Hindawi
Applied Computational Intelligence and Soft Computing
Volume 2023, Article ID 1514801, 12 pages
https://doi.org/10.1155/2023/1514801

Research Article

Book Recommendation Using Collaborative Filtering Algorithm

Esmael Ahmed[1] **and Adane Letta**[2]

[1]*Information System, College of Informatics, Wollo University, Dessie 7200, Ethiopia*
[2]*Computer Science, College of Informatics, University of Gondar, Gondar 6200, Ethiopia*

Correspondence should be addressed to Esmael Ahmed; esmael.ahmed@wu.edu.et

Received 11 November 2022; Revised 16 January 2023; Accepted 21 January 2023; Published 11 March 2023

Academic Editor: Agostino Forestiero

The explosive growth in the amount of available digital information in higher education has created a potential challenge of information overload, which hampers timely access to items of interest. The recommender systems are applied in different domains such as recommendations film, tourist advising, webpages, news, songs, and products. But the recommender systems pay less attention to university library services. The most users of university library are students. These users have a lack of ability to search and select the appropriate materials from the large repository that meet for their needs. A lot of work has been done on recommender system, but there are technical gaps observed in existing works such as the problem of constant item list in using web usage mining, decision tree induction, and association rule mining. Besides, it is observed that there is cold start problem in case-based reasoning approach. Therefore, this research work presents matrix factorization collaborative filtering with some performance enhancement to overcome cold start problem. In addition, it presents a comparative study among memory-based and model-based approaches. In this study, researchers used design science research method. The study dataset, 5189 records and 76,888 ratings, was collected from the University of Gondar student information system and online catalogue system. To develop the proposed model, memory-based and model-based approaches have been tested. In memory-based approach, matrix factorization collaborative filtering with some performance enhancements has been implemented. In model-based approach, K-nearest neighbour (KNN) and singular value decomposition (SVD) algorithms are also assessed experimentally. The SVD model is trained on our dataset optimized with a scored RMSE 0.1623 compared to RMSE 0.1991 before the optimization. The RMSE for a KNN model trained using the same dataset was 1.0535. This indicates that the matrix factorization performs better than KNN models in building collaborative filtering recommenders. The proposed SVD-based model accuracy score is 85%. The accuracy score of KNN model is 53%. So, the comparative study indicates that matrix factorization technique, specifically SVD algorithm, outperforms over neighbourhood-based recommenders. Moreover, using hyperparameter tuning with SVD also has an improvement on model performance compared with the existing SVD algorithm.

1. Introduction

The explosive growth in the amount of available digital information in higher education has created a potential challenge of information overload, which hinders timely access to items of interest [1]. Decision makers have fairly limited cognitive processing capacity. Consequently, when information overload occurs, it is likely that a reduction in decision quality will occur [2]. Information overload is a state in which a decision maker faces a set of information comprising the accumulation of individual informational cues of differing size and complexity that inhibit the decision maker's ability to optimally determine the best possible decision [3]. It has been noted that humans' capacity to find information advances more slowly than the pace at which new information is made available [4]. The current exponential growth of heterogeneous repository of information presents significant challenges to users and service-providers in many types of online environment. Traditional information management approaches in distributed systems are most often unsuitable for modern internet of things environments due to the huge amount and the extreme dynamism of the entities involved [5]. The ICT market is experiencing an important shift from the request/

provisioning of products toward a service-oriented view where everything is provided as a network-enabled service. It often happens that a solution to a problem cannot be offered by a single service, but by composing multiple basic services in a workflow [6].

Recommender systems have emerged as an important means of addressing these challenges [7]. Recommender systems have been developed to alleviate information overload, aid user decision-making, and achieve different forms of personalization [8]. It has become increasingly clear that traditional models of information retrieval frequently fail to best connect users with potentially relevant material [9]. This has increased the demand for recommender systems more than ever before. The recommender system has the ability to predict whether a particular user would prefer an item or not based on the user's profile. Such systems differ from more traditional forms of information retrieval in the sophistication with which an item's potential utility is calculated and the extent to which that they explicitly attempt to add value though a personalized approach [10]. To implement the personalized recommender systems, several machine learning algorithms have been forwarded [11]. In fact, there are two most popular areas of collaborative filtering, which are the latent factor approaches and the neighbourhood approaches. However, there is one main challenge in building collaborative filtering-based recommender systems which we call cold start. Cold start is the inability of the recommender systems to recommend items for new users. Some users may tend to give only high ratings, whereas others are a bit more pessimistic. As a result, the main focus of this study is to handle the cold start problem and improve the accuracy of the recommender system by applying enhancement techniques. The recommender system has the ability to predict whether a particular user would prefer an item or not based on the user's profile. Their effectiveness and usability have been demonstrated in a number of applications, including e-commerce, movies, music, travel, and social networks. The same information filtering and personalization needs are now arising in the area of university libraries [9]. In any university, the library is an information centre, research centre, and has also been an important channel, by which learners acquire the learning resources. Emergence of new technology has brought many new ideas for the information service of library [10]. However, these library services cannot effectively provide personalized information service, according to user preferences and specific needs [11].

The tremendous growth and usage of information have led to the problem of information overload in which users find it difficult to locate the right information at the right time and presents significant challenges to users and service-providers in many types of online environment [12]. It has been noted that humans' capacity to find information advances more slowly than the pace at which new information is made available [13]. Recommender systems have emerged as an important means of addressing these challenges [14]. One of these repositories of information is digital libraries. However, libraries have been slow to add recommendations to their catalogues. There are a number of reasons why it is reasonable to imagine that recommendations might be welcomed by users of library catalogues. Aside from numerous studies which report recommendations as a system requested by library users [15], it has become increasingly clear that traditional models of information retrieval frequently fail to best connect users with potentially relevant material [16]. Implementing the personalized recommender systems is crucial to analyze user profiles, content items, and the connections between them and try to predict future user behavior [17]. Such systems differ from more traditional forms of information retrieval in the sophistication with which an item's potential utility is calculated and the extent to which that they explicitly attempt to add value though a personalized approach [9]. To implement the personalized recommender systems, several machine learning algorithms have been forwarded [10, 11]. In fact, there are two most popular areas of collaborative filtering, which are the latent factor approaches and the neighbourhood approaches. However, there is one main challenge in building collaborative filtering-based recommender systems, which we call cold start. Cold start is the inability of the recommender systems to recommend items for new users. Some users may tend to give only high ratings whereas others are a bit more pessimistic. As a result, the main focus of this study is to handle the cold start problem and improve the accuracy of the recommender system by applying enhancement techniques.

2. Related Works

The recommender systems are applied in different domains such as recommendations film, tourist advising, webpages, news, songs, and products. However, the recommender systems pay less attention to university library services. The most users of university library are students. These users have a lack of ability to search and select the appropriate materials from the large repository that meet for their needs. Especially, in our country Ethiopia, most of the higher institution students come from rural area. Therefore, the importance of the recommender system that simplifies the service of the library is crucial. This study focuses on modelling collaborative filtering recommender system by comparing the popular algorithms. In this section, the review of related research works is discussed as follows.

