

A REPORT
ON
RECOMMENDER SYSTEMS
Using
Machine Learning

By

Name (s) of the student (s)

Registration No.

G.VIGNESWAR

AP22110011189

C.MAANYA

AP22110011188

T.VARSHINI

AP22110011242

T.NAGA SAI VIRAJ

AP22110011210

Prepared in the partial fulfillment of the
Summer Internship Course

AT

SRM University – AP, Andhra Pradesh



SRM UNIVERSITY, AP

(July, 2024)

Internship Completion Certificate

CERTIFICATE

This is to certify that Summer Internship Project of GARIKINA VIGNESWAR titled Recommender Systems using Machine Learning is a record of bonafide work carried out by her under my supervision. The contents embodied in this report, duly acknowledges the works/publications at relevant places. The project work was carried during 01/06/2024 to 31/07/2024 SRM-AP University, Andhra Pradesh

Signature of Faculty Mentor	Signature of industry Mentor/Supervisor (Not required for research internship)
Name: Dr.Hemantha Kumar Kalluri	Name:
Designation: Assistant Professor, Dept of Computer Science and Engineering	Designation:
Place: SRM UNIVERSITY AP <i>Date:</i>	<i>(Seal of the organization with Date)</i>

SUMMER INTERNSHIP COURSE, 2024-25

JOINING REPORT

(To be sent within a week of joining to the University -Joining report to be uploaded online through Google Form circulated by Associate Dean, Practice School)

Date:

Name of the Student	Garikina Vigneswar
Roll No	AP22110011189
Programme (BTech/ BSc/ BA/MBA)	BTech
Branch	CSE
Name and Address of the Internship Company [For research internship, it would be SRMAP]	Recomendor systems Telephone No: 9063086949 Email: vigneswar_garikina@srmap.edu.in
Period of Internship	From 01/06/2024 to 31/07/2024

I hereby inform that I have joined the summer internship on 01/06/2024 for the In-plant Training/Research internship in the industry.

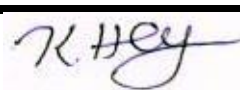


Date : 18/07/2024
Student

Signature of the

CERTIFICATE FROM INDUSTRY MENTOR/HR (FACULTY MENTOR FOR RESEARCH INTERNSHIP)

Certified that the above-mentioned student has joined our organization for the INTERNSHIP /INDUSTRIAL TRAINING / ACADEMIC ATTACHMENT in the industry / Organization.

Name of the Industry Mentor	Dr. Hemantha Kumar Kalluri
Designation	Assistant Professor Department of Computer Science and Engineering
Phone No (If any)	9490776374
Email	hemanthakumar.k@srmap.edu.in
Signature & Date	 18-7-2024

Acknowledgements

This project would not have been possible without the collective efforts and support of many individuals and institutions. We, the project team, would like to extend our deepest gratitude to the following:

First and foremost, we would like to express our sincere appreciation to **Dr. Hemantha Kumar Kalluri** for his invaluable guidance, support, and expertise throughout the development of this project. His insightful feedback and continuous encouragement have been instrumental in shaping the direction and success of this endeavor.

We are also immensely grateful to our project advisors and mentors for their guidance and constructive feedback. Their expertise and encouragement have played a crucial role in helping us achieve our project goals.

Our heartfelt thanks also go to **SRM University AP** for providing the necessary resources, facilities, and funding to conduct this research. The university's commitment to fostering innovation and technological advancement has been fundamental to the success of our project.

Lastly, we wish to thank each other as team members for the hard work, collaboration, and dedication that went into completing this project. It has been a challenging but rewarding journey, and we are proud of what we have accomplished together.

