EasyChair Preprint

№ 10012

Book Recommendation System Using Machine Learning

Anant Duhan and N Arunachalam

May 9, 2023

Book Recommendation System using Machine Learning

Anant Duhan
B.Tech, CSE - Core
SRMIST
Chennai, Tamil Nadu, India
ad8445@srmist.edu.in

Dr. N. Arunachalam
Dept. of Computing Technologies
(C Tech), SRMIST
Chennai, Tamil Nadu, India
arunachn@srmist.edu.in

Abstract - Users may utilize book recommendations to find and search for books on the internet. Given the enormous number of objects and descriptions for the user's needs, this recommendation system will assist the user in selecting the book that matches the description. Rating, Reviews, Description, and Author are all factors that influence recommendation systems.

The efficacy of Book Recommendation Systems heavily relies on the classifier used. Thus, it is crucial to develop a precise classifier to enhance the performance of recommendation systems. Among various supervised learning approaches and algorithms, Decision Tree Classifiers stand out due to their high accuracy, rapid classification speed, robust learning ability, and simple construction.

This paper proposes a Decision Tree-Based recommendation system framework. Other notable supervised learning approaches and algorithms include Naïve Bayes, Random Forest, Logistic Regression, and K-Nearest Neighbor.

Keywords—Recommender System, Machine Learning, Decision Tree Classifier, Logistic Regression, Book, etc.

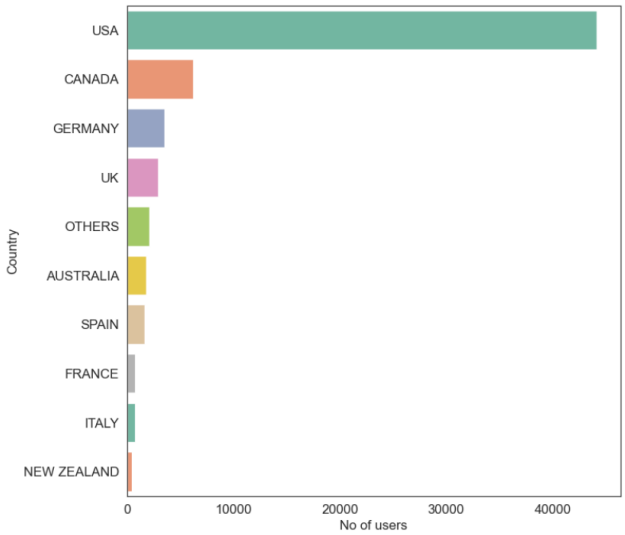
I. INTRODUCTION

When selling things to the internet market, most online websites now have their own recommendation system. Even though websites are intended to serve customers, the reality is that many are not designed with the customer in mind. Rather, organizations encourage buyers to make unnecessary and irrelevant purchases to increase sales. Now, a tailored recommendation system is required to assist individual users in finding relevant, engaging, and helpful things from a collection of products. With the launch of Jio, there has been a tremendous increase in internet usage; currently, customers have a wide range of items available on the e-commerce platform.

Collaboration filtering and content-based filtering are two types of recommendation systems. Items picked via collaborative filtering, also known as social filtering, depend on relationships between the current user and other users on the system. However, depending on user behaviors and preferences, content-based filtering is advised. Users' interests are first examined, then the profile analysis results are compared with goods accessible in the system to generate user suggestions.

In multi-model recommendation systems, multiple classifiers are commonly used, with two levels of learning. The first layer learns basic classifiers, which are then integrated into the second layer using ensemble methods such as XGBoost. The multi-model ensemble approach may also identify spatial patterns. Collaborative Filtering sorts of objects based on shared reactions, searching for a vast number of individuals to identify a smaller group of users with similar preferences for purchasing products or exhibiting specific behaviors.

Content-based, collaborative, hybrid and cross-domain filtering algorithms seem to be the four key strategies for constructing recommendation systems. Initially, collaborative filtering recommends things based on user comments and preferences. It has the unique capability of automatically predicting user preferences by gathering information from multiple users. For instance, collaborative filtering can predict whether a user would like or dislike a TV show based on their past behavior. Collaborative filtering typically requires many users, data sources, and agents, and can be applied in various domains, including e-commerce, weather forecasting, and online applications. Nevertheless, the main disadvantage of collaborative filtering is that it requires a substantial quantity of user data, which may be difficult to gather in certain apps that do not collect user information.