As stated in [18], about 105 cases which are collected from successful students and 13 attributes which are collected from experts are used as case base. These attributes and cases are used as a knowledge base to construct case base recommender. The system calculates similarity between existing case and new queries that are provided by the students and provides a solution or recommendation by taking best cases to the new query. In this study, JCOLIBRI case base development tool is used to develop the prototype of the case-based recommender system [18]. The system provides only one recommendation to the user. This is because the JCOLIBRI programming tool uses cases to recommend to the user based on the new query and only one recommendation supported by it. Since the researcher used nearest neighbour, the problem of linear retrieval time when

there are many cases and it returns the nearest match even with dissimilarity cases in the source and new case occurred.

Another related work is "A personalized recommender system based on web usage mining and decision tree induction." In this work, the author proposes a personalized recommendation methodology, which is able to get further effectiveness and quality of recommendations when applied to an Internet shopping mall. The suggested methodology is based on a variety of data mining techniques such as web usage mining, decision tree induction, association rule mining, and the product taxonomy. The paper's drawback is that it only takes into account the recommendation problem of assisting selective clients in deciding which items they want to buy by providing a list of the top-N products at a given moment [19]. There is also another gap since there is a problem of constant item list.

In addition to that a work entitled "Recommender System in Tourism Using Case-Based Reasoning Approach" has been done to discuss about developing a recommender system for the tourist advising process. A knowledge-based recommender reasons about the fit between a user's need and the features of available products. Providing an effective service in the Ethiopian Tourism sector is critical to attract more foreign and local tourists. However, there are major problems that need immediate solution. The authors of this paper figure out some difficulty of getting fast, reliable, and consistent expert advice in the sector that is suitable to each visitor's characteristic and capabilities. Therefore, the authors aimed to design a recommender system for the tourist attraction area and visiting time selection that can assist experts and tourists to make timely decisions that help them to get fast and consistent advisory service [20]. In this study, about 615 cases, which are collected from the national tour operation, and 10 attributes, which are collected from experts are used as case base. These attributes and cases are used as a knowledge base to construct case-based recommender. The system calculates similarity between existing cases and new queries that are provided by the visitors and provides a solution or recommendation by taking best cases to the new query. In this study, JCOLIBRI case base development tool is used to develop the prototype. JCOLIBRI contains both user interface, which enables visitors to enter their query and programming codes with the help of Java script language. The authors evaluate their system using different evaluation methods and achieved 85% of an average performance. However, the relevant attributes used for this research were not sufficient for the selection of attraction area and visiting time decision [20]. However, the proposed recommender system is derived directly from a retrieved case that matches partially to the problem of the new case. So, some new cases have no solutions.

As described in [21], movie recommendation systems provide a mechanism to assist users in classifying users with similar interests. This makes recommender systems essentially a central part of websites and e-commerce applications. This article focuses on the movie recommendation systems whose primary objective is to suggest a recommender system through data clustering and computational intelligence. In this research article, a novel recommender system has been discussed, which makes use of k-means clustering by adopting cuckoo search optimization algorithm applied to the movie lens dataset. It is also compared with existing approaches, and the results have been analyzed and interpreted. Evaluation metrics such as mean absolute error (MAE), standard deviation (SD), root mean square error (RMSE), and value for the movie recommender system delivers better results as their approach offers lesser value of the mean absolute error, standard deviation, and root mean square error [21]. The experiment results obtained on Movie lens dataset specify that the proposed approach may provide high performance regarding reliability and efficiency and delivers accurate personalized movie recommendations when compared with existing methods [21]. However, the main limitation to this approach is that if the initial partition of the clustering is not a reliable one, then at that point efficiency may decrease.

Study entitled with "application of case-based recommender system in investment sector and investment activity selection to new investors: in the case of Ethiopia" is also reviewed. Author develops case-based recommender system for investment sector and investment activity selection. The authors reported that knowledge was acquired through interviews and document analysis. Twelve domain experts and four investors were interviewed to elicit the required knowledge about investment sector and investment activity. The acquired knowledge was represented using feature value case base representation and implemented using colibri programming tool [22]. The main data source used to develop case-based recommender system for investment sector and investment activity selection is previous investor cases. Nearest neighbour retrieval algorithm is used to measure the similarity of new case (query) with cases in the case base. As a result, if there is a similarity between the new case and the existing case, the system assigns the solution of previous case as a solution to new case [22]. However, one of the weaknesses of this study is if there is no previous case similar to the new cases model unable to recommender or provide a solution. Authors evaluated the system by the domain experts and investors through visual interaction based on the criteria of easiness to use, time efficiency, applicability in the domain area, and providing correct recommendation. However, identifying correct domain experts and investors make the evaluation process difficult [23]. The proposed platform is particularly suited for dynamic and unstable systems because it is entirely decentralized, self-organizing, and automatically adjusts to the changing environment. Multiagent approaches guarantee service continuity, stability, and limited user action, but, often, a central management mechanism is necessary with all resulting disadvantages, among which a central point of failure.

In spite of the aforementioned research works, a lot of work should be done on the recommender system. The reason for this is that there were technological gaps discovered in earlier studies, such as the problem with the constant item list when using online usage mining, decision tree induction, and association rule mining. Besides, it is observed that there is cold start problem in case-based

reasoning approach. If there is no previous case similar to the new cases, the model is unable to provide a recommendation with cold start problem. Therefore, this research work presents matrix factorization collaborative filtering with some performance enhancement to overcome cold start problem. In addition, it presents comparative study among memory-based and model-based approaches. Compared to existing approaches, our proposed platform relies on rating-based recommender system. Moreover, our approach shows the discovery to overcome cold start problem using hyperparameter tuning with enhanced performance.

3. Materials and Methods

Nowadays, machine learning (ML) is used in developing adaptive intelligent systems that can perform complex tasks that are beyond human abilities. Some of the areas of applications of ML algorithms include pattern recognition, image processing, natural language processing, and medical diagnostic, to mention just a few [23]. Different types of ML algorithms exist, in the context of how they are applied to the field of recommender systems. Different machine learning algorithms can be implemented in both memory-based and model-based recommender systems. Model-based recommender system constructs a predictive model to estimate unknown ratings by learning from the observed data. Several existing approaches for multicriteria rating recommenders fall into this category, including aggregation function, probabilistic modelling, and singular value decomposition (SVD).