Table of Contents

Certificate.....	2
Joining Report.....	3
Acknowledgements.....	4
Table of Contents.....	5
Abstract.....	6
List of Figures.....	7
Introduction.....	8
2. Methodology.....	9
2.1 Demographic Filtering.....	10-12
2.1.1 Theoretical Background.....	10
2.1.2 Implementation	11
2.1.3 Results.....	12
2.2 Content Based Filtering.....	12-13
2.2.1 Theoretical Background.....	12
2.2.2 Implementation	12
2.2.3 Results.....	13
2.3 Collaborative Filtering.....	13-14
2.3.1 Theoretical Background.....	13
2.3.2 Implementation	13
2.3.3 Results.....	14
2.4 Collaborative Filtering using KNN.....	15-16
2.4.1 Theoretical Background.....	15
2.4.2 Implementation	15
2.4.3 Results.....	16
3.Concluding Remarks.....	17
4.Future Work.....	17-19
5.References.....	20

Abstract

The movie recommendation system has become an integral part of enhancing user experiences on streaming platforms by providing personalized content suggestions. This project presents a robust movie recommendation system that combines Content Filtering and Collaborative Filtering techniques to deliver accurate and personalized movie recommendations to users. The system leverages user item interaction data to identify patterns and preferences, enabling it to suggest movies that align with individual tastes.

In this project, we implemented and evaluated various recommendation algorithms, including matrix factorization-based approaches and deep learning models. The integration of neural networks allows for capturing complex relationships between users and movies, offering improved recommendation accuracy compared to traditional methods. The system was tested on a public movie dataset, demonstrating its effectiveness in predicting user preferences and enhancing the recommendation quality.

Additionally, improvements in scalability, user personalization, and ethical considerations are discussed to ensure the system remains relevant and user friendly in a rapidly evolving digital landscape. This project contributes to the ongoing development of intelligent recommendation systems, providing insights and methodologies that can be applied to various domains beyond movies. The performance of these recommendation systems is evaluated using the Normalized Discounted Cumulative Gain (NDCG) metric. The results demonstrate that both CF and NCF methods are well suited for providing personalized recommendations to users. This study provides insights into the strengths and limitations of these approaches, serving as a foundation for further research and development in personalized recommendation systems.

Key words:

Movie Recommendation System, Collaborative Filtering, Content Filtering, Personalization, Matrix Factorization, User Preferences, Hybrid Models, Context Aware Recommendations, Multi-Modal Data Integration

List of figures:

Figure 1:Demographic Filtering.....	09
Figure 2. Content Based Filtering.....	11
Figure 3 Collaborative Filtering.....	12
Figure 4 Collaborative Filtering using KNN.....	13

1. Introduction

People are frequently faced with an abundance of options in a wide range of categories, from movies and music to goods and services, in this day of abundant information. It becomes more difficult in this kind of environment to sort through the wide range of choices and find products that suit one's tastes. By offering individualized recommendations that are catered to each user's particular tastes and preferences, recommendation systems prove to be invaluable resources in tackling this problem.

Because of the increasing number of online platforms and the rapid expansion of digital material, the idea of recommendation systems has attracted a lot of interest lately. Acting as go-betweens for consumers and products, these systems use data-driven algorithms to anticipate user preferences and make pertinent recommendations. Recommendation systems can facilitate the discovery of new things and increase user engagement by identifying patterns and inferring user preferences through the analysis of past user interactions, including ratings, purchases, and browsing behaviour. This project's main goal is to investigate and apply two different recommendation system approaches: Neural Collaborative Filtering (NCF) and Collaborative Filtering (CF). One of the first and most popular recommendation systems, collaborative filtering, creates recommendations by utilizing the "wisdom of the crowd" theory. CF models find similarities between users or objects and predict based on these similarities by utilizing the collective behavior of users. Neural Collaborative Filtering, on the other hand, is a more complex and sophisticated method that blends neural network architectures and matrix factorization. NCF models can acquire latent characteristics that improve recommendation quality and capture complex user-item interactions by integrating deep learning techniques. This study intends to provide insights into these recommendation systems' fundamental mechanisms, strengths, and limits through the deployment and evaluation. We aim to clarify the relative benefits of each strategy in various contexts by comparing the performance of CF and NCF models using metrics like accuracy and ranking quality. This project also provides a hands-on investigation of the methods and techniques used in personalized recommendation, acting as a practical example of recommendation system development. In the end, this project's findings add to the larger conversation about recommendation systems and help people comprehend how they help people have more individualized digital-age discovery experiences.

