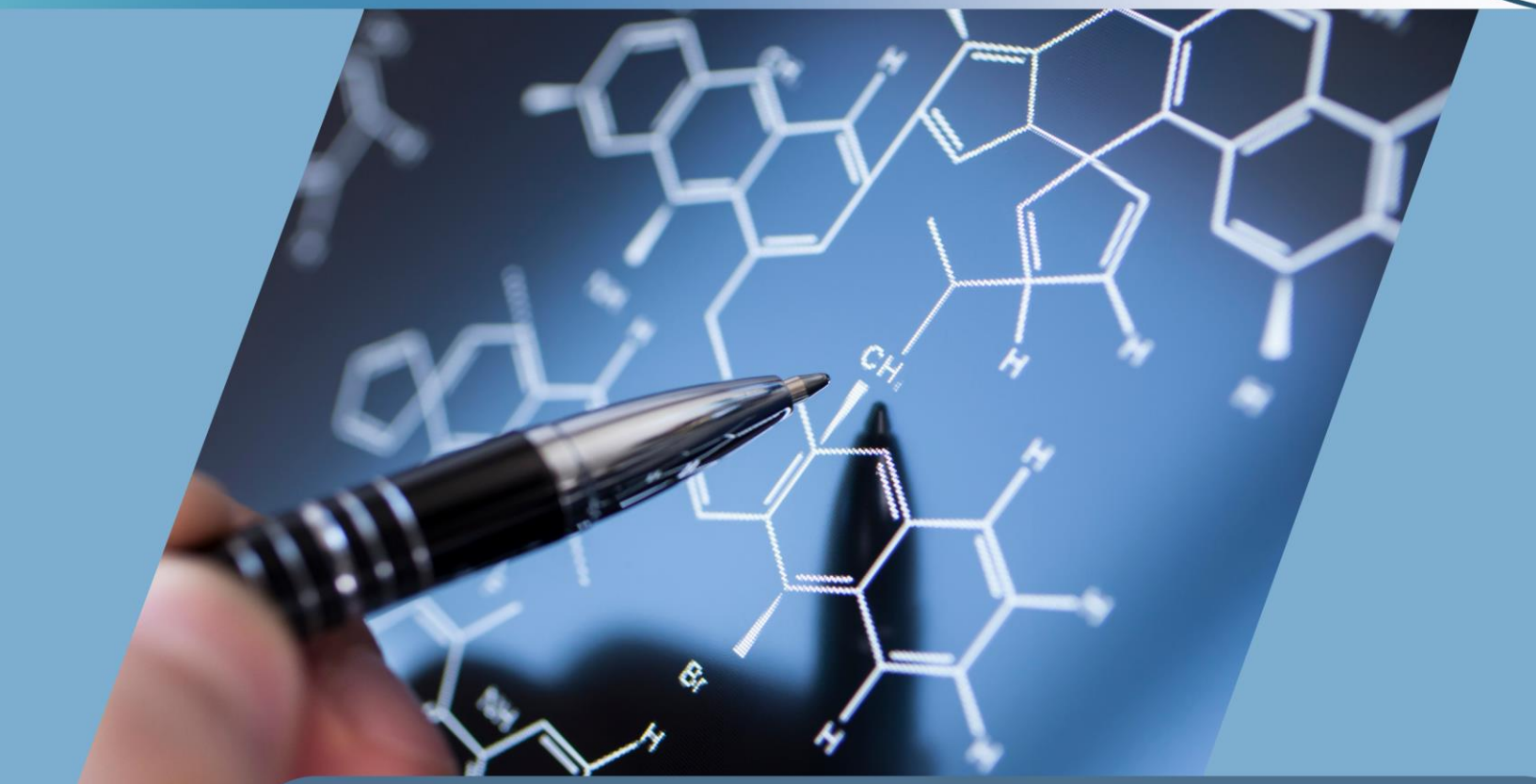
eISSN: 2789-858X

Scientific Journal for the Faculty of Science - Sirte University (SJFSSU)

Bi-annual, Peer- Reviewed, and Open Accessed e-Journal



VOLUME 4 ISSUE 1 APRIL 2024

10.37375/issn.2789-858X

Legal Deposit Number@National Library (Benghazi): 990/2021

jsfsu@su.edu.ly

Scientific Journal for the Faculty of Science-Sirte University Vol. 4, No. 1 (2024) 80-88

Scientific Journal for the Faculty of Science-Sirte University

Journal home page: http://journal.su.edu.ly/index.php/JSFSU/index

DOI: 10.37375/issn.2789-858X

Building Recommender Systems with Machine Learning and Data Mining Techniques

Yousuf A. Maneetah, Suhil M. Elsibai, Ali A. Bouras, Ahmed H. Alhabbh, Fathia Elbadri

Computer Science Department, Benghazi University, Benghazi, Libya.