3.1. Recommender System Architecture. The data preprocessing consists of data cleaning, data integration, data pruning, and dimensionality reduction. The user-item matrix creation is extension of data integration in preprocessing. After three datasets integrated, the matrix also created and stored as sparse matrix to make it suit to train the model. We used hyperparameter tuning for enhancement of the algorithm. Hyperparameter tuning is used for the automatic enhancement of the hyperparameters of a model. Hyperparameters are all the parameters of a model, which are not updated during the learning and are used to configure the algorithm to lower the cost function of learning rate for gradient descent algorithm. We apply this on the features which are fed into the algorithm. In this study, hyperparameter tuning is used just to enhance the loop of model learning to find the set of hyperparameters leading to the lowest error on the validation set. Thus, a validation set has to be set apart, and a loss has to be defined. In this study, researcher applies matrix factorization for factoring the user-item-rating matrix. This often raises difficulties due to the high portion of missing values caused by sparseness in the user-item-ratings matrix. Moreover, researcher used it to take care of addressing the relatively few known entries. Another component is similarity calculation. Similarity in a recommender system is about finding items or users, or user and item that are similar. In this study, researcher uses cosine similarity measurement for model-based approach and memory-based correlation to find similarities. After identifying the similarity, matrix factorization and nearest neighbour models map both users and items to a joint latent factor space, such that user-item interactions are modelled in that space. Figure 1 shows a proposed collaborative recommender component architecture, which combines the components in the proposed collaborative filtering recommender system.

3.2. Data Preprocessing. The initial dataset included three separate data frames, one containing users' rating, second containing books, and the third contains users. The first step in the data preprocessing process was to combine these data frames into a single matrix of ratings, in which each row represented a user and each column represented a book. The matrix was very sparse, because each reader has only rated a small fraction of the books in the dataset. To help reduce the sparseness and make the data easier to process, the books that received less than 10 ratings were dropped. Then, users who had given fewer than 10 ratings to the remaining books also dropped. As attribute level, the column place of publication has been dropped because this is insignificant in this scope of the study. Furthermore, researcher creates the data frame for historical book and removes these old books to reduce and utilize the data. Since the modelling step in this study uses only the explicitly ratings, the zero entries of the rating were removed. There are 98 books with multiple ISBN numbers, so researcher creates and assigns a unique identifier for each book automatically because it is useful for a model. Thereafter the cleaned data are transformed into user-item-rating matrix.

3.3. User-Item-Rating Matrix. A user-item matrix is built from the interaction records, and the size of dimensionality reduction is defined. This matrix created after various activities preprocessing such as data cleaning, data integration, and data transformation. After data preprocessing, the data are stored in the form of user-item-rating of sparse matrices. To build the matrix, the three datasets were integrated using different identifiers. From the unified data frame, some attributes were selected, which are used for building a collaborative recommender. After dropping remain attributes, the matrix was created, which can be fit to the algorithms. To improve the SVD algorithms, researchers apply the hyperparameter tuning to readjust the whole data instance.

3.4. Hyperparameters Tuning. The model parameters are enhanced or tuned by the training process. We run data through the operations of the model, compare the resulting prediction with the actual value for each data instance, evaluate the accuracy, and adjust until to get the best values. Hyperparameters are tuned by running the whole training data to looking at the aggregate accuracy and adjusting. Dataset pruning is the process of removing suboptimal tuples from a dataset to improve the learning of a machine learning model. The idea of pruning is to consider a subset of hyperparameter configuration space to avoid unnecessary

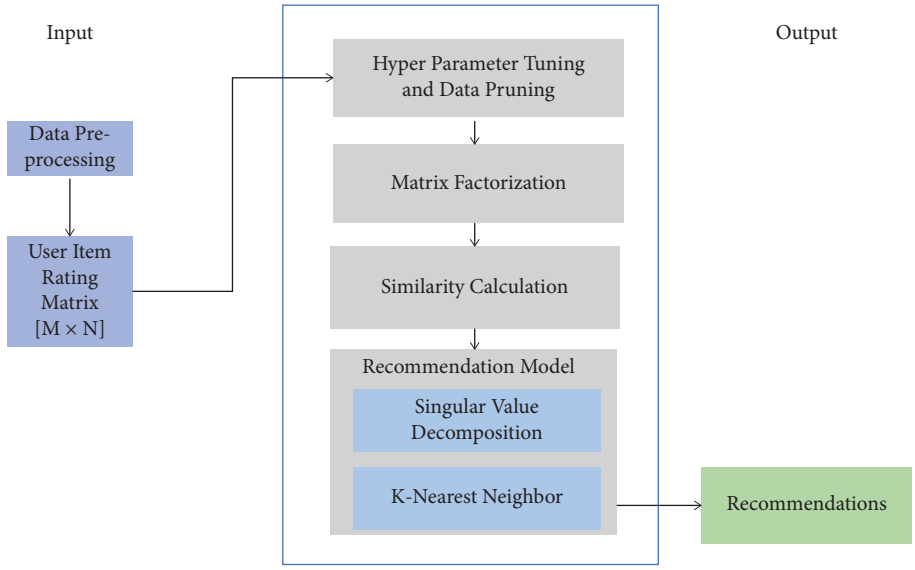

Figure 1: Architecture of the proposed collaborative recommender system.

function or attributes of data. The preprocessed dataset was then pruned based on the number of user reviews for a book. We iteratively pruned the dataset where a book had received less than 10 user reviews. Generally, the model architecture is defined by several parameters. These parameters are referred to as hyperparameters. The process of searching for ideal model architecture for optimal accuracy score is hyperparameter tuning.

3.5. Matrix Factorization. Matrix factorization models map both users and items with a joint latent factor space of dimensionality f, such that user-item interactions are modelled as inner products in that space. Accordingly, each item I is associated with a vector QI $\in$ Rf, and each user u is associated with a vector Pu $\in$ Rf. For a given item I, the elements of qi measure the extent to which the item possesses those factors, positive or negative. For a given user u, the elements of Pu measure the extent of interest the user has in items that are high on the corresponding factors, again, positive or negative. The resulting dot product, Qi. Pu, captures the interaction between user u and item I the user's overall interest in the item's characteristics. This approximates user u's rating of item I, which is denoted by Rui, leading to the estimate [24].

$$\mathbf{Rui} = \mathbf{Qi}.\,\mathbf{Pu}. \tag{1}$$

The major challenge is computing the mapping of each item and user to factor vectors. Since the dataset used in this study contains 100,000 records, it takes some minutes to map items and users to factor vectors. So, the proposed prototype uses parameter tuning to minimize the training time. After the recommender system completes this mapping, it can easily estimate the rating a user will give to any item by using equation (1).

The proposed model is closely related to singular value decomposition (SVD). Applying SVD in the collaborative filtering domain requires factoring the user-item-rating matrix. This often raises difficulties due to the high portion of missing values caused by sparseness in the user-item-ratings matrix. Earlier systems relied on imputation to fill in missing ratings and make the rating matrix dense. However, imputation can be very expensive as it significantly increases the amount of data. In addition, inaccurate imputation might distort the data considerably. In this study, the prototype modelled using directly observed ratings only, while avoiding overfitting through a regularized model. To learn the factor vectors Pu and Qi, the system minimizes the regularized squared error on the set of known ratings. The system learns the model by fitting the previously observed ratings. However, the goal is to generalize those previous ratings in a way that predicts future, unknown ratings. Thus, the system should avoid overfitting the observed data by regularizing the learned parameters, whose magnitudes are penalized. Researchers use cosine similarity metric to find the normalized dot product of the two attributes, which are users and books. The reason why we select cosine similarity from other metric is that it is very efficient to evaluate, especially for sparse vectors. Since we use the sparse matrix, the model could effectively find the cosine of the angle between the two objects, namely users to items, users to users, and items to items by determining the cosine similarity.