Content-based filtering, in contrast, recommends items based on their similarity to other objects. When valuable user information is not available, content-based filtering is often utilized. To quantify item similarity, both unsupervised and supervised machine learning approaches can be applied. The data must be organized, semi-structured, or unstructured to compute item similarity. The Hybrid recommendation system is the next strategy, which combines two or more filtering processes to obtain the desired outcome. Combining collaborative and content-based filtering can improve recommendations. Finally, cross-domain filtering algorithms can access data from

various domains, allowing for predictions to be made by studying the enhancing forecasts in the target domain and source domain.

II. LITERATURE SURVEY

Personal and business organizations utilize recommendation systems, also known as recommendation algorithms, to search for news and information, such as online purchasing, social dating, search optimization, and so on. Recommendation systems improve user experience and system efficiency. With the growing popularity of e-book reading habits and consumers' increasing expectations for discovering preferred books, book recommendation systems are becoming increasingly crucial when it comes to selecting books.

To construct recommendation systems, most researchers currently choose collaborative filtering. Collaborative filtering necessitates a massive quantity of real-time user data, which most recommendation systems may not have.

The most common approach for book recommendation systems is collaborative filtering, however, its accuracy was only 88%, which is poor. A content-based recommendation system, on the other hand, would need a vast quantity of training data, which may not be viable in real-world circumstances. The major drawback of collaborative recommendation systems is data sparsity, cold-start problem, long-tail distribution, and user privacy.

The cold-start problem is one of the primary challenges in the book recommendation system. It only occurs when a new user joins the system, and there we do not have any kind of information about his/her preferences and interests. Similarly, a new book that has no or limited information is also challenging to recommend. In such cases, the traditional recommendation algorithms are not effective and new approaches need to be developed to handle the cold start problem.

Data Sparsity is another significant challenge in book recommendation systems. The vast majority of users only rate or read a small number of books, resulting in sparse data that is insufficient to train accurate recommendation models. This challenge can be addressed by using techniques such as matrix factorization and feature selection.

Book recommendation systems also face a long-tail distribution problem, where a small number of popular books receive most of the attention and ratings, while most books have limited or no ratings. This challenge can be addressed by using a mixture of algorithms to balance the popularity and diversity of recommendations.

The issue of user privacy is increasingly important in book recommendation systems. User data is often used to train and improve recommendation algorithms, but users may be hesitant to share their data due to privacy concerns. Developing privacy-presenting recommendation algorithms is necessary to address this challenge.

The content-based filtering approach is utilized to recommend items based on their similarity. However, this strategy has a drawback in that it disregards existing user ratings when suggesting new products. User feedback is critical when recommending new books and journals. Unfortunately, the documents used in the content-based filtering approach do not include user ratings.

At the moment, the majority of existing systems use artificial intelligence to aid users in searching for goods based on characteristics such as popularity, correlation, and book content. The Influence Discrimination model, Linear Mix model, fixed effect model, NLP for sentiment analysis, k-nearest neighbor, and numerous neural network-based algorithms are all well-known approaches used in recommendation systems.

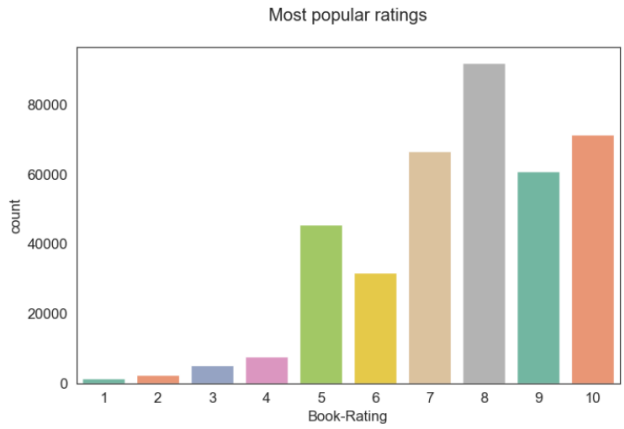
Online search behavior can have an unusual effect on recommendation systems. For instance, clicking on highly-ranked books may not have an impact, while clicking on lower-ranked books can have a positive impact. A major challenge for our recommendation system is infrequent data, which can be addressed by implementing an algorithm using a NN i.e., neural network. For academic journal readers, the k-nearest neighbor algorithm is highly effective in recommending scientific journals.