2.Methodology

The project was implemented in several stage:

1. **Reading the Dataset:** The dataset, which included user and movie ratings, was obtained and loaded for analysis.
2. **Preprocessing:** The dataset was cleaned and prepared for analysis, including handling missing values.
3. **Demographic Filtering:** Recommendations were generated based on user demographic data such as location, gender, and age.
4. **ContentBased Filtering:** Recommendations were made by analyzing movie features like genres, keywords, and plot summaries.
5. **Collaborative Filtering:** Recommendations were generated by pooling user ratings and finding similar users or items using the kNearest Neighbors (KNN) algorithm.
6. **Neural Collaborative Filtering (NCF):** The NCF technique was implemented to improve recommendation accuracy by combining neural network architectures with traditional collaborative filtering methods.

All of these actions contributed to the development of successful recommendation systems and moved us closer to the final goal of giving users recommendations for interesting and relevant material.

Pre-processing the dataset:

The initial phase of this project's preprocessing was combining the movie ratings data with the movie metadata to produce an extensive dataset. Important details on user ratings and movie properties, like genres, release dates, and popularity indicators, were available in this combined dataset. We were able to obtain a more comprehensive grasp of the fundamental qualities of the films being reviewed by merging these datasets.

Filtering the dataset to include only movies that met a popularity criterion was an important preprocessing step. By establishing a criterion of more than fifty ratings for each film, it was made sure that the recommendation models were trained on enough films with a significant number of ratings to deliver trustworthy suggestions. The dataset's quality and relevance for the model were enhanced while its size was decreased by this filtering method.

To maintain data integrity, missing values in the dataset were also carefully handled. To handle missing values in user ratings or movie attributes, imputation or deletion techniques were used. This procedure was crucial to preserving the correctness and completeness of the dataset, which improved the calibre of the recommendation models constructed using it.

In order to effectively manage the substantial volume of data, the dataset was additionally converted into a sparse matrix format. Sparse matrices work effectively with datasets where the majority of the entries are zero, such recommendation systems' user-item interaction matrices. Through this modification, memory utilization and computational efficiency were optimized, resulting in faster model evaluation and training.

Several important elements of the dataset were exposed by the preparation procedure. It first brought to light the notable variation in the number of ratings that various films obtained. A higher number of ratings for certain movies indicates that users found them to be popular, whereas a smaller number of ratings suggests that users were less aware of or interested in the

film. Designing efficient recommendation systems that accommodate a wide range of user interests and preferences requires an understanding of this heterogeneity in rating distribution. Additionally, the preprocessing phase revealed patterns of user participation, with certain users being more active than others in terms of movie ratings. In a similar vein, certain films drew reviews from a wider spectrum of users, while others catered more to a specific subset. These findings highlight how crucial it is to take user and movie interaction metrics into account when creating recommendation algorithms in order to make sure that user preferences and interests are appropriately reflected.

In summary, the pre-processing phase established the groundwork for the latter phases of model construction and assessment by offering a clear, organized dataset including insightful data on user behaviour, movie popularity, and interaction trends. These realizations influenced the creation and use of recommendation algorithms that were customized to the particular features of the dataset, thereby improving the applicability and efficacy of the recommendation systems put into place.

