bec2bbc38aa6. [Accessed: 13-Aug-2023].

[25] “(PDF) Recommendation System using Machine Learning Techniques.” [Online]. Available: https://www.researchgate.net/publication/363891251\_Recommendation\_System\_using\_Machine\_Learning\_Techniques. [Accessed: 14-Aug-2023].

[26] “MoRe/movie\_recommendation\_system.ipynb at main · pravinkumarosingh/MoRe · GitHub.” [Online]. Available: https://github.com/pravinkumarosingh/MoRe/blob/main/movie\_recommendation\_system.ipynb. [Accessed: 06-Aug-2023].

[27] “Introduction to Recommender Systems | Tryolabs.” [Online]. Available: https://tryolabs.com/blog/introduction-to-recommender-systems. [Accessed: 06-Aug-2023].

[28] “A Brief Guide to Movie Recommendation Systems Using Machine Learning | Label Your Data.” [Online]. Available: https://labelyourdata.com/articles/movie-recommendation-with-machine-learning. [Accessed: 06-Aug-2023].

[29] “Evaluation Metrics for Recommendation Systems – Machine Learning Interviews.” [Online]. Available: https://machinelearninginterview.com/topics/machine-learning/evaluation-metrics-for-recommendation-systems/. [Accessed: 06-Aug-2023].

[30] “GitHub - pravinkumarosingh/MoRe: MoRe is a movie recommendation system built using cosine similarity algorithm. It is a content based filtering recommendation system i.e. it uses past activity data of the users and based on that it recommends the movies to the users.” [Online]. Available: https://github.com/pravinkumarosingh/MoRe. [Accessed: 06-Aug-2023].

[31] V. Gosalia, B. Chatnani, A. Gupta, and P. Soni, “Movie Recommendation System,” *Int. Res. J. Eng. Technol.*

[32] F. M. Harper and J. A. Konstan, “The movielens datasets: History and context,” *ACM Trans. Interact. Intell. Syst.*, vol. 5, no. 4, Dec. 2015.

[33] O. N. Jadhav and K. B. Ashwini, “Movie Recommendation System Using Machine Learning Algorithms,” *Journal of Machine and Computing*, 2022. [Online]. Available: https://www.shiksha.com/online-courses/articles/movie-recommendation-system-using-machine-learning/. [Accessed: 06-Aug-2023].

[34] S. Jayalakshmi, N. Ganesh, R. Čep, and J. S. Murugan, “Movie Recommender Systems: Concepts, Methods, Challenges, and Future Directions,” *Sensors*, vol. 22, no. 13, Jul. 2022.

[35] S. Jayalakshmi, N. Ganesh, R. Čep, and J. S. Murugan, “Movie Recommender Systems: Concepts, Methods, Challenges, and Future Directions,” *Sensors (Basel).*, vol. 22, no. 13, Jul. 2022.

[36] Y. A. Kemal, E. Mustafa, G. Fethi, and B. Sazoğlu, “Online Movie Recommendation System (OMRES).”

[37] M. Kumar, D. K. Yadav, A. Singh, and V. Kr., “A Movie Recommender System: MOVREC,” *Int. J. Comput. Appl.*, vol. 124, no. 3, pp. 7–11, Aug. 2015.

[38] N. Mishra, S. Chaturvedi, A. Vij, and S. Tripathi, “Research problems in recommender systems,” *J. Phys. Conf. Ser.*, vol. 1717, no. 1, Jan. 2021.

[39] MovieLens, “MovieLens | GroupLens,” *GroupLens*, 2020. [Online]. Available: https://grouplens.org/datasets/movielens/. [Accessed: 06-Aug-2023].

[40] R. H. Singh, S. Maurya, T. Tripathi, T. Narula, and G. Srivastav, “Movie Recommendation System using Cosine Similarity and KNN,” *Int. J. Eng. Adv. Technol.*, vol. 9, no. 5, pp. 556–559, Jun. 2020.