DOI: https://doi.org/10.37375/sjfssu.v4i1.2677

ARTICLE INFO:

Received: 19 January 2024

Accepted: 06 April 2024

Published: 17 April 2024

Keywords: *Recommender systems, dataset, Content-based filtering, Collaborative filtering, hybrid filtering.*

A B S T R A C T

The current study presents a unique use of machine learning algorithms for developing a recommendation system. Recommender systems are often employed in a wide range of industries, including e-commerce, entertainment, and search engines. Recommender systems are algorithms that utilize user preferences and behavior to recommend relevant objects, such as movies, books, goods, or songs. This article examines the many characteristics and features of different methodologies used in recommendation systems, with an emphasis on filtering and prioritizing important information to serve as a compass for searches. When recommender engines properly recommend individualized content or items, they provide businesses with a competitive advantage over rivals and generate considerable income. This study investigates content-based filtering, which suggests things with comparable attributes to those that a user previously liked. It also investigates hybrid filtering, which combines content-based and collaborative filtering techniques (using user-item interaction data) to solve issues such as the cold start problem (little user data) and data sparsity (few user-item interactions). The installed recommender systems that use content-based and hybrid filtering approaches produce promising individualized suggestions. Content-based filtering is particularly effective at proposing comparable goods, but hybrid filtering provides a more diversified and accurate suggestion pool by including user preferences. Content-based filtering has limits owing to data sparsity, which hybrid filtering addresses. This article discovered that the suggested technique uses content-based filtering when applied to small to medium-sized datasets. However, hybrid filtering is used when the dataset is vast and sparse.

1 Introduction

Recommender systems (RSs) aim to provide meaningful suggestions to consumers based on their preferences, such as suggesting books on Amazon or movies on Netflix. Recommendation systems (RSs) use domain-specific data. For example, Netflix keeps user ratings for movies in a huge table, which allows the algorithm to create suggestions based on this information. There are several recommendation algorithms, each having advantages and disadvantages. Combining these techniques typically results in enhanced performance.

The current study investigates several recommendation approaches, examining their data sources and the algorithms that process that data. RSs are collaborative filters that look for similarities between users. Collaborative filtering is based on previous user-item interactions maintained in a "user-item interactions matrix" (Reddy et al., 2023). In order to improve suggestion diversity and personalization, this study presents a direct comparison of content-based and hybrid approaches, goes deeper into sparsity mitigation strategies, and examines user demographics and implicit feedback data.

***Corresponding author*: E-mail:** yousuf.maneetah@uob.edu.ly

Problem statement

Do various domains, such as e-commerce, entertainment, and scientific libraries require reliable and accurate recommendation engines? The objective is to deliver users trustworthy and pertinent recommendations while taking into account their interests and behavior. In order to boost user satisfaction and recommendation system effectiveness, the main focus is on evaluating various recommendation techniques, data sources, and algorithms.

Research questions

1. What are the different recommendation methods used in various domains, and how do they utilize data sources?
2. How do collaborative filtering and content-based filtering contribute to the performance of recommendation engines?
3. What are the strengths and weaknesses of different recommendation algorithms?

Aiming to analyze, compare, and evaluate different aspects of recommendation systems, including methods, data sources, algorithms, and their impact on user behavior and organizational outcomes. Literature reviews on methods of recommender systems with their challenges and limitations are specified in Section 2, methodology is specified in Section 3, and results obtained and conclusion are shown in Sections 4 and 5.

2 Literature Review

2.1 Related Work

Supermarkets (RS): IBM researchers developed a recommender system to be integrated with Smart Pad. It was created as an online shopper's personal assistant (Zahrawi et al., 2021).A different study focuses on using cosine similarity and the TF-IDF algorithm to a content-based filtering strategy. It investigates how to extract keywords and combine keywords with genres to suggest movies. Performance evaluation criteria, including f1-score, accuracy, and recall, are used, and the Movie Lens dataset, which is accessible to the public, is utilized. The results of the experiment show that the relevance and accuracy of the movie recommendation system have limits. In particular, it is discovered that the content-based filtering approach with TF-IDF and cosine similarity, in addition to keyword extraction utilizing the sclera library, is less successful in producing pertinent movie suggestions (Permana et al., 2023). Recommendation systems, which are extensively utilized in a variety of industries including e-commerce, entertainment, and search engines, are developed using machine learning techniques. Recommender systems make use of algorithms to offer consumers recommendations for related goods, music, movies, and books. As a useful tool for directing user queries, the article examines several strategies and characteristics used in recommendation systems to filter and prioritize content (Zahrawi et al., 2021). In order to forecast user preferences for movie genres and their rating behavior, the study offers a unique method that makes use of psycholinguistic variables collected from social media interactions. It creates links between ideals and personality through conversations on Twitter and IMDb. The research contrasts these strategies with conventional approaches and finds that the combined models perform better than single models based only on personality or values. Using information from Flexible, the research looks at Netflix, the most widely used on-demand broadcasting network worldwide. The authors used the TF-IDF and cosine similarity algorithms to create a recommendation system after analyzing 7,787 distinct data. Although the system has flaws, the authors think it has potential for future features because of the analysis's insights about Netflix's content patterns (Chiny et al., 2022). In order to help readers choose the next book to read, this study presents a hybrid recommendation system. The study evaluates the suggested algorithm's efficacy using the LitRec dataset and focuses on book and author suggestions within a hybrid recommendation framework. Two item-based collaborative filtering algorithms are combined in the hybrid technique to predict books and authors based on user preferences. The booklist that results from extending the author predictions is combined with the original book predictions. Moreover, the best book recommendations are generated using the given booklist. The author illustrates how author recommendations can enhance total book recommendations through a series of studies (Vaz et al., 2021). In the context of e-commerce businesses, recommendation systems (RS) are discussed in this study with a particular emphasis on book recommendation systems utilized by sites such as Amazon and Barnes & Noble. Products are suggested by RS software depending on a user's preferences or previous purchases. In RS, lists of items that are similar