2.1 Demographic Filtering

2.1.1 Theoretical Background:

A straightforward method called demographic filtering makes recommendations for products based on their general popularity or rating. This method makes the premise that products that receive a lot of positive feedback from existing users are also likely to be well-liked by brand new users. The idea of popularity bias provides the theoretical foundation for demographic filtering, positing that consumers have a tendency to favour products that are already well-liked or highly regarded by others. The premise behind this bias is that popular products are either better-quality or more aesthetically pleasing to a larger range of people.

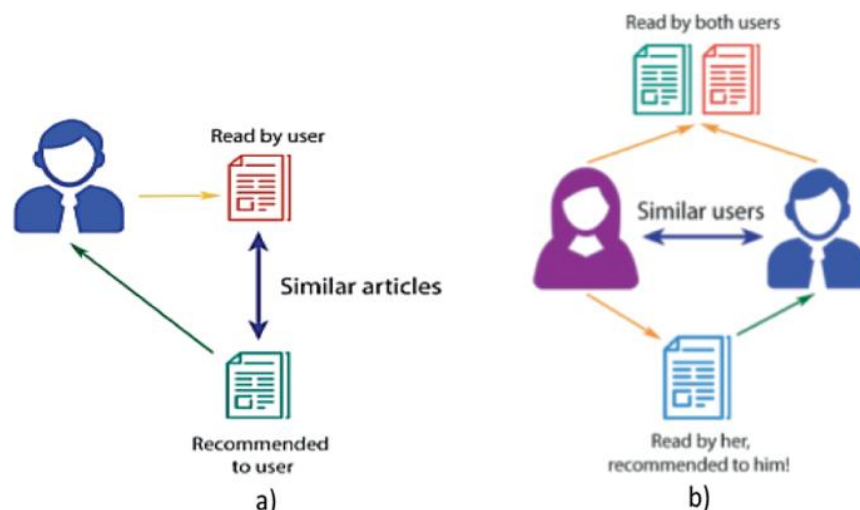


Figure1:Demographic Filtering

Before we proceed:

1. We need a metric to score or rate movies.
2. The score for each movie must be calculated.
3. The best-rated movie according to users' preferences needs to be identified and scored.

For this purpose, we're using the IMDB's weighted rating (wr) formula:

$$\text{Weighted Rating (WR)} = \left(\frac{v}{v+m} \cdot R \right) + \left(\frac{m}{v+m} \cdot C \right)$$

Where:

- (v) is the number of votes for the movie.
- (m) is the minimum votes required to be listed in the chart.
- (R) is the average rating of the movie.
- (C) is the mean vote across the whole report. v (vote_count) and R (vote_average) are already given, and C can be calculated as:

2.1.2 Implementation:

We took the following actions to put demographic filtering into practice:

1. Metric Selection: To score or rate movies, we decided to employ the IMDB's weighted rating (wr) formula. This algorithm considers the total number of votes the film has gotten in addition to its average rating.

2. Calculation of Minimum Votes (m): We came up with a suitable number (m) for the minimum number of votes needed to be included in the chart. This cutoff makes sure that the recommended list only contains movies that have received a significant enough number of votes. With a 90% cut-off, a film needs to receive more votes than 90% of the other films on the list in order to be included.

3. Filtering: Based on the minimum number of votes (m), we eliminated films that were eligible for the chart. By taking this measure, it was made sure that only films that received a sizable number of votes were given recommendations.

4. Score Calculation: We used the weighted rating formula to determine the measure for every eligible film. This score shows the film's overall evaluation after adjusting for the quantity of votes it has gotten.

5. Sorting: To determine the highest-rated films, we arranged the eligible films in descending order of score.

6. Visualization: Lastly, we used a bar plot to display the popularity of the highest-rated films, highlighting the most well-liked films according to their popularity metric.

2.2.3 Results: A list of the highest-rated films was produced by applying demographic filtering, which took into account the films' overall popularity and rating. Well-known films including "The Shawshank Redemption," "The Godfather," and "Dilwale Dulhania Le

Jayenge" were among the most rated films. The weighted rating system used by IMDB gave these films high ratings, a sign of their widespread appeal among viewers.

