International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 15, No. 1 (Feb-24)

http://dx.doi.org/10.12785/ijcds/150156

Matrix Factorization and Cosine Similarity Based Recommendation System For Cold Start Problem in E-Commerce Industries

Md Nadeem Noori [1], Jameel Ahamed [2] and Mumtaz Ahmed [3]

[1,2]*Department of Computer Science Information Technology, Maulana Azad National Urdu University, Hyderabad, India*
[3]*Department of Computer Science and Engineering, Jamia Millia Islamia, New Delhi, India*

Received 14 Jul. 2023 , Revised 27 Jan. 2024 , Accepted 28 Jan. 2024 , Published 1 Feb. 2024

Abstract: A recommendation system, often called a recommender system, is a kind of artificial intelligence (AI) algorithm that suggests or recommends products to both current and potential clients based on big data. It makes use of data to anticipate, target, and pinpoint what customers are looking for from an ever-expanding range of options. To identify them, a variety of indicators may be employed, such as past purchases, search history, demographic data, and other factors. It helps the users locate products and services for people that they are unable to locate to help them. At the initial level with a new customer, due to lack of knowledge, the Recommender System (RS) has a cold-start issue while making suggestions. Upon registering with the system, new users do not have access to any history of their choices or interactions. Without this information, the system is unable to offer customized recommendations. Furthermore, a newly introduced object to the system has no pre-existing interactions or preferences with other things. This poses a challenge for the algorithm to make recommendations based on user preferences. The cold start issue can limit the effectiveness of recommendation systems as they struggle to provide suggestions due to lack of information about individuals and products. This may result in users not returning to the system leading to a user experience. Recommendation systems adopt strategies like content-based filtering, collaborative filtering, hybrid systems, knowledge-based systems and demographic data to overcome the cold start problem. In this paper a method combining Cosine Similarity (CS) and Matrix Factorization (MF) is proposed as a solution, for addressing the cold start problem and further resolving sparsity challenges.

Keywords: Recommendation System, Collaborative Filtering, Cold-Start, Matrix-Factorization, Cosine Similarity, Sparsity

Introduction:

A. *Recommendation System:*

Recommender systems (RSs) are a subclass of AI systems that deliver users recommendations based on their past actions and preferences. RS attempts to offer precise and pertinent recommendations in order to improve the user experience and promote additional participation. In the current world, RS has a number of possible applications, such as social networking, online commerce, and streaming of audio and video. To identify possible future actions and interests, users' past actions and interests—including purchases, ratings, clicks, and search queries—are analysed [1].Three different types of recommender systems can be distinguished: content-based, collaborative filtering, and hybrid. The choice of recommendation system to utilize is influenced by the available data, the application's goals, and the demands of the users.

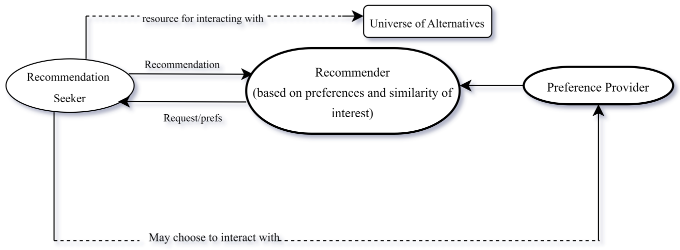

resource for interacting with
Universe of Alternatives
Recommendation Seeker
Recommendation
Request/prefs
Recommender (based on preferences and similarity of interest)
Preference Provider
May choose to interact with

Figure 1. Model of Recommendation Process

B. *Background:*

The origins of recommendation systems may be found in the early days of online shopping and data retrieval. Early recommendation systems used simple algorithms to suggest products based on factors like product popularity and common pairings. Collaborative filtering is one example of how technological progress has led to more nuanced recommendation systems that take into account users' shared interests and behaviours [2]. The advent of the Internet and online purchasing in the late 1990s and early 2000s likely prompted a rise in the use of recommendation systems at that time. Amazon and Netflix, two of the most popular online services, have contributed greatly to the surge in popularity of suggestions [3].Over time, the advent of big data and machine learning has allowed for a continuous rise in the complexity of recommendation systems. Recommendation systems in the future may handle enormous data sets and use advanced algorithms like deep learning and reinforcement learning to improve the quality of their recommendations and make them more relevant to the individual using this information [4].

C. Types of Recommendation System:

Various recommendation systems exist, each with its own advantages and disadvantages:

Content-Based Recommendation Systems: These systems provide suggestions based on the user's previous interactions with similar products or information. A content-based recommender system may, for instance, propose further rock music to a user who already often listens to that genre [5].

Collaborative Filtering Recommendation Systems: These algorithms provide suggestions based on the tastes and actions of customers who are like themselves. A collaborative filtering system may recommend a movie that one user enjoyed to another with comparable viewing preferences [6].

Hybrid Recommendation Systems: To provide more precise and individualised suggestions, these systems combine the capabilities of content-based and collaborative filtering techniques [7].

Matrix Factorization Recommendation Systems: In order to offer suggestions, these systems break down the user-item interaction matrix into latent user and item variables and then compare these factors to arrive at a consensus [8].

Deep Learning Recommendation Systems: Recommender systems like this simulate user-item interactions using deep neural networks to provide suggestions [9].

Reinforcement Learning Recommendation Systems: These systems create suggestions by using reinforcement learning methods, which learn from the user's input and behaviours over time [10] [11].

The choice of recommendation system depends on the specific requirements of the application, the availability of data, and the goals of the recommendation system.

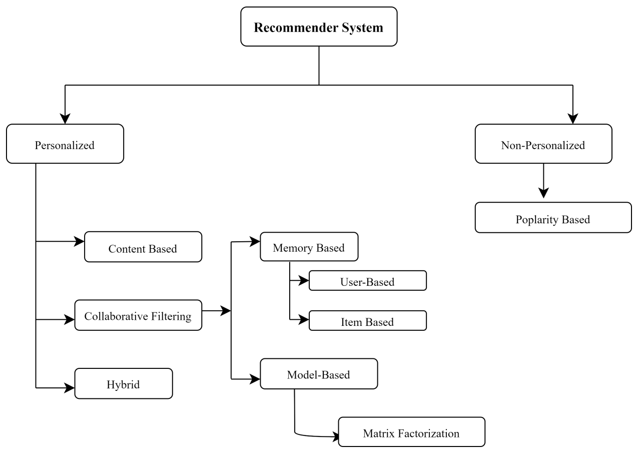


Figure 2. Recommender System's Type

Some systems may be more suitable for certain types of data, while others may be better suited for specific use cases or requirements.

Collaborative Filtering:

Collaborative recommendation systems are based on the premise that it is possible to forecast which items a certain user of the system would find attractive or interesting by analysing data on prior user behaviour or community views. Pure collaborative approaches have as their sole input a matrix of defined user-item ratings and often provide the following sorts of outputs: a prediction of how much the current user will love or hate a certain item, and a list of n recommended things. Methods of determining a user's tastes or opinions based on their similarities to other users' use patterns are known as similarity-based recommendation systems. There are two primary types of collaborative filtering: those that include users helping each other, and those that involve users helping each other with specific items [12].

User-user collaborative: To filter out irrelevant information, we look for people who are similar to our target user and provide recommendations based on the content that these users have shown interest in. Collaborative filtering works on the premise that people with similar tastes and interests would appreciate analogous content. Using techniques like Pearson Correlation and cosine Similarity to provide user-to-user collaborative filtering [13].

Item-item collaborative: In the process of filtering, comparable things to those that the target user has previously enjoyed are found, and suggestions are then made based on these similar items. According to this kind of collaborative filtering, users are more inclined to enjoy things that are similar to one another. Item-item collaborative filtering may be implemented with the help of techniques like Cosine Similarity and Jaccard Similarity [14] [15]. Collaborative filtering's scalability is a big plus since it doesn't need any domain expertise to provide useful recommendations. Collaborative filtering is especially useful for dealing with the ”cold start” issue, in which suggestions must be made for a user who has no past experience with the system [16]. But there are certain restrictions to collaborative filtering as well. For instance, it may fall victim to the ”popular items” bias, in which popular goods are suggested more often than the consumer would want. Furthermore, there may not be enough individuals with comparable likes for collaborative filtering to be successful. This might make accurate suggestions difficult or impossible for users with unusual or specific preferences. The measures by which a collaborative filtering system's efficiency is measured. Some commonly used metrics include accuracy, precision, recall, F1-score, and mean squared error [17].