to the user's preferences are often generated through the use of collaborative filtering (CF). However, CF has issues with sparsity, scalability, and cold start issues, which affects how accurate the recommendations are. The research suggests employing collaborative filtering with Jacquard similarity (JS) to generate recommendations that are more accurate in order to address these issues. When two books are paired together, JS creates an index by dividing the total number of users who have rated each book separately by the ratio of common users who have rated both books. Better recommendations are indicated by higher JS indices, and recommended books with high JS indices are given priority in the list (Rana et al., 2019). Group Lens Research at the University of Minnesota maintains the Movie Lens dataset website. When a new user joins Movie Lens, it suggests some of the most well-known movies for them to evaluate. Movie Lens makes individualized predictions for movies you are likely to appreciate based on your movie ratings. It also assists you in avoiding movies that might not suit your tastes. The user is given recommendations for other movies based on these ratings. Moreover, Movie Lens generates customized suggestions through collaborative filtering using these ratings (Khan et al., 2020).

2.2 Recommendation Systems

The recommendation system provides product recommendations based on user preferences, history, and information in order to obtain desired products (Dong et al., 2022; Attalariq et al., 2023). Collaborative filtering and content-based filtering are two extensively used strategies for developing recommendation systems. Collaborative filtering is a technique for evaluating and predicting items based on the opinions and similarities of other users. On the other hand, content-based filtering is a technique for recommending a product based on the availability of similar material (Ko et al., 2022).

2.2.1 Collaborative filtering

Collaborative filtering is a popular approach in recommender systems for making tailored recommendations. It is based on the idea that people who have had similar preferences in the past are more likely to have similar choices again. The technique entails examining user behavior and item evaluations to identify trends and similarities. There are two main techniques for collaborative filtering: user-based and item-based. User-based filtering compares people based on their prior behavior and ratings, locating those who rated products similarly to the target user. It then suggests things that comparable people have enjoyed or rated highly. Item-based filtering, on the other hand, uses user ratings to determine similarities between things, detecting objects that have been rated similarly by users. It offers things that are comparable to those that the target user has previously rated or liked. Collaborative filtering can be improved using matrix factorization techniques such as singular value decomposition (SVD). These strategies minimize the complexity of the rating matrix, revealing latent variables or characteristics that represent the underlying patterns in user-item interactions (Reddy et al., 2023).

2.2.2 Content-Based Filtering

Content-based filtering is a type of recommendation system that uses data or information about consumers' interests based on previous interactions. Content-based filtering algorithms seek to recommend items by comparing their resemblance to those that previous users have rated highly. The most appropriate things are then recommended by comparing the similarity of the items previously assessed by the user to other items in the collection. Several algorithms are widely employed in content-based recommendation systems. These include TF-IDF (Term Frequency-Inverse Document Frequency), which calculates the importance of words in a film's description or synopsis, and cosine similarity, which measures the similarity between films based on their input for content-based recommendation systems that include various film-related features, as illustrated in Figure 1. This includes a variety of data kinds, including textual data like film titles, genres, synopses, and user-generated tags, as well as graphic elements like film posters and even audio features like soundtracks. However, in this study, the author will concentrate on the movie domain, including aspects such as genres, keywords, and a mix of both. The combination of genre and keyword refers to the merger of both aspects via a concatenation process. Content-based systems can use these inputs to capture unique film attributes and generate recommendations according to the user's preferences and interests (Permana et al., 2023).

i . TF-IDF :