III. PROPOSED WORK

The "Book Recommendation System" initiative seeks to assist individuals in finding a book that interests them and encourages them to read more books. Decision Tree, Logistic Regression, Random Forest, K-Nearest Neighbor, and Naïve Bayes are used here. These methods are used to determine the degree of similarity between these novels. There are three main use cases for this system: suggestions for existing users, recommendations for new users, and ratings for newly submitted books. All of these are addressed in various ways. This project largely relies on collaborative filtering based on user input.

A. Datasets

We have a dataset with three tables: Book Ratings, Users, and Books. The first is Book Ratings, which has eight columns, followed by Users, which has three columns, and Users, which has three columns. The first step is the preliminary stage of pre-processing. It is a means of transforming raw data into a reasonable organization. As a result, we integrated the datasets before eliminating a row with a null value for specific characteristics, as well as a column with further missing values and removing duplicated values from the dataset. The dataset comprises six columns after pre-processing: ISBN, Book Author, Book Title, User-ID, Publisher, and Book Rating.



B. Algorithms

1) K-Nearest Neighbor: It is a machine learning estimator to locate groups of comparable user topics to basic book evaluations, and generates figures utilizing usual ratings to top KNN. In this recommendation system, comparable users are found by converting the table to a 2D matrix and then normalizing the network data frame values (ratings) into a scipy rectangular grid for more efficient computations. Unsupervised techniques with sklearn.neighbors are used to find the Closest Neighbors. Finally, this computation will determine the cosine similarity between rating vectors.

```
The top 10 Recommended books for Harry Potter and the Chamber of Secrets (Book 2) are:

Harry Potter and the Prisoner of Azkaban (Book 3)
Harry Potter and the Goblet of Fire (Book 4)
Harry Potter and the Sorcerer's Stone (Book 1)
Dr. Seuss's A B C (I Can Read It All by Myself Beginner Books)
The Second Generation
Lover Beware
J. K. Rowling: The Wizard Behind Harry Potter
A Dash of Death
So Much to Tell You
Dragonquest Achille Cover
```

Fig. K-Nearest Neighbor Recommendation

To construct a system that is item-based collaborative, the K-Nearest Neighbors Algorithm is employed. The ten most similar novels to the selected book are displayed, along with a measure of similarity. When measuring similarity in terms of distance, the larger the value, the higher the similarity. After that, they are sorted in ascending order. If a user enters their favorite book from the dataset, the algorithm will propose the top ten books that are most comparable to it.

We have applied the K-Nearest Neighbor approach in our model as:

(i) We import the KNeighborsClassifier library from the sklearn.neighbors class.
(ii) Next, we create the classifier object for the model.
(iii) Finally, we fit our data.

```
# K-Nearest Neighbors
from sklearn.neighbors import KNeighborsClassifier
model = KNeighborsClassifier(n_neighbors=3)
model.fit(X_train, Y_train)
```

2) Decision Tree: The Decision Tree Classifier is a supervised learning method that employs the greedy approach and employs a tree structure to represent features and class labels. Its primary objective is to develop a training model that can predict the classification or value of target variables by discovering decision rules from historical data (training data). The decision tree has two components: nodes and leaves, where leaves indicate the final outcomes. The tree's nodes represent test conditions for specific attributes, and each edge descending from a node represents one of the potential solutions to the test condition. This recursive process is repeated for every new sub-tree anchored at a node.

We employed the decision tree technique in our model as follows:

(i) We import the DecisionTreeClassifier library from the sklearn.tree class.
(ii) Next, we create the model_dtc object for the model.
(iii) Finally, we fit our data.

```
# Decision Tree
from sklearn.tree import DecisionTreeClassifier
model_dtc = DecisionTreeClassifier(
    criterion="entropy", random_state=2, max_depth=5)
model_dtc.fit(X_train, Y_train)
```

3) Naïve Bayes: To tackle binary and multi-class arrangement issues, the Nave Bayes order calculation is utilized. The Nave Bayes algorithm is quite simple when given binary or categorical input esteems. A Naive Bayes classifier operates on the assumption that the presence of one element in a class is independent of the presence of another element. The Naive Bayes classifier is based on Bayes theory and is beneficial when the data source has high dimensionality.