Issues in Recommender System:

Data sparsity: Recommender systems frequently experience data sparsity, which is a dearth of information about consumer preferences for certain goods. As a result, some users may receive erroneous recommendations or maybe none.

Cold start issue: A recommender system may be unable to provide accurate suggestions when a new user or item has been introduced to the system. This is often referred to as the cold start problem.

Diversity and novelty: While recommender systems frequently choose to suggest well-liked goods, there is a dearth of diversity and novelty in the recommendations. Because of this, users may pass on items that would be of interest.

Privacy and Security: Issues might arise as a result of the frequent collection of large amounts of user data by recommender systems. User data needs to be safeguarded and handled properly at all times.

Fairness and bias: Based on variables like gender, colour, and age, recommender systems may be prejudiced and produce unjust recommendations. Having a fair and impartial design for the system is crucial.

Interpretability: Given how complicated the algorithms employed by recommender systems to provide recommendations can be, these systems can be challenging to understand. Users may become frustrated as a result, losing faith in the system.

Scalability: Recommender systems need to be able to handle large amounts of data and users, which can be a challenge in terms of processing power and storage capacity.

Matrix Factorization: Factoring a huge matrix into smaller matrices that represent the underlying structure and connections in the data is a common task in data mining and recommendation systems, and is accomplished by a process called matrix factorization. When discussing factorization in the context of recommendation systems, the user-item interaction matrix is the norm. This matrix contains user preferences and actions across a variety of objects. Matrix factorization is a technique where the original matrix is approximated by multiplying two other, lower-dimensional matrices, the user and item matrices. The user matrix reflects consumers' latent preferences across products, whereas the item matrix represents products' latent features [18]. By uncovering the latent user and item factors, matrix factorization may be utilised to create suggestions by pinpointing the things most comparable to those with which a user has previously engaged. Techniques like cosine similarity and Euclidean distance [19], may be used to calculate the degree of similarity between two things, and the most similar items can then be suggested to the user. Because of its scalability and its ability to identify intricate connections between consumers and products, matrix factorization has found widespread use in the field of recommendation systems. Matrix factorization may be accomplished in a number of ways, each with its own set of advantages and disadvantages. Common methods include SVD, NMF, and latent factor models. When used to recommendation systems, matrix factorization proves to be an effective method for making precise, user-specific suggestions in light of hidden correlations and patterns in the data [20].

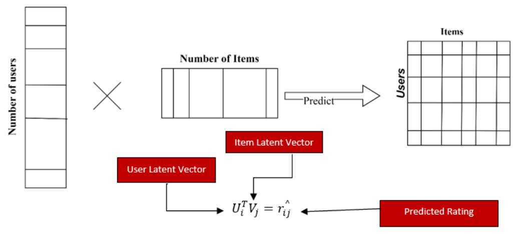


Figure 3. Matrix Factorization

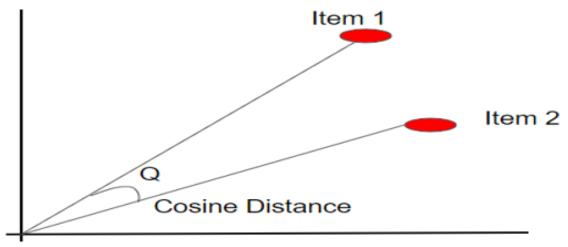
Idea Behind Matrix Factorization:

Data analysis, compression, and dimensionality reduction are just few of the many applications of matrix factorization (MF), a mathematical method used to partition a matrix into numerous matrices. The goal of MF is to locate a set of matrices that, when multiplied together, closely resemble the original matrix. The procedure entails finding a low-rank approximation of the original matrix by lowering the number of dimensions. Matrix factorization is a mathematical process that includes locating two matrices, U and V, whose product is close to the original matrix A. That is, A = U × V. The procedure of determining these matrices entails reducing the difference between the original matrix and the product of the component matrices. This may be performed using many approaches such as Singular

Value Decomposition (SVD), Principal Component Analysis (PCA), and Non-negative Matrix Factorization (NMF) [21].While these methods use various approaches to solving the issue of locating the component matrices, they all have the common goal of finding the most accurate low-rank approximation to the original matrix. After discovering the factor matrices, the information may be put to use in several applications, including recommendation systems, picture and audio compression, and statistical analysis. The factor matrices may be used to anticipate user preferences for items in recommendation systems based on their past ratings and interactions with the system.

Cosine Similarity: Two vectors in an n-dimensional space are said to have a high CS if they are very similar to one another. It is often used in IR and RS as a measurement of item or user similarity. When comparing two vectors in an n-dimensional space, the cosine similarity is determined by taking the cosine of the angle between them [22]. It may take on values between -1 and 1, with a value of 1 indicating that the vectors are identical, a value of 0 indicating that they are orthogonal and share no similarities, and a value of -1 indicating that they are very different. Cosine similarity is useful in recommendation systems since it quantifies the degree to which two users or two objects have common tastes or other attributes. In the user matrix, for instance, the cosine similarity between two latent vectors is near to 1 if the users have many preferences, such as a shared taste in movies [23], their latent vectors in the user matrix would be similar, and the cosine similarity between the vectors would be close to 1. Similarly, if two items have similar characteristics, their latent vectors in the item matrix would be similar [23], Similarly, if two items have similar properties, the cosine similarity between their latent vectors in the item matrix will be near to 1, signifying that the cosine of the similarity is close to 1. Because of its relative ease of usage and understanding, cosine similarity is often used in recommendation systems. It's also quite computationally efficient, thus it works well for massive recommendation networks. Nevertheless, it has several restrictions that make it less ideal for particular applications, such as being sensitive to the size of the vectors and not having a clearly defined distance measure. To sum up, cosine similarity is a helpful measure of similarity in recommendation systems, since it provides a simple approach to compare the preferences or features of various individuals or things and arrive at accurate and individualised suggestions.

Idea Behind Cosine Similarity: The CS is to measure the similarity between two vectors based on their orientation and direction, rather than their magnitudes. In text analysis, documents can be represented as vectors with each dimension corresponding to a word or phrase in the document. This makes it helpful for a range of applications where the magnitudes of the vectors may not be relevant. The document's frequency of words is represented by the vector's magnitude, and its distribution of words is indicated by its direction [24]. The cosine similarity measure can then be used to determine how similar two texts are based on the overlap in their word distributions. Similar to this, images can also be represented as vectors in image processing, where each dimension corresponds to a distinct pixel value. By analyzing the orientation of a pair of photos instead of their pixel magnitude, CS can be used to compare two images [25]. CS is helpful in determining the degree of similarity between two clients or items in terms of RS based on their feature vectors. A feature vector of each movie's genre, director, actors, and other pertinent characteristics, for instance, can be created if we have a dataset of user ratings for movies. Then, using cosine similarity, we may suggest films that are comparable to the ones a user has given good ratings in terms of their feature vectors [26].



$$similarity = Cos(\theta) = \frac{A.B}{||A||\,||B||} = \frac{\sum_{i=1}^{n} AiBi}{\sqrt{\sum_{i=1}^{n} Ai^2}\sqrt{\sum_{i=1}^{n} Bi^2}}$$

$$cosine = np.dot(A, B)/(norm(A) * norm(B))$$

Figure 4. Cosine Similarity

Unification of Matrix Factorization and Cosine Similarity:
MF and CS are commonly applied to model-based and memory-based collaborative filtering respectively. MF is a popular approach for model-based collaborative filtering, where it factors the user-item rating matrix into lower dimension matrices, such that the product of these matrices approximates the original rating matrix [27]. This approach is commonly used in RS to predict user ratings for unrated items. On the other hand, Cosine Similarity is a common technique used in memory-based collaborative filtering, where the similarities between users or items are computed using the CS measure. Based on the hypothesis that users with similar item rating habits would have similar preferences in the future, this method may aid in determining which products a user is most likely to find interesting [28]. By tailoring suggestions to individual users based on their prior interactions with the system, both methods may be utilised to enhance the reliability of a recommendation service.