The acronym TF-IDF stands for Term Frequency Inverse Document Frequency of Records. TF-IDF determines the significance of a word inside a document. The TF-IDF value of a word reflects its level of categorization inside the document. A greater score

for a term indicates its significance in the paper (Ko et al., 2022). According to the above description, TF-IDF is determined using equation (1) as follows:

$$tfi\,df_i = tf_i \times idf_i \quad (1)$$

tf_i (Term Frequency of a term in a document) represents the number of times a term appears in a given document. It measures the importance or frequency of the term within the document. idf_i (Inverse Document Frequency of a term) represents the inverse document frequency of a term across the entire corpus or collection of documents. It measures the rarity or uniqueness of the term in the entire collection (Rani et al., 2021).

ii . *Cosine Similarity*

Cosine similarity is an algorithm for calculating the similarity between a desired collection of data and a provided set of data. It compares two documents based on size differences and computes the cosine angle between the two vectors in a multidimensional space. In this work, the authors use cosine similarity as a computational metric to detect the presence or absence of certain phrases in an item. Recommendations are then generated based on the degree of similarity between the relevant item and the active item connected with a certain user. The likelihood of recommending an item to a user improves with increased levels of resemblance to the active item. Cosine similarity necessitates converting the dataset's data items as vectors, which allows for complete vector-based analysis and comparison. This technique makes it easier to provide efficient and correct suggestions in the research setting (Yunanda et al., 2022). Based on the preceding description, cosine similarity is calculated using the following formula:

$$Cos(x,y) = \frac{x\,.\,y}{\|x\| * \|y\|} \quad (2)$$

The formula $Cos\ (x, y)$ calculates the cosine similarity between x and y. x is the vector acquired from the TF-IDF calculation on the active item, and y is the vector received from the TF-IDF calculation on the reference item. $\|x\|$ represents the unit length of vector x, and $\|y\|$ represents the unit length of vector y.

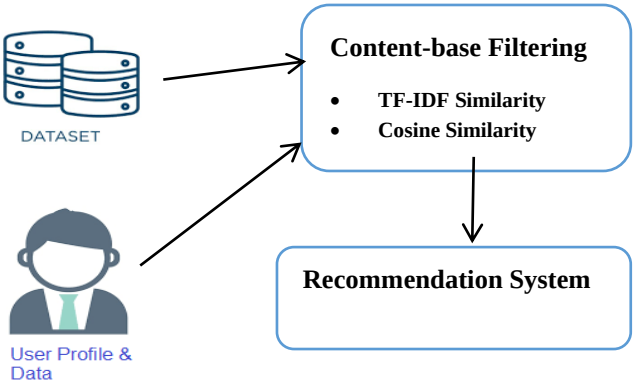


Figure (1) Content-Based Filtering Recommended System Model.

2.2.3 Hybrid Filtering

A hybrid recommender system combines many methods, such as collaborative filtering, content-based filtering, and other approaches, to capitalize on their particular strengths while mitigating their limitations. A hybrid recommender system uses numerous methodologies to deliver more accurate and diversified recommendations, hence enhancing the recommendation engine's overall effectiveness. The hybrid technique enables a more extensive examination of user preferences and item features, resulting in higher suggestion quality.

i. Singular Value Decomposition

Singular Value Decomposition (SVD) is a technique used in the creation of model-based collaborative filtering (CF) recommendation systems. This technique is one of several methods to the matrix factorization method. In this approach, the author has a list of users, things, and user ratings, which are often represented as a user-item rating matrix. Based on this, the SVD algorithm calculates the latent factor and generates suggestions using the user-item matrix.

$$\boldsymbol{A = U \cdot S \cdot VT} \quad (3)$$

The matrix decomposes into three additional matrices, as shown below: $\boldsymbol{M}$ is a $m\ x\ n$ utility matrix, U is a $m\ x\ m$ orthogonal matrix, S is a $r\ x\ r$ diagonal matrix, and VT is a $r\ x\ n$ matrix arising from the orthogonal matrix. Thus, matrix U represents users with latent factors; matrix VT represents objects with latent factors; and the diagonal matrix S is the singular value. Figure 2 depicts this process clearly.

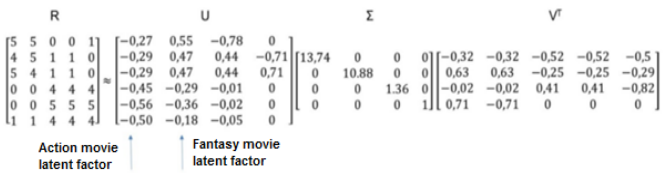


Figure (2) Singular Value Decomposition example (Baesens et al., 2020).

In this study, the authors employ the appropriate singular vector to achieve item-based collaborative filtering. Item-based CF with SVD computes item similarity using SVD vectors and generates recommendations based on user-item interaction. This strategy is efficient in dealing with huge datasets and has the ability to provide correct suggestions by exploiting the latent characteristics recorded by singular value decomposition (SVD), as shown in Figure 3. The matrix has several *k* latent components for each item (Attalariq et al., 2023).

