electronics

Article

E-Learning Course Recommender System Using Collaborative Filtering Models

Kalyan Kumar Jena [1], Sourav Kumar Bhoi [1], Tushar Kanta Malik [1], Kshira Sagar Sahoo [2,3], N Z Jhanjhi [4,*], Sajal Bhatia [5] and Fathi Amsaad [6]

[1] Department of Computer Science and Engineering, Parala Maharaja Engineering College, Berhampur 761003, Odisha, India
[2] Department of Computer Science and Engineering, SRM University, Amravati 522502, Andhra Pradesh, India
[3] Department of Computing Science, Umeå University, 901 87 Umeå, Sweden
[4] School of Computer Science, SCS Taylor's University, Subang Jaya 47500, Malaysia
[5] School of Computer Science and Engineering, Sacred Heart University, 3135 Easton Turnpike, Fairfield, CT 06825, USA
[6] Department of Computer Science and Engineering, Wright State University, 3640 Colonel Glenn Hwy, Dayton, OH 45435, USA
* Correspondence: noorzaman.jhanjhi@taylors.edu.my

Abstract: e-Learning is a sought-after option for learners during pandemic situations. In e-Learning platforms, there are many courses available, and the user needs to select the best option for them. Thus, recommender systems play an important role to provide better automation services to users in making course choices. It makes recommendations for users in selecting the desired option based on their preferences. This system can use machine intelligence (MI)-based techniques to carry out the recommendation mechanism. Based on the preferences and history, this system is able to know what the users like most. In this work, a recommender system is proposed using the collaborative filtering mechanism for e-Learning course recommendation. This work is focused on MI-based models such as K-nearest neighbor (KNN), Singular Value Decomposition (SVD) and neural network–based collaborative filtering (NCF) models. Here, one lakh of Coursera's course review dataset is taken from Kaggle for analysis. The proposed work can help learners to select the e-Learning courses as per their preferences. This work is implemented using Python language. The performance of these models is evaluated using performance metrics such as hit rate (HR), average reciprocal hit ranking (ARHR) and mean absolute error (MAE). From the results, it is observed that KNN is able to perform better in terms of higher HR and ARHR and lower MAE values as compared to other models.

Keywords: recommender system; machine intelligence; collaborative filtering; KNN; SVD; NCF

check for updates

Citation: Jena, K.K.; Bhoi, S.K.; Malik, T.K.; Sahoo, K.S.; Jhanjhi, N.Z.; Bhatia, S.; Amsaad, F. E-Learning Course Recommender System Using Collaborative Filtering Models. *Electronics* **2023**, *12*, 157. https://doi.org/10.3390/electronics12010157

Academic Editors: George Angelos Papadopoulos and Christos J. Bouras

Received: 9 October 2022
Revised: 27 November 2022
Accepted: 30 November 2022
Published: 29 December 2022

1. Introduction

The excessive quantity of digital information and video content for various subjects, and an increasing number of users have created a challenge for technical professionals to predict the preferences of learners. This can increase the demand for e-Learning recommender systems [1–19]. The e-Learning recommender system is a filtration strategy that suggest the contents that one user would like based on information that has been filtered from thumping dynamically generated information. The e-Learning recommender system [20–25] has an exceptional ability to scale back knowledge overload. In addition, during the time of pandemic, the smart e-Learning platform [26–29] was considered as an invaluable way for users to continue with their studies on basis of feedbacks [30]. So, a system that can provide better recommendations for e-Learning courses provides benefits to users.

The recommender system [1–28,31–37] not only helps its users to find the best probable product based on their preferences but also increases the efficiency of finding the product in a shorter timeframe. In this digital world, the recommender system plays a vital role. During the COVID-19 pandemic, when the education sector suffered due to social distancing requirements, these types of systems were able to recommend the best possible e-Learning courses for users based on their preferences. There are different types of recommender systems, such as book recommender systems, movie recommender systems, e-commerce recommender systems, etc. In the book recommender system, the book area unit is suggested to the browsers, considering the genres the user would prefer to browse or the authors and publications the learner would like to read. In the same manner in the film recommender system, the film area unit steers to the users toward the genres or works of certain administrators and production houses the user prefers. In an e-commerce recommender system, the recommender system helps to scale back the price of dealing and additionally helps in selecting the acceptable elements for the user in an online atmosphere.

Machine learning (ML) [29] has created the scope for machines to learn from their past events with the help of a huge amount of information and different algorithms. In the recommender system, the user can feed the system with user data, and with the help of certain algorithms, the proposed system recommends the e-Learning courses to the users. The filtration strategy is generally classified into two categories: content-based filtering and collaborative filtering. The content-based filtration strategy emphasizes the data provided by the items, whereas the collaborative filtration strategy focuses on the learner's behavior, comparing it with other learners' behavior.

In the collaborative filtering system [5,6,8,9,11,12,15,31,34,35], the data of multiple learners influence the result of the recommender system, whereas content-based filtering is concerned with the data of a specific user. Collaborative filtering is again divided into two broad categories: user-based collaborative filtering (UBCF) and item-based collaborative filtering (IBCF). In the UCBF system, learners similar to the given user to whom the e-Learning course needs to be recommended are identified, and e-Learning courses that that similar users have chosen are recommended. However, item-based collaborative filtering systems identify similar e-Learning courses and recommend them to the given user.

There are many applications of recommender systems, such as movie recommendations, book recommendations, etc. However, the current situation of the COVID-19 pandemic has led to a rise in demand for e-Learning courses. So, this work is focused on the e-Learning course recommender system. This system uses different algorithms, such as KNN, SVD and NCF, for analysis. The main objective of this work is to predict the e-Learning courses for an individual with higher accuracy.

The main contribution of this work is as follows:

- In this work, a collaborative filtering-based recommender system is proposed to recommend several e-Learning courses to the learners.
- This work is focused on one lakh of Coursera's course review dataset from Kaggle for analysis using KNN, SVD and NCF models for recommending relevant courses to the user based on their preferences. This will provide benefits to e-learning platforms, providing user friendly options for readers.
- The simulation of this work is carried out using Python 3.4 for evaluation of the collaborative filtering models: KNN, SVD and NCF. The dataset used is trained and tested using these models with a sampling of 60:40, 70:30 and 80:20 ratios.
- The performance of these models is evaluated using performance metrics such as MAE, HR and ARHR. From the results, it is found that the MAE, HR and ARHR values of the KNN model are (0.0130, 0.875 and 0.1346), (0.0136, 0.9187 and 0.1413) and (0.0142, 0.9646 and 0.1483) in 60:40, 70:30 and 80:20 training–testing scenarios, respectively. It is concluded that KNN is able to perform better in terms of higher HR and ARHR and lower MAE values as compared to other models.

The remaining portion of this work is described as follows. Section 2 describes the related works. Section 3 describes the methodology. Section 4 includes the results and discussion. In Section 5, this study is concluded.

2. Related Works

Different studies have been carried out related to recommender systems [1–28,31–37]. Tan et al. [1] focused on an e-Learning recommender system that helps learners to learn various topics without providing any personal information. This essential process of the recommendation algorithm helped the learner to connect with different online courses. Sharma et al. [2] described the various challenges of the techniques that are utilized for recommendations. The collaborative filtering (CF) mechanism was used in this work. Singhal et al. [3] focused on the role of deep learning in the e-Learning recommendation system. This study sought to determine whether deep learning improves the recommendation system or not. Shishehchi et al. [4] stated that due to the catastrophic effects of the COVID-19 pandemic, online courses are commonly used for education. A learner has to create a profile to access the courses. The profile is used to collect the details of the learner and to recommend the preferred courses to a particular learner. Wei et al. [5] focused on a smart system that scrutinizes the past interests of learners and predicts future courses. In this work, an attempt was made to resolve the complete cold start (CCS) and incomplete cold start (ICS) problems. Liu et al. [6] focused on the cooperative filtering recommendation algorithmic rule that supports the influence sets of e-Learning group behavior. Due to the use of memory-based KNN and UBCF, problems such as shortage of memory occurred, as the enormous number of users accessed an uncountable number of items. Due to this problem, IBCF was used. Khanal et al. [7] focused on the overview of e-Learning course recommendation systems. In ML, there are many necessary components, such as data sets, explicit ratings, learner behavior data, implicit ratings, results, etc., which can be helpful to implement recommendation mechanisms.

Youness et al. [8] focused on a recommendation approach for suggesting related e-Learning courses for beginners. The method was based on both social filtering (SF) and CF to recommend E-Learning courses that the user would like to access based on past learning and ratings given by the learners. Mawane et al. [9] stated that the traditional CF recommendation algorithmic rule cannot determine the helpful or necessary courses. Ghauth et al. [10] focused on the analysis of past data of the learners and developed an e-Learning recommendation system using CF and ratings to recommend the courses that will increase the performance and knowledge of learners. Bobadilla et al. [11] stated that users with a great deal of knowledge can affect the outcome of the whole E-Learning recommendation system more than learners with less knowledge. Mannan et al. [12] described the accurateness of a multilayer feed-forward artificial neural network (ANN). Here, the similarity measurement function was analyzed. Similarity was modelled between any two users as a function that comprises a series of adaptive weights and training of a neural network to minimize the weights. Gupta et al. [13] focused on a recommender system using SVD. First, it selects some pertinent contextual variables based on the contextual information of the users and their ratings for a class of entities. SVD is applied to extract the most relevant features corresponding to each and every entity, once the contextual variables are extracted.