Cold-Start Problem: It's not uncommon for recommendation systems to struggle with the "cold start" issue. It happens when the system doesn't know enough about a person or product to make an accurate prediction of their preferences or attributes. A new user may not have provided any information about their preferences; therefore, the system will be unable to provide useful suggestions. Since the system could not know anything about the features of a new product, it can't say anything about how well it will sell [29]. There are several ways to overcome with the cold-start problem in recommendation systems, including:

Content-based recommendation: Using this method, the system generates suggestions based on the objects' qualities and other accessible information. When there is a lot of data to work with, this method excels, but it may backfire if there isn't enough to go on or if the user's tastes don't accurately reflect the attributes of the objects [30].

Collaborative Filtering with Shrunken Nearest Neighbors: In order to offer suggestions, this method takes into account the similarities between users. The technology determines how similar the new user is to all other users and then offers suggestions based on the tastes of those users with whom they have the most in common [31].

Hybrid approaches: These methods use an integrative approach to making suggestions by using a number of different methods, including content-based and collaborative filtering. Content-based recommendations might be used for novel users or products, with collaborative filtering taking over when sufficient data is provided. In conclusion, there is ongoing work to address the cold-start problem that plagues recommendation systems. Application-specific considerations suggest that many methods may need to be used to adequately address the cold-start problem [32].

Factors that contribute to the cold-start problems:

Lack of interaction history: Algorithms have a hard time determining a user's or an item's preferences or interests when there is little or no interaction history [33].

Limit Item or User metadata: Metadata like descriptions, classifications, and qualities that might be utilised to create intelligent suggestions may be lacking or non-existent for new goods or users.

Sparsity of the data: Since customers interact with just a small number of items, the data matrices used in recommendation algorithms tend to be sparse. There isn't enough information to make informed recommendations for new markets or goods.Finding effective solutions to the cold-start problem is a substantial barrier to providing reliable recommendations, but it is a common challenge in recommendation systems overall [34].

Literature Review:

As the e-commerce and information technology industries have flourished since the advent of the Internet, so too has the need for consumers to quickly and efficiently navigate across vast information spaces. The recommendation system (RS) is essential in today's world of millions of daily internet users because it allows people to quickly and easily locate content that is tailored to their specific needs. To this purpose, the recommendation system has grown in prominence as a means of better serving end users. Content-based filtering (CBF) and collaborative filtering (CF) are the two most prominent developments in the field of recommendation systems (RS). The goods that CBF suggests to users are quite similar to those that the user has expressed an interest in before. Instead, CF makes suggestions to its users based on the tastes of their "neighbours," or others who use the service and share their interests. It is the core premise of CF that if two individuals have a viewpoint on one topic, they are more likely to have opposing views on other topics. An information filtering system, like a content recommender system, may be used as part of the information retrieval process to deliver an efficient and satisfactory user experience. As a result of their convenience and accessibility, online marketplaces have rapidly become an essential component of many industries. The recommender system enriches the consumer journey by highlighting previously unnoticed material and streamlining the decision-making process when faced with many options. As a result, the recommender system is crucial to the growth of the e-commerce and IT sectors, and its use is expanding into new fields [35]. A number of studies have been evaluated with the intention of proposing a remedy to this issue via the use of hyper-tuned Restricted Boltzmann Machines (RBM) to regenerate tabular data. Online retailers often use the hyper-tuned RBM, a deep learning-based system for making product suggestions. We use the Movies Lens, Film Trust, and Netflix rating datasets to evaluate the efficacy of hyper-tuned RBM in comparison to SVD, SVD++, Trust SVD, and Stack Auto Encoder (SAE) models on a number of big datasets. Finally, when compared to current recommender systems, the suggested hyper-tuned RBM model for suggestion has been shown to be both effective and reliable. In comparison to the SVD, SVD++, Trust SVD, and Stack Auto Encoder models, its findings indicate promise as a reliable recommender system-building option [36]. Companies like Big Mart and Mall rely heavily on accurate sales forecasting to ensure their continued success. Many businesses rely on statistical or other traditional forecasting methods to anticipate future sales, but these models may be time-consuming and difficult to utilise with non-linear data. To get around these restrictions, Machine Learning (ML) methods has been used, which can deal with both linear and non-linear data and efficiently process large amounts of data, like the Big Mart dataset, which includes a large number of customer and product data. To optimise the parameters and select the best hyperparameters, a Grid Search Optimization (GSO) technique has been used, and then combined with the Xgboost technique, which is commonly used for sales forecasting in retail companies. Comparing the effectiveness of two implementation strategies for collaborative filtering—memory-based and model-based—in Indonesia's burgeoning e-commerce sector, the findings demonstrated that the suggested model yielded superior outcomes compared to other models [37]. Offline testing and user-based testing were utilised to assess how well the sample data from PT X e-commerce performed. Model-based and memory-based collaborative filtering implementations, but fewer methodologies and fewer participants in user-based testing. It was advised that in future research, the number of respondents to be increased as well as the kind of responses they provide, all in an effort to better analyse the data. The most extensively used methods for learning user profiles are also highlighted, along with the classical and state-of-the-art strategies for expressing things and user profiles. User-generated material and accidental suggestions are also discussed [38],Using the POS-tagging mechanism available in the NLTK framework, a user profile model is built that outperforms current state-of-the-art recommender methods. Every time a user checks in, the system updates their tags and presents them with a list of recommendations based on what they've shown to be interested in [39], solves the issue of too much data by bringing together the strengths of deep neural networks and improved matrix factorization. By including both biases and a trustworthy user's neighbourhood, the matrix factorization model is robustly regularised. The interaction between rating and review-based latent components is extracted via Collaboration Matrix Factorization (CMF) through a projection approach and Convolutional Matrix Factorization (ConvMF). By concentrating on the long tail elements, the CMF model achieves far superior performance over competing models [40]. Further investigation into the conditions under which the prediction may fail, as well as the exploration of more complex informational or RS scheme combinations, could be inspired by the use of social trust data to improve the performance of a recommender system (RS) based on collaborative filtering (CF) methods, a similarity metric, and a link prediction method that accounts for the asymmetric nature of social relationships [41], using the collective wisdom of its users to infer its member's hidden tastes. Researchers are attempting to address issues such data sparsity, scalability, synonymy, privacy protection, etc. Metrics like as mean absolute error, recall, accuracy, ROC sensitivity, etc. are used to assess the efficacy of CF methods. By using social tags and contextual information, giving weights to each node, and defining enhanced thresholds for social trust, future CF approaches should be able to withstand shilling assaults and noise [42]. Trust-Sensitive Variable Decomposition, a Matrix Factorization Model, It applies a weighted-regularisation to further regularise the latent feature vectors and solves the problems of data sparsity and cold start in recommender systems by using rating and trust information from several sources [43]. In order to forecast how users would score an item, the ITRA model mines and uses this latent data. To find other people who are trustworthy, the TES algorithm is implemented, and their trust ratings are then utilised to calculate trust similarity. Predictions are made using the trust weighting method, which takes into account the implied level of trust [44]. Users with few social neighbours, near-cold-start users, pure-cold-start users, and long-tail items all benefit from the InSRMF model's ability to capture both individual tastes and social group characteristics at once. Additionally, the model's use of an ontological model of trust between users on a social network and a T-index to measure a user's trustworthiness is particularly useful [45]. Using the cold-start item issue and data-sparsity challenge in collaborative filtering, a matrix factorization-based recommendation model that considers item characteristics is developed: a FeatureMF. With the development of AI methods, RS has progressed and become more relevant, accurate, and robust. The model diffuses item latent factors derived from global ratings with latent factors for item features, named eTrust, formalises trust into multiple types, and leverages type-based dyadic and triadic correlations to predict trust relationships [46]. Despite the progress made in RS, software engineers still face the challenge of determining which model to focus their research efforts on. Bayesian optimization as a solution to the hyper-parameter optimization problem, exploration-exploitation balance of Bayesian optimization [47], technique used in the implementation of model-based collaborative filtering, impact of continuously re-selecting hyperparameters on the prediction accuracy, compare three approaches: a baseline approach (Fixed), an existing state-of-the-art approach (New Data Only), and their proposed approach (New Data New Params) [48]. Included in the criteria used to evaluate CF approaches are mean absolute error, recall and precision, and ROC sensitivity. Using overlapping community detection to mine implicit information in the social network and incorporating social influence into the model to increase recommendation accuracy, the SIER technique integrates both interactive information and social information into personal latent components learning [49]. This highlights the different taxonomies of RS and the importance of a trustworthy recommendation model, RS has evolved and become more relevant, accurate, and robust. Despite the progress made in RS, software engineers still face the challenge of determining which model to focus their research efforts on. Recommendation systems makes use of data mining techniques and prediction algorithm to determine a user's interests in information, items, and other interest [50]. Many research has been focusing on computer science, management, and medicine, and was conducted using keywords such as "Recommender systems, Recommendation systems, Movie Recommend, Music Recommend, Personalized Recommend, and Hybrid Recommend" [51], and by combining collaborative filtering with social behavior of users. The approach employs a computational model that makes predictions by calculating the weighted sum of ratings from socially related and similar-minded people. The growing amount of user and item, data to improve the reliability and efficiency, combination of product and user demographics data is used to address data sparsity and provide recommendations for cold-start users and items [52]. A computational model that predicts ratings based on the weighted sum of ratings from socially related people and similar-minded people from their rating patterns has been derived