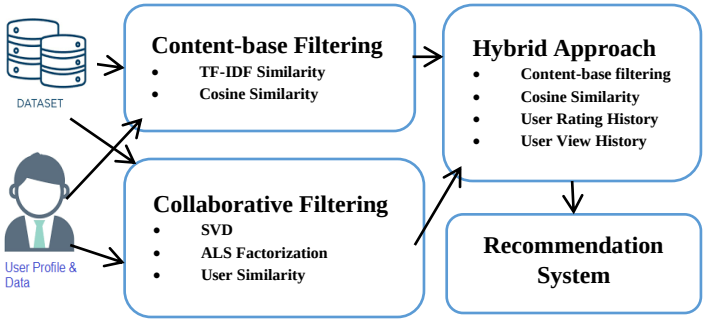


Figure (3) Hybrid Filtering recommended system model.

2.2.4 Challenges and Limitations

i. Cold Start Problem:

For new users, encourage them to evaluate certain things or submit initial preferences so that data may be collected for individualized suggestions. Use the ratings of other users with comparable demographics to generate first suggestions to new users. Regarding new items: Encourage consumers to review and offer comments on new products in order to gather data for recommendations. Collect ratings from a small portion of the community before spreading to the complete community.

ii. Data Sparsity Problem:

Content-based techniques, which depend on item content rather than user evaluations, can assist to alleviate data scarcity. Investigate strategies such as matrix factorization or neighborhood-based algorithms for identifying successful neighbors and making suggestions based on their scores.

iii. Scalability:

Optimize enormous dataset processing time by removing unused data or using scalable techniques and distributed computing technologies. Use techniques such as parallel processing or distributed computing to address the computational complexity of massive datasets.

iv. Dynamic and Evolving Preferences:

User preferences might change over time; therefore recommender systems must adapt and deliver recommendations that match users' changing interests. Tracking and comprehending these changes presents problems for the system. The cold start problem, data sparsity, scalability, and changeable preferences are key problems for recommender systems. To solve these issues, options include requesting users for initial ratings, reusing comparable data, reducing data sparsity, controlling computational complexity with optimization approaches, and reacting to changing user preferences, which necessitate continual knowledge and tracking (Ankam., 2016).

3 Proposed Methodology

Recommendation engines must be arranged a certain way because they are usually built using big datasets. This paper aims to introduce several forms of reinforcement learning (RS) and demonstrate a methodical approach to developing a Python recommendation system. Due to the extensive usage of linear algebra, a fully evolved recommendation system is resource and math-intensive. Content-based and hybrid filtering are the two types of recommender engines that are most often used. The proposed system is done with the following configurations:

- Intel R CoreTMi3-2330 M 2.20 GHz 4 GB RAM 32-bit Windows 7 Home Premium. The experiments are carried out on the dataset files using the Python language version 3.8.6 using several libraries: sklearn, numpy, pandas, and scipy.
- Deploying two types of recommendation systems (RS):
  i. Content-base filtering techniques.
  ii. Hybrid filtering techniques using singular value decomposition.

3.1 Datasets

Movie Lens is an established movie recommendation dataset that is utilized extensively in recommender system research and development. It was built by the

University of Minnesota's Group Lens research group and has been updated and maintained on a regular basis over time. Movie Lens databases include detailed information on movies, ratings, and user preferences. Recommendation engines are generally built using Movie Lens datasets, which include 100,873 rows and 5 columns (user_id, item_id, rating, timestamp, and title), representing 610 users x 9,724 movies and 5,931,640 ratings.

3.2 Applying the system

In this part, the authors will develop two different types of recommender engines. Let's begin with simple RS content-based screening. Table 1 shows the samples from each dataset for enhanced understanding.

i. Content-Based Filtering

Datasets are well-organized and devoid of missing values. Using a content-based filtering approach on the movie lens dataset.

- Load Movie Lens dataset. Load the dataset with movie information, such as names, genres, and descriptions.
- Preprocess data: Clean and preprocess data as needed. This may include deleting unneeded columns or dealing with missing values.
- TF-IDF vectorization. Use a TF-IDF vectorization approach, such as the sklearn library's TfidfVectorizer, to convert movie descriptions or genres into numerical vectors.
- Create a TfidfVectorizer and provide any necessary settings, such as stop words or n-grams. Fit the vectorizer to movie descriptions or genres to learn language and calculate IDF weights. Transform the movie descriptions or genres into TF-IDF vectors using the fitted vectorizer, as indicated in Table 1.