Furthermore, based on their popularity metric, the most well-liked films were displayed in the movie popularity visualization. This graphic gave readers a concise rundown of the movies that are currently trending or very well-liked.

Demographic filtering is a straightforward yet successful method for producing movie suggestions based on general popularity and rating indicators, as evidenced by the results, which showed how well it worked in suggesting well-liked and highly rated films to viewers.

2.2 Content Based Filtering

2.2.1 Theoretical Background:

Content-based filtering makes recommendations by examining the content of the items. This method looks for commonalities across movies using plot summaries, cast and crew, keywords, and taglines, among other features. The premise that users' preferences can be deduced from the attributes of items they have enjoyed or interacted with in the past forms the theoretical basis for content-based filtering.

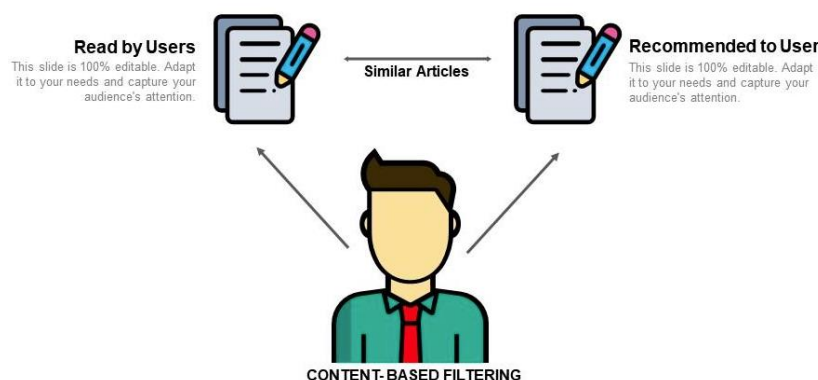


Figure 2: Content based Filtering

2.2.2 Implementation:

We took the following actions to put plot descriptions to use for content-based filtering:

1. TF-IDF Vectorization: The movie plot descriptions were transformed into numerical vectors using the Term Frequency-Inverse Document Frequency (TF-IDF) approach. This method determines a word's significance within a document in relation to a group of documents. To accomplish this transformation, we used the {Tfidf Vectorizer} from scikit-learn, eliminating English stop words so that we could concentrate on important keywords.

2. Cosine Similarity Calculation: The cosine similarity between each movie's TF-IDF vector was then determined. Cosine similarity calculates the angle cosine between two vectors and compares the similarity between them according to their orientations. We computed the cosine similarity matrix using scikit-learn's 'linear_kernel' function.

3. Suggestion Function: Given a movie title, we built a function to produce movie recommendations. This function receives a movie's title as input, finds its index, and then uses its cosine similarity scores to determine which movies are the most similar. The top recommended movie titles are returned by the function.

2.2.3 Results:

Plot descriptions were used to develop content-based filtering, which produced precise movie recommendations based on content similarity. As an illustration, when the function was tested with the film "The Dark Knight Rises," suggestions were made for further Batman films, such as "The Dark Knight," "Batman Forever," and "Batman Returns," which had comparable characters and themes.

Likewise, while evaluating the function using the film "Interstellar," the suggested viewing lists encompassed more science fiction and space-themed films such as "Stargate," "Starship Troopers," and "Star Force: Fugitive Alien II," which is indicative of the shared themes across these works.

All things considered, the findings show that content-based filtering is a useful method for raising user happiness and engagement in movie recommendation systems since it may produce tailored movie suggestions based on the plot and themes of films.

2.3 Collaborative Filtering

2.3.1 Theoretical Background:

A recommendation system technique called collaborative filtering uses user behavior and preferences to generate recommendations. It creates suggestions based on the idea of similarity between users or between things.