Zheng et al. [14] focused on the regularized SVD (RSVD) method. In this work, it was reported that despite the non-convexness of RSVD, it possesses a closed form of a global minimized solution. Finally, RSVD was applied to the recommendation system, and experimental results showed that RSVD outperforms SVD significantly. Zhang et al. [15] described a method that incorporates the approximation of SVD into an expectation-maximization (EM) procedure to optimize the total computational cost by keeping intact the precise predictions. For privacy and security purposes, a new framework was proposed in the distributed recommendation system that allows the users to access their own rating profiles. Gong et al. [16] described an algorithm using collaborative filtering to solve problems effectively. Here, the outcome of SVD is used for filling the empty ratings; then, for the prediction of the unrated items, an item-based approach is used. Garanayak et al. [17] aim to build recommender systems with the help of the IBCF technique and K-means. The algorithm used in the recommender system field is the collaborative filtering technique.

Zriaa et al. [18] conducted a comparative study of KNN and K-means clustering to find better algorithms based on the prediction in the e-Learning recommender systems. MAE is mostly used to measure the effectiveness of algorithm performance in terms of accuracy. The lower the MAE value, the higher the accuracy of the prediction model. Navlani et al. [19] attempted to provide a better recommendation from the large available amount of data. They reported on the introduction of recommender systems, their functions, types, and techniques, the applications of the recommender system, content-based recommendations, collaborative-based recommendations and the evaluation of performance. Ghauth et al. [20] proposed a framework that recommends learning materials with a similar type of content that indicates the quality of the learning materials, based on the learner's ratings. A comprehensive set of experiments were conducted to measure the accuracy of the system and its impact on learners' performance.

From the above analysis, it is observed that few works have focused on e-Learning recommendation systems, and no method is able to recommend well in all scenarios. So, there is a need for the analysis of different methods for several situations.

3. Methodology

In this work, KNN, SVD and NCF models are used for e-Learning recommender systems. Here, datasets such as reviews.csv and reviwsbycourse.csv are merged, and the values where the user has not given any rating are represented as NaN. Sparsity and noise are removed from the merged dataset. Then, the KNN, SVD and NCF algorithms using Tensor flow Keras are used for analyzing the data and recommending the preferable e-Learning courses to the learners. The top n number of e-Learning courses is recommended as output based on cosine similarity distance computation. The generic workflow diagram of this work is shown in Figure 1. After acquiring the dataset, the preprocessing mechanism is applied to the data to convert it to the proper format. Then, KNN, SVD and NCF algorithms are used to recommend the top n similar courses. The overall steps involved in this work are mentioned as follows:

1. The datasets are loaded.
2. Data pre-processing is performed to select the required features needed to recommend the courses to beginners.
3. Exploratory data analysis (EDA) is performed.
4. Checking of sparsity is performed.
5. To reduce the sparsity, the csr matrix function is used, which excludes the null data and converts the not null data to an array form.
6. The data is fit into the KNN, SVD and NCF models.
7. Similarity measures are computed and courses are recommended to the user.

The KNN, SVD and NCF methods used in this work are described as follows.

3.1. KNN

This algorithm calculates the similarity between new data and the available data. Then, it puts the new data into a specific category. It uses different methodologies to determine the similarity between two data; the cosine similarity is used for achieving the desired output. Here, the data manipulation is performed using pandas. Mathematical and scientific calculations are done using the NumPy library. Graphical plotting and data visualization is done using the matplot library function. Scipy library is used to solve mathematical problems. Course ratings and preferences given by learners are collected by collaborative filtering. Then, the courses are suggested to the users based on similar tastes. The clustering mechanism is used for the improvement of the recommender system. Clustering refers to the grouping of like objects in a way that the objects from one cluster have similarities with each other but they have less similarity with the objects from different clusters. Clustering along with the KNN is applied to the reviews by course dataset for achieving optimized output. In the IBCF system, using the KNN algorithm, the name of the course is given as input, and the recommender system recommends the n number of similar courses to the user, whereas in item-based filtering systems, the user_id, number of similar users to consider and the number of courses that will be recommended are given as input. In both scenarios, the proposed recommender system recommends the top n number of similar courses. In the KNN algorithm, the similarity needs to be calculated among the courses using certain distance metrics. KNN has a few parameters for calculating similarity, such as Minkowski distance, Manhattan distance, Euclidean distance, Cosine distance, Jaccard distance, Hamming distance, Pearson-Baseline and Pearson. However, in this case, cosine distance is used for similarity calculation purposes. In KNN, there are several parameters for generating neighbors. Those parameters are KNN-Basic, KNN-Mean, KNN–Zscore and KNN-Baseline. Here, the KNN-Baseline parameter is used for generating neighbors. For ensuring accuracy, MSE, RMSE and HR can be used. Here, HR is used for accuracy purposes.

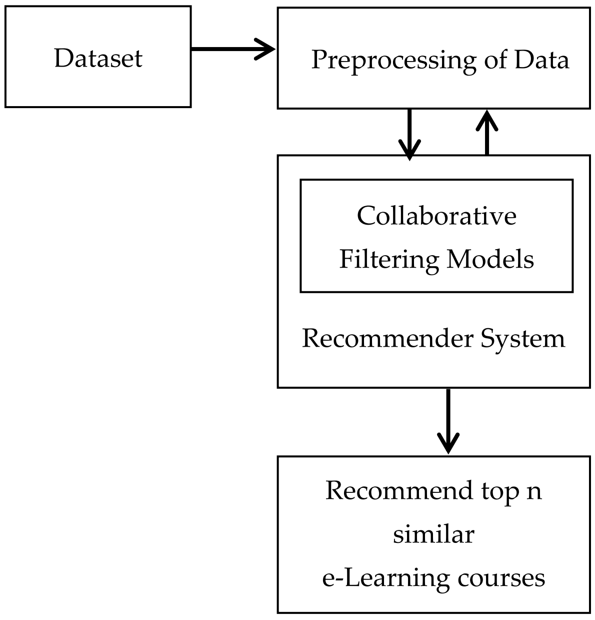
Dataset
Preprocessing of Data
Collaborative Filtering Models
Recommender System
Recommend top n similar e-Learning courses

Figure 1. Generic workflow diagram.

KNN is used to find the k-closest neighbors based on the similarity value. The similarity used here is cosine similarity. In the KNN algorithm, the brute force approach is used to compute the nearest neighbors, and the cosine metric is specified for this purpose. Here, the cosine similarity is calculated between rating vectors. It is measured by the cosine angle between two rating vectors. The formula for cosine similarity (cos θ) is specified in Equation (1).

$$\cos\theta = a\,.b/\,||a|| \times ||b|| \tag{1}$$

where θ represents the angle between the two rating vectors, a and b are two rating vectors, a.b represents the dot product of the vectors a and b, $||a|| \times ||b||$ represents the cross product of the vectors a and b, and $||a||$ and $||b||$ represent the length of two vectors a and b. The cosine distance can be computed as in Equation (2).

$$\text{cosine distance} = 1\text{-cosine similarity} \tag{2}$$

The value of cosine distance varies from 0 to 1. Here, 0 represents that the distance is negligible and thus the courses are similar, whereas 1 represents that the distance is greater, which means the courses are not similar. After applying cosine distance and KNN algorithm, the distance is calculated.

The input given for recommendation of the e-Learning course is User_id, number of similar users and number of courses. After the completion of the process, a specific number of courses are recommended to the user. The KNN algorithm is described in Algorithm 1, as follows.

Algorithm 1: KNN for e-Learning course recommendation

```
Input: Reviews, Reviews by course dataset
Output: e-Learning course recommendation
 1.  Begin
 2.  M = Merge (Reviews, Reviews by course)
 3.  Apply pivot function on M (Course_id,User_id,Ratings)
 4.  if Course_id = = rated
 5.      Assign 1 to the corresponding cell of M
 6.  else
 7.      Assign 0 to the corresponding cell of M
 8.  Compute M_final by considering updated M
 9.  Calculate sparsity
10.  Calculate csr matrix
11.  Compute csr_data=csr matrix(M_final)
12.  Compute knn_fit(csr_data)
13.  Calculate the cosine distance
14.  Sort the courses based on cosine distance
15.  Recommend the top n courses similar to the given course, where n is an integer value
     and n > 0
16.  End
```

Here, two datasets, Reviews and Reviews by course, are taken for the processing, which contain Course_id, Course_name, User_id and Ratings of courses. Then, the two datasets are merged, and the pivot function is applied. The pivot function shows each user–course interaction. As many courses are not rated by the learners, most of the cells in the dataset are filled with NaN values, which are not rated; in the next step, the NaN is filled with zero. Then, the sparsity is calculated, and it is fit into the csr matrix. Finally, it is fit into the KNN model. Then, a recommender function is defined, and a course name is given as a parameter of the function; then, KNN calculates the similar courses based on the cosine distance parameter and sorts the courses based on the cosine distance.