Findings of Literature Review :
The cold-start problem in recommendation systems has been widely studied, and there is a significant body of research on the topic. However, there are still several research gaps in this area that need to be addressed. Some of the research gaps in the cold-start problem are:

Inadequate data: Recommendations that are not ideal can result from the fact that there is often insufficient data available to solve the cold-start issue. More efficient ways of utilizing the little data available to provide precise suggestions are required.

Personalization: Existing solutions to the cold-start issue often include impersonal or generalised strategies. There is a need for techniques that can tailor suggestions to each individual's tastes.

Integration with other data source: When data from other sources is not included into the recommendation system, the cold-start issue may become more severe. Tools that can efficiently combine information from several resources are needed to improve the quality of suggestions.

Scalability: Some methods for addressing the cold-start problem may not be scalable to large datasets, and this can limit their applicability in real-world applications. There is a need for methods that can handle large-scale datasets and make accurate recommendations in real-time.

Incorporating context: Many recommendation systems do not consider the context in which recommendations are made, and this can lead to sub-optimal recommendations. There is a need for methods that can incorporate context information to make more accurate recommendations.
These research gaps highlight the need for further research to address the cold-start problem and make more accurate recommendations in recommendation systems.

Purposed System and Research Methodology:
By combining the benefits of matrix factorization and cosine similarity, the suggested solution for the cold-start problem offers a hybrid recommendation method to overcome the difficulties associated with this issue. The stages of the proposed system are explained as below.

Matrix Factorization: Matrix factorization is used to reduce the dimensionality of the user-item interaction matrix to two. It is a common method used in recommendation systems, and its goal is to discover users' hidden likes and dislikes as well as the goods' hidden qualities by analysing their interactions with one another. Two matrices are generated as outputs of the factorization process; one represents the user's latent preferences, and the other represents the objects' latent attributes.

Cosine Similarity: With the matrices factored, the cosine similarity may be used to determine how similar a cold user is to other users. Cosine similarity, a vector similarity metric, may be used to group people with similar latent preferences together. When comparing two consumers, the cosine similarity is calculated by dividing the dot product of their latent preference vectors by the product of their vector norms.

Hybrid Recommendation: The suggested approach combines data from matrix factorization and cosine similarity to provide advice for dealing with the cold-start issue. Matrix factorization is used to get an estimate of the cold user's latent preferences, and then cosine similarity is used to locate individuals with similar tastes. The algorithm then integrates the findings from the two methods to provide the cold user with tailored suggestions. If a cold user's latent preferences are calculated by matrix factorization, the system may use those estimates to discover related things; from there, it can leverage the preferences of similar users, found via cosine similarity, to generate suggestions.

Evidence Candidate Selection: The suggested method additionally makes use of an evidence candidate selection technique to choose the most differentiating elements for each user, further enhancing the recommendations. The goal is to find things that accurately reflect a user's tastes so that you can confidently promote them to others. To produce ”recommendations for cold users,” the system may use an item selection algorithm to determine which products best match a user's tastes. Several factors, such item popularity, item variety, and item representativeness, may be used to build the evidence candidate selection technique.
To solve the cold-start issue in recommending, the suggested approach combines matrix factorization and cosine similarity as shown in figure 5 that the items matrix consisting of the item's behaviour data, on that item-item collaborating filtering (similarity between two items using cosine) has been performed which is then combined with review matrix of the item by the user. Based upon the users review for the items, whenever a new user enters the system, the system finds the behaviour of the new user by apply KNN approach. User conduct the study to assess the efficiency of the evidence, candidate selection approach, and the system's performance has been compared to that of competing recommendation systems using benchmark datasets. It is a strategy for reducing a matrix's dimensions by breaking it up into smaller, simpler matrices while still accurately capturing the organisation of the data. Matrix factorization has been used to approximate the missing values in the user-item interaction matrix, which is useful in the context of recommender systems.

Collaborative filtering provides a useful method for applying matrix factorization to the cold start issue by decomposing the user-item interaction matrix into two lower-dimensional matrices that respectively reflect the latent properties of the user and the item. Finally, we utilised these estimated characteristics to provide suggestions for new persons or things based on their known properties, such as demographic information or item features. A big enough interaction matrix is necessary for precise estimate of latent characteristics, although the method may be useful for discovering new consumers and products. where the attributes of the new item are compared to the features of previously rated items in the system and their cosine similarity is calculated. Recommendations for complementary products may be made using this similarity metric.