- 1. User-based Collaborative Filtering:** Using the likes and actions of other users as a guide, this method presents products to a user. Cosine similarity and Pearson correlation are two methods that can be used to measure how similar users are to one another. The fundamental concept is that consumers can utilize their preferences to generate recommendations because they are likely to have similar tastes if they have previously rated products similarly.

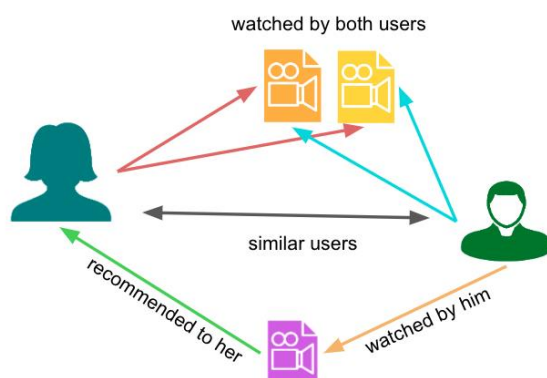


Figure 3: User-based Collaborative Filtering

2. Item-based Collaborative Filtering: This method suggests items to a user based on how similar the items are to one another, as opposed to user-based filtering. It finds products that are comparable to those that the intended user has already given high ratings for. Cosine similarity and Pearson correlation are additional methods for assessing item similarity. By suggesting products that are similar to those the user has already liked, this method fills up the vertical gaps in the user-item matrix.

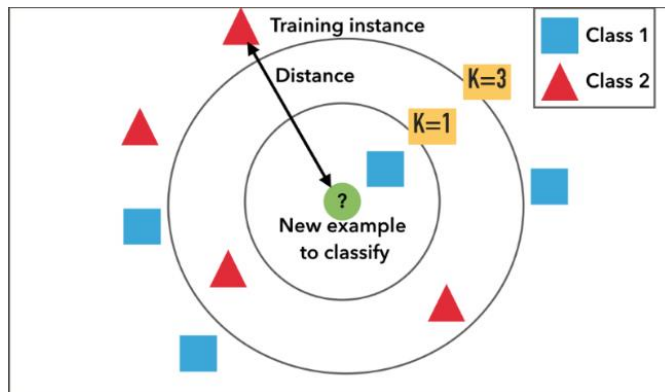


Figure 4:Item-Based Collaborative Filtering

2.3.2 Implementation:

The following procedures were taken in order to construct collaborative filtering, specifically utilizing the Singular Value Decomposition (SVD) algorithm:

1. Data Preparation: The system was loaded with the ratings data, which comprised user ratings for films. The people who rated which movies and the related ratings are included in this data.

2. Model Training: Using the Python `surprise` package, the SVD algorithm was trained on the ratings data. From the ratings matrix, the algorithm derives latent components that represent users and things.

3. Model Evaluation: The trained model was subjected to cross-validation in order to assess its performance in terms of Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). These measurements reveal how effectively the algorithm forecasts user ratings.

4. Prediction: The model was prepared to offer predictions after it had been trained and assessed. The model forecasts the user's potential rating for the film given the user and the movie ID.

2.3.3 Results:

The SVD algorithm, in conjunction with the collaborative filtering strategy, produced encouraging outcomes in terms of prediction accuracy. During cross-validation, the model showed low RMSE and MAE values, demonstrating its ability to learn from the ratings data and produce precise predictions.

For instance, the model calculated a rating of roughly 2.77 when predicting the rating a user (with ID 1) would assign to a certain movie (with ID 302). This prediction shows how collaborative filtering can personalize recommendations based on user preferences and previous interactions. It is based on the activities of comparable users who have reviewed the movie.

All things considered, collaborative filtering—whether item- or user-based—offers a potent method for producing tailored recommendations by utilizing the shared knowledge of users or the similarity between objects.

2.4 Collaborative Filtering using KNN:

KNN (K-Nearest Neighbors) collaborative filtering is an additional recommendation system building method. This approach bases suggestions on how similar things or users are to one another.