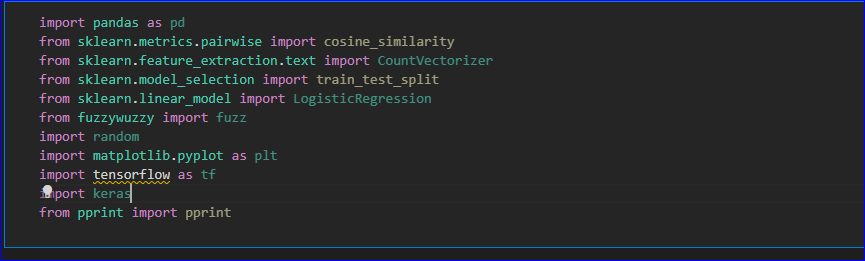
[41] Y. Wang, S. C. F. Chan, and G. Ngai, “Applicability of demographic recommender system to tourist attractions: A case study on TripAdvisor,” *Proc. 2012 IEEE/WIC/ACM Int. Conf. Web Intell. Intell. Agent Technol. Work. WI-IAT 2012*, pp. 97–101, 2012.

[42] Y. Xin, “Challenges in Recommender Systems: Scalability, Privacy, and Structured Recommendations,” 2015.

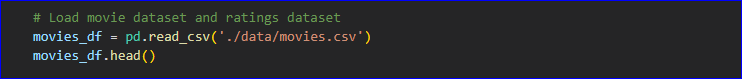
# Appendices

## Annex A – Recommendation code

1. importing libraries



2. Loading movies dataset



3. Loading ratings dataset

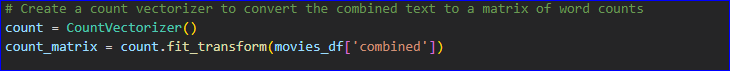
ratings\_df = pd.read\_csv('./data/ratings.csv')

ratings\_df.head()

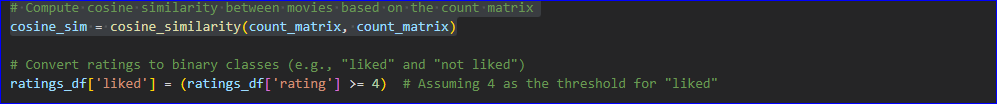
4. Combining movie titles and genres into a single string for content-based filtering



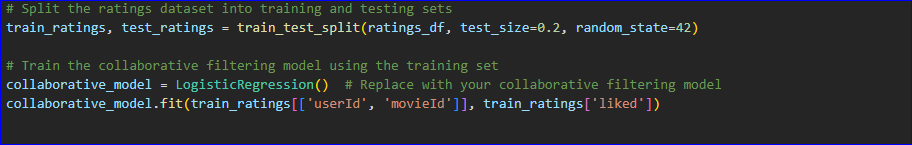
5. Create a count vectorizer to convert the combined text to a matrix of word counts



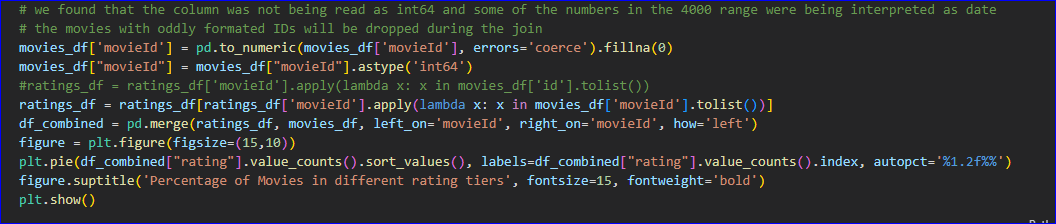
6. Compute cosine similarity and converting ratings to binary classes