Cosine distance and cosine similarity are inversely proportional to each other, which means if the cosine distance increases, then the similarity decreases and the courses are less recommended; if the cosine distance decreases, then the similarity increases and the course is highly recommended. Finally, in the output, the top n similar courses along with their distances from the input course name are shown. The workflow diagram using the KNN algorithm is shown in Figure 2.

3.2. SVD

In the real world, there are many online user-centric recommendation systems, such as book recommendation systems, movie recommendation systems and music recommendation systems, which recommend further items based on the user's or viewer's interest. In the proposed work, predicting and recommending suitable courses to learners would be rated by learners. There are many algorithms used to solve this, and SVD is one that provides a brief initiation to the proposed recommender system to recommend the preferred courses. It uses a method of linear algebra that is specifically used for developing recommender systems. SVD is a technique that is used to reduce dimensionality. In the proposed e-Learning course recommender system, it is used as a technique of CF. It is a matrix factorization method. As SVD is used as a CF technique in the proposed recommender system, it uses a matrix structure. In the dataset, the rows of the matrix represent learners, and the columns represent the courses; the values of the matrix are the ratings given to the course by learners. Matrix factorization refers to the mathematical operations performed on matrices. For the proposed recommender system, datasets are loaded using the NumPy library function of python. The datasets revies.csv and reviewsbyuser.csv are merged. After combining the datasets, the utility matrix is formed. Then, the utility matrix is normalized across the courses.

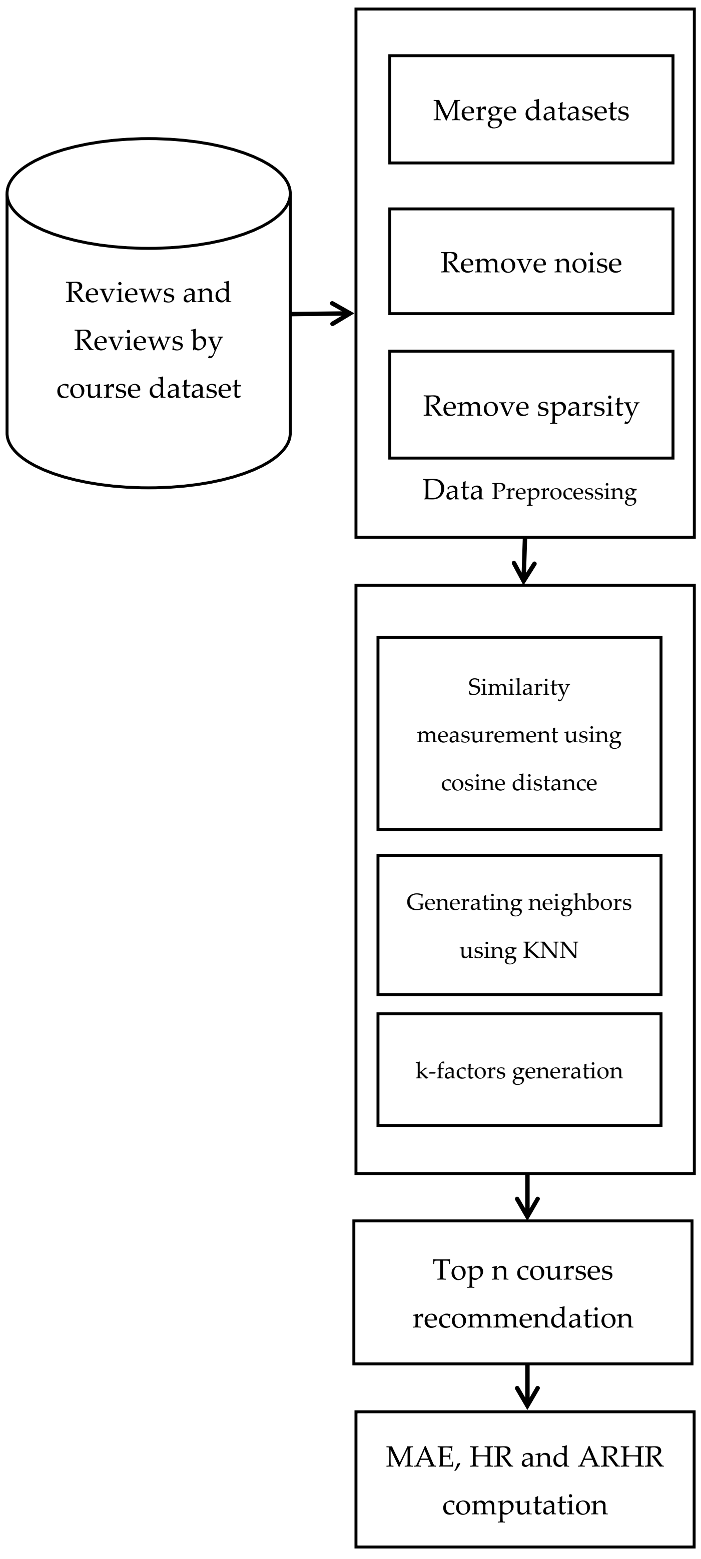
Reviews and Reviews by course dataset
Merge datasets
Remove noise
Remove sparsity
Data Preprocessing
Similarity measurement using cosine distance
Generating neighbors using KNN
k-factors generation
Top n courses recommendation
MAE, HR and ARHR computation

Figure 2. Workflow diagram of KNN.

Singular value decomposition helps in the factorization of the matrix in this case. From the factorization of the user–course rating matrix, the factors are found. One matrix is decomposed into three other matrices with the help of singular value decomposition, as specified in Equation (3).

$$H = IJK^M \tag{3}$$

where H is the a x b matrix; I is the a x c orthogonal matrix, which is a left singular matrix that represents the relationship between the learner and latent factors; J is the c x c diagonal matrix, which shows the strength of latent factors; V is the c x b orthogonal matrix. It is a right singular matrix and represents the similarity between courses and latent factors. Latent factors represent the characteristics of a course from the subject or area of the e-Learning course. The SVD reduces the dimension of the utility matrix H by removing its latent factors. It maps every learner and every course into a c-dimensional latent space. This mapping facilitates a clear representation of relationships between learners and courses. SVD represents the relationship between the learners and courses by mapping. Mapping is the technique where SVD maps each and every learner and each course into a c-dimensional latent space. Cosine similarity is calculated on the given data frame. By calculating the Einstein summation convention on the operands with the help of NumPy einsum, the requested number of e-Learning courses is extracted. Algorithm 2 represents the SVD algorithm.

Algorithm 2: SVD for e-learning course recommendation

```
Input: Reviews, Reviews by course dataset
Output: e-learning course recommendation
  1. Begin
  2. M = Merge (Reviews, Reviews by course)
  3. Refine_M = Apply group by mechanism on M (User_id, Course_id)
  4. Create utility_matrix by considering the length of unique User_id, Course_id
  5. Apply masking operation on utility_matrix
  6. Calculate the cosine distance by considering the utility_matrix
  7. Sort the courses based on cosine distance
  8. Recommend the top n courses similar to the given course
  9. End
```

Here, the two datasets Reviews and Reviews by course are taken, which contain Course_id, Course_name, User_id and Ratings of courses. Then, the two datasets are merged based on Course_id. The number of unique users and unique courses are then calculated. The course list and rating list are printed on the output screen, and a utility matrix is created for the available data and an empty array is created with the number of rows equal to the number of courses and the number of columns equal to the number of users. Here, rows represent courses and columns represent users. Then, the ith entry is taken in the user's list and 1 is subtracted to obtain the index. The same can be done for courses; however, a dictionary is defined to obtain the index. Then, the checking for the NaN values in the utility matrix can be done. Afterwards, the filled matrix is computed and fit into the SVD model. Here, a function is defined to evaluate the cosine similarity and sort the course similarity value. Then, the top n number of courses is returned. Afterwards, a function is defined that takes courses as input and returns suggestions for the user. In addition, a function can be created to deal with the situation in case the entered course is misplaced or wrong: it does not give any error; rather, it will show the relevant courses, starting with the first letter of the entered course, will ask for the valid course name, or will suggest the top n courses to the user. The workflow diagram using SVD is shown in Figure 3.