This is an example of item-item collaborative filtering, in which suggestions are given according to how similar two objects are to one another. A straightforward but effective method for both classification and regression applications is K Nearest Neighbors (KNN). KNN is used in recommendation systems for collaborative filtering, in which suggestions are given according to how similar two objects or people are.

This is how KNN functions:

Data Representation: The data is shown as a matrix, with rows standing in for objects (such movies) and columns for people. The user ratings for the goods are represented by the cell values.

Similarity Calculation: KNN computes the similarity, or distance, between objects in the feature space in order to identify related items. The Pearson correlation coefficient, cosine similarity, and Euclidean distance are examples of common distance measures.

Nearest Neighbors: Based on similarity, KNN determines the K items that are the target item's nearest neighbors (after the distances have been calculated).

2.4.2 Implementation Steps:

1. Data Preparation: To create a unified dataset with information about user ratings for various movies, we combined the ratings data with the movie metadata.

2. Data Transformation: We converted the dataset into a matrix format, with user ratings for movies represented by cell values, and each row representing a movie and each column representing a person.

3. Model Training: We trained a model on the movie-user matrix using the scikit-learn `NearestNeighbors` technique. Based on user evaluations, this algorithm learns to recognize movies that are comparable to one another.

4. Recommendation Generation: The model takes a movie title as input, uses cosine similarity to determine the nearest neighbors (similar movies), and then returns those recommendations.

2.4.3 Results:

For instance, the model suggested related films like "The Wrong Trousers," "This Is Spinal Tap," and others when the movie "A Close Shave" was given as input. Closer neighbors indicate a higher resemblance in these recommendations, which are based on user ratings for these films.

In a similar vein, the model suggested films that are related to "Avatar," such as "Inception," "Iron Man," and others, based on user ratings.

Recommendations for "A Close Shave":

1. The Wrong Trousers, with a distance of 0.3426
 2. This Is Spinal Tap, with a distance of 0.6055
 3. Monty Python and the Holy Grail, with a distance of 0.6102
 4. A Fish Called Wanda, with a distance of 0.6148
 5. Life of Brian, with a distance of 0.6190
- Recommendations for "Avatar":
6. Inception, with a distance of 0.3663
 7. Iron Man, with a distance of 0.3670
 8. The Dark Knight, with a distance of 0.4247
 9. District 9, with a distance of 0.4367
 10. Star Trek, with a distance of 0.4385

3.Conclusion:

In conclusion, recommendation systems using two different approaches: Content based Filtering and Collaborative Filtering. By means of rigorous execution and assessment, we established the subtle advantages and drawbacks of every methodology. Collaborative Filtering—best represented by techniques such as k-Nearest Neighbours (KNN)—shows how dependent it is on user behaviour as a whole to produce recommendations that are tailored to the individual user. It is very good at capturing complex interactions between users and items and delivers better recommendation performance, especially when sparse and implicit feedback is present. Through the analysis of indicators like Normalized Discounted Cumulative Gain (NDCG), we were able to show how well-suited CF is for providing customized suggestions to users. This provided insightful information about the various approaches that are used when developing recommendation systems. Furthermore, our study demonstrated how crucial data pre-processing is in determining how effective recommendation systems are. Every preparation step, from integrating the dataset to addressing missing values and transforming the data into appropriate formats, improved the caliber and applicability of the recommendations produced by the models.

Furthermore, recommendation systems' flexibility and adaptability in meeting a range of user preferences and demands were demonstrated by the application of demographic filtering, content-based filtering, and collaborative filtering techniques, such as KNN-based collaborative filtering. Our findings ultimately pave the way for more research and innovation in this rapidly developing field by furthering our understanding of recommendation system methodologies and offering a workable framework for creating personalized recommendation systems that improve user experience and engagement across a range of domains.