3.6. Similarity Calculation. Similarity is built between the users who share the same rating pattern with active user. For both models, the cosine similarity metric over average rating values was applied. While the original user-item matrix and similarity matrix for the memory-based approach are used in the computation of the correlation on the rating-based correlation. For clear comparison, we also used cosine similarity for memory-based approach.

Researchers use cosine similarity metric to find the normalized dot product of the two attributes, which are users and books. The cosine of 0° is 1, and it is less than 1 for

any other angle. It is thus a judgment of orientation and not magnitude: two vectors with the same orientation have a cosine similarity of 1, two vectors at 90° have a similarity of 0, and two vectors diametrically opposed have a similarity of −1, independent of their magnitude. The reason why we select cosine similarity from other metric is that it is very efficient to evaluate, especially for sparse vectors. Since we use the sparse matrix, the model could effectively find the cosine of the angle between the two objects, namely, users to items, users to users, and items to items by determining the cosine similarity.

3.7. Recommendation Model. In this study, two collaborative filtering algorithms are experimented. The first is singular value decomposition and the other one is nearest neighbour. Singular value decomposition models map both users and items to a joint latent factor space of dimensionality such that user-item interactions are modelled. Some of the most successful realizations of latent factor models are based on matrix factorization. In its basic form, matrix factorization characterizes both items and users by vectors of factors inferred from item rating patterns. High correspondence between item and user factors leads to a recommendation. These methods have become popular in recent years by combining good scalability with predictive accuracy. In addition, they offer much flexibility for modelling various real-life situations. Recommender systems rely on different types of input data, which are often placed in a matrix with one dimension representing users and the other dimension representing items of interest. The most convenient data are high-quality explicit feedback, which includes explicit input by users regarding their interest in item. Usually, explicit feedback comprises a sparse matrix, since any single user is likely to have rated only a small percentage of possible items. One strength of matrix factorization is that it allows incorporation of additional information. When explicit feedback is not available, recommender systems can infer user preferences using implicit feedback, which indirectly reflects opinion by observing user behavior.

Generally, the proposed SVD collaborative filtering algorithm is as follows:

Step 1. Preprocessing: A user-item matrix is built from the interaction records, and the size of dimensionality reduction is defined.

Step 2. Similarity evaluation: This step refers to the neighbourhood formation with each table entry including the corresponding similarity metric equation. Here, SVD is applied on the user-item matrix using a slightly different similarity metric equation with the ratings taken from reduced user-item matrix.

Step 3. Rating process: The calculation of the correlation using the enhanced equation on the rating-based correlation using original user-item matrix and similarity matrix.

Step 4. Recommendation: The filtering process is concluded with prediction generation formulas using the SVD applied user-item matrix.

Another machine learning algorithm used in developing model is KNN. It involves finding the top K-nearest neighbours for an item. Ratings from the list of nearest neighbours are combined to predict the unknown ratings. This normally involves finding all user-user correlation and item-item correlation. The K-nearest neighbour classifier usually applies either the Euclidean distance or the cosine similarity between the training tuples and the test tuple but, for the purpose of this research work, the cosine similarity approach has been applied in implementing the KNN model for our recommendation system. K-nearest neighbour approach involves finding the top K-nearest neighbours for an item by finding all user-user correlation and item-item correlation. KNN algorithms works as follows:

Step 1: Compute the mean rating value of every user according to user-item-rating matrix.

Step 2: Calculate similarity based on distance function. There are many distance functions, but in this study cosine similarity measure is used.

Step 3: Find K neighbours of user by searching for K users closest to specific user, which is most similar to specific user in terms of attributes.

Step 4: List top N similar items for those similar users.

Using the above two machine learning algorithms, we develop model by providing the training data to a machine learning algorithm to learn. The goal of a collaborative filtering model is to suggest new items or to predict the utility of a certain item for a particular user based on the user's previous likings and the opinions of other like-minded users. The task of CF algorithms is to find an item likeliness. So, the model can predict a numerical value that expresses the predicted score of an item for the user. The predicted value is within the same scale that is used by all users for rating. A list of Top N items is recommended that the active user will like the most. The CF algorithm includes user-based and item-based collaborative filtering algorithms. The user-based CF is implemented with singular value decomposition technique of matrix factorization, while the item-based CF is implemented using K-nearest neighbourhood method. The developed model is able to conclude the filtering process with prediction generation formulas using the SVD applied user-item matrix. Thus, the model is used to represent the real-world mathematical process. Finally, it provides the list of top recommendations for the users.

3.8. Recommender Model Evaluation. The evaluation metrics of recommender systems can greatly vary depending on the characteristics of the dataset, namely, size and rating scale, the goal of recommendation, and the purpose of evaluation. In the collaborative filtering model, the goal is to evaluate the predictive accuracy, namely, how closely the recommender system can predict the true ratings of the users, measured in terms of root mean squared error. We used root mean squared error (RMSE) of predicted values in the test set. Since we want to compare only predicted ratings that are in the test set, we can filter out all other predictions that are not in the test matrix. In this study, the most popular metrics to

measure the accuracy of a recommender system, which are the mean absolute error (MAE) and the root mean have been used. The RMSE is the square root of the variance of the residuals. It indicates the absolute fit of the model to the data—how close the observed data points are to the model's predicted values. SVD is usually evaluated with statistical accuracy metrics, MAE, and R MSE. Both metrics measure the deviation of the predicted value to actual value.

4. Results and Discussion

4.1. Dataset. The dataset is comprised of three separate tables: users, books, and ratings. These data are inconsistent and dirty, so data preprocessing has been done. The rating scale is a typical 1–5 integer number, 1 refers to a lowest rating, and 5 being a highest one. Since the quality of input data has an impact on the recommender system, data preparation has a paramount importance. To get a quality data, the following activities are done. The researcher preprocesses the data using Python software. The major problems of the original dataset that need data preprocessing is attributes that have so many missing values; the data contain duplicated records, in the original dataset.

4.2. Data Cleaning. The dataset used in this study which is stored in comma-separated values (csv) had multiple cases of semicolons in the book titles, which were manually cleaned. Mostly semicolons were changed to colons or commas. Also, the symbol "&" (presumably an ampersand character) appeared a lot, which was changed to just "&amp." To get more cleaned data, we tidy up all column names. The data also have duplicated records such as book data contain duplicated ISBN, and users also duplicated in student data. So, we removed the duplicated records and used only unique users. There were also 98 books with inconsistent ISBN identifier; some books with the multiple ISBN and some others were without ISBN number. We assign the unique ISBN for these books. So, only unique books were used in fitting the model.