Naive Bayes has various uses, including real-time prediction, predicting the likelihood of several target attributes, and spam filtering. The likelihood of each attribute in the dataset, also known as class probability, must then be computed. The conditional probability of each information value for each class value.

We employed the Nave Bayes approach in our model as follows:

(i) We import the GaussianNB library from the sklearn.naive_bayes class.
(ii) Next, we create the model_nb object for the model.
(iii) Finally, we fit our data.

```
# Naïve Bayes
from sklearn.naive_bayes import GaussainNB
model_nb = GaussainNB()
model_nb.fit(X_train, Y_train)
```

4) Logistic Regression: Logistic Regression is a frequently used statistical technique that employs a logistic function to model a binary dependent variable. Although more complex variations are available, its basic form is commonly used. Regression analysis utilizes

logistic regression, which is a type of binomial regression, to estimate the parameters of a logistic model.

We utilized Logistic Regression in our model as follows:

(i) We import the LogisticRegression library from the sklearn.linear class.
(ii) Next, we create the model_lr object for the model.
(iii) Finally, we fit our data.

```
# Logistic Regression
from sklearn.linear_model import LogisticRegression
model_lr = LogisticRegression()
model_lr.fit(X_train, Y_train)
```

5) Random Forest: During the training phase, the Random Forest algorithm creates a vast number of decision trees as part of its machine learning approach. The output can be classified into different classes for classification or predict classes for regression. The accuracy of the forecast is proportional to the number of trees generated. There are three parameters that affect the performance of the random forest technique: n tree, which determines the number of trees to be produced, m try, which specifies the number of variables to be considered at each node split, and node size, which represents the minimum number of observations required in terminal nodes.

We utilized Random Forest in our model as follows:

(i) We import the RandomForestClassifier library from the sklearn.ensemble class.
(ii) Next, we create the model_rfc object for the model.
(iii) Finally, we fit our data.

```
# Random Forest
from sklearn.ensemble import RandomForestClassifier
model_rfc = RandomForestClassifier()
model_rfc.fit(X_train, Y_train)
```

6) XGBoost: XGBoost (eXtreme Gradient Boosting) is a versatile and improved gradient boosting technique that is optimized for viability, processing speed, and model performance. XGBoost has gained a reputation for surpassing other machine learning algorithms in terms of their performance. It is a free and open-source library developed by the Distributed Machine Learning Community. Utilizing parallel tree boosting (also referred to as GBDT or GBM), XGBoost can effectively and quickly tackle a range of data science problems with accuracy.

In our model, we applied XGBoost as follows:
(i) We import the xgboost library.
(ii) Next, we create the model_xgb object for the model.
(iii) Finally, we fit our data.

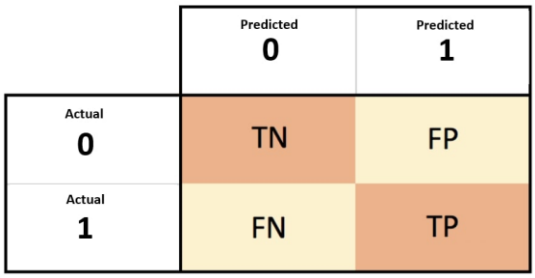
```
# XGBoost
import xgboost as xgb
model_xgb = xgb.XGBClassifier()
model_xgb.fit(X_train, Y_train)
```

7) Training Data: A dataset is utilized to train a machine learning model, comprising labeled examples that teach the model how to make predictions. The training data usually consists of input data i.e., features, and output data i.e., labels. The model learns to identify patterns in the input data and associate them with the correct output labels.

8) Testing Data: It is the dataset that is used to evaluate the performance of a machine learning model after it has been trained. Usually, testing data is distinct from the training data and serves to evaluate how effectively the model generalizes to new, unseen instances. The testing data also consists of input data and output data, but the output labels are not used during the evaluation process. Instead, the model's predictions are compared to the true labels to measure its accuracy.

IV. RESULTS DISCUSSION

At this step, we will analyze the performance of a model and compare the results. We will compare the results that we have got from various algorithms such as K-Nearest Neighbor, K-Nearest Neighbor with cosine similarity, Collaborative Filtering, Content-Based Filtering, and Gaussian Mixture. To do that we will use a confusion matrix, which consists of 4 parts: true positive, true negative, false positive, and false negative. With all these 4 parts we can easily calculate the value of the recall score, accuracy score, and precision score and from these 3 scores will be able to calculate the f1 score as well.