Table (1) The result of the Cosine Similarity Calculation

title	Toy Story (1995)	...	Andrew Dice Clay: Dice Rules (1991)
title		...	
Toy Story (1995)	1.000000	...	0.267586
Jumanji (1995)	0.813578	...	0.000000
Grumpier Old Men (1995)	0.152769	...	0.570915
Waiting to Exhale (1995)	0.135135	...	0.505015
Father of the Bride Part II (1995)	0.267586	...	1.000000
...	...	...	...
Black Butler: Book of the Atlantic (2017)	0.680258	...	0.318581
No Game No Life: Zero (2017)	0.755891	...	0.354002
Flint (2017)	0.000000	...	0.000000
Bungo Stray Dogs: Dead Apple (2018)	0.421037	...	0.000000
Andrew Dice Clay: Dice Rules (1991)	0.267586	...	1.000000

[9742 rows x 9742 columns]

- Use sklearn's cosine similarity function to calculate pairwise cosine similarity between movie TF-IDF vectors. The cosine similarity function returns a similarity matrix in which each member indicates the content similarity of two movies.

ii. Hybrid filtering techniques using singular value decomposition

Using a hybrid filtering technique on the Movie Lens dataset.

- Load Movie Lens dataset. Load the dataset with movie ratings and other pertinent data, such as user IDs, movie IDs, and ratings.
- Preprocess data: Clean and preprocess data as needed. This might include deleting extraneous columns, addressing missing numbers, or standardizing the ratings.

-Separate the dataset into training and test sets. Divide the dataset into two parts: a training set for developing the recommendation models, and a test set for assessment. Create the user-item matrix. Create a user-item matrix, with rows representing people and columns representing movies. The matrix should contain user ratings for movies.

- The matrix should be sparse, with missing values reflecting unrated movies by users, as illustrated in Figure 4.

-To apply Singular Value Decomposition (SVD), a matrix factorization technique, the user-item matrix is divided into three matrices: U, Σ, and V^T.

U stands for user embedding, Σ is a diagonal matrix with singular values, and V^T represents movie embedding. Determine the amount of latent components to keep depending on your needs and domain expertise.

Table (2) The result of SVD Calculation with Five Priorities

```
[[ 0.0921447   0.00082937 -0.01089745 -0.06167385 -0.05555415]
 [-0.00149426 -0.01334201 -0.00442345  0.01773772 -0.0058663 ]
 [ 0.00738945  0.00196068  0.00171517 -0.00206861 -0.00135323]
 ...
 [ 0.04676093  0.08402888 -0.00976291 -0.01184704 -0.11611442]
 [-0.00806337  0.00138157 -0.03974124 -0.01378463 -0.00757944]
 [ 0.15783851  0.01658204  0.09267536  0.20218445 -0.13886488]]
 [  0.         170.42250831   0.           0.           0.
 [  0.           0.         191.1508762    0.           0.
 [  0.           0.           0.         231.23661142   0.
 [  0.           0.           0.           0.         534.41989777]]
   1.40608836e-04 -7.26189695e-04]
 [-1.68374052e-03  5.06728557e-02  3.74528571e-02 ... -4.90112220e-05
  -4.90112220e-05 -8.94587287e-04]
 [-7.84438842e-02 -5.68447103e-02 -1.80051145e-02 ...  8.71093879e-05
   8.71093879e-05 -1.22833344e-04]
 [-2.75911949e-02 -2.06662722e-03 -2.47146155e-02 ...  5.97586244e-04
   5.97586244e-04  1.27236200e-03]
 [-7.04498985e-02 -3.85393459e-02 -1.59129220e-02 ... -6.46836073e-05
  -6.46836073e-05 -2.71729303e-04]]
```

- Collaborative Filtering: Apply the SVD factors (U, Σ, V^T) to anticipate missing ratings in the user-item matrix. To forecast ratings for all user-movie pairs, utilize the dot product of the user embedding (U) and movie embedding (V^T). Adjust the anticipated ratings as needed, such as scaling them between certain ranges or rounding them to a specified accuracy. TABLE 2 illustrates this procedure visually.

Content-Based Filtering: To calculate similarity scores between movies, use content-based filtering techniques such as TF-IDF vectorization on movie attributes such as descriptions or genres. Create movie suggestions based on the similarity scores and user preferences.

- Combine recommendations: Combine expected ratings from collaborative and content-based filtering. Assign weights to each suggestion source depending on its significance or efficacy.

Create final recommendations: Combine the suggestions from collaborative and content-based filtering by computing the weighted average or using other fusion techniques. Sort the final recommendation ratings in descending order to get the top N suggested movies for each user.