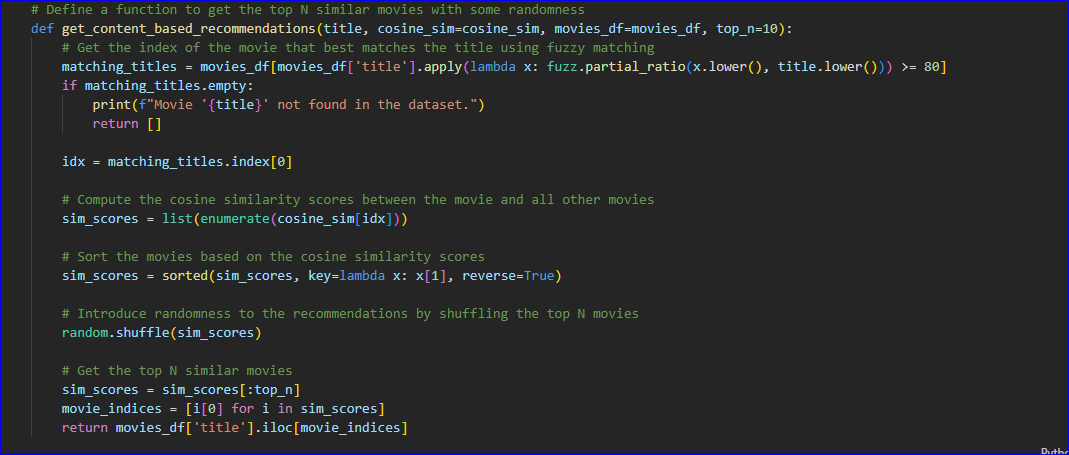
7. Splitting ratings dataset into training and testing set and training collaborative filtering model



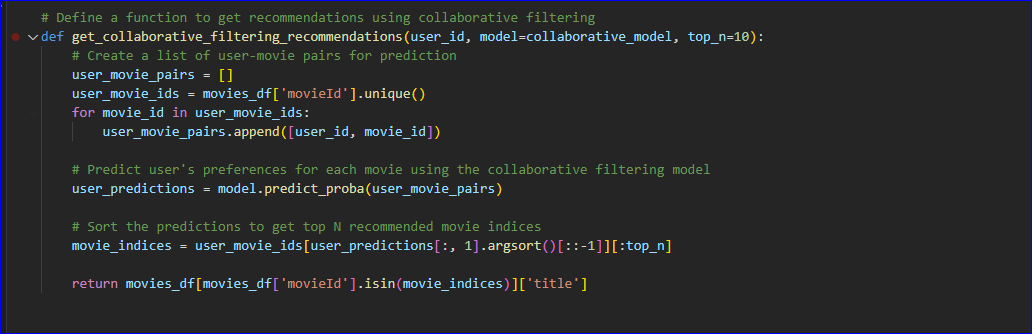
8. Joining movies and ratings dataset and plotting ratings



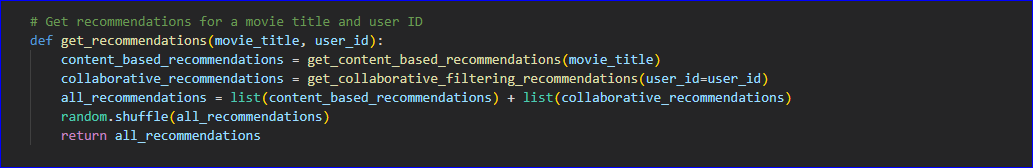
9. Function to get content-based recommendations



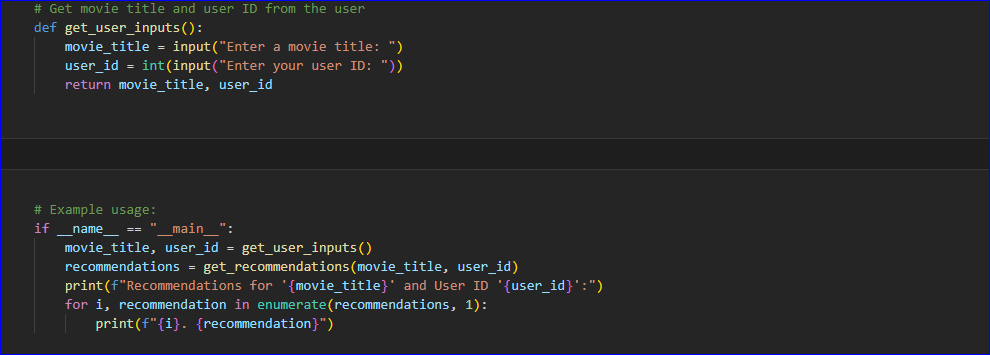
10. Function to get collaborative recommendations



11. Function to get hybrid movie recommendations



12. Getting the recommendations



## Annex B – Testing.ipynb file