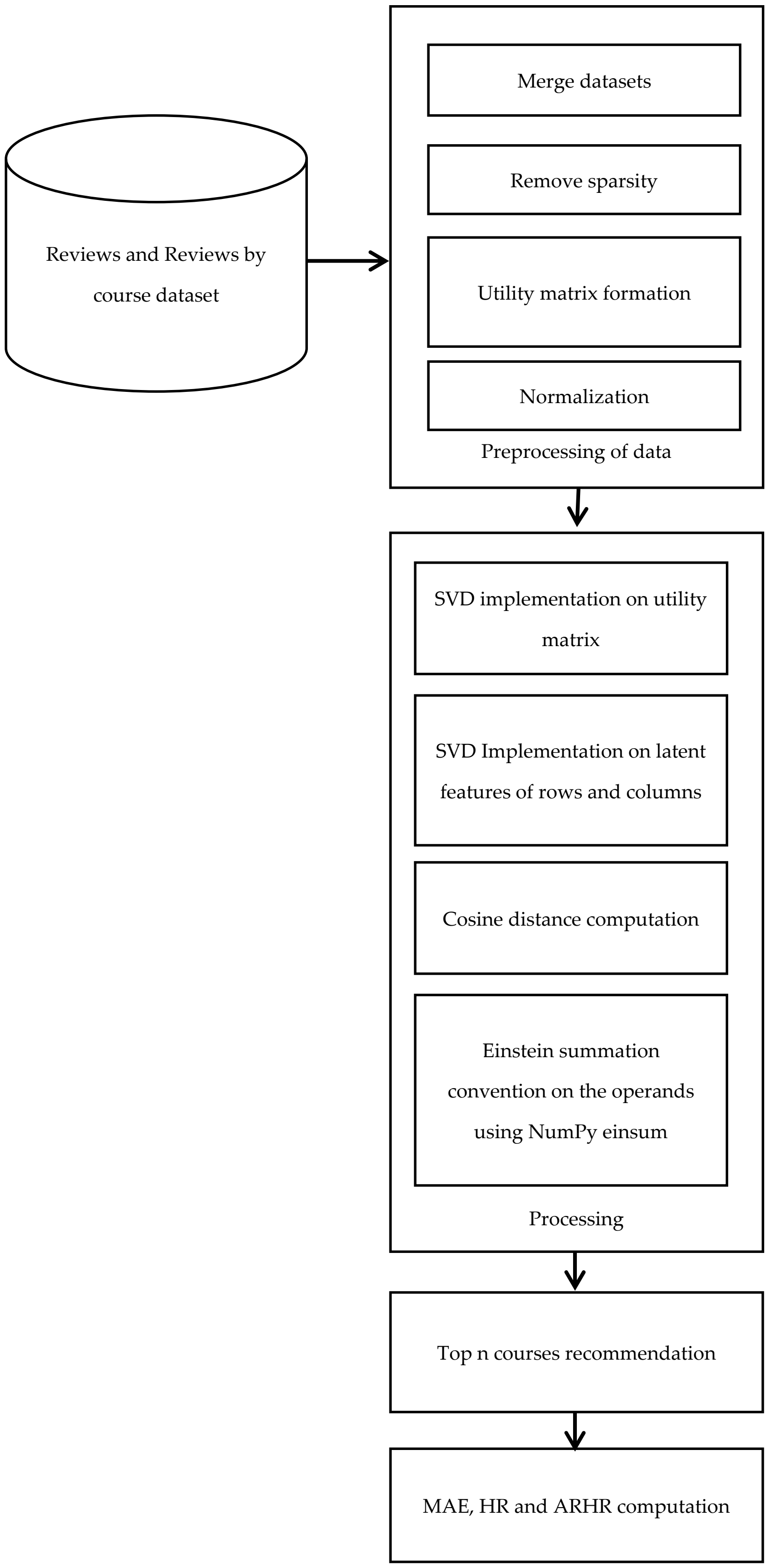
Reviews and Reviews by course dataset
Merge datasets
Remove sparsity
Utility matrix formation
Normalization
Preprocessing of data
SVD implementation on utility matrix
SVD Implementation on latent features of rows and columns
Cosine distance computation
Einstein summation convention on the operands using NumPy einsum
Processing
Top n courses recommendation
MAE, HR and ARHR computation

Figure 3. Workflow diagram of SVD.

3.3. NCF

In this work, item-based CF is used. To find a better result than other algorithms, Neural CF can be applied, which uses the product of the learner–course matrix to determine learner–course interactions and replaces this with a neural architecture in the recommender system. SVD is a factorization technique used for the decomposition of a matrix. It is used to decrease the size of the matrix, which in turn reduces the calculation. Deep neural networks have achieved great success in the prediction and classification of tasks. Here, the main goal is to discover the use of neural networks for building the e-Learning recommender system. The dataset used in the e-Learning course recommender system is Coursera's one lakh data, which has been used for predicting the ratings. NCF solves the problem by modelling learner–course interaction through the help of a neural network.

Here, the libraries are loaded into the recommender system for performing crucial operations. Numpy is loaded for performing mathematical operations. The Pandas library is loaded for data analysis purposes. The Tensorflow library is loaded for classification, perception and prediction purposes. Keras is a neural network library that is loaded for implementing and using deep learning models. Pathlib is loaded for working with files and directories. The Matplot library is loaded for plotting graphs. Then, datasets such as reviews.csv and reviewsbycourse.csv are uploaded. The preprocessing is performed on the datasets. Preprocessing of data includes extraction of files and encoding of users and courses as integer indices. The dataset is split for training and testing purposes. Here, large amounts of data are considered for accuracy purposes, which means determining whether the e-Learning course recommender system predicts preferred e-Learning courses or not.

In the next phase, the neural network models are created. The Keras library has made the task of creating a model easy for the users. The neural network model consists of layers such as input, embedding and output layers.

The input layer takes learner and course vectors as input. For the proposed recommender system, the course is the item vector. The embedding of users and courses consists of an embedding layer. Embedding refers to the random initial values that will be updated during the training phase. This is same as the latent factors in the matrix factorization algorithm. This model focuses on obtaining the best values for embedding by reducing or minimizing the error between actual and predicted values. The predicted values or courses are provided by the output layer. The output layer consists of one or more neurons. However, in the proposed recommender system, the output layer consists of a single neuron as output, which will be the predicted value of the course by the user. The user and course vector is created in the first stage. These concatenated vectors deal with the neural network containing the layers as mentioned below.

The first concealed layer consists of 128 neurons, the second hidden layer consists of 32 neurons, and the third layer consists of 1 neuron, which provides the predicted value provided by the user to the course. Similar courses have similar embedding vectors. So, with the help of the embedding layer, it can recommend similar courses to the learners. This is required to evaluate the dot product of the learner embedding and course embedding, which provides the result. By including the sigmoid function at the end, the neural network morphs the probability to a value between 0 and 1.

In this model, a match score is calculated between the user and course embedding via a dot product and adds per-course and per-learner bias. It scales the match score to [0, 1] interval via sigmoid. The predicted results for interaction between the learner and course are specified in Equation (4).

$$M_{lj} = f(l, j \mid \alpha) \tag{4}$$

where M(l, j) represents the results predicted for interaction between learner l and course j, α represents the parameters of the model and f represents the interaction function. In order to compute alpha, the objective function should be optimized.

After the creation of the model, it is trained on the training dataset. Any number of epochs can be taken for accuracy purposes, but here five epochs are taken for processing. The embedding vector is updated in order to achieve the predicted values close to the actual value. The error between the predicted rating and the actual rating over the entire training dataset is known as the loss. After the training phase, the predicted values are generated for the test dataset for user-id and course-id. This helps in computing the predicted rating.

Here, two datasets, Reviews and Reviews by course, are taken, which contain Course_id, Course_name, User_id and Ratings of courses. First, the data are loaded, and the preprocessing mechanism is applied. Preprocessing is performed to encode learners and courses as integer indices. Then, the number of courses, number of users, maximum rating and minimum rating are computed. Later on, the minimum and maximum ratings are used to normalize the learner's ratings. Afterwards, the preparation of training and validation data is focused. The ratings of learners are normalized between 0 and 1 for training purposes. Here, we assume training on 80% of the data and validation on 20% of the data, as well as 70:30 and 60:40 ratios.

Then, the model is built and embedded into the learners and courses into 50-dimensional vectors. A similarity score is computed between learner and course embeddings. A similar score is maintained between the [0,1] interval. This is done by the sigmoid function. Then, the data is split, and training and testing are performed. The loss and data split is plotted in a graph. Here, the model requires a user_id and shows the course_id and course names. In the backend, it finds the courses that are not seen by the user and finally recommends the required courses to the users. The NCF algorithm is described in Algorithm 3. The workflow diagram using NCF is shown in Figure 4.