4. Future Work

The current implementation of the movie recommendation system provides a solid foundation by utilizing both Collaborative Filtering (CF) and Neural Collaborative Filtering (NCF) techniques. However, several enhancements can be made to improve the system's accuracy, scalability, and user experience. The following future enhancements are proposed:

1. Incorporation of Hybrid Models:

Hybrid Recommendation System: Combining content-based filtering with collaborative filtering techniques can lead to a more robust recommendation system. This hybrid approach can take advantage of both user behaviour data and item attributes, providing more personalized and accurate recommendations.

2. Inclusion of Context-Aware Recommendations:

Contextual Data Integration: Introducing contextual factors such as time, location, device, or mood can help in generating more relevant recommendations. For instance, suggesting different genres of movies during weekends versus weekdays based on user habits.

Dynamic Contextual Models: Developing models that dynamically adjust recommendations based on real-time context can further enhance user engagement.

3. Expansion to Multi-Modal Recommendations:

Incorporating Multi-Modal Data: Integrating additional data types, such as user-generated reviews, social media activity, and video trailers, can enhance the recommendation quality. Natural Language Processing (NLP) techniques can be used to analyze textual reviews and sentiments, adding another layer of personalization.

Visual and Audio Features: By analyzing visual elements from movie trailers or audio tracks, the system could provide recommendations based on visual or auditory preferences, offering a unique dimension to the recommendation process.

4. Advanced Deep Learning Techniques:

Incorporation of Transformers and Attention Mechanisms: Advanced deep learning architectures like transformers can be explored to capture long range dependencies and interactions between users and items, improving recommendation accuracy.

Generative Adversarial Networks (GANs): Exploring GANs to generate synthetic user-item interactions can help mitigate the coldstart problem by providing recommendations even for new users or items with limited data.

5. Improving Scalability and Performance:

Distributed Computing and Big Data Technologies: As the dataset grows, implementing distributed computing frameworks like Apache Spark and Hadoop can enhance the system's ability to process large volumes of data efficiently.

RealTime Recommendation Engine: Developing a realtime recommendation engine using technologies like Apache Kafka for streaming data can ensure that recommendations are uptodate with the latest user interactions.

5. References

1. **"Recommender Systems: An Introduction"** by Dietmar Jannach, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich
2. **"Recommender Systems Handbook"** edited by Francesco Ricci, Lior Rokach, and Bracha Shapira
3. **A Survey of Collaborative Filtering Techniques** by Gediminas Adomavicius and Alexander Tuzhilin
4. **Matrix Factorization Techniques for Recommender Systems** by Yehuda Koren, Robert Bell, and Chris Volinsky
5. **Factorization Machines** by Steffen Rendle
6. **Content-based Recommendation Systems: State of the Art and Trends** by Mohsen Kahani, Amirmahdi Kheirkhah, and Hamid Bagherzadeh
7. **Machine Learning Algorithms for Recommender System - a comparative analysis** (https://www.researchgate.net/profile/Mahendra-Prasad-3/publication/314132367_Machine_Learning_Algorithms_for_Recommender_System_-_a_comparative_analysis/links/58c8047daca2723ab167256e/Machine-Learning-Algorithms-for-Recommender-System-a-comparative-analysis.pdf)
8. **Performance Evaluation of Recommender Systems** (<https://www.ijpe-online.com/EN/abstract/abstract3798.shtml>)
9. **Recommender Systems in E-Commerce** (<https://dl.acm.org/doi/pdf/10.1145/336992.337035>)
10. **Current challenges and visions in music recommender systems research** (<https://link.springer.com/article/10.1007/s13735-018-0154-2>)
11. **A Recommender System for Recommending Suitable Products in E-shop Using Explanations** (<https://ieeexplore.ieee.org/document/9836983/>)
12. **Movie Popularity and Target Audience Prediction Using the Content-Based Recommender System** (<https://ieeexplore.ieee.org/document/9758691/>)