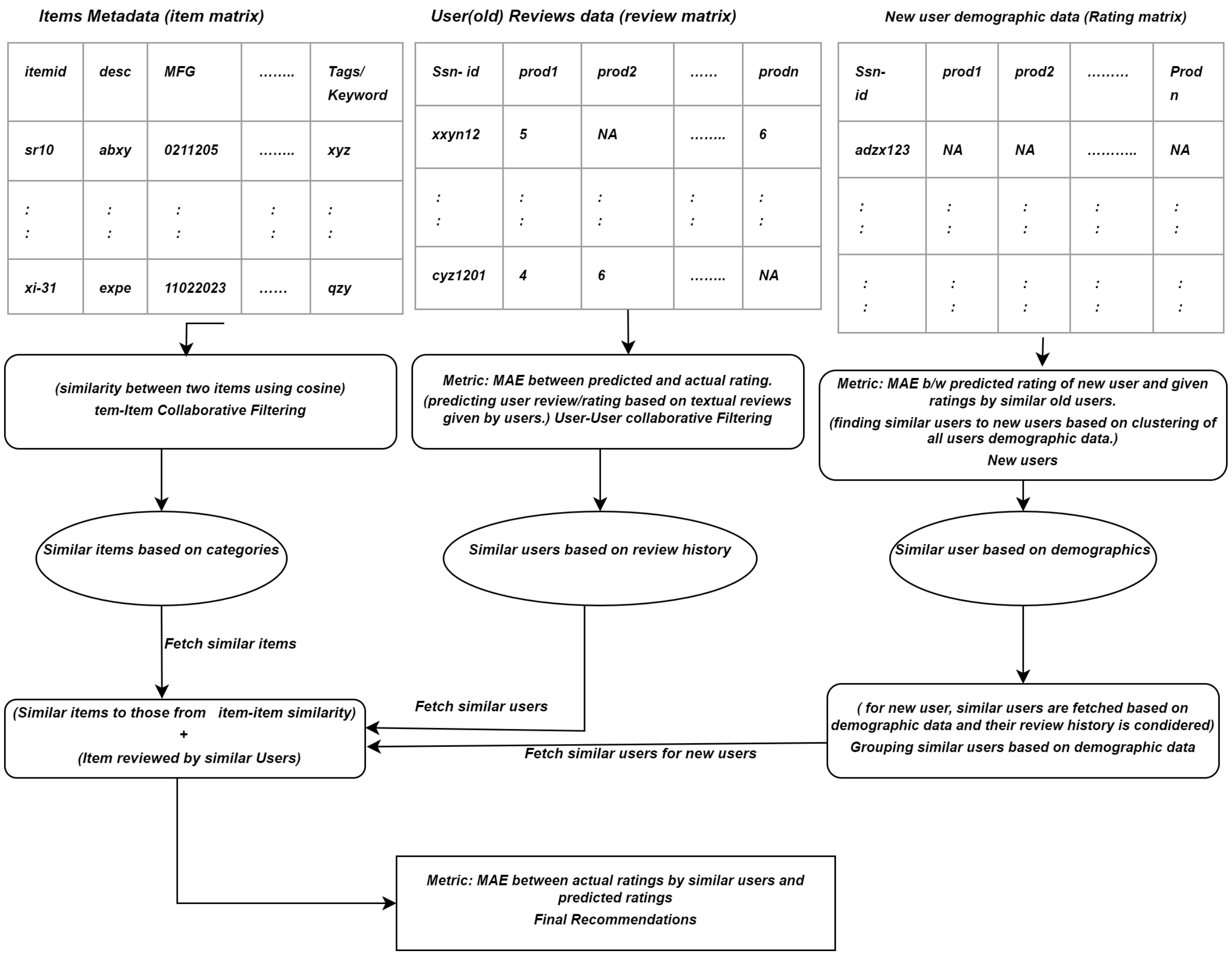
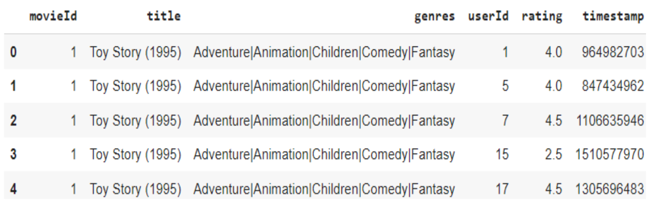

Items Metadata (item matrix)
itemid | desc | MFG | .... | Tags/ Keyword
sr10 | abxy | 0211205 | .... | xyz
xi-31 | expe | 11022023 | ...... | qzy
User(old) Reviews data (review matrix)
Ssn- id | prod1 | prod2 | ...... | prodn
xxyn12 | 5 | NA | .... | 6
cyz1201 | 4 | 6 | .... | NA
New user demographic data (Rating matrix)
Ssn- id | prod1 | prod2 | .... | Prod n
adzx123 | NA | NA | .... | NA
(similarity between two items using cosine) tem-Item Collaborative Filtering
Metric: MAE between predicted and actual rating. (predicting user review/rating based on textual reviews given by users.) User-User collaborative Filtering
Metric: MAE b/w predicted rating of new user and given ratings by similar old users. (finding similar users to new users based on clustering of all users demographic data.) New users
Similar items based on categories
Similar users based on review history
Similar user based on demographics
Fetch similar items
Fetch similar users
Fetch similar users for new users
(Similar items to those from item-item similarity) + (Item reviewed by similar Users)
(for new user, similar users are fetched based on demographic data and their review history is condidered) Grouping similar users based on demographic data
Metric: MAE between actual ratings by similar users and predicted ratings Final Recommendations

Figure 5. Proposed Model

The matrix factorization and similarity metric which is used in this proposed system has been explained in the result section of this paper.

Result:
The complete proposed system is divided into six stages which, (i) Dataset Collection, (ii) Data preprocessing and Cleansing, (iii) Development of Proposed Recommendation System, (iv) Use of Algorithms for model's development, (v) performance metrics used and (vi) final outcomes and results.

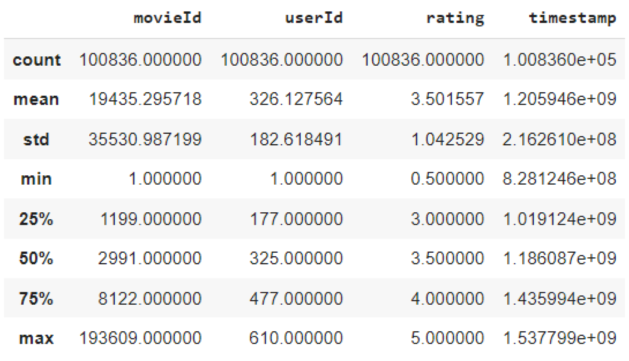
I. Dataset Collection: The MovieLens dataset is widely used since it comprises user reviews of movies from the MovieLens website, a movie recommendation service. The most popular variants of the dataset are the 100k, 1M, and 10M versions. The data set utilised here is ml-latest-small.zip, available for download at group-lens.org. There are a total of 900,000 ratings and 3,600 tags given to a total of 9,000 movies by 600 people. In the data-set folder, you will find two.csv files; the former is titled ”movies,” and the latter ”ratings.” This data analysis made use of the fields’movie-id’, ’title,’ ’genre,’ ’user-id,’ and ’rating.



	movieId	title	genres	userId	rating	timestamp
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1	4.0	964982703
1	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	5	4.0	847434962
2	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	7	4.5	1106635946
3	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	15	2.5	1510577970
4	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	17	4.5	1305696483

Figure 6. Dataset with Features

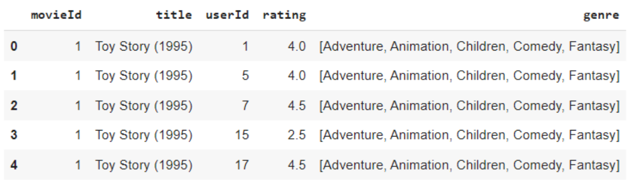
Descriptive statistics of the dataset:



	movieId	userId	rating	timestamp
count	100836.000000	100836.000000	100836.000000	1.008360e+05
mean	19435.295718	326.127564	3.501557	1.205946e+09
std	35530.987199	182.618491	1.042529	2.162610e+08
min	1.000000	1.000000	0.500000	8.281246e+08
25%	1199.000000	177.000000	3.000000	1.019124e+09
50%	2991.000000	325.000000	3.500000	1.186087e+09
75%	8122.000000	477.000000	4.000000	1.435994e+09
max	193609.000000	610.000000	5.000000	1.537799e+09

Figure 7. Description of dataset

II. Data Pre-Processing and Data Cleansing: When working with raw data, this is a crucial step. The column ”genres” in the MovieLens dataset has been mapped to the column ”genre,” and the column ”genres” has been removed.On these datasets, we applied the Surprise Library to create the Surprise dataset.
Splitting of data: After data preprocessing, the surprise dataset has been split into a trainset and a testset using the train_test_split() module of Scikit Learn to train the model and evaluate it.



	movieId	title	userId	rating	genre
0	1	Toy Story (1995)	1	4.0	[Adventure, Animation, Children, Comedy, Fantasy]
1	1	Toy Story (1995)	5	4.0	[Adventure, Animation, Children, Comedy, Fantasy]
2	1	Toy Story (1995)	7	4.5	[Adventure, Animation, Children, Comedy, Fantasy]
3	1	Toy Story (1995)	15	2.5	[Adventure, Animation, Children, Comedy, Fantasy]
4	1	Toy Story (1995)	17	4.5	[Adventure, Animation, Children, Comedy, Fantasy]

Figure 8. Data cleansing

```
# Split into train and test set
trainset,testset =train_test_split(data,test_size=0.2,random_state=40)
```

III. Development of proposed Reccomendation System: The algorithms employed for the development are as:

KNN (K-Nearest Neighbors): Technique used to create similarity matrix that measures the similarity between items, andd then use it to find K most similar items. similarity between items using a distance metric such as cosine similarity or Pearson correlation coefficient.