Algorithm 3: NCF for e-learning course recommendation

```
Input: Reviews, Reviews by course dataset
Output: e-learning course recommendation
    1.   Begin
    2.   M = Merge (Reviews, Reviews by course)
    3.   Encode M (User_id)
    4.   Encode M (Course_id)
    5.   Create M [User] by assigning Encoded M (User_id) values
    6.   Create M [Course] by assigning Encoded M (Course_id) value
    7.   Compute number_of_users = length (M [User])
    8.   Compute number_of_courses = length (M [Course])
    9.   Apply training and testing mechanism on computed number_of_users and
         number_of_courses
    10.  Apply embedding mechanism with embedding_size = 50
    11.  Calculate the cosine distance using the sigmoid function
    12.  Sort the courses based on cosine distance
    13.  Recommend the top n courses similar to the given course
    14.  End
```

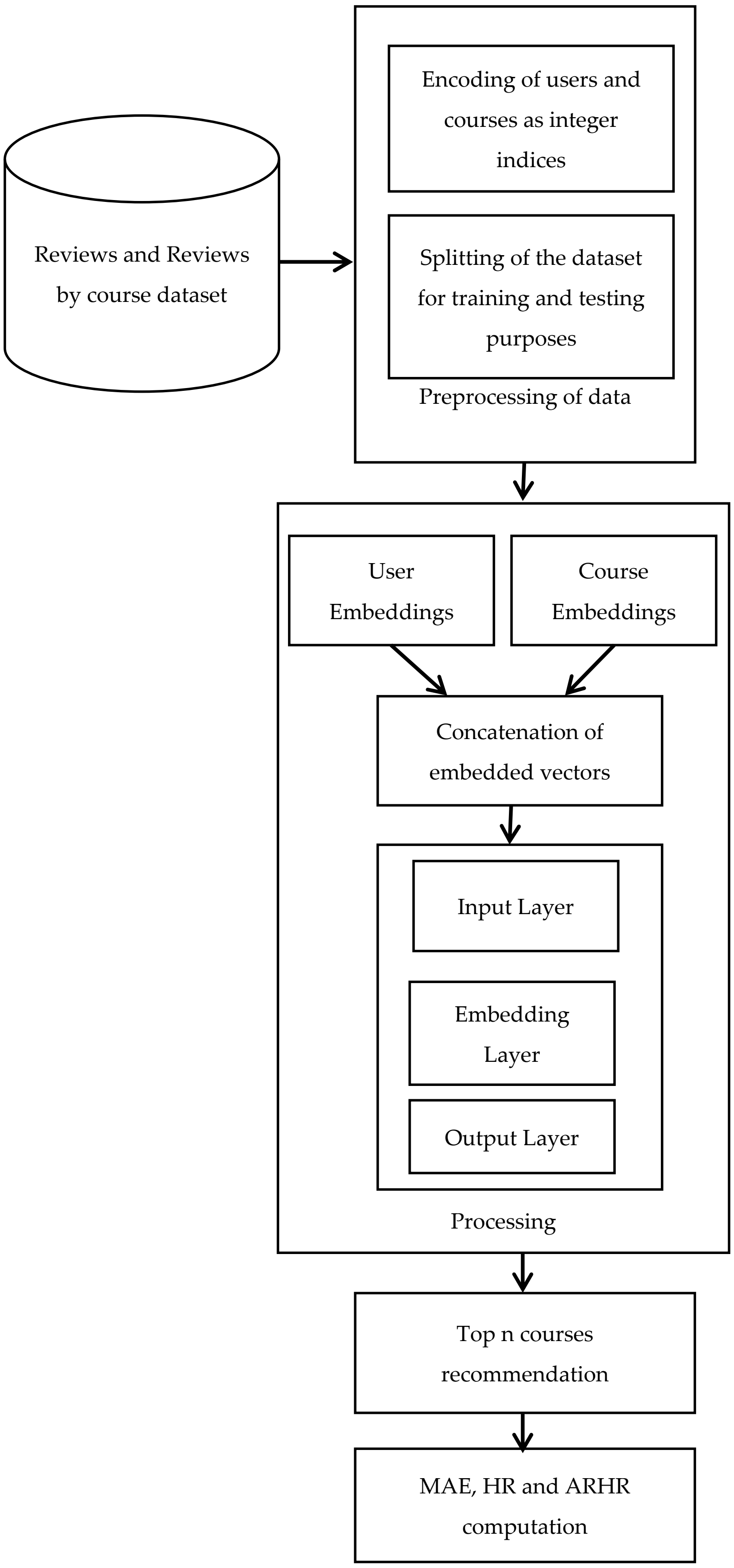
Reviews and Reviews by course dataset
Encoding of users and courses as integer indices
Splitting of the dataset for training and testing purposes
Preprocessing of data
User Embeddings
Course Embeddings
Concatenation of embedded vectors
Input Layer
Embedding Layer
Output Layer
Processing
Top n courses recommendation
MAE, HR and ARHR computation

Figure 4. Workflow diagram of NCF.

4. Results and Discussion

The proposed approach of the recommender system for e-Learning courses using ML techniques is implemented in Python programming language using the Jupyter notebook, which is available in the Anaconda platform. Anaconda is the world's most popular data science platform. There are many integrated development environments (IDEs) available in the Anaconda platform, such as spyder, eclipse, Jupyter notebook, etc.; here, Jupyter notebook is used as it is easy to use, easy to implement and user friendly. Here, the Python programming language is used.

Coursera's one lakh dataset was used for building the e-Learning recommender system from Kaggle [30]. This includes two datasets: reviews.csv and reviewsbycourse.csv. In the review dataset, there are three columns: the id column, review column and label column. There are 107k rows in this dataset, which consist of 107k unique user ids. The review column in this dataset provides 100,038 unique values. Similarly, the label column provides five different labels. In the reviews by course dataset, there are two columns: the review column and label column. A total of 123,243 unique values are present in the review column.

In this section, the results, performance evaluation parameters and their uses are discussed. Python is one of the most used programming languages because of the availability of several libraries and classes. While implementing the models, several libraries, including NumPy, pandas, matplotlib, sklearn, Keras, etc., are used. The NumPy library is used for working with arrays and matrices. The Pandas library is used for loading, reading and working on the dataset. The Matplotlib library is used for the data visualization. The Sklearn library has many useful tools for ML and statistical modelling purposes, whereas Keras is an open-source library used to work with an artificial neural network. It requires minimum human effort for developing neural networks.

The system used while implementing this model has the following characteristics. The operating system used is MS Windows 10 Home, and the model of the system is HP 240 G5 Notebook PC. The processor of the system is Intel(R) Pentium(R) CPU A1020 @ 2.41 GHz, 2408 MHz, 4 Core, 4 Logical Processor, which has 4GB RAM.

The performance of the models is calculated using performance parameters such as MAE, HR and ARHR, which are described as follows.

- MAE: This is the difference between the actual output value (AOV) and the predicted output value (POV). It is represented in Equation (5) by considering the total number of test cases (TNTC).

$$\text{MAE} = (1/\text{TNTC}) \sum |\text{AOV-POV}| \tag{5}$$

- HR: This is the ratio of number of course hits in the test to the sum of all the hits and misses. It is represented in Equation (6) by considering the number of course hits (NCH) and TNTC.

$$\text{HR} = \text{NCH}/\text{TNTC} \tag{6}$$

- ARHR: This is the ratio between the sum of reciprocals of the rank of each hit (SRR) and the TNTC taken. It is represented in Equation (7).

$$\text{ARHR} = \text{SRR}/\text{TNTC} \tag{7}$$

4.1. Results

The performance results of KNN, SVD and NCF models are shown in Tables 1–3 and Figures 5–16. Tables 1–3 represent the performance of each model in terms of MAE, HR and ARHR in 60:40, 70:30 and 80:20 training–testing scenarios. Figures 5–7 represent the performance results of each model in terms of MAE, HR and ARHR, respectively, in 60:40 training–testing scenarios. Figure 8 describes the comparative results of KNN, SVD and NCF models in terms of MAE, HR and ARHR in 60:40 training–testing scenarios. Figures 9–11 represent the performance results of each model in terms of MAE, HR and

ARHR, respectively, in 70:30 training–testing scenarios. Figure 12 describes the comparative results of KNN, SVD and NCF models in terms of MAE, HR and ARHR in 70:30 training–testing scenarios. Figures 13–15 represent the performance results of each model in terms of MAE, HR and ARHR, respectively, in 80:20 training–testing scenarios. Figure 16 describes the comparative results of KNN, SVD and NCF models in terms of MAE, HR and ARHR in 80:20 training–testing scenarios. In this work, the top n courses are recommended by considering the n value as 10.

Table 1. Performance evaluation of different models (in units) in 60:40 training–testing scenario.

Model	MAE	HR	ARHR
KNN	0.0130	0.875	0.1346
SVD	0.0237	0.725	0.1115
NCF	0.0156	0.850	0.1307

Table 2. Performance evaluation of different models (in units) in 70:30 training–testing ratio.

Model	MAE	HR	ARHR
KNN	0.0136	0.9187	0.1413
SVD	0.0248	0.7612	0.1171
NCF	0.0167	0.8925	0.1372

Table 3. Performance evaluation of different models (in units) in 80:20 training–testing ratio.

Model	MAE	HR	ARHR
KNN	0.0142	0.9646	0.1483
SVD	0.0260	0.7992	0.1229
NCF	0.0175	0.9371	0.1441

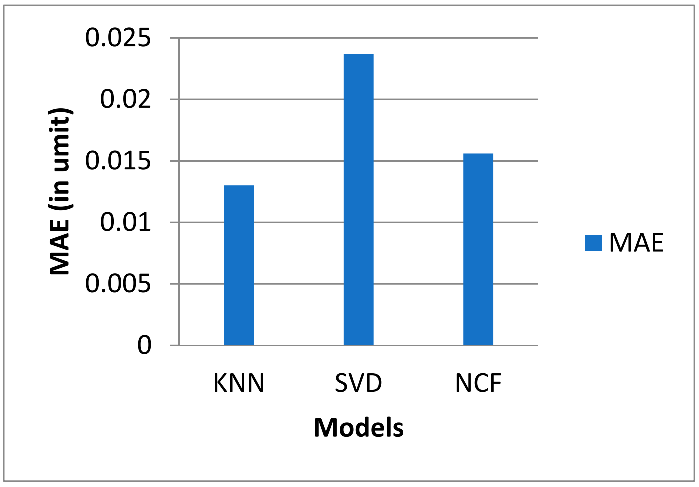

Figure 5. Performance results of different models in terms of MAE in 60:40 training–testing scenario.