1) Binary Predictor: It typically employs standard techniques, including the construction of a ROC curve with its fundamental building blocks. In every classification problem, there are 2 classes: positive and negative. Each instance belongs to one of these 2 sets. A classifier instance can take on one of 4 possible types. When a positive case is correctly classified, it is referred to be a true positive. Yet, if it is wrongly categorized, it is considered a false negative. Similarly, when a negative event is correctly detected, it is termed a genuine negative, and when a negative incidence is erroneously classified, it is dubbed a false positive.

FPR = FP / (TN + FP)

FNR = FN / (TP + FN)

Precision (P) = TP / (TP + TN)

Specificity or TNR = TN / (FP + FTN)

Sensivity or Recall (R) = TP / (TP + FN)

F1 Score = 2 * (R * P) / (R + P)

To broaden the scope of our definition, we incorporate sensitivity and specificity as complementary measures: sensitivity also referred to as the true positive rate, is calculated as I - FPR, while specificity, also called the true negative rate, is computed as I – FNR.

We have compared metrics like f1 score, validation accuracy, training accuracy, RSME score, r2 score, and MAE score for algorithms such as Logistic Regression, XGB Classifier, Naïve Bayes, SVC, and Random Forest Classifier.

Terms	Naïve Bayes	Decision Tree	Logistic Regression	KNN	XGB
Training accuracy	.7814	.9909	.7184	.9427	.9909
Validation accuracy	.7812	.9905	.7189	.8885	.9905
F1 score	.9986	1.0	.9988	.9991	1.0
RSME score	1.2428	1.2428	1.2698	.9929	1.2428
R2 score	.8959	.8959	.8872	.9426	.8959
MAE score	.4653	.4643	.5847	.2392	.4643

V. CONCLUSION

This study looked at the implementation and design of a book recommendation system that used collaborative-filtering techniques. Based on prior evaluations and preferences, the system was able to provide customized suggestions for consumers. In terms of accuracy and efficiency, the collaborative filtering algorithm utilized in this study produced encouraging results.

The system was tested using real user data, and the results showed that it could provide relevant and meaningful suggestions. The assessment metrics used in this study demonstrated that the system outperformed the baseline technique, supporting the collaborative filtering methodology's effectiveness.

Overall, the book recommendation system developed in this study demonstrated the potential of collaborative filtering techniques for building personalized recommendation systems. Further research can be done to improve the system's accuracy and incorporate additional features such as social network analysis, text analysis, and hybrid algorithms.

VI. FUTURE WORK

1) Incorporating new data sources: In this study, we used historical ratings of users to generate recommendations. However, other data sources such as user reviews, book descriptions, and author information can be integrated to enhance the system's accuracy.

2) Enhancing the system's scalability: We focused on a small dataset in this study. However, as the system's user base grows, the scalability of the system becomes crucial. Future work can focus on developing algorithms that can handle large datasets efficiently.

3) Considering the temporal dimension: In this study, we did not consider the temporal dimension of the data. However, user preferences can change over time, and incorporating temporal information can improve the accuracy of the system.

4) Incorporating context awareness: Users' preferences and book choices can be influenced by their contexts, such as their location, time of day, and mood. Future work can focus on incorporating context awareness into the recommendation system to improve its effectiveness.

5) Evaluating the system's effectiveness in real-world settings: Finally, future work can focus on evaluating the system's effectiveness in real-world settings with many users and diverse preferences. This will help determine the system's utility and impact in practical scenarios.

VII. REFERENCES

[1] Avi Rana and K. Deeba et.al, “Online Book Recommendation System using Collaborative Filtering (With Jaccard Similarity)” in IOP ebooks 1362, 2019.

[2] G. Naveen Kishore, V. Dhiraj, SkHasaneAhammad, SivaramireddyGudise, Balaji Kummaraa and LikhitaRavuruAkkala, “Online Book

Recommendation System" International Journal of Scientific & Technology Research vol.8, issue 12, Dec 2019.

[3] Uko E Okon, B O Eke and P O Asaga, "An Improved Online Book Recommender System using Collaborative Filtering Algorithm", International Journal of Computer Applications vol.179-Number 46, 2018.