4 Results and Discussion

The suggested system, which employs both content-based and hybrid filtering algorithms, has demonstrated encouraging results in terms of generating accurate and tailored suggestions. The system overcame issues including cold start, sparsity, and scalability by integrating the qualities of both systems. Present the results obtained by using content-based filtering based on item title similarities. The results of using the hybrid filtering approach, which combines content-based and collaborative filtering, demonstrate the benefit of combining both strategies to maximize their complementary strengths and produce better suggestions. The preceding dataset included 100,873 rows and 5 columns. The authors discovered that the majority of films get zero ratings using explanatory data analysis. That makes sense because most people see renowned or major popular movies, which will have a lot of reviews or ratings, as illustrated in Figure 4.

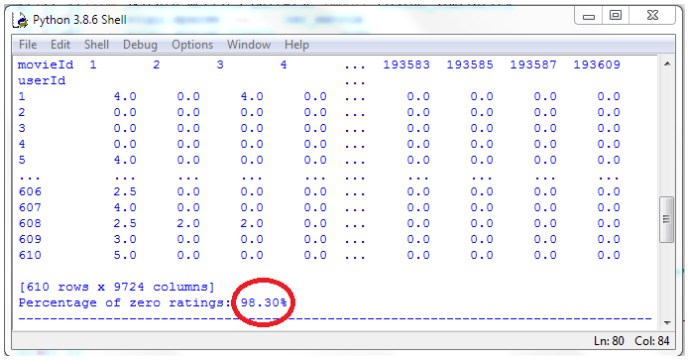


Figure (4) The user-item matrix.

The matrix has around 5,931,640 cells, with a sparsity level of approximately 98.3%. Finally, to improve prediction, the authors built the recommender engine employing memory-based calculation of the Cosine Similarity model-based CF using SVD. Content-based filtering using TF-IDF offers advantages such as independence from user data: Content-based filtering is not primarily reliant on user information or previous interactions. It can provide recommendations even for new or cold-start users with little or no prior experience with the system. Overcoming the Cold Start Problem: By depending on item attributes, content-based filtering can address the Cold Start problem in situations when there is minimal user data available. It can provide recommendations based on item similarities and user preferences derived from item characteristics. Limited Discovery of New Interests: Content-based filtering is mostly focused on item attributes that correspond to the user's existing choices. As a result, it may have limited capacity to discover and propose goods that do not align with the user's defined interests. Dependency on item metadata: Content-based filtering is strongly reliant on precise and detailed item metadata, such as descriptions or genres. If the item metadata is insufficient, incorrect, or does not reflect the item's genuine features, the quality of suggestions suffers. One disadvantage of content-based filtering is its inclination to promote items that are similar to those with which the user has previously engaged, resulting in a lack of diversity in recommendations. This can lead to a lack of diversity in recommendations, thereby narrowing the available item space. Content-based filtering confronts difficulties when proposing new or recently introduced items with insufficient user interactions or features. With little data to examine, it may struggle to make reliable suggestions for these things. Hybrid filtering using SVD provides several advantages, including enhanced recommendation. Accuracy: By combining the advantages of content-based and collaborative filtering, hybrid filtering can improve suggestion accuracy. Content-based filtering focuses on item attributes, whereas collaborative filtering takes into account user-item interactions, resulting in more precise and tailored suggestions. Collaborative filtering can address data sparsity concerns by exploiting user-item interactions. Even if a user has limited interaction or novel items have low ratings, collaborative filtering can still provide suggestions based on the behavior of comparable individuals or objects. Hybrid filtering can help both new users and new goods overcome cold start issues.

Content-based filtering can provide recommendations to new users based on item features, whereas collaborative filtering can make suggestions based on user-item interactions. Either has a downside, such as complexity or computational Cost: Implementing and maintaining a hybrid filtering system might be more difficult and expensive than using separate filtering approaches. The integration of content-based and collaborative filtering required increased processing and storage capacity. Model Selection and Parameter Tuning: Hybrid filtering entails choosing appropriate models and adjusting parameters for both content-based and collaborative filters. Determining the best mix of tactics and striking the correct balance between them may be difficult and requires experience. Cold Start for New Users and Products: While hybrid filtering can help to alleviate cold start issues, it may still have difficulty offering correct suggestions for completely new users or objects with insufficient data or features. The authors discovered that the suggested technique, when applied to small to medium-sized datasets, employs content-based filtering. However, when the dataset is vast and sparse, hybrid filtering is used. This strategy improves speed while also addressing scalability and data sparsity problems. The researchers' findings emphasize the recommendation system's flexibility to dataset features, successfully employing either content-based or hybrid filtering strategies to improve suggestion accuracy and overcome restrictions caused by dataset size and sparsity.