4.3. Data Integration. The initial dataset included three separate data frames, one containing users' rating, second containing books, and the third contains users. The other step in the data preprocessing process was to combine these data frames into a single matrix of ratings, in which each row represented a user and each column represented a book. So, firstly, we join the book dataset with the rating records using the book ISBN number. After unifying the tables, the new table size is minimized. The user table then merged to the new table that was created from the rating and book tables. The relationship formed in the "user id" column.

4.4. Data Pruning. The matrix was very sparse, because each reader has only rated a small fraction of the books in the dataset. So, most of the elements of the matrix are zero. Researchers examine the data and take threshold of ten because after trying different cut-off, we identify that books

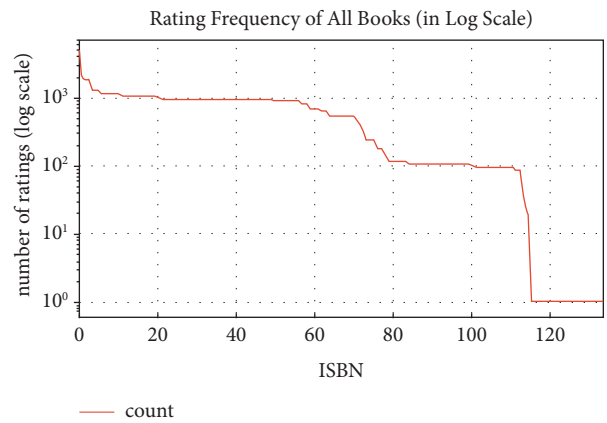

Figure 2: Rating frequency of all books.

which are rated have rating counts more than ten. If we take threshold less than ten, almost all matrices should be taken and else most of the books should be dropped. To help reduce the sparseness and make the data easier to process, the books that received less than 10 ratings were dropped.

Figure 2 shows the rating count distribution of all books along counts of books. As the inspection graph indicates, the rating distribution has variety. Some books have more rating score while some other books have less. For instance, up to 20 books from the collections rated more, while books more than 120 rated less.

As shown in Figure 3 the small numbers of users rate their books. For example, the number of users count up to 1000 have more rating history while more users above users count of 4000 have less rating history. This means most of library users have no rating history. This is one challenge of this study that since the users have no explicit rating, the recommender model cannot be able to recommend books as per their interest. Furthermore, researchers create the data frame for historical book and remove these old books to reduce and utilize the data.

Figure 4 shows the distribution of books along the year of publication. As it is indicated, the historical books have less count compared to the latest books based on the publication year. Researcher used this distribution to cut off the old books and magnify the latest edited and published books to the users. For instance, after 1990s, more books are published compared to the earlier years. So, the latest books are mostly preferred by the users since they contain the updated information than of the earlier books. So, the proposed model takes into account this idea by taking the latest books.

Since the modelling step in this study uses only the explicitly ratings, the zero entries of the rating were removed. There are 98 books with multiple ISBN numbers, so researcher creates and assigns a unique identifier for each book automatically because it is useful for a model. As we are evaluating favourability of review text, it can be helpful to understand how many reviews rate the book as 4 stars or higher the definition of favourable for this analysis. Surprisingly, 78% of all reviews were favourable with only 10% rating the book with 2 stars or less. After finding the average rating for all users and sorting into 5 bins, researcher found

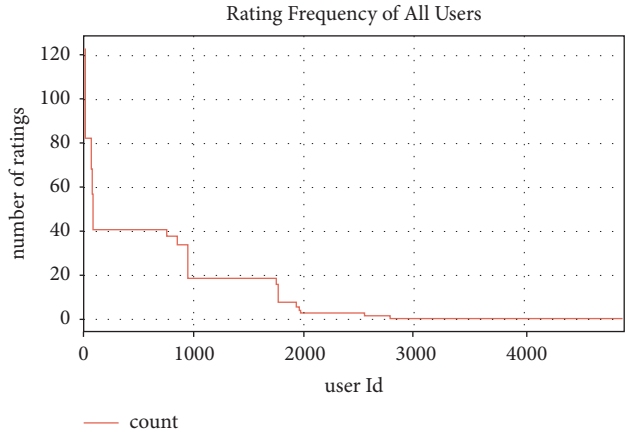

Figure 3: Rating frequency of all users.

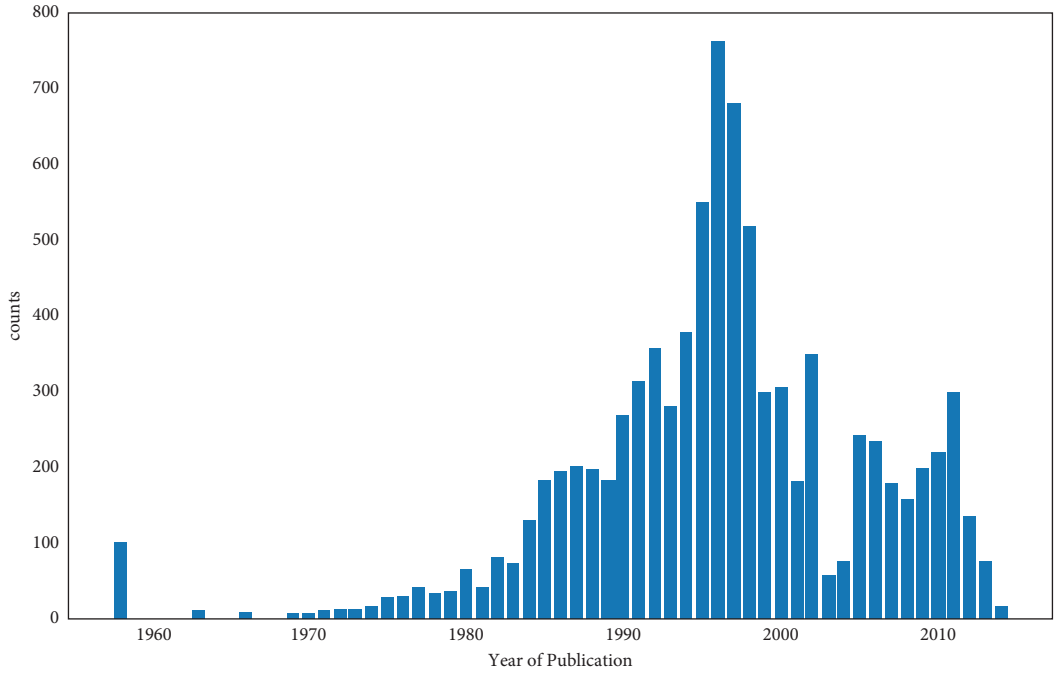

Figure 4: Book distribution along publication year.

that almost 58% of user's rate books 5 stars on average. In stark contrast, only around 8% rate books 1 or 2 stars on average. A little less than 10% of reviewers rate 3 stars on average.