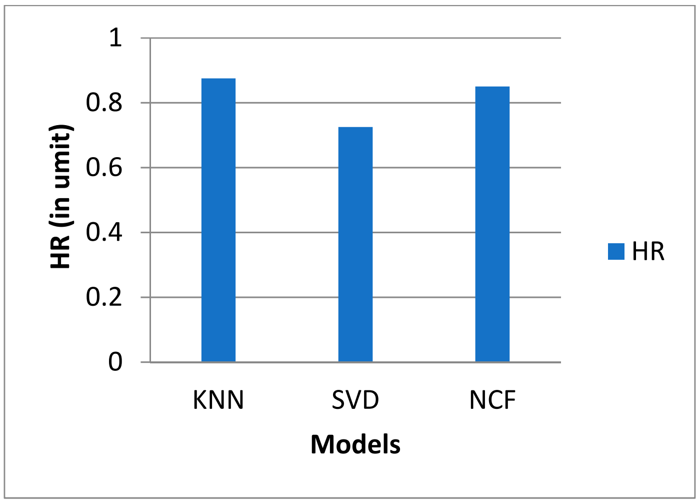

Figure 6. Performance results of different models in terms of HR in 60:40 training–testing scenario.

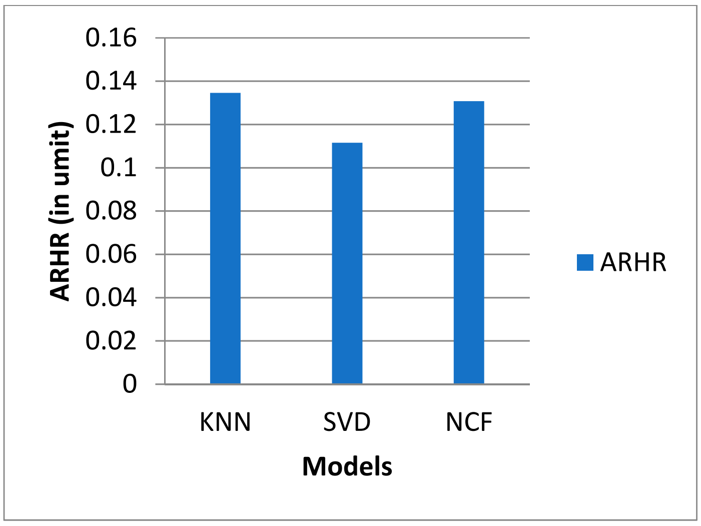

Figure 7. Performance results of different models in terms of ARHR in 60:40 training–testing scenario.

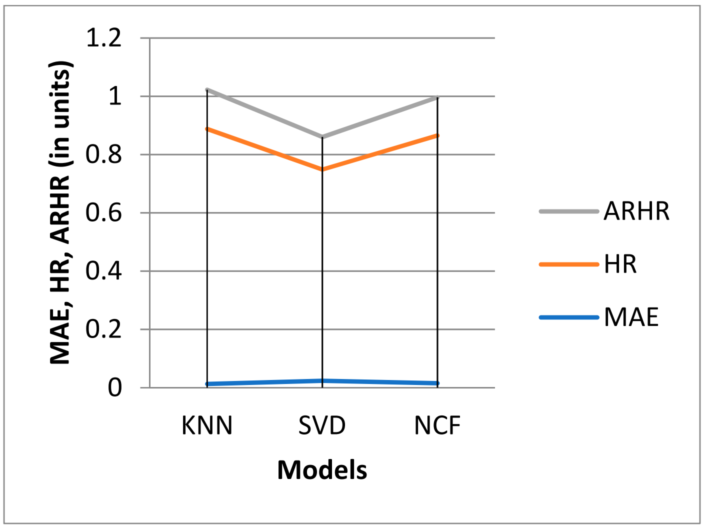

Figure 8. Comparative results of different models in terms of MAE, HR and ARHR in 60:40 training–testing scenario.

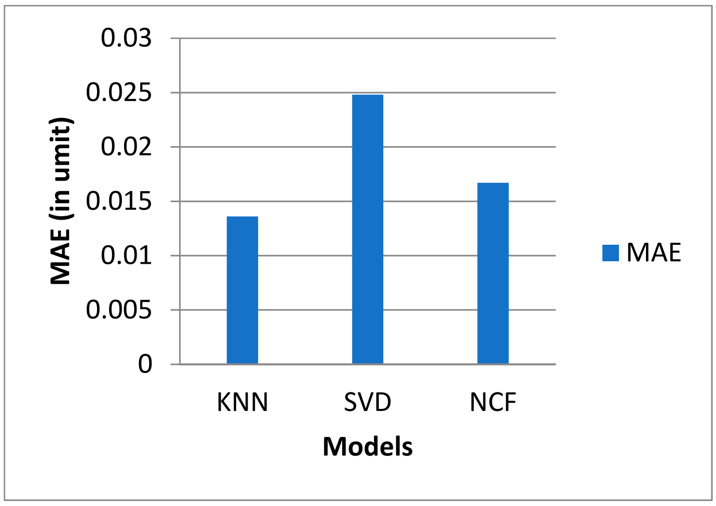

Figure 9. Performance results of different models in terms of MAE in 70:30 training–testing scenario.

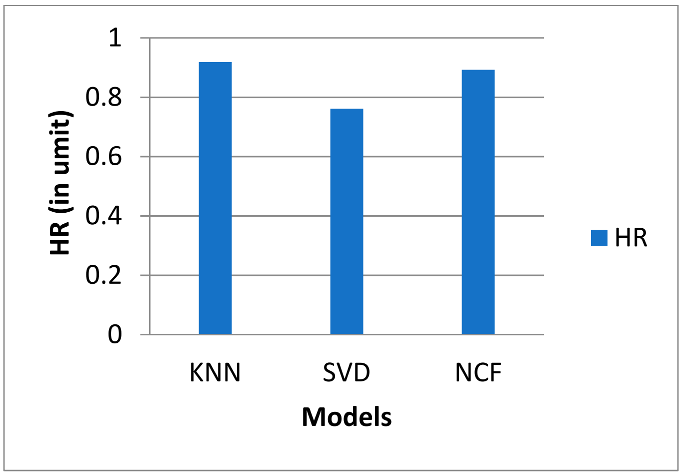

Figure 10. Performance results of different models in terms of HR in 70:30 training–testing scenario.

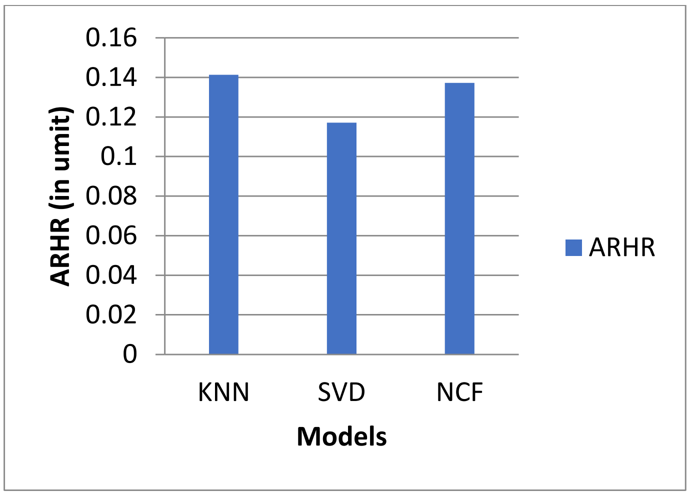

Figure 11. Performance results of different models in terms of ARHR in 70:30 training–testing scenario.

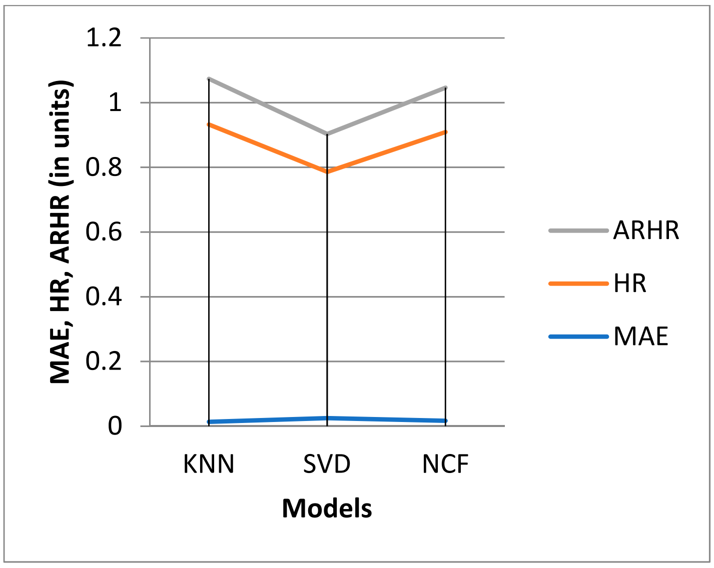

Figure 12. Comparative performance results of different models in terms of MAE, HR and ARHR in 70:30 training–testing scenario.