[4] Ms. Sushma Rjpurkar, Ms. Darshana Bhatt and Ms. Pooja Malhotra, "Book Recommendation System" International Journal for Innovative Research in Science & Technology vol.1, issue 11, April 2015.

[5] Abhay E. Patil, Simran Patil, Karanjit Singh, Parth Saraiya and AayushaSheregar, "Online Book Recommendation System using Association Rule Mining and Collaborative Filtering" International Journal of Computer Science and Mobile Computing vol.8, April 2019.

[6] Suhas Patil and Dr. Varsha Nandeo, "A Proposed Hybrid Book Recommender System" International Journal of Computer Applications vol.6 – No.6,Nov – Dec 2016.

[7] Ankit Khera, "Online Recommendation System" SJSU ScholarWorks, Masters Theorem and Graduate Research, Master's Projects, 2008.

[8] Anagha Vaidya and Dr. Subhash Shinde, "Hybrid Book Recommendation System" International Research Journal of Engineering and Technology (IRJET), July 2019.

[9] Dhirman Sarma, Tanni Mittra and Mohammad Shahadat Hossain, "Personalized Book Recommendation System using Machine Learning Algorithm" The Science and Information Organization vol.12,2019.

[10] Ayub, M., Ghazanfar, M.A., Maqsood, M. and Saleem, A. (2018). A Jaccard base similarity measure to improve performance of CF based recommendation system. Proceedings of International Conference on Information Networking (ICOIN).

[11] Choi, S.H., Jeong, Y.S. and Jeong, M.K. (2010). A Hybrid Recommendation Method with Reduced Data for Large-Scale Application. IEEE Transactions on Systems, Man and Cybernetics-Part C: Applications and Reviews, Vol. 40, No.5, September 2010

[12] Elgohary, A., Nomir, H., Sabek, I., Samir, M., Badawy, M. and Yousri, N.A. (2010). Wiki-rec: A semantic-based recommendation system using Wikipedia as ontology. In 10th International Conference on Intelligent Systems Design and Applications (ISDA).

[13] Oku, K., Nakajima, S., Miyazaki, J. and Uemura, S. (2006). Context-aware SVM for context- dependent information recommendation. In MDM'06 proceedings of International Conference on Mobile Data Management.

[14] Ghani, R. and Fano, A. (2002). Building recommender systems using a knowledge base of product semantics. In proceedings of 2nd International Conference on Adaptive Hypermedia and Adaptive Web Based Systems.

[15] Miyahara, K. and Pazzani, M.J. (2000) Collaborative filtering with the simple Bayesian classifier. In proceedings of Pacific Rim International Conference on Artificial Intelligence.

[16] Asanov D. (2011) Algorithms and Methods in Recommender Systems. Berlin Institute of Technology Berlin, Germany

[17] Lakshmi, S.S. and Lakshmi, T.A. (2014). Recommendation Systems: Issues and challenges. (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (4), 2014, 5771-5772

[18] Okon, E.U., Eke, B.O. and Asagba, P.O. (2018). An improved online book recommender system using collaborative filtering algorithm. International Journal of Computer Applications(0975- 8887) Volume 179-No.46, June 2018

[19] Kurmashov, N., Konstantin, L., Nussipbekov, A. (2015). Online book recommendation System. Proceedings of Twelve International Conference on Electronics Computer and Computation (ICECC)

[20] Mathew, P., Kuriakose, B. And Hegde, V. (2016). Book Recommendation System through content based and collaborative filtering method. Proceedings of International Conference on Data Mining and Advanced Computing (SAPIENCE)

[21] Parvitikar, S. and Dr. Joshi, B. (2015). Online book recommendation system by using collaborative filtering and association mining. Proceedings of IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)

[22] Ayub, M., Ghazanfar, M.A., Maqsood, M. and Saleem, A. (2018). A Jaccard base similarity measure to improve performance of CF based recommendation system. Proceedings of International Conference on Information Networking (ICOIN)

[23] Gogna, A., Majumdar, A. (2015). A Comprehensive Recommender System Model: Improving Accuracy for Both Warm and Cold Start Users. IEEE Access Vol. 3, 2803- 2813, 2015

[24] Chatti, M.A., Dakova, S., Thus, H. and Schroeder, U. (2013). Tag-Based Collaborative Filtering Recommendation in Personal Learning Environments. IEEE Transactions on Learning Technologies, Vol. 6, No. 4, October-December 2013