As Figure 5 indicates, the rating star 5 is dominant scale over others. However, the rating star 4 also has high number of users. The graph shows book rating scale along count of each rating score in logarithmic tail scale inspection. Researcher analyzes the data to magnify a sense of what additional work should be performed to quantify and extract insights from the data. As we are evaluating favourability of review text, it can be helpful to understand how many reviews rate the book as 4 stars or higher the definition of favourable for this analysis. The dataset was cleaned, integrated, and pruned using several metrics, including users with the lowest number of rating history and books with the fewest number of ratings, and thus the rating distribution differed slightly from the original dataset. Figure 6 shows the summary of new rating distribution of the preprocessed data.

4.5. Experimentations. In this study, matrix factorization and K-nearest neighbour have been implemented to develop the recommender system. We performed two main experiments. In the first experiment, memory-based collaborative learning has been experimented. In the second experiment, model-based collaborative learning has been experimented using SVD and KNN algorithms. In each experiment, root

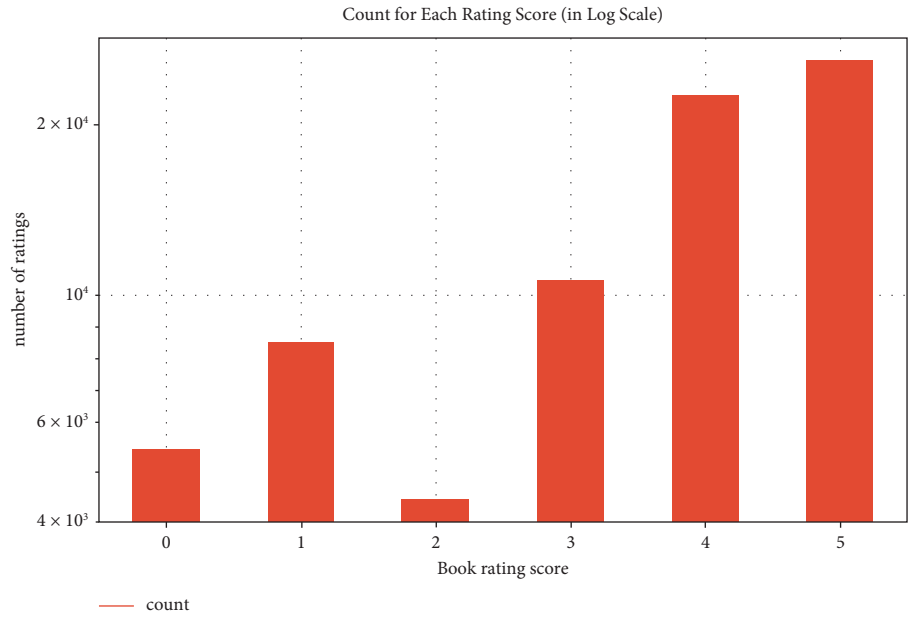

Figure 5: The distribution of ratings before data preprocessing.

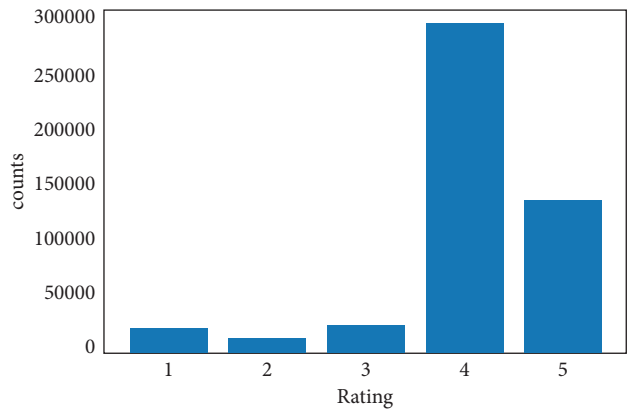

Figure 6: The distribution of ratings after data preprocessing.

mean squared errors were used to measure the performance of the mentioned algorithms. The comparative study was carried out for two types of recommendations techniques, model-based rating prediction, and memory-based prediction. Additionally, comparative study was conducted between SVD and KNN based models. The metric used for validating the model the 5-fold cross validation was performed in the evaluation of all the algorithms. Because of limitations to the available hardware and time constraints, we used 5-fold for validation of the parameters.

4.5.1. Experimental Environment and Tools. In this research, we applied experiments on a machine with properties, that is, Intel (R) Core (™) Duo i3-E7500 CPU@ 2.93 GHz, 4.00 GB RAM 320 GB hard disk drive window 10 operating system installed. To carry out the experimentation, special tools and programs were used. Anaconda IDE is used to navigate Python 3 in building and evaluating the model. Particularly, Jupiter Notebook is used to visualize the data and to use a virtual kernel to evaluate memory-based collaborative filtering technique of the recommender system. Additionally, Microsoft excel was used to handle the data.

4.5.2. Platform Setup. In this research, Python was chosen because of its easy to learn syntax. Software for development of the code was chosen from either a simple editor where only code can be written, such program as notepad and anaconda or an IDE such as Microsoft's Visual Studio code or Eclipse. In this case, anaconda was chosen because it is more easy to import functions. The libraries, packages, and modules used are brought up below. The mean values are calculated by NumPy using its mathematical functions. Pandas use functions for constructing and modifying data frames as well as read functions to retrieve data from files.Scikit-learn, using its random forest and tree module and from the surprise package inside scikit-learn, reader, dataset, SVD, and evaluate. Scikit-learn, sometimes called sclera, is a package available in python. Sclera contains machine learning tools such as classification, regression, clustering, dimensionality reduction, model selection, and preprocessing. In this study, dimensionality reduction, model selection, and preprocessing have been used. Sclera offers a wide range of the most popular machine learning algorithms, for example, decision tree, random forest, SVD, and SVM. Furthermore, in this study, surprise and collections packages are also used. Surprise is a Python scikit building and analyzing recommender systems that deal with explicit rating data. Researcher uses various ready-to-use prediction algorithms such as neighbourhood methods and matrix factorization-based specifically singular value decomposition. Also, cosine similarity measure was used. Researcher also used a cross validation procedures to evaluate, analyze, and compare the algorithm's performance.

Another platform used in this research work is collections, which is containers that are used to store collections of data, particularly the default dictionary used to list ranked recommendations for users.

4.6. Model Comparison. After training the memory-based collaborative filtering model, researchers evaluate the accuracy of the model by root mean squared error (RMSE) of predicted values in the test set. Since the aim is to compare only predicted ratings that are in the test set, researchers filter out all other predictions that are not in the test matrix and then call on test set to get error from each approach user-user CF and item-item CF. The estimated value of RMSE for user-user CF in memory-based approach is 3.35 and 3.52 for item-item CF. Algorithms that have been used in this study were KNN and SVD. However, memory-based correlation method ignored after it estimated low prediction accuracy. SVD was tested with the dataset and scored good result. The SVD model estimated a RMSE of 0.1623 and KNN model estimated a RMSE of 1.0535.