$$p(A,B) = \frac{(\sum(A-\mu A))\ (\sum(B-\mu B))}{(\ \sqrt{(\sum(A-\mu A)^2} * \sqrt{(\sum(A-\mu A)^2)}}$$

Where,
A,B $\rightarrow variablebeingcorrelated.$
$\mu A, \mu B \rightarrow Respectivemeans.$
$p(A,B) \rightarrow PearsonCorrelation.$

SVD (Singular Value Decompostion):

It is a well-known method in linear algebra for exploring the connections between objects by decomposing a matrix into the product of many smaller matrices.

$$M = U \sum V^t$$

Where,
M $\rightarrow original\ matrix\ to\ decompose.$
$U \rightarrow is\ left\ singular\ matrix(columns\ are\ left\ singular\ vectors) contains\ eigenvectors\ of\ matrix\ MM^t.$
$V \rightarrow is\ right\ singular\ matrix(columns\ are\ right\ singular\ vectors) contains\ eigenvectors\ of\ matrix\ M^tM.$

SVD++ (Singular Value Decomposition++):

Extends the popular collaborative filtering method SVD. It considers both explicit (user ratings) and implicit (user-interacted or bought things) feedback. This improves suggestions, particularly when explicit input is scarce. SVD++ does this by adding user and item biases to matrix factorization. Traditional SVD learns these characteristics with user and item latent factors. SVD++ is a strong method for constructing personalised recommender systems that can accept explicit and implicit user input.

$$R_{nm} = \mu + b_n + b_m + q_m^T(P_n + |N(n)|^{-0.5} \sum_{p \in N(n)} Y_p$$

Where,
R $\rightarrow$ $predicted\ rating\ of\ user\ 'n'\ onitem\ 'm'.$
$\mu \rightarrow the\ overall\ meanrating\ in\ the\ entire\ dataset.$
$b_n \rightarrow the\ bias\ term\ for\ user\ 'n'.$
$b_m \rightarrow the\ bias\ term\ for\ user\ 'm'.$
$q_m \rightarrow the\ item\ feature\ vector\ for\ item\ 'm'.$
$p_n \rightarrow the\ user\ feature\ vector\ for\ user\ 'n'.$
$N(n) \rightarrow represents\ the\ set\ of\ items\ rated\ by\ user\ 'n'.$
$|N(n)| \rightarrow the\ number\ of\ iems\ in\ N(n).$
$Y_p \rightarrow the\ implicit\ feedback\ vector\ for\ itme\ 'p'.$

NMF (Non-negative Matrix Factorization):

It is a prominent recommender system approach. It includes splitting a huge matrix of user-item interactions into two smaller matrices, one representing people and the other items. Multiplying these matrices yields a projected rating for each user-item pair. NMF's main benefit over other methods is that it imposes non-negativity requirements on the user and item matrices. This guarantees that the forecasts are always non-negative, which might be beneficial in situations where negative evaluations don't make sense, such as product recommendations. This algorithm is very similar to SVD. The prediction is set as:

$$r_{ui} = q_i^T p_u$$

where user and item factors are kept positive, if biased version is available by setting the biased parameter to True. In that scenario, the prediction is set as:

$$r_{ui} = \mu + b_u + b_i + q_i^t P_u$$

IV. Use of Algorithms for model's development: The metrics used to measure the performance of the system are as follows:

a. Root Mean Square Error (RMSE): It is a regularly used statistic for assessing the accuracy of a recommendation system. It calculates the discrepancy between expected and actual user ratings or scores for a group of objects. The square root of the average of the squared discrepancies between the expected and actual ratings across all users and objects in the dataset can be used to compute the RMSE in the context of recommendation systems. It offers a gauge of how effectively the recommendation engine can correctly forecast user interest.

Root mean Square Error:

$$\sqrt{\sum_{i=1}^{n} \frac{(Y_i^{-} - Y_i)^2}{n}}$$

b. Mean Square Error (MSE): Recommendation systems frequently employ this assessment metric to gauge the accuracy of anticipated ratings. It is determined by calculating the squared difference between the test set's expected and actual ratings, and averaging the values across all user-item combinations. The MSE calculates the average squared difference between a group of items' actual and expected ratings when used in recommendation systems. The performance of the recommendation system improves with a decrease in MSE. It has been used to compare various recommendation algorithms or to adjust their parameters.

Mean Square Error:

$$\frac{1}{n}\sum_{i=1}^{n}(Y_i^{-} - Y_i)^2$$

c. Mean Absolute Error (MAE): When used with recommendation systems, MAE calculates how much the actual rating deviates from the anticipated rating on a scale of 0 to 5 (or any other rating scale that is being utilised). It is determined by averaging out the absolute differences between all of the test set's items' expected and actual ratings.

Mean Absolute Error:

$$\frac{1}{n}\sum_{i=1}^{n}|Y_i^{-} - Y_i|$$

d. Fraction of Concordant Pairs (FCP): It is a standard indicator for assessing how accurate recommendation systems are. The frequency with which users score suggested things higher than non-recommended items in a recommendation system is measured by FCP. To put it another way, FCP determines the percentage of times the anticipated ranking of suggested products matches the real ranking by comparing the predicted rankings to the actual user rankings. In order to obtain a fuller view of the system's performance, FCP should be used in conjunction with other metrics. FCP is only one of several measures that may be used to evaluate recommendation systems. FCP:

$$\frac{(Number\,of\,Concordant\,pairs)}{Number\,of\,concordant\,pairs) + (Number\,of\,discordant\,pairs)}$$

where a concordant pair is a pair of items for which the predicted ranking is the same as the actual ranking, and a discordant pair is a pair of items for which the predicted ranking is different from the actual ranking.

v. Final Outcome and Results: The performance of the algorithms employed in this work is depicted in table I. This section includes all results obtained from the simulations on various performance metrics. Different algorithm like KNN, SVD, SVD++ and NMF etc. were applied in various scenerios and test the performance of each algorithm. The performance measures of the proposed module are tabulated in figure [9], which is being shown, the parameter value of k as set to 40.

Similarity(user_based&item_based)	**Similarity metrics**	**KNN_Algorithm**	**Performance Metrics**			
			MSE	**MAE**	**RMSE**	**FCP**
User_based= ‘False’	**Cosine**	KnnBasic	0.8087	0.6968	0.8993	0.5144
		KnnWithMeans	0.5132	0.5454	0.7164	0.7726
		KnnBaseline	0.6171	0.6001	0.7855	0.7107
		KnnWithZScore	0.5015	0.5351	0.7082	0.7766
User_based = ‘True’	**Cosine**	KnnBasic	0.7225	0.6440	0.8500	0.7514
		KnnWithMeans	0.5934	0.5787	0.7703	0.7524
		KnnBaseline	0.5724	0.5677	0.7566	0.7577
		KnnWithZScore	0.5867	0.5714	0.7660	0.7518
User_based = ‘False’	**Pearson**	KnnBasic	0.4930	0.5423	0.7022	0.6970
		KnnWithMeans	0.2614	0.3785	0.5113	0.8822
		KnnBasline	0.3144	0.4241	0.5607	0.8411
		KnnWithZScore	0.2338	0.3563	0.4835	0.8924
User_based = ‘True’	**Pearson**	KnnBasic	0.4719	0.5165	0.6869	0.8241
		KnnWithMeans	0.3734	0.4488	0.6111	0.8326
		KnnBaseline	0.3623	0.4430	0.6019	0.8333
		KnnWithZScore	0.3639	0.4411	0.6032	0.8358
User_based = ‘False’	**MSD**	KnnBasic	0.2026	0.2758	0.4501	0.9886
		KnnWithMeans	0.3157	0.4283	0.5619	0.8777
		KnnBaseline	0.2166	0.3290	0.4654	0.9543
		KnnWithZScore	0.3010	0.4153	0.5486	0.8847
User_based = ‘True’	**MSD**	KnnBasic	0.5387	0.5511	0.7340	0.7900
		KnnWithMeans	0.4933	0.5247	0.7023	0.7934
		KnnBaseline	0.4710	0.5118	0.6863	0.7947
		KnnWithZScore	0.4879	0.5185	0.6985	0.7949
User_based = ‘False’	**Pearson_baseline**	KnnBasic	0.1367	0.2694	0.3697	0.9475
		KnnWithMeans	0.1127	0.2426	0.3356	0.9372
		KnnBaseline	0.1087	0.2340	0.3298	0.9554
		KnnWithZScore	0.1095	0.2388	0.3310	0.9402
User_based = ‘True’	**Pearson_baseline**	KnnBasic	0.1489	0.2714	0.3858	0.9068
		KnnWithMeans	0.1201	0.2373	0.3465	0.9243
		KnnBaseline	0.1155	0.2331	0.3398	0.9208
		KnnWithZScore	0.1178	0.2350	0.3433	0.9243
SVD parameters →	n_factors = 100, n_epochs = 20, lr_all = 0.005, reg_all = 0.02	SVD	0.4066	0.4961	0.6376	0.8203
SVD++ parameters →	n_factors = 100, n_epochs = 20, lr_all = 0.005, reg_all = 0.02	SVDpp	0.3553	0.4641	0.5961	0.8424
NMF(Non-negative Matrix Factorization)			0.4101	0.4693	0.6404	0.8233