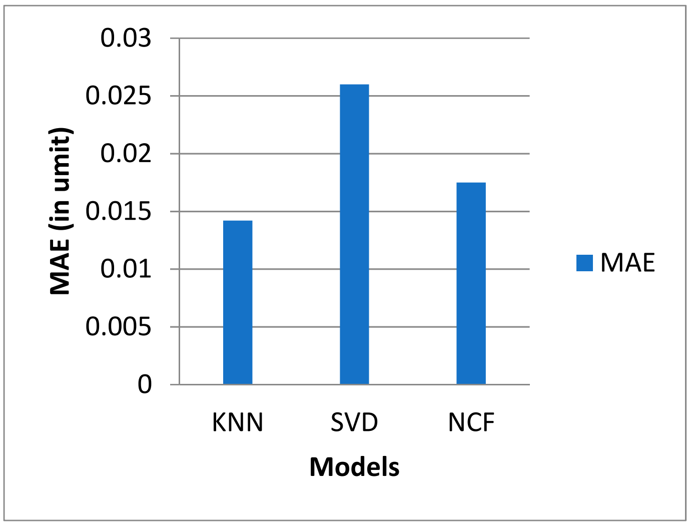

Figure 13. Performance results of different models in terms of MAE in 80:20 training–testing scenario.

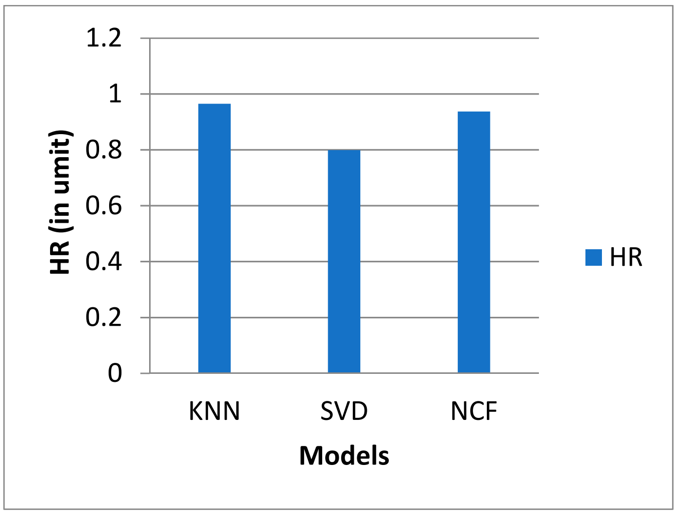

Figure 14. Performance results of different models in terms of HR in 80:20 training–testing scenario.

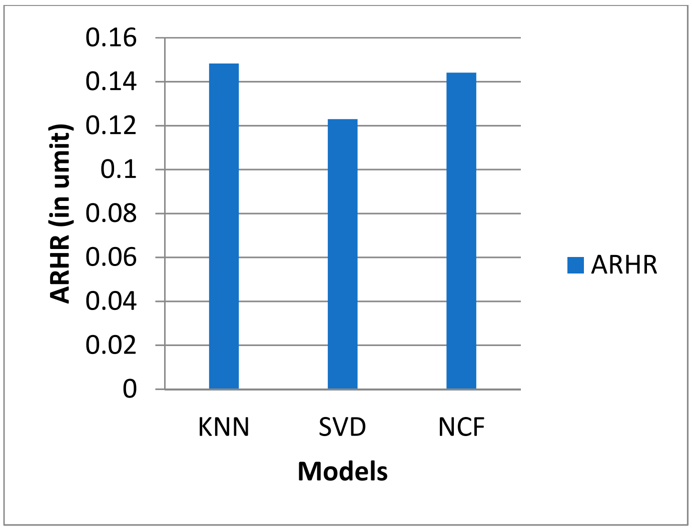

Figure 15. Performance results of different models in terms of ARHR in 80:20 training–testing scenario.

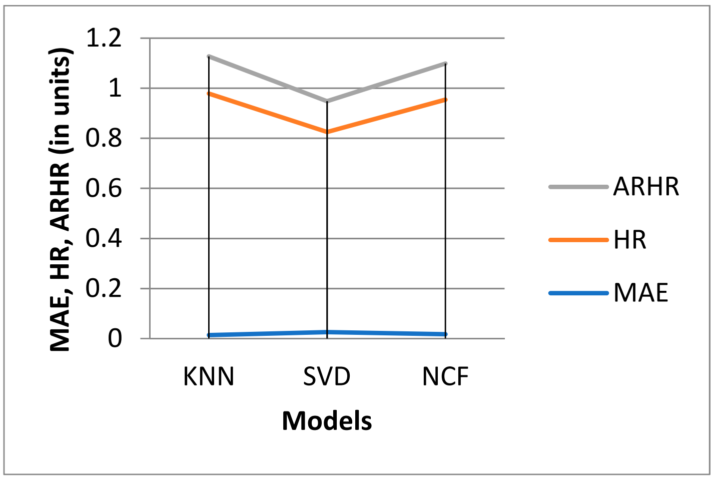

Figure 16. Comparative results of different models in terms of MAE, HR and ARHR in 80:20 training–testing scenario.

4.2. Discussion

From Tables 1–3 and Figures 5–16, it is observed that the MAE values (in units) of KNN, SVD and NCF in 60:40 training–testing scenarios are 0.0130, 0.0237 and 0.0156, respectively. The HR values (in units) of KNN, SVD and NCF in 60:40 training–testing scenarios are 0.875, 0.725 and 0.850, respectively. The ARHR values (in units) of KNN, SVD and NCF in 60:40 training–testing scenarios are 0.1346, 0.1115 and 0.1307, respectively. The MAE values of KNN, SVD and NCF in 70:30 training–testing scenarios are 0.0136, 0.0248 and 0.0167, respectively. The HR values of KNN, SVD and NCF in 70:30 training–testing scenarios are 0.9187, 0.7612 and 0.8925, respectively. The ARHR values of KNN, SVD and NCF in 70:30 training–testing scenarios are 0.1413, 0.1171 and 0.1372, respectively. The MAE values of KNN, SVD and NCF in 80:20 training–testing scenarios are 0.0142, 0.0260 and 0.0175, respectively. The HR values of KNN, SVD and NCF in 80:20 training–testing scenarios are 0.9646, 0.7992 and 0.9371, respectively. The ARHR values of KNN, SVD and NCF in 80:20 training–testing scenarios are 0.1483, 0.1229 and 0.1441, respectively. From the above analysis, it is observed that KNN is able to provide better recommendation results in terms of MAE, HR and ARHR as compared to SVD and NCF models. However, the NCF model is not able to provide better recommendation results as compared to KNN and SVD models in these scenarios. SVD is able to provide intermediate recommendation results as compared to KNN and NCF models. This model can recommend better e-Learning course for a user as per the demand with less MAE, and high HR and ARHR. The limitation of the model is that it has been tested with conventional models with a new e-Learning dataset. The models can be improved to increase the accuracy of recommendations. The main applications of this work are the selection of books, subjects, courses, informative videos, websites, conferences, discussion forums, webinars, learning software, e-Learning platforms, etc.

5. Conclusions

In this work, a collaborative filtering-based recommender system is proposed to recommend e-Learning courses to learners. To recommend the e-Learning courses to the learner, the collaborative filtering models KNN, SVD and NCF are used. To compute the performance of these models, the three performance parameters MAE, HR and ARHR are used in this work. From the results, it is concluded that KNN is able to perform better in terms of higher HR and ARHR and lower MAE values as compared to other models in the mentioned scenarios. The (MAE, HR and ARHR) values of the KNN model are (0.0130, 0.875 and 0.1346), (0.0136, 0.9187 and 0.1413), (0.0142, 0.9646 and 0.1483) in 60:40, 70:30 and 80:20 training–testing scenarios, respectively. This work can help the learners to select the e-Learning courses as per their preferences. This work can be extended to develop enhanced and hybrid models to provide better recommendation results. This work can also be applied in several deep learning models to explore the analysis results.

Author Contributions: Conceptualization, K.K.J., S.K.B. and T.K.M.; methodology, K.K.J., S.K.B. and T.K.M.; software, K.K.J., S.K.B., T.K.M. and K.S.S.; formal analysis, K.K.J., S.K.B., T.K.M. and K.S.S.; investigation, K.K.J., S.K.B., T.K.M., K.S.S., N.Z.J., S.B. and F.A.; resources, K.K.J., S.K.B. and T.K.M.; data curation, K.K.J., S.K.B. and T.K.M.; writing—original draft preparation, K.K.J., S.K.B. and T.K.M.; writing—review and editing, K.K.J., S.K.B., K.S.S., N.Z.J., S.B. and F.A.; visualization, K.K.J., S.K.B., K.S.S., N.Z.J., S.B. and F.A.; supervision, K.K.J., S.K.B.; project administration, K.K.J., S.K.B., N.Z.J., S.B. and F.A.; funding acquisition, N.Z.J., S.B. and F.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not Applicable.

Data Availability Statement: Data are available on request.