As shown in Figure 7 the lower diagonal of the graph shows that the estimated values of RMSE by SVD-based model are too smaller than RMSE of KNN-based model. It shows the big difference between SVD and KNN, so SVD based model is better. The same is true in case of MAE. The upper diagonal of the graph shows taht the estimated values of RMSE by KNN based model are larger than RMSE of SVD-based model. The comparison result of both models using RMSE and MAE is shown in Figures 7 and 8 respectively [25].

So, we conclude that the SVD-based model which improved using hyperparameter tuning performs well when we compared it to the KNN-based model to recommend the book to the library users. The error values against K values showed that as K values and dataset size increase, the error of the KNN model also increased. This indicates that KNN models accuracy decreased when the data increased. However, relatively, the SVD model can handle the problem accuracy reduction when the data size increased.

4.7. Experimental Result. In singular value decomposition, first we need to create a reader object to set the scale or limit of the ratings field to play with dimensionality reduction. Thereafter, load the model and fit the dataset without manually mapping of user id and unique ISBN to integers in a custom dictionary. Lastly, the model is evaluated using RMSE of algorithm SVD on 5 splits of cross validation. Table 1 shows the evaluation result of the model in RMSE [26].

Researchers tried to improve the predictions of the model by enhancing some of hyperparameters of the algorithm. To do this, the grid search cross validation method was implemented. When a range of hyperparameter values is passed, grid search cross validation automatically searches through the parameter-space to find the best-performing set of hyperparameters. We finely tuned the SVD model by implementing fit and score method. We also implemented parameter estimator to apply predict method, which is

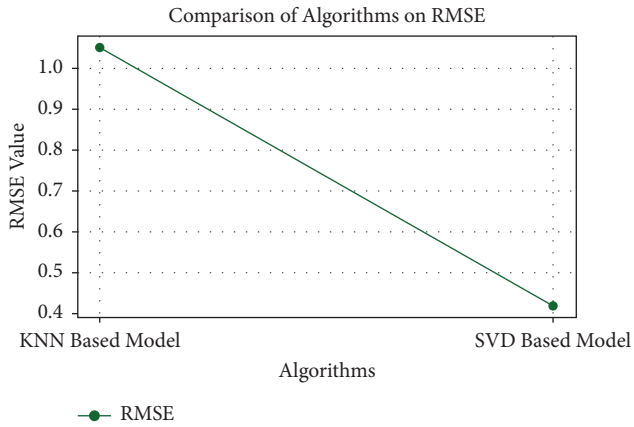

Figure 7: The comparison of algorithms on RMSE.

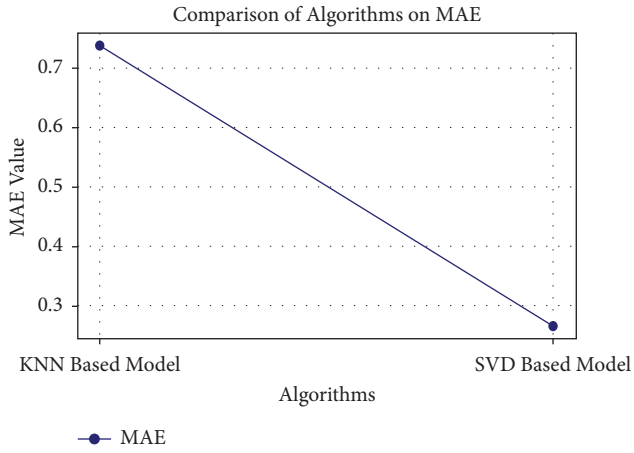

Figure 8: The comparison of algorithms on MAE.

Table 1: Evaluating RMSE of algorithm SVD on 5 splits.

	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Mean	Std
RMSE (test set)	0.2773	0.2432	0.2975	0.2581	0.2933	0.2739	0.0207
Fit time	48.44	50.29	47.25	50.08	51.64	49.54	1.53
Test time	2.72	2.44	2.06	2.68	2.69	2.52	0.25
Total wall time: 4 min 28 s							

enhanced by cross-validated grid search over a parameter grid. As shown in Table 2 the mean error rate [26] of the enhanced model decreased from 0.2739 to 0.1994.

The SVD model registers 99% precession and 40% of recall. The accuracy of SVD model is 85%. The summary of accuracy evaluation report of the result is shown in Table 3. The SVD outperforms in any case as models are evaluated using different metrics. One of the main drawbacks of the KNN algorithms observed from this study is that their predictions are often quite concentrated around the mean. However, the SVD algorithm is more comfortable predicting extreme rating values. The SVD model also needs more

Table 2: Evaluating RMSE and MAE of algorithm SVD on 5 splits after parameter tuning.

	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Mean	Std
RMSE (test set)	0.1614	0.2129	0.2109	0.2031	0.2089	0.1994	0.0193
MAE (test set)	0.1063	0.1549	0.1489	0.1481	0.1480	0.1412	0.0176
Fit time	59.34	51.03	50.98	46.92	47.93	51.24	4.37
Test time	2.91	3.28	2.98	2.95	3.24	3.07	0.16

Table 3: Confusion matrix report registered by SVD model.

Accuracy evaluation report			
Fold number	Precision	Recall	Support
1	0.99	0.40	529
2	0.99	0.40	1402
3	0.99	0.40	792
4	0.99	0.41	1895
5	0.99	0.41	4027
Accuracy		0.85	

improvement since it recommends 85% of accurate recommendation. There are different reasons for why not the SVD model registers more accurate result than this. The first main reason is that we exclude the implicit rating from the training set for decreasing the sparseness of the data frame because the dataset has some ratings of zero.

Another reason is that any latent factor models attempt to find weighted low-rank approximations to the user-item matrix, so the prediction comes from the approximationsThe function used to calculate approximation quality suffers from a few implicit factors because we exclude zero ratings. For latent factor model-based algorithms, we obtained the most favourable results and we observed an optimum accuracy, which is 85%, as shown in Table 3. A grid search cross validation method was applied to enhance some of the hyperparameters for the model, resulting in a slight improvement in model performance from the default SVD performance.

5. Conclusions

Collaborative filtering (CF) algorithms are most commonly used in recommendation system (RS). CF algorithms nowadays face the problem with large dataset, sparseness in rating matrix, and cold start problem. In this study, researchers have studied the role of matrix factorization (MF) model to deal with the collaborative filtering (CF) challenges. From this study, we can say that SVD is able to handle large dataset, sparseness of rating matrix, and scalability problem of CF algorithm efficiently. SVD is able to find a linear projection of high dimensional data into a lower dimensional subspace and the least square reconstruction error is minimized. All the techniques which have been researched until now are trying to increase the accurate model. Building a recommender system that achieves good recommendations in new users or cold start scenario still is a challenge. In order to create a model with acceptable results, it may be necessary to count with more information, not only about the user's profile but also about the items, and this could allow to implement other methodologies. Users probably are looking for books that they have not read or books with different topics. Recommending books with diverse topics allow users to explore different tastes and keeps user engaged with the recommender product. On the other hand, lack of diversity will make users get bored and less engaged with the product. This type of challenges can be minimized in item-based collaborative filtering as indicated by experimental results. Therefore, in this study, researchers experiment both memories based and model-based approaches. We are pleased with the given result because of having big problems with communication and availability of the data. The models are able to recommend books in the form of a list. The SVD model trained on our dataset performed with a RMSE of 0.1623 compared to the existing SVD algorithm that estimated 0.1991 before the enhancement.