5 Conclusion and future work

This study investigates and presents several types of techniques usually used in the development of recommender engines, including content-based and hybrid filters. These techniques have been effectively utilized in a variety of areas, including financial services and popular platforms such as Netflix and Amazon. Throughout our investigation, the authors discussed and utilized content-based filtering algorithms that use item title similarities to provide meaningful suggestions. Furthermore, the author's explored hybrid filtering, which combines content-based and collaborative filtering techniques. This hybrid technique utilized into consideration additionally comparable item names but also user rating profiles to improve suggestion accuracy and customization. In addition to describing these strategies, the author has looked at some of the frequent issues found in recommender systems. Content-based filtering was used to solve the cold start problem, which occurs when there is minimal user or item data. The collaborative filtering strategy, which leverages the aggregate behavior of comparable users or objects, was used to reduce sparsity in user-item interactions. The authors also looked at scalability issues, because recommender systems must efficiently manage big datasets and rising user bases. Overall, the strategies investigated by the authors have shown helpful in producing accurate and tailored suggestions across a variety of disciplines. Using content-based and hybrid filtering, recommender engines can provide significant insights and improve user experiences. To ensure the long-term effectiveness and performance of recommender systems in real-world applications, certain difficulties such as cold start, sparsity, and scalability must be properly addressed. In the future, we hope to investigate approaches for reducing data sparsity, a recurrent difficulty in collaborative filtering. This involves looking at matrix factorization techniques such as singular value decomposition (SVD) and neighborhood-based algorithms to increase recommendation accuracy with minimal data. Furthermore, we will look into ways to increase suggestion diversity while keeping personalization. This might entail combining user demographics or implicit feedback data (browsing history, clicks, etc.) to discover larger user interests and hidden preferences. Finally, to analyze the scalability and generalizability of recommender systems, we will test their performance on larger and more varied datasets, evaluating features such as recommendation time and performance as the amount of data expands.

Conflict of Interest: The authors declare that there are no conflicts of interest.

References

Deldjoo, Y., Nazary, F., Ramisa, A., Mcauley, J., Pellegrini, G., Bellogin, A., & Noia, T. D. (2023). A review of modern fashion recommender systems. ACM Computing Surveys, 56(4), 1-37.

Reddy, S. V. G., Putchakayala Meher Sowjanya, A. P. K. R., Sai, B., Saketh, L. Y. K., & Reddy, K. V. A. (2023). MOVIE RECOMMENDATION SYSTEM USING COLLABORATIVE FILTERING.

Zahrawi, M., & Mohammad, A. (2021). Implementing recommender systems using machine learning and knowledge discovery tools. Knowledge-Based Engineering and Sciences, 2(2), 44-53.

Permana, A. H. J. P. J., & Wibowo, A. T. (2023). Movie Recommendation System Based on Synopsis Using Content-Based Filtering with TF-IDF and Cosine Similarity. International Journal on Information and Communication Technology (IJoICT), 9(2), 1-14.

Khan, E. M., Mukta, M. S. H., Ali, M. E., & Mahmud, J. (2020). Predicting users' movie preference and rating behavior from personality and values. ACM Transactions on Interactive Intelligent Systems (TiiS), 10(3), 1-25.

Chiny, M., Chihab, M., Bencharef, O., & Chihab, Y. (2022). Netflix Recommendation System based on TF-IDF and Cosine Similarity Algorithms. no. Bml, 15-20.

Vaz, P. C., Martins de Matos, D., Martins, B., & Calado, P. (2012, June). Improving a hybrid literary book recommendation system through author ranking. In Proceedings of the 12th ACM/IEEE-CS joint conference on Digital Libraries (pp. 387-388).

Rana, A., & Deeba, K. (2019, November). Online book recommendation system using collaborative filtering (with Jaccard similarity). In Journal of Physics: Conference Series (Vol. 1362, No. 1, p. 012130). IOP Publishing.

Dong, Z., Wang, Z., Xu, J., Tang, R., & Wen, J. (2022). A brief history of recommender systems. arXiv preprint arXiv:2209.01860.

Ko, H., Lee, S., Park, Y., & Choi, A. (2022). A survey of recommendation systems: recommendation models, techniques, and application fields. Electronics, 11(1), 141

Rani, U., & Bidhan, K. (2021). Comparative assessment of extractive summarization: textrank tf-idf and lda. Journal of Scientific Research, 65(1), 304-311.

Yunanda, G., Nurjanah, D., & Meliana, S. (2022). Recommendation system from microsoft news data using TF-IDF and cosine similarity methods. Building of Informatics, Technology and Science (BITS), 4(1), 277-284.

Attalariq, M., & Baizal, Z. K. A. (2023). Chatbot-Based Book Recommender System Using Singular Value Decomposition. Journal of Information System Research (JOSH), 4(4), 1293-1301.

Ankam, V. (2016). Big data analytics. Packt Publishing Ltd.

Baesens, B., & vanden Broucke, S. (2020, February 24). Singular Value Decomposition in Recommender Systems.DataMiningApps.https://www.dataminingapps.com/2020/02/singular-value-decomposition-in-recommender-systems/