Acknowledgments: The authors thank Parala Maharaja Engineering College, Berhampur, Odisha, India, for providing the facility to conduct the research work. They also thank Tushar Kanta Malik for overall coordination with team members to carry out this research work under the guidance of Kalyan Kumar Jena and Sourav Kumar Bhoi.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Tan, H.; Guo, J.; Li, Y. E-learning Recommendation System. In Proceedings of the International Conference on Computer Science and Software Engineering, Wuhan, China, 12–14 December 2008; pp. 430–433.
2. Sharma, L.; Gera, A. A survey of recommendation system. *Int. J. Eng. Trends Technol.* **2013**, *4*, 1989–1992.
3. Singhal, A.; Sinha, P.; Pant, R. Use of deep learning in modern recommendation system: A summary of recent works. *arXiv* **2017**, arXiv:1712.07525. [CrossRef]
4. Shishehchi, S.; Banihashem, S.Y.; Zin, N.A.M. A proposed semantic recommendation system for e-learning: A rule and ontology based e-learning recommendation system. In Proceedings of the International Symposium on Information Technology, Kuala Lumpur, Malaysia, 15–17 June 2010; pp. 1–5.
5. Wei, J.; He, J.; Chen, K.; Zhou, Y.; Tang, Z. Collaborative filtering and deep learning based recommendation system for cold start items. *Expert Syst. Appl.* **2017**, *69*, 29–39. [CrossRef]
6. Liu, X. A collaborative filtering recommendation algorithm based on the influence sets of e-learning group's behavior. *Clust. Comput.* **2019**, *22*, 2823–2833. [CrossRef]

7. Khanal, S.S.; Prasad, P.W.C.; Alsadoon, A.; Maag, A. A systematic review: Machine learning based recommendation systems for e-learning. *Educ. Inf. Technol.* **2020**, *25*, 2635–2664. [CrossRef]
8. Madani, Y.; Erritali, M.; Bengourram, J.; Sailhan, F. Social Collaborative Filtering Approach for Recommending Courses in an E-learning Platform. *Procedia Comput. Sci.* **2019**, *151*, 1164–1169. [CrossRef]
9. Mawane, J.; Naji, A.; Ramdani, M. Unsupervised Deep Collaborative Filtering Recommender System for E-Learning Platforms. In Proceedings of the International Conference on Smart Applications and Data Analysis, Marrakesh, Morocco, 25–26 June 2020; Springer: Cham, Switzerland, 2020; pp. 146–161.
10. Ghauth, K.I.; Abdullah, N.A. Measuring learner's performance in e-learning recommender systems. *Australas. J. Educ. Technol.* **2010**, *26*, 764–774. [CrossRef]
11. Bobadilla JE SU, S.; Serradilla, F.; Hernando, A. Collaborative filtering adapted to recommender systems of e-learning. *Knowl. -Based Syst.* **2009**, *22*, 261–265. [CrossRef]
12. Mannan, N.B.; Sarwar, S.M.; Elahi, N. A new user similarity computation method for collaborative filtering using artificial neural network. In Proceedings of the International Conference on Engineering Applications of Neural Networks, Sofia, Bulgaria, 5–7 September 2014; Springer: Cham, Switzerland,, 2014; pp. 145–154.
13. Gupta, R.; Jain, A.; Rana, S.; Singh, S. Contextual information based recommender system using singular value decomposition. In Proceedings of the 2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI), IEEE, Mysore, India, 22–25 August 2013; pp. 2084–2089.
14. Zheng, S.; Ding, C.; Nie, F. Regularized singular value decomposition and application to recommender system. *arXiv* **2018**, arXiv:1804.05090.
15. Zhang, S.; Wang, W.; Ford, J.; Makedon, F.; Pearlman, J. Using singular value decomposition approximation for collaborative filtering. In Proceedings of the Seventh IEEE International Conference on E-Commerce Technology (CEC'05), IEEE, Munich, Germany, 19–22 July 2005; pp. 257–264.
16. Gong, S.; Ye, H.; Dai, Y. Combining singular value decomposition and item-based recommender in collaborative filtering. In Proceedings of the 2009 Second International Workshop on Knowledge Discovery and Data Mining, IEEE, Moscow, Russia, 23–25 January 2009; pp. 769–772.
17. Garanayak, M.; Mohanty, S.N.; Jagadev, A.K.; Sahoo, S. Recommender system using item based collaborative filtering (CF) and K-means. *Int. J. Knowl. -Based Intell. Eng. Syst.* **2019**, *23*, 93–101. [CrossRef]
18. Zriaa, R.; Amali, S. A Comparative Study Between K-Nearest Neighbors and K-Means Clustering Techniques of Collaborative Filtering in e-Learning Environment. In Proceedings of the Third International Conference on Smart City Applications, Tetouan, Morocco, 10–11 October 2018; Springer: Cham, Switzerland, 2020; pp. 268–282.
19. Navlani, A.; Dadhich, N. Recommender System. In *Applying Predictive Analytics Within the Service Sector*; IGI Global: Hershey, PA, USA, 2017; pp. 176–197.
20. Ghauth, K.I.; Abdullah, N.A. Learning materials recommendation using good learners' ratings and content-based filtering. *Educ. Technol. Res. Dev.* **2010**, *58*, 711–727. [CrossRef]
21. Abdi, S.; Khosravi, H.; Sadiq, S.; Demartini, G. Evaluating the quality of learning resources: A learnersourcing approach. *IEEE Trans. Learn. Technol.* **2021**, *14*, 81–92. [CrossRef]
22. Bhaskaran, S.; Marappan, R.; Santhi, B. Design and analysis of a cluster-based intelligent hybrid recommendation system for e-learning applications. *Mathematics* **2021**, *9*, 197. [CrossRef]
23. Guruge, D.B.; Kadel, R.; Halder, S.J. The state of the art in methodologies of course recommender systems—A review of recent research. *Data* **2021**, *6*, 18. [CrossRef]
24. Rahhali, M.; Oughdir, L.; Jedidi, Y.; Lahmadi, Y.; El Khattabi, M.Z. E-learning Recommendation System Based on Cloud Computing. In *WITS*; Springer: Singapore, 2022; pp. 89–99.
25. Agarwal, A.; Mishra, D.S.; Kolekar, S.V. Knowledge-based recommendation system using semantic web rules based on Learning styles for MOOCs. *Cogent Eng.* **2022**, *9*, 2022568. [CrossRef]
26. Rahayu, N.W.; Ferdiana, R.; Kusumawardani, S.S. A systematic review of ontology use in E-Learning recommender system. *Comput. Educ. Artif. Intell.* **2022**, *3*, 100047. [CrossRef]
27. Ali, S.; Hafeez, Y.; Humayun, M.; Jamail NS, M.; Aqib, M.; Nawaz, A. Enabling recommendation system architecture in virtualized environment for e-learning. *Egypt. Inform. J.* **2022**, *23*, 33–45. [CrossRef]
28. Yang, X.; Tan, L. The Construction of Accurate Recommendation Model of Learning Resources of Knowledge Graph under Deep Learning. *Sci. Program.* **2022**, *2022*, 1010122. [CrossRef]
29. Sen, P.C.; Hajra, M.; Ghosh, M. Supervised classification algorithms in machine learning: A survey and review. In *Emerging Technology in Modelling and Graphics*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 99–111.
30. Available online: https://www.kaggle.com/septa97/100k-courseras-course-reviews-dataset (accessed on 10 November 2021).
31. Tahir, S.; Hafeez, Y.; Abbas, M.A.; Nawaz, A.; Hamid, B. Smart Learning Objects Retrieval for E-Learning with Contextual Recommendation based on Collaborative Filtering. *Educ. Inf. Technol.* **2022**, *27*, 8631–8668. [CrossRef]
32. Amane, M.; Aissaoui, K.; Berrada, M. ERSDO: E-learning Recommender System based on Dynamic Ontology. *Educ. Inf. Technol.* **2022**, *27*, 7549–7561. [CrossRef]
33. Liu, T.; Wu, Q.; Chang, L.; Gu, T. A review of deep learning-based recommender system in e-learning environments. *Artif. Intell. Rev.* **2022**, *55*, 5953–5980. [CrossRef]

34. Do, P.M.; Nguyen, T.T. Semantic-enhanced neural collaborative filtering models in recommender systems. *Knowl. -Based Syst.* **2022**, *257*, 109934. [CrossRef]
35. Iwendi, C.; Ibeke, E.; Eggoni, H.; Velagala, S.; Srivastava, G. Pointer-based item-to-item collaborative filtering recommendation system using a machine learning model. *Int. J. Inf. Technol. Decis. Mak.* **2022**, *21*, 463–484. [CrossRef]
36. Kundu, S.S.; Sarkar, D.; Jana, P.; Kole, D.K. Personalization in Education Using Recommendation System: An Overview. *Comput. Intell. Digit. Pedagog.* **2021**, *197*, 85–111.
37. Aberbach, H.; Jeghal, A.; Sabri, A.; Tairi, H. E-learning Recommendation Systems: A Literature Review. In *International Conference on Digital Technologies and Applications*; Springer: Cham, Switzerland, 2022; pp. 361–370.