Even though the developed model is able to deliver successful recommendations, it is still a long way to go to integrate these presented methods into the other recommendation system methods to generate a better performance and user experience. Another interesting further research is integrating the collaborative filtering techniques with demographic recommender systems and incorporates the implicit feedback to tackle the cold start problem. Additionally, a fascinating future research is to create a model that can grasp the interests and desires of the user accurately and then match it to the best fitting available items.

Data Availability

The data used to support the findings of this study are included within the supplementary information file(s).

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

The authors would like to take this opportunity to express their heart-felt gratitude for their advisor Adana Letta (PhD) for his encouragement, valuable suggestions, and comments that helped them in doing this research work. The authors would like to thank the University of Gondar library directorate for their assistance with the collection of their data. The authors extend their special thanks to Mr. Aleen Adana (University of Gondar library directorate director) and Mr. Tesha Digges (Library technical manager) for their assistance with data collection. Next, their gratitude goes to those who helped them throughout my journey. The authors thank their families for their support not only in this research but throughout their education journey. The authors would also like to thank their colleagues and friends for their encouragement and support. This work was supported by the University of Gondar. This research received no specific

grant from any funding agency in the public, commercial, or not-for-profit sectors other than the University of Gondar advisor assignation.

References

[1] P. A. Burrough, R. McDonnell, R. A. McDonnell, and C. D. Lloyd, *Principles of Geographical Information Systems*, Oxford University Press, Oxford, UK, 2015.

[2] Z. S. Gidadawa and M. B. Dogondaji, "Application of ICT in Nigerian educational system for achieving sustainable development," *International Letters of Social and Humanistic Sciences*, vol. 32, pp. 62–71, 2014.

[3] N. Davis, "Technology in Teacher Education in the USA: what makes for sustainable good practice?" *Technology, Pedagogy and Education*, vol. 12, no. 1, pp. 59–84, 2003.

[4] F. O. Isinkaye, Y. O. Folajimi, and B. A. Ojokoh, "Recommendation systems: principles, methods and evaluation," *Egyptian Informatics Journal*, vol. 16, no. 3, pp. 261–273, 2015.

[5] A. Forestiero and G. Papuzzo, "Agents-based algorithm for a distributed information system in internet of things," *IEEE Internet of Things Journal*, vol. 8, no. 22, pp. 16548–16558, 2021.

[6] A. Forestiero, C. Mastroianni, G. Papuzzo, and G. Spezzano, "A proximity-basedself-organizing framework for service composition and discovery," in *Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pp. 428–437, Melbourne, Australia, May 2010.

[7] P. Zanarini, "Information overload, why it matters and how to combat it," 2017, http://https//www.%20Interact.%20org/literature/article/information-overload-whyit-matters-and-how-to-combat-it.

[8] P. G. Roetzel, "Information overload in the information age: a review of the literature from business administration, business psychology, and related disciplines with a bibliometric approach and framework development," *Business research*, vol. 12, no. 2, pp. 479–522, 2018.

[9] M. Huda, A. Maseleno, P. Atmotiyoso et al., "Big data emerging technology: insights into innovative environment for online learning resources," *International Journal of Emerging Technologies in Learning*, vol. 13, no. 1, pp. 23–36, 2018.

[10] J. A. Konstan and J. Riedl, "Recommender systems: from algorithms to user experience," *User Modeling and User-Adapted Interaction*, vol. 22, no. 1–2, pp. 101–123, 2012.

[11] C. Pan and W. Li, "Research paper recommendation with topic analysis," in *Proceedings of the 2010 International Conference On Computer Design and Applications*, vol. 4, pp. V4–264, Qinhuangdao, China, June 2010.

[12] I. Soboroff, C. Nicholas, and M. Pazzani, "Workshop on recommender systems: algorithms and evaluation," *ACM SIGIR Forum*, vol. 33, no. 1, pp. 36–43, 1999.

[13] 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.

[14] K. I. Ghauth and N. A. Abdullah, "Learning materials recommendation using good learners' ratings and content-based filtering," *Educational Technology Research and Development*, vol. 58, no. 6, pp. 711–727, 2010.

[15] M. Feng, N. Heffernan, and K. Koedinger, "Addressing the assessment challenge with an online system that tutors as it assesses," *User Modeling and User-Adapted Interaction*, vol. 19, no. 3, pp. 243–266, 2009.

[16] N. Hafizovic, *Candidate-job Recommendation System: Building a Prototype of a Machine Learning-Bbased Recommendation System for an Online Recruitment Company*, 2019.

[17] S. Wakeling, P. Clough, and B. Sen, "Investigating the potential impact of non-personalized recommendations in the OPAC: amazon vs. WorldCat. org," in *Proceedings of the 5th Information Interaction in Context Symposium*, pp. 96–105, Regensburg, Germany, August 2014.

[18] B. Getnet, *Application of Case Based Recommender System to Advise Students in Field of Study Selection at Higher Education in Ethiopia*, Addis Ababa University, Addis Ababa, Ethiopia, 2013.

[19] J. Wei, J. He, K. Chen, Y. Zhou, and Z. Tang, "Collaborative filtering and deep learning based recommendation system for cold start items," *Expert Systems with Applications*, vol. 69, pp. 29–39, 2017.

[20] A. M. Elkahky, Y. Song, and X. He, "A multi-view deep learning approach for cross domain user modeling in recommendation systems," in *Proceedings of the 24th International Conference on World Wide Web*, pp. 278–288, Florence, Italy, May 2015.

[21] Y. H. Cho, J. K. Kim, and S. H. Kim, "A personalized recommender system based on web usage mining and decision tree induction," *Expert Systems with Applications*, vol. 23, no. 3, pp. 329–342, 2002.

[22] T. Anteneh Alemu, A. K. Tegegne, and A. Nega Tarekegn, "Recommender system in tourism using case based reasoning approach," *International Journal of Information Engineering and Electronic Business*, vol. 9, no. 5, pp. 34–43, 2017.

[23] A. Forestiero and G. Papuzzo, "Recommendation platform in Internet of Things leveraging on a self-organizing multiagent approach," *Neural Computing and Applications*, vol. 34, pp. 1–12, 2022.

[24] M. Balabanović and Y. Shoham, "Fab: content-based, collaborative recommendation," *Communications of the ACM*, vol. 40, no. 3, pp. 66–72, 1997.

[25] N. Hug, "Surprise: a Python library for recommender systems," *Journal of Open Source Software*, vol. 5, no. 52, 2020.