Figure 9. Performance Table of 1M MovieLens dataset

The results obtained in this work are compared with the existing results depicted in the research paper [53] with the same dataset (1M dataset) are shown in Figure 9.

Name	RMSE	MAE	MSE	FCP
SVD	0.873	0.686	MSE and FCP are not available in[36].	
NMF	0.916	0.724		
KNN	0.923	0.727		
Cantered KNN	0.909	0.719		
Baseline	0.909	0.719		
KNN-Improved	0.870	0.683		
Proposed Knn Baseline With (User_based = “False”) and (similarity metric) Pearson_baseline.	**0.3298**	**0.2340**	**0.1087**	**0.9372**

Figure 10. Comparative Performance

As seen in the Figure 9,10. it is evident that the purposed model performs well on same 1M movie lens dataset with RMSE, MAE, FCP and MSE, when User_based = “False” with pearson_baseline similarity.

Conclusion and Discussion :

In order to address the problems with the cold-start problem for users, the outcome of the suggested system utilizing matrix factorization and cosine similarity has been applied. The method is made to get over the drawbacks of conventional recommendation systems, which find it difficult to provide consumers with reliable recommendations when there is minimal interaction data available. The proposed system estimates a cold user's preferences with many objects by combining the advantages of matrix factorization and cosine similarity, allowing it to provide recommendations that are more suited to the user's requirements. This helps to prevent the widespread issue of suggesting well-liked products to every user, which may lead to a deficiency in customization. The study offered in the proposed method also focuses on fixing the issue of spars matrix or sparsity, by fusing cosine similarity with matrix factorization technique. The research found that the suggested technique has the ability to boost the performance of recommendation systems even when dealing with limited information. Parameter tweaking is being used to evaluate the performance of memory-based CF and model-based CF, and the findings demonstrated that the sparsity issue in CF may be successfully tackled by combining cosine similarity and matrix factorization. In future work, it can be implemented with a large dataset and with algorithm like LSTM etc.

References

[1] L. Lü, M. Medo, C. H. Yeung, Y.-C. Zhang, Z.-K. Zhang, and T. Zhou, “Recommender systems,” *Physics reports*, vol. 519, no. 1, pp. 1–49, 2012.

[2] N. Vaidya and A. Khachane, “Recommender systems-the need of the ecommerce era,” in *2017 International Conference on Computing Methodologies and Communication (ICCMC).* IEEE, 2017, pp. 100–104.

[3] C. A. Gomez-Uribe and N. Hunt, “The netflix recommender system: Algorithms, business value, and innovation,” *ACM Transactions on Management Information Systems (TMIS)*, vol. 6, no. 4, pp. 1–19, 2015.

[4] L. Terveen and W. Hill, “Beyond recommender systems: Helping people help each other,” *HCI in the New Millennium*, vol. 1, no. 2001, pp. 487–509, 2001.

[5] A. Sarnobat and D. Kalola, “A survey on recommender systems,” *IJSRP, p p9356*, 2019.

[6] L. Jerrold, “Is seeing believing?” *American Journal of Orthodontics and Dentofacial Orthopedics*, vol. 142, no. 6, pp. 888–889, 2012.

[7] N. Manouselis, H. Drachsler, R. Vuorikari, H. Hummel, and R. Koper, “Recommender systems in technology enhanced learning,” *Recommender systems handbook*, pp. 387–415, 2011.

[8] K. RaviKanth, K. ChandraShekar, K. Sreekanth, and P. S. Kumar, “Recommendation system for e-commerce by memory based and model based collaborative filtering,” in *Proceedings of the 11th International Conference on Soft Computing and Pattern Recognition (SoCPaR 2019) 11.* Springer, 2021, pp. 123–129.

[9] R. H. Singh, S. Maurya, T. Tripathi, T. Narula, and G. Srivastav, “Movie recommendation system using cosine similarity and knn,” *International Journal of Engineering and Advanced Technology*, vol. 9, no. 5, pp. 556–559, 2020.

[10] N. Idrissi and A. Zellou, "A systematic literature review of sparsity issues in recommender systems," *Social Network Analysis and Mining*, vol. 10, pp. 1–23, 2020.

[11] N. Mishra, S. Chaturvedi, A. Vij, and S. Tripathi, "Research problems in recommender systems," in *Journal of Physics: Conference Series*, vol. 1717, no. 1. IOP Publishing, 2021, p. 012002.

[12] S. Cleger-Tamayo, J. M. Fernández-Luna, and J. F. Huete, "A new criteria for selecting neighborhood in memory-based recommender systems," in *Conference of the Spanish Association for Artificial Intelligence*. Springer, 2011, pp. 423–432.

[13] J. B. Schafer, J. A. Konstan, and J. Riedl, "Meta-recommendation systems: user-controlled integration of diverse recommendations," in *Proceedings of the eleventh international conference on Information and knowledge management*, 2002, pp. 43–51.

[14] M. Y. H. Al-Shamri, "Power coefficient as a similarity measure for memory-based collaborative recommender systems," *Expert Systems with Applications*, vol. 41, no. 13, pp. 5680–5688, 2014.

[15] T. C. T. Tran, L. P. Phan, and H. X. Huynh, "Energy-based collaborative filtering recommendation," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 7, 2022.

[16] E. Varga, *Practical data science with python 3: synthesizing actionable insights from data*. Apress, 2019.

[17] M. Gupta, A. Thakkar, V. Gupta, D. P. S. Rathore *et al.*, "Movie recommender system using collaborative filtering," in *2020 international conference on electronics and sustainable communication systems (ICESC)*. IEEE, 2020, pp. 415–420.

[18] H. Zhang, I. Ganchev, and N. S. Nikolov, "Matrix factorization enriched with item features," in *2020 International Conference on Mathematics and Computers in Science and Engineering (MACISE)*. IEEE, 2020, pp. 77–80.

[19] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, "Recommender systems survey," *Knowledge-based systems*, vol. 46, pp. 109–132, 2013.

[20] Z. Wu, H. Liu, Y. Xu, and L. Jing, "Collaboration matrix factorization on rate and review for recommendation," *Journal of Database Management (JDM)*, vol. 30, no. 2, pp. 27–43, 2019.

[21] A. Noulapeu Ngaffo and Z. Choukair, "A deep neural network-based collaborative filtering using a matrix factorization with a twofold regularization," *Neural Computing and Applications*, vol. 34, no. 9, pp. 6991–7003, 2022.

[22] D. Pancholi and C. Selvi, "Study of cold-start product recommendations and its solutions," in *International Conference on Data Management, Analytics & Innovation*. Springer, 2023, pp. 45–58.

[23] A. R. Lahitani, A. E. Permanasari, and N. A. Setiawan, "Cosine similarity to determine similarity measure: Study case in online essay assessment," in *2016 4th International Conference on Cyber and IT Service Management*. IEEE, 2016, pp. 1–6.

[24] R. Dehak, J. Glass, D. Reynolds, and P. Kenny41, "Cosine similarity scoring without score normalization techniquesnajim dehak1."

[25] N. Dehak, R. Dehak, J. R. Glass, D. A. Reynolds, P. Kenny *et al.*, "Cosine similarity scoring without score normalization techniques." in *Odyssey*, 2010, p. 15.

[26] A. Zarghami, S. Fazeli, N. Dokoohaki, and M. Matskin, "Social trust-aware recommendation system: A t-index approach," in *2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, vol. 3. IEEE, 2009, pp. 85–90.

[27] M.-P. T. Do, D. Nguyen, and L. Nguyen, "Model-based approach for collaborative filtering," in *6th International conference on information technology for education*, 2010, pp. 217–228.

[28] F. Khawar and N. L. Zhang, "Cleaned similarity for better memory-based recommenders," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2019, pp. 1193–1196.

[29] M. F. Aljunid and M. D. Huchaiah, "An efficient hybrid recommendation model based on collaborative filtering recommender systems," *CAAI Transactions on Intelligence Technology*, vol. 6, no. 4, pp. 480–492, 2021.

[30] S. Ghazarian and M. A. Nematbakhsh, "Enhancing memory-based collaborative filtering for group recommender systems," *Expert systems with applications*, vol. 42, no. 7, pp. 3801–3812, 2015.

[31] J. Lu, D. Wu, M. Mao, W. Wang, and G. Zhang, "Recommender system application developments: a survey," *Decision support systems*, vol. 74, pp. 12–32, 2015.

[32] R. Reshma, G. Ambikesh, and P. S. Thilagam, "Alleviating data sparsity and cold start in recommender systems using social behaviour," in *2016 International Conference on Recent Trends in Information Technology (ICRTIT)*. IEEE, 2016, pp. 1–8.

[33] J. B. Schafer, J. Konstan, and J. Riedl, "Recommender systems in e-commerce," in *Proceedings of the 1st ACM conference on Electronic commerce*, 1999, pp. 158–166.

[34] G. Guo, "Resolving data sparsity and cold start in recommender systems," in *User Modeling, Adaptation, and Personalization: 20th International Conference, UMAP 2012, Montreal, Canada, July 16-20, 2012. Proceedings 20*. Springer, 2012, pp. 361–364.

[35] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Advances in artificial intelligence*, vol. 2009, 2009.

[36] G. K. Jha, M. Gaur, P. Ranjan, and H. K. Thakur, "A trustworthy model of recommender system using hyper-tuned restricted boltzmann machine," *Multimedia Tools and Applications*, vol. 82, no. 6, pp. 8261–8285, 2023.

[37] G. Behera and N. Nain, "Grid search optimization (gso) based future sales prediction for big mart," in *2019 15th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)*. IEEE, 2019, pp. 172–178.

[38] W. Fan, Z. Zhao, J. Li, Y. Liu, X. Mei, Y. Wang, J. Tang, and Q. Li, "Recommender systems in the era of large language models (llms). arxiv 2023," *arXiv preprint arXiv:2307.02046*, 2023.

[39] S. Amara and R. R. Subramanian, "Collaborating personalized recommender system and content-based recommender system using textcorpus," in *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*. IEEE, 2020, pp. 105–109.

[40] B. G. Galuzzi, I. Giordani, A. Candelieri, R. Perego, and F. Archetti, "Hyperparameter optimization for recommender sys-

tems through bayesian optimization," *Computational Management Science*, vol. 17, pp. 495–515, 2020.

[41] M. Dadgar and A. Hamzeh, "How to boost the performance of recommender systems by social trust? studying the challenges and proposing a solution," *IEEE Access*, vol. 10, pp. 13 768–13 779, 2022.

[42] G. Bathla, H. Aggarwal, and R. Rani, "A graph-based model to improve social trust and influence for social recommendation," *The Journal of Supercomputing*, vol. 76, pp. 4057–4075, 2020.

[43] G. Guo, J. Zhang, and N. Yorke-Smith, "A novel recommendation model regularized with user trust and item ratings," *ieee transactions on knowledge and data engineering*, vol. 28, no. 7, pp. 1607–1620, 2016.

[44] Y. Li, J. Liu, J. Ren, and Y. Chang, "A novel implicit trust recommendation approach for rating prediction," *IEEE Access*, vol. 8, pp. 98 305–98 315, 2020.

[45] Y. Wei, H. Ma, and R. Zhang, "Social influence-based personal latent factors learning for effective recommendation," *Advances in Computational Intelligence*, vol. 2, pp. 1–12, 2022.

[46] Y. Cen, J. Zhang, G. Wang, Y. Qian, C. Meng, Z. Dai, H. Yang, and J. Tang, "Trust relationship prediction in alibaba e-commerce platform," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 5, pp. 1024–1035, 2019.

[47] P. M. Alamdari, N. J. Navimipour, M. Hosseinzadeh, A. A. Safaei, and A. Darwesh, "A systematic study on the recommender systems in the e-commerce," *Ieee Access*, vol. 8, pp. 115 694–115 716, 2020.

[48] S. Chan, P. Treleaven, and L. Capra, "Continuous hyperparameter optimization for large-scale recommender systems," in *2013 IEEE international conference on big data*. IEEE, 2013, pp. 350–358.

[49] H. Zhang, I. Ganchev, N. S. Nikolov, Z. Ji, and M. O'Droma, "Featuremf: an item feature enriched matrix factorization model for item recommendation," *IEEE Access*, vol. 9, pp. 65 266–65 276, 2021.

[50] G. K. Jha, M. Gaur, P. Ranjan, and H. K. Thakur, "A survey on trustworthy model of recommender system," *International Journal of System Assurance Engineering and Management*, vol. 14, no. Suppl 3, pp. 789–806, 2023.

[51] D. Roy and M. Dutta, "A systematic review and research perspective on recommender systems," *Journal of Big Data*, vol. 9, no. 1, p. 59, 2022.

[52] J. Yu, H. Yin, X. Xia, T. Chen, J. Li, and Z. Huang, "Self-supervised learning for recommender systems: A survey," *IEEE Transactions on Knowledge and Data Engineering*, 2023.

[53] G. Li and J. Zhang, "Music personalized recommendation system based on improved knn algorithm," in *2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*. IEEE, 2018, pp. 777–781.

Md Nadeem Noori has done his Master of Technology from Maulana Azad National Urdu University, Hyderabad, India. He has completed Bachelore of Engineering from Anna University Chennai, India. Currently he is working as a guest faculty at MANUU Hyderabad. His research area includes Machine Learning, Natural Language Processing.
E-mail: *mdnadeemnoori@gmail.com*

Jameel Ahamed has done his Doctor of Philosophy (Ph.D.) from the National Institute of Technology Srinagar, JK (NIT Srinagar). He has completed Bachelor of Technology (B.Tech.) from the Faculty of Engineering, Jamia Millia Islamia, New Delhi and Master of Technology (M.Tech.) from the National Institute of Technology, Srinagar, Jammu and Kashmir (NIT Srinagar). Before joining the School of Technology, MANUU, Hyderabad, as an Assistant Professor, he worked as Guest Faculty in the Department of Electronics Engineering, Aligarh Muslim University, Aligarh, U.P. for more than a year. He coordinated two Faculty Development Programmes sponsored by DST and UGC. He also completed one Research Project of 01.05 Lakh INR. He has more than 15 research publications to his credit and delivered many invited talks. He has visited countries like Czech Republic and United Kingdom for paper presentation in the conferences. He has supervised 09 Master of Technology (M.Tech.) and several Bachelor of Technology (B.Tech.) and MCA research projects. His research areas include the Internet of Things, Computer Networks, Machine Learning and Deep Learning.
E-mail: *jameel.shaad@gmail.com*

Dr. Mumtaz Ahmed has done his Doctor of Philosophy (Ph.D.) from the Faculty of Engineering, Jamia Millia Islamia, New Delhi. He is working as Associate Professor in the Department of CSE, Jamia Millia Islamia, New Delhi. His research areas include the Internet of Things, Computer Networks, Machine Learning and Deep Learning.
E-mail: *mahmed1@jmi.ac.in*